

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ  
ЯРОСЛАВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. П. Г. ДЕМИДОВА

**С. И. Яблокова**

**ВВЕДЕНИЕ  
В БЫСТРЫЕ АЛГОРИТМЫ  
ЦИФРОВОЙ ОБРАБОТКИ СИГНАЛОВ**

*Учебное пособие*

*Рекомендовано  
Научно-методическим советом университета  
для студентов, обучающихся по специальностям  
Математика и Компьютерная безопасность*

ЯРОСЛАВЛЬ      2009

УДК 519.725  
ББК 3811.3я73  
Я 14

*Рекомендовано  
Редакционно-издательским советом университета  
в качестве учебного издания. План 2009/10 года*

**Рецензенты:**

кафедра алгебры ЯГПУ им. К. Д. Ушинского;  
кандидат физ.-мат. наук, доцент ЯГТУ Е. Р. Матвеев

**Я 14** **Яблокова, С. И.** Введение в быстрые алгоритмы цифровой обработки сигналов: учебное пособие / С. И. Яблокова; Яросл. гос. ун-т. им. П. Г. Демидова. – Ярославль: ЯрГУ, 2009. – 136 с.  
ISBN 978-5-8397-0685-9

Учебное пособие составлено в соответствии с программой курса "Прикладная алгебра (Быстрые алгоритмы)". Рассматриваются основные преобразования, используемые в цифровой обработке сигналов, и быстрые алгоритмы их вычисления. Пособие содержит большое количество примеров построения быстрых алгоритмов вычисления линейных и циклических сверток и дискретного преобразования Фурье.

Издание предназначено для студентов четвертого курса, обучающихся по специальностям 010100.62, 010101.65 Математика, и студентов второго курса, обучающихся по специальности 090102.65 Компьютерная безопасность (дисциплины "Прикладная алгебра", "Алгебраическая алгоритмика" (блоки СД, ОПД)), очной формы обучения.

Библиогр. : 4 назв.

УДК 519.725  
ББК 3811.3я73

ISBN 978-5-8397-0685-9

© Ярославский государственный  
университет им. П. Г. Демидова, 2009

Учебное издание  
**Яблокова Светлана Ивановна**  
**Введение**  
**в быстрые алгоритмы**  
**цифровой обработки сигналов**  
*Учебное пособие*

Редактор, корректор М. В. Никулина  
Компьютерный набор, верстка С. И. Яблокова

Подписано в печать 27.10.2009 г. Формат 60 × 84 1/8. Бумага тип.  
Усл. печ. л. 15,81. Уч.-изд. л. 6,0. Тираж 100 экз. Заказ 589

Оригинал-макет подготовлен в редакционно-издательском отделе  
Ярославского государственного университета им. П. Г. Демидова

Отпечатано ООО "Ремдер" ЛР ИД N 06151 от 26.10.01.

г. Ярославль, пр. Октября, 94, оф. 37

тел. (4852) 73 - 35 - 03, 58 - 03 - 48,

факс 58 - 03 - 49



## ПРЕДИСЛОВИЕ

Учебное пособие содержит лекции по курсу "Прикладная алгебра (Быстрые алгоритмы)", изучаемому студентами специальности "Математика" в 8 семестре. Часть этого материала излагается также студентам специальности "Компьютерная безопасность" в 3 семестре в рамках курса "Алгебраическая алгоритмика".

Пособие включает в себя материал, связанный с основными преобразованиями, используемыми в цифровой обработке сигналов: линейными и циклическими свертками и дискретным преобразованием Фурье. Связь между циклической сверткой и дискретным преобразованием Фурье дается теоремой о свертке (§ 8). Быстрое вычисление сверток представлено алгоритмом Кука – Тоома и алгоритмом Винограда для малых длин. Кроме того, в § 24 рассмотрен метод Агарвала – Кули вычисления свертки, сводящий одномерную циклическую свертку непростой длины в двумерную. Методы быстрого преобразования Фурье представлены алгоритмами Кули – Тьюки, в том числе для длин, являющихся степенями двойки и четверки, алгоритмом Гуда – Томаса, а также алгоритмом Рейдера, сводящим дискретное преобразование Фурье к циклической свертке. На основе алгоритмов Рейдера и Винограда для сверток получается БПФ-алгоритм Винограда.

В параграфах 21 – 23 рассматриваются быстрые алгоритмы свертки и дискретного преобразования Фурье в конечных полях, в частности, числовые преобразования Фурье и Мерсенна.

В заключение (§ 25) доказываются теоремы, дающие нижнюю оценку сложности алгоритмов вычисления свертки.



## ИСТОРИЯ БЫСТРЫХ АЛГОРИТМОВ И ИХ ПРИЛОЖЕНИЯ

Цифровая обработка сигналов сейчас переживает бурное развитие. Ее активно используют в различных областях человеческой деятельности. Сейсмография, радиолокация, связь, радиоастрономия, медицинская электроника используют последние достижения в области цифровой обработки сигналов. Разрабатываются и активно используются цифровые процессоры – специализированные цифровые компьютеры для обработки сигналов.

С точки зрения математики развитие быстрых алгоритмов цифровой обработки сигналов также является важным шагом в дальнейшем развитии такой, к примеру, ее области, как теория чисел. Во все времена большие числа, и особенно большие простые числа, интересовали людей – и математиков, и нематематиков. Математики искали новые и новые простые числа, которые становились очень большими. Нахождение таких чисел являлось все более трудной задачей. Даже появление быстродействующих вычислительных машин помогло в решении этой задачи лишь отчасти. Действительно, например, простейший тест простоты при помощи последовательных делений числа с 200 десятичными цифрами требует (при использовании машины, работающей со скоростью  $10^6$  операций в секунду) примерно  $10^{86}$  лет вычислений! Можно, конечно, постараться повысить быстродействие компьютера от  $10^6$  операций в секунду, например, до  $5 \cdot 10^6$  операций в секунду, но гораздо разумнее так организовать вычисления, чтобы имеющегося быстродействия компьютера оказалось достаточно. Именно разработкой таких алгоритмов вычисления и занимается область математики, которую называют быстрыми алгоритмами цифровой обработки сигналов.

Историю быстрых алгоритмов обработки сигналов принято отсчитывать с того момента, когда в 1965 году Кули и Тьюки опубликовали быстрый алгоритм вычисления дискретного преобразования Фурье (БПФ-алгоритм). На самом деле эта история началась раньше, когда в 1947 году Левинсон опубликовал свой эффективный метод решения некоторых теплицевых систем уравнений. С тех пор этот алгоритм широко используется при обработке сейсмических данных. Долгое время литература, посвященная алгоритму Левинсона, не пересекалась с литературой по быстрым алгоритмам преобразования Фурье. Первый БПФ-алгоритм был разработан Гудом (1960 г.) и Томасом (1963 г.). Публикация Гуда – Томаса прошла незамеченной. Алгоритм же Кули – Тьюки появился, как говорят, в нужный момент и послужил катализатором применения метода цифровой обработки сигналов в новом контексте. Дело в том, что после опубликования этого алгоритма Стогхэм заметил, что БПФ-алгоритмы могут служить удобным способом вычисления сверток. В силу множества возможных приложений алгоритм Кули – Тьюки и получил широкую известность. Позже Виноград (1976, 1978 гг.) опубликовал свой более эффективный, хотя и более сложный БПФ-алгоритм, который позволяет глубже понять, что в действительности означает процесс вычисления дискретного преобразования Фурье.

Различные вариации БПФ-алгоритма Кули – Тьюки появились позднее в работах других математиков. Сейчас существуют хорошие БПФ-алгоритмы практически для произвольной длины блока данных, а не только для блока длины, равной степени двойки, как в исходном алгоритме Кули – Тьюки. Та же идея используется, к примеру, для построения многолучевой плоско-фазированной радиолокационной антенны и известна под названием матрицы Батлера.

Для малых длин блоков быстрые алгоритмы сверток были впервые построены Агарвалом и Кули (1977 г.) с использованием остроумных догадок, но без использования общего универсального метода. Общий же метод построения быстрых алгоритмов свертки описал Виноград в 1978 году, доказав при этом важные теоремы несуществования лучших алгоритмов свертки для полей вещественных и комплексных чисел. Агарвал и Кули также указали основанный на китайской теореме об остатках метод разбиения задачи вычисления длинных сверток на задачи вычисления коротких сверток. Этот метод, объединяемый с методом Винограда, дает очень хорошие результаты для вычисления коротких сверток.

Разработка эффективных быстрых алгоритмов стала возможной также благодаря работам таких математиков, как Поллард, Штрассен, Рейдер и многих других.



В настоящее время для проведения вычислений широко используются сверхбольшие интегральные схемы, называемые чипами. Чип может содержать порядка 100000 логических элементов. Иногда выбор алгоритма позволяет существенно улучшить характеристики чипа. Используя быстрый алгоритм, можно реализовать улучшение характеристик чипа, для чего конструктор чипа должен перенести архитектуру алгоритма в архитектуру чипа. Разработка оптимальных конструкций в эпоху больших интегральных схем невозможна без понимания быстрых алгоритмов, лежащих в основе этой архитектуры. Большие цифровые процессоры часто сами создают потребность в разработке быстрых алгоритмов, поскольку размерность решаемых на них задач растет. Будет ли процессорное время алгоритма, предназначенного для решения некоторой задачи, пропорционально  $n^2$  или  $n^3$ , несущественно при малом  $n$ , но при  $n$ , равном  $10^3$ , это становится критичным.

Быстрые алгоритмы, рассматриваемые в данном пособии, связаны с цифровой обработкой сигналов, и их приложения столь же широки, сколь и приложения самой цифровой обработки сигналов. Мы сейчас даже не можем угадать масштабы будущих приложений цифровой обработки сигналов, но несложно предвидеть приложения, в которых объем необходимых вычислений будет на несколько порядков больше, чем тот, обработку которого может обеспечить современная технология вычислений.

Системы звуковой локации в наше время стали почти полностью цифровыми. Эти системы выполняют десятки миллионов или сотни миллионов умножений в секунду и еще больше сложений. Они уже сейчас требуют мощного цифрового оборудования.

Радиолокационные системы тоже становятся цифровыми. В принципе они очень похожи на системы звуковой локации, отличаясь тем, что используемая полоса частот их работы в 1000 и более раз больше, чем у звуковой локации.

Цифровая обработка сейсмической информации является главным методом разведки земных недр, в частности, одним из важнейших методов поиска залежей нефти.

Компьютерная томография широко используется в медицине. Это способ объемного синтеза изображений внутренних органов человека с помощью множественных проекций, получаемых при просвечивании рентгеновскими лучами. Разрабатываются алгоритмы, позволяющие существенно снизить дозы облучения, но требования к цифровой обработке намного превосходят те, что возможно реализовать в настоящее время.

Неразрушающий контроль качества продукции возможен с помощью воссоздания на компьютере изображений внутренних областей изделия по результатам эхолокации.

Обработка сигналов может быть использована при улучшении качества плохих фотографий, смазанных движением камеры или расфокусировкой. Такая цифровая обработка требует очень большого объема вычислений.

Обработка на цифровом процессоре спутниковых фотографий позволяет совместить несколько изображений, или выделить особенности, или скомбинировать полученную на различных длинах волн информацию, или создать синтетический стереоскопический образ. Так, в метеорологических исследованиях можно создать подвижное трехмерное изображение облачного покрова, движущегося над поверхностью земли, используя для этого последовательность спутниковых фотографий, снятых с нескольких точек.

Цифровые телефоны, цифровые видеокамеры, цифровое телевидение прочно вошли в современную жизнь. Потребность использования быстрых алгоритмов для цифровой обработки сигналов продолжает расти.

Каждое из описанных выше приложений связано с большим количеством вычислений, структура которых довольно прозрачна и сильно упорядочена. Это и позволяет создавать эффективные вычислительные алгоритмы для их решения.

Для оценки эффективности алгоритма обычно используют число необходимых умножений и сложений. Эти вычислительные характеристики и являются для нас главными критериями, описывающими сложность вычислительного устройства. На более низком уровне эта сложность описывается площадью чипа или числом логических элементов на нем и временем, необходимым для проведения вычислений. Мы не будем здесь пытаться оценивать характеристики на этом уровне, так как это выходит за рамки разработчика алгоритмов и относится к техническим вопросам конструкции чипа.



## §1. Понятие быстрого алгоритма. Цифровые фильтры

Любой алгоритм можно описывать либо через соотношение между входом и выходом, либо детально объясняя его внутреннюю структуру. Применяя к некоторой ветви цифровой обработки сигналов, мы сталкиваемся с заданием алгоритмов через вход - выход. При заданном сигнале или записанных некоторым образом данных основное внимание обращается на то, что надо с этими данными сделать, т. е. каков должен быть выход алгоритма. Обычно предполагают, что алгоритм типа вход - выход задан (в терминах фильтров, преобразований Фурье, интерполяций, прореживаний, корреляций, матричных операций и т. д.). Быстрые алгоритмы могут быть записаны математической формулой, и, следовательно, вычислений в соответствии с этой записью. Такую реализацию алгоритма вычислений называют прямой. Но прямая реализация не всегда является самым эффективным путем решения задачи.

Под *быстрым алгоритмом* будем понимать детальное описание вычислительной процедуры, которая не является очевидным способом вычисления выхода по данному входу, но позволяет провести вычисления более эффективно.

Как правило, быстрый алгоритм жертвует концептуальной ясностью вычислений в пользу их эффективности. В нашем случае эту эффективность мы оцениваем количеством необходимых операций умножения и сложения.

Например, допустим требуется вычислить число  $N$ :

$$N = ad + bd + cd + ae + be + ce + af + bf + cf.$$

В таком виде вычисление  $N$  содержит 9 умножений и 8 сложений. Нетрудно заметить, что

$$N = (a + b + c)(d + e + f)$$

представляет собой эквивалентную форму, требующую лишь одного умножения и четырех сложений, поэтому с точки зрения эффективности вычислений она предпочтительнее.

Простой, но нетривиальный пример быстрого алгоритма, который нам пригодится в дальнейшем, — умножение двух комплексных чисел, использующее всего три (вместо четырех) вещественных умножения. Произведение комплексных чисел

$$(a + ib)(c + id) = e + if$$

можно записать через умножение и сложение вещественных чисел:

$$\begin{aligned} e &= ac - bd, \\ f &= ad + bc. \end{aligned} \tag{1}$$

Эти формулы содержат четыре вещественных умножения и два вещественных сложения. Умножение — это более трудоемкая операция по сравнению со сложением, поэтому более эффективный алгоритм дается равенствами

$$\begin{aligned} e &= (a - b)d + a(c + d), \\ f &= (a + b)d + b(c + d). \end{aligned} \tag{2}$$

В таком виде алгоритм содержит три вещественных умножения и пять вещественных сложений. Если проводится серия умножений комплексных чисел, в которой число  $c + id$  повторяется, то числа  $c - d$  и  $c + d$  можно вычислить один раз и запомнить. Тогда для вычисления одного произведения комплексных чисел потребуется три умножения и три сложения. Можно сказать, что формула (2) задает быстрый алгоритм умножения комплексных чисел, а формула (1) — прямой алгоритм. Оба эти алгоритма мы сейчас запишем в матрично-векторной форме.

Пусть вектор  $(a \ b)^T$  представляет собой комплексное число  $a + ib$ ,  $(e \ f)^T$  — комплексное число  $e + if$ , а матрица

$$\begin{pmatrix} c & -d \\ d & c \end{pmatrix}$$



– комплексное число  $c + id$ . Тогда формулу (1) можно записать в виде

$$\begin{pmatrix} e \\ f \end{pmatrix} = \begin{pmatrix} c & -d \\ d & c \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}.$$

Алгоритм, задаваемый формулами (2), можно записать в виде

$$\begin{pmatrix} e \\ f \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} c-d & 0 & 0 \\ 0 & c+d & 0 \\ 0 & 0 & d \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix},$$

или в сокращенной записи

$$(e \ f)^T = B D A (a \ b)^T,$$

где  $A$  –  $3 \times 2$  матрица *предсложений*,  $D$  – диагональная  $3 \times 3$  матрица включает в себя все произведения алгоритма,  $B$  –  $2 \times 3$  матрица *постсложений*. Таким образом, быстрый алгоритм умножения двух комплексных чисел состоит из серии сложений, задаваемых матрицей  $A$ , затем – серии умножений, задаваемых матрицей  $D$ , и, наконец, еще из одной серии сложений, задаваемых матрицей  $B$ .

Многие из лучших процедур вычисления свертки и дискретного преобразования Фурье могут быть приведены к такому же матричному виду, в котором центральная матрица является диагональной, а стоящие по бокам матрицы содержат только элементы 0 и  $\pm 1$ .

В качестве еще одного примера рассмотрим быстрый алгоритм умножения матриц. Пусть  $C = AB$ , где  $A$  – матрица размера  $l \times n$ , а  $B$  – матрица размера  $n \times m$ . Тогда

$$c_{ij} = \sum_{k=1}^n a_{ik} b_{kj} \quad (i = 1, \dots, l; j = 1, \dots, m).$$

Для нахождения всех элементов матрицы  $C$  по данной формуле потребуется  $nlm$  умножений и  $(n-1)lm$  сложений.

Построим алгоритм, который почти в два раза уменьшает число умножений, правда при этом увеличивается число сложений. В основе нового алгоритма лежит очевидное тождество

$$a_1 b_1 + a_2 b_2 = (a_1 + b_2)(a_2 + b_1) - a_1 a_2 - b_1 b_2. \quad (3)$$

Считаем, что  $n$  – четно (в противном случае можно дополнить матрицу  $A$  нулевым столбцом, а матрицу  $B$  – нулевой строкой). Применяя тождество (3) к парам столбцов матрицы  $A$  и к парам строк матрицы  $B$ , получаем

$$c_{ij} = \sum_{k=1}^{\frac{n}{2}} (a_{i2k-1} + b_{2kj}) (a_{i2k} + b_{2k-1j}) - \sum_{k=1}^{\frac{n}{2}} a_{i2k-1} a_{i2k} - \sum_{k=1}^{\frac{n}{2}} b_{2k-1j} b_{2kj} \quad (i = 1, \dots, l; j = 1, \dots, m). \quad (4)$$

Поскольку вторая сумма в (4) зависит только от  $i$ , то ее не надо вычислять заново для каждого  $j$ ; третья сумма в (4) зависит только от  $j$ , поэтому ее не надо заново вычислять для каждого  $i$ . Значит, полное число умножений для вычисления элементов матрицы  $C$  по формуле (4) равно

$$\frac{n}{2} lm + \frac{n}{2} l + \frac{n}{2} m = \frac{1}{2} nlm + \frac{1}{2} n(l+m),$$

а полное число сложений равно

$$\left(n + \frac{n}{2} - 1\right) lm + 2lm + \left(\frac{n}{2} - 1\right) l + \left(\frac{n}{2} - 1\right) m = \frac{3}{2} nlm + lm + \left(\frac{n}{2} - 1\right) (l+m).$$

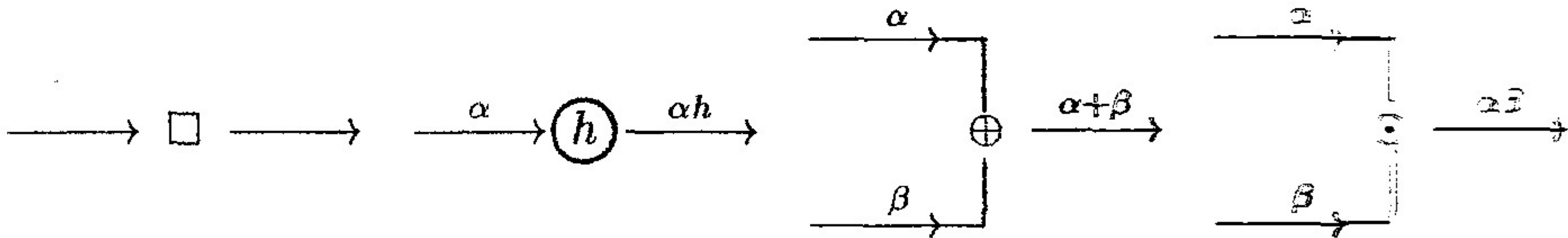
Число умножений уменьшилось почти в два раза. Однако полученный алгоритм более чувствителен к погрешностям округления, если об этом специально не позаботиться, вводя масштабные множители. Вопрос о точности вычислений возникает почти всегда при использовании быстрых



алгоритмов. Иногда при уменьшении числа операций уменьшается и количество вычислений поскольку уменьшается число источников помех вычисления. В других случаях может быть столь чувствителен к одному из таких источников, что с уменьшением их числа в помех погрешность вычислений увеличивается.

Наиболее важной задачей цифровой обработки сигналов является анализ дискретных длинной последовательности чисел, а наиболее важным устройством – цифровой фильтр.

Цифровой фильтр – это устройство, которое по заданной числовой последовательности (называемой входной) формирует новую числовую последовательность (называемую выходной). Основные элементы, из которых строятся фильтры: разряд регистра сдвига, сумматор, умножитель на скаляр, умножитель. Схематично эти элементы можно изображать следующими образом:

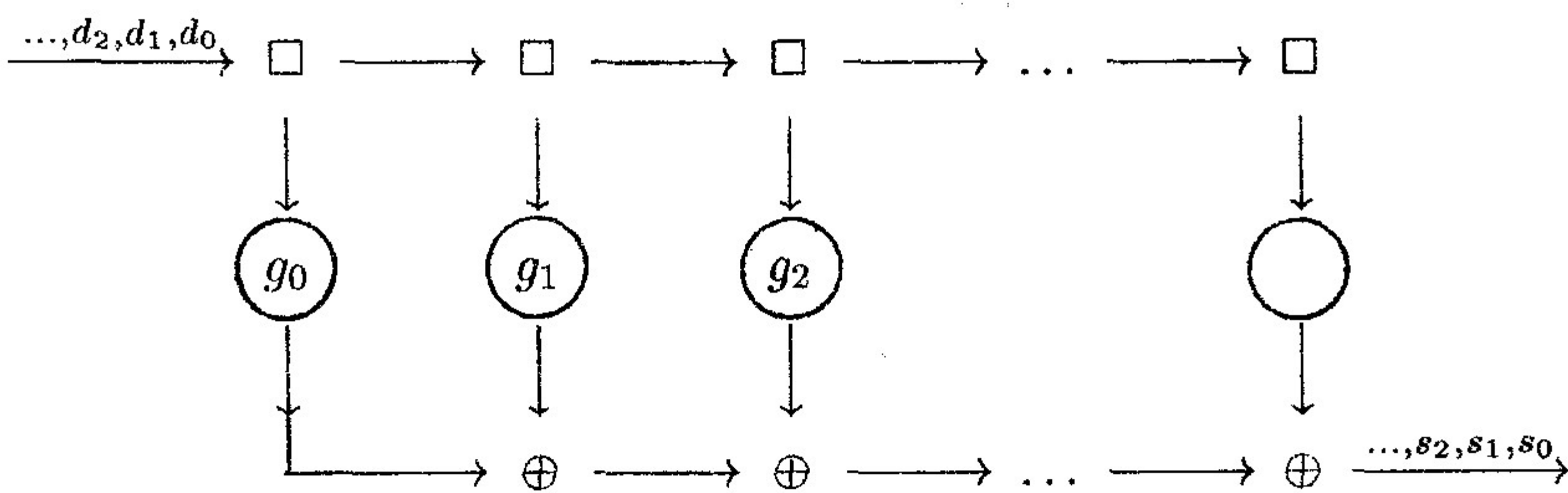


Разряд регистра сдвига    Умножитель на скаляр    Сумматор    Умножитель

Разряд регистра сдвига содержит одно число, которое поступает на его выход. В дискретные моменты времени (называемые тактами) разряд регистра сдвига замещает содержащееся в нем число другим, поступающим на его вход, отбрасывая предыдущее содержимое. Регистр сдвига представляет собой несколько соединенных в цепь разрядов регистра сдвига. Регистр сдвига называется также линией задержки.

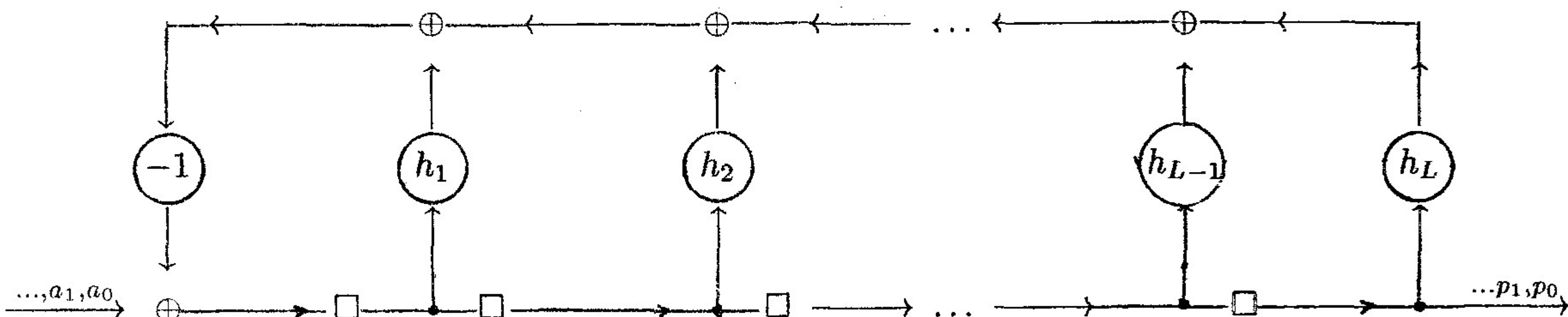
Два наиболее важных типа регистров сдвига известны под названием фильтра с конечным импульсным откликом (КИО-фильтр) и авторегрессионного фильтра.

КИО-фильтр представляет собой линию задержки с отводами, в которых выход каждого разряда умножается на фиксированную константу, а результаты складываются вместе:



Как мы увидим в дальнейшем, выходная последовательность КИО-фильтра равна линейной свертке входной последовательности  $\{d_0, d_1, d_2, \dots\}$  и последовательности, описываемой весовыми множителями в отводах фильтра  $\{g_0, g_1, g_2, \dots\}$ .

Авторегрессионный фильтр также является линией задержки с отводами, но с обратной связью, соединяющей выход с входом:





## §2. ЛИНЕЙНАЯ И ЦИКЛИЧЕСКАЯ СВЕРТКИ

Линейная свертка является едва ли не самой общей вычислительной задачей цифровой обработки сигналов, и нас будет интересовать ее эффективная реализация. Кроме того, рассмотрим также методы эффективного вычисления циклической свертки, которая может быть использована для вычисления очень длинных линейных сверток.

**Определение 1.** Для заданных двух последовательностей – последовательности данных

$$\mathbf{d} = \{d_i | i = 0, 1, \dots, N-1\}$$

длины  $N$  и последовательности фильтра

$$\mathbf{g} = \{g_i | i = 0, 1, \dots, L-1\}$$

длины  $L$  *линейной сверткой* называется новая последовательность

$$\mathbf{s} = \{s_i | i = 0, 1, \dots, N+L-2\}$$

длины  $N+L-1$ , элементы которой определяются равенствами

$$s_i = \sum_{k=0}^{N-1} g_{i-k} d_k, \quad (i = 0, 1, \dots, N+L-2), \quad (1)$$

где  $g_{i-k} = 0$  при  $i-k < 0$  и  $i-k \geq L$ .

Последовательность  $\mathbf{s}$  называется *выходной*, или *сигнальной*.

Прямой метод вычисления свертки содержит  $NL$  умножений, так как все компоненты последовательности  $\mathbf{g}$  умножаются на все компоненты последовательности  $\mathbf{d}$ .

**Определение 2.** Для заданных двух последовательностей

$$\mathbf{d} = \{g_i | i = 0, 1, \dots, N-1\},$$

$$\mathbf{g} = \{g_i | i = 0, 1, \dots, L-1\}$$

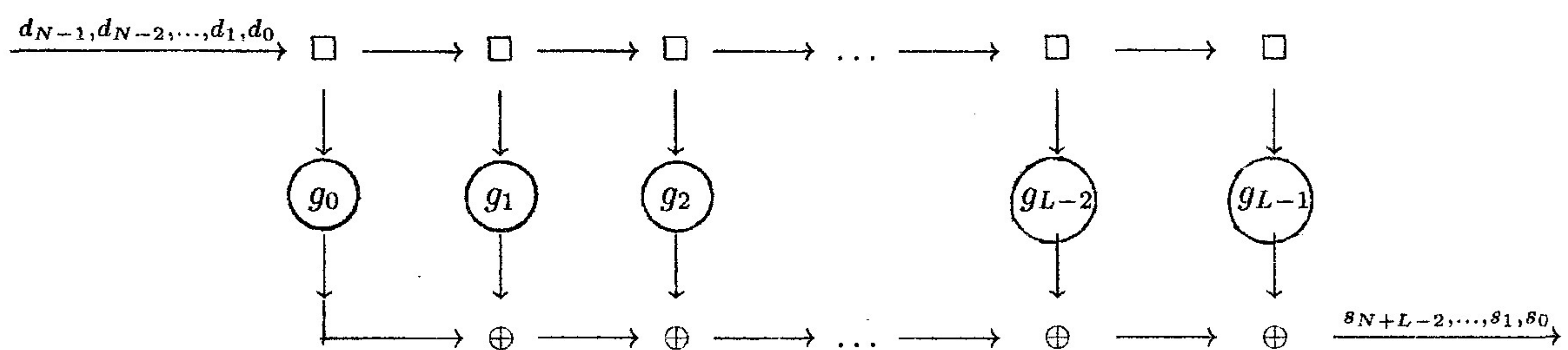
их *корреляцией* называется новая последовательность  $\mathbf{r}$  (длины  $N+L-1$ ), элементы которой определяются равенствами

$$r_i = \sum_{k=0}^{N-1} g_{i+k} d_k, \quad (i = 0, 1, \dots, N+L-2), \quad (2)$$

где  $g_{i+k} = 0$  при  $i+k \geq L$ .

Корреляцию можно вычислить как свертку, если просто прочесть одну из определяющих ее последовательностей в обратном порядке.

Рассмотрим КИО-фильтр с отводами, в которых находятся элементы последовательности  $\mathbf{g}$ , а на вход поступают элементы последовательности  $\mathbf{d}$ :



Очевидно, на выходе этого фильтра мы получим последовательность  $N \geq L$

$$\begin{aligned}
s_0 &= d_0 g_0, \\
s_1 &= d_0 g_1 + d_1 g_0, \\
s_2 &= d_0 g_2 + d_1 g_1 + d_2 g_0, \\
&\dots\dots\dots \\
s_{L-1} &= d_0 g_{L-1} + d_1 g_{L-2} + \dots + d_{L-1} g_0, \\
s_L &= d_1 g_{L-1} + d_2 g_{L-2} + \dots + d_L g_0, \\
&\dots\dots\dots \\
s_{N-1} &= d_{N-L} g_{L-1} + d_{N-L+1} g_{L-2} + \dots + d_{N-1} g_0, \\
s_N &= d_{N-L+1} g_{L-1} + d_{N-L+2} g_{L-2} + \dots + d_N g_0, \\
&\dots\dots\dots \\
s_{N+L-3} &= d_{N-2} g_{L-1} + d_{N-1} g_{L-2}, \\
s_{N+L-2} &= d_{N-1} g_{L-1}.
\end{aligned}$$

Это и есть элементы линейной свертки последовательностей  $\mathbf{d}$  и  $\mathbf{g}$ , вычисляемые по формуле (1). Значит, КИО-фильтр вычисляет линейную свертку двух данных последовательностей.

Линейную свертку можно интерпретировать в терминах умножения многочленов. Пусть даны многочлены

$$d(x) = \sum_{i=0}^{N-1} d_i x^i \quad \text{и} \quad g(x) = \sum_{i=0}^{L-1} g_i x^i$$

степеней  $N-1$  и  $L-1$  соответственно.

Рассмотрим их произведение

$$s(x) = d(x)g(x),$$

где

$$s(x) = \sum_{i=0}^{N+L-2} s_i x^i$$

— многочлен степени  $N+L-2$ .

Так как

$$\sum_{i=0}^{N+L-2} s_i x^i = \left( \sum_{k=0}^{N-1} d_k x^k \right) \left( \sum_{j=0}^{L-1} g_j x^j \right) = \sum_{i=0}^{N+L-2} \left( \sum_{\substack{k+j=i \\ 0 \leq k \leq N-1 \\ 0 \leq j \leq L-1}} d_k g_j \right) x^i,$$

то

$$s_i = \sum_{\substack{k+j=i \\ 0 \leq k \leq N-1 \\ 0 \leq j \leq L-1}} d_k g_j = \sum_{\substack{k=0 \\ 0 \leq i-k \leq L-1}}^{N-1} d_k g_{i-k}, \quad (i = 0, 1, \dots, N+L-2),$$

значит, мы получили формулы (1). Таким образом, элементы линейной свертки равны коэффициентам многочлена  $s(x)$ , равного произведению многочленов  $d(x)$  и  $g(x)$ , коэффициенты которых есть соответствующие элементы последовательностей  $\mathbf{d}$  и  $\mathbf{g}$ . Но умножение многочленов коммутативно, т. е.

$$s(x) = d(x)g(x) = g(x)d(x).$$

Это значит, что роли последовательностей  $\mathbf{d}$  и  $\mathbf{g}$  можно поменять местами, а, следовательно, элементы линейной свертки можно также вычислять по формулам

$$s_i = \sum_{j=0}^{L-1} g_j d_{i-j}, \quad (i = 0, 1, \dots, N+L-2),$$

где  $d_{i-j} = 0$ , если  $i-j < 0$  или  $i-j \geq N$ .

В дальнейшем мы будем интерпретировать линейную свертку двух последовательностей через умножение многочленов.



**Определение 3.** Для заданных двух числовых последовательностей  $\mathbf{d} = \{d_0, d_1, \dots, d_{n-1}\}$  и  $\mathbf{g} = \{g_0, g_1, \dots, g_{n-1}\}$  одной и той же длины  $n$  их *циклической сверткой* называется новая числовая последовательность  $\tilde{\mathbf{s}}$  длины  $n$ , элементы которой определяются равенствами

$$\tilde{s}_i = \sum_{k=0}^{n-1} d_k g_{i-k \pmod{n}}, \quad (i = 0, 1, \dots, n-1). \quad (3)$$

Для упрощения записи индексов введем обозначение

$$((i-k)) \equiv i-k \pmod{n},$$

тогда формула (3) перепишется в виде

$$\tilde{s}_i = \sum_{k=0}^{n-1} d_k g_{((i-k))}, \quad (i = 0, 1, \dots, n-1). \quad (4)$$

Поскольку  $0 \leq ((i-k)) < n$ , то в циклической свертке при каждом значении  $i$  каждое  $d_k$  умножается на некоторую величину  $g_{((i-k))}$ , т. е. каждая из сумм (3) содержит  $n$  слагаемых. Это существенно отличается от линейной свертки, в которой  $d_k$  часто умножается на член  $g_{i-k}$ , индекс которого выходит за диапазон определения последовательности  $\mathbf{g}$ , так что  $g_{i-k} = 0$ .

Найдем связь между циклической и линейной свертками. Разделим сумму

$$\tilde{s}_i = \sum_{k=0}^{n-1} d_k g_{((i-k))}$$

на две, выделяя в первую сумму члены, индексы которых удовлетворяют условию  $i-k \geq 0$  (или  $k \leq i$ ), а во вторую – члены, индексы которых удовлетворяют условию  $i-k < 0$  (или  $k > i$ ):

$$\tilde{s}_i = \sum_{k=0}^i d_k g_{i-k} + \sum_{k=i+1}^{n-1} d_k g_{i-k+n}.$$

Положим теперь в первой сумме  $g_{i-k} = 0$  при  $k > i$ , тогда

$$\sum_{k=0}^i d_k g_{i-k} = \sum_{k=0}^{n-1} d_k g_{i-k} = s_i,$$

положим во второй сумме  $g_{n+i-k} = 0$  при  $k \leq i$ , тогда

$$\sum_{k=i+1}^{n-1} d_k g_{n+i-k} = \sum_{k=0}^{n-1} d_k g_{n+i-k} = s_{n+i},$$

и, следовательно,

$$\tilde{s}_i = \sum_{k=0}^{n-1} d_k g_{i-k} + \sum_{k=0}^{n-1} d_k g_{n+i-k} = s_i + s_{n+i} \quad (i = 0, 1, \dots, n-2).$$

Очевидно, что при  $i$ , равном  $n-1$ , разбивать сумму (4) на две не надо, поэтому

$$\tilde{s}_{n-1} = s_{n-1}.$$

Итак, связь между линейной и циклической свертками дается равенствами

$$\begin{aligned} \tilde{s}_i &= s_i + s_{n+i}, & i &= 0, 1, \dots, n-2, \\ \tilde{s}_{n-1} &= s_{n-1}. \end{aligned} \quad (5)$$



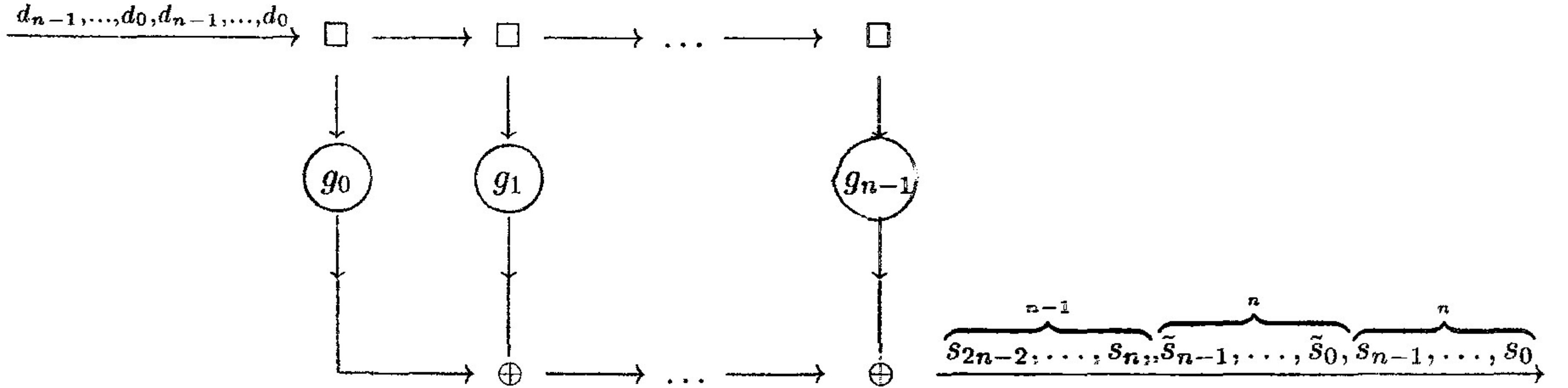
Говорят, что члены последовательности  $\mathbf{s}$ , индексы которых больше  $n-1$ , "вкладываются" обратно в члены, индексы которых меньше  $n$ .

Если произведения  $g_{n+i-k}d_k$  равны нулю для всех  $i$  и  $k$ , то линейную свертку можно вычислять как циклическую, так как в этом случае

$$\tilde{s}_i = s_i.$$

Чтобы обеспечить эти условия, можно так выбрать длину  $n$  циклической свертки, чтобы она была больше, чем  $N + L - 1$ , пополая последовательности  $\mathbf{g}$  и  $\mathbf{d}$  нулями до длины  $n$ . Тогда для вычисления линейной свертки можно пользоваться алгоритмом вычисления циклической свертки.

Схема КИО-фильтра, вычисляющего циклическую свертку, имеет вид



Последовательность  $\mathbf{d}$  повторяется дважды на входе фильтра. На выходе формируется последовательность из  $3n-1$  компонент, первые  $n$  из которых равны первым  $n$  компонентам линейной свертки, средние  $n$  — компонентам циклической свертки, последние  $n-1$  — последним компонентам линейной свертки.

Циклическую свертку можно также интерпретировать в терминах умножения многочленов. Пусть

$$d(x) = \sum_{i=0}^{n-1} d_i x^i, \quad g(x) = \sum_{i=0}^{n-1} g_i x^i$$

— многочлены степени  $n-1$ , а  $s(x) = d(x)g(x)$ . Мы уже видели, что циклическая свертка вычисляется по линейной "обратным вложением" коэффициентов при высших степенях  $x$ . Это можно представить в виде записи

$$\tilde{s}(x) \equiv s(x) \pmod{x^n - 1},$$

т. е. коэффициенты многочлена  $\tilde{s}(x)$  являются компонентами циклической свертки последовательностей  $\mathbf{d}$  и  $\mathbf{g}$ . Действительно, так как  $x^n \equiv 1 \pmod{x^n - 1}$ , то

$$\begin{aligned} s(x) \pmod{x^n - 1} &= \sum_{i=0}^{2n-2} s_i x^i \pmod{x^n - 1} = \sum_{i=0}^{n-1} s_i x^i + \sum_{i=n}^{2n-2} s_i x^i \pmod{x^n - 1} \equiv \\ &\sum_{i=0}^{n-1} s_i x^i + \sum_{i=n}^{2n-2} s_i x^{i-n} = \sum_{j=0}^{n-1} s_j x^j + \sum_{\substack{j=0 \\ (j=i-n)}}^{n-2} s_{n+j} x^j = \sum_{j=0}^{n-2} (s_j + s_{n+j}) x^j + s_{n-1} x^{n-1}. \end{aligned}$$

В силу формул (5):

$$\begin{aligned} s_j + s_{n+j} &= \tilde{s}_j, \quad (j = 0, 1, \dots, n-2) \\ s_{n-1} &= \tilde{s}_{n-1}, \end{aligned}$$

поэтому

$$s(x) \pmod{x^n - 1} = \sum_{j=0}^{n-2} \tilde{s}_j x^j + \tilde{s}_{n-1} x^{n-1} = \sum_{j=0}^{n-1} \tilde{s}_j x^j = \tilde{s}(x),$$

т. е. коэффициенты многочлена  $\tilde{s}(x)$  являются компонентами циклической свертки последовательностей  $\mathbf{d}$  и  $\mathbf{g}$  длины  $n$ .



### §3. АЛГОРИТМ КУКА – ТООМА

Этот алгоритм вычисления линейной свертки был разработан как метод умножения двух многочленов.

Запишем линейную свертку в виде произведения двух многочленов

$$s(x) = d(x)g(x),$$

где  $\deg d(x) = N - 1$ ,  $\deg g(x) = L - 1$ . Тогда  $\deg s(x) = N + L - 2$ , поэтому многочлен  $s(x)$  однозначно определяется своими значениями в  $N + L - 1$  различных точках.

Пусть  $\beta_0, \beta_1, \dots, \beta_{N+L-2}$  – множество из  $N + L - 1$  различных вещественных чисел. Если известны значения  $s(\beta_k)$  для  $k = 0, 1, \dots, N + L - 2$ , то многочлен  $s(x)$  можно отыскать, пользуясь интерполяционной формулой Лагранжа

$$s(x) = \sum_{k=0}^{N+L-2} s(\beta_k) \prod_{\substack{j=0 \\ j \neq k}}^{N+L-2} \frac{x - \beta_j}{\beta_k - \beta_j}. \quad (1)$$

Идея алгоритма Кука – Тоома состоит в том, чтобы сначала вычислить значения  $s(\beta_k)$  ( $k = 0, 1, \dots, N + L - 2$ ), а затем воспользоваться интерполяцией Лагранжа. Поэтому алгоритм выглядит следующим образом:

1. Вычислить  $d(\beta_k)$  и  $g(\beta_k)$ ,  $k = 0, 1, \dots, N + L - 2$ .
2. Вычислить  $s(\beta_k) = d(\beta_k)g(\beta_k)$ ,  $k = 0, 1, \dots, N + L - 2$ .
3. Построить  $s(x)$  по формуле (1).

Умножения даются равенствами  $s(\beta_k) = d(\beta_k)g(\beta_k)$ ,  $k = 0, 1, \dots, N + L - 2$ . Всего таких равенств  $N + L - 1$ , т. е. здесь мы имеем  $N + L - 1$  умножений, и если разумно выбирать точки  $\beta_k$ , то полное число умножений будет исчерпываться только этими умножениями.

Имеются и другие умножения, а именно те, которые входят в вычисление величин  $d(\beta_k)$  и  $g(\beta_k)$ , и те, которые участвуют в интерполяционной формуле Лагранжа, но мы их пока не учитываем. На самом деле, не учитывать можно только умножение на 0 и  $\pm 1$ , но, как правило, за счет некоторых предварительных вычислений, сделанных раз и навсегда, удастся свести указанные константы к 0 и  $\pm 1$ .

В качестве примера рассмотрим, как работает алгоритм Кука – Тоома в случае вычисления  $2 \times 2$  - свертки. К такому вычислению сводится прохождение двух чисел (данных) через КИО-фильтр с двумя отводами. Прямой алгоритм вычисления содержит 4 умножения и одно сложение. Мы построим два алгоритма с тремя умножениями, отличающиеся друг от друга числом сложений.

Пусть

$$d(x) = d_0 + d_1x, \quad g(x) = g_0 + g_1x \quad \text{и} \quad s(x) = d(x)g(x).$$

*1 алгоритм.* Так как  $\deg s(x) = 2$ , то надо выбрать три различных вещественных числа. Возьмем  $\beta_0 = 0$ ,  $\beta_1 = 1$ ,  $\beta_2 = -1$ . Тогда

$$\begin{aligned} d(\beta_0) &= d(0) = d_0, & g(\beta_0) &= g(0) = g_0, \\ d(\beta_1) &= d(1) = d_0 + d_1, & g(\beta_1) &= g(1) = g_0 + g_1, \\ d(\beta_2) &= d(-1) = d_0 - d_1, & g(\beta_2) &= g(-1) = g_0 - g_1. \end{aligned}$$

Вычислим  $s(\beta_k)$ :

$$\begin{aligned} s(\beta_0) &= g(\beta_0)d(\beta_0) = g_0d_0, \\ s(\beta_1) &= g(\beta_1)d(\beta_1) = (g_0 + g_1)(d_0 + d_1), \\ s(\beta_2) &= g(\beta_2)d(\beta_2) = (g_0 - g_1)(d_0 - d_1). \end{aligned}$$

Если КИО-фильтр задается фиксированным многочленом  $g(x)$ , то константы  $g(\beta_k)$  не надо каждый раз вычислять заново, их можно вычислить один раз и запомнить, поэтому в общую сложность алгоритма эти вычисления можно не включать. (При этом коэффициенты многочлена  $g(x)$  можно не запоминать.)

По интерполяционной формуле Лагранжа

$$s(x) = s(\beta_0)L_0(x) + s(\beta_1)L_1(x) + s(\beta_2)L_2(x),$$

где

$$L_0(x) = \prod_{j=1}^2 \frac{x - \beta_j}{\beta_0 - \beta_j} = \frac{(x-1)(x-1)}{(0-1)(0-1)} = -x^2 + 1.$$

$$L_1(x) = \prod_{\substack{j=0 \\ j \neq 1}}^2 \frac{x - \beta_j}{\beta_1 - \beta_j} = \frac{x(x+1)}{1(1+1)} = \frac{1}{2}(x^2 + x),$$

$$L_2(x) = \prod_{j=0}^1 \frac{x - \beta_j}{\beta_2 - \beta_j} = \frac{x(x-1)}{(-1)(-1-1)} = \frac{1}{2}(x^2 - x).$$

Значит,

$$\begin{aligned} s(x) &= s(\beta_0)(1 - x^2) + s(\beta_1)\frac{1}{2}(x^2 + x) + s(\beta_2)\frac{1}{2}(x^2 - x) = \\ &= s(\beta_0) + \frac{1}{2}(s(\beta_1) - s(\beta_2))x + \{-s(\beta_0) + \frac{1}{2}(s(\beta_1) + s(\beta_2))\}x^2, \end{aligned}$$

т. е.

$$s_0 = s(\beta_0),$$

$$s_1 = \frac{1}{2}(s(\beta_1) - s(\beta_2)),$$

$$s_2 = -s(\beta_0) + \frac{1}{2}(s(\beta_1) + s(\beta_2)).$$

Деление на 2 можно "убрать", для чего заменить константы  $g(\beta_k)$  новыми, поглощающими этот делитель. Пусть

$$G_0 = g_0, \quad G_1 = \frac{1}{2}(g_0 + g_1), \quad G_2 = \frac{1}{2}(g_0 - g_1),$$

тогда положим

$$\tilde{L}_0(x) = 1 - x^2, \quad \tilde{L}_1(x) = x^2 + x, \quad \tilde{L}_2(x) = x^2 - x.$$

В результате получаем

$$s(x) = s(\beta_0)\tilde{L}_0(x) + \left(\frac{1}{2}s(\beta_1)\right)\tilde{L}_1(x) + \left(\frac{1}{2}s(\beta_2)\right)\tilde{L}_2(x),$$

причем

$$s(\beta_0) = d(\beta_0)G_0,$$

$$\frac{1}{2}s(\beta_1) = d(\beta_1)G_1,$$

$$\frac{1}{2}s(\beta_2) = d(\beta_2)G_2.$$

Введем также обозначения

$$\begin{aligned} D_0 &= d(\beta_0) = d_0, & D_1 &= d(\beta_1) = d_0 + d_1, & D_2 &= d(\beta_2) = d_0 - d_1, \\ S_0 &= D_0 G_0, & S_1 &= D_1 G_1, & S_2 &= D_2 G_2, \end{aligned}$$

тогда

$$s(x) = S_0(1 - x^2) + S_1(x^2 + x) + S_2(x^2 - x) = S_0 + (S_1 - S_2)x + (-S_0 + S_1 + S_2)x^2,$$

откуда коэффициенты  $s_i$  ( $i = 0, 1, 2$ ) многочлена  $s(x)$  выражаются следующим образом:

$$s_0 = S_0,$$

$$s_1 = S_1 - S_2,$$

$$s_2 = -S_0 + S_1 + S_2.$$



Мы получили алгоритм, в котором три умножения  $D_i G_i$  ( $i = 0, 1, 2$ ) и пять сложений (два умножения для вычисления  $D_1$  и  $D_2$  и три сложения для вычисления  $s_1, s_2$ ).

Полученный алгоритм запишем в матрично-векторной форме:

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -1 \\ -1 & 1 & 1 \end{pmatrix} \begin{pmatrix} G_0 & & \\ & G_1 & \\ & & G_2 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \end{pmatrix}$$

или  $\mathbf{s} = C G A \mathbf{d}$ , где  $A$  – матрица предсложений,  $C$  – матрица постсложений,  $G$  – диагональная матрица, отвечающая за все умножения алгоритма.

Вычисление констант  $G_i$  ( $i = 0, 1, 2$ ) в принципе тоже можно записать в матрично-векторной форме следующим образом:

$$\begin{pmatrix} G_0 \\ G_1 \\ G_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{pmatrix} \begin{pmatrix} g_0 \\ g_1 \end{pmatrix}.$$

Матрично-векторная запись дает удобную форму записи алгоритма, но вычисления, конечно, проводятся не посредством умножения матриц, а с помощью следующей последовательности сложений и умножений:

$$\begin{aligned} D_0 &:= d_0, \\ D_1 &:= d_0 + d_1, \\ D_2 &:= d_0 - d_1, \\ S_0 &:= G_0 D_0, \\ S_1 &:= G_1 D_1, \\ S_2 &:= G_2 D_2, \\ s_0 &:= S_0, \\ s_1 &:= S_1 - S_2, \\ s_2 &:= S_1 + S_2 - S_0. \end{aligned}$$

На алгоритм Кука – Тоома можно смотреть как на метод факторизации матрицы. Прямой алгоритм вычисления  $2 \times 2$  - свертки тоже можно записать в матрично-векторном виде

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} g_0 & 0 \\ g_1 & g_0 \\ 0 & g_1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \end{pmatrix},$$

или  $\mathbf{s} = T \mathbf{d}$ . С помощью алгоритма Кука – Тоома мы представили матрицу  $T$  в виде произведения трех матриц

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -1 \\ -1 & 1 & 1 \end{pmatrix} \begin{pmatrix} G_0 & & \\ & G_1 & \\ & & G_2 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & -1 \end{pmatrix}$$

т. е.  $T = C G A$ .

Полученный нами алгоритм можно модифицировать, оставив то же число умножений, но сократив количество сложений.

Очевидно, что

$$s_{N+L-2} = g_{L-1} d_{N-1}.$$

Для вычисления этого коэффициента необходимо одно умножение. Степень модифицированного многочлена

$$\tilde{s}(x) = s(x) - s_{N+L-2} x^{N+L-2} = g(x) d(x) - s_{N+L-2} x^{N+L-2}$$

равна  $N + L - 3$ , так что использующее идеи Кука – Тоома вычисление этого многочлена требует  $N + L - 2$  умножений. Общее число умножений опять равно  $N + L - 1$ , но сложений будет меньше, поскольку для построения  $\tilde{s}(x)$  требуется меньше точек  $\beta_k$ .



II алгоритм. Так как в случае  $2 \times 2$  - свертки модифицированный многочлен  $\tilde{s}(x)$  имеет степень 1, то нам требуется всего две точки. Выберем  $\beta_0 = 0$ ,  $\beta_1 = -1$ . Тогда

$$\begin{aligned} d(\beta_0) &= d(0) = d_0, & g(\beta_0) &= g(0) = g_0, \\ d(\beta_1) &= d(-1) = d_0 - d_1, & g(\beta_1) &= g(-1) = g_0 - g_1. \end{aligned}$$

Находим значения многочлена  $\tilde{s}(x)$  в этих точках:

$$\begin{aligned} \tilde{s}(\beta_0) &= g(\beta_0) d(\beta_0) - g_1 d_1 \beta_0^2 = g(0) d(0) = g_0 d_0, \\ \tilde{s}(\beta_1) &= g(\beta_1) d(\beta_1) - g_1 d_1 \beta_1^2 = g(-1) d(-1) - g_1 d_1 = (g_0 - g_1)(d_0 - d_1) - g_1 d_1, \end{aligned}$$

так как  $\tilde{s}(x) = g(x) d(x) - g_1 d_1 x^2$ .

По интерполяционной формуле Лагранжа

$$\tilde{s}(x) = \tilde{s}(\beta_0) L_0(x) + \tilde{s}(\beta_1) L_1(x),$$

где  $L_0(x) = x + 1$ ,  $L_1(x) = -x$ .

В результате получаем

$$\begin{aligned} s(x) &= g_1 d_1 x^2 + \tilde{s}(x) = g_1 d_1 x^2 + \tilde{s}(\beta_0)(x + 1) + \tilde{s}(\beta_1)(-x) = g_1 d_1 x^2 + \{\tilde{s}(\beta_0) - \tilde{s}(\beta_1)\}x + \tilde{s}(\beta_0) = \\ &= g_1 d_1 x^2 + \{g_0 d_0 - (d_0 - d_1)(g_0 - g_1) + g_1 d_1\}x + g_0 d_0. \end{aligned}$$

Это и есть искомый алгоритм, содержащий три умножения и три сложения. Его матричная форма имеет вид

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} g_0 & & \\ & g_0 - g_1 & \\ & & g_1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & -1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \end{pmatrix}.$$

Алгоритм Кука – Тоома эффективен по числу умножений, но если объем задачи растет, то число необходимых сложений также начинает быстро расти. Это происходит потому, что хороший выбор чисел  $\beta_k$  в виде 0, 1 и  $-1$  уже сделан, и с увеличением длины свертки требуется использовать  $\pm 2$ ,  $\pm 3$  и другие малые целые числа. При этом матрицы  $A$  и  $C$  будут содержать эти числа. Умножения на них, конечно, можно заменить соответствующим числом сложений, но это имеет смысл делать только в случае, когда выбираемые числа малы. Алгоритм становится слишком громоздким при вычислении сверток, больших чем  $3 \times 4$  и  $4 \times 4$ .

Посмотрим на алгоритм Кука – Тоома "с точки зрения китайской теоремы об остатках" (см. в [4] § 8). Вместо того, чтобы выбирать множество различных чисел  $\{\beta_0, \beta_1, \dots, \beta_{N+L-2}\}$ , выберем множество многочленов  $\{x - \beta_0, x - \beta_1, \dots, x - \beta_{N+L-2}\}$ , тогда, очевидно,

$$\begin{aligned} g(\beta_k) &\equiv g(x) \pmod{x - \beta_k}, \\ d(\beta_k) &\equiv d(x) \pmod{x - \beta_k}, \quad k = 0, 1, \dots, N + L - 2, \end{aligned}$$

и алгоритм Кука – Тоома можно записать в виде:

1. 
$$\begin{aligned} g(\beta_k) &\equiv g(x) \pmod{x - \beta_k}, & k &= 0, 1, \dots, N + L - 2, \\ d(\beta_k) &\equiv d(x) \pmod{x - \beta_k}, & k &= 0, 1, \dots, N + L - 2; \end{aligned}$$
2. 
$$s(\beta_k) = g(\beta_k) d(\beta_k), \quad k = 0, 1, \dots, N + L - 2;$$

3. 
$$s(x) = \sum_{k=0}^{N+L-2} s(\beta_k) L_k(x), \quad \text{где} \quad L_k(x) = \prod_{\substack{j=0 \\ j \neq k}}^{N+L-2} \frac{x - \beta_j}{\beta_k - \beta_j}.$$

С этой точки зрения интерполяционную формулу Лагранжа можно рассматривать как некоторую форму обращения китайской теоремы об остатках для того частного случая, когда в качестве многочленов-модулей выбираются только многочлены первой степени. Если же заменить многочлены первой степени многочленами более высоких степеней, то это позволяет увеличить конструктивные возможности алгоритма и приводит к идее следующего алгоритма, который носит имя Винограда.



#### §4. АЛГОРИТМЫ ВИНОГРАДА ВЫЧИСЛЕНИЯ КОРОТКИХ СВЕРТОК

Рассмотрим следующую общую задачу. Пусть требуется вычислить

$$s(x) = g(x) d(x) \pmod{m(x)}, \quad (1)$$

где  $m(x)$  — фиксированный многочлен степени  $n$  над полем  $F$ ,  $g(x)$  и  $d(x)$  — многочлены над тем же полем, причем их степени меньше  $n$ .

Задача вычисления линейной свертки

$$s(x) = g(x) d(x)$$

может быть вложена в эту задачу, если в качестве  $n$  выбрать целое число, большее  $\deg d(x) + \deg g(x) = \deg s(x)$ , а в качестве  $m(x)$  — произвольный многочлен степени  $n$ . Поскольку степень  $m(x)$  больше степени произведения  $d(x)g(x)$ , то  $d(x)g(x) \pmod{m(x)}$  совпадает с  $d(x)g(x) = s(x)$ .

Задача вычисления циклической свертки также может быть вложена в задачу (1), если  $\deg d(x) = \deg g(x) = n - 1$  и в качестве  $m(x)$  взять многочлен  $x^n - 1$ . Получаем

$$\tilde{s}(x) = g(x) d(x) \pmod{x^n - 1},$$

что, как мы видели выше, эквивалентно вычислению циклической свертки последовательностей коэффициентов многочленов  $d(x)$  и  $g(x)$ .

Наша цель — заменить задачу (1) некоторым множеством задач с меньшим числом вычислений. Чтобы разбить задачу (1) на подзадачи, разложим многочлен  $m(x)$  на взаимно простые множители над некоторым подполем  $K$  поля  $F$  (считаем, что коэффициенты  $m(x)$  принадлежат этому подполю):

$$m(x) = m_0(x) m_1(x) \dots m_{s-1}(x). \quad (2)$$

Обычно, если  $F$  — поле вещественных чисел  $\mathbb{R}$  или поле комплексных чисел  $\mathbb{C}$ , то в качестве подполей разложения выбираются соответственно поле рациональных чисел  $\mathbb{Q}$  и поле гауссовых рациональных чисел  $\mathbb{Q}[i]$ , хотя в принципе можно выбирать и другие подполя. Если свертка вычисляется в конечном поле  $GF(p^m)$ , то в качестве подполя разложения обычно выбирается  $GF(p)$ . Нашей целью является минимизация числа умножений в поле  $F$ , при этом мы не будем пытаться минимизировать число умножений в подполе  $K$ . В большинстве случаев этими умножениями в подполе оказываются умножения на малые целые числа, которые всегда можно заменить сложениями, или на числа 0 и  $\pm 1$ .

Таким образом, решая задачу над полем  $\mathbb{R}$  или  $\mathbb{C}$ , мы в дальнейшем не будем учитывать умножения на рациональные числа, хотя всегда будем следить за тем, чтобы это были на самом деле *малые целые числа*.

Заменяем задачу (1) на  $s$  подзадач нахождения многочленов  $s_k(x)$ :

$$s_k(x) \equiv s(x) \pmod{m_k(x)}, \quad k = 0, 1, \dots, s-1.$$

Если известны многочлены  $s_k(x)$  ( $k = 0, 1, \dots, s-1$ ), то по китайской теореме об остатках для многочленов  $s(x)$  можно восстановить, пользуясь формулой § 8 из [4]

$$s(x) = \sum_{k=0}^{s-1} s_k(x) M_k(x) N_k(x) \pmod{m(x)}, \quad (3)$$

где  $M_k(x) = \frac{m(x)}{m_k(x)}$ ,  $M_k(x) N_k(x) + m_k(x) n_k(x) = 1$  для  $k = 0, 1, \dots, s-1$ .

Так как

$$\begin{aligned} s_k(x) &\equiv s(x) \pmod{m_k(x)} \equiv d(x) g(x) \pmod{m_k(x)} \equiv \\ &\equiv (d(x) \pmod{m_k(x)}) (g(x) \pmod{m_k(x)}) \pmod{m_k(x)} \equiv d_k(x) g_k(x) \pmod{m_k(x)}, \end{aligned}$$



где

$$\begin{aligned} d_k(x) &\equiv d(x) \pmod{m_k(x)} \\ g_k(x) &\equiv g(x) \pmod{m_k(x)}. \end{aligned}$$

то сначала следует найти вычеты многочленов  $d(x)$  и  $g(x)$  по модулям  $m_k(x)$  ( $k = 0, 1, \dots, s-1$ ), затем вычислить  $s_k(x)$

$$s_k(x) \equiv d_k(x) g_k(x) \pmod{m_k(x)} \quad (4)$$

и, наконец, восстановить  $s(x)$  по формуле (3).

Таким образом, алгоритм Винограда состоит из следующих шагов:

1. Вычислить вычеты

$$\begin{aligned} d_k(x) &\equiv d(x) \pmod{m_k(x)} \\ g_k(x) &\equiv g(x) \pmod{m_k(x)}, \quad k = 0, 1, \dots, s-1; \end{aligned}$$

2. Найти многочлены

$$s_k(x) \equiv d_k(x) g_k(x) \pmod{m_k(x)}, \quad k = 0, 1, \dots, s-1;$$

3. Восстановить  $s(x)$  по формуле (3), предварительно отыскав многочлены  $M_k(x)$  и  $N_k(x)$  ( $k = 0, 1, \dots, s-1$ ).

Так как коэффициенты многочленов  $m_k(x)$  принадлежат подполю  $K$ , то на первом шаге нет умножений (остаются только сложения коэффициентов многочленов  $d(x)$  и  $g(x)$ ), коэффициенты многочленов  $M_k(x)$  и  $N_k(x)$  ( $k = 0, 1, \dots, s-1$ ) также принадлежат подполю  $K$ , поэтому и на третьем шаге нет умножений. Умножения возникают только на втором шаге алгоритма при вычислении коротких сверток, задаваемых произведениями многочленов  $d_k(x) g_k(x)$ . Так как число коэффициентов у многочленов  $d_k(x)$  и  $g_k(x)$  равно степени многочлена  $m_k(x)$ , то полное число умножений при прямом алгоритме для перемножения  $d_k(x)$  на  $g_k(x)$  равно  $\{ \deg m_k(x) \}^2$ . Тогда для решения всех подзадач потребуется

$$\sum_{k=0}^{s-1} \{ \deg m_k(x) \}^2$$

умножений в поле  $F$ . Это дает существенное уменьшение числа необходимых умножений для решения задачи (1). Действительно,

$$n = \deg m(x) = \sum_{k=0}^{s-1} \deg m_k(x) = \sum_{k=0}^{s-1} n_k \quad (n_k = \deg m_k(x)),$$

поэтому

$$n^2 = \left( \sum_{k=0}^{s-1} n_k \right)^2 > \sum_{k=0}^{s-1} n_k^2.$$

Заметим, что ту же самую идею разбиения на подзадачи можно применить к решению подзадач (4), что позволит еще больше снизить число необходимых умножений в поле  $F$ .

В качестве иллюстрации работы алгоритма Винограда рассмотрим пример вычисления свертки 3-точечного вектора с 2-точечным вектором ( $3 \times 2$  - свертка). Пусть

$$\begin{aligned} g(x) &= g_0 + g_1 x \\ d(x) &= d_0 + d_1 x + d_2 x^2. \end{aligned}$$

Прямой алгоритм вычисления  $s(x) = d(x) g(x)$  содержит 6 умножений и два сложения. Действительно,

$$s(x) = g_0 d_0 + (d_0 g_1 + d_1 g_0) x + (d_1 g_1 + d_2 g_0) x^2 + d_2 g_1 x^3.$$



Сначала мы построим алгоритм, который содержит пять умножений и восемь сложений. Это очень хороший алгоритм, но его построение будет содержать нетривиальную подзадачу, которая часто будет использоваться в дальнейшем.

Степень многочлена  $s(x)$  равна 3, поэтому выберем многочлен  $m(x)$  четвертой степени, пусть

$$m(x) = x(x-1)(x^2+1) = m_0(x)m_1(x)m_2(x).$$

Основное поле – поле вещественных чисел, в качестве подполя рассматриваем поле рациональных чисел. Множители  $m_i(x)$  попарно взаимно просты, их коэффициенты – рациональные, даже целые, числа.

На первом шаге вычисляются вычеты

$$\begin{aligned} d_0(x) &\equiv d(x) \pmod{x} = d_0, & g_0(x) &\equiv g(x) \pmod{x} = g_0, \\ d_1(x) &\equiv d(x) \pmod{x-1} = d_0 + d_1 + d_2, & g_1(x) &\equiv g(x) \pmod{x-1} = g_0 + g_1, \\ d_2(x) &\equiv d(x) \pmod{x^2+1} = (d_0 - d_2) + d_1x, & g_2(x) &\equiv g(x) \pmod{x^2+1} = g_0 + g_1x. \end{aligned}$$

Как и утверждалось выше, на этом шаге мы не получили ни одного умножения. На втором шаге вычисляем многочлены  $s_k(x) \equiv d_k(x)g_k(x) \pmod{m_k(x)}$ ,  $k = 0, 1, 2$ :

$$\begin{aligned} s_0(x) &\equiv d_0(x)g_0(x) \pmod{x} = d_0g_0, \\ s_1(x) &\equiv d_1(x)g_1(x) \pmod{x-1} = (d_0 + d_1 + d_2)(g_0 + g_1), \\ s_2(x) &\equiv d_2(x)g_2(x) \pmod{x^2+1} = \{(d_0 - d_2) + d_1x\}(g_0 + g_1x) \pmod{x^2+1} \\ &\equiv \{(d_0 - d_2)g_0 - d_1g_1\} + \{(d_0 - d_2)g_1 + d_1g_0\}x. \end{aligned}$$

Для вычисления  $s_0(x)$  и  $s_1(x)$  требуется по одному умножению. Для вычисления коэффициентов  $s_2(x)$  требуется 4 умножения. Однако эту подзадачу можно решить отдельно, построив для нее быстрый алгоритм с тремя умножениями.

Для упрощения выкладок сначала введем обозначения:

$$d_0^{(2)} = d_0 - d_2, \quad d_1^{(2)} = d_1, \quad g_0^{(2)} = g_0, \quad g_1^{(2)} = g_1,$$

тогда

$$d_2(x) = d_0^{(2)} + d_1^{(2)}x, \quad g_2(x) = g_0^{(2)} + g_1^{(2)}x$$

и

$$s_2(x) = (d_0^{(2)}g_0^{(2)} - d_1^{(2)}g_1^{(2)}) + (d_0^{(2)}g_1^{(2)} + d_1^{(2)}g_0^{(2)})x = s_0^{(2)} + s_1^{(2)}x,$$

где

$$\begin{aligned} s_0^{(2)} &= d_0^{(2)}g_0^{(2)} - d_1^{(2)}g_1^{(2)}, \\ s_1^{(2)} &= d_0^{(2)}g_1^{(2)} + d_1^{(2)}g_0^{(2)}. \end{aligned}$$

Последние две формулы напоминают структуру произведения двух комплексных чисел (на самом деле, перемножение двух линейных многочленов по модулю  $x^2 + 1$  в точности эквивалентно умножению комплексных чисел), поэтому можно, например, воспользоваться алгоритмом §1:

$$\begin{aligned} s_0^{(2)} &= (d_0^{(2)} - d_1^{(2)})g_1^{(2)} + d_0^{(2)}(g_0^{(2)} - g_1^{(2)}), \\ s_1^{(2)} &= (d_0^{(2)} - d_1^{(2)})g_1^{(2)} + d_1^{(2)}(g_0^{(2)} + g_1^{(2)}), \end{aligned} \tag{5}$$

или

$$\begin{aligned} s_0^{(2)} &= (g_0^{(2)} - g_1^{(2)})d_1^{(2)} + g_0^{(2)}(d_0^{(2)} - d_1^{(2)}), \\ s_1^{(2)} &= (g_0^{(2)} - g_1^{(2)})d_1^{(2)} + g_1^{(2)}(d_0^{(2)} + d_1^{(2)}). \end{aligned} \tag{6}$$

Можно подобрать и другие формулы, требующие трех умножений, например,

$$\begin{aligned} s_0^{(2)} &= (d_0^{(2)} + d_1^{(2)})g_0^{(2)} - d_1^{(2)}(g_0^{(2)} + g_1^{(2)}), \\ s_1^{(2)} &= (d_0^{(2)} + d_1^{(2)})g_0^{(2)} + d_0^{(2)}(g_1^{(2)} - g_0^{(2)}). \end{aligned} \tag{7}$$



Теперь сравним эффективность формул (5), (6) и (7) по числу операций. При этом будем считать, что константы, связанные с многочленом  $g(x)$ , являющиеся постоянными фильтра, которые в серии вычислений не меняются. Эти константы можно вычислить один раз и запомнить, т. е. их вычисление не влияет на сложность алгоритма. Тогда в формулах (5) и (7) имеется по три сложения, а в формулах (6) – четыре. Значит, будем выбирать одну из формул (5) или (7) для вычисления коэффициентов многочлена  $s_2(x)$ .

Выберем, например, формулы (7), тогда алгоритм решения последней подзадачи дается равенством

$$\begin{pmatrix} s_0^{(2)} \\ s_0^{(2)} \end{pmatrix} = \begin{pmatrix} 1 & 0 & -1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} g_0^{(2)} & & \\ & g_1^{(2)} - g_0^{(2)} & \\ & & g_1^{(2)} - g_0^{(2)} \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} d_0^{(2)} \\ d_1^{(2)} \end{pmatrix}$$

и содержит три умножения и три сложения.

Переходим к последнему шагу алгоритма. Для нахождения  $s(x)$  по формуле (3) следует найти многочлены  $M_k(x)$  и  $N_k(x)$ ,  $k = 0, 1, 2$ . Это нетрудно сделать, воспользовавшись алгоритмом Евклида или расширенным алгоритмом Евклида. Результаты вычислений запишем в следующую таблицу:

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$	$n_k(x)$
0	$x$	$x^3 - x^2 + x - 1$	$-1$	$x^2 - x + 1$
1	$x - 1$	$x^3 + x$	$\frac{1}{2}$	$-\frac{1}{2}(x^2 + x + 2)$
2	$x^2 + 1$	$x^2 - x$	$\frac{1}{2}(x - 1)$	$-\frac{1}{2}(x - 2)$

Действительно,

$$x^3 - x^2 + x - 1 = x(x^2 - x + 1) - 1$$

или

$$M_0(x) = m_0(x)(x^2 - x + 1) - 1,$$

откуда

$$1 = -M_0(x) + m_0(x)(x^2 - x + 1)$$

и  $N_0(x) = -1$ ,  $n_0(x) = x^2 - x + 1$ .

Далее

$$x^3 + x = (x - 1)(x^2 + x + 2) + 2,$$

или

$$M_1(x) = m_1(x)(x^2 + x + 2) + 2,$$

откуда

$$1 = \frac{1}{2} M_1(x) - \frac{1}{2} (x^2 + x + 2) m_1(x).$$

Наконец,

$$x^2 - x = (x^2 + 1) + (-x - 1),$$

$$x^2 + 1 = (-x - 1)(-x + 1) + 2$$

или

$$M_2(x) = m_2(x) + r(x),$$

$$m_2(x) = r(x)(-x + 1) + 2,$$

откуда

$$1 = \frac{1}{2} m_2(x) - \frac{1}{2} (1 - x) r(x) = \frac{1}{2} m_2(x) - \frac{1}{2} (1 - x) \{M_2(x) - m_2(x)\} =$$

$$\frac{1}{2} (x - 1) M_2(x) - \frac{1}{2} (x - 2) m_2(x).$$



По формуле (3) получаем

$$\begin{aligned}
s(x) &= s_0(x)(-x^3 + x^2 - x + 1) + s_1(x) \frac{1}{2}(x^3 + x) + s_2(x) \frac{1}{2}(x^2 - x)(x - 1) \pmod{m(x)} \\
&= (-x^3 + x^2 - x + 1)s_0(x) + \frac{1}{2}(x^3 + x)s_1(x) + \frac{1}{2}(x^3 - 2x^2 + x)(s_0^{(2)} + s_1^{(2)}x) \pmod{m(x)} \\
&= (-x^3 + x^2 - x + 1)s_0(x) + (x^3 + x) \left( \frac{1}{2}s_1(x) \right) + (x^3 - 2x^2 + x) \left( \frac{1}{2}s_0^{(2)} \right) \\
&\quad + (x^4 - 2x^3 + x^2) \left( \frac{1}{2}s_1^{(2)} \right) \pmod{m(x)}.
\end{aligned}$$

Поскольку

$$m(x) = x(x-1)(x^2+1) = x^4 - x^3 + x^2 - x,$$

то

$$x^4 - 2x^3 + x^2 \equiv x^3 - x^2 + x - 2x^3 + x^2 \equiv -x^3 + x \pmod{m(x)},$$

значит,

$$s(x) = (-x^3 + x^2 - x + 1)s_0(x) + (x^3 + x) \left( \frac{1}{2}s_1(x) \right) + (x^3 - 2x^2 + x) \left( \frac{1}{2}s_0^{(2)} \right) + (-x^3 + x) \left( \frac{1}{2}s_1^{(2)} \right). \quad (8)$$

Уберем множители  $\frac{1}{2}$ , для чего включим их в константы, связанные с коэффициентами многочлена  $g(x)$ . Введем обозначения:

$$\begin{aligned}
G_0 &= g_0(x) = g_0, & G_1 &= \frac{1}{2}g_1(x) = \frac{1}{2}(g_0 + g_1), & G_2 &= \frac{1}{2}g_0^{(2)} = \frac{1}{2}g_0, \\
G_3 &= \frac{1}{2}(g_1^{(2)} - g_0^{(2)}) = \frac{1}{2}(g_1 - g_0), & G_4 &= \frac{1}{2}(g_1^{(2)} + g_0^{(2)}) = \frac{1}{2}(g_1 + g_0), \\
D_0 &= d_0(x) = d_0, & D_1 &= d_1(x) = d_0 + d_1 + d_2, & D_2 &= d_0^{(2)} + d_1^{(2)} = d_0 + d_1 - d_2, \\
D_3 &= d_0^{(2)} = d_0 - d_2, & D_4 &= d_1^{(2)} = d_1, & S_k &= G_k D_k \quad (k = 0, 1, \dots, 4).
\end{aligned}$$

Тогда

$$\begin{aligned}
s_0(x) &= S_0, \\
\frac{1}{2}s_1(x) &= S_1, \\
\frac{1}{2}s_0^{(2)} &= S_2 - S_4, \\
\frac{1}{2}s_1^{(2)} &= S_2 + S_3
\end{aligned}$$

и (8) переписывается в виде

$$s(x) = (1 - x + x^2 - x^3)S_0 + (x + x^3)S_1 + (x - 2x^2 + x^3)(S_2 - S_4) + (x - x^3)(S_2 + S_3).$$

Собирая коэффициенты при одинаковых степенях  $x$ , окончательно получаем

$$s(x) = S_0 + (-S_0 + S_1 + 2S_2 + S_3 - S_4)x + (S_0 - 2S_2 + 2S_4)x^2 + (-S_0 + S_1 - S_3 - S_4)x^3. \quad (9)$$

В матрично-векторной форме полученный алгоритм записывается в виде

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 2 & 1 & -1 \\ 1 & 0 & -2 & 0 & 2 \\ -1 & 1 & 0 & -1 & -1 \end{pmatrix} \begin{pmatrix} G_0 & & & & \\ & G_1 & & & \\ & & G_2 & & \\ & & & G_3 & \\ & & & & G_4 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \end{pmatrix}. \quad (10)$$



Поясним, как появилась формула (10). С помощью матрицы предположений получаем константы  $D_k$  ( $k = 0, 1, \dots, 4$ ). Согласно введенным выше обозначениям.

$$\begin{pmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \\ D_4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \end{pmatrix}.$$

Теперь нужно выполнить умножения алгоритма:

$$S_k = D_k G_k \quad (k = 0, 1, \dots, 4),$$

для чего пользуемся диагональной матрицей:

$$\begin{pmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \\ S_4 \end{pmatrix} = \begin{pmatrix} G_0 & & & & \\ & G_1 & & & \\ & & G_2 & & \\ & & & G_3 & \\ & & & & G_4 \end{pmatrix} \begin{pmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \\ D_4 \end{pmatrix},$$

и, наконец, для получения коэффициентов  $s_i$  ( $i = 0, 1, 2, 3$ ) многочлена  $s(x)$  выполняем сложения, соответствующие формуле (9)

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 2 & 1 & -1 \\ 1 & 0 & -2 & 0 & 2 \\ -1 & 1 & 0 & -1 & -1 \end{pmatrix} \begin{pmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \\ S_4 \end{pmatrix}.$$

Подсчитаем число сложений, требуемых для выполнения алгоритма. Никакой специальной теории для минимизации количества сложений не разработано, поэтому здесь каждый может экспериментировать с порядком их выполнения.

Предположения в нашем примере легко выполнить за четыре сложения. Действительно,

$$\begin{aligned} D_0 &:= d_0, \\ D_3 &:= d_0 - d_2, \\ D_2 &:= D_3 + d_1, \\ D_1 &:= d_0 + d_1 + d_2, \\ D_4 &:= d_1. \end{aligned}$$

Постсложения можно реализовать с помощью восьми сложений, например, следующим образом:

$$\begin{aligned} s_0 &:= S_0, \\ t_1 &:= S_4 - S_2, \\ t_2 &:= t_1 + t_1, \\ t_3 &:= S_3 + S_4, \\ s_2 &:= S_0 + t_2, \\ s_1 &:= -s_2 + t_3 + S_1, \\ s_3 &:= -S_0 + S_1 - t_3. \end{aligned}$$

Итак, наш алгоритм содержит пять вещественных умножений и 12 вещественных сложений. Его, как было отмечено выше, можно улучшить.

Более общая форма алгоритма Винограда соответствует выбору многочлена  $m(x)$  меньшей степени. Это приводит к неправильному вычислению свертки, но ошибку легко подправить с помощью нескольких дополнительных вычислений.



Согласно алгоритму деления для многочленов можно записать

$$s(x) = q(x) m(x) + r(x),$$

где  $r(x) \equiv s(x) \pmod{m(x)}$ .

Мы рассмотрели случай, когда  $\deg m(x) > \deg s(x)$ , в котором частное  $q(x)$  равно нулю. Если  $\deg m(x) \leq \deg s(x)$ , то алгоритм Винограда позволяет вычислить  $s(x)$  по модулю  $m(x)$ . Член  $q(x) m(x)$  представляет собой погрешность, которую можно вычислить дополнительно и прибавить к результату. Простейшим является случай, когда  $\deg m(x) = \deg s(x)$ . Тогда многочлен  $q(x)$  является константой. Если  $m(x)$  — приведенный многочлен степени  $n$ , то, очевидно,  $q(x) = s_n$ , где  $s_n$  — старший коэффициент многочлена  $s(x)$ , он легко вычисляется как произведение старших коэффициентов многочленов  $d(x)$  и  $g(x)$ .

Эту модификацию алгоритма Винограда мы сейчас и применим к нашему примеру.

В качестве многочлена  $m(x)$  возьмем многочлен третьей степени  $x(x-1)(x+1)$ , тогда

$$s(x) = d(x) g(x) = d_2 g_1 m(x) + r(x),$$

где

$$r(x) \equiv s(x) \pmod{x(x-1)(x+1)}.$$

Применим алгоритм Винограда для нахождения  $r(x)$ . Вычеты  $d(x)$  и  $g(x)$  по модулям  $m_0(x) = x$ ,  $m_1(x) = x-1$ ,  $m_2(x) = x+1$  равны

$$\begin{aligned} d_0(x) &= d_0, & g_0(x) &= g_0, \\ d_1(x) &= d_0 + d_1 + d_2, & g_1(x) &= g_0 + g_1, \\ d_2(x) &= d_0 - d_1 + d_2, & g_2(x) &= g_0 - g_1. \end{aligned}$$

Кроме того, нам потребуются

$$d_3(x) = d_2 \quad \text{и} \quad g_3(x) = g_1.$$

Тогда

$$\begin{aligned} s_0(x) &= d_0 g_0, \\ s_1(x) &= (d_0 + d_1 + d_2) (g_0 + g_1), \\ s_2(x) &= (d_0 - d_1 + d_2) (g_0 - g_1), \\ s_3(x) &= d_2 g_1. \end{aligned}$$

Чтобы воспользоваться китайской теоремой об остатках, найдем многочлены  $M_k(x)$  и  $N_k(x)$  ( $k = 0, 1, 2$ ), записав результаты вычислений в следующую таблицу:

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$	$n_k(x)$
0	$x$	$x^2 - 1$	$-1$	$x$
1	$x - 1$	$x^2 + x$	$\frac{1}{2}$	$-\frac{1}{2}(x + 2)$
2	$x + 1$	$x^2 - x$	$\frac{1}{2}$	$-\frac{1}{2}(x - 2)$

Таким образом,

$$r(x) = s_0(x) (1 - x^2) + s_1(x) \frac{1}{2} (x^2 + x) + s_2(x) \frac{1}{2} (x^2 - x) \pmod{m(x)}.$$

Уберем множители  $\frac{1}{2}$  в константы, связанные с коэффициентами многочлена  $g(x)$ , вводя обозначения

$$G_0 = g_0, \quad G_1 = \frac{1}{2} (g_0 + g_1), \quad G_2 = \frac{1}{2} (g_0 - g_1), \quad G_3 = g_1.$$

Обозначим также через  $D_k$  ( $k = 0, 1, 2, 3$ ) константы, связанные с коэффициентами многочлена  $d(x)$ :

$$D_0 = d_0, \quad D_1 = d_0 + d_1 + d_2, \quad D_2 = d_0 - d_1 + d_2, \quad D_3 = d_2.$$



Пусть

$$S_k = D_k G_k \quad (k = 0, 1, 2, 3).$$

Тогда

$$r(x) = S_0 (1 - x^2) + S_1 (x^2 + x) - S_2 (x^2 - x)$$

и

$$s(x) = S_3 m(x) + r(x) = S_0 (1 - x^2) + S_1 (x^2 + x) - S_2 (x^2 - x) + S_3 (x^3 - x) = \\ S_0 + (S_1 - S_2 - S_3)x + (-S_0 + S_1 + S_2)x^2 + S_3x^3. \quad (11)$$

В матрично-векторной форме полученный алгоритм может быть записан в виде

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & -1 & -1 \\ -1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} G_0 & & & \\ & G_1 & & \\ & & G_2 & \\ & & & G_3 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & -1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \end{pmatrix}.$$

Он содержит, как нетрудно видеть, всего 4 умножения и 7 сложений. Предложений всего три:

$$\begin{aligned} D_0 &:= d_0, \\ t_1 &:= d_0 + d_2, \\ D_1 &:= t_1 + d_1, \\ D_2 &:= t_1 - d_1, \\ D_3 &:= d_2. \end{aligned}$$

Постсложений – 4, как видно из формулы (11).

Алгоритм с четырьмя умножениями можно построить, не используя модифицированный алгоритм Винограда. Достаточно взять в качестве  $m(x)$  многочлен четвертой степени, раскладывающийся на линейные множители над полем  $\mathbb{Q}$ , например,  $m(x) = x(x-1)(x+1)(x-2)$ . Однако число сложений в алгоритме, полученном при таком выборе многочлена  $m(x)$ , будет гораздо больше, чем в построенном нами алгоритме.

В заключение приведем без доказательства несколько полезных теорем, которые дают нижние оценки числа умножений, требующихся для вычисления свертки и решения задачи (1).

**Теорема 1.** *Каждый алгоритм вычисления линейной свертки*

$$s(x) = d(x) g(x)$$

где  $\deg d(x) = N - 1$  и  $\deg g(x) = L - 1$ , содержит по меньшей мере  $N + L - 1$  умножений.

**Теорема 2.** *Пусть  $p(x)$  – неприводимый многочлен степени  $n$ . Каждый алгоритм вычисления произведения многочленов*

$$s(x) = d(x) g(x) \pmod{p(x)}$$

содержит по меньшей мере  $2n - 1$  умножений.

**Теорема 3.** *Пусть многочлен  $p(x)$  степени  $n$  раскладывается в произведение  $k$  (попарно взаимно простых) неприводимых многочленов. Каждый алгоритм вычисления произведения многочленов*

$$s(x) = d(x) g(x) \pmod{p(x)}$$

содержит по меньшей мере  $2n - k$  умножений.



## §5. ПРИМЕРЫ ПОСТРОЕНИЯ АЛГОРИТМОВ КОРОТКИХ ЛИНЕЙНЫХ СВЕРТОК

Сначала мы вернемся к  $2 \times 2$  - свертке и построим еще один алгоритм ее вычисления с использованием модифицированного алгоритма Винограда. Пусть

$$d(x) = d_0 + d_1 x, \quad g(x) = g_0 + g_1 x,$$

тогда  $r(x) = d(x)g(x)$  есть многочлен второй степени. В качестве  $m(x)$  выберем также многочлен второй степени, равный  $x(x-1)$ , тогда

$$s(x) = d_1 g_1 m(x) + r(x),$$

$$r(x) \equiv s(x) \pmod{m(x)}.$$

Из нахождения  $r(x)$  вычислим вычеты  $d(x)$  и  $g(x)$  по модулям  $x$  и  $x-1$ :

$$\begin{aligned} d_0(x) &= d_0, & g_0(x) &= g_0, \\ d_1(x) &= d_0 + d_1, & g_1(x) &= g_0 + g_1, \end{aligned}$$

тогда находим

$$\begin{aligned} s_0(x) &= d_0 g_0 = S_0, \\ s_1(x) &= (d_0 + d_1)(g_0 + g_1) = S_1, \end{aligned}$$

и строим многочлены  $M_k(x)$ ,  $N_k(x)$ , ( $k = 0, 1$ ):

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$	$n_k(x)$
0	$x$	$x-1$	-1	1
1	$x-1$	$x$	1	-1

тогда

$$r(x) = s_0(x)(1-x) + s_1(x)x$$

и

$$s(x) = s_0(x)(1-x) + s_1(x)x + s_2(x)(x^2-x),$$

$$\text{где } s_2(x) = d_1 g_1 = S_2.$$

Значит,

$$s(x) = S_0(1-x) + S_1 x + S_2(x^2-x) = S_0 + (-S_0 + S_1 - S_2)x + S_2 x^2$$

или в матрично-векторной форме

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & -1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} g_0 & & \\ & g_0 + g_1 & \\ & & g_1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \end{pmatrix}.$$

В построенном алгоритме всего три умножения. Сложений также три: одно – в матрице предположений и два – в матрице постсложений.

Если в качестве  $m(x)$  взять многочлен  $x(x+1)$ , то получим точно такой же алгоритм, как алгоритм II в §3. Оба алгоритма имеют одинаковую сложность и отличаются только видом сложений.

Рассмотрим  $3 \times 3$  - свертку и три разных алгоритма ее вычисления. Пусть

$$d(x) = d_0 + d_1 x + d_2 x^2, \quad g(x) = g_0 + g_1 x + g_2 x^2$$

и

$$s(x) = d(x)g(x) = d_0 g_0 + (d_0 g_1 + d_1 g_0)x + (d_0 g_2 + d_1 g_1 + d_2 g_0)x^2 + (d_1 g_2 + d_2 g_1)x^3 + d_2 g_2 x^4.$$



Прямой алгоритм вычисления коэффициентов многочлена  $s(x)$  содержит 9 умножений и 4 сложения. Его также можно записать в матрично-векторной форме

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \end{pmatrix} = \begin{pmatrix} g_0 & 0 & 0 \\ g_1 & g_0 & 0 \\ g_2 & g_1 & g_0 \\ 0 & g_2 & g_1 \\ 0 & 0 & g_2 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \end{pmatrix}.$$

Рассмотрим теперь следующие тождества

$$\begin{aligned} d_0 g_1 + d_1 g_0 &= (g_0 + g_1)(d_0 + d_1) - g_0 d_0 - g_1 d_1, \\ d_0 g_2 + d_1 g_1 + d_2 g_0 &= (g_0 + g_2)(d_0 + d_2) - g_0 d_0 - g_2 d_2 + g_1 d_1, \\ d_1 g_2 + d_2 g_1 &= (g_1 + g_2)(d_1 + d_2) - g_1 d_1 - g_2 d_2. \end{aligned}$$

Эти тождества позволяют вычислить коэффициенты свертки с помощью 6 умножений и 10 сложений.

Этот алгоритм нельзя построить с помощью алгоритма Винограда. Матричная форма записи алгоритма дается равенством:

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 1 & 0 & 0 \\ -1 & 1 & -1 & 0 & 1 & 0 \\ 0 & -1 & -1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} g_0 & & & & & \\ & g_1 & & & & \\ & & g_2 & & & \\ & & & g_0 + g_1 & & \\ & & & & g_0 + g_2 & \\ & & & & & g_1 + g_2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \end{pmatrix}.$$

Оптимальный же по числу умножений алгоритм содержит 5 умножений и 20 сложений. Построением его мы сейчас и займемся.

Воспользуемся модифицированным алгоритмом Винограда, взяв в качестве  $m(x)$  многочлен

$$x(x-1)(x+1)(x-2) = x^4 - 2x^3 - x^2 + 2x.$$

Вычеты  $d(x)$  и  $g(x)$  по выбранным модулям соответственно равны

$$\begin{aligned} d_0(x) &= d_0, & g_0(x) &= g_0, \\ d_1(x) &= d_0 + d_1 + d_2, & g_1(x) &= g_0 + g_1 + g_2, \\ d_2(x) &= d_0 - d_1 + d_2, & g_2(x) &= g_0 - g_1 + g_2, \\ d_3(x) &= d_0 + 2d_1 + 4d_2, & g_3(x) &= g_0 + 2g_1 + 4g_2. \end{aligned}$$

Тогда

$$\begin{aligned} s_0(x) &= d_0 g_0, \\ s_1(x) &= (d_0 + d_1 + d_2)(g_0 + g_1 + g_2), \\ s_2(x) &= (d_0 - d_1 + d_2)(g_0 - g_1 + g_2), \\ s_3(x) &= (d_0 + 2d_1 + 4d_2)(g_0 + 2g_1 + 4g_2). \end{aligned}$$

Кроме того, нам потребуется

$$s_4(x) = d_2 g_2.$$

Далее, получим многочлены  $M_k(x)$  и  $N_k(x)$  ( $k = 0, 1, 2, 3$ ):

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$	$n_k(x)$
0	$x$	$x^3 - 2x^2 - x + 2$	$\frac{1}{2}$	$-\frac{1}{2}(x^2 - 2x - 1)$
1	$x - 1$	$x^3 - x^2 - 2x$	$-\frac{1}{2}$	$\frac{1}{2}(x^2 - 2)$
2	$x + 1$	$x^3 - 3x^2 + 2x$	$-\frac{1}{6}$	$\frac{1}{6}(x^2 - 4x + 6)$
3	$x - 2$	$x^3 - x$	$\frac{1}{6}$	$-\frac{1}{6}(x^2 + 2x + 3)$



Теперь получаем  $s(x)$ , пользуясь китайской теоремой об остатках и добавляя погрешность  $(x^3 - x^2 - x + 2)$ :

$$s(x) = s_0(x) \frac{1}{2} (x^3 - 2x^2 - x + 2) + s_1(x) \left(-\frac{1}{2}\right) (x^3 - x^2 - 2x) + s_2(x) \left(-\frac{1}{6}\right) (x^3 - 3x^2 + 2x) + \\ s_3(x) \left(\frac{1}{6}\right) (x^3 - x) + s_4(x) (x^4 - 2x^3 - x^2 + 2x).$$

Выбираем множители  $\frac{1}{2}$  и  $\frac{1}{6}$  в константы, связанные с многочленом  $g(x)$ :

$$G_0 = \frac{1}{2}g_0(x) = \frac{1}{2}g_0, \quad G_1 = \frac{1}{2}g_1(x) = \frac{1}{2}(g_0 + g_1 + g_2), \quad G_2 = \frac{1}{6}g_2(x) = \frac{1}{6}(g_0 - g_1 + g_2),$$

$$G_3 = \frac{1}{6}g_3(x) = \frac{1}{6}(g_0 + 2g_1 + 4g_2), \quad G_4 = g_2.$$

Тогда, если ввести обозначения

$$D_0 = d_0, \quad D_1 = d_0 + d_1 + d_2, \quad D_2 = d_0 - d_1 + d_2,$$

$$D_3 = d_0 + 2d_1 + 4d_2, \quad D_4 = d_2$$

$$S_k = D_k G_k, \quad k = 0, 1, \dots, 4,$$

$$s(x) = S_0(x^3 - 2x^2 - x + 2) + S_1(-x^3 + x^2 + 2x) + S_2(-x^3 + 3x^2 - 2x) + S_3(x^3 - x) + S_4(x^4 - 2x^3 - x^2 + 2x) =$$

$$2S_0 - (-S_0 + 2S_1 - 2S_2 - S_3 + 2S_4)x + (-2S_0 + S_1 + 3S_2 - S_4)x^2 + (S_0 - S_1 - S_2 + S_3 - 2S_4)x^3 + S_4x^4.$$

Полученный алгоритм содержит 5 умножений и в матрично-векторной форме может быть записан в виде

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \end{pmatrix} = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ -1 & 2 & -2 & -1 & 2 \\ -2 & 1 & 3 & 0 & -1 \\ 1 & -1 & -1 & 1 & -2 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} G_0 & & & & \\ & G_1 & & & \\ & & G_2 & & \\ & & & G_3 & \\ & & & & G_4 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & 2 & 4 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \end{pmatrix}.$$

Предложения можно выполнить, например, следующим образом:

$$\begin{aligned} t_1 &:= d_0 + d_2, \\ D_0 &:= d_0, \\ D_1 &:= t_1 + d_1, \\ D_2 &:= t_1 - d_1, \\ t_2 &:= d_2 + d_2, \\ t_3 &:= d_1 + d_2, \\ D_3 &:= D_1 + t_3 + t_2, \\ D_4 &:= d_2. \end{aligned}$$



Всего получаем 7 операций сложения. Эффективное выполнение постсложений требует большей изобретательности. К примеру, их можно реализовать так:

$$\begin{aligned}
s_0 &:= S_0 + S_0, \\
T_1 &:= S_1 + S_1, \\
T_2 &:= S_2 + S_2, \\
T_3 &:= S_4 + S_4, \\
T_4 &:= T_3 - S_0 - S_3, \\
T_5 &:= S_1 + S_2, \\
s_1 &:= T_1 - T_2 + T_4, \\
s_2 &:= -s_0 + T_2 + T_5 - S_4, \\
s_3 &:= -T_4 - T_5, \\
s_4 &:= S_4.
\end{aligned}$$

Здесь всего 13 сложений. Таким образом, построенный алгоритм содержит всего 20 операций сложения.

Нельзя априори сказать, какой из рассмотренных трех алгоритмов вычисления  $3 \times 3$  - свертки лучше. Это зависит от области применения. Если строить большие алгоритмы из малых, то они часто строятся таким образом, что и число умножений, и число сложений большого алгоритма в основном зависит от числа умножений в малом алгоритме. В такой ситуации число сложений малого алгоритма не играет существенной роли.

Наш алгоритм построен для поля вещественных чисел, но его можно использовать и в поле комплексных чисел. При этом 5 умножений становятся комплексными умножениями, а 20 сложений – 20-ю комплексными сложениями. Однако можно построить специально алгоритм над полем комплексных чисел. Он также будет содержать 5 комплексных умножений, но число сложений будет меньше.

Воспользуемся опять модифицированным алгоритмом Винограда, взяв в качестве модуля многочлен  $m(x) = x(x-1)(x+1)(x-i) = x^4 - ix^3 - x^2 + ix$ . Тогда вычеты равны

$$\begin{aligned}
d_0(x) &= d_0, & g_0(x) &= g_0, \\
d_1(x) &= d_0 + d_1 + d_2, & g_1(x) &= g_0 + g_1 + g_2, \\
d_2(x) &= d_0 - d_1 + d_2, & g_2(x) &= g_0 - g_1 + g_2, \\
d_3(x) &= d_0 + id_1 - d_2, & g_3(x) &= g_0 + ig_1 - g_2,
\end{aligned}$$

откуда

$$\begin{aligned}
s_0(x) &= d_0 g_0, \\
s_1(x) &= (d_0 + d_1 + d_2)(g_0 + g_1 + g_2), \\
s_2(x) &= (d_0 - d_1 + d_2)(g_0 - g_1 + g_2), \\
s_3(x) &= (d_0 + id_1 - d_2)(g_0 + ig_1 - g_2).
\end{aligned}$$

Кроме того, нам потребуется  $s_4(x) = d_2 g_2$ .

Вычислим многочлены  $M_k(x)$ ,  $N_k(x)$  ( $k = 0, 1, 2, 3$ ):

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$
0	$x$	$x^3 - ix^2 - x + i$	$\frac{-i}{2 - 2i} = \frac{1+i}{4}$
1	$x - 1$	$x^3 + (1 - i)x^2 - ix$	$\frac{1}{2 + 2i} = -\frac{1-i}{4}$
2	$x + 1$	$x^3 - (1 + i)x^2 + ix$	$\frac{1}{-2i} = \frac{i}{2}$
3	$x - i$	$x^3 - x$	

(Здесь мы не выписываем многочлены  $n_k(x)$ , так как они не нужны в последующих вычислениях.)



Теперь можно восстановить  $s(x)$  :

$$s(x) = s_0(x)(-i)(x^3 - ix^2 - x + i) + s_1(x)\frac{1+i}{4}(x^3 + (1-i)x^2 - ix) + s_2(x)\left(-\frac{1-i}{4}\right)(x^3 - (1+i)x^2 + ix) + s_3(x)\frac{i}{2}(x^3 - x) + s_4(x)(x^4 - ix^3 - x^2 + ix).$$

Далее, уберем множители, содержащие деление на 4 и на 2, в константы, связанные с многочленом  $g(x)$ . При этом возможны два варианта: либо внести в константы  $g_k(x)$  только множители  $\frac{1}{4}$  и  $\frac{1}{2}$ , либо внести в них также и умножение на комплексные числители полученных дробей. Рассмотрим оба варианта действий.

Пусть

$$G_0 = g_0(x) = g_0, \quad G_1 = \frac{1}{4}g_1(x) = \frac{1}{4}(g_0 + g_1 + g_2), \quad G_2 = \frac{1}{4}g_2(x) = \frac{1}{4}(g_0 - g_1 + g_2),$$

$$G_3 = \frac{1}{2}g_3(x) = \frac{1}{2}(g_0 + ig_1 - g_2), \quad G_4 = g_2,$$

$$D_k = d_k(x) \quad (k = 0, 1, 2, 3), \quad D_4 = d_2$$

$$S_k = D_k G_k \quad (k = 0, 1, \dots, 4),$$

тогда

$$\begin{aligned} s(x) &= S_0(-ix^3 - x^2 + ix + 1) + S_1((1+i)x^3 + 2x^2 + (-i+1)x) + S_2((i-1)x^3 + 2x^2 + (-i-1)x) + \\ &\quad S_3(ix^3 - ix) + S_4(x^4 - ix^3 - x^2 + ix) = \\ &= S_0 + (iS_0 + (1-i)S_1 + (-1-i)S_2 - iS_3 + iS_4)x + (-S_0 + 2S_1 + 2S_2 - S_4)x^2 + \\ &\quad (-iS_0 + (1+i)S_1 + (-1+i)S_2 + iS_3 - iS_4)x^3 + S_4x^4 \end{aligned}$$

Изны получаем алгоритм, который может быть записан в виде

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ i & 1-i & -1-i & -i & i \\ -1 & 2 & 2 & 0 & -1 \\ -i & 1+i & -1+i & i & -i \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} G_0 & & & & \\ & G_1 & & & \\ & & G_2 & & \\ & & & G_3 & \\ & & & & G_4 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & i & -1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \end{pmatrix}.$$

Предложения можно реализовать пятью операциями сложения:

$$\begin{aligned} D_0 &:= d_0, \\ t_1 &:= d_0 + d_2, \\ D_1 &:= t_1 + d_1, \\ D_2 &:= t_1 - d_1, \\ D_3 &:= d_0 + id_1 - d_2, \\ D_4 &:= d_2. \end{aligned}$$

Постсложения реализуются с помощью 10 операций сложения:

$$\begin{aligned} T_1 &:= S_3 - S_4, & s_0 &:= S_0, \\ T_2 &:= S_1 - S_2, & s_1 &:= T_2 - iT_5, \\ T_3 &:= S_1 + S_2, & s_3 &:= T_2 + iT_5, \\ T_4 &:= T_1 - S_0, & s_2 &:= T_3 + T_3 - S_0 - S_4, \\ T_5 &:= T_3 + T_4, & s_4 &:= S_4. \end{aligned}$$



Таким образом, построенный алгоритм требует 5 ~~комплексных~~ умножений и 15 комплексных сложений.

Теперь пусть

$$\tilde{G}_0 = ig_0, \quad \tilde{G}_1 = \frac{1+i}{4}(g_0 + g_1 + g_2), \quad \tilde{G}_2 = \frac{1-i}{4}(g_0 - g_1 + g_2),$$

$$\tilde{G}_3 = \frac{i}{2}(g_0 + ig_1 - g_2), \quad \tilde{G}_4 = g_2.$$

константы  $D_k$  определены как и выше,

$$S_k = D_k \tilde{G}_k \quad (k = 0, 1, \dots, 4),$$

тогда

$$\begin{aligned} s(x) = & S_0(-x^3 + ix^2 + x - i) + S_1(x^3 + (1-i)x^2 - ix) + S_2(-x^3 + (1+i)x^2 - ix) + \\ & S_3(x^3 - x) + S_4(x^4 - ix^3 - x^2 + ix) = \\ & -iS_0 + (S_0 - iS_1 - iS_2 - S_3 + iS_4)x + (iS_0 + (1-i)S_1 + (1+i)S_2 - S_4)x^2 + \\ & (-S_0 + S_1 - S_2 + S_3 - iS_4)x^3 + S_4x^4, \end{aligned}$$

и полученный алгоритм можно записать в виде

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \end{pmatrix} = \begin{pmatrix} -i & 0 & 0 & 0 & 0 \\ 1 & -i & -i & -1 & i \\ i & 1-i & 1+i & 0 & -1 \\ -1 & 1 & -1 & 1 & -i \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \tilde{G}_0 & & & & \\ & \tilde{G}_1 & & & \\ & & \tilde{G}_2 & & \\ & & & \tilde{G}_3 & \\ & & & & \tilde{G}_4 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & -1 & 1 \\ 1 & i & -1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \end{pmatrix}.$$

Число умножений и число предсложений, очевидно, не изменилось. Найдём число постсложений. Их можно реализовать, например, следующей серией сложений:

$$\begin{aligned} T_1 &:= S_1 + S_2, & s_1 &:= T_4 - iT_1, \\ T_2 &:= S_1 - S_2, & s_3 &:= -T_4 + T_2, \\ T_3 &:= S_0 + iS_4, & s_2 &:= T_1 - iT_2 + iT_3, \\ T_4 &:= T_3 - S_3, & s_4 &:= S_4. \\ s_0 &:= -iS_0, \end{aligned}$$

Получили 8 операций сложения. Таким образом, всего в этом случае алгоритм содержит 13 операций сложения.



## §6. ПОСТРОЕНИЕ АЛГОРИТМОВ КОРОТКИХ ЦИКЛИЧЕСКИХ СВЕРТОК

Рассмотрим задачу вычисления циклической свертки

$$s(x) = g(x) d(x) \pmod{x^n - 1}, \quad (1)$$

где  $\deg g(x) = n - 1$ .

Один из способов решения состоит в том, чтобы найти сначала линейную свертку, т. е.  $s_0 = g_0 d_0$ , а затем выполнить приведение по модулю  $x^n - 1$ . Чтобы реализовать такой способ, надо выбрать модуль  $m(x)$  для вычисления линейной свертки так, чтобы его степень была по крайней мере равна  $2n - 1$ , так как она должна быть больше суммы степеней многочленов  $d(x)$  и  $g(x)$ .

Альтернативный способ состоит в использовании китайской теоремы об остатках, причем в качестве модуля  $m(x)$  используется многочлен  $x^n - 1$ , а в качестве взаимно простых модулей  $m_0(x)$ ,  $m_1(x)$ , ...,  $m_{s-1}(x)$  — его сомножители. Так как сумма степеней всех этих сомножителей равна  $n$ , то она меньше, чем  $2n - 1$ , то можно ожидать упрощения вычислений. С другой стороны, делители  $m_k(x)$  надо выбирать произвольно и так, как нам удобно. Они должны быть делителями  $x^n - 1$ . Этот способ допускает выбор произвольных делителей.

Приведем второй способ построения алгоритма циклической свертки на примере 4-точечной свертки, которую будем вычислять над полем вещественных чисел.

$$s(x) = g(x) d(x) \pmod{x^4 - 1},$$

где  $\deg g(x) = \deg d(x) = 3$ , т. е.

$$\begin{aligned} d(x) &= d_0 + d_1 x + d_2 x^2 + d_3 x^3, \\ g(x) &= g_0 + g_1 x + g_2 x^2 + g_3 x^3. \end{aligned}$$

Над полем рациональных чисел  $\mathbb{Q}$  многочлен  $x^4 - 1$  можно разложить на взаимно простые факторы следующим образом:

$$x^4 - 1 = (x - 1)(x + 1)(x^2 + 1) = m_0(x) m_1(x) m_2(x).$$

Теперь воспользуемся алгоритмом Винограда для решения задачи (1). Вычеты  $g(x)$  и  $d(x)$  по модулям  $x - 1$ ,  $x + 1$  и  $x^2 + 1$  соответственно равны

$$\begin{aligned} d_0(x) &= d_0 + d_1 + d_2 + d_3, & g_0(x) &= g_0 + g_1 + g_2 + g_3, \\ d_1(x) &= d_0 - d_1 + d_2 - d_3, & g_1(x) &= g_0 - g_1 + g_2 - g_3, \\ d_2(x) &= (d_0 - d_2) + (d_1 - d_3)x = d_0^{(2)} + d_1^{(2)}x, & g_2(x) &= (g_0 - g_2) + (g_1 - g_3)x = g_0^{(2)} + g_1^{(2)}x. \end{aligned}$$

Найдем  $s_k(x)$  ( $k = 0, 1, 2$ ):

$$\begin{aligned} s_0(x) &= (d_0 + d_1 + d_2 + d_3)(g_0 + g_1 + g_2 + g_3), \\ s_1(x) &= (d_0 - d_1 + d_2 - d_3)(g_0 - g_1 + g_2 - g_3), \\ s_2(x) &= (d_0^{(2)} + d_1^{(2)}x)(g_0^{(2)} + g_1^{(2)}x) \pmod{x^2 + 1} \\ &= (d_0^{(2)}g_0^{(2)} - d_1^{(2)}g_1^{(2)}) + (d_0^{(2)}g_1^{(2)} + d_1^{(2)}g_0^{(2)})x = s_0^{(2)} + s_1^{(2)}x. \end{aligned}$$

Мы уже построили несколько алгоритмов вычисления произведения многочленов по модулю  $x^2 + 1$ , поэтому для нахождения коэффициентов  $s_0^{(2)}$ ,  $s_1^{(2)}$  многочлена  $s_2(x)$  воспользуемся одним из них, например, используем формулы (7) § 4:

$$\begin{pmatrix} s_0^{(2)} \\ s_1^{(2)} \end{pmatrix} = \begin{pmatrix} 1 & 0 & -1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} g_0^{(2)} & & \\ & g_1^{(2)} - g_0^{(2)} & \\ & & g_1^{(2)} + g_0^{(2)} \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} d_0^{(2)} \\ d_1^{(2)} \end{pmatrix}.$$



Для восстановления многочлена  $s(x)$ , построим таблицу многочленов  $m_k(x)$ ,  $M_k(x)$ ,  $N_k(x)$ , ( $k = 0, 1, 2$ ):

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$
0	$x - 1$	$x^3 + x^2 + x - 1$	$\frac{1}{4}$
1	$x + 1$	$x^3 - x^2 + x - 1$	$-\frac{1}{4}$
2	$x^2 + 1$	$x^2 - 1$	$-\frac{1}{2}$

Значит,

$$s(x) = s_0(x) \frac{1}{4} (x^3 + x^2 + x + 1) + s_1(x) \left(-\frac{1}{4}\right) (x^3 - x^2 + x - 1) + (s_0^{(2)} + s_1^{(2)} x) \left(-\frac{1}{2}\right) (x^2 - 1) \pmod{x^4 - 1} =$$

$$\frac{1}{4} s_0(x) (x^3 + x^2 + x + 1) + \frac{1}{4} s_1(x) (-x^3 + x^2 - x + 1) + \frac{1}{2} s_0^{(2)} (1 - x^2) + \frac{1}{2} s_1^{(2)} (x - x^3).$$

Введем обозначения:

$$\begin{aligned} G_0 &= \frac{1}{4} g_0(x) = \frac{1}{4} (g_0 + g_1 + g_2 + g_3), & G_1 &= \frac{1}{4} g_1(x) = \frac{1}{4} (g_0 - g_1 + g_2 - g_3), \\ G_2 &= \frac{1}{2} g_0^{(2)} = \frac{1}{2} (g_0 - g_2), & G_3 &= \frac{1}{2} (g_1^{(2)} - g_0^{(2)}) = \frac{1}{2} (-g_0 + g_1 + g_2 - g_3), \\ G_4 &= \frac{1}{2} (g_1^{(2)} + g_0^{(2)}) = \frac{1}{2} (g_0 + g_1 - g_2 - g_3), \end{aligned}$$

$$\begin{aligned} D_0 &= d_0(x) = d_0 + d_1 + d_2 + d_3, & D_1 &= d_1(x) = d_0 - d_1 + d_2 - d_3, \\ D_2 &= d_0^{(2)} + d_1^{(2)} = d_0 + d_1 - d_2 - d_3, & D_3 &= d_0^{(2)} = d_0 - d_2, \\ D_4 &= d_1^{(2)} = d_1 - d_3, & S_k &= G_k D_k \quad (k = 0, 1, \dots, 4). \end{aligned}$$

Тогда в силу

$$\begin{aligned} s_0^{(2)} &= S_2 - S_4, \\ s_1^{(2)} &= S_2 + S_3, \end{aligned}$$

получаем

$$\begin{aligned} s(x) &= S_0(x^3 + x^2 + x + 1) + S_1(-x^3 + x^2 - x + 1) + (S_2 - S_4)(1 - x^2) + (S_2 + S_3)(x - x^3) = \\ &= (S_0 + S_1 + S_2 - S_4) + (S_0 - S_1 + S_2 + S_3)x + (S_0 + S_1 - S_2 + S_4)x^2 + (S_0 - S_1 - S_2 - S_3)x^3. \end{aligned}$$

В результате полученный алгоритм можно записать в виде

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 0 & -1 \\ 1 & -1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 & 1 \\ 1 & -1 & -1 & -1 & 0 \end{pmatrix} \begin{pmatrix} G_0 & & & & \\ & G_1 & & & \\ & & G_2 & & \\ & & & G_3 & \\ & & & & G_4 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix}.$$

Очевидно, алгоритм содержит 5 вещественных умножений. Число предложений равно 7. Действительно, предложения можно реализовать, например, следующим образом:

$$\begin{aligned} t_1 &:= d_0 + d_2, \\ t_2 &:= d_1 + d_3, \\ D_0 &:= t_1 + t_2, \\ D_1 &:= t_1 - t_2, \\ D_3 &:= d_0 - d_2, \\ D_4 &:= d_1 - d_3, \\ D_2 &:= D_3 + D_4. \end{aligned}$$



Представления можно реализовать с помощью 8 операций сложения:

$$\begin{aligned} T_1 &:= S_0 + S_1, & s_0 &:= T_1 + T_2, \\ T_2 &:= S_2 - S_4, & s_2 &:= T_1 - T_2, \\ T_3 &:= S_2 + S_3, & s_1 &:= T_3 + T_4, \\ T_4 &:= S_0 - S_1, & s_3 &:= -T_3 + T_4. \end{aligned}$$

Таким образом, алгоритм содержит всего 15 вещественных сложений.

Этот алгоритм можно использовать и для поля комплексных чисел  $\mathbb{C}$ , в этом случае все операции становятся комплексными. Однако в данном случае для поля  $\mathbb{C}$  имеет смысл построить свой алгоритм, который будет более эффективным по числу умножений. Это возможно потому, что неприводимый над  $\mathbb{Q}$  многочлен  $x^2 + 1$  над подполем  $\mathbb{Q}(i)$  поля  $\mathbb{C}$  раскладывается на линейные множители  $(x - i)(x + i)$ .

Таким образом,

$$x^4 - 1 = (x - 1)(x + 1)(x - i)(x + i) = m_0(x)m_1(x)m_2(x)m_3(x).$$

Вычеты  $d(x)$  и  $g(x)$  равны

$$\begin{aligned} d_0(x) &= d_0 + d_1 + d_2 + d_3, & g_0(x) &= g_0 + g_1 + g_2 + g_3, \\ d_1(x) &= d_0 - d_1 + d_2 - d_3, & g_1(x) &= g_0 - g_1 + g_2 - g_3, \\ d_2(x) &= d_0 + id_1 - d_2 - id_3, & g_2(x) &= g_0 + ig_1 - g_2 - ig_3, \\ d_3(x) &= d_0 - id_1 - d_2 + id_3, & g_3(x) &= g_0 - ig_1 - g_2 + ig_3, \end{aligned}$$

Заметим, все  $s_k(x) = d_k(x)g_k(x)$  ( $k = 0, 1, 2, 3$ ) являются константами. Далее, находим многочлены  $M_k(x)$  и  $N_k(x)$ :

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$
0	$x - 1$	$x^3 + x^2 + x + 1$	$\frac{1}{4}$
1	$x + 1$	$x^3 - x^2 + x - 1$	$-\frac{1}{4}$
2	$x - i$	$x^3 + ix^2 - x - i$	$\frac{i}{4}$
3	$x + i$	$x^3 - ix^2 - x + i$	$-\frac{i}{4}$

Представляя полученные многочлены в представление  $s(x)$ , по китайской теореме об остатках получим

$$\begin{aligned} s(x) &= s_0(x) \frac{1}{4} (x^3 + x^2 + x + 1) + s_1(x) \left(-\frac{1}{4}\right) (x^3 - x^2 + x - 1) + s_2(x) \frac{i}{4} (x^3 + ix^2 - x - i) \\ &\quad + s_3(x) \left(-\frac{i}{4}\right) (x^3 - ix^2 - x + i). \end{aligned}$$

Введем обозначения

$$\begin{aligned} G_0 &= \frac{1}{4} g_0(x), \quad g_1 = \frac{1}{4} g_1(x), \quad G_2 = \frac{1}{4} g_2(x), \quad G_3 = \frac{1}{4} g_3(x), \\ D_k &= d_k(x) \quad (k = 0, 1, 2, 3), \quad S_k = G_k D_k \quad (k = 0, 1, 2, 3). \end{aligned}$$

Тогда

$$\begin{aligned} s(x) &= S_0 (x^3 + x^2 + x + 1) + S_1 (-x^3 + x^2 - x + 1) + S_2 (ix^3 - x^2 - ix + 1) + S_3 (-ix^3 - x^2 + ix + 1) = \\ &= (S_0 + S_1 + S_2 + S_3) + (S_0 - S_1 - iS_2 + iS_3)x + (S_0 + S_1 - S_2 - S_3)x^2 + (S_0 - S_1 + iS_2 - iS_3)x^3. \end{aligned}$$

Таким образом, полученный алгоритм можно записать в виде

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -i & i \\ 1 & 1 & -1 & -1 \\ 1 & -1 & i & -i \end{pmatrix} \begin{pmatrix} G_0 & & & \\ & G_1 & & \\ & & G_2 & \\ & & & G_3 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \\ 1 & -i & -1 & i \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix}.$$



Этот алгоритм содержит 4 умножения и 16 сложений.

В заключение построим быстрый алгоритм вычисления 6-точечной циклической свертки в поле вещественных чисел.

Пусть

$$s(x) = g(x) d(x) \pmod{x^6 - 1}.$$

где

$$d(x) = d_0 + d_1 x + d_2 x^2 + d_3 x^3 + d_4 x^4 + d_5 x^5,$$

$$g(x) = g_0 + g_1 x + g_2 x^2 + g_3 x^3 + g_4 x^4 + g_5 x^5.$$

Так как над полем  $\mathbb{Q}$

$$x^6 - 1 = (x - 1)(x^2 + x + 1)(x + 1)(x^2 - x + 1) = m_0(x) m_1(x) m_2(x) m_3(x),$$

то задача разбивается на четыре подзадачи:

$$s_k(x) = d_k(x) g_k(x) \pmod{m_k(x)} \quad (k = 0, 1, 2, 3),$$

где  $s_k(x) \equiv s(x) \pmod{m_k(x)}$ .

При  $k = 0, 2$  многочлены  $s_k(x)$  являются константами и вычисление их не представляет затруднений. При  $k = 1, 3$  получаем многочлены первой степени, для отыскания коэффициентов которых придется строить отдельные алгоритмы.

Вычеты многочленов  $d(x)$  и  $g(x)$  по модулям  $x - 1$ ,  $x^2 + x + 1$ ,  $x + 1$ ,  $x^2 - x + 1$  соответственно равны

$$d_0(x) = d_0 + d_1 + d_2 + d_3 + d_4 + d_5,$$

$$d_1(x) = (d_0 - d_2 + d_3 - d_5) + (d_1 - d_2 + d_4 - d_5)x = d_0^{(1)} + d_1^{(1)}x,$$

$$d_2(x) = d_0 - d_1 + d_2 - d_3 + d_4 - d_5,$$

$$d_3(x) = (d_0 - d_2 - d_3 + d_5) + (d_1 + d_2 - d_4 - d_5)x = d_0^{(3)} + d_1^{(3)}x,$$

$$g_0(x) = g_0 + g_1 + g_2 + g_3 + g_4 + g_5,$$

$$g_1(x) = (g_0 - g_2 + g_3 - g_5) + (g_1 - g_2 + g_4 - g_5)x = g_0^{(1)} + g_1^{(1)}x,$$

$$g_2(x) = g_0 - g_1 + g_2 - g_3 + g_4 - g_5,$$

$$g_3(x) = (g_0 - g_2 - g_3 + g_5) + (g_1 + g_2 - g_4 - g_5)x = g_0^{(3)} + g_1^{(3)}x$$

в силу следующих сравнений:

$$x^2 \equiv -x - 1 \pmod{x^2 + x + 1}$$

$$x^2 \equiv x - 1 \pmod{x^2 - x + 1}$$

$$x^3 \equiv -x^2 - x \equiv 1 \pmod{x^2 + x + 1}$$

$$x^3 \equiv x^2 - x \equiv -1 \pmod{x^2 - x + 1}$$

$$x^4 \equiv x \pmod{x^2 + x + 1}$$

$$x^4 \equiv -x \pmod{x^2 - x + 1}$$

$$x^5 \equiv x^2 \equiv -x - 1 \pmod{x^2 + x + 1}$$

$$x^5 \equiv -x^2 \equiv -x + 1 \pmod{x^2 - x + 1}.$$

Находим многочлены  $s_k(x)$  ( $k = 0, 1, 2, 3$ ):

$$s_0(x) = d_0(x) g_0(x) = (d_0 + d_1 + d_2 + d_3 + d_4 + d_5)(g_0 + g_1 + g_2 + g_3 + g_4 + g_5),$$

$$s_1(x) = d_1(x) g_1(x) \pmod{x^2 + x + 1} \equiv s_0^{(1)} + s_1^{(1)}x,$$

$$s_2(x) = d_2(x) g_2(x) = (d_0 - d_1 + d_2 - d_3 + d_4 - d_5)(g_0 - g_1 + g_2 - g_3 + g_4 - g_5),$$

$$s_3(x) = d_3(x) g_3(x) \pmod{x^2 - x + 1} \equiv s_0^{(3)} + s_1^{(3)}x.$$

Найдем многочлены  $M_k(x)$  и  $N_k(x)$ :

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$
0	$x - 1$	$x^5 + x^4 + x^3 + x^2 + x + 1$	$\frac{1}{6}$
1	$x^2 + x + 1$	$x^4 - x^3 + x - 1$	$-\frac{1}{6}(2 + x)$
2	$x + 1$	$x^5 - x^4 + x^3 - x^2 + x - 1$	$-\frac{1}{6}$
3	$x^2 - x + 1$	$x^4 + x^3 - x - 1$	$\frac{1}{6}(x - 2)$

Тогда

$$\begin{aligned}
s_1(x) &= s_0(x) \frac{1}{6} (x^5 + x^4 + x^3 + x^2 + x + 1) + (s_0^{(1)} + s_1^{(1)} x) \left( -\frac{1}{6} (2 + x) \right) (x^4 - x^3 + x - 1) \\
&+ s_2(x) \left( -\frac{1}{6} \right) (x^5 - x^4 + x^3 - x^2 + x - 1) + (s_0^{(3)} + s_1^{(3)} x) \frac{1}{6} (x - 2) (x^4 + x^3 - x - 1) \pmod{x^6 - 1} \\
&= \frac{1}{6} s_0(x) (x^5 + x^4 + x^3 + x^2 + x + 1) + \frac{1}{6} s_0^{(1)} (-x^5 - x^4 + 2x^3 - x^2 - x + 2) \\
&+ \frac{1}{6} s_1^{(1)} (-x^6 - x^5 + 2x^4 - x^3 - x^2 + 2x) + \frac{1}{6} s_2(x) (-x^5 + x^4 - x^3 + x^2 - x + 1) \\
&+ \frac{1}{6} s_0^{(3)} (x^5 - x^4 - 2x^3 - x^2 + x + 2) + \frac{1}{6} s_1^{(3)} (x^6 - x^5 - 2x^4 - x^3 + x^2 + 2x) \pmod{x^6 - 1} \\
&\equiv \frac{1}{6} s_0(x) (x^5 + x^4 + x^3 + x^2 + x + 1) + \frac{1}{6} s_0^{(1)} (-x^5 - x^4 + 2x^3 - x^2 - x + 2) \\
&+ \frac{1}{6} s_1^{(1)} (-x^5 + 2x^4 - x^3 - x^2 + 2x - 1) + \frac{1}{6} s_2(x) (-x^5 + x^4 - x^3 + x^2 - x + 1) \\
&+ \frac{1}{6} s_0^{(3)} (x^5 - x^4 - 2x^3 - x^2 + x + 2) + \frac{1}{6} s_1^{(3)} (-x^5 - 2x^4 - x^3 + x^2 + 2x + 1).
\end{aligned}$$

Переходим к решению подзадачи

$$s_1(x) \equiv d_1(x) g_1(x) \pmod{x^2 + x + 1},$$

где  $d_1(x) = d_0^{(1)} + d_1^{(1)} x$ ,  $g_1(x) = g_0^{(1)} + g_1^{(1)} x$ . Для решения этой подзадачи воспользуемся модифицированным алгоритмом Винограда, выбрав в качестве модуля многочлен  $\tilde{m}(x) = x(x + 1)$ . Сначала мы вычислим линейную свертку

$$\tilde{s}(x) = d_1(x) g_1(x),$$

а затем приведем результат по модулю  $x^2 + x + 1$ . (Последнее действие не содержит умножений.)

Очевидно,

$$\tilde{s}(x) = d_1^{(1)} g_1^{(1)} \tilde{m}(x) + \tilde{r}(x),$$

где  $\tilde{r}(x) \equiv \bar{s}(x) \pmod{\tilde{m}(x)}$ .

Вычеты  $d_1(x)$  и  $g_1(x)$  по модулям  $\tilde{m}_0(x) = x$  и  $\tilde{m}_1(x) = x + 1$  соответственно равны

$$\begin{aligned}
d_{10}(x) &= d_0^{(1)}, & g_{10}(x) &= g_0^{(1)}, \\
d_{11}(x) &= d_0^{(1)} - d_1^{(1)}, & g_{11}(x) &= g_0^{(1)} - g_1^{(1)}.
\end{aligned}$$

Тогда

$$\begin{aligned}
s_{10}(x) &= d_0^{(1)} g_0^{(1)}, \\
s_{11}(x) &= (d_0^{(1)} - d_1^{(1)}) (g_0^{(1)} - g_1^{(1)}).
\end{aligned}$$

Кроме того, нам потребуется  $s_{12}(x) = d_1^{(1)} g_1^{(1)}$ .

Из таблицы

$k$	$\tilde{m}_k(x)$	$\tilde{M}_k(x)$	$\tilde{N}_k(x)$
0	$x$	$x + 1$	1
1	$x + 1$	$x$	-1

следует, что

$$\tilde{s}(x) = s_{10}(x) (x + 1) + s_{11}(x) (-x) + s_{12}(x) (x^2 + x),$$

откуда

$$\begin{aligned}
s_1(x) &\equiv \tilde{s}(x) \pmod{x^2 + x + 1} = s_{10}(x) (x + 1) + s_{11}(x) (-x) - s_{12}(x) \\
&= \{s_{10}(x) - s_{12}(x)\} + \{s_{10}(x) - s_{11}(x)\} x.
\end{aligned}$$



Таким образом,

$$\begin{pmatrix} s_0^{(1)} \\ s_1^{(1)} \end{pmatrix} = \begin{pmatrix} 1 & 0 & -1 \\ 1 & -1 & 0 \end{pmatrix} \begin{pmatrix} g_0^{(1)} & & \\ & g_0^{(1)} - g_1^{(1)} & \\ & & g_1^{(1)} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & -1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} d_0^{(1)} \\ d_1^{(1)} \end{pmatrix}, \quad (2)$$

и решение подзадачи требует трех операций умножения.

Теперь решим подзадачу

$$s_3(x) \equiv d_3(x) g_3(x) \pmod{x^2 - x + 1},$$

где  $d_3(x) = d_0^{(3)} + d_1^{(3)}x$ ,  $g_3(x) = g_0^{(3)} + g_1^{(3)}x$ . Также воспользуемся модифицированным алгоритмом Винограда, вычислив сначала линейную свертку

$$\hat{s}_3(x) = d_3(x) g_3(x),$$

а затем приведем результат по модулю  $x^2 - x + 1$ . Для вычисления линейной свертки выберем модуль  $\hat{m}(x) = x(x-1) = \hat{m}_0(x)\hat{m}_1(x)$ .

Тогда вычеты  $d_3(x)$  и  $g_3(x)$  равны

$$\begin{aligned} d_{30}(x) &= d_0^{(3)}, & g_{30}(x) &= g_0^{(3)}, \\ d_{31}(x) &= d_0^{(3)} + d_1^{(3)}, & g_{31}(x) &= g_0^{(3)} + g_1^{(3)}, \end{aligned}$$

откуда

$$\begin{aligned} s_{30}(x) &= d_0^{(3)} g_0^{(3)}, \\ s_{31}(x) &= (d_0^{(3)} + d_1^{(3)}) (g_0^{(3)} + g_1^{(3)}). \end{aligned}$$

Нам еще потребуется константа  $s_{32}(x) = d_1^{(3)} g_1^{(3)}$ .

Из таблицы

$k$	$\hat{m}_k(x)$	$\hat{M}_k(x)$	$\hat{N}_k(x)$
0	$x$	$x-1$	$-1$
1	$x-1$	$x$	$1$

следует, что

$$\hat{s}(x) = s_{30}(x)(1-x) + s_{31}(x)x + s_{32}(x)(x^2-x),$$

значит,

$$\begin{aligned} s_3(x) &\equiv \hat{s}(x) \pmod{x^2 - x + 1} = s_{30}(x)(1-x) + s_{31}(x)x - s_{32}(x) \\ &= \{s_{30}(x) - s_{32}(x)\} + \{-s_{30}(x) + s_{31}(x)\}x. \end{aligned}$$

В результате

$$\begin{pmatrix} s_0^{(3)} \\ s_1^{(3)} \end{pmatrix} = \begin{pmatrix} 1 & 0 & -1 \\ -1 & 1 & 0 \end{pmatrix} \begin{pmatrix} g_0^{(3)} & & \\ & g_0^{(3)} + g_1^{(3)} & \\ & & g_1^{(3)} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} d_0^{(3)} \\ d_1^{(3)} \end{pmatrix}, \quad (3)$$

т. е. решение этой подзадачи также требует трех умножений.

Осталось собрать все полученные результаты. Введем обозначения:

$$\begin{aligned}
D_0 &= d_0(x) = d_0 + d_1 + d_2 + d_3 + d_4 + d_5, & G_0 &= \frac{1}{6} g_0(x) = \frac{1}{6} (g_0 + g_1 + g_2 + g_3 + g_4 + g_5), \\
D_1 &= d_0^{(1)} = d_0 - d_2 + d_3 - d_5, & G_1 &= \frac{1}{6} g_0^{(1)} = \frac{1}{6} (g_0 - g_2 + g_3 - g_5), \\
D_2 &= d_0^{(1)} - d_1^{(1)} = d_0 - d_1 + d_3 - d_4, & G_2 &= \frac{1}{6} (g_0^{(1)} - g_1^{(1)}) = \frac{1}{6} (g_0 - g_1 + g_3 - g_4), \\
D_3 &= d_1^{(1)} = d_1 - d_2 + d_4 - d_5, & G_3 &= \frac{1}{6} g_1^{(1)} = \frac{1}{6} (g_1 - g_2 + g_4 - g_5), \\
D_4 &= d_2(x) = d_0 - d_1 + d_2 - d_3 + d_4 - d_5, & G_4 &= \frac{1}{6} g_2(x) = \frac{1}{6} (g_0 - g_1 + g_2 - g_3 + g_4 - g_5), \\
D_5 &= d_0^{(3)} = d_0 - d_2 - d_3 + d_5, & G_5 &= \frac{1}{6} g_0^{(3)} = \frac{1}{6} (g_0 - g_2 - g_3 + g_5), \\
D_6 &= d_0^{(3)} + d_1^{(3)} = d_0 + d_1 - d_3 - d_4, & G_6 &= \frac{1}{6} (g_0^{(3)} + g_1^{(3)}) = \frac{1}{6} (g_0 + g_1 - g_3 - g_4), \\
D_7 &= d_1^{(3)} = d_1 + d_2 - d_4 - d_5, & G_7 &= \frac{1}{6} g_1^{(3)} = \frac{1}{6} (g_1 + g_2 - g_4 - g_5), \\
S_k &= D_k G_k \quad (k = 0, 1, \dots, 7),
\end{aligned}$$

Из (2) и (3) имеем

$$\begin{aligned}
s_0^{(1)} &= S_1 - S_3, \\
s_1^{(1)} &= S_1 - S_2, \\
s_0^{(3)} &= S_5 - S_7, \\
s_1^{(3)} &= -S_5 + S_6.
\end{aligned}$$

Теперь можно вернуться к вычислению многочлена  $s(x)$ :

$$\begin{aligned}
s(x) &= S_0 (x^5 + x^4 + x^3 + x^2 + x + 1) + (S_1 - S_3) (-x^5 - x^4 + 2x^3 - x^2 - x + 2) + (S_1 - S_2) (-x^5 + \\
&+ 2x^4 - x^3 - x^2 + 2x - 1) + S_4 (-x^5 + x^4 - x^3 + x^2 - x + 1) + (S_5 - S_7) (x^5 - x^4 - 2x^3 - x^2 + x \\
&+ 2) + (-S_5 + S_6) (-x^5 - 2x^4 - x^3 + x^2 + 2x + 1) = (S_0 + S_1 + S_2 - 2S_3 + S_4 + S_5 + S_6 - 2S_7) + \\
&+ (S_0 + S_1 - 2S_2 + S_3 - S_4 - S_5 + 2S_6 - S_7) x + (S_0 - 2S_1 + S_2 + S_3 + S_4 - 2S_5 + S_6 + S_7) x^2 + \\
&+ (S_0 + S_1 + S_2 - 2S_3 - S_4 - S_5 - S_6 + 2S_7) x^3 + (S_0 + S_1 - 2S_2 + S_3 + S_4 + S_5 - 2S_6 + S_7) x^4 + \\
&+ (S_0 - 2S_1 + S_2 + S_3 - S_4 + 2S_5 - S_6 - S_7) x^5.
\end{aligned}$$

Полученный алгоритм запишем в матрично-векторной форме:

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1-2 & 1 & 1 & 1-2 \\ 1 & 1-2 & 1-1-1 & 2-1 & & \\ 1-2 & 1 & 1 & 1-2 & 1 & 1 \\ 1 & 1 & 1-2-1-1-1 & 2 & & \\ 1 & 1-2 & 1 & 1 & 1-2 & 1 \\ 1-2 & 1 & 1-1 & 2-1-1 & & \end{pmatrix} \begin{pmatrix} G_0 & & & & & \\ & G_1 & & & & \\ & & G_2 & & & \\ & & & G_3 & & \\ & & & & G_4 & \\ & & & & & G_5 \\ & & & & & & G_6 \\ & & & & & & & G_7 \end{pmatrix}.$$

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0-1 & 1 & 0-1 & & \\ 1-1 & 0 & 1-1 & 0 & & \\ 0 & 1-1 & 0 & 1-1 & & \\ 1-1 & 1-1 & 1-1 & & & \\ 1 & 0-1-1 & 0 & 1 & & \\ 1 & 1 & 0-1-1 & 0 & & \\ 0 & 1 & 1 & 0-1-1 & & \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{pmatrix}.$$



Мы получили алгоритм, содержащий 8 вещественных умножений. Предложения можно реализовать следующим образом:

$$\begin{aligned}
t_1 &:= d_0 + d_3, & D_1 &:= t_1 - t_3, \\
t_2 &:= d_1 + d_4, & D_2 &:= t_1 - t_2, \\
t_3 &:= d_2 + d_5, & D_3 &:= D_1 - D_2, \\
t_4 &:= d_0 - d_3, & D_4 &:= t_4 - t_5 + t_6, \\
t_5 &:= d_1 - d_4, & D_5 &:= t_4 - t_6, \\
t_6 &:= d_2 - d_5, & D_6 &:= t_4 + t_5, \\
D_0 &:= t_1 + t_2 + t_3, & D_7 &:= D_6 - D_5.
\end{aligned}$$

Эта последовательность содержит 16 операций сложения. Постсложения можно реализовать за 26 операций, например, так:

$$\begin{aligned}
T_1 &:= S_1 + S_3, & T_{12} &:= T_4 + S_0, \\
T_2 &:= S_1 - S_3, & T_{13} &:= -T_5 + S_0, \\
T_3 &:= T_2 + S_2 - S_3, & T_{14} &:= T_8 + S_4, \\
T_4 &:= T_1 - S_2 - S_2, & T_{15} &:= T_9 - S_4, \\
T_5 &:= T_3 + T_4, & T_{16} &:= T_{10} - S_4, \\
T_6 &:= S_5 + S_7, & s_0 &:= T_{11} + T_{14}, \\
T_7 &:= S_5 - S_7, & s_1 &:= T_{12} + T_{15}, \\
T_8 &:= T_7 + S_6 - S_7, & s_2 &:= T_{13} - T_{16}, \\
T_9 &:= -T_6 + S_6 + S_6, & s_3 &:= T_{11} - T_{14}, \\
T_{10} &:= T_8 - T_9, & s_4 &:= T_{12} - T_{15}, \\
T_{11} &:= T_3 + S_0, & s_5 &:= T_{13} + T_{16}.
\end{aligned}$$

Итак, построенный алгоритм содержит 8 умножений и 42 сложения.

Любой алгоритм свертки представляет собой некоторое тождество, опирающееся на свойство операций в данном поле. Алгоритм, построенный в одном поле, может быть использован в любом расширении этого поля. Но он может оказаться не столь хорошим, как алгоритм, разработанный специально для данного поля.

В частности, алгоритм свертки вещественных последовательностей можно без изменений использовать для свертки последовательностей комплексных чисел. Если  $M$  и  $A$  соответственно число вещественных умножений и сложений, необходимых для вычисления вещественной свертки, и для вычисления комплексной свертки использовать тот же алгоритм с обычным комплексным умножением и сложением, то получится  $4M$  вещественных умножений и  $2A + 2M$  вещественных сложений. Если воспользоваться быстрым комплексным умножением (см. §1) и рассматривать одну из последовательностей как отводы фиксированного фильтра, то потребуется  $3M$  вещественных умножений и  $2A + 3M$  вещественных сложений.

Если длина комплексной циклической свертки равна степени двойки, то можно построить еще более эффективный алгоритм. Рассмотрим свертку по модулю  $x^{2^j} + 1$ . Допустим, требуется вычислить комплексную свертку

$$s(x) = g(x) d(x) \pmod{x^{2^j} + 1},$$

где  $g(x) = g_R(x) + i g_I(x)$  и  $d(x) = d_R(x) + i d_I(x)$ . Конечно, можно вычислить по модулю  $x^{2^j} + 1$  четыре вещественные свертки:

$$g_R(x) d_R(x), \quad g_R(x) d_I(x), \quad g_I(x) d_R(x), \quad g_I(x) d_I(x),$$

а затем вычислить

$$\begin{aligned}
s_R(x) &= g_R(x) d_R(x) - g_I(x) d_I(x), \\
s_I(x) &= g_R(x) d_I(x) + g_I(x) d_R(x)
\end{aligned}$$



Итак, пусть

$$s(x) = s_R(x) + i s_I(x).$$

Однако можно обойтись вычислением всего двух вещественных сверток. В кольце многочленов  $\mathbb{R}[x]$  многочлена  $x^{2^j} + 1$ ,  $j \geq 1$ , выполняется

$$x^{2^j} \equiv -1 \pmod{x^{2^j} + 1},$$

так как  $x^{2^j-1} = \sqrt{-1}$ . Воспользуемся этим элементом, чтобы выписать комплексную свертку через вещественные.

Возьмем многочлены

$$\begin{aligned} u(x) &= \frac{1}{2} \{g_R(x) - x^{2^{j-1}} g_I(x)\} \{d_R(x) - x^{2^{j-1}} d_I(x)\} \pmod{x^{2^j} + 1}, \\ v(x) &= \frac{1}{2} \{g_R(x) + x^{2^{j-1}} g_I(x)\} \{d_R(x) + x^{2^{j-1}} d_I(x)\} \pmod{x^{2^j} + 1}. \end{aligned}$$

Вычисление каждого из них представляет собой вычисление вещественной свертки по модулю  $x^{2^j} + 1$ . Заметим, что

$$u(x) + v(x) = g_R(x)d_R(x) + x^{2^j} g_I(x)d_I(x) \pmod{x^{2^j} + 1} = g_R(x)d_R(x) - g_I(x)d_I(x) = s_R(x),$$

$$u(x) - v(x) = -x^{2^{j-1}} g_I(x)d_R(x) - x^{2^{j-1}} g_R(x)d_I(x) = -x^{2^{j-1}} (g_I(x)d_R(x) + g_R(x)d_I(x)),$$

откуда

$$-x^{2^{j-1}} (u(x) - v(x)) = -x^{2^j} (g_I(x)d_R(x) + g_R(x)d_I(x)) \pmod{x^{2^j} + 1} = g_I(x)d_R(x) + g_R(x)d_I(x) = s_I(x).$$

Воспользуемся этой процедурой для вычисления комплексной циклической свертки

$$s(x) = g(x) d(x) \pmod{x^n - 1},$$

где  $n = 2^k$ . Так как

$$x^n - 1 = (x - 1)(x + 1)(x^2 + 1)(x^{2^2} + 1) \dots (x^{2^{k-1}} + 1),$$

задачу можно свести к вычислению коротких сверток вида

$$s_m(x) = g_m(x) d_m(x) \pmod{x^{2^m} + 1}, \quad m = 0, 1, \dots, k-1.$$

Комплексные свертки по модулям  $x - 1$  и  $x + 1$  являются скалярами и их произведение вычисляется так же, как произведение комплексных чисел. Их вклад в общую сложность очень мал. Если остальные комплексные свертки вычислять через две вещественные свертки, то такой способ вычисления комплексной свертки потребует примерно вдвое больше вычислений, чем вещественная свертка той же длины.

Этот метод соперничает с методом разложения многочлена  $x^{2^k} - 1$  в поле  $\mathbb{C}$ :

$$x^{2^k} - 1 = (x - 1)(x + 1)(x - i)(x + i)(x^2 - i)(x^2 + i)(x^{2^2} - i)(x^{2^2} + i) \dots (x^{2^{k-2}} - i)(x^{2^{k-2}} + i).$$

Каждая из подзадач в данном случае короче, но арифметика является комплексной.



## §7. ТЕОРЕМА ОБ ОБМЕНЕ МАТРИЦ

Алгоритм Винограда можно записать равенством

$$s = C \{ (Bg) \bullet (Ad) \}, \quad (1)$$

где  $\bullet$  означает покомпонентное умножение векторов  $Bg$  и  $Ad$ . Мы также записывали его равенством

$$s = C G A d, \quad (2)$$

где  $G$  – диагональная матрица, и на диагонали стоят компоненты вектора  $Bg$ . Прямой же алгоритм вычисления свертки можно записать в виде

$$s = T d. \quad (3)$$

Таким образом, алгоритм Винограда можно рассматривать как алгоритм разложения матрицы

$$T = C G A,$$

где  $G$  – диагональная матрица, а  $A$  и  $C$  – матрицы, элементами которых являются малые целые числа.

В большинстве приложений многочлену  $g(x)$  соответствует фиксированный КИО-фильтр, или фильтр, коэффициенты которого меняются не слишком часто, поэтому вычисление  $Bg$  приходится выполнять не слишком часто и при подсчете общей вычислительной нагрузки им можно пренебречь. Следовательно, сложность матрицы  $B$  особой роли не играет.

Роли матриц  $A$  и  $B$  симметричны, так как их можно поменять местами путем переименования векторов  $d$  и  $g$ . Алгоритм Винограда вычисления свертки допускает построение матриц  $A$  и  $B$ , отличающихся только перестановкой строк, и, следовательно, имеющих одну и ту же сложность; но в более общем случае более целесообразно одну из матриц строить более сложной и именно ей приписывать роль матрицы  $B$ .

Самое удивительное, что в алгоритме циклической свертки можно менять ролями даже матрицы  $C$  и  $A$ . Это следует из теоремы об обмене матриц, доказательство которой мы сейчас рассмотрим.

Дело в том, что циклическая свертка

$$s(x) = d(x) g(x) \pmod{x^n - 1}$$

формулой (3) может быть записана в виде матричного произведения

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ \vdots \\ s_{n-2} \\ s_{n-1} \end{pmatrix} = \begin{pmatrix} g_0 & g_{n-1} & g_{n-2} & \dots & g_2 & g_1 \\ g_1 & g_0 & g_{n-1} & \dots & g_3 & g_2 \\ g_2 & g_1 & g_0 & \dots & g_4 & g_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ g_{n-2} & g_{n-3} & g_{n-4} & \dots & g_0 & g_{n-1} \\ g_{n-1} & g_{n-2} & g_{n-3} & \dots & g_1 & g_0 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ \vdots \\ d_{n-2} \\ d_{n-1} \end{pmatrix} \quad (s = T d).$$

Матрица  $T$  здесь имеет особый вид. На ее главной диагонали стоят одинаковые элементы, а также на любой диагонали, параллельной главной, стоят одинаковые элементы. Такие матрицы называют *теплицевыми*.

**Теорема (об обмене матриц).** Если теплицева матрица  $D$  допускает факторизацию вида

$$D = C G A,$$

то она может быть также записана в виде

$$D = \tilde{A}^T G^T \hat{C}^T,$$

получается из матрицы  $A$  обращением порядка столбцов, а  $\hat{C}$  получается из матрицы  $C$  обращением порядка строк.

**Лемма 1.** Пусть  $D$  – теплицева матрица порядка  $n$

$$D = \begin{pmatrix} a_1 & a_2 & a_3 & \dots & a_{n-1} & a_n \\ b_2 & a_1 & a_2 & \dots & a_{n-2} & a_{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ b_{n-1} & b_{n-2} & b_{n-3} & \dots & a_1 & a_2 \\ b_n & b_{n-1} & b_{n-2} & \dots & b_2 & a_1 \end{pmatrix}$$

$J$  – обратная матрица того же порядка, т. е.

$$J = \begin{pmatrix} 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & \dots & 1 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 1 & \dots & 0 & 0 \\ 1 & 0 & \dots & 0 & 0 \end{pmatrix}.$$

**Доказательство.**

$$D^T = J D J.$$

Видно, умножение на матрицу  $J$  слева меняет (обращает) порядок строк:

$$\begin{pmatrix} 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & \dots & 1 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 1 & \dots & 0 & 0 \\ 1 & 0 & \dots & 0 & 0 \end{pmatrix} \begin{pmatrix} a_1 & a_2 & a_3 & \dots & a_{n-1} & a_n \\ b_2 & a_1 & a_2 & \dots & a_{n-2} & a_{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ b_{n-1} & b_{n-2} & b_{n-3} & \dots & a_1 & a_2 \\ b_n & b_{n-1} & b_{n-2} & \dots & b_2 & a_1 \end{pmatrix} =$$

$$\begin{pmatrix} b_n & b_{n-1} & b_{n-2} & \dots & b_2 & a_1 \\ b_{n-1} & b_{n-2} & b_{n-3} & \dots & a_1 & a_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ b_2 & a_1 & a_2 & \dots & a_{n-2} & a_{n-1} \\ a_1 & a_2 & a_3 & \dots & a_{n-1} & a_n \end{pmatrix},$$

умножение на матрицу  $J$  справа меняет (обращает) порядок столбцов:

$$\begin{pmatrix} b_n & b_{n-1} & b_{n-2} & \dots & b_2 & a_1 \\ b_{n-1} & b_{n-2} & b_{n-3} & \dots & a_1 & a_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ b_2 & a_1 & a_2 & \dots & a_{n-2} & a_{n-1} \\ a_1 & a_2 & a_3 & \dots & a_{n-1} & a_n \end{pmatrix} \begin{pmatrix} 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & \dots & 1 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 1 & \dots & 0 & 0 \\ 1 & 0 & \dots & 0 & 0 \end{pmatrix} =$$

$$\begin{pmatrix} a_1 & b_2 & \dots & b_{n-2} & b_{n-1} & b_n \\ a_2 & a_1 & \dots & b_{n-3} & b_{n-2} & b_{n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ a_{n-1} & a_{n-2} & \dots & a_2 & a_1 & b_2 \\ a_n & a_{n-1} & \dots & a_3 & a_2 & a_1 \end{pmatrix}.$$

В результате получаем транспонированную матрицу  $D$ . Значит,

$$D^T = J D J = J (C G A) J = (J C) G (A J). \quad (4)$$

Обозначим  $J C$  через  $\hat{C}$ , а  $A J$  – через  $\tilde{A}$ , тогда (4) переписется в виде

$$D^T = \hat{C} G \tilde{A},$$



откуда

$$D = (D^T)^T = (\hat{C} G \tilde{A})^T = \tilde{A}^T G^T \hat{C}^T.$$

Применим эту теорему к умножению многочленов по модулю многочлена  $x^n - \alpha$ , где  $\alpha$  — некоторая константа. Пусть требуется вычислить

$$s(x) = d(x) g(x) \pmod{x^n - \alpha},$$

где  $\deg g(x) \leq n-1$ ,  $\deg d(x) \leq n-1$ .

Если  $\alpha = 1$ , то мы получаем задачу вычисления циклической свертки. Так как

$$\begin{aligned} s(x) &= (g_0 + g_1x + g_2x^2 + \dots + g_{n-1}x^{n-1})(d_0 + d_1x + d_2x^2 + \dots + d_{n-1}x^{n-1}) \pmod{x^n - \alpha} = \\ &= g_0d_0 + (g_1d_0 + g_0d_1)x + (g_2d_0 + g_1d_1 + g_0d_2)x^2 + \dots + (g_{n-1}d_0 + g_{n-2}d_1 + \dots + g_0d_{n-1})x^{n-1} + \\ &\quad (g_{n-1}d_1 + g_{n-2}d_2 + \dots + g_1d_{n-1})x^n + (g_{n-1}d_2 + g_{n-2}d_3 + \dots + g_2d_{n-1})x^{n+1} + \dots + \\ &\quad (g_{n-1}d_{n-2} + g_{n-2}d_{n-1})x^{2n-3} + g_{n-1}d_{n-1}x^{2n-2} \pmod{x^n - \alpha} \equiv (g_0d_0 + \alpha g_{n-1}d_1 + \alpha g_{n-2}d_2 + \dots + \alpha g_1d_{n-1}) \\ &\quad + (g_1d_0 + g_0d_1 + \alpha g_{n-1}d_2 + \alpha g_{n-2}d_3 + \dots + \alpha g_2d_{n-1})x + (g_2d_0 + g_1d_1 + g_0d_2 + \alpha g_{n-1}d_3 + \\ &\quad \alpha g_{n-2}d_4 + \dots + \alpha g_3d_{n-1})x^2 + \dots + (g_{n-1}d_0 + g_{n-2}d_1 + \dots + g_0d_{n-1})x^{n-1}, \end{aligned}$$

то в матричной записи вычисление коэффициентов многочлена  $s(x)$  имеет вид

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ \vdots \\ s_{n-2} \\ s_{n-1} \end{pmatrix} = \begin{pmatrix} g_0 & \alpha g_{n-1} & \alpha g_{n-2} & \dots & \alpha g_2 & \alpha g_1 \\ g_1 & g_0 & \alpha g_{n-1} & \dots & \alpha g_3 & \alpha g_2 \\ g_2 & g_1 & g_0 & \dots & \alpha g_4 & \alpha g_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ g_{n-2} & g_{n-3} & g_{n-4} & \dots & g_0 & \alpha g_{n-1} \\ g_{n-1} & g_{n-2} & g_{n-3} & \dots & g_1 & g_0 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ \vdots \\ d_{n-2} \\ d_{n-1} \end{pmatrix}$$

или  $\mathbf{s} = P\mathbf{d}$ , где  $P$  — теплицева матрица. Поэтому если

$$P = C G A,$$

где  $G$  — диагональная матрица (откуда  $G^T = G$ ), то согласно доказанной теореме

$$P = \tilde{A}^T G \hat{C}^T.$$

А это означает, что можно поменять ролями в алгоритме вычисления циклической свертки матрицы  $A$  и  $C$ . Таким образом, применительно к задаче вычисления циклической свертки теорема об обмене матриц означает, что наиболее сложную из трех матриц  $A$ ,  $B$ ,  $C$  можно выбрать множителем при векторе  $\mathbf{g}$  и "включить" ее в матрицу  $G$ .

## §8. ДИСКРЕТНОЕ ПРЕОБРАЗОВАНИЕ ФУРЬЕ. ТЕОРЕМА О СВЕРТКЕ

Одним из важных видов вычисления в цифровой обработке сигналов является дискретное преобразование Фурье.

**Определение.** Пусть  $\mathbf{v} = \{v_i, i = 0, 1, \dots, n-1\}$  — вектор с вещественными или комплексными компонентами. Дискретным преобразованием Фурье (ДПФ) вектора  $\mathbf{v}$  называется вектор  $\mathbf{V} = \{V_k, k = 0, 1, \dots, n-1\}$  длины  $n$  с комплексными компонентами, задаваемыми формулами

$$V_k = \sum_{j=0}^{n-1} \omega^{jk} v_j, \quad k = 0, 1, \dots, n-1, \quad (1)$$

где  $\omega = e^{-\frac{2\pi i}{n}}$  ( $i$  — мнимая единица).

Вычисление компонент вектора  $\mathbf{V}$  можно записать в матрично-векторном виде

$$\mathbf{V} = W \mathbf{v}$$

$$\begin{pmatrix} V_0 \\ V_1 \\ V_2 \\ \vdots \\ V_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \dots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \dots & \omega^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \dots & \omega^{(n-1)^2} \end{pmatrix} \begin{pmatrix} v_0 \\ v_1 \\ v_2 \\ \dots \\ v_{n-1} \end{pmatrix}. \quad (2)$$

Из равенства  $\omega^n = 1$  нетрудно заметить, что положительные степени  $\omega$  в строке матрицы  $W$ , соответствующей  $k$ -й компоненте выходного вектора ( $k \geq 1$ ), равны степеням  $\omega$  в строке, соответствующей  $(n-k)$ -й компоненте, идущим в обратном порядке. Действительно,

$$\omega^{kj} = \omega^{(n-k)(n-j)}, \quad 1 \leq j \leq n-1.$$

Из выходному вектору  $\mathbf{V}$  можно вычислить компоненты входного вектора. Такое преобразование будем называть обратным преобразованием Фурье. Связь между прямым и обратным преобразованиями дается следующей леммой.

**Лемма.** Пусть вектор  $\mathbf{V}$  равен преобразованию Фурье вектора  $\mathbf{v}$ , тогда

$$v_j = \frac{1}{n} \sum_{k=0}^{n-1} \omega^{-jk} V_k, \quad j = 0, 1, \dots, n-1. \quad (3)$$

**Доказательство.** Преобразуем правую часть формулы (3), подставляя вместо  $V_k$  ее выражение (1):

$$\sum_{k=0}^{n-1} \omega^{-jk} V_k = \sum_{k=0}^{n-1} \omega^{-jk} \sum_{l=0}^{n-1} \omega^{lk} v_l = \sum_{l=0}^{n-1} v_l \left\{ \sum_{k=0}^{n-1} \omega^{k(l-j)} \right\}. \quad (4)$$

Рассмотрим внутреннюю сумму в (4). Если  $j \neq l$ , то

$$\sum_{k=0}^{n-1} \omega^{k(l-j)} = 1 + \omega^{l-j} + \omega^{2(l-j)} + \dots + \omega^{(n-1)(l-j)} = \frac{1 - \omega^{n(l-j)}}{1 - \omega^{l-j}} = 0,$$

так как  $\omega^n = 1$ . Если  $j = l$ , то

$$\sum_{k=0}^{n-1} \omega^{0k} = \sum_{k=0}^{n-1} 1 = n.$$

Следовательно,

$$\sum_{k=0}^{n-1} \omega^{-jk} V_k = n v_j,$$

откуда немедленно следует формула (3).

Между циклической сверткой и преобразованием Фурье имеется важная связь, которая известна как теорема о свертке.



**Теорема.** Вектор  $s$  равен циклической свертке векторов  $f$  и  $g$

$$s_j = \sum_{l=0}^{n-1} g_l f_{((j-l))}, \quad j = 0, 1, \dots, n-1 \quad (5)$$

тогда и только тогда, когда преобразования Фурье  $S$ ,  $G$  и  $F$  этих векторов удовлетворяют равенствам

$$S_k = F_k G_k, \quad k = 0, 1, \dots, n-1.$$

*Доказательство.* По формуле (3) компоненты вектора  $g$  можно выразить через компоненты его преобразования Фурье  $G$ :

$$g_l = \frac{1}{n} \sum_{k=0}^{n-1} \omega^{-lk} G_k, \quad l = 0, 1, \dots, n-1.$$

Подставим это выражение в (5):

$$s_j = \sum_{l=0}^{n-1} g_l f_{((j-l))} = \sum_{l=0}^{n-1} f_{((j-l))} \left\{ \frac{1}{n} \sum_{k=0}^{n-1} \omega^{-kl} G_k \right\} = \frac{1}{n} \sum_{k=0}^{n-1} \omega^{-jk} G_k \left\{ \sum_{l=0}^{n-1} \omega^{(j-l)k} f_{((j-l))} \right\},$$

$j = 0, 1, \dots, n-1$ . Рассмотрим внутреннюю сумму:

$$\begin{aligned} \sum_{l=0}^{n-1} \omega^{(j-l)k} f_{((j-l))} &= \omega^{jk} f_j + \omega^{(j-1)k} f_{j-1} + \dots + \omega^k f_1 + \omega^0 f_0 + \omega^{-k} f_{((-1))} + \omega^{-2k} f_{((-2))} + \dots + \\ &\omega^{(j-n+1)k} f_{((j-n+1))} = \omega^0 f_0 + \omega^k f_1 + \dots + \omega^{(j-1)k} f_{j-1} + \omega^{jk} f_j + \omega^{(j+1)k} f_{j+1} + \dots + \omega^{(n-2)k} f_{n-2} \\ &+ \omega^{(n-1)k} f_{n-1} = \sum_{l=0}^{n-1} \omega^{kl} f_l = F_k. \end{aligned}$$

(Мы воспользовались соотношениями  $\omega^n = 1$  и  $f_{((-l))} = f_{n-l}$  и изменили порядок слагаемых.) Значит,

$$s_j = \frac{1}{n} \sum_{k=0}^{n-1} \omega^{-jk} G_k F_k, \quad j = 0, 1, \dots, n-1.$$

С другой стороны, в силу леммы

$$s_j = \frac{1}{n} \sum_{k=0}^{n-1} \omega^{-jk} S_k, \quad j = 0, 1, \dots, n-1,$$

поэтому

$$\frac{1}{n} \sum_{k=0}^{n-1} \omega^{-jk} G_k F_k = \frac{1}{n} \sum_{k=0}^{n-1} \omega^{-jk} S_k, \quad j = 0, 1, \dots, n-1.$$

Последние равенства можно переписать в виде

$$\frac{1}{n} \sum_{k=0}^{n-1} \omega^{-jk} (G_k F_k - S_k) = 0, \quad j = 0, 1, \dots, n-1. \quad (6)$$

Обозначим через  $E_k$  разность  $G_k F_k - S_k$  ( $k = 0, 1, \dots, n-1$ ), тогда (6) принимает вид

$$\frac{1}{n} \sum_{k=0}^{n-1} \omega^{-jk} E_k = 0, \quad j = 0, 1, \dots, n-1.$$

Равенства можно записать в матричном виде (отбросив множитель  $\frac{1}{n}$ ):

$$\tilde{W} \mathbf{E} = \mathbf{0}.$$

$$\tilde{W} = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega^{-1} & \omega^{-2} & \dots & \omega^{-(n-1)} \\ 1 & \omega^{-2} & \omega^{-4} & \dots & \omega^{-2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{-(n-1)} & \omega^{-2(n-1)} & \dots & \omega^{-(n-1)^2} \end{pmatrix}, \quad \mathbf{E} = \begin{pmatrix} E_0 \\ E_1 \\ E_2 \\ \vdots \\ E_{n-1} \end{pmatrix}.$$

$\tilde{W}$  не вырождена, так как ее определитель есть

$$\det \tilde{W} = \prod_{1 \leq l < j \leq n-1} (\omega^{-j} - \omega^{-l}) = \prod_{1 \leq l < j \leq n-1} \left( e^{\frac{2\pi i j}{n}} - e^{\frac{2\pi i l}{n}} \right) \neq 0.$$

Следовательно,  $\tilde{W}$  обратима, и из  $\tilde{W} \mathbf{E} = \mathbf{0}$  получаем

$$\tilde{W}^{-1}(\tilde{W} \mathbf{E}) = \tilde{W}^{-1} \mathbf{0} = \mathbf{0}$$

$$\mathbf{E} = \mathbf{0},$$

т.е.  $G_k F_k - S_k = 0$  для  $k = 0, 1, \dots, n-1$ .

Обратное утверждение доказывается легко, если повторить проделанные выкладки в обратном направлении.



## §9. ДИСКРЕТНОЕ ПРЕОБРАЗОВАНИЕ ФУРЬЕ И ЦИКЛИЧЕСКАЯ СВЕРТКА

Одним из способов вычисления циклической свертки является метод, использующий теорему о свертке и дискретное преобразование Фурье. Этот метод сводится к выполнению дискретного преобразования Фурье векторов  $\mathbf{d}$  и  $\mathbf{g}$ , поточечному умножению полученных компонент векторов  $\mathbf{D}$  и  $\mathbf{G}$  и обратному преобразованию Фурье, т. е. алгоритм состоит из следующих шагов:

1. Выполнение ДПФ над  $\mathbf{d}$  и  $\mathbf{g}$ :

$$D_k = \sum_{j=0}^{n-1} \omega^{jk} d_j, \quad k = 0, 1, \dots, n-1;$$

$$G_k = \sum_{j=0}^{n-1} \omega^{jk} g_j, \quad k = 0, 1, \dots, n-1;$$

2. Поточечное умножение компонент полученных векторов

$$S_k = D_k G_k, \quad k = 0, 1, \dots, n-1;$$

3. Вычисление компонент циклической свертки с помощью обратного преобразования Фурье

$$s_j = \frac{1}{n} \sum_{k=0}^{n-1} \omega^{-jk} S_k, \quad j = 0, 1, \dots, n-1.$$

Второй шаг содержит  $n$  умножений. Если  $n$  имеет много делителей, то с помощью быстрых алгоритмов преобразования Фурье можно существенно уменьшить число умножений на первом и третьем шагах.

Рассмотренная процедура вычисления циклической свертки относится к комплексному случаю. Естественно ожидать, что в случае вещественных последовательностей этот алгоритм сильнее, чем требуется, и, следовательно, его эффективность можно повысить. Действительно, небольшая модификация алгоритма позволяет вычислять одновременно две вещественные свертки. Рассмотрим эту модификацию.

**Лемма.** Если компоненты вектора  $\mathbf{v}$  являются вещественными числами, то компоненты вектора  $\mathbf{V}$ , являющегося дискретным преобразованием Фурье вектора  $\mathbf{v}$ , обладают следующим свойством:

$$V_k = \overline{V_{n-k}}, \quad k = 0, 1, \dots, n-1.$$

*Доказательство.* По определению ДПФ

$$V_k = \sum_{j=0}^{n-1} \omega^{kj} v_j,$$

откуда

$$\overline{V_{n-k}} = \overline{\left( \sum_{j=0}^{n-1} \omega^{(n-k)j} v_j \right)} = \sum_{j=0}^{n-1} \overline{\omega^{(n-k)j}} \overline{v_j} = \sum_{j=0}^{n-1} \overline{\omega^{(n-k)j}} v_j,$$

поскольку  $\overline{v_j} = v_j$ .

Так как

$$\overline{\omega^{(n-k)j}} = \overline{\omega^{-kj}} = \omega^{kj} \quad (\omega^n = 1),$$

то получаем

$$\overline{V_{n-k}} = \sum_{j=0}^{n-1} \omega^{kj} v_j = V_k,$$

и требуется доказать.

Пусть  $\mathbf{d}^{(1)}$  и  $\mathbf{d}^{(2)}$  – вещественные числовые последовательности длины  $n$ . Определим комплексную последовательность  $\mathbf{d}$  длины  $n$ , компоненты которой задаются следующим образом:

$$d_j = d_j^{(1)} + i d_j^{(2)}, \quad j = 0, 1, \dots, n-1.$$

Дискретное преобразование Фурье последовательности  $\mathbf{d}$  есть

$$D_k = \sum_{j=0}^{n-1} \omega^{jk} d_j = \sum_{j=0}^{n-1} \omega^{jk} (d_j^{(1)} + i d_j^{(2)}) = \sum_{j=0}^{n-1} \omega^{jk} d_j^{(1)} + i \sum_{j=0}^{n-1} \omega^{jk} d_j^{(2)} = D_k^{(1)} + i D_k^{(2)}. \quad (1)$$

$$D_{n-k} = D_{n-k}^{(1)} + i D_{n-k}^{(2)},$$

$$\overline{D}_{n-k} = \overline{D}_{n-k}^{(1)} - i \overline{D}_{n-k}^{(2)}.$$

Поскольку  $\mathbf{d}^{(1)}$  и  $\mathbf{d}^{(2)}$  – вещественные последовательности, значит, в силу леммы имеем

$$\overline{D}_{n-k}^{(1)} = D_k^{(1)},$$

$$\overline{D}_{n-k}^{(2)} = D_k^{(2)},$$

$$\overline{D}_{n-k} = D_k^{(1)} - i D_k^{(2)}. \quad (2)$$

Складывая левые и правые части формул (1) и (2), получаем

$$D_k + \overline{D}_{n-k} = 2D_k^{(1)},$$

вычитая из формулы (1) соответствующие части (2), получаем

$$D_k - \overline{D}_{n-k} = 2i D_k^{(2)},$$

$$\begin{aligned} D_k^{(1)} &= \frac{1}{2} \{D_k + \overline{D}_{n-k}\}, & k &= 0, 1, \dots, n-1, \\ D_k^{(2)} &= \frac{1}{2i} \{D_k - \overline{D}_{n-k}\}, & k &= 0, 1, \dots, n-1. \end{aligned} \quad (3)$$

Эти формулы позволяют вычислить циклические свертки двух пар вещественных последовательностей, выполняя одно дискретное преобразование Фурье и некоторые дополнительные сложения. Рассмотрим этот алгоритм.

Пусть  $\mathbf{g}^{(1)}$  и  $\mathbf{g}^{(2)}$  – также вещественные числовые последовательности длины  $n$ . Аналогично тому, как это сделано для  $\mathbf{d}^{(1)}$  и  $\mathbf{d}^{(2)}$ , определим комплексную последовательность  $\mathbf{g}$ , компоненты которой определяются равенствами

$$g_j = g_j^{(1)} + i g_j^{(2)}, \quad j = 0, 1, \dots, n-1.$$

Для компонент дискретного преобразования Фурье  $\mathbf{G}$  вектора  $\mathbf{g}$  выполняются соотношения (3):

$$G_k^{(1)} = \frac{1}{2} \{G_k + \overline{G}_{n-k}\}, \quad k = 0, 1, \dots, n-1,$$

$$G_k^{(2)} = \frac{1}{2i} \{G_k - \overline{G}_{n-k}\}, \quad k = 0, 1, \dots, n-1.$$



Далее, вычисляем покомпонентные произведения

$$S_k^{(1)} = D_k^{(1)} G_k^{(1)} = \frac{1}{4} \{ D_k + \overline{D}_{n-k} \} \{ G_k + \overline{G}_{n-k} \},$$

$$S_k^{(2)} = D_k^{(2)} G_k^{(2)} = -\frac{1}{4} \{ D_k - \overline{D}_{n-k} \} \{ G_k - \overline{G}_{n-k} \},$$

( $k = 0, 1, \dots, n-1$ ) и формируем последовательность  $S$  длины  $n$  с компонентами

$$S_k = S_k^{(1)} + i S_k^{(2)}, \quad k = 0, 1, \dots, n-1.$$

К последовательности  $S$  применим обратное преобразование Фурье:

$$s_j = \frac{1}{n} \sum_{k=0}^{n-1} \omega^{-jk} S_k = \frac{1}{n} \sum_{k=0}^{n-1} \omega^{-jk} \{ S_k^{(1)} + i S_k^{(2)} \} =$$

$$\frac{1}{n} \sum_{k=0}^{n-1} \omega^{-jk} S_k^{(1)} + i \frac{1}{n} \sum_{k=0}^{n-1} \omega^{-jk} S_k^{(2)} = s_j^{(1)} + i s_j^{(2)},$$

$j = 0, 1, \dots, n-1$ .

Таким образом,

$$s_j^{(1)} = \operatorname{Re}\{s_j\},$$

$$s_j^{(2)} = \operatorname{Im}\{s_j\}, \quad j = 0, 1, \dots, n-1,$$

где  $s^{(1)} = \{s_j^{(1)}, j = 0, 1, \dots, n-1\}$  есть циклическая свертка последовательностей  $\mathbf{d}^{(1)}$  и  $\mathbf{g}^{(1)}$ , а  $s^{(2)} = \{s_j^{(2)}, j = 0, 1, \dots, n-1\}$  — циклическая свертка последовательностей  $\mathbf{d}^{(2)}$  и  $\mathbf{g}^{(2)}$ .

Полученный алгоритм вычисления свертки можно сравнить с алгоритмом Винограда. Для этого оба алгоритма запишем в матричных обозначениях:

Вычисление свертки с  
помощью ДПФ

↓

$$\mathbf{D} = W \mathbf{d}$$

$$\mathbf{G} = W \mathbf{g},$$

где  $W = (w^{ik})$   
матрица  $n \times n$

↓

$$S_k = G_k D_k \quad (k = 0, 1, \dots, n-1)$$

$$\mathbf{s} = W^{-1} \mathbf{S}$$

↓

Выход свертка ( $\bmod x^n - 1$ )

Вычисление свертки с  
помощью алгоритма Винограда

↓

$$\mathbf{D} = A \mathbf{d}$$

$$\mathbf{G} = B \mathbf{g},$$

где  $A$  и  $B$  — матрицы  
размера  $M(n) \times n$  содержат  
малые целые числа

↓

$$S_k = G_k D_k \quad (k = 0, 1, \dots, M(n)-1)$$

↓

$$\mathbf{s} = C \mathbf{S},$$

где  $C$  — матрица  $n \times M(n)$   
содержит малые целые числа

↓

Выход свертка ( $\bmod m(x)$ )

Оба алгоритма записываются в виде матричных равенств соответственно

$$\mathbf{s} = W^{-1} \{W \mathbf{g} \bullet W \mathbf{d}\}$$

и

$$\mathbf{s} = C \{B \mathbf{g} \bullet A \mathbf{d}\},$$

где  $\bullet$  обозначает покомпонентное произведение векторов. Отсюда видно, что алгоритм Винограда является обобщением метода вычисления свертки с помощью преобразования Фурье.

## §10. ДИСКРЕТНОЕ ПРЕОБРАЗОВАНИЕ ФУРЬЕ И МНОГОЧЛЕНЫ

На дискретное преобразование Фурье можно посмотреть с точки зрения многочленов. Поставим в соответствие входному вектору  $\mathbf{v} = \{v_0, v_1, \dots, v_{n-1}\}$  многочлен

$$f(x) = \sum_{j=0}^{n-1} v_j x^j.$$

Тогда ДПФ соответствует вычислению значений многочлена  $f(x)$  в точках  $\omega^k$ ,  $0 \leq k \leq n-1$ :

$$V_k = f(\omega^k) = \sum_{j=0}^{n-1} \omega^{kj} v_j,$$

а обратное преобразование

$$v_j = \frac{1}{n} \sum_{k=0}^{n-1} V_k \omega^{-jk}, \quad 0 \leq j \leq n-1$$

– интерполяции многочлена  $f(x)$  по его значениям в точках  $\omega^k$ ,  $0 \leq k \leq n-1$ . То, что точки  $\{\omega^k\}_{k=0}^{n-1}$  образуют циклическую мультипликативную группу, и позволяет построить быстрый алгоритм вычисления значения как прямого, так и обратного ДПФ.

Значение многочлена  $f(x)$  в точке  $x = \omega^k$  совпадает с остатком от деления многочлена  $f(x)$  на многочлен  $x - \omega^k$ ,  $0 \leq k \leq n-1$ , т. е. с вычетом  $f(x) \pmod{x - \omega^k}$ . При этом многочлены  $x - \omega^k$ ,  $k = 0, 1, \dots, n-1$  попарно взаимно просты и их произведение равно  $x^n - 1$ . (Множество чисел  $\omega^k$ ,  $k = 0, 1, \dots, n-1$  представляет собой набор всех корней  $n$ -й степени из 1.) Поэтому в случае  $n = 2^m$  можно применить следующий подход для вычисления  $V_k$ ,  $k = 0, 1, \dots, n-1$ .

Поскольку  $\omega^{\frac{n}{2}} = -1$ , то

$$x^n - 1 = (x^{\frac{n}{2}} - 1)(x^{\frac{n}{2}} + \omega^{\frac{n}{2}}).$$

Далее,

$$\begin{aligned} x^{\frac{n}{2}} - 1 &= (x^{\frac{n}{4}} - 1)(x^{\frac{n}{4}} + \omega^{\frac{n}{2}}), \\ x^{\frac{n}{2}} + \omega^{\frac{n}{2}} &= (x^{\frac{n}{4}} - \omega^{\frac{n}{4}})(x^{\frac{n}{4}} + \omega^{\frac{n}{4}}) = (x^{\frac{n}{4}} - \omega^{\frac{n}{4}})(x^{\frac{n}{4}} - \omega^{\frac{3n}{4}}). \end{aligned}$$

Продолжаем разложение каждой разности двух квадратов:

$$\begin{aligned} x^{\frac{n}{4}} - 1 &= (x^{\frac{n}{8}} - 1)(x^{\frac{n}{8}} + \omega^{\frac{n}{2}}), \\ x^{\frac{n}{4}} - \omega^{\frac{n}{2}} &= (x^{\frac{n}{8}} - \omega^{\frac{n}{4}})(x^{\frac{n}{8}} + \omega^{\frac{3n}{4}}), \\ x^{\frac{n}{4}} - \omega^{\frac{n}{4}} &= (x^{\frac{n}{8}} - \omega^{\frac{n}{8}})(x^{\frac{n}{8}} + \omega^{\frac{5n}{4}}), \\ x^{\frac{n}{4}} - \omega^{\frac{3n}{4}} &= (x^{\frac{n}{8}} - \omega^{\frac{3n}{8}})(x^{\frac{n}{8}} + \omega^{\frac{7n}{4}}) \end{aligned}$$

и так далее до получения линейных сомножителей  $x - \omega^k$ ,  $k = 0, 1, \dots, n-1$ .

Нахождение остатков от деления многочлена  $f(x)$  на  $x - \omega^k$  будем проводить с использованием техники "разделяй и властвуй" путем деления  $f(x)$  сначала на многочлены  $x^{\frac{n}{2}} - 1$  и  $x^{\frac{n}{2}} - \omega^{\frac{n}{2}}$ :

$$f_{11}(x) \equiv f(x) \pmod{x^{\frac{n}{2}} - 1}, \quad f_{12}(x) \equiv f(x) \pmod{x^{\frac{n}{2}} - \omega^{\frac{n}{2}}},$$

затем каждый из двух полученных остатков два раза делим на многочлены вида  $x^{\frac{n}{4}} - \omega^{\frac{jn}{4}}$ :

$$\begin{aligned} f_{21}(x) &\equiv f_{11}(x) \pmod{x^{\frac{n}{4}} - 1}, & f_{23}(x) &\equiv f_{12}(x) \pmod{x^{\frac{n}{4}} - \omega^{\frac{n}{4}}}, \\ f_{22}(x) &\equiv f_{11}(x) \pmod{x^{\frac{n}{4}} - \omega^{\frac{n}{2}}}, & f_{24}(x) &\equiv f_{12}(x) \pmod{x^{\frac{n}{4}} - \omega^{\frac{3n}{4}}}, \end{aligned}$$



далее, каждый из четырех полученных остатков два раза делим на многочлены вида  $x^{\frac{n}{8}} - \omega^{\frac{jn}{8}}$ :

$$\begin{aligned} f_{31}(x) &\equiv f_{21}(x) \pmod{x^{\frac{n}{8}} - 1}, & f_{32}(x) &\equiv f_{21}(x) \pmod{x^{\frac{n}{8}} - \omega^{\frac{n}{2}}}, \\ f_{33}(x) &\equiv f_{22}(x) \pmod{x^{\frac{n}{8}} - \omega^{\frac{n}{4}}}, & f_{34}(x) &\equiv f_{22}(x) \pmod{x^{\frac{n}{8}} - \omega^{\frac{3n}{4}}}, \\ f_{35}(x) &\equiv f_{23}(x) \pmod{x^{\frac{n}{8}} - \omega^{\frac{n}{8}}}, & f_{36}(x) &\equiv f_{23}(x) \pmod{x^{\frac{n}{8}} - \omega^{\frac{5n}{4}}}, \\ f_{37}(x) &\equiv f_{24}(x) \pmod{x^{\frac{n}{8}} - \omega^{\frac{3n}{8}}}, & f_{38}(x) &\equiv f_{24}(x) \pmod{x^{\frac{n}{8}} - \omega^{\frac{7n}{8}}}, \end{aligned}$$

и так далее, пока на  $m$ -м шаге не дойдем до вычетов по модулям  $x - \omega^k$ ,  $k = 0, 1, \dots, n-1$ :

$$\begin{aligned} f_{m1}(x) &\equiv f_{m-11}(x) \pmod{x-1}, & f_{m2}(x) &\equiv f_{m-11}(x) \pmod{x-\omega^{\frac{n}{2}}}, \\ f_{m3}(x) &\equiv f_{m-12}(x) \pmod{x-\omega}, & f_{m4}(x) &\equiv f_{m-12}(x) \pmod{x-\omega^{\frac{n}{2}+1}}, \\ &\dots\dots\dots & &\dots\dots\dots \\ f_{mn-1}(x) &\equiv f_{m-1\frac{n}{2}}(x) \pmod{x-\omega^{\frac{n}{2}-1}}, & f_{mn}(x) &\equiv f_{m-1\frac{n}{2}}(x) \pmod{x-\omega^{n-1}}. \end{aligned}$$

При этом, как будет видно из следующей леммы, вычисление вычета многочлена степени  $k-1$  по модулю многочлена  $x^{\frac{k}{2}} - c$  провести очень просто, выполнив  $\frac{k}{2}$  умножений и столько же сложений коэффициентов этого многочлена.

**Лемма.** Пусть

$$f(x) = \sum_{j=0}^{k-1} a_j x^j$$

— многочлен из кольца  $K[x]$ ,  $c \in K$  ( $K$  — кольцо). Тогда остаток от деления  $f(x)$  на  $x^{\frac{k}{2}} - c$  равен

$$r(x) = \sum_{j=0}^{\frac{k}{2}-1} (a_j + ca_{j+\frac{k}{2}}) x^j.$$

**Доказательство.** Поскольку  $x^{\frac{k}{2}} \equiv c \pmod{x^{\frac{k}{2}} - c}$ , то

$$\begin{aligned} f(x) &= a_{k-1} x^{k-1} + a_{k-2} x^{k-2} + \dots + a_{\frac{k}{2}} x^{\frac{k}{2}} + a_{\frac{k}{2}-1} x^{\frac{k}{2}-1} + \dots + a_1 x^1 + a_0 \pmod{x^{\frac{k}{2}} - c} \equiv \\ &a_{k-1} c x^{\frac{k}{2}-1} + a_{k-2} c x^{\frac{k}{2}-2} + \dots + a_{\frac{k}{2}+1} c x + a_{\frac{k}{2}} c + a_{\frac{k}{2}-1} x^{\frac{k}{2}-1} + \dots + a_1 x + a_0 = \\ &(a_{\frac{k}{2}-1} + ca_{k-1}) x^{\frac{k}{2}-1} + (a_{\frac{k}{2}-2} + ca_{k-2}) x^{\frac{k}{2}-2} + \dots + (a_1 + ca_{\frac{k}{2}+1}) x + (a_0 + ca_{\frac{k}{2}}). \end{aligned}$$

Итак,

$$r(x) = \sum_{j=0}^{\frac{k}{2}-1} (a_j + ca_{j+\frac{k}{2}}) x^j.$$

В результате для вычисления всех значений  $f(\omega^k)$ ,  $k = 0, 1, \dots, n-1$  потребуется

$$2 \left( \frac{n}{2} + 2 \frac{n}{4} + 2^2 \frac{n}{8} + \dots + 2^{m-1} \frac{n}{2^m} \right) = nm,$$

где  $m = \log_2 n$ , умножений. Таким образом, сложность предложенного алгоритма быстрого преобразования Фурье равна  $O(n \log_2 n)$ .

Благодаря алгоритму БПФ дискретное преобразование Фурье является удобным инструментом при проведении вычислений с многочленами. С его помощью можно вычислять произведение двух многочленов со сложностью  $O(n \log_2 n)$ , выполнив сначала преобразование Фурье и получив значения этих многочленов в точках, затем перемножить полученные значения ( $n$  умножений) в

на этой точке и восстановить коэффициенты многочлена-произведения с помощью обратного преобразования Фурье, которое, очевидно, может быть выполнено по той же схеме, а значит, имеет ту же сложность.

Эта же идея лежит в основе быстрого битового умножения больших целых чисел. Соответствующий алгоритм известен как алгоритм Шёнхаге – Штрассена. Идея этого алгоритма состоит в следующем: каждое из перемножаемых чисел в двоичном его представлении разбивается на  $b$  блоков по  $b$  разрядов в каждом блоке. Далее, каждый блок рассматривается как соответствующий коэффициент многочлена. Чтобы определить коэффициенты произведения этих многочленов, вычисляются значения на некотором подходящем множестве точек; найденные значения перемножаются и затем интерполируется многочлен-произведение, блоки коэффициентов которого позволяют получить двоичное представление произведения двух исходных чисел. Выбрав в качестве точек, в которых вычисляются значения многочленов, корни из 1, можно воспользоваться преобразованием Фурье и теоремой о свертке. Если сделать число  $b$  функцией от  $n$  и применить рекурсию, то можно умножить два  $n$ -разрядных двоичных числа за  $O(n \log n \log \log n)$  шагов. И хотя алгоритм Шёнхаге – Штрассена в реальности не очень эффективен, он весьма полезен для целей теории.



## §11. Алгоритм Кули – Тьюки

Основных стратегий построения быстрых алгоритмов преобразования Фурье две. Одна состоит в сведении дискретного преобразования Фурье к свертке, которая затем вычисляется одним из быстрых алгоритмов вычисления свертки. Другая стратегия состоит в переходе от одномерного преобразования Фурье к двумерному (или многомерному), которое вычисляется проще. Хорошие алгоритмы вычисления ДПФ используют обе эти стратегии. Алгоритм Кули – Тьюки – это алгоритм второго типа.

Прямое вычисление компонент вектора  $V$ , являющегося дискретным преобразованием Фурье вектора  $v$ , требует  $n^2$  умножений и  $n(n-1)$  операций сложения. Если  $n$  – составное число, то можно перейти от одномерного преобразования

$$V_k = \sum_{j=0}^{n-1} \omega^{jk} v_j, \quad k = 0, 1, \dots, n-1 \quad (1)$$

к двумерному. Такие алгоритмы известны под общим названием *быстрого преобразования Фурье (БПФ-алгоритмы)*. Алгоритм, который мы рассмотрим здесь, и является одним из них.

Предположим, что  $n = n' n''$ . Перейдем в формуле (1) к двойным индексам, заменяя каждый из индексов  $k$  и  $j$  на два следующим образом:

$$\begin{aligned} j &= j' + n' j'', & j' &= 0, 1, \dots, n' - 1; \quad j'' = 0, 1, \dots, n'' - 1; \\ k &= n'' k' + k'', & k' &= 0, 1, \dots, n' - 1; \quad k'' = 0, 1, \dots, n'' - 1. \end{aligned}$$

Тогда (1) принимает вид

$$V_{n'' k' + k''} = \sum_{j'=0}^{n'-1} \sum_{j''=0}^{n''-1} \omega^{(j'+n'j'')(n''k'+k'')} v_{j'+n'j''}.$$

Так как  $\omega^n = 1$  ( $n = n' n''$ ), то

$$\omega^{(j'+n'j'')(n''k'+k'')} = \omega^{n''j'k' + n'j''k'' + j'k''} = (\omega^{n''})^{j'k'} (\omega^{n'})^{j''k''} \omega^{j'k''}.$$

Введем обозначения:

$$\omega^{n''} = \beta, \quad \omega^{n'} = \gamma,$$

тогда

$$V_{n'' k' + k''} = \sum_{j'=0}^{n'-1} \sum_{j''=0}^{n''-1} \beta^{j'k'} \gamma^{j''k''} \omega^{j'k''} v_{j'+n'j''}. \quad (2)$$

Определим теперь двумерные переменные равенствами:

$$v_{j' j''} = v_{j'+n'j''} \quad (j' = 0, 1, \dots, n' - 1; \quad j'' = 0, 1, \dots, n'' - 1),$$

$$V_{k' k''} = V_{n'' k' + k''} \quad (k' = 0, 1, \dots, n' - 1; \quad k'' = 0, 1, \dots, n'' - 1),$$

тогда (2) принимает вид

$$V_{k' k''} = \sum_{j'=0}^{n'-1} \beta^{j'k'} \left\{ \omega^{j'k''} \sum_{j''=0}^{n''-1} \gamma^{j''k''} v_{j' j''} \right\} \quad (3)$$

$k' = 0, 1, \dots, n' - 1; \quad k'' = 0, 1, \dots, n'' - 1.$

Эта формула сложнее, чем формула (1), но число сложений и умножений, входящих в нее, существенно меньше чем в формуле (1).

На самом деле, произошло следующее: когда мы ввели двумерные переменные, мы фактически преобразовали векторы входных и выходных данных в двумерные массивы. При этом компоненты преобразования  $V$  упорядочены не так, как компоненты входного сигнала  $v$ .

Компоненты входного вектора  $v$  упорядочены по столбцам:

$$\begin{pmatrix} v_{00} & v_{01} & \dots & v_{0n''-1} \\ v_{10} & v_{11} & \dots & v_{1n''-1} \\ v_{20} & v_{21} & \dots & v_{2n''-1} \\ \dots & \dots & \dots & \dots \\ v_{n'-10} & v_{n'-11} & \dots & v_{n'-1n''-1} \end{pmatrix} = \begin{pmatrix} v_0 & v_{n'} & v_{2n'} & \dots & v_{(n''-1)n'} \\ v_1 & v_{n'+1} & v_{2n'+1} & \dots & v_{(n''-1)n'+1} \\ v_2 & v_{n'+2} & v_{2n'+2} & \dots & v_{(n''-1)n'+2} \\ \dots & \dots & \dots & \dots & \dots \\ v_{n'-1} & v_{2n'-1} & v_{3n'-1} & \dots & v_{n'n''-1} \end{pmatrix}.$$

Далее, поскольку

$$\gamma = \omega^{n'} = \left( e^{-\frac{2i\pi}{n}} \right)^{n'} = e^{-\frac{2i\pi}{n''}},$$

$$\sum_{j''=0}^{n''-1} \gamma^{j''k''} v_{j'j''} = u_{j'k''} \quad (j' = 0, 1, \dots, n' - 1; k'' = 0, 1, \dots, n'' - 1)$$

это  $n''$ -точечное преобразование Фурье элементов  $j'$ -й строки полученной таблицы. Следовательно, на каждой строке матрицы, полученной из вектора  $v$ , проводится  $n''$ -точечное преобразование Фурье. В результате получаем новую матрицу

$$\begin{pmatrix} u_{00} & u_{01} & \dots & u_{0n''-1} \\ u_{10} & u_{11} & \dots & u_{1n''-1} \\ u_{20} & u_{21} & \dots & u_{2n''-1} \\ \dots & \dots & \dots & \dots \\ u_{n'-10} & u_{n'-11} & \dots & u_{n'-1n''-1} \end{pmatrix},$$

каждый элемент которой следует умножить на соответствующий множитель  $\omega^{j'k''}$ . Далее, поскольку

$$\beta = \omega^{n''} = \left( e^{-\frac{2i\pi}{n}} \right)^{n''} = e^{-\frac{2i\pi}{n'}},$$

$$\sum_{j'=0}^{n'-1} \beta^{j'k'} \left\{ \omega^{j'k''} u_{j'k''} \right\} \quad (k' = 0, 1, \dots, n' - 1; k'' = 0, 1, \dots, n'' - 1)$$

это  $n'$ -точечное преобразование Фурье каждого столбца полученной на предыдущем шаге матрицы.

В результате получаем таблицу, содержащую компоненты выходного вектора  $V$ , которые упорядочены по строкам:

$$\begin{pmatrix} V_{00} & V_{01} & \dots & V_{0n''-1} \\ V_{10} & V_{11} & \dots & V_{1n''-1} \\ V_{20} & V_{21} & \dots & V_{2n''-1} \\ \dots & \dots & \dots & \dots \\ V_{n'-10} & V_{n'-11} & \dots & V_{n'-1n''-1} \end{pmatrix} = \begin{pmatrix} V_0 & V_1 & V_2 & \dots & V_{n''-1} \\ V_{n''} & V_{n''+1} & V_{n''+2} & \dots & V_{2n''-1} \\ V_{2n''} & V_{2n''+1} & V_{2n''+2} & \dots & V_{3n''-1} \\ \dots & \dots & \dots & \dots & \dots \\ V_{(n'-1)n''} & V_{(n'-1)n''+1} & V_{(n'-1)n''+2} & \dots & V_{n'n''-1} \end{pmatrix}.$$

Если внутреннее и внешнее преобразования Фурье выполнять в соответствии с прямыми алгоритмами, то для  $n'$ -точечного преобразования Фурье требуется  $(n')^2$  умножений и  $n'(n' - 1)$  сложений, для  $n''$ -точечного —  $(n'')^2$  умножений и  $n''(n'' - 1)$  сложений.  $n''$ -точечное преобразование Фурье используется для каждой строки таблицы, т. е.  $n'$  раз, а  $n'$ -точечное преобразование Фурье используется для каждого столбца, т. е.  $n''$  раз, значит, число умножений равно

$$n'(n'')^2 + n''(n')^2 + n'n'',$$

так как имеется еще  $n = n'n''$  поточечных умножений на множители  $\omega^{j'k''}$ .



Число сложений равно

$$n'n''(n'' - 1) + n''n'(n' - 1).$$

Таким образом, получаем полное число (комплексных) умножений  $M_{\mathbf{C}}(n)$  и сложений  $A_{\mathbf{C}}(n)$  :

$$M_{\mathbf{C}}(n) = n'(n'')^2 + n''(n')^2 + n'n'' = n'n''(n'' + n' + 1) = n(n' + n'' + 1),$$

$$A_{\mathbf{C}}(n) = n'n''(n'' - 1) + n''n'(n' - 1) = n'n''(n' + n'' - 2) = n(n' + n'' - 2).$$

Среди умножений имеются и тривиальные умножения на 1. Это происходит каждый раз, когда в множителе  $\omega^{j'k''}$  в показателе степени  $j'$  или  $k''$  обращаются в 0. Алгоритм можно изменить так, чтобы выбросить эти умножения, тогда число комплексных умножений станет равным

$$\begin{aligned} M_{\mathbf{C}}(n) &= n'(n'')^2 + n''(n')^2 + (n' - 1)(n'' - 1) = n(n' + n'') + (n'n'' - n' - n'' + 1) = \\ &= (n - 1)(n' + n'') + (n'n'' + 1) = (n' + n'')(n - 1) + (n + 1). \end{aligned}$$

Внешнее и внутреннее преобразования Фурье в свою очередь могут быть вычислены с помощью быстрых алгоритмов, не обязательно с помощью БПФ-алгоритма Кули – Тьюки. Если  $n'$  или  $n''$  опять является составным числом, то соответствующее преобразование можно вычислять с помощью БПФ-алгоритма Кули – Тьюки. На этом пути преобразование длины  $n = n_1 n_2 \dots n_s$  может быть представлено в такой форме, в которой потребуется выполнить  $n(n_1 + n_2 + \dots + n_s)$  комплексных умножений.

## §12. Алгоритм Кули – Тьюки по основанию два

Во многих приложениях длина преобразования равна степени двойки. Для построения БПФ-алгоритма длины  $n = 2^m$  число  $n$  представляется в виде  $2 \cdot 2^{m-1}$  или  $2^{m-1} \cdot 2$ . В этом случае говорят об алгоритме Кули – Тьюки по основанию два.

Если  $n' = 2$ ,  $n'' = 2^{m-1}$ , то БПФ-алгоритм называется *алгоритмом Кули – Тьюки по основанию два с прореживанием по времени*. Если  $n' = 2^{m-1}$ ,  $n'' = 2$ , то он называется *алгоритмом Кули – Тьюки по основанию два с прореживанием по частоте*.

Выведем формулы БПФ-алгоритма Кули – Тьюки с прореживанием по времени. Так как  $n' = 2$ ,  $n'' = 2^{m-1} = \frac{n}{2}$ , то формула (2) §10 принимает вид

$$V_{n''k'+k''} = \sum_{j'=0}^1 \sum_{j''=0}^{\frac{n}{2}-1} \beta^{j'k'} \gamma^{j''k''} \omega^{j'k''} v_{j'+n'j''},$$

где  $\beta = \omega^{\frac{n}{2}} = -1$ ,  $\gamma = \omega^2$ . Итак, расписывая внешнюю сумму, получаем

$$V_{n''k'+k''} = \sum_{j'=0}^1 \sum_{j''=0}^{\frac{n}{2}-1} (-1)^{j'k'} \omega^{2j''k''} \omega^{j'k''} v_{j'+2j''} = \sum_{j''=0}^{\frac{n}{2}-1} \omega^{2j''k''} v_{2j''} + (-1)^{k'} \omega^{k''} \sum_{j''=0}^{\frac{n}{2}-1} \omega^{2j''k''} v_{2j''+1} \quad (1)$$

где  $k' = 0, 1$ ;  $k'' = 0, 1, \dots, \frac{n}{2} - 1$ .

При  $k' = 0$  получаем

$$V_{k''} = \sum_{j''=0}^{\frac{n}{2}-1} \omega^{2j''k''} v_{2j''} + \omega^{k''} \sum_{j''=0}^{\frac{n}{2}-1} \omega^{2j''k''} v_{2j''+1} \quad (k'' = 0, 1, \dots, \frac{n}{2} - 1), \quad (2)$$

при  $k' = 1$  –

$$V_{k''+\frac{n}{2}} = \sum_{j''=0}^{\frac{n}{2}-1} \omega^{2j''k''} v_{2j''} - \omega^{k''} \sum_{j''=0}^{\frac{n}{2}-1} \omega^{2j''k''} v_{2j''+1} \quad (k'' = 0, 1, \dots, \frac{n}{2} - 1). \quad (3)$$

Прореживание по времени разбивает множество компонент входного вектора на два подмножества: множество компонент с четными индексами и множество компонент с нечетными индексами. Множество компонент выходного вектора разбивается на множество первых  $\frac{n}{2}$  компонент и множество последних  $\frac{n}{2}$  компонент.

В качестве примера рассмотрим 8-точечное преобразование Фурье. Тогда формулы (2) и (3) принимают вид

$$\begin{aligned} V_k &= \sum_{j=0}^3 \omega^{2jk} v_{2j} + \omega^k \sum_{j=0}^3 \omega^{2jk} v_{2j+1} \quad (k = 0, 1, 2, 3), \\ V_{k+4} &= \sum_{j=0}^3 \omega^{2jk} v_{2j} - \omega^k \sum_{j=0}^3 \omega^{2jk} v_{2j+1} \quad (k = 0, 1, 2, 3). \end{aligned} \quad (4)$$

Достаточно вычислить

$$V_{k1} = \sum_{j=0}^3 \omega^{2jk} v_{2j} \quad \text{и} \quad V_{k2} = \sum_{j=0}^3 \omega^{2jk} v_{2j+1} \quad (k = 0, 1, 2, 3),$$

затем умножить каждое  $V_{k2}$  на  $\omega^k$  и сложить или вычесть полученные суммы согласно формулам (4).

$$\begin{aligned} V_k &= V_{k1} + \omega^k V_{k2} \quad (k = 0, 1, 2, 3), \\ V_{k+4} &= V_{k1} - \omega^k V_{k2} \quad (k = 0, 1, 2, 3). \end{aligned} \quad (5)$$



Для вычисления  $V_{k1}$  ( $V_{k2}$ ) воспользуемся снова формулами (2) и (3). Введем обозначения

$$\tilde{v}_j = v_{2j} \quad (j = 0, 1, 2, 3), \quad \tilde{\omega} = \omega^2.$$

Тогда

$$V_{k1} = \sum_{j=0}^3 \tilde{\omega}^{jk} \tilde{v}_j \quad (k = 0, 1, 2, 3)$$

есть 4-точечное преобразование Фурье. По формулам (2) и (3) имеем

$$\begin{aligned} V_{k1} &= \sum_{j=0}^1 \tilde{\omega}^{2jk} \tilde{v}_{2j} + \tilde{\omega}^k \sum_{j=0}^1 \tilde{\omega}^{2jk} \tilde{v}_{2j+1} \quad (k = 0, 1), \\ V_{k+21} &= \sum_{j=0}^1 \tilde{\omega}^{2jk} \tilde{v}_{2j} - \tilde{\omega}^k \sum_{j=0}^1 \tilde{\omega}^{2jk} \tilde{v}_{2j+1} \quad (k = 0, 1) \end{aligned}$$

и так как

$$\begin{aligned} V_{k11} &= \sum_{j=0}^1 \tilde{\omega}^{2jk} \tilde{v}_{2j} = \tilde{v}_0 + \tilde{\omega}^{2k} \tilde{v}_2 \quad (k = 0, 1), \\ V_{k12} &= \sum_{j=0}^1 \tilde{\omega}^{2jk} \tilde{v}_{2j+1} = \tilde{v}_1 + \tilde{\omega}^{2k} \tilde{v}_3 \quad (k = 0, 1), \end{aligned}$$

то при  $k = 0$  получаем

$$\begin{aligned} V_{011} &= \tilde{v}_0 + \tilde{v}_2 = v_0 + v_4, \\ V_{012} &= \tilde{v}_1 + \tilde{v}_3 = v_2 + v_6, \end{aligned}$$

при  $k = 1$  -

$$\begin{aligned} V_{111} &= \tilde{v}_0 + \tilde{\omega}^2 \tilde{v}_2 = v_0 + \omega^4 v_4 = v_0 - v_4, \\ V_{112} &= \tilde{v}_1 + \tilde{\omega}^2 \tilde{v}_3 = v_2 + \omega^4 v_6 = v_2 - v_6. \end{aligned}$$

Подставляя полученные выражения в формулы (6), получаем

$$\begin{aligned} V_{01} &= V_{011} + V_{012} = (v_0 + v_4) + (v_2 + v_6), \\ V_{11} &= V_{111} + \tilde{\omega} V_{112} = (v_0 - v_4) + \omega^2 (v_2 - v_6) = (v_0 - v_4) - i(v_2 - v_6), \\ V_{21} &= V_{011} - V_{012} = (v_0 + v_4) - (v_2 + v_6), \\ V_{31} &= V_{111} - \tilde{\omega} V_{112} = (v_0 - v_4) - \omega^2 (v_2 - v_6) = (v_0 - v_4) + i(v_2 - v_6), \end{aligned}$$

так как  $\omega^2 = -i$ .

Поскольку для вычисления  $V_{k2}$  ( $k = 0, 1, 2, 3$ ) применим тот же метод с заменой компонент входного вектора на компоненты с нечетными номерами, то, заменяя  $v_{2j}$  на  $v_{2j+1}$ , получаем

$$\begin{aligned} V_{02} &= (v_1 + v_5) + (v_3 + v_7), \\ V_{12} &= (v_1 - v_5) - i(v_3 - v_7), \\ V_{22} &= (v_1 + v_5) - (v_3 + v_7), \\ V_{32} &= (v_1 - v_5) + i(v_3 - v_7). \end{aligned}$$

Теперь по формулам (5) имеем

$$\begin{aligned} V_0 &= V_{01} + V_{02} = \{(v_0 + v_4) + (v_2 + v_6)\} + \{(v_1 + v_5) + (v_3 + v_7)\}, \\ V_1 &= V_{11} + \omega V_{12} = \{(v_0 - v_4) - i(v_2 - v_6)\} + \omega \{(v_1 - v_5) - i(v_3 - v_7)\}, \\ V_2 &= V_{21} + \omega^2 V_{22} = \{(v_0 + v_4) - (v_2 + v_6)\} - i \{(v_1 + v_5) - (v_3 + v_7)\}, \\ V_3 &= V_{31} + \omega^3 V_{32} = \{(v_0 - v_4) + i(v_2 - v_6)\} - i \omega \{(v_1 - v_5) + i(v_3 - v_7)\}, \\ V_4 &= V_{01} - V_{02} = \{(v_0 + v_4) + (v_2 + v_6)\} - \{(v_1 + v_5) + (v_3 + v_7)\}, \\ V_5 &= V_{11} - \omega V_{12} = \{(v_0 - v_4) - i(v_2 - v_6)\} - \omega \{(v_1 - v_5) - i(v_3 - v_7)\}, \\ V_6 &= V_{21} - \omega^2 V_{22} = \{(v_0 + v_4) - (v_2 + v_6)\} + i \{(v_1 + v_5) - (v_3 + v_7)\}, \\ V_7 &= V_{31} - \omega^3 V_{32} = \{(v_0 - v_4) + i(v_2 - v_6)\} + i \omega \{(v_1 - v_5) + i(v_3 - v_7)\}. \end{aligned}$$

В матричном виде полученный алгоритм можно записать следующим образом:

$$\begin{pmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \\ V_6 \\ V_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & \omega & \\ & & & & & & & -i \\ & & & & & & & & -i\omega \end{pmatrix} \begin{pmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{pmatrix}.$$

Этот алгоритм содержит два нетривиальных комплексных умножения и 24 комплексных сложения (16 предложений и 8 постсложений).

Теперь получим формулы БПФ-алгоритма Кули – Тьюки с прореживанием по частоте. При  $n' = \frac{n}{2}$ ,  $n'' = 2$  формула (2) §10 принимает вид

$$V_{n''k'+k''} = \sum_{j'=0}^{\frac{n}{2}-1} \sum_{j''=0}^1 \beta^{j'k'} \gamma^{j''k''} \omega^{j'k''} v_{j'+n'j''},$$

где  $\beta = \omega^2$ ,  $\gamma = \omega^{\frac{n}{2}} = -1$ . Расписывая внутреннюю сумму, получаем

$$V_{2k'+k''} = \sum_{j'=0}^{\frac{n}{2}-1} \beta^{j'k'} \omega^{j'k''} (v_{j'} + (-1)^{k''} v_{j'+\frac{n}{2}}) = \sum_{j'=0}^{\frac{n}{2}-1} \omega^{2j'k'} \omega^{j'k''} (v_{j'} + (-1)^{k''} v_{j'+\frac{n}{2}}), \quad (7)$$

где  $k' = 0, 1, \dots, \frac{n}{2} - 1$ ;  $k'' = 0, 1$ .

При  $k'' = 0$  формула (7) принимает вид

$$V_{2k'} = \sum_{j'=0}^{\frac{n}{2}-1} \omega^{2j'k'} (v_{j'} + v_{j'+\frac{n}{2}}) \quad (k' = 0, 1, \dots, \frac{n}{2} - 1), \quad (8)$$

при  $k'' = 1$  –

$$V_{2k'+1} = \sum_{j'=0}^{\frac{n}{2}-1} \omega^{2j'k'} \omega^{j'} (v_{j'} - v_{j'+\frac{n}{2}}) \quad (k' = 0, 1, \dots, \frac{n}{2} - 1). \quad (9)$$

Этот алгоритм разбивает компоненты входного вектора на два подмножества, содержащие соответственно первые  $\frac{n}{2}$  компонент и последние  $\frac{n}{2}$  компонент. Компоненты выходного вектора разбиваются на подмножество компонент с четными индексами и подмножество компонент с нечетными индексами.

Вернемся к 8-точечному преобразованию Фурье и посмотрим, как выглядит алгоритм с прореживанием по частоте в этом случае. Так как  $n' = 4$ ,  $n'' = 2$ , то из формул (8) и (9) получаем

$$V_{2k} = \sum_{j=0}^3 \omega^{2jk} (v_j + v_{4+j}) \quad (k = 0, 1, 2, 3), \quad (10)$$



$$V_{2k+1} = \sum_{j=0}^3 \omega^{2jk} \omega^j (v_j - v_{4+j}) \quad (k = 0, 1, 2, 3). \quad (11)$$

Для вычисления  $V_{2k}$  воспользуемся снова формулами (8) и (9). Введем обозначения:

$$\tilde{\omega} = \omega^2, \quad \tilde{v}_j = v_j + v_{j+4} \quad (j = 0, 1, 2, 3), \quad \tilde{V}_k = V_{2k} \quad (k = 0, 1, 2, 3).$$

Тогда (10) запишется в виде

$$\tilde{V}_k = \sum_{j=0}^3 \tilde{\omega}^{jk} \tilde{v}_j \quad (k = 0, 1, 2, 3),$$

т. е. в виде 4-точечного преобразования Фурье. Имеем

$$\begin{aligned} \tilde{V}_{2k} &= \sum_{j=0}^1 \tilde{\omega}^{2jk} (\tilde{v}_j + \tilde{v}_{2+j}) \quad (k = 0, 1), \\ \tilde{V}_{2k+1} &= \sum_{j=0}^1 \tilde{\omega}^{2jk} \tilde{\omega}^j (\tilde{v}_j - \tilde{v}_{2+j}) \quad (k = 0, 1) \end{aligned}$$

или

$$\begin{aligned} \tilde{V}_{2k} &= (\tilde{v}_0 + \tilde{v}_2) + \tilde{\omega}^{2k} (\tilde{v}_1 + \tilde{v}_3) = \{(v_0 + v_4) + (v_2 + v_6)\} + \omega^{4k} \{(v_1 + v_5) + (v_3 + v_7)\}, \\ \tilde{V}_{2k+1} &= (\tilde{v}_0 - \tilde{v}_2) + \tilde{\omega}^{2k} \tilde{\omega} (\tilde{v}_1 - \tilde{v}_3) = \{(v_0 + v_4) - (v_2 + v_6)\} + \omega^{4k+2} \{(v_1 + v_5) - (v_3 + v_7)\} \end{aligned} \quad (12)$$

( $k = 0, 1$ ).

Подставляя в полученные формулы (12) значения  $k$ , получаем

при  $k = 0$  —

$$V_0 = \tilde{V}_0 = \{(v_0 + v_4) + (v_2 + v_6)\} + \{(v_1 + v_5) + (v_3 + v_7)\},$$

$$V_2 = \tilde{V}_1 = \{(v_0 + v_4) - (v_2 + v_6)\} + \omega^2 \{(v_1 + v_5) - (v_3 + v_7)\},$$

при  $k = 1$  —

$$V_4 = \tilde{V}_2 = \{(v_0 + v_4) + (v_2 + v_6)\} + \omega^4 \{(v_1 + v_5) + (v_3 + v_7)\},$$

$$V_6 = \tilde{V}_3 = \{(v_0 + v_4) - (v_2 + v_6)\} + \omega^6 \{(v_1 + v_5) - (v_3 + v_7)\}.$$

(13)

Теперь получим формулы для компонент  $V_{2k+1}$ . Так как формулы (11) отличаются от формул (10) тем, что вместо суммы соответствующих компонент входного вектора стоит разность этих компонент, а также присутствует дополнительный множитель  $\omega^j$ , то аналогичные выкладки можно опустить и записать результат сразу

$$\begin{aligned} V_1 &= \{(v_0 - v_4) + \omega^2 (v_2 - v_6)\} + \{\omega (v_1 - v_5) + \omega^3 (v_3 - v_7)\}, \\ V_3 &= \{(v_0 - v_4) - \omega^2 (v_2 - v_6)\} + \omega^2 \{\omega (v_1 - v_5) - \omega^3 (v_3 - v_7)\}, \\ V_5 &= \{(v_0 - v_4) + \omega^2 (v_2 - v_6)\} + \omega^4 \{\omega (v_1 - v_5) + \omega^3 (v_3 - v_7)\}, \\ V_7 &= \{(v_0 - v_4) - \omega^2 (v_2 - v_6)\} + \omega^6 \{\omega (v_1 - v_5) - \omega^3 (v_3 - v_7)\}. \end{aligned} \quad (14)$$

Учитывая, что  $\omega^8 = 1$ , откуда

$$\omega^2 = -i, \quad \omega^4 = -1, \quad \omega^6 = i, \quad \omega^3 = -i\omega, \quad \omega^5 = -\omega, \quad \omega^7 = i\omega,$$

запишем полученный алгоритм из формул (13) и (14) в матричном виде

$$\begin{pmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \\ V_6 \\ V_7 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & i & -1 & -i \end{pmatrix} \begin{pmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & \omega & & \\ & & & & & & -i & \\ & & & & & & & \omega^3 \end{pmatrix}.$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{pmatrix}.$$

Полученный алгоритм содержит два нетривиальных комплексных умножения и 24 комплексных сложений (8 предсложений и 16 постсложений). Таким образом, характеристики двух рассмотренных алгоритмов вычисления 8-точечного преобразования Фурье совпадают.

Эти два алгоритма отличаются структурой и последовательностью вычислений, но имеют одно и то же число операций. Подробно рассмотрим только характеристики алгоритма с прореживанием по времени. Этот алгоритм сводит  $n$ -точечное преобразование Фурье к двум  $\frac{n}{2}$ -точечным и некоторым дополнительным умножениям и сложениям. Часть из умножений представляют собой умножения на  $\pm 1$  или на  $\pm i$ . Они тривиальны и не требуют действительного вычисления. В алгоритме их можно обрабатывать отдельно.

Алгоритм работает рекурсивно, разбивая на каждом шаге  $n$ -точечное преобразование на два  $\frac{n}{2}$ -точечных, которые в свою очередь разбиваются точно таким же образом. Из формул (2), (3) видно, что число  $M_{\mathbb{C}}(n)$  комплексных умножений  $n$ -точечного БПФ удовлетворяет рекуррентному уравнению

$$M_{\mathbb{C}}(n) = 2 M_{\mathbb{C}}\left(\frac{n}{2}\right) + \frac{n}{2}, \quad (15)$$

а число  $A_{\mathbb{C}}(n)$  комплексных сложений – уравнению

$$A_{\mathbb{C}}(n) = 2 A_{\mathbb{C}}\left(\frac{n}{2}\right) + n. \quad (16)$$

Если  $n = 2^m$ , то решения этих уравнений (что нетрудно проверить) даются равенствами

$$M_{\mathbb{C}}(n) = \frac{n}{2} \log_2 n, \quad A_{\mathbb{C}}(n) = n \log_2 n.$$

Если реализовать комплексное умножение с помощью четырех вещественных умножений и двух вещественных сложений, то характеристики БПФ даются равенствами

$$M_{\mathbb{R}}(n) = 2n \log_2 n, \quad A_{\mathbb{R}}(n) = 2n \log_2 n + 2 \left(\frac{n}{2}\right) \log_2 n = 3n \log_2 n.$$

Если реализовать комплексное умножение с помощью трех вещественных умножений и трех вещественных сложений, то характеристики имеют вид

$$M_{\mathbb{R}}(n) = \frac{3n}{2} \log_2 n, \quad A_{\mathbb{R}}(n) = \frac{3n}{2} \log_2 n + 2n \log_2 n = \frac{7n}{2} \log_2 n.$$

Если выбросить все тривиальные умножения, а также воспользоваться специальным умножением на

$$\omega^{\frac{n}{8}} = \left(e^{-\frac{2i\pi}{n}}\right)^{\frac{n}{8}} = e^{-\frac{i\pi}{4}} = \frac{1-i}{\sqrt{2}},$$

то число умножений еще уменьшится.

Умножение на  $\frac{1-i}{\sqrt{2}}$  требует всего двух вещественных умножений и двух вещественных сложений. Действительно,

$$(a+ib) \cdot \frac{1-i}{\sqrt{2}} = \frac{a+b}{\sqrt{2}} + i \frac{b-a}{\sqrt{2}}.$$

Если учесть все это, то характеристики БПФ имеют вид

$$M_{\mathbb{R}}(n) = \frac{n}{2} (3 \log_2 n - 10) + 8, \quad A_{\mathbb{R}}(n) = \frac{n}{2} (7 \log_2 n - 10) + 8.$$



### §13. АЛГОРИТМ КУЛИ – ТЬЮКИ ПО ОСНОВАНИЮ ЧЕТЫРЕ

Рассмотрим БПФ-алгоритм Кули – Тьюки по основанию четыре с прореживанием по времени. Уравнения этой формы алгоритма можно получить простой подстановкой  $n' = 4$  и  $n'' = \frac{n}{4}$  ( $n = 4^m$ ) в общее уравнение (2) §10 :

$$V_{n''k'+k''} = \sum_{j'=0}^3 \beta^{j'k'} \omega^{j'k''} \sum_{j''=0}^{\frac{n}{4}-1} \gamma^{j''k''} v_{j'+n'j''} \quad (k' = 0, 1, 2, 3; k'' = 0, 1, \dots, \frac{n}{4} - 1),$$

где  $\beta = \omega^{\frac{n}{4}}$ ,  $\gamma = \omega^4$ .

Так как  $\omega = e^{-\frac{2i\pi}{n}}$ , то  $\beta = e^{-\frac{2i\pi}{4}} = e^{-\frac{i\pi}{2}} = -i$ , поэтому формулы принимают вид

$$V_{\frac{n}{4}k'+k''} = \sum_{j'=0}^3 (-i)^{j'k'} \omega^{j'k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{j'+4j''} \quad (k' = 0, 1, 2, 3; k'' = 0, 1, \dots, \frac{n}{4} - 1).$$

Раскроем внешнюю сумму по  $j'$  :

$$\begin{aligned} V_{\frac{n}{4}k'+k''} &= \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j''} + (-i)^{k'} \omega^{k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+1} \\ &\quad + (-i)^{2k'} \omega^{2k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+2} + (-i)^{3k'} \omega^{3k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+3}. \end{aligned}$$

Для  $k' = 0$ ,  $k'' = 0, 1, \dots, \frac{n}{4} - 1$  получаем

$$V_{k''} = \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j''} + \omega^{k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+1} + \omega^{2k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+2} + \omega^{3k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+3};$$

для  $k' = 1$ ,  $k'' = 0, 1, \dots, \frac{n}{4} - 1$  —

$$\begin{aligned} V_{\frac{n}{4}+k''} &= \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j''} + (-i)\omega^{k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+1} - \omega^{2k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+2} \\ &\quad + i\omega^{3k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+3}; \end{aligned}$$

для  $k' = 2$ ,  $k'' = 0, 1, \dots, \frac{n}{4} - 1$  —

$$V_{\frac{n}{2}+k''} = \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j''} - \omega^{k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+1} + \omega^{2k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+2} - \omega^{3k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+3};$$

для  $k' = 3$ ,  $k'' = 0, 1, \dots, \frac{n}{4} - 1$  —

$$\begin{aligned} V_{\frac{3n}{4}+k''} &= \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j''} + i\omega^{k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+1} - \omega^{2k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+2} \\ &\quad + (-i)\omega^{3k''} \sum_{j''=0}^{\frac{n}{4}-1} \omega^{4j''k''} v_{4j'+3}. \end{aligned}$$

Полученные формулы можно записать в виде

$$\begin{pmatrix} V_k \\ V_{k+\frac{n}{4}} \\ V_{k+\frac{n}{2}} \\ V_{k+\frac{3n}{4}} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix} \begin{pmatrix} \sum_{j=0}^{\frac{n}{4}-1} \omega^{4jk} v_{4j} \\ \omega^k \sum_{j=0}^{\frac{n}{4}-1} \omega^{4jk} v_{4j+1} \\ \omega^{2k} \sum_{j=0}^{\frac{n}{4}-1} \omega^{4jk} v_{4j+2} \\ \omega^{3k} \sum_{j=0}^{\frac{n}{4}-1} \omega^{4jk} v_{4j+3} \end{pmatrix} \quad (1)$$

для  $k = 0, 1, \dots, \frac{n}{4} - 1$ .

Для каждого из  $\frac{n}{4}$  рассматриваемых значений индекса  $k$  такое матричное уравнение определяет четыре компоненты преобразования. Этот метод позволяет  $n$ -точечное преобразование Фурье заменить четырьмя  $\frac{n}{4}$ -точечными преобразованиями Фурье и некоторыми дополнительными вычислениями. Из соотношений (1) следует, что для каждого  $k$  имеется только три различных комплексных умножения и 8 комплексных сложений, так как матрицу из (1) можно разложить в произведение

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & -i \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & i \end{pmatrix} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 \end{pmatrix}. \quad (2)$$

Таким образом, рекуррентное уравнение для числа умножений имеет вид

$$M_{\mathbf{C}}(n) = 4 M_{\mathbf{C}}\left(\frac{n}{4}\right) + \frac{3n}{4}.$$

Его решение дается формулой

$$M_{\mathbf{C}}(n) = \frac{3}{4} n \log_4 n.$$

Так как на самом внутреннем шаге умножений вообще нет (4-точечное преобразование Фурье не содержит умножений), то общее число комплексных умножений при рекуррентном применении алгоритма равно

$$M_{\mathbf{C}}(n) = \frac{3}{4} n \log_4 n - \frac{3}{4} n = \frac{3}{4} n (\log_4 n - 1) = \frac{3}{8} n (\log_2 n - 2).$$

Рекуррентное уравнение для числа комплексных сложений алгоритма

$$A_{\mathbf{C}}(n) = 4 A_{\mathbf{C}}\left(\frac{n}{4}\right) + 2n$$

имеет решение

$$A_{\mathbf{C}}(n) = 2n \log_4 n = n \log_2 n.$$

Если для вычисления комплексных умножений использовать алгоритм с тремя вещественными умножениями и тремя вещественными сложениями, то характеристики алгоритма даются равенствами

$$M_{\mathbf{R}}(n) = \frac{9}{8} n (\log_2 n - 2), \quad A_{\mathbf{R}}(n) = \frac{9}{8} n (\log_2 n - 2) + 2n \log_2 n = \frac{25}{8} n \log_2 n - \frac{9}{4} n.$$

**Пример 1.** Начнем с 4-точечного преобразования Фурье. Так как  $\omega = e^{-\frac{2i\pi}{4}} = -i$ , то общая формула принимает вид

$$V_k = \sum_{j=0}^3 (-i)^{jk} v_j \quad (k = 0, 1, 2, 3),$$

т. е.

$$\begin{pmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix} \begin{pmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \end{pmatrix}. \quad (3)$$

Этот алгоритм, очевидно, не содержит нетривиальных умножений. Комплексных сложений, согласно сделанному выше разложению матрицы (см. формулу (2)), потребуется всего 8.



**Пример 2.** Построим алгоритм 16-точечного преобразования Фурье. Так как  $n = 4^2$ , то  $n' = 4$ ,  $n'' = 4$ . Кроме того,  $\omega^4 = -i$ ,  $\omega^8 = -1$ ,  $\omega^{12} = i$ . Тогда формулы (1) имеют вид

$$\begin{pmatrix} V_k \\ V_{k+4} \\ V_{k+8} \\ V_{k+12} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix} \begin{pmatrix} \sum_{j=0}^3 (-i)^{jk} v_{4j} \\ \omega^k \sum_{j=0}^3 (-i)^{jk} v_{4j+1} \\ \omega^{2k} \sum_{j=0}^3 (-i)^{jk} v_{4j+2} \\ \omega^{3k} \sum_{j=0}^3 (-i)^{jk} v_{4j+3} \end{pmatrix} \quad (k = 0, 1, 2, 3). \quad (4)$$

Введем обозначения

$$V_{kl} = \sum_{j=0}^3 (-i)^{jk} v_{4j+l} \quad (l = 0, 1, 2, 3; k = 0, 1, 2, 3).$$

Тогда

$$\begin{pmatrix} V_k \\ V_{k+4} \\ V_{k+8} \\ V_{k+12} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix} \begin{pmatrix} V_{k0} \\ \omega^k V_{k1} \\ \omega^{2k} V_{k2} \\ \omega^{3k} V_{k3} \end{pmatrix} \quad (k = 0, 1, 2, 3). \quad (5)$$

Каждое  $V_{kl}$  ( $l = 0, 1, 2, 3$ ;  $k = 0, 1, 2, 3$ ) есть компонента соответствующего 4-точечного преобразования Фурье, а именно вектор  $(V_{0l}, V_{1l}, V_{2l}, V_{3l})$  есть 4-точечное преобразование Фурье вектора  $\mathbf{v}_l = (v_l, v_{l+4}, v_{l+8}, v_{l+12})$ , т. е.

$$\begin{pmatrix} V_{0l} \\ V_{1l} \\ V_{2l} \\ V_{3l} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix} \begin{pmatrix} v_l \\ v_{l+4} \\ v_{l+8} \\ v_{l+12} \end{pmatrix} \quad (l = 0, 1, 2, 3).$$

Таким образом, вычисление  $V_{kl}$  ( $l = 0, 1, 2, 3$ ;  $k = 0, 1, 2, 3$ ) не содержит операций умножения, операций сложения здесь требуется  $8 \cdot 4 = 32$ .

Далее, для вычисления компонент  $V_k$  по формулам (5) потребуется 9 нетривиальных умножений (по 3 умножения для  $k = 1, 2, 3$ ) и  $8 \cdot 4 = 32$  сложения. Таким образом, алгоритм требует всего 9 комплексных умножений и 64 комплексных сложения.

В матричном виде полученный алгоритм можно записать так:

$$\mathbf{V} = \mathbf{B} \mathbf{W} \mathbf{A} \mathbf{v},$$

где

$$\mathbf{B} = \begin{pmatrix} 1 & 1 & 1 & 1 & & & & \\ & & & & 1 & 1 & 1 & 1 \\ & & & & & & & 1 & 1 & 1 & 1 \\ & 1 & -i & -1 & i & & & & & & & \\ & & & & 1 & -i & -1 & i & & & & \\ & & & & & & & 1 & -i & -1 & i \\ & 1 & -1 & 1 & -1 & & & & & & & \\ & & & & 1 & -1 & 1 & -1 & & & & \\ & & & & & & & 1 & -1 & 1 & -1 \\ & 1 & i & -1 & -i & & & & & & & \\ & & & & 1 & i & -1 & -i & & & & \\ & & & & & & & 1 & i & -1 & -i \\ & & & & & & & & & & 1 & i & -1 & -i \end{pmatrix},$$

$$W = \begin{pmatrix} 1 & & & & & & & & & & & & & & & & \\ & 1 & & & & & & & & & & & & & & & \\ & & 1 & & & & & & & & & & & & & & \\ & & & 1 & & & & & & & & & & & & & \\ & & & & 1 & & & & & & & & & & & & \\ & & & & & 1 & & & & & & & & & & & \\ & & & & & & \omega & & & & & & & & & & \\ & & & & & & & \omega^2 & & & & & & & & & \\ & & & & & & & & \omega^3 & & & & & & & & \\ & & & & & & & & & 1 & & & & & & & \\ & & & & & & & & & & \omega^2 & & & & & & \\ & & & & & & & & & & & \omega^4 & & & & & \\ & & & & & & & & & & & & \omega^6 & & & & \\ & & & & & & & & & & & & & 1 & & & \\ & & & & & & & & & & & & & & \omega^3 & & \\ & & & & & & & & & & & & & & & \omega^6 & \\ & & & & & & & & & & & & & & & & \omega^9 \end{pmatrix},$$

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & -i & 0 & 0 & 0 & -1 & 0 & 0 & 0 & i & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -i & 0 & 0 & 0 & -1 & 0 & 0 & 0 & i & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -i & 0 & 0 & 0 & -1 & 0 & 0 & 0 & i & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -i & 0 & 0 & 0 & -1 & 0 & 0 & 0 & i \\ 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & i & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -i & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & i & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -i & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & i & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -i & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & i & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -i \end{pmatrix}.$$

Получим формулы алгоритма Кули – Тьюки по основанию четыре с прореживанием по частоте, т. е.  $n' = \frac{n}{4}$ ,  $n'' = 4$  ( $n = n'n'' = 4^m$ ). Тогда в общей формуле (2) §10  $\beta = \omega^4$ ,  $\gamma = \omega^{\frac{n}{4}}$  и получаем

$$V_{4k'+k''} = \sum_{j'=0}^{\frac{n}{4}-1} \omega^{4j'k'} \omega^{j'k''} \sum_{j''=0}^3 (\omega^{\frac{n}{4}})^{j''k''} v_{j'+\frac{n}{4}j''} \quad (k' = 0, 1, \dots, \frac{n}{4} - 1; k'' = 0, 1, 2, 3). \quad (6)$$

Учитывая, что

$$\omega^{\frac{n}{4}} = \left(e^{-\frac{2i\pi}{n}}\right)^{\frac{n}{4}} = e^{-\frac{i\pi}{2}} = -i,$$

распишем внутреннюю сумму в (6)

$$V_{4k'+k''} = \sum_{j'=0}^{\frac{n}{4}-1} \omega^{4j'k'} \omega^{j'k''} \{v_{j'} + (-i)^{k''} v_{j'+\frac{n}{4}} + (-i)^{2k''} v_{j'+\frac{n}{2}} + (-i)^{3k''} v_{j'+\frac{3n}{4}}\} =$$

$$\sum_{j=0}^{\frac{n}{4}-1} \omega^{4jk'} \omega^{jk''} \{v_j + (-i)^{k''} v_{j+\frac{n}{4}} + (-1)^{k''} v_{j+\frac{n}{2}} + i^{k''} v_{j+\frac{3n}{4}}\} \quad (7)$$

$$(k' = 0, 1, \dots, \frac{n}{4} - 1; k'' = 0, 1, 2, 3).$$



Для  $k'' = 0$  получаем

$$V_{4k} = \sum_{j=0}^{\frac{n}{4}-1} \omega^{4jk} \{v_j + v_{j+\frac{n}{4}} + v_{j+\frac{n}{2}} + v_{j+\frac{3n}{4}}\} \quad (k = 0, 1, \dots, \frac{n}{4} - 1);$$

для  $k'' = 1$  —

$$V_{4k+1} = \sum_{j=0}^{\frac{n}{4}-1} \omega^{4jk} \omega^j \{v_j - i v_{j+\frac{n}{4}} - v_{j+\frac{n}{2}} + i v_{j+\frac{3n}{4}}\} \quad (k = 0, 1, \dots, \frac{n}{4} - 1);$$

для  $k'' = 2$  —

$$V_{4k+2} = \sum_{j=0}^{\frac{n}{4}-1} \omega^{4jk} \omega^{2j} \{v_j - v_{j+\frac{n}{4}} + v_{j+\frac{n}{2}} - v_{j+\frac{3n}{4}}\} \quad (k = 0, 1, \dots, \frac{n}{4} - 1);$$

для  $k'' = 3$  —

$$V_{4k+3} = \sum_{j=0}^{\frac{n}{4}-1} \omega^{4jk} \omega^{3j} \{v_j + i v_{j+\frac{n}{4}} - v_{j+\frac{n}{2}} - i v_{j+\frac{3n}{4}}\} \quad (k = 0, 1, \dots, \frac{n}{4} - 1).$$

Введем обозначения

$$v_{jl} = \omega^{jl} (v_j + (-i)^l v_{j+\frac{n}{4}} + (-1)^l v_{j+\frac{n}{2}} + (i)^l v_{j+\frac{3n}{4}}) \quad (l = 0, 1, 2, 3; j = 0, 1, \dots, \frac{n}{4} - 1),$$

тогда полученные формулы перепишутся в виде

$$\begin{aligned} V_{4k} &= \sum_{j=0}^{\frac{n}{4}-1} \omega^{4jk} v_{j0}, \\ V_{4k+1} &= \sum_{j=0}^{\frac{n}{4}-1} \omega^{4jk} v_{j1}, \\ V_{4k+2} &= \sum_{j=0}^{\frac{n}{4}-1} \omega^{4jk} v_{j2}, \\ V_{4k+3} &= \sum_{j=0}^{\frac{n}{4}-1} \omega^{4jk} v_{j3}, \end{aligned} \quad (k = 0, 1, \dots, \frac{n}{4} - 1)$$

т. е. опять  $n$ -точечное преобразование Фурье сводится к четырем  $\frac{n}{4}$ -точечным и некоторым дополнительным умножениям и сложениям. Рекуррентные уравнения для определения числа умножений и сложений этого алгоритма такие же, как для алгоритма с прореживанием по времени, а значит, и сложность та же, что и у алгоритма с прореживанием по времени.

#### §14. АЛГОРИТМ ГУДА – ТОМАСА БЫСТРОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ

Этот алгоритм также переводит одномерное преобразование Фурье в двумерное. При этом разложение длины преобразования  $n$  в произведение  $n'n''$  должно быть таким, что  $n'$  и  $n''$  взаимно простые числа. Основой же отображения линейной последовательности в двумерную таблицу служит китайская теорема об остатках.

Входные индексы задаются вычетами по правилу

$$j' \equiv j \pmod{n'}, \quad j'' \equiv j \pmod{n''}.$$

Согласно китайской теореме об остатках существуют такие целые числа  $N'$  и  $N''$ , что выполняется

$$j = j'N''n'' + j''N'n' \pmod{n},$$

где  $N'n' + N''n'' = 1$ .

Входные индексы определяются иначе. Пусть

$$k' \equiv N''k \pmod{n'}, \quad k'' \equiv N'k \pmod{n''},$$

тогда

$$k = n''k' + n'k'' \pmod{n}.$$

Действительно,

$$\begin{aligned} k &= n''(N''k + n'Q_1) + n'(N'k + n''Q_2) \pmod{n'n''} = \\ &= k(N''n'' + N'n') + (Q_1 + Q_2)n'n'' \pmod{n'n''} = k(N''n'' + N'n') \pmod{n} = k. \end{aligned}$$

Во вновь введенных обозначениях индексов формула

$$V_k = \sum_{j=0}^{n-1} \omega^{jk} v_j \quad (k = 0, 1, \dots, n-1)$$

преобразуется к виду

$$V_{n''k' + n'k'' \pmod{n}} = \sum_{j'=0}^{n'-1} \sum_{j''=0}^{n''-1} \omega^{(j'N''n'' + j''N'n')(n''k' + n'k'')} v_{j'N''n'' + j''N'n' \pmod{n}}. \quad (1)$$

Так как порядок элемента  $\omega$  равен  $n = n'n''$ , то

$$\begin{aligned} \omega^{(j'N''n'' + j''N'n')(n''k' + n'k'')} &= \omega^{N''(n'')^2 j'k'} \omega^{N'(n')^2 j''k''} \omega^{(j'N''k'' + j''N'k')n'n''} = \\ &= \omega^{N''(n'')^2 j'k'} \omega^{N'(n')^2 j''k''}. \end{aligned}$$

Введем обозначения

$$\beta = \omega^{N''(n'')^2}, \quad \gamma = \omega^{N'(n')^2}, \quad V_{k'k''} = V_{n''k' + n'k'' \pmod{n}}, \quad v_{j'j''} = v_{j'N''n'' + j''N'n' \pmod{n}},$$

тогда (1) примет вид

$$V_{k'k''} = \sum_{j'=0}^{n'-1} \sum_{j''=0}^{n''-1} \beta^{j'k'} \gamma^{j''k''} v_{j'j''} \quad (k' = 0, 1, \dots, n'-1; k'' = 0, 1, \dots, n''-1). \quad (2)$$

Поскольку  $\omega = e^{-\frac{2i\pi}{n}}$ , то

$$\beta = \omega^{N''(n'')^2} = \left( e^{-\frac{2i\pi}{n'n''}} \right)^{N''(n'')^2} = \left( e^{-\frac{2i\pi}{n'}} \right)^{N''n''} = e^{-\frac{2i\pi}{n'}},$$



так как  $N''n'' \equiv 1 \pmod{n'}$ . Значит,  $\beta$  есть корень степени  $n'$  из единицы. Аналогично,

$$\gamma = \omega^{N'(n')^2} = \left( e^{-\frac{2i\pi}{n'n''}} \right)^{N'(n')^2} = \left( e^{-\frac{2i\pi}{n''}} \right)^{N'n'} = e^{-\frac{2i\pi}{n''}},$$

так как  $N'n' \equiv 1 \pmod{n''}$ . Значит,  $\gamma$  есть корень степени  $n''$  из единицы. Перепишем формулы (2) в виде

$$V_{k'k''} = \sum_{j'=0}^{n'-1} \beta^{j'k'} \left\{ \sum_{j''=0}^{n''-1} \gamma^{j''k''} v_{j'j''} \right\} \quad (k' = 0, 1, \dots, n' - 1; k'' = 0, 1, \dots, n'' - 1). \quad (3)$$

Здесь внутреннее преобразование есть  $n''$ -точечное преобразование Фурье вектора  $(v_{j'0}, v_{j'1}, \dots, v_{j'n''-1})$ , а внешнее –  $n'$ -точечное преобразование Фурье вектора, полученного в результате внутреннего преобразования.

Число умножений и число сложений равно примерно  $n(n' + n'')$ .

Если  $n'$  или  $n''$  является составным числом, то можно в свою очередь внешнее или внутреннее преобразование разбить еще раз, применяя алгоритм БПФ. Если длина преобразования  $n$  разлагается в произведение попарно взаимно простых множителей

$$n = \prod_{l=1}^s n_l,$$

то описанная форма БПФ-алгоритма требует примерно

$$n \sum_{l=1}^s n_l$$

умножений и столько же сложений.

**Пример.** Посмотрим, как выглядят входная и выходная таблицы при  $n = 15$ .

В этом случае  $n = 3 \cdot 5$ , пусть  $n' = 3$ ,  $n'' = 5$ , тогда  $N' = -3$ ,  $N'' = 2$  ( $-3n' + 2n'' = 1$ ). Входной вектор отображается в таблицу

$$\begin{pmatrix} v_0 & v_6 & v_{12} & v_3 & v_9 \\ v_{10} & v_1 & v_7 & v_{13} & v_4 \\ v_5 & v_{11} & v_2 & v_8 & v_{14} \end{pmatrix} = \begin{pmatrix} v_{00} & v_{01} & v_{02} & v_{03} & v_{04} \\ v_{10} & v_{11} & v_{12} & v_{13} & v_{14} \\ v_{20} & v_{21} & v_{22} & v_{23} & v_{24} \end{pmatrix}.$$

Над каждой строкой этой таблицы производится 5-точечное преобразование Фурье

$$r_{j'k''} = \sum_{j''=0}^4 \gamma^{j''k''} v_{j'j''} \quad (j' = 0, 1, 2),$$

в результате получается новая двумерная таблица

$$\begin{pmatrix} r_{00} & r_{01} & r_{02} & r_{03} & r_{04} \\ r_{10} & r_{11} & r_{12} & r_{13} & r_{14} \\ r_{20} & r_{21} & r_{22} & r_{23} & r_{24} \end{pmatrix},$$

над каждым столбцом которой производится 3-точечное преобразование Фурье

$$V_{k'k''} = \sum_{j'=0}^2 \beta^{j'k'} r_{j'k''} \quad (k'' = 0, 1, 2, 3, 4).$$

Полученные результаты записываются в таблицу

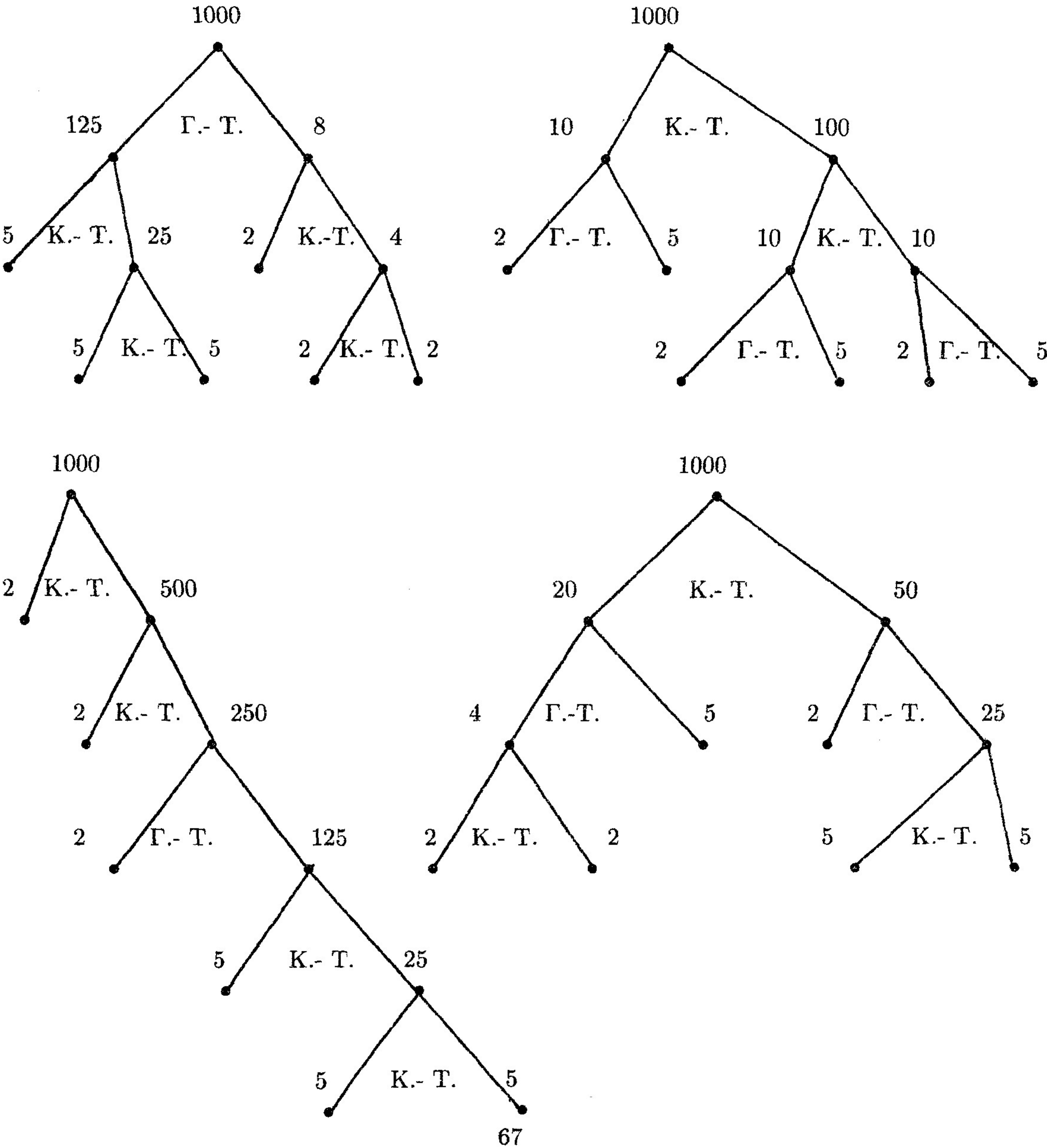
$$\begin{pmatrix} V_{00} & V_{01} & V_{02} & V_{03} & V_{04} \\ V_{10} & V_{11} & V_{12} & V_{13} & V_{14} \\ V_{20} & V_{21} & V_{22} & V_{23} & V_{24} \end{pmatrix} = \begin{pmatrix} V_0 & V_3 & V_6 & V_9 & V_{12} \\ V_5 & V_8 & V_{11} & V_{14} & V_2 \\ V_{10} & V_{13} & V_1 & V_4 & V_7 \end{pmatrix}$$

$$(k' \equiv 2k \pmod{3}, \quad k'' \equiv -3k \pmod{5}, \quad k = 5k' + 3k'' \pmod{15}).$$

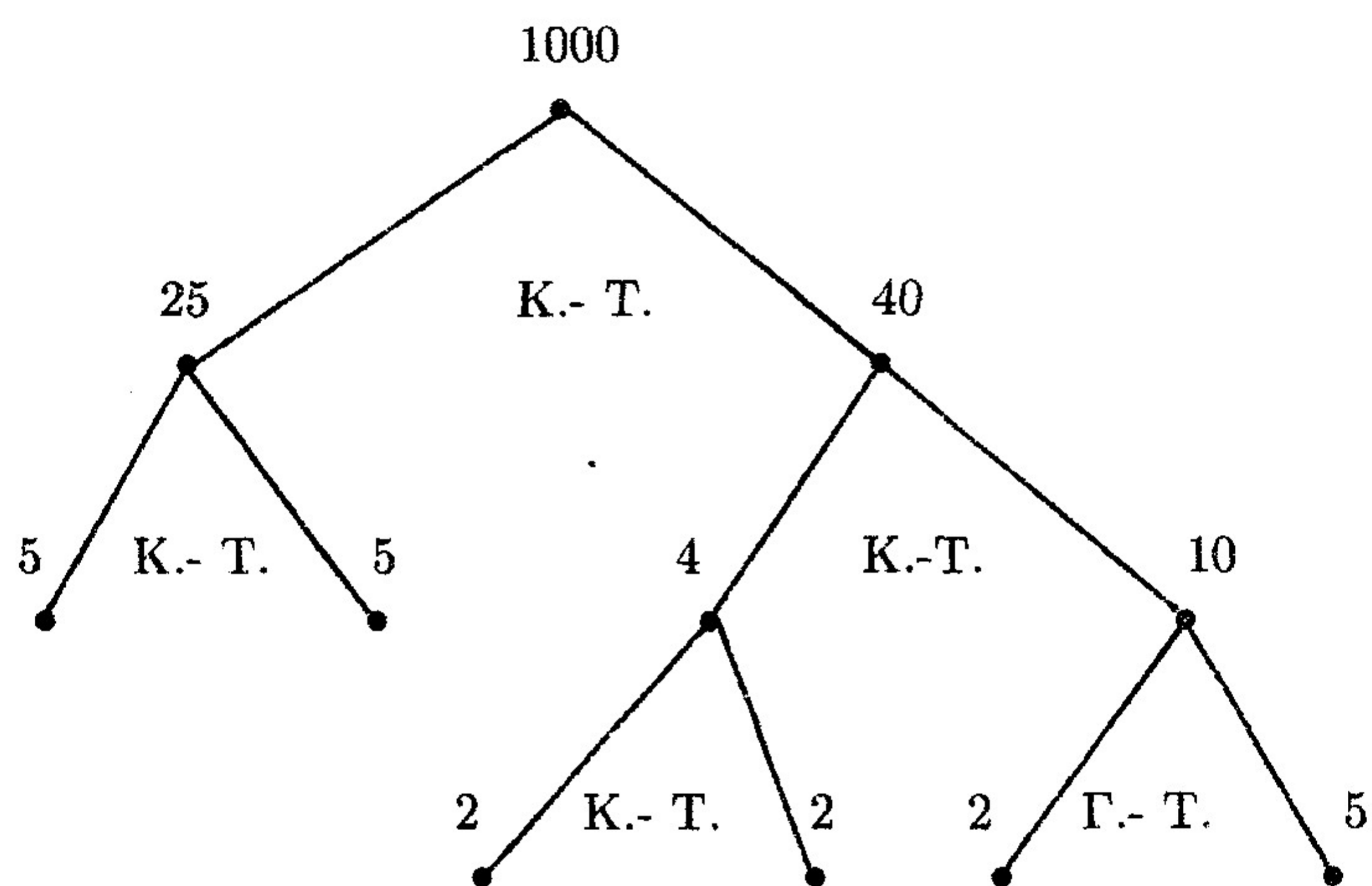
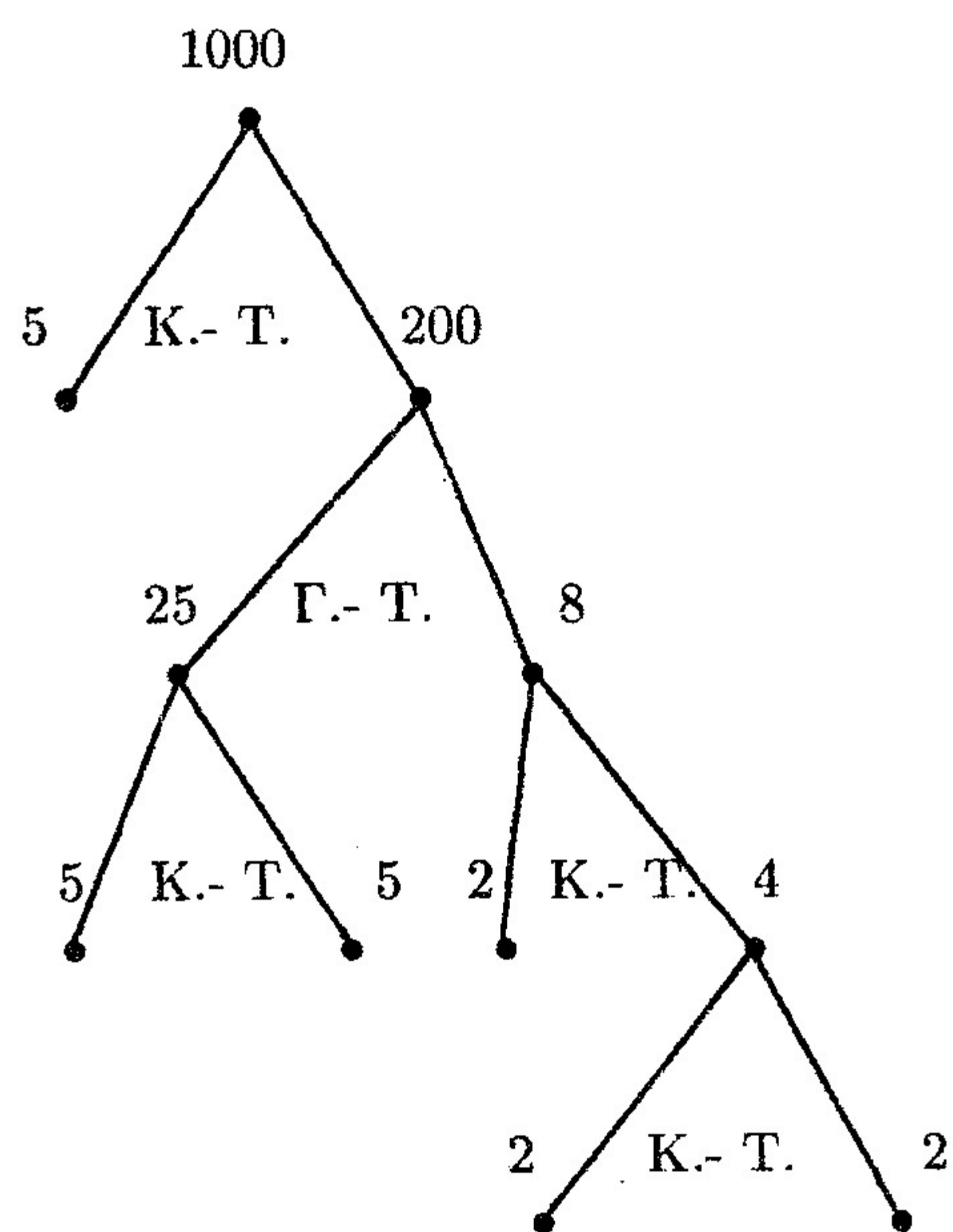
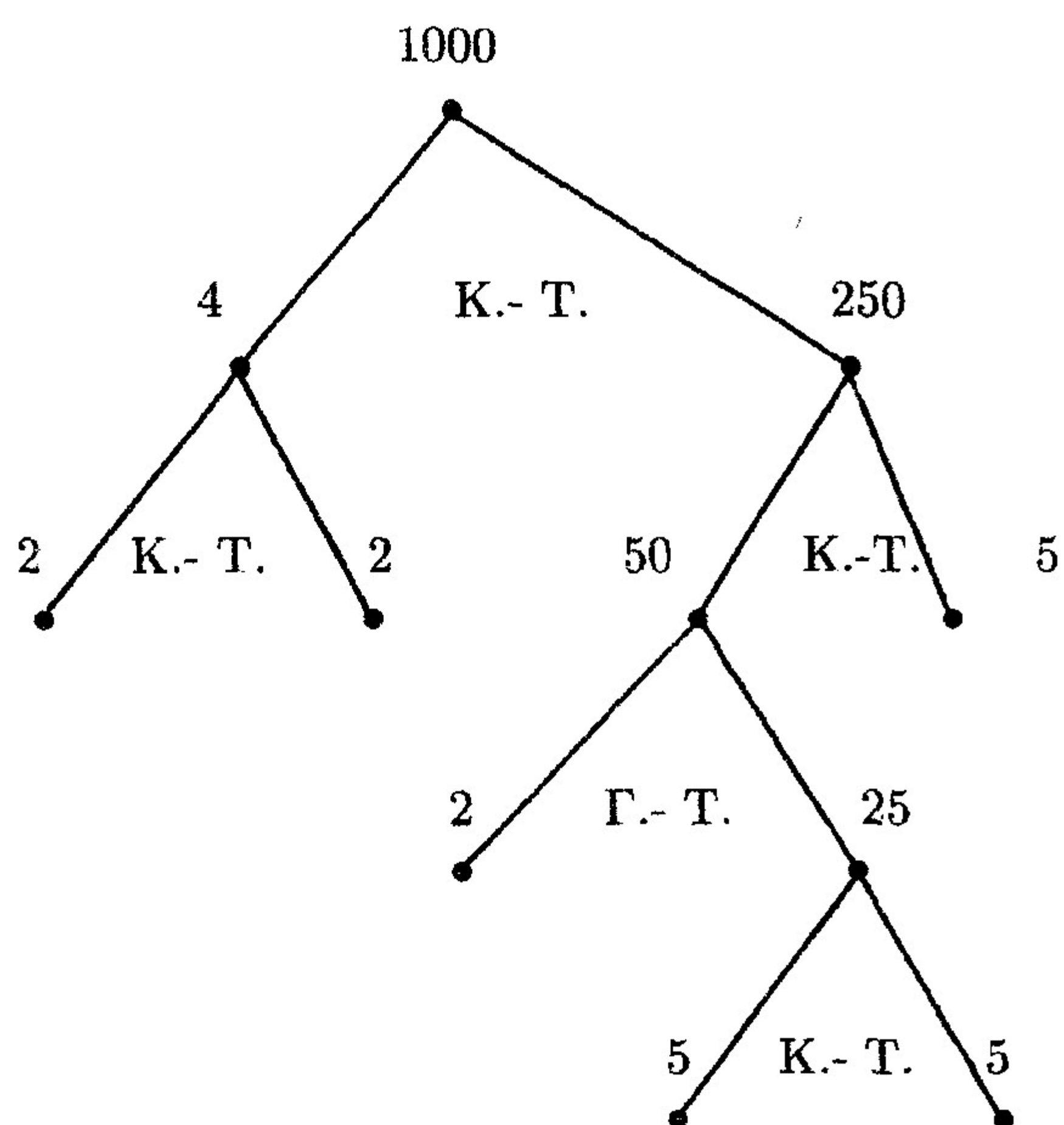
Далее, из полученной таблицы выбираются компоненты выходного вектора  $(V_0, V_1, \dots, V_{14})$ .

Можно строить алгоритмы, содержащие БПФ-алгоритм Кули – Тьюки и БПФ-алгоритм Гуда – Томаса одновременно. Например, используя БПФ-алгоритм Гуда – Томаса, можно 63-точечное преобразование Фурье разбить на 7-точечное и 9-точечное, а затем, используя БПФ-алгоритм Кули – Тьюки, 9-точечное разбить на два 3-точечных. Вычисления при этом преобразуются в форму, аналогичную трехмерному  $(3 \times 3 \times 7)$ -преобразованию Фурье.

1000-точечное преобразование Фурье можно разбивать на преобразования меньшей длины разными способами, используя БПФ-алгоритмы Гуда – Томаса и Кули – Тьюки. Каждый способ будет содержать 2-точечный модуль три раза и 5-точечный модуль три раза:







Все эти способы существенно различны, так как БПФ-алгоритмы для малых модулей используются в разной последовательности. Число умножений и сложений, чувствительность алгоритма к погрешностям вычислений и легкость реализации алгоритмов также различны.

# §15. АЛГОРИТМ РЕЙДЕРА В СЛУЧАЕ, КОГДА ДЛИНА ПРЕОБРАЗОВАНИЯ РАВНА ПРОСТОМУ ЧИСЛУ

Одним из эффективных способов вычисления дискретного преобразования Фурье является сведение к вычислению свертки. Алгоритм Рейдера позволяет свести ДПФ к циклической свертке, если только длина преобразования является простым числом. Его можно использовать при вычислениях в любом поле.

Для переиндексации компонент входного вектора воспользуемся структурой поля  $GF(p) \cong \mathbb{Z}_p$ . ( $\mathbb{Z}_p^* \cong GF(p)^*$  при простом  $p$  является циклической группой порядка  $p - 1$ .) Пусть  $\pi$  — примитивный элемент этого поля, т. е. элемент порядка  $p - 1$ . Тогда каждое не превосходящее  $p$  целое положительное число можно однозначно записать в виде степени элемента  $\pi$ . Так как индексы  $j$  и  $k$  в формулах преобразования Фурье принимают и нулевые значения, то эти компоненты входного и выходного векторов удобно записывать отдельно:

$$\begin{aligned} V_0 &= \sum_{j=0}^{p-1} v_j, \\ V_k &= v_0 + \sum_{j=1}^{p-1} \omega^{jk} v_j \quad (k = 1, 2, \dots, p-1). \end{aligned} \tag{1}$$

Пусть  $r(j)$  обозначает единственное для каждого  $j$  от 1 до  $p - 1$  целое число такое, что в поле  $GF(p)$  выполняется равенство

$$\pi^{r(j)} = j.$$

Функция  $r(j)$  есть взаимно однозначное отображение множества  $\{1, 2, \dots, p-1\}$  на себя, т. е. она задает перестановку на множестве  $\{1, 2, \dots, p-1\}$ . Теперь  $V_k$  ( $k = 1, 2, \dots, p-1$ ) можно записать в виде

$$V_{\pi^{r(k)}} = v_0 + \sum_{j=1}^{p-1} \omega^{\pi^{r(j)+r(k)}} v_{\pi^{r(j)}}. \tag{2}$$

Введем обозначения:

$$l = r(k), \quad m = p - 1 - r(j).$$

Выберем  $m$  в качестве индекса суммирования, в результате (2) перепишется в виде

$$V_{\pi^l} = v_0 + \sum_{m=1}^{p-1} \omega^{\pi^{l-m}} v_{\pi^{p-1-m}} \tag{3}$$

так как  $\pi^{p-1} = 1$ .

Обозначим  $V_{\pi^l}$  через  $V'_l$  и  $v_{\pi^{p-1-m}}$  — через  $v'_m$ , тогда (3) принимает вид

$$V'_l = v_0 + \sum_{m=1}^{p-1} \omega^{\pi^{l-m}} v'_m. \tag{4}$$

Заметим, что при  $m = p - 1$  получаем в сумме (4) член  $\omega^{\pi^{l-(p-1)}} v_{\pi^0} = \omega^{\pi^l} v_1$ , при  $m = 0$  получаем  $\omega^{\pi^l} v'_0 = \omega^{\pi^l} v_{\pi^{p-1}} = \omega^{\pi^l} v_1$ , т. е. тот же член, поэтому пределы суммирования в формуле (4) можно изменить

$$V'_l = v_0 + \sum_{m=0}^{p-2} \omega^{\pi^{l-m}} v'_m. \tag{5}$$

Мы получили уравнение для вычисления циклической свертки последовательностей  $\{v'_m\}$  и  $\{\omega^{\pi^m}\}$ . Таким образом, переставляя индексы входных и выходных данных, мы записали преобразование Фурье в виде свертки. Однако для того вида, в котором выписаны формулы (5), необходимое число операций для вычисления свертки опять имеет порядок  $p^2$ , поэтому далее следует применить



быстрый алгоритм вычисления свертки. Далее (в § 18) мы скомбинируем алгоритм Рейдера для простой длины с алгоритмом Винограда для свертки и получим малый БПФ-алгоритм Винограда.

В качестве примера сведения преобразования Фурье простой длины к свертке рассмотрим случай  $p = 5$ .

Пусть требуется вычислить 5-точечное преобразование Фурье:

$$V_k = \sum_{j=0}^4 \omega^{jk} v_j \quad (k = 0, 1, \dots, 4),$$

где  $\omega = e^{-\frac{2i\pi}{5}}$ .

Сначала запишем формулы (1):

$$V_0 = \sum_{j=0}^4 v_j,$$

$$V_k = v_0 + \sum_{j=1}^4 \omega^{jk} v_j \quad (k = 1, 2, 3, 4).$$

На самом деле, удобно выделить компоненту  $V_0$  в каждой из остальных компонент  $V_k$  ( $k = 1, 2, 3, 4$ ), переписав последнюю формулу в виде

$$V_k = V_0 + \sum_{j=1}^4 (\omega^{jk} - 1) v_j \quad (k = 1, 2, 3, 4).$$

Далее можно заниматься только членами

$$V_k - V_0 = \sum_{j=1}^4 (\omega^{jk} - 1) v_j.$$

Примитивным элементом поля  $GF(5)$  является 2. Действительно,

$$2^0 \equiv 1, \quad 2^1 \equiv 2, \quad 2^2 \equiv 4, \quad 2^3 \equiv 3 \pmod{5} \quad (2^4 \equiv 1 \pmod{5}).$$

Кроме того, нам еще потребуются отрицательные степени примитивного элемента:

$$2^{-0} \equiv 1, \quad 2^{-1} \equiv 3, \quad 2^{-2} \equiv 4, \quad 2^{-3} \equiv 2 \pmod{5}.$$

Значит,

$$V'_l - V_0 = \sum_{j=0}^3 (\omega^{2^{l-j}} - 1) v'_j = \sum_{j=0}^3 (\omega^{2^{l-j}} - 1) v_{2^{-j}} \quad (l = 0, 1, 2, 3). \quad (6)$$

Так как

$$v'_0 = v_{2^{-0}} = v_1, \quad v'_1 = v_{2^{-1}} = v_3, \quad v'_2 = v_{2^{-2}} = v_4, \quad v'_3 = v_{2^{-3}} = v_2,$$

$$V'_0 = V_{2^0} = V_1, \quad V'_1 = V_{2^1} = V_2, \quad V'_2 = V_{2^2} = V_4, \quad V'_3 = V_{2^3} = V_3,$$

то (6) можно расписать следующим образом:

$$V'_0 - V_0 = V_1 - V_0 = \sum_{j=0}^3 (\omega^{2^{-j}} - 1) v'_{2^{-j}} = (\omega - 1) v_1 + (\omega^3 - 1) v_3 + (\omega^4 - 1) v_4 + (\omega^2 - 1) v_2,$$

$$V'_1 - V_0 = V_2 - V_0 = \sum_{j=0}^3 (\omega^{2^{1-j}} - 1) v'_{2^{-j}} = (\omega^2 - 1) v_1 + (\omega - 1) v_3 + (\omega^3 - 1) v_4 + (\omega^4 - 1) v_2, \quad (7)$$

$$V'_2 - V_0 = V_4 - V_0 = \sum_{j=0}^3 (\omega^{2^{2-j}} - 1) v'_{2^{-j}} = (\omega^4 - 1) v_1 + (\omega^2 - 1) v_3 + (\omega - 1) v_4 + (\omega^3 - 1) v_2,$$

$$V'_3 - V_0 = V_3 - V_0 = \sum_{j=0}^3 (\omega^{2^{3-j}} - 1) v'_{2^{-j}} = (\omega^3 - 1) v_1 + (\omega^4 - 1) v_3 + (\omega^2 - 1) v_4 + (\omega - 1) v_2.$$

Определим *фильтр Рейдера*, задавая его многочленом  $g(x)$  над полем  $\mathbb{C}$

$$g(x) = (\omega^3 - 1)x^3 + (\omega^4 - 1)x^2 + (\omega^2 - 1)x + (\omega - 1).$$

Пусть

$$d(x) = v_2 x^3 + v_4 x^2 + v_3 x + v_1.$$

Тогда формулы (7) представляют собой коэффициенты многочлена

$$s(x) = g(x) d(x) \pmod{x^4 - 1},$$

т. е. 4-точечную циклическую свертку последовательностей  $\mathbf{g} = \{\omega - 1, \omega^2 - 1, \omega^4 - 1, \omega^3 - 1\}$  и  $\mathbf{d} = \{v_1, v_3, v_4, v_2\}$ . При этом

$$s(x) = (V_3 - V_0)x^3 + (V_4 - V_0)x^2 + (V_2 - V_0)x + (V_1 - V_0).$$

В матричной записи полученную циклическую свертку можно записать следующим образом:

$$\begin{pmatrix} V_1 - V_0 \\ V_2 - V_0 \\ V_4 - V_0 \\ V_3 - V_0 \end{pmatrix} = \begin{pmatrix} \omega - 1 & \omega^3 - 1 & \omega^4 - 1 & \omega^2 - 1 \\ \omega^2 - 1 & \omega - 1 & \omega^3 - 1 & \omega^4 - 1 \\ \omega^4 - 1 & \omega^2 - 1 & \omega - 1 & \omega^3 - 1 \\ \omega^3 - 1 & \omega^4 - 1 & \omega^2 - 1 & \omega - 1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_3 \\ v_4 \\ v_2 \end{pmatrix}.$$

Таким образом, алгоритм Рейдера сводит 5-точечное преобразование Фурье к 4-точечной циклической свертке и некоторым дополнительным сложениям.

Поучительно переписать этот алгоритм в матричной форме, исходя из прямого вычисления по формулам

$$V_k = \sum_{j=0}^4 \omega^{jk} v_j \quad (k = 0, 1, \dots, 4).$$

Прямой алгоритм:

$$\begin{pmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \\ V_4 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 \\ 1 & \omega^2 & \omega^4 & \omega & \omega^3 \\ 1 & \omega^3 & \omega & \omega^4 & \omega^2 \\ 1 & \omega^4 & \omega^3 & \omega^2 & \omega \end{pmatrix} \begin{pmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix}.$$

Выделяем компоненту

$$V_0 = v_0 + v_1 + v_2 + v_3 + v_4$$

и вычитаем ее из остальных компонент:

$$\begin{pmatrix} V_1 - V_0 \\ V_2 - V_0 \\ V_3 - V_0 \\ V_4 - V_0 \end{pmatrix} = \begin{pmatrix} \omega - 1 & \omega^2 - 1 & \omega^3 - 1 & \omega^4 - 1 \\ \omega^2 - 1 & \omega^4 - 1 & \omega - 1 & \omega^3 - 1 \\ \omega^3 - 1 & \omega - 1 & \omega^4 - 1 & \omega^2 - 1 \\ \omega^4 - 1 & \omega^3 - 1 & \omega^2 - 1 & \omega - 1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix}.$$

Теперь переставляем в выходном векторе, а значит, и в матрице 3-ю и 4-ю строки:

$$\begin{pmatrix} V_1 - V_0 \\ V_2 - V_0 \\ V_4 - V_0 \\ V_3 - V_0 \end{pmatrix} = \begin{pmatrix} \omega - 1 & \omega^2 - 1 & \omega^3 - 1 & \omega^4 - 1 \\ \omega^2 - 1 & \omega^4 - 1 & \omega - 1 & \omega^3 - 1 \\ \omega^4 - 1 & \omega^3 - 1 & \omega^2 - 1 & \omega - 1 \\ \omega^3 - 1 & \omega - 1 & \omega^4 - 1 & \omega^2 - 1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix}.$$

Наконец, переставляем компоненты входного вектора вместе с соответствующими столбцами матрицы и приходим к 4-точечной циклической свертке:

$$\begin{pmatrix} V_1 - V_0 \\ V_2 - V_0 \\ V_4 - V_0 \\ V_3 - V_0 \end{pmatrix} = \begin{pmatrix} \omega - 1 & \omega^3 - 1 & \omega^4 - 1 & \omega^2 - 1 \\ \omega^2 - 1 & \omega - 1 & \omega^3 - 1 & \omega^4 - 1 \\ \omega^4 - 1 & \omega^2 - 1 & \omega - 1 & \omega^3 - 1 \\ \omega^3 - 1 & \omega^4 - 1 & \omega^2 - 1 & \omega - 1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_3 \\ v_4 \\ v_2 \end{pmatrix}.$$



§16. АЛГОРИТМ РЕЙДЕРА В СЛУЧАЕ, КОГДА ДЛИНА  
ПРЕОБРАЗОВАНИЯ РАВНА СТЕПЕНИ ПРОСТОГО НЕЧЕТНОГО ЧИСЛА

Идею алгоритма Рейдера можно перенести на случай, когда длина  $n$  преобразования Фурье равна степени простого нечетного числа. Рассмотрим кольцо  $\mathbb{Z}_{p^m}$ . Поскольку мультипликативная группа  $\mathbb{Z}_{p^m}^*$  этого кольца имеет порядок  $p^{m-1}(p-1)$  и является циклической, то в  $\mathbb{Z}_{p^m}$  существует элемент  $\pi$  порядка  $p^{m-1}(p-1)$ , но нет элемента порядка  $p^m-1$ . Поэтому не все ненулевые элементы кольца  $\mathbb{Z}_{p^m}$  могут быть представлены как степени элемента  $\pi$ .

При вычислении

$$V_k = \sum_{j=0}^{p^m-1} \omega^{jk} v_j \quad (k = 0, 1, \dots, p^m - 1)$$

из  $(p^m \times p^m)$ -матрицы  $W = (\omega^{jk})$  удалим все вызывающие трудности строки и столбцы так, чтобы к оставшейся матрице порядка  $p^{m-1}(p-1)$  была применима идея Рейдера. Т. е. следует удалить те строки и столбцы, которые соответствуют компонентам входного и выходного векторов, индексы которых не являются элементами группы  $\mathbb{Z}_{p^m}^*$ . Выброшенные строки и столбцы обрабатываются отдельно. Обработку этих строк и столбцов можно организовать как вычисление еще меньших циклических сверток.

Рассмотрим, например, 9-точечное преобразование Фурье. Группа  $\mathbb{Z}_{3^2}^*$  является циклической и имеет порядок 6, значит, можно найти элемент порядка 6. В  $\mathbb{Z}_{3^2}^*$  не попали следующие элементы кольца  $\mathbb{Z}_9$ : 0, 3, 6. Значит, из исходной матрицы  $W$  будем выбрасывать строки, соответствующие компонентам выходного вектора  $V_0, V_3, V_6$ , и столбцы, соответствующие компонентам входного вектора  $v_0, v_3, v_6$ :

$$\begin{aligned} V_0 &= v_0 + v_1 + v_2 + v_3 + v_4 + v_5 + v_6 + v_7 + v_8, \\ V_3 &= v_0 + \omega^3 v_1 + \omega^6 v_2 + v_3 + \omega^3 v_4 + \omega^6 v_5 + v_6 + \omega^3 v_7 + \omega^6 v_8, \\ V_6 &= v_0 + \omega^6 v_1 + \omega^3 v_2 + v_3 + \omega^6 v_4 + \omega^3 v_5 + v_6 + \omega^6 v_7 + \omega^3 v_8, \\ V_k &= V_k^1 + V_k^2 \quad (k = 1, 2, 4, 5, 7, 8), \end{aligned} \tag{1}$$

где

$$V_k^1 = \sum_{\substack{j=1 \\ j \neq 3,6}}^8 \omega^{jk} v_j, \quad V_k^2 = \sum_{\{j=0,3,6\}} \omega^{jk} v_j.$$

Кроме того, если  $V_0$  также разбить на две части

$$V_0 = (v_1 + v_2 + v_4 + v_5 + v_7 + v_8) + (v_0 + v_3 + v_6) = V_0^1 + V_0^2$$

и рассматривать вместо каждой компоненты  $V_k$  разность  $V_k - V_0$ , т. е.

$$V_k - V_0 = (V_k^1 - V_0^1) + (V_k^2 - V_0^2) \quad (k = 1, 2, 4, 5, 7, 8),$$

то в матричном виде вычисления могут быть записаны так:

$$\begin{pmatrix} V_1^1 - V_0^1 \\ V_2^1 - V_0^1 \\ V_4^1 - V_0^1 \\ V_5^1 - V_0^1 \\ V_7^1 - V_0^1 \\ V_8^1 - V_0^1 \end{pmatrix} = \begin{pmatrix} \omega - 1 & \omega^2 - 1 & \omega^4 - 1 & \omega^5 - 1 & \omega^7 - 1 & \omega^8 - 1 \\ \omega^2 - 1 & \omega^4 - 1 & \omega^8 - 1 & \omega - 1 & \omega^5 - 1 & \omega^7 - 1 \\ \omega^4 - 1 & \omega^8 - 1 & \omega^7 - 1 & \omega^2 - 1 & \omega - 1 & \omega^5 - 1 \\ \omega^5 - 1 & \omega - 1 & \omega^2 - 1 & \omega^7 - 1 & \omega^8 - 1 & \omega^4 - 1 \\ \omega^7 - 1 & \omega^5 - 1 & \omega - 1 & \omega^8 - 1 & \omega^4 - 1 & \omega^2 - 1 \\ \omega^8 - 1 & \omega^7 - 1 & \omega^5 - 1 & \omega^4 - 1 & \omega^2 - 1 & \omega - 1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_4 \\ v_5 \\ v_7 \\ v_8 \end{pmatrix} \tag{2}$$

( $\omega^9 = 1$ ).

Поскольку 2 является образующей группы  $\mathbb{Z}_{3^2}^*$ , то рассмотрим ее в качестве элемента  $\pi$ :

$$2^0 \equiv 1, \quad 2^1 \equiv 2, \quad 2^2 \equiv 4, \quad 2^3 \equiv 8, \quad 2^4 \equiv 7, \quad 2^5 \equiv 5 \pmod{9},$$



$$2^{-0} \equiv 1, \quad 2^{-1} \equiv 5, \quad 2^{-2} \equiv 7, \quad 2^{-3} \equiv 8, \quad 2^{-4} \equiv 4, \quad 2^{-5} \equiv 2 \pmod{9}.$$

Переставляем компоненты выходного вектора (а значит, и строки матрицы в формуле (2)) в соответствии с положительными степенями двойки, а компоненты выходного вектора (а значит, и столбцы матрицы в формуле (2)) – в соответствии с отрицательными степенями двойки. В результате получаем

$$\begin{pmatrix} V_1^1 - V_0^1 \\ V_2^1 - V_0^1 \\ V_4^1 - V_0^1 \\ V_8^1 - V_0^1 \\ V_7^1 - V_0^1 \\ V_5^1 - V_0^1 \end{pmatrix} = \begin{pmatrix} \omega - 1 & \omega^5 - 1 & \omega^7 - 1 & \omega^8 - 1 & \omega^4 - 1 & \omega^2 - 1 \\ \omega^2 - 1 & \omega - 1 & \omega^5 - 1 & \omega^7 - 1 & \omega^8 - 1 & \omega^4 - 1 \\ \omega^4 - 1 & \omega^2 - 1 & \omega - 1 & \omega^5 - 1 & \omega^7 - 1 & \omega^8 - 1 \\ \omega^8 - 1 & \omega^4 - 1 & \omega^2 - 1 & \omega - 1 & \omega^5 - 1 & \omega^7 - 1 \\ \omega^7 - 1 & \omega^8 - 1 & \omega^4 - 1 & \omega^2 - 1 & \omega - 1 & \omega^5 - 1 \\ \omega^5 - 1 & \omega^7 - 1 & \omega^8 - 1 & \omega^4 - 1 & \omega^2 - 1 & \omega - 1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_5 \\ v_7 \\ v_8 \\ v_4 \\ v_2 \end{pmatrix},$$

т. е. 6-точечную циклическую свертку или произведение двух многочленов  $d(x)$  и  $g(x)$  по модулю многочлена  $x^6 - 1$ , где

$$d(x) = v_2 x^5 + v_4 x^4 + v_8 x^3 + v_7 x^2 + v_5 x + v_1,$$

$$g(x) = (\omega^5 - 1)x^5 + (\omega^7 - 1)x^4 + (\omega^8 - 1)x^3 + (\omega^4 - 1)x^2 + (\omega^2 - 1)x + (\omega - 1),$$

$$s(x) = d(x)g(x) \pmod{x^6 - 1} =$$

$$(V_5^1 - V_0^1)x^5 + (V_7^1 - V_0^1)x^4 + (V_8^1 - V_0^1)x^3 + (V_4^1 - V_0^1)x^2 + (V_2^1 - V_0^1)x + (V_1^1 - V_0^1).$$

Рассмотрим оставшиеся вторые части компонент  $V_k - V_0$  ( $k = 1, 2, 4, 5, 7, 8$ ). Так как

$$\begin{pmatrix} V_1^2 - V_0^2 \\ V_2^2 - V_0^2 \\ V_4^2 - V_0^2 \\ V_5^2 - V_0^2 \\ V_7^2 - V_0^2 \\ V_8^2 - V_0^2 \end{pmatrix} = \begin{pmatrix} \omega^3 - 1 & \omega^6 - 1 \\ \omega^6 - 1 & \omega^3 - 1 \\ \omega^3 - 1 & \omega^6 - 1 \\ \omega^6 - 1 & \omega^3 - 1 \\ \omega^3 - 1 & \omega^6 - 1 \\ \omega^6 - 1 & \omega^3 - 1 \end{pmatrix} \begin{pmatrix} v_3 \\ v_6 \end{pmatrix},$$

то

$$V_1^2 - V_0^2 = V_4^2 - V_0^2 = V_7^2 - V_0^2,$$

$$V_2^2 - V_0^2 = V_5^2 - V_0^2 = V_8^2 - V_0^2,$$

и вычислять нужно только две компоненты

$$\begin{pmatrix} V_1^2 - V_0^2 \\ V_2^2 - V_0^2 \end{pmatrix} = \begin{pmatrix} \omega^3 - 1 & \omega^6 - 1 \\ \omega^6 - 1 & \omega^3 - 1 \end{pmatrix} \begin{pmatrix} v_3 \\ v_6 \end{pmatrix}. \quad (3)$$

Как видно из (3), вычисление этих компонент сводится к вычислению 2-точечной циклической свертки, т. е.

$$\tilde{s}(x) = (V_2^2 - V_0^2)x + (V_1^2 - V_0^2) = \tilde{d}(x)\tilde{g}(x) \pmod{x^2 - 1},$$

где  $\tilde{d}(x) = v_6 x + v_3$ ,  $\tilde{g}(x) = (\omega^6 - 1)x + (\omega^3 - 1)$ .

Если вычислять вместо  $V_3, V_6$  разности  $V_3 - V_0, V_6 - V_0$ , то из формул (1) следует, что

$$\begin{pmatrix} V_3 - V_0 \\ V_6 - V_0 \end{pmatrix} = \begin{pmatrix} \omega^3 - 1 & \omega^6 - 1 \\ \omega^6 - 1 & \omega^3 - 1 \end{pmatrix} \begin{pmatrix} v_1 + v_4 + v_7 \\ v_2 + v_5 + v_8 \end{pmatrix}, \quad (4)$$

т. е. получаем еще одну 2-точечную циклическую свертку:

$$\hat{s}(x) = (V_6 - V_0)x + (V_3 - V_0) = \hat{d}(x)\hat{g}(x) \pmod{x^2 - 1},$$

где

$$\hat{d}(x) = (v_2 + v_5 + v_8)x + (v_1 + v_4 + v_7),$$

$$\hat{g}(x) = (\omega^6 - 1)x + (\omega^3 - 1).$$

Таким образом, вычисление 9-точечного преобразования Фурье свелось к вычислению 6-точечной циклической свертки, двум 2-точечным циклическим сверткам и некоторым дополнительным сложениям.



# §17. АЛГОРИТМ РЕЙДЕРА В СЛУЧАЕ, КОГДА ДЛИНА ПРЕОБРАЗОВАНИЯ РАВНА СТЕПЕНИ ДВОЙКИ

Случай, когда длина преобразования Фурье равна степени двойки, более сложен и требует еще одного уровня вычислений. Это объясняется тем, что мультипликативная группа кольца  $\mathbb{Z}_{2^m}$  при  $m \geq 3$  не является циклической. В группу  $\mathbb{Z}_{2^m}^*$  попадают все нечетные элементы (индексы входных и выходных компонент). При  $m \geq 3$  группа  $\mathbb{Z}_{2^m}^*$  изоморфна декартову произведению циклической группы порядка 2 и циклической группы порядка  $2^{m-2}$ . Этот изоморфизм используется для того, чтобы определить такую перестановку строк и столбцов матрицы  $\tilde{W} = (w^{jk})$  ( $j = 2s+1$ ,  $k = 2l+1$ ;  $s, l = 0, 1, \dots, 2^{m-1} - 1$ ), которая приводит ее к клеточному виду

$$\tilde{W} = \begin{pmatrix} W_1 & W_2 \\ W_2 & W_1 \end{pmatrix}, \quad (1)$$

где  $W_1$  и  $W_2$  – квадратные матрицы порядка  $2^{m-2}$ , каждая из которых задает структуру циклической свертки. Матрица же  $\tilde{W}$  есть квадратная матрица порядка  $2^{m-1}$ , полученная из  $2^m \times 2^m$ -матрицы  $W = (w^{jk})$  ( $j, k = 0, 1, \dots, 2^m - 1$ ) после выбрасывания из нее строк и столбцов, соответствующих компонентам выходного и входного векторов с четными индексами.

В качестве образующей циклической группы порядка  $2^{m-2}$  возьмем  $\pi = 3$  (можно взять и  $\pi = 5$  см. § 16 [3]), а в качестве образующей циклической группы порядка 2 – элемент  $\sigma = 2^m - 1$ . На самом деле мультипликативная группа  $\mathbb{Z}_{2^m}^*$  порождается элементами  $\pi$  и  $\sigma$ , т. е. каждый элемент этой группы может быть однозначно представлен в виде

$$\sigma^l \pi^r,$$

где  $l = 0, 1$  и  $r = 0, 1, \dots, 2^{m-2} - 1$ . Значит, если  $j = \sigma^{l_1} \pi^{r_1}$  и  $k = \sigma^{l_2} \pi^{r_2}$ , то

$$\omega^{jk} = \omega^{\sigma^{l_1} \pi^{r_1} \sigma^{l_2} \pi^{r_2}} = \omega^{\sigma^{l_1+l_2} \pi^{r_1+r_2}}$$

и подходящей перестановкой строк и столбцов матрица  $\tilde{W}$  приводится к виду

$$\tilde{W} = \begin{pmatrix} \begin{pmatrix} \omega^{\pi^{r_1+r_2}} \\ \omega^{\sigma \pi^{r_1+r_2}} \end{pmatrix} & \begin{pmatrix} \omega^{\sigma \pi^{r_1+r_2}} \\ \omega^{\pi^{r_1+r_2}} \end{pmatrix} \end{pmatrix},$$

где  $r_1$  и  $r_2$  соответственно индексы строк и столбцов в каждой из подматриц. Каждая из четырех подматриц задает циклическую свертку длины  $2^{m-2}$ .

После выполнения перестановки рассматриваемое вычисление приводится к виду *матричной* 2-точечной циклической свертки

$$\begin{pmatrix} \tilde{V}_1 \\ \tilde{V}_2 \end{pmatrix} = \begin{pmatrix} W_1 & W_2 \\ W_2 & W_1 \end{pmatrix} \begin{pmatrix} \tilde{v}_1 \\ \tilde{v}_2 \end{pmatrix},$$

один из способов вычисления которой дается формулой

$$\begin{pmatrix} \tilde{V}_1 \\ \tilde{V}_2 \end{pmatrix} = \begin{pmatrix} E & E \\ E & -E \end{pmatrix} \begin{pmatrix} \frac{1}{2}(W_1 + W_2) & O \\ O & \frac{1}{2}(W_1 - W_2) \end{pmatrix} \begin{pmatrix} E & E \\ E & -E \end{pmatrix} \begin{pmatrix} \tilde{v}_1 \\ \tilde{v}_2 \end{pmatrix}. \quad (2)$$

Таким образом, задача сводится к вычислению двух комплексных циклических сверток длины  $2^{m-2}$ .

Выброшенные же из исходной матрицы строки и столбцы обрабатываются отдельно, причем эти вычисления также сводятся к вычислению циклических сверток еще меньшей длины.

В качестве примера рассмотрим 16-точечное преобразование Фурье. Пусть

$$V_k = \sum_{j=0}^{15} \omega^{jk} v_j \quad (k = 0, 1, \dots, 15), \quad (3)$$



где  $\omega^{16} = 1$  (т. е.  $\omega = e^{-\frac{2i\pi}{16}}$ ).

Разобьем сразу сумму в (3) на две части, в первой оставим только слагаемые, соответствующие нечетным индексам входного вектора, а во второй – слагаемые, соответствующие четным индексам входного вектора, обозначив эти части соответственно  $V_k^1$  и  $V_k^2$ . Т. е.

$$V_k = \sum_{j=0}^7 \omega^{(2j+1)k} v_{2j+1} + \sum_{j=0}^7 \omega^{2jk} v_{2j} = V_k^1 + V_k^2 \quad (k = 0, 1, \dots, 15). \quad (4)$$

Сначала займемся вычислением компонент  $V_k^1$  в случае, когда  $k$  – нечетно, т. е. рассмотрим матрицу  $\tilde{W}$ , образуемую в результате выбрасывания из  $W$  всех ее строк и столбцов, соответствующих четным индексам входного и выходного векторов:

$$\begin{pmatrix} V_1^1 \\ V_3^1 \\ V_5^1 \\ V_7^1 \\ V_9^1 \\ V_{11}^1 \\ V_{13}^1 \\ V_{15}^1 \end{pmatrix} = \begin{pmatrix} \omega & \omega^3 & \omega^5 & \omega^7 & \omega^9 & \omega^{11} & \omega^{13} & \omega^{15} \\ \omega^3 & \omega^9 & \omega^{15} & \omega^5 & \omega^{11} & \omega & \omega^7 & \omega^{13} \\ \omega^5 & \omega^{15} & \omega^9 & \omega^3 & \omega^{13} & \omega^7 & \omega & \omega^{11} \\ \omega^7 & \omega^5 & \omega^3 & \omega & \omega^{15} & \omega^{13} & \omega^{11} & \omega^9 \\ \omega^9 & \omega^{11} & \omega^{13} & \omega^{15} & \omega & \omega^3 & \omega^5 & \omega^7 \\ \omega^{11} & \omega & \omega^7 & \omega^{13} & \omega^3 & \omega^9 & \omega^{15} & \omega^5 \\ \omega^{13} & \omega^7 & \omega & \omega^{11} & \omega^5 & \omega^{15} & \omega^9 & \omega^3 \\ \omega^{15} & \omega^{13} & \omega^{11} & \omega^9 & \omega^7 & \omega^5 & \omega^3 & \omega \end{pmatrix} \begin{pmatrix} v_1 \\ v_3 \\ v_5 \\ v_7 \\ v_9 \\ v_{11} \\ v_{13} \\ v_{15} \end{pmatrix}. \quad (4)$$

Чтобы найти нужную перестановку, выпишем индексы в виде чисел  $15^l 5^r$  для  $l = 0, 1$  и  $r = 0, 1, 2, 3$ . Степени числа 5 по модулю 16 имеют вид:

$$\begin{aligned} 5^0 &\equiv 1, \quad 5^1 \equiv 5, \quad 5^2 \equiv 9, \quad 5^3 \equiv 13 \pmod{16}, \\ 5^{-0} &\equiv 1, \quad 5^{-1} \equiv 13, \quad 5^{-2} \equiv 9, \quad 5^{-3} \equiv 5 \pmod{16} \end{aligned}$$

и, значит,

$$\begin{aligned} 15 \cdot 5^0 &\equiv 15, \quad 15 \cdot 5^1 \equiv 11, \quad 15 \cdot 5^2 \equiv 7, \quad 15 \cdot 5^3 \equiv 3 \pmod{16}, \\ 15 \cdot 5^{-0} &\equiv 15, \quad 15 \cdot 5^{-1} \equiv 3, \quad 15 \cdot 5^{-2} \equiv 7, \quad 15 \cdot 5^{-3} \equiv 11 \pmod{16}. \end{aligned}$$

Выполним перестановку входных индексов, записывая их в порядке  $15^l 5^{-r} \pmod{16}$ , и перестановку выходных индексов, записывая их в порядке  $15^l 5^r \pmod{16}$ . Тогда формула (4) примет вид

$$\begin{pmatrix} V_1^1 \\ V_5^1 \\ V_9^1 \\ V_{13}^1 \\ V_{15}^1 \\ V_{11}^1 \\ V_7^1 \\ V_3^1 \end{pmatrix} = \begin{pmatrix} \omega & \omega^{13} & \omega^9 & \omega^5 & \omega^{15} & \omega^3 & \omega^7 & \omega^{11} \\ \omega^5 & \omega & \omega^{13} & \omega^9 & \omega^{11} & \omega^{15} & \omega^3 & \omega^7 \\ \omega^9 & \omega^5 & \omega & \omega^{13} & \omega^7 & \omega^{11} & \omega^{15} & \omega^3 \\ \omega^{13} & \omega^9 & \omega^5 & \omega & \omega^3 & \omega^7 & \omega^{11} & \omega^{15} \\ \omega^{15} & \omega^3 & \omega^7 & \omega^{11} & \omega & \omega^{13} & \omega^9 & \omega^5 \\ \omega^{11} & \omega^{15} & \omega^3 & \omega^7 & \omega^5 & \omega & \omega^{13} & \omega^9 \\ \omega^7 & \omega^{11} & \omega^{15} & \omega^3 & \omega^9 & \omega^5 & \omega & \omega^{13} \\ \omega^3 & \omega^7 & \omega^{11} & \omega^{15} & \omega^{13} & \omega^9 & \omega^5 & \omega \end{pmatrix} \begin{pmatrix} v_1 \\ v_{13} \\ v_9 \\ v_5 \\ v_{15} \\ v_3 \\ v_7 \\ v_{11} \end{pmatrix}. \quad (5)$$

Если преобразование Фурье вычисляется в поле комплексных чисел, то полученные блоки матрицы  $\tilde{W}$  связаны соотношением комплексной сопряженности. Действительно, так как  $\omega^{16} = 1$ , то

$$\omega^{15} = \omega^{-1}, \quad \omega^3 = \omega^{-13}, \quad \omega^7 = \omega^{-9}, \quad \omega^{11} = \omega^{-5},$$

поэтому матричное уравнение (5) перепишется в виде

$$\begin{pmatrix} V_1^1 \\ V_5^1 \\ V_9^1 \\ V_{13}^1 \\ V_{15}^1 \\ V_{11}^1 \\ V_7^1 \\ V_3^1 \end{pmatrix} = \begin{pmatrix} \omega & \omega^{13} & \omega^9 & \omega^5 & \omega^{-1} & \omega^{-13} & \omega^{-9} & \omega^{-5} \\ \omega^5 & \omega & \omega^{13} & \omega^9 & \omega^{-5} & \omega^{-1} & \omega^{-13} & \omega^{-9} \\ \omega^9 & \omega^5 & \omega & \omega^{13} & \omega^{-9} & \omega^{-5} & \omega^{-1} & \omega^{-13} \\ \omega^{13} & \omega^9 & \omega^5 & \omega & \omega^{-13} & \omega^{-9} & \omega^{-5} & \omega^{-1} \\ \omega^{-1} & \omega^{-13} & \omega^{-9} & \omega^{-5} & \omega & \omega^{13} & \omega^9 & \omega^5 \\ \omega^{-5} & \omega^{-1} & \omega^{-13} & \omega^{-9} & \omega^5 & \omega & \omega^{13} & \omega^9 \\ \omega^{-9} & \omega^{-5} & \omega^{-1} & \omega^{-13} & \omega^9 & \omega^5 & \omega & \omega^{13} \\ \omega^{-13} & \omega^{-9} & \omega^{-5} & \omega^{-1} & \omega^{13} & \omega^9 & \omega^5 & \omega \end{pmatrix} \begin{pmatrix} v_1 \\ v_{13} \\ v_9 \\ v_5 \\ v_{15} \\ v_3 \\ v_7 \\ v_{11} \end{pmatrix},$$



но

$$\omega^{-1} = \bar{\omega}, \quad \omega^{-13} = \overline{\omega^{13}}, \quad \omega^{-9} = \overline{\omega^9}, \quad \omega^{-5} = \overline{\omega^5}, \quad \omega = \cos \left( -\frac{2\pi}{16} \right) + i \sin \left( -\frac{2\pi}{16} \right).$$

Кроме того, верхние четыре строки матрицы комплексно сопряжены с четырьмя нижними строками, поэтому если компоненты входного вектора  $v$  вещественны, то выполнять надо только вычисления, связанные с первыми четырьмя строками. Значит, дальнейшие вычисления можно организовать в виде пары циклических сверток

$$V_{13}^1 x^3 + V_9^1 x^2 + V_5^1 x + V_1^1 = (\omega^{13} x^3 + \omega^9 x^2 + \omega^5 x + \omega) (v_5 x^3 + v_9 x^2 + v_{13} x + v_1) +$$

$$(\omega^{-13} x^3 + \omega^{-9} x^2 + \omega^{-5} x + \omega^{-1}) (v_{11} x^3 + v_7 x^2 + v_3 x + v_{15}) \pmod{x^4 - 1}.$$

Для поля  $\mathbb{C}$  можно использовать и альтернативный способ, основанный на формуле (2). Посмотрим, что в этом случае представляют из себя матричные блоки  $\frac{1}{2}(W_1 + W_2)$  и  $\frac{1}{2}(W_1 - W_2)$ . Так как

$$\frac{1}{2}(\omega^k + \omega^{-k}) = \frac{1}{2}\{(\cos k\theta + i \sin k\theta) + (\cos k\theta - i \sin k\theta)\} = \cos k\theta,$$

$$\frac{1}{2}(\omega^k - \omega^{-k}) = \frac{1}{2}\{(\cos k\theta + i \sin k\theta) - (\cos k\theta - i \sin k\theta)\} = i \sin k\theta \quad (\theta = -\frac{2\pi}{16}),$$

то

$$\frac{1}{2}(W_1 + W_2) = \begin{pmatrix} \cos \theta & \cos 13\theta & \cos 9\theta & \cos 5\theta \\ \cos 5\theta & \cos \theta & \cos 13\theta & \cos 9\theta \\ \cos 9\theta & \cos 5\theta & \cos \theta & \cos 13\theta \\ \cos 13\theta & \cos 9\theta & \cos 5\theta & \cos \theta \end{pmatrix} \quad (6)$$

и

$$\frac{1}{2}(W_1 - W_2) = i \begin{pmatrix} \sin \theta & \sin 13\theta & \sin 9\theta & \sin 5\theta \\ \sin 5\theta & \sin \theta & \sin 13\theta & \sin 9\theta \\ \sin 9\theta & \sin 5\theta & \sin \theta & \sin 13\theta \\ \sin 13\theta & \sin 9\theta & \sin 5\theta & \sin \theta \end{pmatrix}. \quad (7)$$

Мы получили две 4-точечные циклические свертки, одна из которых чисто вещественная, а другая – чисто мнимая.

В случае вещественного входного вектора необходимо вычислить только две вещественные свертки:

$$s_1(x) = (\cos 13\theta x^3 + \cos 9\theta x^2 + \cos 5\theta x + \cos \theta) d_1(x) \pmod{x^4 - 1}, \quad (8)$$

$$s_2(x) = (\sin 13\theta x^3 + \sin 9\theta x^2 + \sin 5\theta x + \sin \theta) d_2(x) \pmod{x^4 - 1}, \quad (9)$$

где

$$d_1(x) = (v_5 + v_{11}) x^3 + (v_9 + v_7) x^2 + (v_{13} + v_3) x + (v_1 + v_{15}),$$

$$d_2(x) = (v_5 - v_{11}) x^3 + (v_9 - v_7) x^2 + (v_{13} - v_3) x + (v_1 - v_{15}).$$

При вычислении сверток (8) и (9) можно воспользоваться алгоритмом Винограда, использующим китайскую теорему об остатках и разложение

$$x^4 - 1 = (x^2 - 1)(x^2 + 1).$$

Заметим, что вычисления, связанные с модулем  $x^2 - 1$ , излишни. Действительно,

$$\cos 13\theta x^3 + \cos 9\theta x^2 + \cos 5\theta x + \cos \theta \pmod{x^2 - 1} = (\cos 13\theta - \cos 5\theta) x + (\cos 9\theta + \cos \theta) =$$

$$2 \cos 9\theta \cos 4\theta x + 2 \cos 5\theta \cos 4\theta = 2 \cos 4\theta (\cos 9\theta x + \cos 5\theta) = 0.$$

и

$$\sin 13\theta x^3 + \sin 9\theta x^2 + \sin 5\theta x + \sin \theta \pmod{x^2 - 1} = \sin 13\theta - \sin 5\theta x - \sin 9\theta + \sin \theta =$$



$$2 \sin 9\theta \cos 4\theta x + 2 \sin 5\theta \cos 4\theta = 2 \cos 4\theta (\sin 9\theta x + \sin 5\theta) = 0,$$

так как  $4\theta = -\frac{\pi}{2}$ .

Таким образом, умножения, связанные с модулем  $x^2 - 1$ , не нужны. Вычеты же по модулю  $x^2 + 1$  потребуют трех умножений для каждой из сверток (8) и (9), следовательно, для вычисления этих сверток потребуется всего 6 умножений, причем все эти умножения являются вещественными, если вектор  $v$  имеет вещественные компоненты.

Перейдем теперь к вычислению компонент  $V_k^2$  в случае нечетного  $k$ , т. е. рассмотрим матрицу, составленную из выброшенных выше столбцов, соответствующих четным компонентам входного вектора

$$\begin{pmatrix} V_1^2 \\ V_3^2 \\ V_5^2 \\ V_7^2 \\ V_9^2 \\ V_{11}^2 \\ V_{13}^2 \\ V_{15}^2 \end{pmatrix} = \begin{pmatrix} 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} & \omega^{14} \\ 1 & \omega^6 & \omega^{12} & \omega^2 & \omega^8 & \omega^{14} & \omega^4 & \omega^{10} \\ 1 & \omega^{10} & \omega^4 & \omega^{14} & \omega^8 & \omega^2 & \omega^{12} & \omega^6 \\ 1 & \omega^{14} & \omega^{12} & \omega^{10} & \omega^8 & \omega^6 & \omega^4 & \omega^2 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} & \omega^{14} \\ 1 & \omega^6 & \omega^{12} & \omega^2 & \omega^8 & \omega^{14} & \omega^4 & \omega^{10} \\ 1 & \omega^{10} & \omega^4 & \omega^{14} & \omega^8 & \omega^2 & \omega^{12} & \omega^6 \\ 1 & \omega^{14} & \omega^{12} & \omega^{10} & \omega^8 & \omega^6 & \omega^4 & \omega^2 \end{pmatrix} \begin{pmatrix} v_0 \\ v_2 \\ v_4 \\ v_6 \\ v_8 \\ v_{10} \\ v_{12} \\ v_{14} \end{pmatrix}. \quad (10)$$

Заметим, что последние четыре строки матрицы совпадают с первыми четырьмя строками, значит, надо выполнять вычисления, связанные только с первыми четырьмя строками. Так как  $\omega^8 = -1$ ,  $\omega^4 = -i$ ,  $\omega^{12} = i$ , то эти первые четыре строки можно записать в виде

$$\begin{pmatrix} V_1^2 \\ V_3^2 \\ V_5^2 \\ V_7^2 \end{pmatrix} = \begin{pmatrix} 1 & \omega^2 & -i & \omega^6 & -1 & -\omega^2 & i & -\omega^6 \\ 1 & \omega^6 & i & \omega^2 & -1 & -\omega^6 & -i & -\omega^2 \\ 1 & -\omega^2 & -i & -\omega^6 & -1 & \omega^2 & i & \omega^6 \\ 1 & -\omega^6 & i & -\omega^2 & -1 & \omega^6 & -i & \omega^2 \end{pmatrix} \begin{pmatrix} v_0 \\ v_2 \\ v_4 \\ v_6 \\ v_8 \\ v_{10} \\ v_{12} \\ v_{14} \end{pmatrix}.$$

Столбцы, соответствующие компонентам  $v_0$ ,  $v_4$ ,  $v_8$  и  $v_{12}$ , не содержат умножений, поэтому можно вынести константы, связанные с ними, например, вводя

$$T_1 = v_0 - v_8, \quad T_2 = v_4 - v_{12}, \quad T_3 = T_1 - iT_2, \quad T_4 = T_1 + iT_2,$$

тогда

$$\begin{aligned} V_1^2 &= T_3 + \tilde{V}_1^2, \\ V_3^2 &= T_4 + \tilde{V}_3^2, \\ V_5^2 &= T_3 + \tilde{V}_5^2, \\ V_7^2 &= T_4 + \tilde{V}_7^2 \end{aligned}$$

и

$$\begin{pmatrix} \tilde{V}_1^2 \\ \tilde{V}_3^2 \\ \tilde{V}_5^2 \\ \tilde{V}_7^2 \end{pmatrix} = \begin{pmatrix} \omega^2 & \omega^6 & -\omega^2 & -\omega^6 \\ \omega^6 & \omega^2 & -\omega^6 & -\omega^2 \\ -\omega^2 & -\omega^6 & \omega^2 & \omega^6 \\ -\omega^6 & -\omega^2 & \omega^6 & \omega^2 \end{pmatrix} \begin{pmatrix} v_2 \\ v_6 \\ v_{10} \\ v_{14} \end{pmatrix}. \quad (11)$$

Очевидно, что следует выполнить вычисления, связанные только с первыми двумя строками матричного уравнения (11), поскольку последние две строки отличаются от первых двух только знаком. Кроме того, последние два столбца отличаются от первых двух тоже только знаком, поэтому вычисления сводятся к виду

$$\begin{pmatrix} \tilde{V}_1^2 \\ \tilde{V}_3^2 \end{pmatrix} = \begin{pmatrix} \omega^2 & \omega^6 \\ \omega^6 & \omega^2 \end{pmatrix} \begin{pmatrix} v_2 - v_{10} \\ v_6 - v_{14} \end{pmatrix},$$



т. е. к 2-точечной циклической свертке. Вычисление такой свертки, как мы знаем, требует двух операций умножения. Если использовать разложение матрицы по формуле (2), то эти умножения становятся вещественными. Действительно,

$$\begin{pmatrix} \tilde{V}_1^2 \\ \tilde{V}_3^2 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} \frac{1}{2}(\omega^2 + \omega^6) & 0 \\ 0 & \frac{1}{2}(\omega^2 - \omega^6) \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} v_2 - v_{10} \\ v_6 - v_{14} \end{pmatrix},$$

где

$$\begin{aligned} \frac{1}{2}(\omega^2 + \omega^6) &= \frac{1}{2}\omega^4(\omega^2 + \omega^{-2}) = -i \cos 2\theta = -i \cos \frac{\pi}{4}, \\ \frac{1}{2}(\omega^2 - \omega^6) &= -\frac{1}{2}\omega^4(\omega^2 - \omega^{-2}) = i(i \sin 2\theta) = -\sin \frac{\pi}{4}. \end{aligned}$$

Таким образом, вычисление компонент выходного вектора с нечетными индексами требует выполнения всего 8 операций умножения.

Переходим к вычислению компонент выходного вектора с четными индексами. Сначала вычислим вклад столбцов, соответствующих компонентам входного вектора с нечетными индексами, т. е. вычислим компоненты  $V_{2k}^1$  ( $k = 0, 1, \dots, 7$ ):

$$\begin{pmatrix} V_0^1 \\ V_2^1 \\ V_4^1 \\ V_6^1 \\ V_8^1 \\ V_{10}^1 \\ V_{12}^1 \\ V_{14}^1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \omega^2 & \omega^6 & \omega^{10} & \omega^{14} & \omega^2 & \omega^6 & \omega^{10} & \omega^{14} \\ \omega^4 & \omega^{12} & \omega^4 & \omega^{12} & \omega^4 & \omega^{12} & \omega^4 & \omega^{12} \\ \omega^6 & \omega^2 & \omega^{14} & \omega^{10} & \omega^6 & \omega^2 & \omega^{14} & \omega^{10} \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ \omega^{10} & \omega^{14} & \omega^2 & \omega^6 & \omega^{10} & \omega^{14} & \omega^2 & \omega^6 \\ \omega^{12} & \omega^4 & \omega^{12} & \omega^4 & \omega^{12} & \omega^4 & \omega^{12} & \omega^4 \\ \omega^{14} & \omega^{10} & \omega^6 & \omega^2 & \omega^{14} & \omega^{10} & \omega^6 & \omega^2 \end{pmatrix} \begin{pmatrix} v_1 \\ v_3 \\ v_5 \\ v_7 \\ v_9 \\ v_{11} \\ v_{13} \\ v_{15} \end{pmatrix}. \quad (12)$$

В силу соотношений

$$\omega^{10} = -\omega^2, \quad \omega^{14} = -\omega^6, \quad \omega^4 = -i, \quad \omega^{12} = i$$

нетрудно заметить, что последние четыре строки в матрице (12) отличаются от соответствующих первых четырех строк только знаком, поэтому вычислять нужно только первые четыре компоненты. Кроме того, в первой и третьей строках стоят лишь 1 и  $\pm i$ , поэтому эти строки не содержат умножений. Оставшиеся две компоненты  $V_2^1$  и  $V_6^1$  получаются из следующего соотношения

$$\begin{pmatrix} V_2^1 \\ V_6^1 \end{pmatrix} = \begin{pmatrix} \omega^2 & \omega^6 & \omega^{10} & \omega^{14} \\ \omega^6 & \omega^2 & \omega^{14} & \omega^{10} \end{pmatrix} \begin{pmatrix} v_1 + v_9 \\ v_3 + v_{11} \\ v_5 + v_{13} \\ v_7 + v_{15} \end{pmatrix}$$

и если вспомнить, что  $\omega^{10} = -\omega^2$ ,  $\omega^{14} = -\omega^6$ , то

$$\begin{pmatrix} V_2^1 \\ V_6^1 \end{pmatrix} = \begin{pmatrix} \omega^2 & \omega^6 \\ \omega^6 & \omega^2 \end{pmatrix} \begin{pmatrix} (v_1 + v_9) - (v_5 + v_{13}) \\ (v_3 + v_{11}) - (v_7 + v_{15}) \end{pmatrix},$$

т. е. получаем 2-точечную циклическую свертку. Ее вычисление, как мы видели выше, требует двух умножений на вещественные множители.

Вычисление же  $V_{2k}^2$  ( $k = 0, 1, \dots, 7$ ) не содержит операций умножения, так как

$$\begin{pmatrix} V_0^2 \\ V_2^2 \\ V_4^2 \\ V_6^2 \\ V_8^2 \\ V_{10}^2 \\ V_{12}^2 \\ V_{14}^2 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i & 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i & 1 & i & -1 & -i \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i & 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i & 1 & i & -1 & -i \end{pmatrix} \begin{pmatrix} v_0 \\ v_2 \\ v_4 \\ v_6 \\ v_8 \\ v_{10} \\ v_{12} \\ v_{14} \end{pmatrix}.$$

Таким образом, 16-точечное преобразование Фурье требует всего 10 вещественных умножений.



§18. АЛГОРИТМ ВИНОГРАДА ДЛЯ БЫСТРОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ  
В СЛУЧАЕ, КОГДА ДЛИНА ПРЕОБРАЗОВАНИЯ ЕСТЬ ПРОСТОЕ ЧИСЛО

Алгоритм, который мы сейчас рассмотрим, предназначен для эффективного вычисления преобразования Фурье малой длины. В его основе лежат две идеи: алгоритм Рейдера, сводящий  $p$ -точечное преобразование Фурье к  $(p-1)$ -точечной циклической свертке при простом  $p$ , и алгоритм Винограда для вычисления циклической свертки.

Итак, первый шаг – замена преобразования Фурье сверткой. Если  $p$  – небольшое простое число, то используется алгоритм Рейдера.  $p$  должно быть не слишком большим, так как алгоритм выписывается вручную. Переход к свертке осуществляется перестановкой индексов, этот шаг не содержит умножений. Далее выполняется свертка, вычисляемая с помощью алгоритма Винограда. Алгоритм Винограда, как мы видели выше, сводит вычисление свертки к серии сложений, за которой следует серия умножений и, наконец, опять некоторое множество сложений.

Рассмотрим применение изложенной идеи к вычислению 5-точечного преобразования Фурье, т. е. 5-точечный БПФ-алгоритм Винограда. Итак, требуется вычислить компоненты

$$V_k = \sum_{j=0}^4 \omega^{jk} v_j \quad (k = 0, 1, \dots, 4),$$

где  $\omega = e^{-\frac{2i\pi}{5}}$ .

В §15 мы уже видели, что в результате применения алгоритма Рейдера в этом случае мы сведем преобразование Фурье к 4-точечной циклической свертке

$$s(x) = g(x) d(x) \pmod{x^4 - 1},$$

где фильтр Рейдера

$$g(x) = (\omega^3 - 1)x^3 + (\omega^4 - 1)x^2 + (\omega^2 - 1)x + (\omega - 1)$$

имеет фиксированные коэффициенты. Вход и выход фильтра описываются многочленами

$$d(x) = v_2 x^3 + v_4 x^2 + v_3 x + v_1$$

и

$$s(x) = (V_3 - V_0)x^3 + (V_4 - V_0)x^2 + (V_2 - V_0)x + (V_1 - V_0).$$

5-точечный БПФ-алгоритм Винограда получается, если произведение  $d(x)g(x)$  вычислять с помощью алгоритма Винограда для малых свертки. Воспользуемся рассмотренным ранее и содержащим 5 умножений алгоритмом 4-точечной циклической свертки, построенным в §6. Его можно приспособить для вычисления преобразования Фурье, введя в матрицу свертки операцию перестановки соответствующих строк и столбцов. Так как коэффициенты многочлена  $g(x)$  фиксированы, то вычисления, связанные с компонентами вектора  $\mathbf{g}$ , можно выполнить заранее один раз, поэтому в общей сложности алгоритма вычисление констант, связанных с коэффициентами  $g(x)$ , можно не учитывать. Если в алгоритме 4-точечной циклической свертки произвести все эти изменения и внести в него члены  $v_0$  и  $V_0$ , то получится 5-точечный БПФ-алгоритм Винограда.

По формулам, полученным в §6, с учетом перестановки алгоритма Рейдера имеем

$$\begin{pmatrix} V_1 - V_0 \\ V_2 - V_0 \\ V_4 - V_0 \\ V_3 - V_0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 0 & -1 \\ 1 & -1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 & 1 \\ 1 & -1 & -1 & -1 & 0 \end{pmatrix} \begin{pmatrix} G_1 & & & & \\ & G_2 & & & \\ & & G_3 & & \\ & & & G_4 & \\ & & & & G_5 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_3 \\ v_4 \\ v_2 \end{pmatrix}, \quad (1)$$

где

$$\begin{pmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \end{pmatrix} = \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 2 & 0 & -2 & 0 \\ -2 & 2 & 2 & -2 \\ 2 & 2 & -2 & -2 \end{pmatrix} \begin{pmatrix} \omega - 1 \\ \omega^2 - 1 \\ \omega^4 - 1 \\ \omega^3 - 1 \end{pmatrix}. \quad (2)$$



Переставим местами две последние компоненты выходного вектора, что влечет за собой перестановку последних двух строк в матрице постсложений. Во входном векторе переставим местами компоненты так, чтобы их нумерация шла в естественном порядке, это влечет за собой соответствующую перестановку столбцов в матрице предсложений. В результате получим

$$\begin{pmatrix} V_1 - V_0 \\ V_2 - V_0 \\ V_3 - V_0 \\ V_4 - V_0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 0 & -1 \\ 1 & -1 & 1 & 1 & 0 \\ 1 & -1 & -1 & -1 & 0 \\ 1 & 1 & -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} G_1 & & & & \\ & G_2 & & & \\ & & G_3 & & \\ & & & G_4 & \\ & & & & G_5 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 0 & 0 & -1 \\ 0 & -1 & 1 & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix}.$$

Теперь вводим в алгоритм члены  $V_0$  и  $v_0$  и заменяем все разности  $V_i - V_0$  ( $i = 1, 2, 3, 4$ ) на  $V_i$ :

$$\begin{pmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \\ V_4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & -1 \\ 1 & 1 & -1 & 1 & 1 & 0 \\ 1 & 1 & -1 & -1 & -1 & 0 \\ 1 & 1 & 1 & -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} G_0 & & & & \\ & G_1 & & & \\ & & G_2 & & \\ & & & G_3 & \\ & & & & G_4 \\ & & & & & G_5 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & -1 & -1 & 1 \\ 0 & 1 & -1 & 1 & -1 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & -1 & 1 & 0 \end{pmatrix} \begin{pmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix}, \quad (3)$$

где  $G_0 = 1$ ,  $V_0 = v_0 + v_1 + v_2 + v_3 + v_4$ .

Это и есть 5-точечный БПФ-алгоритм Винограда.

Теперь обратим внимание на константы, связанные с многочленом  $g(x)$  и вычисляемые по формуле (2):

$$4G_1 = (\omega - 1) + (\omega^2 - 1) + (\omega^4 - 1) + (\omega^3 - 1) = \omega + \omega^2 + \omega^3 + \omega^4 - 4 =$$

$$(1 + \omega + \omega^2 + \omega^3 + \omega^4) - 5 = \frac{1 - \omega^5}{1 - \omega} - 5 = -5 \quad (\text{так как } \omega^5 = 1),$$

откуда  $G_1 = -\frac{5}{4}$ ;

$$4G_2 = (\omega - 1) - (\omega^2 - 1) + (\omega^4 - 1) - (\omega^3 - 1) = \omega - \omega^2 + \omega^4 - \omega^3 =$$

$$(\omega + \omega^4) - (\omega^2 + \omega^3) = (\omega + \omega^{-1}) - (\omega^2 + \omega^{-2}) = 2 \cos \frac{2\pi}{5} - 2 \cos \frac{4\pi}{5} = 4 \sin \frac{3\pi}{5} \sin \frac{\pi}{5},$$

откуда  $G_2 = \sin \frac{3\pi}{5} \sin \frac{\pi}{5}$ ;

$$4G_3 = 2(\omega - 1) - 2(\omega^4 - 1) = 2(\omega - \omega^4) = 2(\omega - \omega^{-1}) = 4i \sin \left( \frac{-2\pi}{5} \right) = -4i \sin \frac{2\pi}{5},$$

откуда  $G_3 = -i \sin \frac{2\pi}{5}$ ;

$$4G_4 = -2(\omega - 1) + 2(\omega^2 - 1) + 2(\omega^4 - 1) - 2(\omega^3 - 1) = 2(-\omega + \omega^2 + \omega^4 - \omega^3) =$$

$$2(-\omega + \omega^{-1}) + 2(\omega^2 - \omega^{-2}) = -4i \sin \left( -\frac{2\pi}{5} \right) + 4i \sin \left( -\frac{4\pi}{5} \right) = 4i \left( \sin \frac{2\pi}{5} - \sin \frac{4\pi}{5} \right) =$$

$$4i \cdot 2 \sin \left( -\frac{\pi}{5} \right) \cos \frac{3\pi}{5} = -8i \sin \frac{\pi}{5} \cos \frac{3\pi}{5},$$

откуда  $G_4 = -2i \sin \frac{\pi}{5} \cos \frac{3\pi}{5}$ ;

$$4G_5 = 2(\omega - 1) + 2(\omega^2 - 1) - 2(\omega^4 - 1) - 2(\omega^3 - 1) = 2\{(\omega - \omega^4) + (\omega^2 - \omega^3)\} =$$

$$2\{(\omega - \omega^{-1}) + (\omega^2 - \omega^{-2})\} = 2\{2i \sin\left(-\frac{2\pi}{5}\right) + 2i \sin\left(-\frac{4\pi}{5}\right)\} = -4i \left(\sin\frac{2\pi}{5} + \sin\frac{4\pi}{5}\right) =$$

$$-4i \cdot 2 \sin\frac{3\pi}{5} \cos\frac{\pi}{5} = -8i \sin\frac{3\pi}{5} \cos\frac{\pi}{5},$$

откуда  $G_5 = -2i \sin\frac{3\pi}{5} \cos\frac{\pi}{5}$ .

Таким образом,

$$\begin{pmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \end{pmatrix} = \begin{pmatrix} -\frac{5}{4} \\ \sin\frac{3\pi}{5} \sin\frac{\pi}{5} \\ -i \sin\frac{2\pi}{5} \\ -2i \sin\frac{\pi}{5} \cos\frac{3\pi}{5} \\ -2i \sin\frac{3\pi}{5} \cos\frac{\pi}{5} \end{pmatrix}.$$

Хотя в средней матрице алгоритма (3) стоят 6 констант  $G_i$ , но поскольку  $G_0 = 1$ , то нетривиальных умножений в алгоритме осталось по-прежнему 5, причем остальные константы  $G_i$  ( $i = 1, 2, \dots, 5$ ) являются чисто вещественными или чисто мнимыми. Это существенно, так как означает, что в случае вещественного входного вектора каждому умножению отвечает одно вещественное умножение, а в случае комплексного входного вектора каждому умножению отвечают два вещественных умножения. Это не случайно, а является общей ситуацией. Поскольку  $p$  нечетное число, то

$$x^{p-1} - 1 = (x^{\frac{p-1}{2}} - 1)(x^{\frac{p-1}{2}} + 1). \quad (4)$$

Когда алгоритм Винограда разбивает задачу вычисления произведения двух многочленов по модулю многочлена  $x^{p-1} - 1$  на подзадачи, соответствующие делителям многочлена  $x^{p-1} - 1$ , то эти делители делят один из множителей, стоящих в правой части (4). В этом случае общая ситуация может быть описана следующей теоремой.

**Теорема.** Пусть

$$g(x) = \sum_{k=0}^{p-2} (\omega^{\pi^k} - 1) x^k$$

— многочлен Рейдера. Для каждого нечетного  $p$  коэффициенты многочлена  $g(x) \pmod{x^{\frac{p-1}{2}} - 1}$  являются вещественными числами, а коэффициенты многочлена  $g(x) \pmod{x^{\frac{p-1}{2}} + 1}$  являются чисто мнимыми.

**Доказательство.** Так как  $\pi$  — примитивный элемент поля  $\mathbb{Z}_p$ , т. е.  $\pi^{p-1} = 1$ , то  $\pi^{\frac{p-1}{2}} = -1$ . Тогда

$$g(x) \pmod{x^{\frac{p-1}{2}} - 1} = \sum_{k=0}^{p-2} (\omega^{\pi^k} - 1) x^k \pmod{x^{\frac{p-1}{2}} - 1} \equiv \sum_{k=0}^{\frac{p-1}{2}-1} \{(\omega^{\pi^k} - 1) + (\omega^{\pi^k + \frac{p-1}{2}} - 1)\} x^k$$

$$= \sum_{k=0}^{\frac{p-1}{2}-1} (\omega^{\pi^k} + \omega^{-\pi^k} - 2) x^k = \sum_{k=0}^{\frac{p-1}{2}-1} \{2 \cos(\theta \pi^k) - 2\} x^k \quad \left(\theta = -\frac{2\pi}{p}\right),$$

а

$$g(x) \pmod{x^{\frac{p-1}{2}} + 1} = \sum_{k=0}^{p-2} (\omega^{\pi^k} - 1) x^k \pmod{x^{\frac{p-1}{2}} + 1} \equiv \sum_{k=0}^{\frac{p-1}{2}-1} \{(\omega^{\pi^k} - 1) - (\omega^{\pi^k + \frac{p-1}{2}} - 1)\} x^k$$

$$= \sum_{k=0}^{\frac{p-1}{2}-1} (\omega^{\pi^k} - \omega^{-\pi^k}) x^k = \sum_{k=0}^{\frac{p-1}{2}-1} 2i \sin(\theta \pi^k) x^k.$$



§19. АЛГОРИТМ ВИНОГРАДА ДЛЯ БЫСТРОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ В СЛУЧАЕ, КОГДА ДЛИНА ПРЕОБРАЗОВАНИЯ ЕСТЬ СТЕПЕНЬ ПРОСТОГО ЧИСЛА

Идея здесь та же, что и в случае, когда длина преобразования Фурье равна простому числу: применить сначала алгоритм Рейдера, сведя часть вычислений к соответствующей циклической свертке, а для вычисления свертки воспользоваться алгоритмом Винограда. При этом для выделения циклической свертки из множества всех индексов удаляются целые числа, кратные  $p$ , т. е. выделяем циклическую группу  $\mathbb{Z}_{p^m}^*$  кольца  $\mathbb{Z}_{p^m}$ . Перестановка индексов из  $\mathbb{Z}_{p^m}^*$  приводит к  $p^{m-1}(p-1)$ -точечной циклической свертке, которая и вычисляется быстрым алгоритмом Винограда. Затем полученные результаты следует подправить членами, учитывающими выброшенные  $p^{m-1}$  строк и столбцов. Эти поправочные члены можно вычислить алгоритмами еще меньших свертки, поскольку они представимы в виде алгоритмов преобразования Фурье малой длины.

Вернемся к примеру, рассмотренному в § 16. Мы видели, что 9-точечное преобразование Фурье можно свести к вычислению одной 6-точечной циклической свертки и двум 2-точечным. Если в исходном матричном уравнении

$$\begin{pmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \\ V_6 \\ V_7 \\ V_8 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 & \omega^8 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega & \omega^3 & \omega^5 & \omega^7 \\ 1 & \omega^3 & \omega^6 & 1 & \omega^3 & \omega^6 & 1 & \omega^3 & \omega^6 \\ 1 & \omega^4 & \omega^8 & \omega^3 & \omega^7 & \omega^2 & \omega^6 & \omega & \omega^5 \\ 1 & \omega^5 & \omega & \omega^6 & \omega^2 & \omega^7 & \omega^3 & \omega^8 & \omega^4 \\ 1 & \omega^6 & \omega^3 & 1 & \omega^6 & \omega^3 & 1 & \omega^6 & \omega^3 \\ 1 & \omega^7 & \omega^5 & \omega^3 & \omega & \omega^8 & \omega^6 & \omega^4 & \omega^2 \\ 1 & \omega^8 & \omega^7 & \omega^6 & \omega^5 & \omega^4 & \omega^3 & \omega^2 & \omega \end{pmatrix} \begin{pmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \end{pmatrix}$$

переставить строки и столбцы с номерами 0, 3, 6 так, чтобы они оказались в матрице первыми, а остальные строки были расположены в порядке степеней двойки по модулю 9, т. е.

$$2^0 = 1, \quad 2^1 = 2, \quad 2^2 = 4, \quad 2^3 = 8, \quad 2^4 = 7, \quad 2^5 = 5,$$

а столбцы – в порядке степеней  $2^{-1}$  по модулю 9, т. е.

$$2^{-0} = 1, \quad 2^{-1} = 5, \quad 2^{-2} = 7, \quad 2^{-3} = 8, \quad 2^{-4} = 4, \quad 2^{-5} = 2,$$

то вычисление преобразуется к виду

$$\begin{pmatrix} V_0 \\ V_3 \\ V_6 \\ V_1 \\ V_2 \\ V_4 \\ V_8 \\ V_7 \\ V_5 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & \omega^3 & \omega^6 & \omega^3 & \omega^6 & \omega^3 \\ 1 & 1 & 1 & \omega^6 & \omega^3 & \omega^6 & \omega^3 & \omega^6 \\ 1 & \omega^3 & \omega^6 & \omega & \omega^5 & \omega^7 & \omega^8 & \omega^4 \\ 1 & \omega^6 & \omega^3 & \omega^2 & \omega & \omega^5 & \omega^7 & \omega^8 \\ 1 & \omega^3 & \omega^6 & \omega^4 & \omega^2 & \omega & \omega^5 & \omega^8 \\ 1 & \omega^6 & \omega^3 & \omega^8 & \omega^4 & \omega^2 & \omega & \omega^5 \\ 1 & \omega^3 & \omega^6 & \omega^7 & \omega^8 & \omega^4 & \omega^2 & \omega \\ 1 & \omega^6 & \omega^3 & \omega^5 & \omega^7 & \omega^8 & \omega^4 & \omega^2 \end{pmatrix} \begin{pmatrix} v_0 \\ v_3 \\ v_6 \\ v_1 \\ v_2 \\ v_4 \\ v_8 \\ v_7 \\ v_5 \end{pmatrix}. \quad (1)$$

Циклические свертки, расположенные в строках, соответствующих компонентам  $V_3$  и  $V_6$ , содержат повторяющиеся вычисления, поэтому их можно переписать в виде

$$\begin{pmatrix} V_3 \\ V_6 \end{pmatrix} = \begin{pmatrix} \omega^3 & \omega^6 \\ \omega^6 & \omega^3 \end{pmatrix} \begin{pmatrix} v_1 + v_7 + v_4 \\ v_5 + v_8 + v_2 \end{pmatrix} + \begin{pmatrix} v_0 + v_3 + v_6 \\ v_0 + v_3 + v_6 \end{pmatrix}. \quad (2)$$

Тогда оставшаяся часть вычислений принимает вид

$$\begin{pmatrix} V_1 \\ V_2 \\ V_4 \\ V_8 \\ V_7 \\ V_5 \end{pmatrix} = \begin{pmatrix} \omega & \omega^5 & \omega^7 & \omega^8 & \omega^4 & \omega^2 \\ \omega^2 & \omega & \omega^5 & \omega^7 & \omega^8 & \omega^4 \\ \omega^4 & \omega^2 & \omega & \omega^5 & \omega^7 & \omega^8 \\ \omega^8 & \omega^4 & \omega^2 & \omega & \omega^5 & \omega^7 \\ \omega^7 & \omega^8 & \omega^4 & \omega^2 & \omega & \omega^5 \\ \omega^5 & \omega^7 & \omega^8 & \omega^4 & \omega^2 & \omega \end{pmatrix} \begin{pmatrix} v_1 \\ v_5 \\ v_7 \\ v_8 \\ v_4 \\ v_2 \end{pmatrix} + \begin{pmatrix} w_1 \\ w_2 \\ w_1 \\ w_2 \\ w_1 \\ w_2 \end{pmatrix}, \quad (3)$$



где

$$\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} \omega^3 & \omega^6 \\ \omega^6 & \omega^3 \end{pmatrix} \begin{pmatrix} v_3 \\ v_6 \end{pmatrix}. \quad (4)$$

Свертки в (2) и (4) отличаются только компонентами входного вектора. Рассмотрим, например, свертку, задаваемую формулой (4). Пусть

$$d(x) = v_3 + v_6 x, \quad g(x) = \omega^3 + \omega^6 x, \quad s(x) = d(x) g(x) \pmod{x^2 - 1}.$$

Так как

$$x^2 - 1 = (x - 1)(x + 1) = m_0(x) m_1(x),$$

то

$$\begin{aligned} d_0(x) &= v_3 + v_6 = D_0, & g_0(x) &= \omega^3 + \omega^6, & s_0(x) &= d_0(x) g_0(x) = (v_3 + v_6)(\omega^3 + \omega^6), \\ d_1(x) &= v_3 - v_6 = D_1, & g_1(x) &= \omega^3 - \omega^6, & s_1(x) &= d_1(x) g_1(x) = (v_3 - v_6)(\omega^3 - \omega^6). \end{aligned}$$

В силу

$$\frac{1}{2}(x + 1) - \frac{1}{2}(x - 1) = 1,$$

получаем

$$s(x) = \frac{1}{2} s_0(x)(x + 1) + \left(-\frac{1}{2}\right) s_1(x)(x - 1) = S_0(x + 1) + S_1(1 - x),$$

где  $S_i = D_i G_i$  ( $i = 0, 1$ ),

$$\begin{pmatrix} G_0 \\ G_1 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} \omega^3 \\ \omega^6 \end{pmatrix}.$$

Таким образом, алгоритм содержит два умножения чисел  $D_i$  на константы  $G_i$ . Поскольку

$$G_0 = \frac{1}{2}(\omega^3 + \omega^6) = \frac{1}{2}(\omega^3 + \omega^{-3}) = \cos 3\theta = \cos \frac{6\pi}{9} = \cos \frac{2\pi}{3} = -\frac{1}{2},$$

$$G_1 = \frac{1}{2}(\omega^3 - \omega^6) = \frac{1}{2}(\omega^3 - \omega^{-3}) = i \sin 3\theta = -i \sin \frac{2\pi}{3} = -\frac{i\sqrt{3}}{2}$$

в силу  $\omega^9 = 1$  и  $\theta = -\frac{2\pi}{9}$ , то оба умножения являются умножениями на чисто вещественное и чисто мнимое число, т. е. в случае вещественных компонент входного вектора мы получим два вещественных умножения.

Перейдем теперь к 6-точечной циклической свертке (3). В § 6 мы рассмотрели быстрый алгоритм Винограда, вычисляющий эту свертку и содержащий 8 умножений и 42 сложения.

Если

$$\begin{aligned} g(x) &= \omega + \omega^2 x + \omega^4 x^2 + \omega^8 x^3 + \omega^7 x^4 + \omega^5 x^5, \\ d(x) &= v_1 + v_5 x + v_7 x^2 + v_8 x^3 + v_4 x^4 + v_2 x^5, \\ s(x) &= d(x) g(x) \pmod{x^6 - 1} = \tilde{V}_1 + \tilde{V}_2 x + \tilde{V}_4 x^2 + \tilde{V}_8 x^3 + \tilde{V}_7 x^4 + \tilde{V}_5 x^5, \end{aligned}$$

то

$$\begin{pmatrix} \tilde{V}_1 \\ \tilde{V}_2 \\ \tilde{V}_4 \\ \tilde{V}_8 \\ \tilde{V}_7 \\ \tilde{V}_5 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & -2 & 1 & 1 & 1 & -2 \\ 1 & 1 & -2 & 1 & -1 & -1 & 2 & -1 \\ 1 & -2 & 1 & 1 & 1 & -2 & 1 & 1 \\ 1 & 1 & 1 & -2 & -1 & -1 & -1 & 2 \\ 1 & 1 & -2 & 1 & 1 & 1 & -2 & 1 \\ 1 & -2 & 1 & 1 & -1 & 2 & -1 & -1 \end{pmatrix} \begin{pmatrix} G_0 \\ G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \\ G_6 \\ G_7 \end{pmatrix}.$$



$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & -1 & 1 & 0 & -1 \\ 1 & -1 & 0 & 1 & -1 & 0 \\ 0 & 1 & -1 & 0 & 1 & -1 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 0 & -1 & -1 & 0 & 1 \\ 1 & 1 & 0 & -1 & -1 & 0 \\ 0 & 1 & 1 & 0 & -1 & -1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_5 \\ v_7 \\ v_8 \\ v_4 \\ v_2 \end{pmatrix},$$

где

$$\begin{pmatrix} G_0 \\ G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \\ G_6 \\ G_7 \end{pmatrix} = \frac{1}{6} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & -1 & 1 & 0 & -1 \\ 1 & -1 & 0 & 1 & -1 & 0 \\ 0 & 1 & -1 & 0 & 1 & -1 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 0 & -1 & -1 & 0 & 1 \\ 1 & 1 & 0 & -1 & -1 & 0 \\ 0 & 1 & 1 & 0 & -1 & -1 \end{pmatrix} \begin{pmatrix} \omega \\ \omega^2 \\ \omega^4 \\ \omega^8 \\ \omega^7 \\ \omega^5 \end{pmatrix}.$$

Вычислим константы  $G_i$  ( $i = 0, 1, \dots, 7$ ):

$$\begin{aligned} G_0 &= \frac{1}{6} (\omega + \omega^2 + \omega^4 + \omega^8 + \omega^7 + \omega^5) = \frac{1}{6} \{(\omega + \omega^8) + (\omega^2 + \omega^7) + (\omega^4 + \omega^5)\} \\ &= \frac{1}{6} \{(\omega + \omega^{-1}) + (\omega^2 + \omega^{-2}) + (\omega^4 + \omega^{-4})\} = \frac{1}{6} (2 \cos \theta + 2 \cos 2\theta + 2 \cos 4\theta) \\ &= \frac{1}{3} \{\cos \theta + (\cos 2\theta + \cos 4\theta)\} = \frac{1}{3} \{\cos \theta + 2 \cos \theta \cos 3\theta\} = \frac{1}{3} \cos \theta (1 + 2 \cos 3\theta) \\ &= \frac{1}{3} \cos \theta (1 + 2 \cos \frac{2\pi}{3}) = \frac{1}{3} \cos \theta \cdot 0 = 0, \\ G_1 &= \frac{1}{6} (\omega - \omega^4 + \omega^8 - \omega^5) = \frac{1}{6} \{(\omega + \omega^8) - (\omega^4 + \omega^5)\} = \frac{1}{6} \{(\omega + \omega^{-1}) - (\omega^4 + \omega^{-4})\} \\ &= \frac{1}{6} (2 \cos \theta - 2 \cos 4\theta) = \frac{1}{3} (\cos \theta - \cos 4\theta) = \frac{2}{3} \sin \frac{5\theta}{2} \sin \frac{3\theta}{2}, \\ G_2 &= \frac{1}{6} (\omega - \omega^2 + \omega^8 - \omega^7) = \frac{1}{6} \{(\omega + \omega^8) - (\omega^2 + \omega^7)\} = \frac{1}{6} \{(\omega + \omega^{-1}) - (\omega^2 + \omega^{-2})\} \\ &= \frac{1}{3} (\cos \theta - \cos 2\theta) = \frac{2}{3} \sin \frac{3\theta}{2} \sin \frac{\theta}{2}, \\ G_3 &= \frac{1}{6} (\omega^2 - \omega^4 + \omega^7 - \omega^5) = \frac{1}{6} \{(\omega^2 + \omega^7) - (\omega^4 + \omega^5)\} = \frac{1}{6} \{(\omega^2 + \omega^{-2}) - (\omega^4 + \omega^{-4})\} \\ &= \frac{1}{3} (\cos 2\theta - \cos 4\theta) = \frac{2}{3} \sin 3\theta \sin \theta, \\ G_4 &= \frac{1}{6} (\omega - \omega^2 + \omega^4 - \omega^8 + \omega^7 - \omega^5) = \frac{1}{6} \{(\omega - \omega^8) - (\omega^2 - \omega^7) + (\omega^4 - \omega^5)\} \\ &= \frac{1}{6} \{(\omega - \omega^{-1}) - (\omega^2 - \omega^{-2}) + (\omega^4 - \omega^{-4})\} = \frac{1}{6} (2i \sin \theta - 2i \sin 2\theta + 2i \sin 4\theta) \\ &= \frac{i}{3} \{\sin \theta - (\sin 2\theta - \sin 4\theta)\} = \frac{i}{3} \{\sin \theta + 2 \sin \theta \cos 3\theta\} = \frac{i}{3} \sin \theta (1 + 2 \cos 3\theta) \\ &= \frac{i}{3} \sin \theta (1 + 2 \cos \frac{2\pi}{3}) = \frac{i}{3} \sin \theta \cdot 0 = 0, \\ G_5 &= \frac{1}{6} (\omega - \omega^4 - \omega^8 + \omega^5) = \frac{1}{6} \{(\omega - \omega^8) - (\omega^4 - \omega^5)\} = \frac{1}{6} \{(\omega - \omega^{-1}) - (\omega^4 - \omega^{-4})\} \\ &= \frac{i}{3} (\sin \theta - \sin 4\theta) = -\frac{2i}{3} \sin \frac{3\theta}{2} \cos \frac{5\theta}{2}, \\ G_6 &= \frac{1}{6} (\omega + \omega^2 - \omega^8 - \omega^7) = \frac{1}{6} \{(\omega - \omega^8) + (\omega^2 - \omega^7)\} = \frac{1}{6} \{(\omega - \omega^{-1}) + (\omega^2 - \omega^{-2})\} \\ &= \frac{i}{3} (\sin \theta + \sin 2\theta) = \frac{2i}{3} \sin \frac{3\theta}{2} \cos \frac{\theta}{2}, \end{aligned}$$

$$G_7 = \frac{1}{6} (\omega^2 + \omega^4 - \omega^7 - \omega^5) = \frac{1}{6} \{(\omega^2 - \omega^7) + (\omega^4 - \omega^5)\} = \frac{1}{6} \{(\omega^2 - \omega^{-2}) + (\omega^4 - \omega^{-4})\} \\ = \frac{i}{3} (\sin 2\theta + \sin 4\theta) = \frac{2i}{3} \sin 3\theta \cos \theta.$$

Таким образом, мы видим, что все константы являются либо вещественными, либо чисто мнимыми, причем две из них обращаются в нуль, т. е. умножения, связанные с модулями  $x - 1$  и  $x + 1$  не нужны, поэтому для вычисления 6-точечной циклической свертки нам потребуется всего 6 умножений, причем если компоненты входного вектора вещественные числа, то требуется 6 вещественных умножений, если же входной вектор имеет комплексные компоненты, то потребуется 12 вещественных умножений. В результате получаем, что для вычисления 9-точечного преобразования Фурье требуется всего 10 умножений на вещественные константы. Эта ситуация является общей и описывается следующей теоремой.

**Теорема.** Пусть  $m > 1$  целое,  $p$  – простое нечетное число. Пусть  $b = p^{m-1}(p - 1)$  и  $g(x)$  – многочлен Рейдера

$$g(x) = \sum_{k=0}^{b-1} \omega^{\pi^k} x^k,$$

где  $\omega$  – корень из единицы степени  $p^m$  и  $\pi$  – целое число порядка  $b$  относительно умножения по модулю  $p^m$ . Тогда:

- (1) коэффициенты многочлена  $g(x) \pmod{x^{\frac{b}{2}} - 1}$  являются вещественными числами;
- (2) коэффициенты многочлена  $g(x) \pmod{x^{\frac{b}{2}} + 1}$  являются чисто мнимыми числами;
- (3) коэффициенты многочлена  $g(x) \pmod{x^{\frac{b}{p}} - 1}$  равны нулю.

**Доказательство.** Порядок числа  $\pi$  равен  $b$ , и так как  $b$  – четно, то  $\pi^b = 1$ ,  $\pi^{\frac{b}{2}} = -1$ . Тогда

$$g(x) \pmod{x^{\frac{b}{2}} - 1} = \sum_{k=0}^{\frac{b}{2}-1} g_k x^k,$$

где  $g_k = \omega^{\pi^k} + \omega^{\pi^{k+\frac{b}{2}}} = \omega^{\pi^k} + \omega^{-\pi^k} = \{\cos(\theta\pi^k) + i\sin(\theta\pi^k)\} + \{\cos(\theta\pi^k) - i\sin(\theta\pi^k)\} = 2\cos(\theta\pi^k)$ ,  
а

$$g(x) \pmod{x^{\frac{b}{2}} + 1} = \sum_{k=0}^{\frac{b}{2}-1} \tilde{g}_k x^k,$$

где  $\tilde{g}_k = \omega^{\pi^k} - \omega^{\pi^{k+\frac{b}{2}}} = \omega^{\pi^k} - \omega^{-\pi^k} = 2i\sin(\theta\pi^k)$ .

Таким образом, утверждения (1) и (2) справедливы. Для доказательства (3) рассмотрим многочлен

$$\hat{g}(x) = g(x) \pmod{x^{\frac{b}{p}} - 1} = \sum_{k=0}^{b-1} \omega^{\pi^k} x^k \pmod{x^{\frac{b}{p}} - 1} = \sum_{k=0}^{\frac{b}{p}-1} \hat{g}_k x^k.$$

Коэффициент  $\hat{g}_0$  равен сумме тех коэффициентов многочлена  $g(x)$ , индекс  $k$  которых кратен числу  $\frac{b}{p}$ . Таких коэффициентов имеется  $p$  штук, т. е.

$$\hat{g}_0 = \sum_{j=0}^{p-1} \omega^{\pi^{\frac{jb}{p}}}.$$

В общем случае коэффициент  $\hat{g}_r$  дается равенством

$$\hat{g}_r = \sum_{j=0}^{p-1} \omega^{\pi^{r+\frac{jb}{p}}}, \quad r = 0, 1, \dots, \frac{b}{p} - 1.$$



Требуется доказать, что  $\hat{g}_r$  равен нулю для всех  $r = 0, 1, \dots, \frac{b}{p} - 1$ . Перепишем  $\hat{g}_r$  в виде

$$\hat{g}_r = \sum_{j=0}^{p-1} (\omega^{\pi^r})^{\pi^{\frac{jb}{p}}}.$$

Так как  $\omega^{\pi^r}$  опять является корнем степени  $p^m$  из единицы, то утверждение достаточно доказать только для  $r = 0$ , т. е. для

$$\hat{g}_0 = \sum_{j=0}^{p-1} \omega^{\pi^{\frac{jb}{p}}} = \sum_{j=0}^{p-1} \omega^{(\pi^{\frac{b}{p}})^j}.$$

Заметим, что  $\pi^{\frac{b}{p}}$  есть элемент порядка  $p$ , т. е.  $(\pi^{\frac{b}{p}})^p = \pi^b = 1$ , значит,

$$\hat{g}_0 = \omega^{\pi^0} + \omega^{\pi^{\frac{b}{p}}} + \omega^{\pi^{\frac{2b}{p}}} + \dots + \omega^{\pi^{\frac{(p-1)b}{p}}} = \frac{\omega^{\pi^0} - \omega^{\pi^{\frac{pb}{p}}}}{1 - \omega^{\pi^{\frac{b}{p}}}} = \frac{\omega - \omega}{1 - \omega^{\pi^{\frac{b}{p}}}} = 0,$$

что и требовалось доказать.

Итак, если длина преобразования Фурье равна  $p^m$ , т. е. степени простого числа, то матрица преобразования размера  $p^m \times p^m$  разбивается на  $p^{m-1}(p-1)$ -точечную циклическую свертку и  $p^{m-1}+1$   $(p-1)$ -точечных циклических сверток. Все эти свертки вычисляются с помощью алгоритма Винограда для циклических сверток, содержащего только чисто вещественные и чисто мнимые умножения.

## §20. АЛГОРИТМ ВИНОГРАДА БЫСТРОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ В СЛУЧАЕ, КОГДА ДЛИНА ПРЕОБРАЗОВАНИЯ ЕСТЬ СТЕПЕНЬ ДВОЙКИ

Объединим алгоритмы Рейдера и Винограда для вычисления дискретного преобразования Фурье в случае, когда длина преобразования  $n = 2^m$ . Мы уже видели в § 17, что алгоритм Рейдера перестановки компонент входного и выходного векторов можно применить только к тем компонентам, индексы которых являются нечетными числами. Это подмножество компонент переупорядовывается таким образом, что в результате получаются четыре циклические свертки длины  $2^{m-2}$ . Мы также видели, что эти четыре свертки можно свести к двум  $2^{m-2}$ -точечным циклическим сверткам, если добавить некоторые дополнительные предположения и постсложения.

Строки и столбцы с четными индексами можно представить в виде преобразования Фурье длины  $2^{m-1}$  несколькими способами. Эта часть вычислений представляет собой  $2^{m-1}$ -точечное преобразование Фурье. Разбиение в некотором смысле аналогично одному шагу алгоритма Кули – Тьюки по основанию два. Используем, например, алгоритм с прореживанием по частоте.

Для заданного  $2^m$ -точечного преобразования Фурье

$$V_k = \sum_{j=0}^{2^m-1} \omega^{jk} v_j, \quad k = 0, 1, \dots, 2^m - 1 \quad (1)$$

компоненты с четными значениями индекса  $k$  можно записать в виде

$$V_{2k} = \sum_{j=0}^{2^{m-1}-1} (v_j + v_{j+2^{m-1}}) \omega^{2jk}, \quad k = 0, 1, \dots, 2^{m-1} - 1. \quad (2)$$

Это и есть  $2^{m-1}$ -точечное преобразование Фурье вектора, компоненты которого получаются сложением  $j$ -й и  $(2^{m-1} + j)$ -й компонент исходного вектора  $\mathbf{v}$  ( $j = 0, 1, \dots, 2^{m-1} - 1$ ).

Компоненты с нечетными значениями индекса  $k$  можно записать в виде

$$V_{2k+1} = \sum_{j=0}^{2^{m-1}-1} \omega^{2j(2k+1)} v_{2j} + \sum_{j=0}^{2^{m-1}-1} \omega^{(2j+1)(2k+1)} v_{2j+1}, \quad k = 0, 1, \dots, 2^{m-1} - 1. \quad (3)$$

Рассмотрим первую сумму в формуле (3):

$$\sum_{j=0}^{2^{m-1}-1} \omega^{2j(2k+1)} v_{2j} = \sum_{j=0}^{2^{m-1}-1} \omega^{4kj+2j} v_{2j} = \sum_{j=0}^{2^{m-1}-1} \omega^{4kj} \omega^{2j} v_{2j}. \quad (4)$$

Так как  $\omega$  — корень степени  $2^m$  из единицы, то  $\omega^4$  есть корень степени  $2^{m-2}$  из единицы, поэтому

$$\omega^{4k(j+2^{m-2})} = \omega^{4kj+2^m k} = \omega^{4kj}.$$

Значит, сумма (4) может быть записана в виде

$$\sum_{j=0}^{2^{m-2}-1} \omega^{4kj} (\omega^{2j} v_{2j} + \omega^{2(j+2^{m-2})} v_{2(j+2^{m-2})}) = \sum_{j=0}^{2^{m-2}-1} \omega^{4kj} (\omega^{2j} v_{2j} - \omega^{2j} v_{2(j+2^{m-2})}),$$

так как  $\omega^{2^{m-1}} = -1$ . Эта сумма вычисляется только для  $k = 0, 1, \dots, 2^{m-2} - 1$ .

В результате формула (3) переписывается в виде

$$V_{2k+1} = \sum_{j=0}^{2^{m-2}-1} \omega^{4kj} (\omega^{2j} v_{2j} - \omega^{2j} v_{2j+2^{m-1}}) + \sum_{j=0}^{2^{m-1}-1} \omega^{(2j+1)(2k+1)} v_{2j+1}, \quad (5)$$



$k = 0, 1, \dots, 2^{m-1} - 1$ , причем первая сумма вычисляется для первых  $2^{m-2}$  значений индекса  $k$ , так как потом эти значения повторяются. Если входная последовательность является вещественной, то эта часть вычислений состоит из  $2(2^{m-2} - 3) = 2^{m-1} - 6$  вещественных умножений и  $2^{m-2}$ -точечного преобразования Фурье. Вторая сумма в (5) может быть вычислена с помощью обобщения алгоритма Рейдера для случая  $n = 2^m$ , это вычисление сводится, как мы уже знаем, к вычислению двух циклических сверток длины  $2^{m-2}$ .

Вернемся к примеру, рассмотренному в § 17. Начнем с вычисления компонент 16-точечного преобразования Фурье с четными индексами. Согласно (2) их вычисление сводится к 8-точечному преобразованию Фурье, которое описывается следующим уравнением

$$\begin{pmatrix} V_0 \\ V_2 \\ V_4 \\ V_6 \\ V_8 \\ V_{10} \\ V_{12} \\ V_{14} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} & \omega^{14} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & 1 & \omega^4 & \omega^8 & \omega^{12} \\ 1 & \omega^6 & \omega^{12} & \omega^2 & \omega^8 & \omega^{14} & \omega^4 & \omega^{10} \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & \omega^{10} & \omega^4 & \omega^{14} & \omega^8 & \omega^2 & \omega^{12} & \omega^6 \\ 1 & \omega^{12} & \omega^8 & \omega^4 & 1 & \omega^{12} & \omega^8 & \omega^4 \\ 1 & \omega^{14} & \omega^{12} & \omega^{10} & \omega^8 & \omega^6 & \omega^4 & \omega^2 \end{pmatrix} \begin{pmatrix} v_0 + v_8 \\ v_1 + v_9 \\ v_2 + v_{10} \\ v_3 + v_{11} \\ v_4 + v_{12} \\ v_5 + v_{13} \\ v_6 + v_{14} \\ v_7 + v_{15} \end{pmatrix} =$$

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega^2 & -i & \omega^6 & -1 & -\omega^2 & i & -\omega^6 \\ 1 & -i & -1 & i & 1 & -i & -1 & i \\ 1 & \omega^6 & i & \omega^2 & -1 & -\omega^6 & -i & -\omega^2 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -\omega^2 & -i & -\omega^6 & -1 & \omega^2 & i & \omega^6 \\ 1 & i & -1 & -i & 1 & i & -1 & -i \\ 1 & -\omega^6 & i & -\omega^2 & -1 & \omega^6 & -i & \omega^2 \end{pmatrix} \begin{pmatrix} v_0 + v_8 \\ v_1 + v_9 \\ v_2 + v_{10} \\ v_3 + v_{11} \\ v_4 + v_{12} \\ v_5 + v_{13} \\ v_6 + v_{14} \\ v_7 + v_{15} \end{pmatrix} \quad (6)$$

в силу  $\omega^{16} = 1$ ,  $\omega^8 = -1$ ,  $\omega^4 = -i$ ,  $\omega^{12} = i$ .

Переставим компоненты выходного и входного векторов, а также соответствующие строки и столбцы матрицы  $W = (\omega^{2jk})$  так, чтобы строки и столбцы, соответствующие четным значениям  $k$ , стояли первыми:

$$\begin{pmatrix} V_0 \\ V_4 \\ V_8 \\ V_{12} \\ V_2 \\ V_6 \\ V_{10} \\ V_{14} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & -i & i & -i & i \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & i & -i & i & -i \\ 1 & -i & -1 & i & \omega^2 & \omega^6 & -\omega^2 & -\omega^6 \\ 1 & i & -1 & -i & \omega^6 & \omega^2 & -\omega^6 & -\omega^2 \\ 1 & -i & -1 & i & -\omega^2 & -\omega^6 & \omega^2 & \omega^6 \\ 1 & i & -1 & -i & -\omega^6 & -\omega^2 & \omega^6 & \omega^2 \end{pmatrix} \begin{pmatrix} v_0 + v_8 \\ v_2 + v_{10} \\ v_4 + v_{12} \\ v_6 + v_{14} \\ v_1 + v_9 \\ v_3 + v_{11} \\ v_5 + v_{13} \\ v_7 + v_{15} \end{pmatrix}. \quad (7)$$

Первые четыре строки не содержат умножений, и соответствующие компоненты выходного вектора могут быть вычислены с помощью 12 сложений. Первые четыре столбца в последних четырех строках также не содержат умножений, поэтому последние четыре выходные компоненты в (7) разбиваем на две части:

$$V_{2+4l} = V_{2+4l}^1 + V_{2+4l}^2 \quad (l = 0, 1, 2, 3),$$

где  $V_{2+4l}^1$  содержит вклад первых четырех столбцов, а  $V_{2+4l}^2$  — вклад последних четырех. Причем

$$\begin{aligned} V_2^1 &= V_{10}^1, & V_6^1 &= V_{14}^1, \\ V_2^2 &= -V_{10}^2, & V_6^2 &= -V_{14}^2, \end{aligned}$$

поэтому вычислять следует только две компоненты. Для вычисления  $V_2^1$  и  $V_6^1$  требуется четыре сложения. Вычисление  $V_2^2$  и  $V_6^2$  можно выполнять в виде 2-точечной циклической свертки

$$\begin{pmatrix} V_2^2 \\ V_6^2 \end{pmatrix} = \begin{pmatrix} \omega^2 & \omega^6 \\ \omega^6 & \omega^2 \end{pmatrix} \begin{pmatrix} (v_1 + v_9) - (v_5 + v_{13}) \\ (v_3 + v_{11}) - (v_7 + v_{15}) \end{pmatrix},$$



которая вычисляется по формуле

$$\begin{pmatrix} V_2^2 \\ V_6^2 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} \frac{1}{2}(\omega^2 + \omega^6) & 0 \\ 0 & \frac{1}{2}(\omega^2 - \omega^6) \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} (v_1 + v_9) - (v_5 + v_{13}) \\ (v_3 + v_{11}) - (v_7 + v_{15}) \end{pmatrix} =$$

$$\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} -i \cos 2\theta & 0 \\ 0 & -\sin 2\theta \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} (v_1 + v_9) - (v_5 + v_{13}) \\ (v_3 + v_{11}) - (v_7 + v_{15}) \end{pmatrix}$$

и требует 6 сложений и 2 умножений на вещественные числа. С учетом требуемых сложений компонент входного вектора всего эта часть вычислений требует 34 сложения.

Перейдем теперь к вычислению компонент выходного вектора с нечетными индексами. Начнем с первой суммы в формуле (5). Пусть

$$V_{2k+1} = V_{2k+1}^1 + V_{2k+1}^2,$$

где

$$V_{2k+1}^2 = \sum_{j=0}^{2^{m-2}-1} \omega^{4kj} (\omega^{2j} v_{2j} - \omega^{2j} v_{2j+2^{m-1}}) = \sum_{j=0}^{2^{m-2}-1} \omega^{4kj} \omega^{2j} (v_{2j} - v_{2j+2^{m-1}}).$$

Как мы уже заметили, вычислять эту сумму надо только для  $k = 0, 1, 2, 3$ . Вычислим сначала компоненты  $\omega^{2j} (v_{2j} - v_{2j+8})$  ( $j = 0, 1, 2, 3$ ), для чего потребуется 2 нетривиальных умножения:

$$\begin{aligned} \tilde{v}_0 &= v_0 - v_8, \\ \tilde{v}_2 &= \omega^2 (v_2 - v_{10}), \\ \tilde{v}_4 &= -i (v_4 - v_{12}), \\ \tilde{v}_6 &= \omega^6 (v_6 - v_{14}). \end{aligned} \tag{8}$$

Осталось вычислить 4-точечное преобразование Фурье

$$V_{2k+1}^2 = \sum_{j=0}^3 \omega^{4jk} \tilde{v}_{2j},$$

т. е.

$$\begin{pmatrix} V_1^2 \\ V_3^2 \\ V_5^2 \\ V_7^2 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega^4 & \omega^8 & \omega^{12} \\ 1 & \omega^8 & 1 & \omega^8 \\ 1 & \omega^{12} & \omega^8 & \omega^4 \end{pmatrix} \begin{pmatrix} \tilde{v}_0 \\ \tilde{v}_2 \\ \tilde{v}_4 \\ \tilde{v}_6 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix} \begin{pmatrix} \tilde{v}_0 \\ \tilde{v}_2 \\ \tilde{v}_4 \\ \tilde{v}_6 \end{pmatrix}.$$

Последнее вычисление не содержит умножений, требуя только 8 сложений. С учетом сложений компонент входного вектора в формулах (8) на эту часть вычислений требуется всего 12 сложений.

Осталось вычислить компоненты  $V_{2k+1}^1$  ( $k = 0, 1, \dots, 7$ ), для чего применяется алгоритм Рейдера перестановки соответствующих компонент входного и выходного векторов, сводящий это вычисление к циклическим сверткам. В § 17 мы видели, что за счет добавления некоторых сложений можно свести это вычисление к паре 4-точечных циклических сверток с вещественной и чисто мнимой матрицами, при этом часть вычислений, связанных с модулем  $x^2 - 1$ , выполнять не надо, поэтому вычисление этих двух сверток требует всего 6 умножений на вещественные константы.

Подробно рассмотрим вычисление только одной из этих сверток, поскольку вторая вычисляется аналогично с заменой входных компонент и коэффициентов многочлена  $g(x)$  с  $\cos k\theta$  на  $\sin k\theta$ .

Из формул (2), (6) и (7) §17 следует, что

$$\begin{pmatrix} \tilde{V}_1^1 \\ \tilde{V}_1^2 \end{pmatrix} = \begin{pmatrix} E_4 & E_4 \\ E_4 & -E_4 \end{pmatrix} \begin{pmatrix} \tilde{W}_1 & \mathbf{O} \\ \mathbf{O} & \tilde{W}_2 \end{pmatrix} \begin{pmatrix} E_4 & E_4 \\ E_4 & -E_4 \end{pmatrix} \begin{pmatrix} \tilde{v}_1 \\ \tilde{v}_2 \end{pmatrix}, \tag{9}$$



$$\begin{aligned} \hat{V}_1^1 &= (V_1^1, V_5^1, V_9^1, V_{13}^1)^T, \\ \hat{V}_2^1 &= (V_1^{15}, V_1^{11}, V_7^1, V_3^1)^T, \\ \hat{V}_1^4 &= (v_1, v_{13}, v_9, v_5)^T, \\ \hat{V}_2^4 &= (v_{15}, v_3, v_7, v_{11})^T, \end{aligned}$$

$$E_4 = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix},$$

$$\begin{pmatrix} \hat{W}_1 & \hat{O} \\ \hat{O} & \hat{W}_2 \end{pmatrix} = \begin{pmatrix} \cos \theta & \cos 13\theta & \cos 9\theta & \cos 5\theta \\ \cos 9\theta & \cos 5\theta & \cos \theta & \cos 13\theta \\ \cos 5\theta & \cos 13\theta & \cos \theta & \cos 9\theta \\ \cos 13\theta & \cos 9\theta & \cos 5\theta & \cos \theta \end{pmatrix} \begin{pmatrix} i \sin \theta & i \sin 13\theta & i \sin 9\theta & i \sin 5\theta \\ i \sin 5\theta & i \sin \theta & i \sin 13\theta & i \sin 9\theta \\ i \sin 9\theta & i \sin 13\theta & i \sin 5\theta & i \sin \theta \\ i \sin 13\theta & i \sin 9\theta & i \sin \theta & i \sin 5\theta \end{pmatrix}$$

Введем обозначения

$$\begin{pmatrix} E_4 & E_4 \\ E_4 & -E_4 \end{pmatrix} \begin{pmatrix} \hat{V}_1 \\ \hat{V}_2 \end{pmatrix} = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix} \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix} \begin{pmatrix} v_1 & v_{13} & v_9 & v_5 \\ v_{11} & v_3 & v_7 & v_{15} \\ v_1 & v_{13} & v_9 & v_5 \\ v_{11} & v_3 & v_7 & v_{15} \end{pmatrix} = \begin{pmatrix} v_1 + v_{15} & v_{13} + v_3 & v_9 + v_7 & v_5 + v_{11} \\ v_5 + v_{11} & v_9 + v_7 & v_{13} + v_3 & v_1 + v_{15} \\ v_3 - v_{13} & v_7 - v_7 & v_5 - v_{11} & v_1 - v_{15} \\ v_5 - v_{11} & v_9 - v_7 & v_1 - v_{15} & v_3 - v_{13} \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \\ w_8 \end{pmatrix}$$

$$\begin{pmatrix} \hat{V}_1 \\ \hat{V}_5 \\ \hat{V}_9 \\ \hat{V}_{13} \\ \hat{V}_{15} \\ \hat{V}_{11} \\ \hat{V}_7 \\ \hat{V}_3 \end{pmatrix} = \begin{pmatrix} \cos \theta & \cos 13\theta & \cos 9\theta & \cos 5\theta \\ \cos 9\theta & \cos 5\theta & \cos \theta & \cos 13\theta \\ \cos 5\theta & \cos 13\theta & \cos \theta & \cos 9\theta \\ \cos 13\theta & \cos 9\theta & \cos 5\theta & \cos \theta \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} i \sin \theta & i \sin 13\theta & i \sin 9\theta & i \sin 5\theta \\ i \sin 5\theta & i \sin \theta & i \sin 13\theta & i \sin 9\theta \\ i \sin 9\theta & i \sin 13\theta & i \sin 5\theta & i \sin \theta \\ i \sin 13\theta & i \sin 9\theta & i \sin \theta & i \sin 5\theta \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \\ w_8 \end{pmatrix}$$

откуда

$$\begin{pmatrix} \hat{V}_1 \\ \hat{V}_5 \\ \hat{V}_9 \\ \hat{V}_{13} \\ \hat{V}_{15} \\ \hat{V}_{11} \\ \hat{V}_7 \\ \hat{V}_3 \end{pmatrix} = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix} \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix} \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix} \begin{pmatrix} \hat{V}_1 \\ \hat{V}_5 \\ \hat{V}_9 \\ \hat{V}_{13} \\ \hat{V}_{15} \\ \hat{V}_{11} \\ \hat{V}_7 \\ \hat{V}_3 \end{pmatrix} = \begin{pmatrix} \hat{V}_1 + \hat{V}_{15} & \hat{V}_5 + \hat{V}_{11} & \hat{V}_9 + \hat{V}_7 & \hat{V}_{13} + \hat{V}_3 \\ \hat{V}_5 + \hat{V}_{11} & \hat{V}_9 + \hat{V}_7 & \hat{V}_{13} + \hat{V}_3 & \hat{V}_1 + \hat{V}_{15} \\ \hat{V}_9 + \hat{V}_7 & \hat{V}_{13} + \hat{V}_3 & \hat{V}_1 + \hat{V}_{15} & \hat{V}_5 + \hat{V}_{11} \\ \hat{V}_{13} + \hat{V}_3 & \hat{V}_1 + \hat{V}_{15} & \hat{V}_5 + \hat{V}_{11} & \hat{V}_9 + \hat{V}_7 \\ \hat{V}_1 - \hat{V}_{13} & \hat{V}_5 - \hat{V}_{11} & \hat{V}_9 - \hat{V}_7 & \hat{V}_3 - \hat{V}_5 \end{pmatrix}$$

Из формулы (10) следует, что

$$s_1(x) = \hat{V}_1 + \hat{V}_5 x + \hat{V}_9 x^2 + \hat{V}_{13} x^3$$

есть 4-точечная циклическая свертка многочленов

$$d_1(x) = (v_1 + v_{15}) + (v_3 + v_{13})x + (v_7 + v_9)x^2 + (v_5 + v_{11})x^3$$

и

$$g_1(x) = \cos \theta + \cos 5\theta x + \cos 9\theta x^2 + \cos 13\theta x^3,$$

т. е.

$$s_1(x) = d_1(x) g_1(x) \pmod{x^4 - 1}, \quad (11)$$

а

$$s_2(x) = \hat{V}_{15} + \hat{V}_{11} x + \hat{V}_7 x^2 + \hat{V}_3 x^3$$

есть 4-точечная циклическая свертка многочленов

$$d_2(x) = (v_1 - v_{15}) + (v_{13} - v_3) x + (v_9 - v_7) x^2 + (v_5 - v_{11}) x^3$$

и

$$g_2(x) = i\{\sin \theta + \sin 5\theta x + \sin 9\theta x^2 + \sin 13\theta x^3\},$$

т. е.

$$s_2(x) = d_2(x) g_2(x) \pmod{x^4 - 1}. \quad (12)$$

Решим задачу (11). Мы уже видели в § 16, что если воспользоваться алгоритмом Винограда и разложить  $x^4 - 1$  в произведение

$$(x^2 - 1)(x^2 + 1),$$

то  $g_1(x) \pmod{x^2 - 1} \equiv 0$  (и  $g_2(x) \pmod{x^2 - 1} \equiv 0$ ), т. е. соответствующую подзадачу можно не решать. Поэтому требуется лишь вычислить

$$d_1(x) g_1(x) \pmod{x^2 + 1}.$$

Поскольку

$$d_1(x) \pmod{x^2 + 1} \equiv \{(v_1 + v_{15}) - (v_9 + v_7)\} + \{(v_3 + v_{13}) - (v_5 + v_{11})\} x = d_0 + d_1 x$$

и

$$g_1(x) \pmod{x^2 + 1} \equiv (\cos \theta - \cos 9\theta) + (\cos 5\theta - \cos 13\theta) x = g_0 + g_1 x,$$

то выберем в качестве первоначального модуля  $m(x) = x(x-1)(x+1)$  и найдем линейную свертку многочленов  $d_1(x)$  и  $g_1(x)$ .

Имеем

$$\begin{aligned} d_0^1(x) &= d_1(x) \pmod{x} \equiv d_0, & g_0^1(x) &= g_1(x) \pmod{x} \equiv g_0, \\ d_1^1(x) &= d_1(x) \pmod{x-1} \equiv d_0 + d_1, & g_1^1(x) &= g_1(x) \pmod{x-1} \equiv g_0 + g_1, \\ d_2^1(x) &= d_1(x) \pmod{x+1} \equiv d_0 - d_1, & g_2^1(x) &= g_1(x) \pmod{x+1} \equiv g_0 - g_1, \end{aligned}$$

$$\begin{aligned} s_0^1(x) &= d_0^1(x) g_0^1(x) = d_0 g_0, \\ s_1^1(x) &= d_1^1(x) g_1^1(x) = (d_0 + d_1)(g_0 + g_1), \\ s_2^1(x) &= d_2^1(x) g_2^1(x) = (d_0 - d_1)(g_0 - g_1). \end{aligned}$$

Используя следующую таблицу многочленов  $m_k(x)$ ,  $M_k(x)$  и  $N_k(x)$  ( $k = 0, 1, 2$ )

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$
0	$x$	$x^2 - 1$	$-\frac{1}{2}$
1	$x - 1$	$x^2 + x$	$\frac{1}{2}$
2	$x + 1$	$x^2 - x$	$\frac{1}{2}$

получаем

$$d_1(x) g_1(x) = s_0^1(x)(1 - x^2) + \frac{1}{2} s_1^1(x)(x^2 + x) + \frac{1}{2} s_2^1(x)(x^2 - x).$$



Введем обозначения

$$\begin{aligned} D_0 &= d_0, & G_0 &= 2g_0, & S_k &= D_k G_k \quad (k = 0, 1, 2), \\ D_1 &= d_0 + d_1, & G_1 &= \frac{1}{2}(g_0 + g_1), \\ D_2 &= d_0 - d_1, & G_2 &= \frac{1}{2}(g_0 - g_1). \end{aligned}$$

Тогда

$$\begin{aligned} s_1(x) &= \frac{1}{2} S_0 (1 - x^2) + S_1 (x^2 + x) + S_2 (x^2 - x) \pmod{x^2 + 1} \equiv S_0 + S_1 (x - 1) + S_2 (-1 - x) = \\ &= (S_0 - S_1 - S_2) + (S_1 - S_2) x. \end{aligned}$$

Итак, получаем

$$\begin{pmatrix} s_0^1 \\ s_1^1 \end{pmatrix} = \begin{pmatrix} 1 & -1 & -1 \\ 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} G_0 & & \\ & G_1 & \\ & & G_2 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \end{pmatrix}, \quad (13)$$

где

$$\begin{pmatrix} G_0 \\ G_1 \\ G_2 \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{pmatrix} \begin{pmatrix} g_0 \\ g_1 \end{pmatrix}.$$

Мы получили алгоритм, содержащий 3 умножения и 5 сложений. Вычислим константы  $G_k$  ( $k = 0, 1, 2$ ):

$$G_0 = 2g_0 = 2(\cos \theta - \cos 9\theta) = 4 \sin 5\theta \sin 4\theta = -4 \sin 5\theta,$$

$$\begin{aligned} G_1 &= \frac{1}{2} \{(\cos \theta - \cos 9\theta) + (\cos 5\theta - \cos 13\theta)\} = \sin 5\theta \sin 4\theta + \sin 9\theta \sin 4\theta \\ &= -(\sin 5\theta + \sin 9\theta) = -2 \sin 7\theta \cos 2\theta, \end{aligned}$$

$$G_2 = \frac{1}{2} \{(\cos \theta - \cos 9\theta) - (\cos 5\theta - \cos 13\theta)\} = -(\sin 5\theta - \sin 9\theta) = 2 \sin 2\theta \cos 7\theta,$$

в силу  $\sin 4\theta = \sin \left(-\frac{4 \cdot 2\pi}{16}\right) = \sin \left(-\frac{\pi}{2}\right) = -1$ .

При решении задачи (12) используем тот же алгоритм (13), заменяя компоненты входного вектора  $d_0$  и  $d_1$  соответственно на  $\{(v_1 - v_{15}) - (v_9 - v_7)\}$  и  $\{(v_{13} - v_3) - (v_5 - v_{11})\}$ , а  $g_0$  и  $g_1$  — на  $i(\sin \theta - \sin 9\theta)$  и  $i(\sin 5\theta - \sin 13\theta)$ . В этом случае константы  $G_k$  ( $k = 0, 1, 2$ ) равны:

$$G_0 = 2i(\sin \theta - \sin 9\theta) = -4i \cos 5\theta \sin 4\theta = 4i \cos 5\theta,$$

$$G_1 = \frac{1}{2} i \{(\sin \theta - \sin 9\theta) + (\sin 5\theta - \sin 13\theta)\} = i(\cos 5\theta + \cos 9\theta) = 2i \sin 7\theta \cos 2\theta,$$

$$G_2 = \frac{1}{2} i \{(\sin \theta - \sin 9\theta) - (\sin 5\theta - \sin 13\theta)\} = i(\cos 5\theta - \cos 9\theta) = 2i \sin 7\theta \cos 2\theta.$$

Из формул (9) следует, что вычисление компонент  $V_{2k+1}^1$  ( $k = 0, 1, \dots, 7$ ) потребует всего  $16 + 2 \cdot 5 = 26$  сложений и 6 умножений на вещественные или чисто мнимые числа. Тогда для вычисления компонент с нечетными индексами  $V_{2k+1}$  ( $k = 0, 1, \dots, 7$ ) потребуется всего  $26 + 8 + 12$  сложений и 6 умножений. А всего 16-точечное преобразование Фурье может быть вычислено с помощью 10 умножений и 80 сложений.

Рассмотрим еще один пример — 8-точечное преобразование Фурье. Согласно изложенному выше, вычисление этого преобразования должно сводиться к:

1) вычислению 4-точечного преобразования Фурье (для компонент выходного вектора с четными индексами);

2) вычислению 2-точечного преобразования Фурье, которому предшествуют 3 комплексных умножения (для первой части, входящей в компоненты выходного вектора с нечетными индексами);

3) вычислению двух 2-точечных циклических сверток, которое должно содержать не более двух умножений на вещественные или чисто мнимые числа (для второй части, входящей в компоненты выходного вектора с нечетными индексами).

Прямой алгоритм вычисления 8-точечного преобразования Фурье имеет вид

$$\begin{pmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \\ V_6 \\ V_7 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \\ 1 & \omega^2 & \omega^4 & \omega^6 & 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^3 & \omega^6 & \omega & \omega^4 & \omega^7 & \omega^2 & \omega^5 \\ 1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 \\ 1 & \omega^5 & \omega^2 & \omega^7 & \omega^4 & \omega & \omega^6 & \omega^3 \\ 1 & \omega^6 & \omega^4 & \omega^2 & 1 & \omega^6 & \omega^4 & \omega^2 \\ 1 & \omega^7 & \omega^6 & \omega^5 & \omega^4 & \omega^3 & \omega^2 & \omega \end{pmatrix} \begin{pmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{pmatrix},$$

где  $\omega = e^{-\frac{2i\pi}{8}}$ .

4-точечное преобразование Фурье для вычисления компонент с четными индексами:

$$\begin{pmatrix} V_0 \\ V_2 \\ V_4 \\ V_6 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^4 & 1 & \omega^4 \\ 1 & \omega^6 & \omega^4 & \omega^2 \end{pmatrix} \begin{pmatrix} v_0 + v_4 \\ v_1 + v_5 \\ v_2 + v_6 \\ v_3 + v_7 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix} \begin{pmatrix} v_0 + v_4 \\ v_1 + v_5 \\ v_2 + v_6 \\ v_3 + v_7 \end{pmatrix}$$

не содержит умножений и может быть выполнено с помощью 8 сложений.

Вычисление компонент выходного вектора с нечетными индексами разобьем на две части:

$$V_{2k+1} = V_{2k+1}^1 + V_{2k+1}^2 \quad (k = 0, 1, 2, 3),$$

$$V_{2k+1}^1 = \sum_{j=0}^1 \omega^{4jk} \omega^{2j} (v_{2j} - v_{2j+4}) \quad (k = 0, 1),$$

так как  $V_1^1 = V_5^1$ ,  $V_3^1 = V_7^1$ ,

$$V_{2k+1}^2 = \sum_{j=0}^3 \omega^{(2j+1)(2k+1)} v_{2j+1} \quad (k = 0, 1, 2, 3).$$

Для компонент  $V_{2k+1}^1$  ( $k = 0, 1$ ) имеем

$$\begin{pmatrix} V_1^1 \\ V_3^1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & \omega^2 \end{pmatrix} \begin{pmatrix} v_0 - v_4 \\ v_2 - v_6 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -i \end{pmatrix} \begin{pmatrix} v_0 - v_4 \\ v_2 - v_6 \end{pmatrix},$$

т. е. эти вычисления требуют только 4 операции сложения. Умножения здесь тривиальны.

Для вычисления  $V_{2k+1}^2$  ( $k = 0, 1, 2, 3$ ) применим алгоритм Рейдера. Так как

$$5^0 = 1, \quad 5^1 = 5, \quad 5^0 \cdot 7 = 7, \quad 5^1 \cdot 7 = 3 \pmod{8},$$

$$5^{-0} = 1, \quad 5^{-1} = 5, \quad 5^{-0} \cdot 7 = 7, \quad 5^{-1} \cdot 7 = 3 \pmod{8},$$

то получаем

$$\begin{pmatrix} V_1^2 \\ V_5^2 \\ V_7^2 \\ V_3^2 \end{pmatrix} = \begin{pmatrix} \omega & \omega^5 & \omega^7 & \omega^3 \\ \omega^5 & \omega & \omega^3 & \omega^7 \\ \omega^7 & \omega^3 & \omega & \omega^5 \\ \omega^3 & \omega^7 & \omega^5 & \omega \end{pmatrix} \begin{pmatrix} v_1 \\ v_5 \\ v_7 \\ v_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix}.$$



$$\begin{pmatrix} \frac{1}{2}(\omega + \omega^7) & \frac{1}{2}(\omega^5 + \omega^3) & 0 & 0 \\ \frac{1}{2}(\omega^5 + \omega^3) & \frac{1}{2}(\omega + \omega^7) & 0 & 0 \\ 0 & 0 & \frac{1}{2}(\omega - \omega^7) & \frac{1}{2}(\omega^5 - \omega^3) \\ 0 & 0 & \frac{1}{2}(\omega^5 - \omega^3) & \frac{1}{2}(\omega - \omega^7) \end{pmatrix} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_5 \\ v_7 \\ v_3 \end{pmatrix} =$$

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} \cos \theta & \cos 5\theta & 0 & 0 \\ \cos 5\theta & \cos \theta & 0 & 0 \\ 0 & 0 & i \sin \theta & i \sin 5\theta \\ 0 & 0 & i \sin 5\theta & i \sin \theta \end{pmatrix} \begin{pmatrix} v_1 + v_7 \\ v_5 + v_3 \\ v_1 - v_7 \\ v_5 - v_3 \end{pmatrix},$$

т. е. две циклические свертки и некоторые дополнительные сложения.

Для вычисления каждой из двух 2-точечных циклических сверток потребуется одно умножение, так как умножения, связанные с модулем  $x - 1$ , выполнять не требуется. Действительно, надо вычислить

$$s_1(x) = d_1(x) g_1(x) \pmod{x^2 - 1}$$

и

$$s_2(x) = d_2(x) g_2(x) \pmod{x^2 - 1},$$

где

$$\begin{aligned} d_1(x) &= (v_1 + v_7) + (v_5 + v_3)x, & g_1(x) &= \cos \theta + \cos 5\theta x, \\ d_2(x) &= (v_1 - v_7) + (v_5 - v_3)x, & g_2(x) &= i \sin \theta + i \sin 5\theta x. \end{aligned}$$

Применим алгоритм Винограда, при этом рассмотрим разложение

$$x^2 - 1 = (x - 1)(x + 1)$$

и заметим, что

$$\begin{aligned} g_1(x) \pmod{x - 1} &= \cos \theta + \cos 5\theta = 2 \cos 3\theta \cos 2\theta = 0, \\ g_2(x) \pmod{x - 1} &= i(\sin \theta + \sin 5\theta) = 2i \sin 3\theta \cos 2\theta = 0, \end{aligned}$$

поскольку  $2\theta = -\frac{4\pi}{8} = -\frac{\pi}{2}$  и  $\cos\left(-\frac{\pi}{2}\right) = 0$ .

Так как

$$\begin{aligned} g_1(x) \pmod{x + 1} &= \cos \theta - \cos 5\theta = 2 \sin 3\theta \sin 2\theta = -2 \sin 3\theta = G_1, \\ d_1(x) \pmod{x + 1} &= (v_1 + v_7) - (v_5 + v_3) = D_1, \end{aligned}$$

то

$$s_1(x) = G_1 D_1 \left(-\frac{1}{2}\right) (x + 1) = \sin 3\theta D_1 (1 + x)$$

и для вычисления  $s_1(x)$  требуется одно умножение на вещественное число и 3 сложения. Аналогично, так как

$$\begin{aligned} g_2(x) \pmod{x + 1} &= i(\sin \theta - \sin 5\theta) = -i2 \sin 2\theta \cos 3\theta = 2i \cos 3\theta = G_2, \\ d_2(x) \pmod{x + 1} &= (v_1 - v_7) - (v_5 - v_3) = D_2, \end{aligned}$$

то

$$s_2(x) = G_2 D_2 \left(\frac{1}{2}\right) (x + 1) = i \cos 3\theta D_2 (1 + x)$$

и опять требуется одно умножение и 3 сложения. Тогда с учетом постсложений, требующихся для окончательного нахождения  $V_{2k+1}^2$  ( $k = 0, 1, 2, 3$ ), получаем всего 10 сложений и 2 умножения, и для вычисления  $V_{2k+1}$  ( $k = 0, 1, 2, 3$ ) —  $10 + 4 + 4 = 18$  операций сложения и 2 умножения. Значит, для вычисления всех выходных компонент 8-точечного преобразования Фурье потребуется всего 2 нетривиальных умножения (на вещественное и чисто мнимое числа) и 26 операций сложения.

Мы видели, что в обоих примерах при вычислении циклических сверток, получающихся в результате применения алгоритма Рейдера, все умножения являются либо вещественными, либо чисто мнимыми, причем часть умножений при вычислении циклических сверток длины  $\frac{n}{4}$  выполнять не нужно, так как многочлен  $g(x)$  по модулю  $x^{\frac{n}{8}} - 1$  оказывается равным нулю. На самом деле, это общая ситуация, которая описывается следующей теоремой.

**Теорема.** Пусть  $m > 2$  целое число и  $g(x, y)$  – обобщенный двумерный многочлен Рейдера

$$g(x, y) = \sum_{j=0}^{2^{m-2}-1} \sum_{l=0}^1 \omega^{3^j(-1)^l} x^j y^l,$$

где  $\omega$  – корень степени  $2^m$  из единицы. Тогда:

- (1). Коэффициенты многочлена  $g(x, y) \pmod{y-1}$  являются вещественными числами;
- (2). Коэффициенты многочлена  $g(x, y) \pmod{y+1}$  являются чисто мнимыми числами;
- (3). Коэффициенты многочлена  $g(x, y) \pmod{x^{2^{m-3}}-1}$  равны нулю.

*Доказательство.* Действительно,

$$g(x, y) \pmod{y-1} = \sum_{j=0}^{2^{m-2}-1} \omega^{3^j} x^j + \sum_{j=0}^{2^{m-2}-1} \omega^{-3^j} x^j = \sum_{j=0}^{2^{m-2}-1} (\omega^{3^j} + \omega^{-3^j}) x^j = \sum_{j=0}^{2^{m-2}-1} 2 \cos(3^j \theta) x^j,$$

$$g(x, y) \pmod{y+1} = \sum_{j=0}^{2^{m-2}-1} \omega^{3^j} x^j - \sum_{j=0}^{2^{m-2}-1} \omega^{-3^j} x^j = \sum_{j=0}^{2^{m-2}-1} (\omega^{3^j} - \omega^{-3^j}) x^j = \sum_{j=0}^{2^{m-2}-1} 2i \sin(3^j \theta) x^j.$$

Рассмотрим

$$g(x, y) \pmod{x^{2^{m-3}}-1} = \sum_{j=0}^{2^{m-3}-1} \sum_{l=0}^1 \{\omega^{3^j(-1)^l} + \omega^{3^{j+2^{m-3}}(-1)^l}\} x^j y^l =$$

$$\sum_{j=0}^{2^{m-3}-1} (\omega^{3^j} + \omega^{3^{j+2^{m-3}}}) x^j + \sum_{j=0}^{2^{m-3}-1} (\omega^{-3^j} + \omega^{-3^{j+2^{m-3}}}) x^j y = \sum_{j=0}^{2^{m-3}-1} \tilde{g}_j x^j + \sum_{j=0}^{2^{m-3}-1} \hat{g}_j x^j y.$$

В общем случае коэффициент  $\tilde{g}_k$  дается равенством  $\tilde{g}_k = \omega^{3^k} + \omega^{3^{k+2^{m-3}}}$ ,  $k = 0, 1, \dots, 2^{m-3} - 1$ , а коэффициент  $\hat{g}_k$  – равенством  $\hat{g}_k = \omega^{-3^k} + \omega^{-3^{k+2^{m-3}}}$ ,  $k = 0, 1, \dots, 2^{m-3} - 1$ . Так как

$$\tilde{g}_k = (\omega^{3^k})^{3^0} + (\omega^{3^k})^{3^{2^{m-3}}}, \quad \hat{g}_k = (\omega^{-3^k})^{-3^0} + (\omega^{-3^k})^{-3^{2^{m-3}}}$$

и  $\omega^{3^k}$ ,  $\omega^{-3^k}$  опять являются корнями степени  $2^m$  из единицы, то достаточно доказать, что  $\tilde{g}_0$  и  $\hat{g}_0$  равны нулю. Очевидно, имеем

$$\tilde{g}_0 = \omega^{3^0} + \omega^{3^{2^{m-3}}} = \frac{\omega^{3^0} - \omega^{3^{2^{m-2}}}}{1 - \omega^{3^{2^{m-3}}}} = \frac{\omega - \omega}{1 - \omega^{3^{2^{m-3}}}} = 0$$

в силу  $3^{2^{m-2}} \equiv 1 \pmod{2^m}$ .

Аналогично,

$$\hat{g}_0 = \omega^{-3^0} + \omega^{-3^{2^{m-3}}} = \frac{\omega^{-3^0} - \omega^{-3^{2^{m-2}}}}{1 - \omega^{-3^{2^{m-3}}}} = \frac{\omega^{-1} - \omega^{-1}}{1 - \omega^{-3^{2^{m-3}}}} = 0.$$

Значит, все коэффициенты многочлена  $g(x, y) \pmod{x^{2^{m-3}}-1}$  равны нулю.



## §21. СВЕРТКА В КОНЕЧНОМ ПОЛЕ

Свертка в поле вещественных чисел в некоторых случаях может быть заменена сверткой в подходящем поле Галуа. В реальных задачах вещественные числа записываются в виде конечного числа десятичных или двоичных знаков; в цифровой обработке сигналов обычно используют запись с фиксированной запятой. При вычислениях можно предполагать, что эта запятая находится справа, т. е. все числа являются целыми (это эквивалентно умножению всех данных на некоторый масштабный множитель). Таким образом, линейную свертку в поле вещественных чисел можно заменить сверткой в кольце целых чисел. Внешний вид уравнения, описывающего свертку, не меняется, но операции приобретают смысл стандартной целочисленной арифметики. Затем уравнение в кольце целых чисел можно погрузить в подходящее поле Галуа, например, в поле  $GF(p) \cong \mathbb{Z}_p$ , если простое число  $p$  достаточно велико. Если все входящие в свертку целые числа положительны, то  $p$  должно быть столь большим, чтобы выходные коэффициенты свертки не превосходили  $p - 1$ . Тогда выполняется сравнение

$$s(x) = g(x) d(x) \pmod{p},$$

и условие вычислений по модулю  $p$  излишне. Если выходные коэффициенты могут принимать и отрицательные значения, то  $p$  надо выбирать таким образом, чтобы  $s_i$  попадали в диапазон  $-\frac{p-1}{2} < s_i \leq \frac{p-1}{2}$ . Тогда отрицательные целые числа могут быть представлены положительными целыми числами в интервале от  $\frac{p+1}{2}$  до  $p-1$ .

Свертка в поле Галуа может быть вычислена любым из пригодных в данном поле методов. Можно строить прямые методы вычисления свертки в конечном поле, пользуясь теми же быстрыми алгоритмами, которые работают в поле вещественных чисел, можно модифицировать уже построенные алгоритмы свертки для поля вещественных чисел так, чтобы их можно было использовать в конечном поле.

Так как циклическая свертка в поле Галуа приводит к тому же самому ответу, что и циклическая свертка в кольце целых чисел, то можно использовать преобразование Фурье данного поля Галуа и соответствующий алгоритм.

Для преобразования алгоритма свертки, построенного для поля вещественных чисел, в алгоритм свертки в поле Галуа  $GF(q)$  характеристики  $p$  начнем с алгоритма свертки, записанного в виде

$$s = C[(Ag) \bullet (Bd)],$$

где  $A$ ,  $B$  и  $C$  – матрицы с рациональными элементами, а знак  $\bullet$  означает покомпонентное перемножение векторов  $Ag$  и  $Bd$ . Умножим обе части равенства на наименьшее целое число  $L$  такое, чтобы ликвидировать все знаменатели во всех компонентах; тогда равенство переписывается в виде

$$Ls = C'[(A'g) \bullet (B'd)],$$

где  $L$  – целое число и  $A'$ ,  $B'$ ,  $C'$  – целочисленные матрицы. Это равенство можно рассматривать как алгоритм вычисления целочисленной свертки; следовательно, его можно рассматривать как уравнение по модулю  $p$ :

$$Ls = C'[(A'g) \bullet (B'd)] \pmod{p}.$$

Если  $L \not\equiv 0 \pmod{p}$ , то, умножив обе части равенства на  $L^{-1} \pmod{p}$ , получаем алгоритм свертки в  $GF(p)$  и, следовательно, в любом расширении поля  $GF(p)$ .

Если  $L \equiv 0 \pmod{p}$ , то алгоритм свертки для поля вещественных чисел не переносится в поле Галуа, и надо строить такой алгоритм прямо в самом поле Галуа. Эта задача возникает даже тогда, когда  $L$  не равно нулю по модулю  $p$ , так как может оказаться, что алгоритм, построенный специально для данного поля Галуа, лучше алгоритма, перенесенного из другого поля. Для построения такого алгоритма можно пользоваться всеми, рассмотренными выше методами (см. § 3 – 6); при этом, конечно, требуется, чтобы разложение

$$m(x) = m_0(x) m_1(x) \dots m_{s-1}(x)$$

было разложением в том поле, в котором вычисляется свертка.



В поле большой характеристики  $p$  множитель  $L$  будет намного меньше  $p$ , поэтому все алгоритмы свертки для поля рациональных чисел могут быть перенесены в  $GF(p)$ . Для поля малой характеристики перенести алгоритм, разработанный для поля вещественных чисел, в конечное поле не всегда удастся. Важную роль во многих приложениях играют поля Галуа характеристики 2. Займемся такими полями.

Начнем с построения алгоритма 3-точечной циклической свертки для полей характеристики два. В поле вещественных чисел этот алгоритм записывается уравнением

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & -1 \\ 1 & -1 & -1 & 2 \\ 1 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} G_0 & & & \\ & G_1 & & \\ & & G_2 & \\ & & & G_3 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & -1 \\ 0 & 1 & -1 \\ 1 & 1 & -2 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \end{pmatrix},$$

$$G_0 = \frac{1}{3}(g_0 + g_1 + g_2), \quad G_1 = g_0 - g_2, \quad G_2 = g_1 - g_2, \quad G_3 = \frac{1}{3}(g_0 + g_1 - 2g_2).$$

Ни один из знаменателей не кратен двум. Следовательно, алгоритм может быть перенесен в поле характеристики два. Соответственно новый алгоритм в поле характеристики два записывается в виде

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} g_0 + g_1 + g_2 & & & \\ & g_0 + g_2 & & \\ & & g_1 + g_2 & \\ & & & g_0 + g_1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \end{pmatrix}.$$

С другой стороны, алгоритм 2-точечной свертки содержит элементы с четными знаменателями

$$\begin{pmatrix} s_0 \\ s_1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} \frac{1}{2}(g_0 + g_1) & \\ & \frac{1}{2}(g_0 - g_1) \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \end{pmatrix}$$

и, следовательно, не может быть перенесен в поле характеристики два, так как  $2 \equiv 0 \pmod{2}$ . Лучший алгоритм 2-точечной циклической свертки в полях характеристики два задается равенством

$$\begin{pmatrix} s_0 \\ s_1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} g_0 + g_1 & & \\ & g_1 & \\ & & g_0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \end{pmatrix}.$$

Действительно,

$$s(x) = (d_0 + d_1 x)(g_0 + g_1 x) \pmod{x^2 - 1} = (d_0 g_0 + d_1 g_1) + (d_0 g_1 + d_1 g_0)x,$$

и так как

$$(g_0 + g_1)d_0 + g_1(d_0 + d_1) = g_0 d_0 + g_1 d_1 + 2d_0 g_1 \equiv g_0 d_0 + d_1 g_1 \pmod{2},$$

$$(g_0 + g_1)d_0 + g_0(d_0 + d_1) = 2g_0 d_0 + g_1 d_0 + g_0 d_1 \equiv g_1 d_0 + g_0 d_1 \pmod{2},$$

$$\Rightarrow s_0 = d_0 g_0 + d_1 g_1 \text{ и } s_1 = d_0 g_1 + d_1 g_0.$$

Заметим, что алгоритм содержит 3 умножения. Причина необходимости трех умножений заключается в том, что над полем  $GF(2)$  многочлен  $x^2 - 1$  не разлагается в произведение двух взаимно простых множителей, так как  $-1 \equiv 1 \pmod{2}$  и, следовательно,  $x^2 - 1 \equiv x^2 + 1 \equiv (x + 1)^2$ , поэтому в конструкцию алгоритма в качестве модуля входит многочлен второй степени  $x^2 + 1$ .

Такая ситуация, когда число умножений в конечном поле больше числа умножений в поле вещественных (или рациональных) чисел, встречается для циклических сверток многих длин. Это происходит потому, что несколько простых делителей многочлена  $x^n - 1$  оказываются равными. С другой стороны, некоторые циклические свертки в полях конечной характеристики требуют меньше умножений, чем свертки тех же длин в поле  $\mathbb{R}$  (или  $\mathbb{Q}$ ).

Например, над полем  $\mathbb{Q}$  разложение на неприводимые множители многочлена  $x^7 - 1$  имеет вид

$$x^7 - 1 = (x - 1)(x^6 + x^5 + x^4 + x^3 + x^2 + x + 1),$$



а над полем  $GF(2)$  это разложение можно продолжить:

$$x^7 - 1 = (x - 1)(x^3 + x^2 + 1)(x^3 + x + 1).$$

Следовательно, оптимальный по числу умножений алгоритм 7-точечной циклической свертки над полем  $\mathbb{Q}$  должен содержать 12 умножений, а оптимальный алгоритм той же свертки над полем Галуа характеристики два должен содержать 11 умножений. А вот в поле характеристики 127 тот же многочлен раскладывается на линейные множители

$$x^7 - 1 = (x - 1)(x - 2)(x - 4)(x - 8)(x - 16)(x - 32)(x - 64) = \\ (x - 1)(x - 2)(x - 2^2)(x - 2^3)(x - 2^4)(x - 2^5)(x - 2^6).$$

Это нетрудно увидеть, если заметить, что  $127 = 2^7 - 1$ , откуда  $2^7 - 1 \equiv 0 \pmod{127}$ , а значит, и  $(2^k)^7 \equiv 1 \pmod{127}$ , т. е.  $2^k$  также является корнем уравнения  $x^7 - 1 = 0$ . Значит, в поле характеристики 127 оптимальный по числу умножений алгоритм 7-точечной циклической свертки должен содержать всего 7 умножений.

В поле характеристики два можно построить алгоритм вычисления 7-точечной циклической свертки, используя методы § 3 – 6, который содержит 13 умножений.

Пусть

$$d(x) = d_0 + d_1x + d_2x^2 + d_3x^3 + d_4x^4 + d_5x^5 + d_6x^6, \\ g(x) = g_0 + g_1x + g_2x^2 + g_3x^3 + g_4x^4 + g_5x^5 + g_6x^6.$$

В поле характеристики два имеем следующее разложение многочлена  $x^7 - 1$  в произведение попарно взаимно простых неприводимых многочленов

$$x^7 - 1 = (x + 1)(x^3 + x + 1)(x^3 + x^2 + 1).$$

Найдем вычеты  $d(x)$  и  $g(x)$  по модулям  $x + 1$ ,  $x^3 + x + 1$ ,  $x^3 + x^2 + 1$ . Получаем

$$d_0(x) = d_0 + d_1 + d_2 + d_3 + d_4 + d_5 + d_6, \\ d_1(x) = (d_0 + d_3 + d_5 + d_6) + (d_1 + d_3 + d_4 + d_5)x + (d_2 + d_4 + d_5 + d_6)x^2 = d_0^{(1)} + d_1^{(1)}x + d_2^{(1)}x^2, \\ d_2(x) = (d_0 + d_3 + d_4 + d_5) + (d_1 + d_4 + d_5 + d_6)x + (d_2 + d_3 + d_4 + d_6)x^2 = d_0^{(2)} + d_1^{(2)}x + d_2^{(2)}x^2, \\ g_0(x) = g_0 + g_1 + g_2 + g_3 + g_4 + g_5 + g_6, \\ g_1(x) = (g_0 + g_3 + g_5 + g_6) + (g_1 + g_3 + g_4 + g_5)x + (g_2 + g_4 + g_5 + g_6)x^2 = g_0^{(1)} + g_1^{(1)}x + g_2^{(1)}x^2, \\ g_2(x) = (g_0 + g_3 + g_4 + g_5) + (g_1 + g_4 + g_5 + g_6)x + (g_2 + g_3 + g_4 + g_6)x^2 = g_0^{(2)} + g_1^{(2)}x + g_2^{(2)}x^2$$

в силу следующих сравнений:

$$\begin{aligned} x^3 &\equiv x + 1 \pmod{x^3 + x + 1}, & x^3 &\equiv x^2 + 1 \pmod{x^3 + x^2 + 1}, \\ x^4 &\equiv x^2 + x \pmod{x^3 + x + 1}, & x^4 &\equiv x^2 + x + 1 \pmod{x^3 + x^2 + 1}, \\ x^5 &\equiv x^2 + x + 1 \pmod{x^3 + x + 1}, & x^5 &\equiv x + 1 \pmod{x^3 + x^2 + 1}, \\ x^6 &\equiv x^2 + 1 \pmod{x^3 + x + 1}, & x^6 &\equiv x^2 + x \pmod{x^3 + x^2 + 1}. \end{aligned}$$

Далее,

$$s_0(x) = d_0(x)g_0(x), \\ s_1(x) = d_1(x)g_1(x) \pmod{x^3 + x + 1} = s_0^{(1)} - s_1^{(1)}x - s_2^{(1)}x^2, \quad (1)$$

$$s_2(x) = d_2(x)g_2(x) \pmod{x^3 + x^2 + 1} = s_0^{(2)} - s_1^{(2)}x - s_2^{(2)}x^2. \quad (2)$$

В дальнейшем подзадачи (1) и (2) придется решать отдельно. А пока воспользуемся китайской теоремой об остатках для восстановления многочлена  $s(x) = d(x)g(x) \pmod{x^7 - 1}$ , считая коэффициенты многочленов  $s_1(x)$  и  $s_2(x)$  известными. Для нахождения  $s(x)$  используем таблицу

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$
0	$x + 1$	$x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$	1
1	$x^3 + x + 1$	$x^4 + x^2 + x + 1$	1
2	$x^3 + x^2 + 1$	$x^4 + x^3 + x^2 + 1$	$x^2 + 1$ ,

откуда

$$\begin{aligned}
s(x) &= s_0(x)(x^6 + x^5 + x^4 + x^3 + x^2 + x + 1) + s_1(x)(x^4 + x^2 + x + 1) + s_2(x)(x^2 + 1)(x^4 + x^3 + x^2 + 1) \pmod{x^7 - 1} \\
&= s_0(x)(x^6 + x^5 + x^4 + x^3 + x^2 + x + 1) + s_0^{(1)}(x^4 + x^2 + x + 1) + s_1^{(1)}(x^5 + x^3 + x^2 + x) + s_2^{(1)}(x^6 + x^4 + x^3 + x^2) + \\
&\quad s_0^{(2)}(x^6 + x^5 + x^3 + 1) + s_1^{(2)}(x^7 + x^6 + x^4 + x) + s_2^{(1)}(x^8 + x^7 + x^5 + x^2) \pmod{x^7 - 1} = \\
&= s_0(x)(x^6 + x^5 + x^4 + x^3 + x^2 + x + 1) + s_0^{(1)}(x^4 + x^2 + x + 1) + s_1^{(1)}(x^5 + x^3 + x^2 + x) + s_2^{(1)}(x^6 + x^4 + x^3 + x^2) + \\
&\quad s_0^{(2)}(x^6 + x^5 + x^3 + 1) + s_1^{(2)}(x^6 + x^4 + x + 1) + s_2^{(2)}(x^5 + x^2 + x + 1).
\end{aligned}$$

Перейдем к решению подзадач. Начнем с подзадачи (1):

$$s_1(x) = d_1(x)g_1(x) = (d_0^{(1)} + d_1^{(1)}x + d_2^{(1)}x^2)(g_0^{(1)} + g_1^{(1)}x + g_2^{(1)}x^2) \pmod{x^3 + x + 1}.$$

Найдем сначала линейную свертку многочленов  $d_1(x)$  и  $g_1(x)$ , а затем полученный результат приведем по модулю  $x^3 + x + 1$ . Для вычисления линейной свертки воспользуемся модифицированным алгоритмом Винограда. Выберем многочлен

$$m(x) = x(x + 1)(x^2 + x + 1),$$

$$\deg m(x) = \deg d_1(x) + \deg g_1(x) = \deg d_1(x)g_1(x) \text{ и вычислим}$$

$$r_1(x) = d_1(x)g_1(x) \pmod{m(x)},$$

тогда

$$s_1(x) = r_1(x) + d_2^{(1)}g_2^{(1)}m(x) = r_1(x) + d_2^{(1)}g_2^{(1)}(x^4 + x) \pmod{x^3 + x + 1}.$$

Вычеты  $d_1(x)$  и  $g_1(x)$  по модулям  $x$ ,  $x + 1$ ,  $x^2 + x + 1$  соответственно равны

$$d_{10}(x) = d_0^{(1)},$$

$$d_{11}(x) = d_0^{(1)} + d_1^{(1)} + d_2^{(1)},$$

$$d_{12}(x) = (d_0^{(1)} + d_2^{(1)}) + (d_1^{(1)} + d_2^{(1)})x = d_0^{(12)} + d_1^{(12)}x,$$

$$g_{10}(x) = g_0^{(1)},$$

$$g_{11}(x) = g_0^{(1)} + g_1^{(1)} + g_2^{(1)},$$

$$g_{12}(x) = (g_0^{(1)} + g_2^{(1)}) + (g_1^{(1)} + g_2^{(1)})x = g_0^{(12)} + g_1^{(12)}x.$$

Найдем попарные произведения соответствующих вычетов по модулям  $x$ ,  $x + 1$ ,  $x^2 + x + 1$ :

$$s_{10}(x) = d_{10}(x)g_{10}(x) = d_0^{(1)}g_0^{(1)},$$

$$s_{11}(x) = d_{11}(x)g_{11}(x) = (d_0^{(1)} + d_1^{(1)} + d_2^{(1)})(g_0^{(1)} + g_1^{(1)} + g_2^{(1)})$$

$$s_{12}(x) = d_{12}(x)g_{12}(x) \pmod{x^2 + x + 1} = s_0^{(12)} + s_1^{(12)}x. \quad (3)$$



При этом мы приходим к новой подзадаче вычисления произведения двух многочленов  $d_{12}(x)$  и  $g_{12}(x)$  по модулю  $x^2 + x + 1$ . Считая пока  $s_0^{(12)}$  и  $s_1^{(12)}$  известными, продолжим решение подзадачи (1). Из таблицы

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$
0	$x$	$x^3 + 1$	1
1	$x + 1$	$x^3 + x^2 + x$	1
2	$x^2 + x + 1$	$x^2 + x$	1

следует, что

$$r_1(x) = s_{10}(x)(x^3 + 1) + s_{11}(x)(x^3 + x^2 + x) + (s_0^{(12)} + s_1^{(12)}x)(x^2 + x) \pmod{x^4 + x} =$$

$$s_{10}(x)(x^3 + 1) + s_{11}(x)(x^3 + x^2 + x) + s_0^{(12)}(x^2 + x) + s_1^{(12)}(x^3 + x^2),$$

откуда

$$s_1(x) = s_{10}(x)(x^3 + 1) + s_{11}(x)(x^3 + x^2 + x) + s_0^{(12)}(x^2 + x) + s_1^{(12)}(x^3 + x^2) + d_2^{(1)}g_2^{(1)}(x^4 +$$

$$+ x) \pmod{x^3 + x + 1} = s_{10}(x)x + s_{11}(x)(x^2 + 1) + s_0^{(12)}(x^2 + x) + s_1^{(12)}(x^2 + x + 1) + d_2^{(1)}g_2^{(1)}x^2.$$

Чтобы закончить вычисление  $s_1(x)$ , требуется решить подзадачу (3):

$$s_{12}(x) = (d_0^{(12)} + d_1^{(12)}x)(g_0^{(12)} + g_1^{(12)}x) \pmod{x^2 + x + 1}.$$

Применим тот же метод, что и при решении подзадачи (1). Выберем многочлен второй степени  $\tilde{m}(x) = x(x + 1)$  и воспользуемся модифицированным алгоритмом Винограда:

$$d_{120}(x) = d_0^{(12)}, \quad g_{120}(x) = g_0^{(12)},$$

$$d_{121}(x) = d_0^{(12)} + d_1^{(12)}, \quad g_{121}(x) = g_0^{(12)} + g_1^{(12)},$$

$$s_{120}(x) = d_{120}(x)g_{120}(x) = d_0^{(12)}g_0^{(12)},$$

$$s_{121}(x) = d_{121}(x)g_{121}(x) = (d_0^{(12)} + d_1^{(12)})(g_0^{(12)} + g_1^{(12)}).$$

Кроме того, нам нужна константа  $d_1^{(12)}g_1^{(12)}$ . Из таблицы

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$
0	$x$	$x + 1$	1
1	$x + 1$	$x$	1

следует

$$s_{12}(x) = s_{120}(x)(x + 1) + s_{121}(x)x + d_1^{(12)}g_1^{(12)}(x^2 + x) \pmod{x^2 + x + 1} =$$

$$s_{120}(x)(x + 1) + s_{121}(x)x + d_1^{(12)}g_1^{(12)} = (s_{120}(x) + d_1^{(12)}g_1^{(12)}) + (s_{120}(x) + s_{121}(x))x = s_0^{(12)} + s_1^{(12)}x,$$

т. е.  $s_0^{(12)} = s_{120}(x) + d_1^{(12)}g_1^{(12)}$ ,  $s_1^{(12)} = s_{120}(x) + s_{121}(x)$ .

Итак, получено решение подзадачи (1). Мы можем записать вычисление коэффициентов многочлена  $s_1(x)$  в матричном виде

$$\begin{pmatrix} s_0^{(1)} \\ s_1^{(1)} \\ s_2^{(1)} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} s_{10}(x) \\ s_{11}(x) \\ s_0^{(12)} \\ s_1^{(12)} \\ d_2^{(1)}g_2^{(1)} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} s_{10}(x) \\ s_{11}(x) \\ s_{120}(x) \\ s_{121}(x) \\ d_1^{(12)}g_1^{(12)} \\ d_2^{(1)}g_2^{(1)} \end{pmatrix} =$$

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} g_0^{(1)} & & & & & \\ & g_0^{(1)} + g_1^{(1)} + g_2^{(1)} & & & & \\ & & g_0^{(12)} & & & \\ & & & g_0^{(12)} + g_1^{(12)} & & \\ & & & & g_1^{(12)} & \\ & & & & & g_2^{(1)} \end{pmatrix} \begin{pmatrix} d_0^{(1)} + d_1^{(1)} + d_2^{(1)} \\ d_0^{(12)} + d_1^{(12)} \\ d_0^{(12)} + d_1^{(12)} \\ d_1^{(12)} \\ d_2^{(1)} \end{pmatrix} =$$

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} G_1 & & & & & \\ & G_2 & & & & \\ & & G_3 & & & \\ & & & G_4 & & \\ & & & & G_5 & \\ & & & & & G_6 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{pmatrix},$$

где

$$\begin{pmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \\ G_6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} g_0 \\ g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_5 \\ g_6 \end{pmatrix}.$$

Осталось решить подзадачу (2), т. е. найти

$$s_2(x) = d_2(x) g_2(x) = (d_0^{(2)} + d_1^{(2)} x + d_2^{(2)} x^2) (g_0^{(2)} + g_1^{(2)} x + g_2^{(2)} x^2) \pmod{x^3 + x^2 + 1}.$$

Решение этой подзадачи аналогично решению подзадачи (1). Возьмем тот же многочлен  $m(x) = x(x+1)(x^2+x+1)$  и найдем

$$r_2(x) = d_2(x) g_2(x) \pmod{m(x)},$$

тогда

$$s_2(x) = r_2(x) + d_2^{(2)} g_2^{(2)} m(x) \pmod{x^3 + x^2 + 1}.$$

Вычеты  $d_2(x)$  и  $g_2(x)$  соответственно равны

$$\begin{aligned} d_{20}(x) &= d_0^{(2)}, \\ d_{21}(x) &= d_0^{(2)} + d_1^{(2)} + d_2^{(2)}, \\ d_{22}(x) &= (d_0^{(2)} + d_2^{(2)}) + (d_1^{(2)} + d_2^{(2)}) x = d_0^{(22)} + d_1^{(22)} x, \\ g_{20}(x) &= g_0^{(2)}, \\ g_{21}(x) &= g_0^{(2)} + g_1^{(2)} + g_2^{(2)}, \\ g_{22}(x) &= (g_0^{(2)} + g_2^{(2)}) + (g_1^{(2)} + g_2^{(2)}) x = g_0^{(22)} + g_1^{(22)} x, \end{aligned}$$

откуда

$$\begin{aligned} s_{20}(x) &= d_{20}(x) g_{20}(x) = d_0^{(2)} g_0^{(2)}, \\ s_{21}(x) &= d_{21}(x) g_{21}(x) = (d_0^{(2)} + d_1^{(2)} + d_2^{(2)}) (g_0^{(2)} + g_1^{(2)} + g_2^{(2)}), \\ s_{22}(x) &= d_{22}(x) g_{22}(x) \pmod{x^2 + x + 1} = s_0^{(22)} + s_1^{(22)} x. \end{aligned} \tag{4}$$

Значит,

$$r_2(x) = s_{20}(x)(x^3 + 1) + s_{21}(x)(x^3 + x^2 + x) + (s_0^{(22)} + s_1^{(22)} x)(x^2 + x) \pmod{x^4 + x} =$$



$$s_{20}(x)(x^3 + 1) + s_{21}(x)(x^3 + x^2 + x) + s_0^{(22)}(x^2 + x) + s_1^{(22)}(x^3 + x^2),$$

и, следовательно,

$$s_2(x) = s_{20}(x)(x^3 + 1) + s_{21}(x)(x^3 + x^2 + x) + s_0^{(22)}(x^2 + x) + s_1^{(22)}(x^3 + x^2) + d_2^{(2)}g_2^{(2)}(x^4 +$$

$$+ x) \pmod{x^3 + x^2 + 1} = s_{20}(x)x^2 + s_{21}(x)(x + 1) + s_0^{(22)}(x^2 + x) + s_1^{(22)} + d_2^{(1)}g_2^{(1)}(x^2 + 1).$$

Для нахождения коэффициентов  $s_0^{(22)}$  и  $s_1^{(22)}$  решаем подзадачу (4) аналогично тому, как решали подзадачу (3). Опять выберем  $\tilde{m}(x) = x(x + 1)$  и вычислим вычеты многочленов  $d_{22}(x)$  и  $g_{22}(x)$ :

$$d_{220}(x) = d_0^{(22)}, \quad g_{220}(x) = g_0^{(22)},$$

$$d_{221}(x) = d_0^{(22)} + d_1^{(22)}, \quad g_{221}(x) = g_0^{(22)} + g_1^{(22)}.$$

Далее,

$$s_{220}(x) = d_{220}(x)g_{220}(x) = d_0^{(22)}g_0^{(22)},$$

$$s_{221}(x) = d_{221}(x)g_{221}(x) = (d_0^{(22)} + d_1^{(22)})(g_0^{(22)} + g_1^{(22)}),$$

откуда

$$s_{22}(x) = s_{220}(x)(x + 1) + s_{221}(x)x + d_1^{(22)}g_1^{(22)}(x^2 + x) \pmod{x^2 + x + 1} =$$

$$s_{220}(x)(x + 1) + s_{221}(x)x + d_1^{(22)}g_1^{(22)} = (s_{220}(x) + d_1^{(22)}g_1^{(22)}) + (s_{220}(x) + s_{221}(x))x = s_0^{(22)} + s_1^{(22)}x,$$

$$\text{т. е. } s_0^{(22)} = s_{220}(x) + d_1^{(22)}g_1^{(22)}, \quad s_1^{(22)} = s_{220}(x) + s_{221}(x).$$

Таким образом, решение подзадачи (2) в матричном виде:

$$\begin{pmatrix} s_0^{(2)} \\ s_1^{(2)} \\ s_2^{(2)} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} s_{20}(x) \\ s_{21}(x) \\ s_0^{(22)} \\ s_1^{(22)} \\ d_2^{(2)}g_2^{(2)} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} s_{20}(x) \\ s_{21}(x) \\ s_{220}(x) \\ s_{221}(x) \\ d_1^{(22)}g_1^{(22)} \\ d_2^{(2)}g_2^{(2)} \end{pmatrix} =$$

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} g_0^{(2)} \\ g_0^{(2)} + g_1^{(2)} + g_2^{(2)} \\ g_0^{(22)} \\ g_0^{(22)} + g_1^{(22)} \\ g_1^{(22)} \\ g_2^{(2)} \end{pmatrix} \begin{pmatrix} d_0^{(2)} \\ d_0^{(2)} + d_1^{(2)} + d_2^{(2)} \\ d_0^{(22)} \\ d_0^{(22)} + d_1^{(22)} \\ d_1^{(22)} \\ d_2^{(2)} \end{pmatrix} =$$

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} G_7 & & & & & \\ & G_8 & & & & \\ & & G_9 & & & \\ & & & G_{10} & & \\ & & & & G_{11} & \\ & & & & & G_{12} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{pmatrix},$$

где

$$\begin{pmatrix} G_7 \\ G_8 \\ G_9 \\ G_{10} \\ G_{11} \\ G_{12} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} g_0 \\ g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_5 \\ g_6 \end{pmatrix}.$$





Вычеты  $d(x)$  и  $g(x)$  по модулям  $x - 1, x - 10, x + 11, x + 1, x + 10$  и  $x - 11$  соответственно равны:

$$\begin{aligned} d_0(x) &= d_0 + d_1 + d_2 + d_3 + d_4 + d_5, & g_0(x) &= g_0 + g_1 + g_2 + g_3 + g_4 + g_5, \\ d_1(x) &= d_0 + 10d_1 - 11d_2 + d_3 + 10d_4 - 11d_5, & g_1(x) &= g_0 + 10g_1 - 11g_2 + g_3 + 10g_4 - 11g_5, \\ d_2(x) &= d_0 - 11d_1 + 10d_2 + d_3 - 11d_4 + 10d_5, & g_2(x) &= g_0 - 11g_1 + 10g_2 + g_3 - 11g_4 + 10g_5, \\ d_3(x) &= d_0 - d_1 + d_2 - d_3 + d_4 - d_5, & g_3(x) &= g_0 - g_1 + g_2 - g_3 + g_4 - g_5, \\ d_4(x) &= d_0 - 10d_1 - 11d_2 - d_3 + 10d_4 + 11d_5, & g_4(x) &= g_0 - 10g_1 - 11g_2 - g_3 + 10g_4 + 11g_5, \\ d_5(x) &= d_0 + 11d_1 + 10d_2 - d_3 - 11d_4 - 10d_5, & g_5(x) &= g_0 + 11g_1 + 10g_2 - g_3 - 11g_4 - 10g_5 \end{aligned}$$

в силу  $10^2 \equiv -11, 10 \cdot 11 \equiv -1, 11^2 \equiv 10 \pmod{37}$ .

Таким образом, все многочлены

$$s_k(x) = d_k(x) g_k(x) \quad (k = 0, 1, \dots, 5)$$

являются константами.

Для восстановления многочлена  $s(x)$  используем китайскую теорему об остатках. Воспользуемся таблицей многочленов  $m_k(x), M_k(x), N_k(x)$ :

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$
0	$x - 1$	$x^5 + x^4 + x^3 + x^2 + x + 1$	$6^{-1} \equiv -6 \pmod{37}$
1	$x - 10$	$x^5 + 10x^4 - 11x^3 + x^2 + 10x - 11$	$8^{-1} \equiv 14 \pmod{37}$
2	$x + 11$	$x^5 - 11x^4 + 10x^3 + x^2 - 11x + 10$	$(-14)^{-1} \equiv -8 \pmod{37}$
3	$x + 1$	$x^5 - x^4 + x^3 - x^2 + x - 1$	$(-6)^{-1} \equiv 6 \pmod{37}$
4	$x + 10$	$x^5 - 10x^4 - 11x^3 - x^2 + 10x + 11$	$(-8)^{-1} \equiv -14 \pmod{37}$
5	$x - 11$	$x^5 + 11x^4 + 10x^3 - x^2 - 11x - 10$	$14^{-1} \equiv 8 \pmod{37}$

Тогда

$$\begin{aligned} s(x) &= (-6) s_0(x) (x^5 + x^4 + x^3 + x^2 + x + 1) + 14 s_1(x) (x^5 + 10x^4 - 11x^3 + x^2 + 10x - 11) + \\ &+ (-8) s_2(x) (x^5 - 11x^4 + 10x^3 + x^2 - 11x + 10) + 6 s_3(x) (x^5 - x^4 + x^3 - x^2 + x - 1) + \\ &+ (-14) s_4(x) (x^5 - 10x^4 - 11x^3 - x^2 + 10x + 11) + 8 s_5(x) (x^5 + 11x^4 + 10x^3 - x^2 - 11x - 10). \end{aligned}$$

Уберем множители  $N_k(x)$  ( $k = 0, 1, \dots, 5$ ) в константы, связанные с коэффициентами многочлена  $g(x)$ , т. е. введем обозначения

$$D_i = d_i(x) \quad (i = 0, 1, \dots, 5),$$

$$G_0 = -6g_0(x), G_1(x) = 14g_1(x), G_2 = -8g_2(x), G_3 = 6g_3(x), G_4 = -14g_4(x), G_5 = 8g_5(x),$$

$$S_i = D_i G_i \quad (i = 0, 1, \dots, 5).$$

В новых обозначениях выражение для  $s(x)$  переписывается в виде

$$\begin{aligned} s(x) &= S_0 (x^5 + x^4 + x^3 + x^2 + x + 1) + S_1 (x^5 + 10x^4 - 11x^3 + x^2 + 10x - 11) + \\ &+ S_2 (x^5 - 11x^4 + 10x^3 + x^2 - 11x + 10) + S_3 (x^5 - x^4 + x^3 - x^2 + x - 1) + \\ &+ S_4 (x^5 - 10x^4 - 11x^3 - x^2 + 10x + 11) + S_5 (x^5 + 11x^4 + 10x^3 - x^2 - 11x - 10) = \\ &= (S_0 - 11S_1 + 10S_2 - S_3 + 11S_4 - 10S_5) + (S_0 + 10S_1 - 11S_2 + S_3 + 10S_4 - 11S_5)x + \\ &+ (S_0 + S_1 + S_2 - S_3 - S_4 - S_5)x^2 + (S_0 - 11S_1 + 10S_2 + S_3 - 11S_4 + 10S_5)x^3 + \\ &+ (S_0 + 10S_1 - 11S_2 - S_3 - 10S_4 + 11S_5)x^4 + (S_0 + S_1 + S_2 + S_3 + S_4 + S_5)x^5. \end{aligned}$$

В матрично-векторной форме полученный алгоритм можно записать следующим тождеством:

$$\mathbf{s} = (s_0, s_1, s_2, s_3, s_4, s_5)^T =$$



$$\begin{pmatrix} 1 & -11 & 10 & -1 & 11 & -10 \\ 1 & 10 & -11 & 1 & 10 & -11 \\ 1 & 1 & 1 & -1 & -1 & -1 \\ 1 & -11 & 10 & 1 & -11 & 10 \\ 1 & 10 & -11 & -1 & -10 & 11 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} G_0 & & & & & \\ & G_1 & & & & \\ & & G_2 & & & \\ & & & G_3 & & \\ & & & & G_4 & \\ & & & & & G_5 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 10 & -11 & 1 & 10 & -11 \\ 1 & -11 & 10 & 1 & -11 & 10 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -10 & -11 & -1 & 10 & 11 \\ 1 & 11 & 10 & -1 & -11 & -10 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{pmatrix}. \quad (5)$$

Итак, данный алгоритм содержит 6 операций умножения.

Можно построить алгоритм вычисления этой же свертки, используя преобразование Фурье и теорему о свертке. Для того чтобы существовало соответствующее преобразование Фурье в поле  $\mathbb{Z}_{37}$ , необходимо, чтобы 6 делило число  $p-1 = 37-1 = 36$ . Так как в нашем случае это выполняется, то в  $\mathbb{Z}_{37}$  существует 6-точечное преобразование Фурье. Нам нужен элемент группы  $\mathbb{Z}_{37}^*$  порядка 6. Элемент 2, как нетрудно проверить, является образующей циклической группы  $\mathbb{Z}_{37}^*$ , его порядок равен 36, поэтому порядок элемента  $2^6 \equiv 27 \equiv -10 \pmod{37}$  равен 6.

6-точечное преобразование Фурье вектора  $\mathbf{d} = (d_0, d_1, \dots, d_5)$  задается равенством

$$D_k = \sum_{j=0}^5 (-10)^{jk} d_j \quad (k = 0, 1, \dots, 5),$$

для вектора  $\mathbf{g} = (g_0, g_1, \dots, g_5)$  имеют место аналогичные соотношения. В матричной записи соответственно имеем

$$\begin{pmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \\ D_4 \\ D_5 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -10 & -11 & -1 & 10 & 11 \\ 1 & -11 & 10 & 1 & -11 & 10 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 10 & -11 & 1 & 10 & -11 \\ 1 & 11 & 10 & -1 & -11 & -10 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{pmatrix}. \quad (6)$$

Заметим, что матрица в формуле (6) отличается от матрицы предположений формулы (5) только перестановкой второй и пятой строк.

Компоненты  $G_i$  преобразования Фурье вектора  $\mathbf{g}$  получаются также с помощью матрицы формулы (6). Затем производим умножения

$$S_k = D_k G_k \quad (k = 0, 1, \dots, 5).$$

Очевидно, всего имеется 6 умножений.

Для выполнения обратного преобразования Фурье нужны элементы  $\omega^{-1} = (-10)^{-1}$  и  $n^{-1} = 6^{-1}$  из  $\mathbb{Z}_{37}$ . Так как

$$(-10)^{-1} \equiv 11, \quad 6^{-1} \equiv -6 \pmod{37},$$

то обратное преобразование Фурье задается равенством

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{pmatrix} = (-6) \cdot \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 11 & 10 & -1 & -11 & -10 \\ 1 & 10 & -11 & 1 & 10 & -11 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -11 & 10 & 1 & -11 & 10 \\ 1 & -10 & -11 & -1 & 10 & 11 \end{pmatrix} \begin{pmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \\ S_4 \\ S_5 \end{pmatrix}.$$

Отметим, что если входные компоненты большие, то некоторые выходные компоненты свертки окажутся больше 36. В этом случае вычисление в поле  $\mathbb{Z}_{37}$  приведет к переполнению выходных компонент и даст неправильный результат. Тогда нужно использовать большее поле; в практических приложениях полезны поля, содержащие по меньшей мере  $2^{16}$  элементов.

Если входные компоненты очень большие для поля, выбранного для вычислений, то можно воспользоваться китайской теоремой об остатках и работать с вычетами целых чисел. Китайская теорема в этом случае используется для разбиения самих чисел, не затрагивая индексацию данных, которые следуют в своем естественном порядке. От вычисления

$$s(x) = d(x) g(x)$$



мы переходим к нескольким сверткам

$$s^{(l)}(x) = d^{(l)}(x) g^{(l)}(x) \pmod{m_l}, \quad (7)$$

$l = 1, 2, \dots, r$ ; где  $m_1, m_2, \dots, m_r$  – попарно взаимно простые целые числа,

$$d^{(l)}(x) = \sum_{i=0}^{N-1} d_i^{(l)} x^i, \quad g^{(l)}(x) = \sum_{i=0}^{L-1} g_i^{(l)} x^i,$$

$$d_i^{(l)} \equiv d_i \pmod{m_l}, \quad g_i^{(l)} \equiv g_i \pmod{m_l}.$$

По китайской теореме об остатках коэффициенты  $s_i$  многочлена  $s(x)$  восстанавливаются по вычислениям  $s_i^{(l)} \pmod{m_l}$  ( $l = 1, 2, \dots, r$ ). Таким образом, вычисление свертки, содержащей большие целые числа, сводится к вычислению  $r$  сверток, содержащих малые целые числа, и последующему решению систем сравнений. Для вычисления сверток (7) можно пользоваться быстрыми алгоритмами вычисления сверток в кольце  $\mathbb{Z}$  или строить новые для соответствующего кольца  $\mathbb{Z}_{m_l}$ , пользуясь уже известными алгоритмами.

## §22. ПРЕОБРАЗОВАНИЯ ФУРЬЕ В ПОЛЯХ $GF(2^m + 1)$ (ПРЕОБРАЗОВАНИЯ ФЕРМА)

Простейший вид алгоритмы вычисления преобразования Фурье имеют в полях Галуа вида  $GF(2^m + 1)$ , где  $p = 2^m + 1$  есть простое число.

Число  $2^m + 1$  не является простым, если  $m$  не равно степени числа 2. Это нетрудно проверить. Действительно, пусть  $m = pq$ , где  $p, q$  – нечетные числа, тогда

$$2^m + 1 = 2^{pq} + 1 = (2^p + 1)(2^{p(q-1)} - 2^{p(q-2)} + 2^{p(q-3)} - \dots - 2^p + 1)$$

– составное число;

пусть  $m = 2p$ , где  $p$  – нечетное число, тогда

$$2^m + 1 = 2^{2p} + 1 = (2^p - 2^{\frac{p+1}{2}} + 1)(2^p + 2^{\frac{p+1}{2}} + 1)$$

– составное число;

пусть  $m = 2^k p$ , где  $p$  – нечетное число, тогда

$$2^m + 1 = 2^{2^k p} + 1 = (2^{2^k} + 1)(2^{2^k(p-1)} - 2^{2^k(p-2)} + 2^{2^k(p-3)} - \dots - 2^{2^k} + 1)$$

– составное число.

Но и в случае, когда  $m = 2^n$  число  $2^m + 1 = 2^{2^n} + 1$  не всегда является простым. При  $n = 1, 2, 3, 4$  мы получим числа

$$5, 17, 257, 65537,$$

которые являются простыми. Но при  $n = 5$  получаем число

$$2^{2^5} + 1 = (2^{16})^2 + 1 = (65537)^2 + 1 = 4\,294\,967\,297,$$

которое не является простым, так как

$$4\,294\,967\,297 = 641 \times 6\,700\,417.$$

Числа вида  $2^{2^n} + 1$  известны под названием *чисел Ферма*. Простые числа вида  $2^{2^n} + 1$  называют *простыми числами Ферма*. Если  $q$  – простое число Ферма, то в поле  $GF(q)$  число  $q - 1$  и любой его делитель равны некоторой степени двойки. Преобразование Фурье

$$V_k = \sum_{j=0}^{n-1} \omega^{jk} v_j, \quad k = 0, 1, \dots, n-1,$$

определено в поле  $GF(2^m + 1)$ , если  $n$  делит  $2^m$  и в  $GF(2^m + 1)$  существует элемент  $\omega$  порядка  $n$ . Такое преобразование Фурье называют *числовым преобразованием Ферма*. Например, в поле  $GF(2^{16} + 1)$  определены преобразования Фурье длин  $2^{16}, 2^{15}, 2^{14}, \dots, 2^2, 2$ . Используя алгоритм Кули – Тьюки по основанию два, преобразование Фурье в поле  $GF(2^m + 1)$  можно представить в виде последовательности преобразований по основанию два. Преобразования длины 32 и меньше в поле  $GF(2^{16} + 1)$  делаются на самом деле намного проще. Дело в том, что порядок элемента 2 в этом поле равен 32. Действительно,

$$2^{16} + 1 \equiv 0 \pmod{2^{16} + 1},$$

откуда

$$2^{16} \equiv -1, \quad 2^{32} \equiv 1 \pmod{2^{16} + 1}.$$

Поэтому 32-точечное преобразование Фурье записывается в виде

$$V_k = \sum_{j=0}^{31} 2^{jk} v_j, \quad k = 0, 1, \dots, 31,$$



и умножение в данной арифметике сводится просто к циклическому сдвигу, так как представляет собой умножение на степень двойки. Такое преобразование не содержит умножений и вычисляется совсем просто, но для практических приложений длина 32 слишком мала.

В общем случае простого числа Ферма  $2^m + 1$  порядок элемента 2 в поле  $GF(2^m + 1)$  равен  $2m$ , так что 2 может быть использована в качестве  $\omega$  – корня из единицы степени  $2m$ . Для построения преобразования Фурье длины  $4m$  можно воспользоваться  $\omega = \sqrt{2} \pmod{2^m + 1}$ , где

$$\sqrt{2} = 2^{\frac{m}{4}} (2^{\frac{m}{2}} - 1).$$

Действительно,

$$\begin{aligned} 2 &= (\sqrt{2})^2 = (2^{\frac{m}{4}} (2^{\frac{m}{2}} - 1))^2 = 2^{\frac{m}{2}} (2^m - 2^{\frac{m}{2}+1} + 1) = 2^m 2^{\frac{m}{2}} - 2^{\frac{m}{2}+\frac{m}{2}+1} + 2^{\frac{m}{2}} \equiv \\ &\equiv -2^{\frac{m}{2}} - 2^{m+1} + 2^{\frac{m}{2}} \equiv 2 \pmod{2^m + 1}, \end{aligned}$$

так как  $2^m \equiv -1 \pmod{2^m + 1}$ .

Значит, каждая степень элемента  $\sqrt{2}$  имеет вид  $2^a \pm 2^b$ , поэтому умножение на  $(\sqrt{2})^{kj}$  может быть реализовано как пара циклических сдвигов и сложение (или вычитание).

Например, преобразование Фурье длины 64 в поле  $GF(2^{16} + 1)$  записывается в виде

$$V_k = \sum_{j=0}^{63} (2^{12} - 2^4)^{jk} v_j, \quad k = 0, 1, \dots, 63. \quad (1)$$

Если для этого вычисления воспользоваться БПФ-алгоритмом Кули – Тьюки по основанию два, то все умножения на константы будут умножениями на числа вида  $2^a \pm 2^b$  и, следовательно, могут быть реализованы как пара циклических сдвигов и сложение. В общем случае преобразование Фурье в поле  $GF(2^m + 1)$  длины большей чем  $2m$ , всегда содержит нетривиальные умножения. Число этих умножений можно уменьшить сведением БПФ-алгоритма Кули – Тьюки в форму многомерного преобразования, в каждом из измерений которого используется не более, чем  $2m$ -точечное преобразование.

Например, рассмотрим 64-точечное преобразование Фурье в поле  $GF(2^8 + 1)$ :

$$V_k = \sum_{j=0}^{63} \omega^{jk} v_j, \quad k = 0, 1, \dots, 63,$$

где  $\omega$  – элемент порядка 64, который можно полагать равным  $\pi^4$ , где  $\pi$  – примитивный элемент поля  $GF(2^8 + 1)$ , т. е.  $\pi^{256} \equiv 1 \pmod{257}$ . В качестве примитивного элемента  $\pi$  можно взять, например, элемент 3. Однако этот выбор не является удачным, так как  $3^4 = 81$ . Мы хотим выбрать  $\omega$  так, чтобы выполнялось равенство  $\omega^8 = 2$ , значит,  $\pi$  надо выбрать так, чтобы  $\pi^{32} = 2$ . Так как элемент 2 имеет порядок 16 в поле  $GF(2^8 + 1)$ , то элемент

$$\sqrt{2} \equiv 2^2 (2^4 - 1) = 2^6 - 2^2 \equiv 60 \pmod{257}$$

является элементом порядка 32, а  $\sqrt{60} \pmod{257}$  – элементом порядка 64. В силу

$$\sqrt{60} \equiv 46 \pmod{257}$$

$(46^2 = 2116 \equiv 60 \pmod{257})$  получаем  $\omega = 46$ . С другой стороны,

$$46 \equiv 3^{76} \pmod{257},$$

значит, в качестве примитивного элемента лучше выбрать  $\pi = 3^{19} \equiv 41 \pmod{257}$ . Тогда БПФ-алгоритм Кули – Тьюки приводит преобразование (1) к виду

$$V_{8k'+k''} = \sum_{j'=0}^7 2^{j'k'} \{ \omega^{j'k''} \sum_{j''=0}^7 2^{j''k''} v_{j'+8j''} \}, \quad k' = 0, 1, \dots, 7; \quad k'' = 0, 1, \dots, 7.$$



Внешнее и внутреннее преобразования Фурье являются 8-точечными, каждое из них в свою очередь можно разбить на два по алгоритму Кули – Тьюки по основанию два и вычислить с помощью только циклических сдвигов и сложений. Умножения на  $\omega^{j'k''}$  представляют собой нетривиальные умножения, но таких умножений имеется всего 64, и все они являются целочисленными. Кроме того, при  $j'$  или  $k''$  равном нулю, эти умножения тривиальны, поэтому остается только 49 нетривиальных умножений.

Арифметикой поля  $GF(2^8+1)$  являются обычные целочисленные сложение и умножение с последующим приведением результата по модулю  $2^8+1$ . Посмотрим, что представляет собой умножение на степень двойки с точки зрения такой арифметики для чисел, представленных в двоичной системе счисления. Так как  $2^8 \equiv -1 \pmod{2^8+1}$ , то число  $N = 2^8$  можно заменить на  $-1$ , поэтому рассмотрим число  $N < 2^8$ . Пусть

$$N = \sum_{i=0}^7 2^i a_i, \quad a_i = \begin{cases} 0, \\ 1. \end{cases}$$

Найдем число  $N \cdot 2^l$ ,  $l \leq 7$  (иначе,  $2^l \equiv 2^{l-8r}(-1)^r \pmod{2^8+1}$ , где  $0 \leq l-8r < 7$ ):

$$\begin{aligned} N \cdot 2^l \pmod{2^8+1} &= \sum_{i=0}^7 2^{i+l} a_i \pmod{2^8+1} = \sum_{0 \leq i+l \leq 7} 2^{i+l} a_i + \sum_{i+l > 7}^{7+l} 2^8 2^{i+l-8} a_i \pmod{2^8+1} \equiv \\ &\sum_{0 \leq i+l \leq 7} 2^{i+l} a_i - \sum_{i=8-l}^7 2^{i+l-8} a_i = \sum_{j=0}^7 2^j a_{j-l} - \sum_{j=0}^{l-1} 2^j a_{8+j-l}. \end{aligned}$$

Таким образом, получаем

$$N \cdot 2^l \pmod{2^8+1} = \begin{cases} a_{7-l} a_{6-l} \dots a_0 \overbrace{00 \dots 0}^l 0_{(2)} \\ - \\ 00 \dots \dots a_7 \dots \dots a_{9-l} a_{8-l(2)}, \end{cases}$$

т. е. сначала происходит сдвиг двоичных битов числа  $N$  влево на  $l$  позиций, затем биты переполнения (порядка больше 8) переводятся в младшие разряды и вычитаются. Это и есть циклический сдвиг. Аналогично выполняется операция приведения по модулю  $2^8+1$  при выполнении операций сложения и умножения, если результат превышает данный модуль.



### §23. ПРЕОБРАЗОВАНИЯ ФУРЬЕ В ПОЛЯХ $GF(2^m - 1)$ (ПРЕОБРАЗОВАНИЯ МЕРСЕННА)

Полями Галуа, в которых умножение выглядит наиболее естественным образом, являются поля вида  $GF(2^m - 1)$ . Чтобы  $GF(2^m - 1)$  было полем, требуется, чтобы  $2^m - 1$  было простым числом. Если  $m$  — число составное, пусть  $m = st$ , то

$$2^m - 1 = 2^{st} - 1 = (2^s - 1)(2^{s(t-1)} + 2^{s(t-2)} + \dots + 2^s + 1),$$

т. е.  $2^m - 1$  является также составным числом. Значит,  $m$  должно быть простым числом. Но не для всякого простого  $m$  число  $2^m - 1$  является простым. Простые числа вида  $2^m - 1$  называются *простыми числами Мерсенна* в честь Марена Мерсенна, который первым сделал предположение о том, при каких значениях  $m \leq 257$  число вида  $2^m - 1$  будет простым. Наименьшие значения  $m$  равны 2, 3, 5, 7, 13, 17, 19 и 31; соответствующие им простые числа Мерсенна равны

$$3, 7, 31, 127, 8191, 131071, 524287 \text{ и } 2147483647.$$

В настоящее время известно 39 простых чисел Мерсенна, которые можно найти в специальных таблицах (см. [2]).

Арифметика поля  $GF(2^m - 1)$  хорошо согласуется с представлением целых чисел в виде двоичных  $m$ -битовых чисел, так как в этом поле  $2^m = 1$ . Следовательно, арифметикой поля  $GF(2^m - 1)$  является обычная целочисленная арифметика, в которой биты переполнения переносятся в соответствующие низшие разряды.

В поле Галуа  $GF(q)$  преобразование Фурье существует для всех длин  $n$ , которые делят число  $q - 1$ . Следовательно, в поле  $GF(2^m - 1)$  преобразование Фурье существует для всех длин  $n$ , которые делят число  $2^m - 2$ . Эти преобразования называют *числовыми преобразованиями Мерсенна*.

Так как число  $2^m - 2$  не равно степени двойки, то числовые преобразования Мерсенна нельзя вычислять с помощью БПФ-алгоритма Кули — Тьюки по основанию два. Этим они существенно отличаются от числовых преобразований Ферма. Для вычисления числового преобразования Мерсенна можно воспользоваться любым БПФ-алгоритмом по смешанному основанию.

Например, в поле  $GF(2^{13} - 1)$  существует преобразование Фурье длины  $n$ , если  $n$  делит  $2^{13} - 2$ . В данном случае выбор возможной длины  $n$  дается разложением  $2^{13} - 2 = 2 \cdot 5 \cdot 7 \cdot 9 \cdot 13$ . В качестве корня из единицы степени 26 можно рассматривать элемент  $-2$ , так как из  $2^{13} \equiv 1 \pmod{2^{13} - 1}$  следует  $(-2)^{26} \equiv 1 \pmod{2^{13} - 1}$ , т. е. порядок элемента  $-2$  равен 26, поэтому существует 26-точечное преобразование Фурье в поле  $GF(2^{13} - 1)$ , которое записывается в виде

$$V_k = \sum_{j=0}^{25} (-2)^{jk} v_j, \quad k = 0, 1, \dots, 25.$$

В этом преобразовании Фурье все умножения исчерпываются умножениями на степень двойки, и, следовательно, вычисления можно реализовать одними циклическими сдвигами.

Для построения преобразований на других длинах приходится допустить наличие умножений. Алгоритм на большой длине строится обычно из алгоритмов малых длин. Существуют специальные методы перехода от преобразования большой длины к нескольким преобразованиям малой длины. Алгоритмы же малых длин строятся так же, как для любого конечного поля.

В качестве примера рассмотрим 5-точечный и 6-точечный БПФ-алгоритмы в поле Галуа  $GF(31) = GF(2^5 - 1)$ . Для построения алгоритма 5-точечного преобразования Фурье нам нужен элемент порядка 5, в качестве такого элемента можно взять 2. Действительно,

$$2^1 = 2, \quad 2^2 = 4, \quad 2^3 = 8, \quad 2^4 = 16, \quad 2^5 = 1 \pmod{31}.$$

Значит, 5-точечное преобразование Фурье можно записать в виде

$$V_k = \sum_{j=0}^4 2^{jk} v_j, \quad k = 0, 1, \dots, 4,$$



откуда следует, что для вычисления этого преобразования умножения не нужны, достаточно сдвигов и сложений.

Для построения 6-точечного преобразования Фурье нужен элемент порядка 6. Примитивным элементом поля  $GF(31)$  является 3, т. е. порядок элемента 3 равен 30, тогда порядок элемента  $3^5 \equiv 26 \pmod{31}$  равен 6. Возможно, это не лучший выбор элемента порядка 6, в качестве такого элемента можно, к примеру, выбрать 6, так как

$$6^1 = 6, \quad 6^2 = 5, \quad 6^3 = 30, \quad 6^4 = 25, \quad 6^5 = 26, \quad 6^6 = 1 \pmod{31}.$$

Возьмем  $\omega = 6$ , тогда 6-точечное преобразование Фурье записывается в виде

$$V_k = \sum_{j=0}^5 6^{jk} v_j, \quad k = 0, 1, \dots, 5.$$

Воспользуемся алгоритмом Кули – Тьюки, переходя к двумерному преобразованию ( $n'n'' = 2 \cdot 3$ ):

$$V_{3k'+k''} = \sum_{j'=0}^1 (-1)^{j'k'} \{ 6^{j'k''} \sum_{j''=0}^2 5^{jj''k''} v_{j'+2j''} \}, \quad k' = 0, 1; \quad k'' = 0, 1, 2.$$

Здесь  $\beta = \omega^3 = 6^3 \equiv -1 \pmod{31}$ ,  $\gamma = \omega^2 = 6^2 \equiv 5 \pmod{31}$ . Расписывая внешнюю сумму (по  $j'$ ), получаем

$$V_{3k'+k''} = \sum_{j=0}^2 5^{jk''} v_{2j} + (-1)^{k'} 6^{k'} \sum_{j=0}^2 5^{jk''} v_{2j+1}, \quad k' = 0, 1; \quad k'' = 0, 1, 2,$$

откуда при  $k' = 0$

$$V_k = \sum_{j=0}^2 5^{jk} v_{2j} + 6^k \sum_{j=0}^2 5^{jk} v_{2j+1}, \quad k = 0, 1, 2,$$

при  $k' = 1$  –

$$V_{k+3} = \sum_{j=0}^2 5^{jk} v_{2j} - 6^k \sum_{j=0}^2 5^{jk} v_{2j+1}, \quad k = 0, 1, 2.$$

Таким образом, вычисление сводится к выполнению двух 3-точечных преобразований Фурье, двум дополнительным умножениям и 6 дополнительным сложениям:

$$\begin{pmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \end{pmatrix} = \begin{pmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ 1 & & & -1 & \\ & 1 & & & -1 \\ & & 1 & & -1 \end{pmatrix} \begin{pmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & 6 \\ & & & & & 5 \end{pmatrix} \begin{pmatrix} V_{01} \\ V_{11} \\ V_{21} \\ V_{02} \\ V_{12} \\ V_{22} \end{pmatrix},$$

где

$$V_{k1} = \sum_{j=0}^2 5^{jk} v_{2j} \quad (k = 0, 1, 2) \quad \text{и} \quad V_{k2} = \sum_{j=0}^2 5^{jk} v_{2j+1} \quad (k = 0, 1, 2)$$

являются компонентами 3-точечных преобразований Фурье соответственно векторов  $\mathbf{v}_1 = (v_0, v_2, v_4)$  и  $\mathbf{v}_2 = (v_1, v_3, v_5)$ .

3-точечное преобразование Фурье вектора  $\mathbf{v}_1$  в матричном виде записывается уравнением

$$\begin{pmatrix} V_{01} \\ V_{11} \\ V_{21} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 5 & 5^2 \\ 1 & 5^2 & 5^4 \end{pmatrix} \begin{pmatrix} v_0 \\ v_2 \\ v_4 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 5 & -6 \\ 1 & -6 & 5 \end{pmatrix} \begin{pmatrix} v_0 \\ v_2 \\ v_4 \end{pmatrix}$$



и может быть вычислено следующим образом:

$$V_{01} = v_0 + v_2 + v_4,$$

$$\begin{pmatrix} V_{11} - V_{01} \\ V_{21} - V_{01} \end{pmatrix} = \begin{pmatrix} 4 & -7 \\ -7 & 4 \end{pmatrix} \begin{pmatrix} v_2 \\ v_4 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} (-3)2^{-1} & 0 \\ 0 & (11)2^{-1} \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} v_2 \\ v_4 \end{pmatrix} = \\ \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 14 & 0 \\ 0 & -10 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} v_2 \\ v_4 \end{pmatrix},$$

так как  $2^{-1} \equiv 16 \pmod{31}$ ,  $-3 \cdot 16 \equiv 14 \pmod{31}$ ,  $11 \cdot 16 \equiv -10 \pmod{31}$ . Значит, требуется две операции умножения. Аналогично вычисляются компоненты  $V_{k2}$  ( $k = 0, 1, 2$ ). Таким образом, всего алгоритм требует 6 нетривиальных умножений в поле  $GF(31)$ , сложений всего потребуется 22 (для каждого 3-точечного преобразования Фурье по 8 сложений и 6 дополнительных для вычисления  $V_k$ ). При желании мы можем умножения на  $-10$  и  $14$  заменить каждое парой сдвигов и сложением, исходя из того, что

$$14 = 2^4 - 2, \quad -10 = -(2^3 + 2),$$

тогда число нетривиальных умножений уменьшится до 2.

В качестве второго примера рассмотрим поле  $GF(2^{17} - 1)$ , элементы которого записаны как 17-битовые двоичные числа. Линейная свертка последовательностей 16-битовых целых чисел может быть вычислена как свертка в поле  $GF(2^{17} - 1)$ . Для вычисления свертки можно воспользоваться БПФ-алгоритмами в поле  $GF(2^{17} - 1)$  и теоремой о свертке.

Длины преобразования Фурье в поле  $GF(2^{17} - 1)$  исчерпываются делителями числа  $2^{17} - 2$  и полностью определяются разложением  $2^{17} - 2 = 2 \cdot 3 \cdot 5 \cdot 17 \cdot 257$ . Можно выбрать, например,  $n = 510$ , и строить 510-точечное БПФ-преобразование из 2-точечного, 3-точечного, 5-точечного и 17-точечного БПФ-алгоритмов. Модули для 2-точечного, 3-точечного и 5-точечного преобразований очень просты и даются малыми БПФ-алгоритмами Винограда. Хотя эти малые алгоритмы рассматриваются над полем  $GF(2^{17} - 1)$ , записываются они точно так же, как и в случае поля комплексных чисел.

17-точечное преобразование тоже представляет собой простой модуль, но строится он другим способом. Элемент 2 поля  $GF(2^{17} - 1)$  имеет порядок 17, так как  $2^{17} \equiv 1 \pmod{2^{17} - 1}$ . Следовательно, 17-точечное преобразование Фурье в этом поле задается равенствами

$$V_k = \sum_{j=0}^{16} 2^{jk} v_j, \quad k = 0, 1, \dots, 16,$$

и может быть вычислено с помощью одних циклических сдвигов и сложений.

Если требуется вычислить свертку, длина которой больше 510, то надо использовать также множитель 257. 257-точечное преобразование Фурье с помощью алгоритма Рейдера сводится к 256-точечной циклической свертке, которая затем вычисляется с помощью БПФ-алгоритма Кули - Тьюки по основанию два и алгоритма свертки.



## §24. АЛГОРИТМ АГАРВАЛА – КУЛИ ВЫЧИСЛЕНИЯ СВЕРТКИ

Алгоритм Агарвала – Кули представляет собой метод преобразования  $n'n''$ -точечной одномерной циклической свертки в двумерную циклическую  $(n' \times n'')$ -свертку при условии, что числа  $n'$  и  $n''$  взаимно просты. Его используют для того, чтобы скомбинировать алгоритм Винограда вычисления  $n'$ -точечной циклической свертки, содержащий  $M(n')$  умножений, с алгоритмом Винограда вычисления  $n''$ -точечной циклической свертки, содержащим  $M(n'')$  умножений, и построить алгоритм вычисления  $n'n''$ -точечной циклической свертки, содержащий  $M(n')M(n'')$  умножений.

В основе алгоритма Агарвала – Кули лежит китайская теорема об остатках для целых чисел. Он не обеспечивает такого же большого уменьшения числа умножений, как алгоритм Винограда вычисления свертки, но его преимущество в том, что он не имеет тенденции к чрезмерному росту числа сложений с увеличением длины преобразования, как это происходит в алгоритме Винограда. Кроме того, этот алгоритм лучше структурирован. Если  $n$  – длина свертки – является большим числом, то алгоритм Агарвала – Кули может быть разбит на составные части (подпрограммы). Для построения хороших алгоритмов свертки используют алгоритм Агарвала – Кули разбиения длинной свертки на короткие циклические свертки, которые затем вычисляют, пользуясь алгоритмом Винограда вычисления коротких циклических сверток.

Алгоритм Агарвала – Кули основан на трех идеях. Сначала одномерная циклическая свертка с помощью китайской теоремы об остатках (К.Т.О.) для целых чисел преобразуется в многомерную циклическую свертку; затем вдоль каждой координаты используется алгоритм Винограда вычисления короткой свертки; и, наконец, используется теорема о кронекеровском произведении матриц для того, чтобы собрать вместе все полученные на предыдущем шаге преобразования.

Сначала дадим определение кронекеровского произведения.

**Определение.** Пусть  $A = (a_{ij})$  и  $B = (b_{ij})$  – матрицы соответственно размеров  $h \times k$  и  $m \times l$ . Тогда *кронекеровским произведением* матриц  $A$  и  $B$ , обозначаемым  $A \times B$ , называется матрица  $C$ , содержащая  $hm$  строк и  $kl$  столбцов, у которой на пересечении строки с номером  $(i-1)m + j$  и столбца с номером  $(t-1)l + s$  стоит элемент

$$c_{ij,ts} = a_{it} b_{js}.$$

Кронекеровское произведение  $A \times B$  представляет собой  $(h \times k)$ -таблицу, состоящую из  $(m \times l)$ -блоков,  $(i, k)$ -й из которых равен  $a_{ik} B$ .

Справедливо следующее утверждение.

**Теорема.** Если все матричные произведения определены, то кронекеровское произведение удовлетворяет равенству

$$(A \times B)(C \times D) = (AC) \times (BD).$$

Рассмотрим переход от одномерной циклической свертки к двумерной.

Пусть  $\mathbf{d} = \{d_i \mid i = 0, 1, \dots, n-1\}$  и  $\mathbf{g} = \{g_i \mid i = 0, 1, \dots, n-1\}$  – две последовательности длины  $n = n'n''$ , где  $\text{НОД}(n', n'') = 1$ , а  $\mathbf{s} = \{s_i \mid i = 0, 1, \dots, n-1\}$  – их циклическая свертка, т. е.

$$s_i = \sum_{k=0}^{n-1} d_k g_{((i-k))} \quad (i = 0, 1, \dots, n-1), \quad (1)$$

где  $((i-k)) \equiv i-k \pmod{n}$ .

Заменим индексы  $i$  и  $k$  парами индексов  $(i', i'')$  и  $(k', k'')$  по правилу

$$\begin{aligned} i' &\equiv i \pmod{n'}, & i'' &\equiv i \pmod{n''}, \\ k' &\equiv k \pmod{n'}, & k'' &\equiv k \pmod{n''}. \end{aligned}$$

В силу китайской теоремы об остатках индексы  $i$  и  $k$  можно восстановить по парам индексов  $(i', i'')$  и  $(k', k'')$  по формулам:

$$\begin{aligned} i &= N''n''i' + N'n'i'' \pmod{n}, & i' &= 0, 1, \dots, n'-1; & i'' &= 0, 1, \dots, n''-1, \\ k &= N''n''k' + N'n'k'' \pmod{n}, & k' &= 0, 1, \dots, n'-1; & k'' &= 0, 1, \dots, n''-1, \end{aligned} \quad (2)$$



где целые числа  $N'$  и  $N''$  определяются равенством

$$N'n' + N''n'' = 1.$$

Свертку (1) можно записать в виде

$$s_{N''n''i' + N'n'i''} = \sum_{k'=0}^{n'-1} \sum_{k''=0}^{n''-1} g((N''n''(i'-k') + N'n'(i''-k''))) d_{N''n''k' + N'n'k''}.$$

(Очевидно, из (2) следует, что  $i - k = N''n''(i' - k') + N'n'(i'' - k'')$ .)

Определим двумерные переменные, задавая их равенствами

$$d_{k',k''} = d_{N''n''k' + N'n'k''}, \quad g_{k',k''} = g_{N''n''k' + N'n'k''}, \quad s_{k',k''} = s_{N''n''k' + N'n'k''},$$

( $k' = 0, 1, \dots, n' - 1$ ;  $k'' = 0, 1, \dots, n'' - 1$ ).

Введение двумерных переменных означает, что одномерные векторы  $\mathbf{d}$ ,  $\mathbf{g}$ , и  $\mathbf{s}$  заменены двумерными таблицами  $d$ ,  $g$  и  $s$ . В каждой такой таблице компоненты вектора записываются на "расширенную диагональ" матрицы размера  $n' \times n''$ , т. е.

$$\begin{aligned} (d_0, d_1, \dots, d_{n'n''-1}) &\rightarrow d = \begin{pmatrix} d_{0,0} & d_{0,1} & \dots & d_{0,n''-2} & d_{0,n''-1} \\ d_{1,0} & d_{1,1} & \dots & d_{1,n''-2} & d_{1,n''-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ d_{n'-1,0} & d_{n'-1,1} & \dots & d_{n'-1,n''-2} & d_{n'-1,n''-1} \end{pmatrix}, \\ (g_0, g_1, \dots, g_{n'n''-1}) &\rightarrow g = \begin{pmatrix} g_{0,0} & g_{0,1} & \dots & g_{0,n''-2} & g_{0,n''-1} \\ g_{1,0} & g_{1,1} & \dots & g_{1,n''-2} & g_{1,n''-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ g_{n'-1,0} & g_{n'-1,1} & \dots & g_{n'-1,n''-2} & g_{n'-1,n''-1} \end{pmatrix}, \\ (s_0, s_1, \dots, s_{n'n''-1}) &\rightarrow s = \begin{pmatrix} s_{0,0} & s_{0,1} & \dots & s_{0,n''-2} & s_{0,n''-1} \\ s_{1,0} & s_{1,1} & \dots & s_{1,n''-2} & s_{1,n''-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ s_{n'-1,0} & s_{n'-1,1} & \dots & s_{n'-1,n''-2} & s_{n'-1,n''-1} \end{pmatrix}. \end{aligned}$$

Столбцы таблиц  $d$ ,  $g$  и  $s$  будем обозначать соответственно  $\mathbf{d}^{(k)}$ ,  $\mathbf{g}^{(k)}$  и  $\mathbf{s}^{(k)}$  ( $k = 0, 1, \dots, n'' - 1$ ). Через  $\bar{\mathbf{d}}$ ,  $\bar{\mathbf{g}}$ ,  $\bar{\mathbf{s}}$  обозначим векторы, получающиеся, если столбцы соответствующей таблицы выписать в стек и образовать одномерный массив.

В новых обозначениях свертка записывается в виде

$$s_{i',i''} = \sum_{k'=0}^{n'-1} \sum_{k''=0}^{n''-1} g((i'-k'),((i''-k''))) d_{k',k''} \quad (3)$$

где

$$\begin{aligned} ((i' - k')) &\equiv N''n''(i' - k') + N'n'(i'' - k'') \pmod{n'}, \\ ((i'' - k'')) &\equiv N''n''(i' - k') + N'n'(i'' - k'') \pmod{n''}. \end{aligned}$$

Очевидно, формулы (3) задают двумерную циклическую свертку. Заметим, что пока мы не получили никакого уменьшения сложности вычислений. Полное число умножений по-прежнему равно  $(n'n'')^2 = n^2$ . Действительно, можно переписать (3) в виде

$$s_{i',i''} = \sum_{k'=0}^{n'-1} \left[ \sum_{k''=0}^{n''-1} g((i'-k'),((i''-k''))) d_{k',k''} \right] \quad (i' = 0, 1, \dots, n' - 1; \quad i'' = 0, 1, \dots, n'' - 1), \quad (4)$$

где внутренняя сумма представляет собой  $n''$ -точечную циклическую свертку, которая должна быть вычислена для всех  $i'$  и  $k'$ . Прямое вычисление  $n''$ -точечной циклической свертки требует  $n''^2$  умножений. Эту свертку придется вычислять для каждого  $i'$ , принимающего  $n'$  значений, и каждого  $k'$ , принимающего  $n'$  значений, т. е.  $n'^2$  раз. Таким образом, прямое вычисление требует  $n'^2 n''^2$  умножений.

Понять двумерную свертку легче, если выписать ее в виде свертки многочленов. Каждый из векторов  $\mathbf{d}$ ,  $\mathbf{g}$  и  $\mathbf{s}$  отобразился в двумерную таблицу. Каждую такую таблицу можно записать как вектор многочленов

$$d_{k'}(y) = \sum_{k''=0}^{n''-1} d_{k',k''} y^{k''}, \quad g_{k'}(y) = \sum_{k''=0}^{n''-1} g_{k',k''} y^{k''}, \quad s_{k'}(y) = \sum_{k''=0}^{n''-1} s_{k',k''} y^{k''} \quad (k' = 0, 1, \dots, n' - 1) \quad (5)$$

или как многочлен от двух переменных

$$d(x, y) = \sum_{k'=0}^{n'-1} \sum_{k''=0}^{n''-1} d_{k',k''} y^{k''} x^{k'}, \quad g(x, y) = \sum_{k'=0}^{n'-1} \sum_{k''=0}^{n''-1} g_{k',k''} y^{k''} x^{k'},$$

$$s(x, y) = \sum_{k'=0}^{n'-1} \sum_{k''=0}^{n''-1} s_{k',k''} y^{k''} x^{k'}.$$

Тогда формулу (4) можно записать в виде свертки многочленов

$$s_{i'}(y) = \sum_{k'=0}^{n'-1} g_{((i'-k'))}(y) d_{k'}(y), \quad i' = 0, 1, \dots, n' - 1. \quad (6)$$

Рассмотренные нами в § 3 – 6 алгоритмы вычисления сверток строились над полями, но на самом деле они выполнимы и в некоторых кольцах, в частности, в кольце многочленов над полем. Следовательно, свертка многочленов может быть вычислена одним из этих быстрых алгоритмов. Сложение в этих алгоритмах становится сложением многочленов, а умножение – умножением многочленов. Последние умножения, в свою очередь, являются свертками и могут быть вычислены с помощью быстрых алгоритмов свертки. Таким образом, получается способ вычисления двумерной свертки, состоящий во включении одного быстрого алгоритма одномерной свертки в другой быстрый алгоритм одномерной свертки.

Распишем в (4) внутреннюю сумму:

$$s_{i',i''} = \sum_{k'=0}^{n'-1} g_{((i'-k')),((i''))} d_{k',0} + \sum_{k'=0}^{n'-1} g_{((i'-k')),((i''-1))} d_{k',1} + \dots + \sum_{k'=0}^{n'-1} g_{((i'-k')),((i''-n''+1))} d_{k',n''-1} \quad (7)$$

( $i' = 0, 1, \dots, n' - 1$ ;  $i'' = 0, 1, \dots, n'' - 1$ ).

Полагая  $i'' = 0, 1, \dots, n'' - 1$ , получаем

$$s_{i',0} = \sum_{k'=0}^{n'-1} g_{((i'-k')),0} d_{k',0} + \sum_{k'=0}^{n'-1} g_{((i'-k')),((n''-1))} d_{k',1} + \sum_{k'=0}^{n'-1} g_{((i'-k')),((n''-2))} d_{k',2} + \dots$$

$$+ \sum_{k'=0}^{n'-1} g_{((i'-k')),1} d_{k',n''-1},$$

$$s_{i',1} = \sum_{k'=0}^{n'-1} g_{((i'-k')),1} d_{k',0} + \sum_{k'=0}^{n'-1} g_{((i'-k')),0} d_{k',1} + \sum_{k'=0}^{n'-1} g_{((i'-k')),((n''-1))} d_{k',2} + \dots$$

$$+ \sum_{k'=0}^{n'-1} g_{((i'-k')),2} d_{k',n''-1},$$



$$\begin{aligned}
s_{i',2} &= \sum_{k'=0}^{n'-1} g((i'-k'),2) d_{k',0} + \sum_{k'=0}^{n'-1} g((i'-k'),1) d_{k',1} + \sum_{k'=0}^{n'-1} g((i'-k'),0) d_{k',2} + \dots \\
&+ \sum_{k'=0}^{n'-1} g((i'-k'),3) d_{k',n''-1}, \\
&\dots \dots \dots (8) \\
s_{i',n''-1} &= \sum_{k'=0}^{n'-1} g((i'-k'),n''-1) d_{k',0} + \sum_{k'=0}^{n'-1} g((i'-k'),n''-2) d_{k',1} + \sum_{k'=0}^{n'-1} g((i'-k'),n''-3) d_{k',2} + \dots \\
&+ \sum_{k'=0}^{n'-1} g((i'-k'),0) d_{k',n''-1}
\end{aligned}$$

$(i' = 0, 1, \dots, n' - 1).$

Положим  $i'' = 0$ , тогда из первого выражения в формулах (8) имеем

$$\begin{aligned}
s_{0,0} &= \sum_{k'=0}^{n'-1} g((-k'),0) d_{k',0} + \sum_{k'=0}^{n'-1} g((-k'),n''-1) d_{k',1} + \dots + \sum_{k'=0}^{n'-1} g((-k'),1) d_{k',n''-1}, \\
s_{1,0} &= \sum_{k'=0}^{n'-1} g((1-k'),0) d_{k',0} + \sum_{k'=0}^{n'-1} g((1-k'),n''-1) d_{k',1} + \dots + \sum_{k'=0}^{n'-1} g((1-k'),1) d_{k',n''-1}, \\
s_{2,0} &= \sum_{k'=0}^{n'-1} g((2-k'),0) d_{k',0} + \sum_{k'=0}^{n'-1} g((2-k'),n''-1) d_{k',1} + \dots + \sum_{k'=0}^{n'-1} g((2-k'),1) d_{k',n''-1}, \\
&\dots \dots \dots (9) \\
s_{n'-1,0} &= \sum_{k'=0}^{n'-1} g((n'-1-k'),0) d_{k',0} + \sum_{k'=0}^{n'-1} g((n'-1-k'),n''-1) d_{k',1} + \dots + \sum_{k'=0}^{n'-1} g((n'-1-k'),1) d_{k',n''-1}.
\end{aligned}$$

Введем обозначения

$$\begin{aligned}
\tilde{s}_0^{(0)} &= (\tilde{s}_0^{(0)}, \tilde{s}_1^{(0)}, \dots, \tilde{s}_{n'-1}^{(0)})^T, \\
\tilde{s}_1^{(n''-1)} &= (\tilde{s}_0^{(1)}, \tilde{s}_1^{(1)}, \dots, \tilde{s}_{n'-1}^{(1)})^T, \\
&\dots \dots \dots \\
\tilde{s}_{n''-1}^{(1)} &= (\tilde{s}_0^{(n''-1)}, \tilde{s}_1^{(n''-1)}, \dots, \tilde{s}_{n'-1}^{(n''-1)})^T.
\end{aligned}$$

Здесь  $\tilde{s}_0^{(0)}$  —  $n'$ -точечная циклическая свертка последовательности  $\mathbf{d}^{(0)} = (d_{0,0}, d_{1,0}, \dots, d_{n'-1,0})$ , образующей первый столбец таблицы  $d$ , и последовательности  $\mathbf{g}^{(0)} = (g_{0,0}, g_{1,0}, \dots, g_{n'-1,0})$ , образующей первый столбец таблицы  $g$ ;  $\tilde{s}_1^{(n''-1)}$  —  $n'$ -точечная циклическая свертка последовательности  $\mathbf{d}^{(1)} = (d_{0,1}, d_{1,1}, \dots, d_{n'-1,1})$ , образующей второй столбец таблицы  $d$ , и последовательности  $\mathbf{g}^{(n''-1)} = (g_{0,n''-1}, g_{1,n''-1}, \dots, g_{n'-1,n''-1})$ , образующей  $n''$ -й (последний) столбец таблицы  $g$ ;  $\tilde{s}_2^{(n''-2)}$  —  $n'$ -точечная циклическая свертка последовательности  $\mathbf{d}^{(2)} = (d_{0,2}, d_{1,2}, \dots, d_{n'-1,2})$ , образующей третий столбец таблицы  $d$ , и последовательности  $\mathbf{g}^{(n''-2)} = (g_{0,n''-2}, g_{1,n''-2}, \dots, g_{n'-1,n''-2})$ , образующей  $(n'' - 1)$ -й столбец таблицы  $g$ , и т. д. Наконец,  $\tilde{s}_{n''-1}^{(1)}$  —  $n'$ -точечная циклическая свертка последовательности  $\mathbf{d}^{(n''-1)} = (d_{0,n''-1}, d_{1,n''-1}, \dots, d_{n'-1,n''-1})$ , образующей  $n''$ -й (последний) столбец таблицы  $d$ , и последовательности  $\mathbf{g}^{(1)} = (g_{0,1}, g_{1,1}, \dots, g_{n'-1,1})$ , образующей второй столбец таблицы  $g$ . В новых обозначениях формулы (7) можно записать в виде

$$\mathbf{s}^{(0)} = \tilde{s}_0^{(0)} + \tilde{s}_1^{(n''-1)} + \tilde{s}_2^{(n''-2)} + \dots + \tilde{s}_{n''-1}^{(1)}.$$

Положим  $i'' = 1$ , тогда из формул (8) получаем

$$\begin{aligned}
s_{0,1} &= \sum_{k'=0}^{n'-1} g((-k'),1) d_{k',0} + \sum_{k'=0}^{n'-1} g((-k'),0) d_{k',1} + \dots + \sum_{k'=0}^{n'-1} g((-k'),2) d_{k',n''-1}, \\
s_{1,1} &= \sum_{k'=0}^{n'-1} g((1-k'),1) d_{k',0} + \sum_{k'=0}^{n'-1} g((1-k'),0) d_{k',1} + \dots + \sum_{k'=0}^{n'-1} g((1-k'),2) d_{k',n''-1}, \\
s_{2,2} &= \sum_{k'=0}^{n'-1} g((2-k'),1) d_{k',0} + \sum_{k'=0}^{n'-1} g((2-k'),0) d_{k',1} + \dots + \sum_{k'=0}^{n'-1} g((2-k'),2) d_{k',n''-1}, \\
&\dots\dots\dots \\
s_{n'-1,1} &= \sum_{k'=0}^{n'-1} g((n'-1-k'),1) d_{k',0} + \sum_{k'=0}^{n'-1} g((n'-1-k'),0) d_{k',1} + \dots + \sum_{k'=0}^{n'-1} g((n'-1-k'),2) d_{k',n''-1}.
\end{aligned} \tag{10}$$

Используя аналогичные введенным после формул (9) обозначения, имеем

$$s^{(1)} = \tilde{s}_0^{(1)} + \tilde{s}_1^{(0)} + \tilde{s}_2^{(n''-1)} + \dots + \tilde{s}_{n''-1}^{(2)}.$$

Продолжая аналогичные рассуждения, получим

$$s^{(l)} = \tilde{s}_0^{(l)} + \tilde{s}_1^{(l-1)} + \dots + \tilde{s}_l^{(0)} + \tilde{s}_{l+1}^{(n''-1)} + \dots + \tilde{s}_{n''-1}^{(l+1)}$$

( $l = 0, 1, \dots, n'' - 1$ ).

Будем считать, что алгоритм вычисления  $n'$ -точечной циклической свертки  $\mathbf{h}$  последовательностей  $\mathbf{f}$  и  $\mathbf{e}$  получен и имеет вид

$$\mathbf{h} = C' F' A' \mathbf{e}, \tag{11}$$

где  $A'$  – матрица размера  $M(n') \times n'$ ,  $F'$  – диагональная матрица порядка  $M(n')$ ,  $C'$  – матрица размера  $n' \times M(n')$ . На диагонали матрицы  $F'$  стоят компоненты вектора  $B' \mathbf{f}$  (размеры матриц  $A'$  и  $B'$  одинаковы).

Тогда

$$\begin{aligned}
s^{(0)} &= C' G_0 A' \mathbf{d}^{(0)} + C' G_{n''-1} A' \mathbf{d}^{(1)} + C' G_{n''-2} A' \mathbf{d}^{(2)} + \dots + C' G_1 A' \mathbf{d}^{(n''-1)}, \\
s^{(1)} &= C' G_1 A' \mathbf{d}^{(0)} + C' G_0 A' \mathbf{d}^{(1)} + C' G_{n''-1} A' \mathbf{d}^{(2)} + \dots + C' G_2 A' \mathbf{d}^{(n''-1)}, \\
&\dots\dots\dots \\
s^{(n''-1)} &= C' G_{n''-1} A' \mathbf{d}^{(0)} + C' G_{n''-2} A' \mathbf{d}^{(1)} + C' G_{n''-3} A' \mathbf{d}^{(2)} + \dots + C' G_0 A' \mathbf{d}^{(n''-1)}
\end{aligned}$$

или

$$\bar{\mathbf{s}} = \begin{pmatrix} \mathbf{s}^{(0)} \\ \mathbf{s}^{(1)} \\ \mathbf{s}^{(2)} \\ \vdots \\ \mathbf{s}^{(n''-1)} \end{pmatrix} = C' * \begin{pmatrix} G_0 & G_{n''-1} & G_{n''-2} & \dots & G_1 \\ G_1 & G_0 & G_{n''-1} & \dots & G_2 \\ G_2 & G_1 & G_0 & \dots & G_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ G_{n''-1} & G_{n''-2} & G_{n''-3} & \dots & G_0 \end{pmatrix} \begin{pmatrix} A' \mathbf{d}^{(0)} \\ A' \mathbf{d}^{(1)} \\ A' \mathbf{d}^{(2)} \\ \vdots \\ A' \mathbf{d}^{(n''-1)} \end{pmatrix}, \tag{12}$$

где знак  $*$  означает, что матрицу  $C'$  следует применить к каждому вектору

$$S_i = \sum_{k=0}^{n''-1} G_{((i-k))} A' \mathbf{d}^{(k)} \quad (i = 0, 1, \dots, n'' - 1). \tag{13}$$

Здесь  $G_l$  ( $l = 0, 1, \dots, n'' - 1$ ) есть диагональная матрица порядка  $M(n')$ , на главной диагонали которой стоят компоненты вектора  $B' \mathbf{g}^{(l)}$ .

Равенство (13) означает, что  $\mathbf{S} = (S_0, S_1, \dots, S_{n''-1})^T$  представляет собой  $n''$ -точечную циклическую свертку последовательности  $A' \bar{\mathbf{d}}$ , состоящей из векторных блоков  $A' \mathbf{d}^{(i'')}$  ( $i'' = 0, \dots, n'' - 1$ )



длины  $M(n')$  каждый, и последовательности  $B'\bar{g}$ , состоящей из матричных блоков  $G_{i''}$  ( $i'' = 0, 1, \dots, n'' - 1$ ). (Каждую матрицу  $G_{i''}$  можно считать вектором, выписав в столбец элементы ее главной диагонали, в результате получим вектор длины  $M(n')$ .)

Пусть алгоритм вычисления  $n''$ -точечной циклической свертки последовательностей  $f$  и  $e$  имеет вид

$$h = C'' F'' A'' e, \quad (14)$$

где  $A''$  – матрица размера  $M(n'') \times n''$ ,  $F''$  – диагональная матрица порядка  $M(n'')$ ,  $C''$  – матрица размера  $n'' \times M(n'')$ . На диагонали матрицы  $F''$  стоят компоненты вектора  $B''f$  (размеры матриц  $A''$  и  $B''$  одинаковы).

Тогда из (13) и (14) получаем

$$S = C'' G A'' (A' \bar{d}), \quad (15)$$

где  $G$  – диагональная матрица, состоящая из  $M(n'')$  блоков, каждый из которых имеет длину  $M(n')$ , т. е. на диагонали  $G$  стоят "компоненты" вектора  $B''(B'\bar{g})$ , и каждая такая "компонента" есть векторный блок длины  $M(n')$ .

Рассмотрим сначала сомножитель  $A'' (A' \bar{d})$  из формулы (15). Поскольку компонентами  $A' \bar{d}$  являются векторные блоки  $A' d^{(0)}, A' d^{(1)}, \dots, A' d^{(n''-1)}$ , то

$$D = A'' (A' \bar{d}) = \begin{pmatrix} a''_{10} & \dots & a''_{1n''-1} \\ a''_{20} & \dots & a''_{2n''-1} \\ \vdots & \ddots & \vdots \\ a''_{m0} & \dots & a''_{mn''-1} \end{pmatrix} \begin{pmatrix} A' d^{(0)} \\ A' d^{(1)} \\ \vdots \\ A' d^{(n''-1)} \end{pmatrix} = \begin{pmatrix} \sum_{j=0}^{n''-1} a''_{1j} A' d^{(j)} \\ \sum_{j=0}^{n''-1} a''_{2j} A' d^{(j)} \\ \vdots \\ \sum_{j=0}^{n''-1} a''_{mj} A' d^{(j)} \end{pmatrix} =$$

$$\begin{pmatrix} \sum_{j=0}^{n''-1} a''_{kj} A' d^{(j)} \end{pmatrix}_{k=0,1,\dots,n''-1} = A \bar{d}, \quad (m = M(n'')),$$

где  $A$  – матрица, состоящая из блоков  $a''_{kj} A'$  ( $k = 1, \dots, M(n'')$ ;  $j = 0, 1, \dots, n'' - 1$ ), значит,  $A$  есть кронекеровское произведение  $A'' \times A'$ .

Рассмотрим вектор  $B''(B'\bar{g})$ , компоненты которого стоят на диагонали матрицы  $G$ :

$$B''(B'\bar{g}) = \begin{pmatrix} b''_{10} & b''_{11} & \dots & b''_{1n''-1} \\ b''_{20} & b''_{21} & \dots & b''_{2n''-1} \\ \vdots & \vdots & \ddots & \vdots \\ b''_{m0} & b''_{m1} & \dots & b''_{mn''-1} \end{pmatrix} \begin{pmatrix} B' g^{(0)} \\ B' g^{(1)} \\ \vdots \\ B' g^{(n''-1)} \end{pmatrix} = \begin{pmatrix} \sum_{j=0}^{n''-1} b''_{1j} B' g^{(j)} \\ \sum_{j=0}^{n''-1} b''_{2j} B' g^{(j)} \\ \vdots \\ \sum_{j=0}^{n''-1} b''_{mj} B' g^{(j)} \end{pmatrix} =$$

$$\begin{pmatrix} \sum_{j=0}^{n''-1} b''_{kj} B' g^{(j)} \end{pmatrix}_{k=0,1,\dots,n''-1} = B \bar{g} = G, \quad (m = M(n'')),$$

т. е. матрица  $B$  состоит из блоков  $b''_{kj} B'$  ( $k = 1, 2, \dots, M(n'')$ ;  $j = 0, 1, \dots, n'' - 1$ ), значит,  $B$  есть кронекеровское произведение  $B'' \times B'$ .

Теперь формулу (15) можно переписать в виде

$$S = C'' G A \bar{d} = C'' \{G \bullet D\},$$

где  $\bullet$  означает покомпонентное умножение, вектор  $\{G \bullet D\}$  имеет длину  $M(n'')$ , а каждая его "компонента" есть векторный блок длины  $M(n')$ . Пусть

$$G \bullet D = \tilde{S},$$



где  $\tilde{\mathbf{S}} = (\tilde{\mathbf{S}}_0, \tilde{\mathbf{S}}_1, \dots, \tilde{\mathbf{S}}_{M(n'')-1})$ .

Матрицу  $C''$  применяем к "компонентам" вектора  $\tilde{\mathbf{S}}$ , т. е.

$$\mathbf{S} = C'' \tilde{\mathbf{S}} = \begin{pmatrix} c''_{10} & c''_{11} & \dots & c''_{1t-1} \\ c''_{20} & c''_{21} & \dots & c''_{2t-1} \\ \vdots & \vdots & \ddots & \vdots \\ c''_{n''0} & c''_{n''1} & \dots & c''_{n''t-1} \end{pmatrix} \begin{pmatrix} \tilde{\mathbf{S}}_0 \\ \tilde{\mathbf{S}}_1 \\ \vdots \\ \tilde{\mathbf{S}}_{t-1} \end{pmatrix} = \begin{pmatrix} \sum_{j=0}^{t-1} c''_{1j} \tilde{\mathbf{S}}_j \\ \sum_{j=0}^{t-1} c''_{2j} \tilde{\mathbf{S}}_j \\ \vdots \\ \sum_{j=0}^{t-1} c''_{n''j} \tilde{\mathbf{S}}_j \end{pmatrix} \quad (t = M(n'')),$$

где  $\sum_{j=0}^{M(n'')-1} c''_{kj} \tilde{\mathbf{S}}_j$  ( $k = 1, 2, \dots, n''$ ) есть векторный блок длины  $M(n')$ , т. е.  $\mathbf{S} = C'' \tilde{\mathbf{S}}$  есть вектор длины  $n''$ , каждая "компонента" которого есть векторный блок длины  $M(n')$ .

Чтобы найти вектор  $\bar{\mathbf{s}}$  по формуле (12), требуется к каждой "компоненте" полученного вектора  $\mathbf{S}$  применить матрицу  $C'$ , т. е.

$$\bar{\mathbf{s}} = C' \star \begin{pmatrix} S_0 \\ S_1 \\ \vdots \\ S_{n''-1} \end{pmatrix} = \begin{pmatrix} C' S_0 \\ C' S_1 \\ \vdots \\ C' S_{n''-1} \end{pmatrix} = \begin{pmatrix} \sum_{j=0}^{M(n'')-1} c'_{1j} C' \tilde{\mathbf{S}}_j \\ \sum_{j=0}^{M(n'')-1} c'_{2j} C' \tilde{\mathbf{S}}_j \\ \vdots \\ \sum_{j=0}^{M(n'')-1} c'_{n''j} C' \tilde{\mathbf{S}}_j \end{pmatrix} = C \tilde{\mathbf{S}},$$

где  $C$  – матрица, состоящая из блоков  $c''_{kj} C'$  ( $k = 1, 2, \dots, n''$ ;  $j = 0, 1, \dots, M(n'') - 1$ ), значит, она является кронекеровским произведением  $C'' \times C'$ .

Итак,

$$\bar{\mathbf{s}} = (C'' \times C') G (A'' \times A') \bar{\mathbf{d}}, \quad (16)$$

где  $G$  – квадратная матрица порядка  $M(n')M(n'')$ , на диагонали которой стоят компоненты вектора  $\mathbf{G} = (B'' \times B') \bar{\mathbf{g}}$ .

Используя последовательные операции над двумерными таблицами, полученный составной вычислительный алгоритм можно представить в виде единого алгоритма. Сначала умножим каждый столбец таблицы  $d$  на матрицу  $A'$ , а каждый столбец таблицы  $g$  на матрицу  $B'$ . В результате получим две новые таблицы размера  $M(n') \times n''$ . Затем умножим каждую строку первой новой таблицы на матрицу  $A''$ , и каждую строку второй новой таблицы на матрицу  $B''$  (т. е.  $A''$  умножается на векторы, стоящие в строках первой новой таблицы,  $B''$  – на векторы, стоящие в строках второй новой таблицы). Это даст нам две таблицы размера  $M(n'') \times M(n')$ . Перемножим покомпонентно эти таблицы. Полученную таким образом  $M(n'') \times M(n')$ -таблицу преобразуем к нужной выходной таблице  $s$  размера  $n' \times n''$ , умножив сначала каждый ее столбец на матрицу  $C''$ , а затем каждую строку полученной таблицы размера  $n'' \times M(n')$  – на матрицу  $C'$ .

Если входные двумерные таблицы  $d$  и  $g$  записаны каждая в стек по столбцам и представлены, таким образом, в виде одномерных массивов, выходная двумерная таблица также представлена в виде стека столбцов, то алгоритм может быть задан формулой (16).

Следует помнить, что компоненты выходного вектора  $\mathbf{s}$  и входного вектора  $\mathbf{d}$  выписаны не в своем естественном порядке. Сначала они выписывались в двумерную матрицу вдоль главной диагонали, а затем из этой матрицы в виде стека столбцов. Чтобы вернуть компоненты на их естественное место, надо переставить столбцы матрицы  $A$  и строки матрицы  $C$ .

Характеристики  $n$ -точечного ( $n = n'n''$ ) алгоритма Агарвала – Кули, т. е. полное число умножений  $M(n)$  и сложений  $A(n)$ , задаются формулами

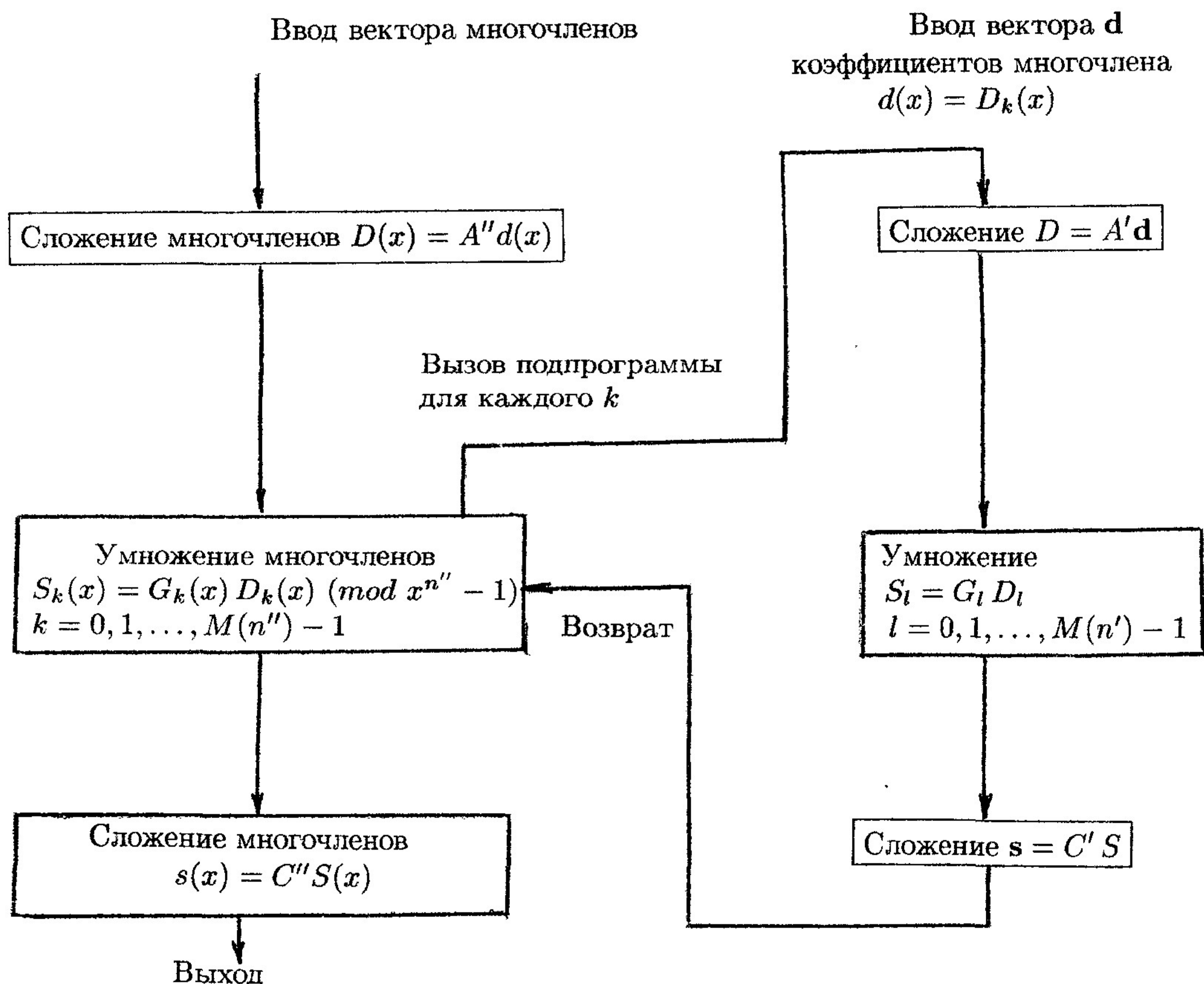
$$M(n) = M(n') M(n''),$$



$$A(n) = n'' A(n') + M(n') A(n'').$$

Из формул видно, что число умножений в малых алгоритмах играет существенно более важную роль, чем число сложений.

Сама же вычислительная процедура может быть представлена следующей схемой:



Поясним наши рассуждения на примере 6-точечной циклической свертки. Пусть  $n' = 2$ ,  $n'' = 3$ . Векторы  $(d_0, d_1, \dots, d_5)$  и  $(g_0, g_1, \dots, g_5)$  отображаются в  $2 \times 3$ -таблицы:

$$(d_0, d_1, \dots, d_5) \rightarrow \begin{pmatrix} d_0 & d_4 & d_2 \\ d_3 & d_1 & d_5 \end{pmatrix}, \quad (g_0, g_1, \dots, g_5) \rightarrow \begin{pmatrix} g_0 & g_4 & g_2 \\ g_3 & g_1 & g_5 \end{pmatrix},$$

выходной вектор  $(s_0, s_1, \dots, s_5)$  также отображается в таблицу

$$\begin{pmatrix} s_0 & s_4 & s_2 \\ s_3 & s_1 & s_5 \end{pmatrix}.$$

Тогда

$$\begin{aligned} d_0(y) &= d_0 + d_4 y + d_2 y^2, & g_0(y) &= g_0 + g_4 y + g_2 y^2, & s_0(y) &= s_0 + s_4 y + s_2 y^2, \\ d_1(y) &= d_3 + d_1 y + d_5 y^2, & g_1(y) &= g_3 + g_1 y + g_5 y^2, & s_1(y) &= s_3 + s_1 y + s_5 y^2, \end{aligned}$$

где

$$\begin{aligned} s_0(y) &= \sum_{k'=0}^1 g_{((-k'))}(y) d_{k'}(y) = d_0(y) g_0(y) + d_1(y) g_1(y) \pmod{y^3 - 1}, \\ s_1(y) &= \sum_{k'=0}^1 g_{((1-k'))}(y) d_{k'}(y) = d_0(y) g_1(y) + d_1(y) g_0(y) \pmod{y^3 - 1}. \end{aligned}$$

Значит,

$$s(x, y) = s_0(y) + s_1(y)x = (d_0(y) + d_1(y)x)(g_0(y) + g_1(y)x) = d(x, y)g(x, y) \pmod{x^2 - 1}. \quad (17)$$

Решаем задачу (17). Поскольку  $x^2 - 1 = (x - 1)(x + 1) = m_0(x)m_1(x)$ , то, применяя алгоритм Винограда, получаем

$$\begin{aligned} d(x, y) \pmod{x - 1} &= d_0(x, y) = d_0(y) + d_1(y), & g(x, y) \pmod{x - 1} &= g_0(x, y) = g_0(y) + g_1(y), \\ d(x, y) \pmod{x + 1} &= d_1(x, y) = d_0(y) - d_1(y), & g(x, y) \pmod{x + 1} &= g_1(x, y) = g_0(y) - g_1(y), \end{aligned}$$

откуда

$$\begin{aligned} s_0(x, y) &= d_0(x, y)g_0(x, y) \pmod{y^3 - 1} = (d_0(y) + d_1(y))(g_0(y) + g_1(y)) \pmod{y^3 - 1}, \\ s_1(x, y) &= d_1(x, y)g_1(x, y) \pmod{y^3 - 1} = (d_0(y) - d_1(y))(g_0(y) - g_1(y)) \pmod{y^3 - 1}. \end{aligned}$$

Из таблицы

$k$	$m_k(x)$	$M_k(x)$	$N_k(x)$
0	$x - 1$	$x + 1$	$\frac{1}{2}$
1	$x + 1$	$x - 1$	$-\frac{1}{2}$

по китайской теореме об остатках получаем

$$s(x, y) = \frac{1}{2} s_0(x, y)(x + 1) + \frac{1}{2} s_1(x, y)(1 - x) = \frac{1}{2} \{s_0(x, y) + s_1(x, y)\} + \frac{1}{2} \{s_0(x, y) - s_1(x, y)\}x,$$

откуда

$$\begin{aligned} s_0(y) &= \frac{1}{2} \{s_0(x, y) + s_1(x, y)\}, \\ s_1(y) &= \frac{1}{2} \{s_0(x, y) - s_1(x, y)\}. \end{aligned}$$

В матричной форме этот алгоритм записывается в виде

$$\begin{pmatrix} s_0(y) \\ s_1(y) \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} \frac{1}{2}(g_0(y) + g_1(y)) & 0 \\ 0 & \frac{1}{2}(g_0(y) - g_1(y)) \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} d_0(y) \\ d_1(y) \end{pmatrix} = C'G'A'd(y)$$

или

$$\begin{pmatrix} s_0(y) \\ s_1(y) \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} \frac{1}{2} s_0(x, y) \\ \frac{1}{2} s_1(x, y) \end{pmatrix}. \quad (18)$$

Очевидно,  $A(n') = 4$ ,  $M(n') = 2$ .

Теперь следует построить алгоритм решения задачи

$$s_i(x, y) = d_i(x, y)g_i(x, y) \pmod{y^3 - 1} \quad (i = 0, 1),$$

т. е. алгоритм 3-точечной циклической свертки.

Обозначим коэффициенты многочленов  $d_i(x, y)$ ,  $g_i(x, y)$  и  $s_i(y)$  соответственно  $d_k^{(i)}$ ,  $g_k^{(i)}$ ,  $s_k^{(i)}$  ( $k = 0, 1, 2$ ;  $i = 0, 1$ ), т. е.

$$\begin{aligned} d_i(x, y) &= d_0^{(i)} + d_1^{(i)}y + d_2^{(i)}y^2, \\ g_i(x, y) &= g_0^{(i)} + g_1^{(i)}y + g_2^{(i)}y^2, \\ s_i(y) &= s_0^{(i)} + s_1^{(i)}y + s_2^{(i)}y^2. \end{aligned}$$

Воспользуемся известным алгоритмом вычисления 3-точечной циклической свертки

$$\begin{pmatrix} s_0^{(i)} \\ s_1^{(i)} \\ s_2^{(i)} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & -1 \\ 1 & -1 & -1 & 2 \\ 1 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} G_0^{(i)} & & & \\ & G_1^{(i)} & & \\ & & G_2^{(i)} & \\ & & & G_3^{(i)} \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & -1 \\ 0 & 1 & -1 \\ 1 & 1 & -2 \end{pmatrix} \begin{pmatrix} d_0^{(i)} \\ d_1^{(i)} \\ d_2^{(i)} \end{pmatrix} = C''G''_iA''\bar{d}^{(i)} \quad (19)$$



где

$$\begin{pmatrix} G_0^{(i)} \\ G_1^{(i)} \\ G_2^{(i)} \\ G_3^{(i)} \end{pmatrix} = \frac{1}{3} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & -1 \\ 0 & 1 & -1 \\ 1 & 1 & -2 \end{pmatrix} \begin{pmatrix} g_0^{(i)} \\ g_1^{(i)} \\ g_2^{(i)} \end{pmatrix} \quad (i = 0, 1, ).$$

Здесь  $A(n'') = 11$ ,  $M(n'') = 4$ .

Объединим полученные алгоритмы. Из (19) следует, что

$$\begin{pmatrix} s_0^{(0)} \\ s_0^{(1)} \\ s_1^{(0)} \\ s_1^{(1)} \\ s_2^{(0)} \\ s_2^{(1)} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & -1 \\ 1 & 0 & -1 & 0 & -1 & 0 & 2 & 0 \\ 0 & 1 & 0 & -1 & 0 & -1 & 0 & 2 \\ 1 & 0 & 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} G_0^{(0)} & & & & & & \\ & G_0^{(1)} & & & & & \\ & & G_1^{(0)} & & & & \\ & & & G_1^{(1)} & & & \\ & & & & G_2^{(0)} & & \\ & & & & & G_2^{(1)} & \\ & & & & & & G_3^{(0)} \\ & & & & & & & G_3^{(1)} \end{pmatrix}.$$

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \\ 1 & 1 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & -2 \end{pmatrix} \begin{pmatrix} d_0^{(0)} \\ d_1^{(0)} \\ d_2^{(0)} \\ d_0^{(1)} \\ d_1^{(1)} \\ d_2^{(1)} \end{pmatrix},$$

а из (18) имеем

$$\begin{pmatrix} s_0 \\ s_3 \\ s_4 \\ s_1 \\ s_2 \\ s_5 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} s_0^{(0)} \\ s_0^{(1)} \\ s_1^{(0)} \\ s_1^{(1)} \\ s_2^{(0)} \\ s_2^{(1)} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & -1 & -1 \\ 1 & -1 & 1 & -1 & 0 & 0 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 2 & 2 \\ 1 & -1 & -1 & 1 & -1 & 1 & 2 & -2 \\ 1 & 1 & 0 & 0 & 1 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 & 1 & -1 & -1 & 1 \end{pmatrix}.$$

$$\begin{pmatrix} G_0^{(0)} & & & & & & \\ & G_0^{(1)} & & & & & \\ & & G_1^{(0)} & & & & \\ & & & G_1^{(1)} & & & \\ & & & & G_2^{(0)} & & \\ & & & & & G_2^{(1)} & \\ & & & & & & G_3^{(0)} \\ & & & & & & & G_3^{(1)} \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \\ 1 & 1 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & -2 \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} d_0 \\ d_3 \\ d_4 \\ d_1 \\ d_2 \\ d_5 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & -1 & -1 \\ 1 & -1 & 1 & -1 & 0 & 0 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 2 & 2 \\ 1 & -1 & -1 & 1 & -1 & 1 & 2 & -2 \\ 1 & 1 & 0 & 0 & 1 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 & 1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} G_0^{(0)} & & & & & & \\ & G_0^{(1)} & & & & & \\ & & G_1^{(0)} & & & & \\ & & & G_1^{(1)} & & & \\ & & & & G_2^{(0)} & & \\ & & & & & G_2^{(1)} & \\ & & & & & & G_3^{(0)} \\ & & & & & & & G_3^{(1)} \end{pmatrix}.$$

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & 0 & 0 & -1 & -1 \\ 1 & -1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & 1 & -1 & -1 \\ 0 & 0 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -2 & -2 \\ 1 & -1 & 1 & -1 & -2 & 2 \end{pmatrix} \begin{pmatrix} d_0 \\ d_3 \\ d_4 \\ d_1 \\ d_2 \\ d_5 \end{pmatrix} = CGA\bar{d}.$$

Очевидно, матрица постсложений равна  $C'' \times C'$ , а матрица предсложений —  $A'' \times A'$ . Осталось найти константы  $G_k^{(i)}$  ( $i = 0, 1$ ;  $k = 0, 1, 2, 3$ ). Рассуждая так же, как и выше, из (19) и (18) получим

$$\begin{pmatrix} G_0^{(0)} \\ G_0^{(1)} \\ G_1^{(0)} \\ G_1^{(1)} \\ G_2^{(0)} \\ G_2^{(1)} \\ G_3^{(0)} \\ G_3^{(1)} \end{pmatrix} = \frac{1}{3} \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \\ 1 & 1 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & -2 \end{pmatrix} \begin{pmatrix} g_0^{(0)} \\ g_1^{(0)} \\ g_2^{(0)} \\ g_0^{(1)} \\ g_1^{(1)} \\ g_2^{(1)} \end{pmatrix} =$$

$$\frac{1}{3} \cdot \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \\ 1 & 1 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & -2 \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} g_0 \\ g_3 \\ g_4 \\ g_1 \\ g_2 \\ g_5 \end{pmatrix} =$$

$$\frac{1}{6} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & 0 & 0 & -1 & -1 \\ 1 & -1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & 1 & -1 & -1 \\ 0 & 0 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -2 & -2 \\ 1 & -1 & 1 & -1 & -2 & 2 \end{pmatrix} \begin{pmatrix} g_0 \\ g_3 \\ g_4 \\ g_1 \\ g_2 \\ g_5 \end{pmatrix}, \quad (20)$$

т. е. матрица для нахождения  $G_k^{(i)}$  есть кронекеровское произведение матриц  $B''$  и  $B'$ , где  $B''$  — матрица для констант из алгоритма (19), а  $B'$  — матрица для констант из алгоритма (18). Если записать (20) в матричном виде, то

$$G = (B'' \times B') \bar{g}.$$

Таким образом, построенный алгоритм записан в виде

$$\bar{s} = (C'' \times C') (B'' \times B') (A'' \times A') \bar{d},$$

где векторы  $\bar{d}$  и  $\bar{s}$  представляют вектор-столбцы, полученные при выписывании в стек компонент таблиц  $d$  и  $s$  по столбцам. Аналогично,  $\bar{g}$  — вектор, получаемый при выписывании в стек компонент таблицы  $g$  по столбцам.

Если теперь переставить выходные компоненты и соответствующие строки матрицы  $C'' \times C'$ , а также входные компоненты и соответствующие столбцы матрицы  $A'' \times A'$ , то получим оконча-



тельный алгоритм

$$\begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & -1 & -1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 2 & -2 \\ 1 & 1 & 0 & 0 & 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 & 0 & 0 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 2 & 2 \\ 1 & -1 & 0 & 0 & 1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} G_0^{(0)} & & & & & & & \\ & G_0^{(1)} & & & & & & \\ & & G_1^{(0)} & & & & & \\ & & & G_1^{(1)} & & & & \\ & & & & G_2^{(0)} & & & \\ & & & & & G_2^{(1)} & & \\ & & & & & & G_3^{(0)} & \\ & & & & & & & G_3^{(1)} \end{pmatrix}.$$

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 0 & -1 & 1 & 0 & -1 \\ 1 & 0 & -1 & -1 & 0 & 1 \\ 0 & 1 & -1 & 0 & 1 & -1 \\ 0 & -1 & -1 & 0 & 1 & 1 \\ 1 & 1 & -2 & 1 & 1 & -2 \\ 1 & -1 & -2 & -1 & 1 & 2 \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{pmatrix},$$

где

$$\begin{pmatrix} G_0^{(0)} \\ G_0^{(1)} \\ G_1^{(0)} \\ G_1^{(1)} \\ G_2^{(0)} \\ G_2^{(1)} \\ G_3^{(0)} \\ G_3^{(1)} \end{pmatrix} = \frac{1}{6} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 0 & -1 & 1 & 0 & -1 \\ 1 & 0 & -1 & -1 & 0 & 1 \\ 0 & 1 & -1 & 0 & 1 & -1 \\ 0 & -1 & -1 & 0 & 1 & 1 \\ 1 & 1 & -2 & 1 & 1 & -2 \\ 1 & -1 & -2 & -1 & 1 & 2 \end{pmatrix} \begin{pmatrix} g_0 \\ g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_5 \end{pmatrix}.$$

Заметим, что полученный алгоритм отличается от построенного в § 6 структурой сложений. Число умножений также равно 8, а вот последовательность сложений отлична от той, что была в алгоритме § 6. Всего этот алгоритм потребует 34 операции сложения. Действительно, предположений потребуется всего 16:

$$\begin{aligned} t_1 &= d_0 + d_3, & t_2 &= d_4 + d_1, & t_3 &= d_2 + d_5, \\ t_4 &= d_0 - d_3, & t_5 &= d_4 - d_1, & t_6 &= d_2 - d_5, \\ D_0^{(0)} &= t_1 + t_2 + t_3, & D_1^{(0)} &= t_1 - t_3, & D_2^{(0)} &= t_2 - t_3, \\ D_3^{(0)} &= D_1^{(0)} + D_2^{(0)}, & D_0^{(1)} &= t_4 + t_5 + t_6, & D_1^{(1)} &= t_4 - t_6, \\ D_2^{(1)} &= t_5 - t_6, & D_3^{(1)} &= D_1^{(1)} + D_2^{(1)}. \end{aligned}$$

Постсложения можно реализовать за 18 операций:

$$\begin{aligned} T_1 &= S_2 - S_6, & T_2 &= S_4 - S_6, & T_3 &= T_1 + T_2, \\ T_4 &= S_3 - S_7, & T_5 &= S_5 - S_7, & T_6 &= T_4 + T_5, \\ s_0^{(0)} &= S_0 + T_1, & s_1^{(0)} &= S_0 - T_3, & s_2^{(0)} &= S_0 + T_2, \\ s_0^{(1)} &= S_1 + T_4, & s_1^{(1)} &= S_1 - T_6, & s_2^{(1)} &= S_1 + T_5, \\ s_0 &= s_0^{(0)} + s_0^{(1)}, & s_3 &= s_0^{(0)} - s_0^{(1)}, & s_4 &= s_1^{(0)} + s_1^{(1)}, \\ s_1 &= s_1^{(0)} - s_1^{(1)}, & s_2 &= s_2^{(0)} + s_2^{(1)}, & s_5 &= s_2^{(0)} - s_2^{(1)}. \end{aligned}$$



## §25. СЛОЖНОСТЬ АЛГОРИТМОВ СВЕРТКИ

Здесь мы будем исследовать характеристики оптимального по числу умножений алгоритма свертки. Для этого сначала уточним понятие умножения. Определим умножение таким образом, чтобы под произведением  $d \cdot g$  понималось произведение произвольных вещественных чисел, но туда не входило бы, например, произведение  $2g$ , которое можно интерпретировать как  $g + g$  и т. п.

Рассматриваемые идеи не зависят от поля. Пусть  $F$  – поле, которому принадлежат элементы последовательностей  $\mathbf{d} = (d_0, d_1, \dots, d_{n-1})$  и  $\mathbf{g} = (g_0, g_1, \dots, g_{r-1})$ , это поле будем называть *полем вычислений*. Пусть  $E$  – подполе поля  $F$ , называемое *полем констант* или *основным полем*. Элементы поля  $E$  будем называть *скалярами*.

Компоненты векторов  $\mathbf{d}$  и  $\mathbf{g}$  будем называть *неопределенными переменными* или просто *переменными*. Эти  $n + r$  величин независимы, их не связывают никакие соотношения.

**Определение 1.** Вычисление  $d \cdot g$  будем называть *умножением* только тогда, когда оба сомножителя являются произвольными элементами поля  $F$ , и не будем считать умножением, если только один из них является произвольным элементом поля  $F$ , а другой принадлежит полю констант  $E$ .

С точки зрения определения 1 алгоритмы Винограда вычисления коротких сверток являются оптимальными. Ни один алгоритм вычисления короткой свертки не содержит умножений меньше, чем алгоритм Винограда. Для проверки этого утверждения мы докажем, что:

1. Никакой алгоритм вычисления линейной свертки двух последовательностей длин  $N$  и  $L$  не может содержать меньше, чем  $N + L - 1$  умножений.

2. Если число попарно взаимно простых неприводимых делителей многочлена  $x^n - 1$  равно  $k$ , то никакой алгоритм вычисления  $n$ -точечной циклической свертки не может содержать меньше, чем  $2n - k$  умножений.

3. Если число попарно взаимно простых неприводимых делителей многочлена  $p(x)$  равно  $k$  и  $\deg p(x) = n$ , то никакой алгоритм вычисления произведения многочленов  $d(x)g(x)$  по модулю многочлена  $p(x)$  не может содержать умножений меньше, чем  $2n - k$ .

Очевидно, что второе утверждение является частным случаем третьего. На самом деле, мы докажем третье утверждение, из которого следует второе. Мы выделили утверждение 2, так как оно дает оценку нижней границы числа умножений для  $n$ -точечной циклической свертки, что для нас важно.

Введем формальное определение вычислительного алгоритма.

**Определение 2.** *Вычислительным алгоритмом* назовем правило вычисления последовательности элементов  $f_1, f_2, \dots, f_t$  из поля  $F$  таких, что каждый элемент  $f_j$  последовательности равен одной из следующих величин:

- а) либо компоненте вектора  $\mathbf{d}$ , либо компоненте вектора  $\mathbf{g}$ , либо сумме, разности или произведению двух таких компонент;
- б) сумме, разности или произведению компоненты вектора  $\mathbf{d}$  или вектора  $\mathbf{g}$  и элемента  $f_k$  последовательности, где  $k < j$ ;
- в) сумме, разности или произведению двух элементов  $f_k$  и  $f_l$  последовательности таких, что  $k < j$  и  $l < j$ ;
- г) элементу поля  $E$ .

Будем говорить, что *алгоритм вычисляет* выходной вектор  $\mathbf{s}$ , если компоненты вектора  $\mathbf{s}$  содержатся в последовательности  $f_1, f_2, \dots, f_t$ . Алгоритм, вычисляющий вектор  $\mathbf{s}$ , представляет собой фиксированную процедуру правильного вычисления компонент вектора  $\mathbf{s}$  для любых возможных значений входящих в  $\mathbf{d}$  и  $\mathbf{g}$  переменных.

Данное определение алгоритма можно проиллюстрировать примером комплексного умножения. Прямой алгоритм умножения комплексных чисел в матрично-векторной форме имеет вид

$$\begin{pmatrix} e \\ f \end{pmatrix} = \begin{pmatrix} c & -d \\ d & c \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}.$$

Тогда равенства

$$f_1 = ca, \quad f_2 = db, \quad f_3 = f_1 - f_2, \quad f_4 = da, \quad f_5 = cb, \quad f_6 = f_4 + f_5$$



дают описание алгоритма в смысле определения 2.

Быстрый алгоритм комплексного умножения

$$\begin{pmatrix} e \\ f \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} c-d & 0 & 0 \\ 0 & c+d & 0 \\ 0 & 0 & d \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}$$

можно записать в виде следующей последовательности вычислений:

$$f_1 = a - b, \quad f_2 = c - d, \quad f_3 = c + d, \quad f_4 = f_2 a, \quad f_5 = f_3 b, \quad f_6 = d f_1, \quad f_7 = f_4 + f_6, \quad f_8 = f_5 + f_6.$$

Рассмотрим набор всех множеств правил вычисления комплексного умножения. Оптимальным из этого набора алгоритмов является тот из них, который содержит наименьшее число умножений.

Теперь формализуем определение задачи вычислений. Будем рассматривать только задачи, которые можно записать в векторно-матричной форме

$$s = H d, \quad (1)$$

где  $d$  – входной вектор данных длины  $n$ , а  $s$  – выходной вектор длины  $k$ . Элементы матрицы  $H$  представляют собой линейные комбинации  $r$  неопределенных переменных  $g_0, g_1, \dots, g_{r-1}$  (компонент вектора  $g$ ). В типичных случаях  $r$  меньше числа элементов матрицы  $H$ , т. е. каждая переменная может присутствовать в выражении нескольких элементов матрицы  $H$ . Такая структура матрицы  $H$  и вид задачи (1) включают в себя все рассмотренные нами выше задачи вычисления свертки.

Элементы матрицы  $H = (h_{ij})$  представляют собой линейные комбинации неопределенных переменных  $g_0, g_1, \dots, g_{r-1}$  с коэффициентами из поля скаляров  $E$ , т. е.

$$h_{ij} = \sum_{k=0}^{r-1} \alpha_{ijk} g_k, \quad \alpha_{ijk} \in E. \quad (2)$$

Две линейные формы вида (2) можно складывать, любую линейную форму можно умножить на скаляр из поля  $E$ . Нетрудно видеть, что относительно этих операций множество таких линейных форм над полем  $E$  образует линейное (векторное) пространство, обозначаемое  $E[g_0, g_1, \dots, g_{r-1}]$ . Значит,  $H$  является матрицей над множеством линейных форм. Для таких матриц не выполняются многие известные свойства матриц над полем. В частности, ранг по столбцам матрицы  $H$  над линейным пространством не обязательно равен рангу ее по строкам. Например, для матрицы

$$H = \begin{pmatrix} 4g_0 & -g_0 & g_0 \\ 2g_1 & -g_1 & 0 \\ 2g_1 - 2g_0 & g_0 & g_1 \end{pmatrix}$$

ее ранг по столбцам равен 2, а ранг по строкам равен 3. Но именно ранги матрицы  $H$  по строкам и по столбцам дают нижнюю оценку для числа умножений в задаче  $s = H d$ , как мы увидим из нижеследующих утверждений.

**Определение 3.** Рангом по строкам матрицы  $H$  называется максимальное число ее линейно независимых строк. Рангом по столбцам матрицы  $H$  называется максимальное число ее линейно независимых столбцов.

**Теорема 1.** Для произвольного алгоритма вычисления  $s = H d$  число умножений не меньше ранга по строкам матрицы  $H$ .

*Доказательство.* Без ограничения общности будем считать, что линейно независимыми являются первые  $k$  строк матрицы  $H$ . Далее будем рассматривать только эти строки. Обозначим через  $M$  матрицу, составленную из линейно независимых строк матрицы  $H$ . Рассмотрим лишь частичное вычисление компонент выходного вектора

$$\begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{k-1} \end{pmatrix} = \begin{pmatrix} h_{00} & \dots & h_{0n-1} \\ h_{10} & \dots & h_{1n-1} \\ \vdots & \ddots & \vdots \\ h_{k-10} & \dots & h_{k-1n-1} \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ \vdots \\ d_{n-1} \end{pmatrix}$$



или

$$\mathbf{s}' = M \mathbf{d}. \quad (3)$$

Предположим, что в алгоритме  $f_1, f_2, \dots, f_N$  имеется  $l$  умножений, задаваемых соответственно  $l$  членами этой последовательности  $f_{i_1}, f_{i_2}, \dots, f_{i_l}$ . Тогда первые  $k$  компонент вектора  $\mathbf{s}$  должны быть линейными комбинациями этих членов-произведений, линейных членов и элементов основного поля, т. е.

$$\begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{k-1} \end{pmatrix} = \begin{pmatrix} a_{11} & \dots & a_{1l} \\ a_{21} & \dots & a_{2l} \\ \vdots & \ddots & \vdots \\ a_{k1} & \dots & a_{kl} \end{pmatrix} \begin{pmatrix} f_{i_1} \\ f_{i_2} \\ \vdots \\ f_{i_l} \end{pmatrix} + \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{k-1} \end{pmatrix}$$

или

$$\mathbf{s}' = A \mathbf{e} + \mathbf{b}. \quad (4)$$

Элементы матрицы  $A$  принадлежат основному полю  $E$ , т. е. являются скалярами, а компоненты вектора  $\mathbf{b}$  являются линейными комбинациями неопределенных переменных и элементов основного поля.

Предположим, что  $k > l$ . Тогда в матрице  $A$  строк больше, чем столбцов, следовательно, строки матрицы  $A$  линейно зависимы, т. е. существует такой ненулевой вектор  $\mathbf{c}$  с констант из поля  $E$ , что

$$\mathbf{c}^T A = \mathbf{0}^T.$$

Приравнивая правые части формул (3) и (4), получаем

$$M \mathbf{d} = A \mathbf{e} + \mathbf{b},$$

откуда

$$\mathbf{c}^T (M \mathbf{d}) = \mathbf{c}^T (A \mathbf{e} + \mathbf{b}). \quad (5)$$

Так как  $\mathbf{c}^T A = \mathbf{0}^T$ , то из (5) получаем

$$\mathbf{c}^T (M \mathbf{d}) = \mathbf{c}^T \mathbf{b}. \quad (6)$$

В равенстве (6) правая часть не содержит произведений неопределенных переменных, так как  $\mathbf{c}$  содержит только элементы поля  $E$ , а в  $\mathbf{b}$  не входят произведения неопределенных переменных, следовательно, и левая часть равенства (6) не содержит произведений неопределенных переменных. Однако компонентами вектора  $\mathbf{d}$  являются неопределенные переменные, значит,  $\mathbf{c}^T M$  не содержит неопределенных переменных. Но  $\mathbf{c}^T M$  есть вектор, компоненты которого равны линейным комбинациям неопределенных переменных, следовательно, он равен нулю, т. е.

$$\mathbf{c}^T M = \mathbf{0}.$$

Из последнего равенства следует, что строки матрицы  $M$  линейно зависимы. Полученное противоречие означает, что  $l \geq k$ , т. е. число умножений алгоритма не меньше, чем ранг по строкам матрицы  $H$ .

Доказанная теорема позволяет получить нижнюю границу числа умножений в алгоритмах вычисления линейной свертки.

**Теорема 2.** *Каждый алгоритм вычисления линейной свертки*

$$s(x) = d(x) g(x),$$

где  $\deg d(x) = N - 1$ ,  $\deg g(x) = L - 1$ , содержит по меньшей мере  $N + L - 1$  умножений.

**Доказательство.** Вычисление линейной свертки в матричном виде можно записать как  $\mathbf{s} = H \mathbf{d}$ , где

$$H = \begin{pmatrix} g_0 & 0 & 0 & \dots & 0 & 0 \\ g_1 & g_0 & 0 & \dots & 0 & 0 \\ g_2 & g_1 & g_0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & g_{L-1} & g_{L-2} \\ 0 & 0 & 0 & \dots & 0 & g_{L-1} \end{pmatrix}$$



является  $(N + L - 1) \times N$  - матрицей. В строках этой матрицы стоят элементы векторного пространства  $E[g_0, g_1, \dots, g_{L-1}]$ . Строки этой матрицы, очевидно, линейно независимы, следовательно, в силу теоремы 1 число умножений в любом алгоритме вычисления линейной свертки не меньше, чем число строк матрицы  $H$ , т. е. не меньше, чем  $N + L - 1$ .

**Теорема 3.** Для произвольного алгоритма вычисления  $s = H d$  число умножений не меньше ранга по столбцам матрицы  $H$ .

*Доказательство.* Проведем доказательство по индукции. Индукция проводится по величине ранга матрицы  $H$  по столбцам. Если этот ранг равен единице, то, очевидно, хотя бы одно умножение будет содержаться в любом алгоритме вычисления  $s = H d$ .

Предположим, что теорема верна, если ранг по столбцам матрицы  $H$  равен  $l - 1$ , т. е. любой алгоритм вычисления  $s = H d$  в этом случае содержит не меньше, чем  $l - 1$  умножение.

Пусть задан алгоритм вычисления  $s = H d$ , причем ранг по столбцам матрицы  $H$  равен  $l$ . Без ограничения общности можно считать, что линейно независимы первые  $l$  столбцов матрицы  $H$  (в противном случае можно переставить линейно независимые столбцы на первые  $l$  мест, одновременно производя перестановку соответствующих компонент входного вектора  $d$ ). Так как первые  $l$  столбцов  $H$  линейно независимы, то  $l$ -й столбец не может быть нулевым, т. е. он содержит по крайней мере один ненулевой элемент, следовательно, переменная  $d_l$  содержится в некотором члене-произведении. Это означает, что в алгоритме вычисления имеется умножение, в котором участвует в качестве сомножителя линейная форма

$$\sum_{i=0}^{n-1} \alpha_i d_i = \alpha_l d_l + \sum_{\substack{i=0 \\ i \neq l}}^{n-1} \alpha_i d_i, \quad (7)$$

где  $\alpha_l \neq 0$ ,  $\alpha_i \in E$  ( $i = 0, 1, \dots, n - 1$ ).

Построим новую искусственную задачу  $s' = H' d'$ , где ранг по столбцам матрицы  $H'$  равен  $l - 1$ , такую, что ее можно решить нашим алгоритмом, удаляя из него одно умножение. Для этого заменим в исходной задаче переменную  $d_l$  на

$$-\sum_{\substack{i=0 \\ i \neq l}}^{n-1} \frac{\alpha_i}{\alpha_l} d_i = -\sum_{\substack{i=0 \\ i \neq l}}^{n-1} \beta_i d_i \quad (\beta_i = \frac{\alpha_i}{\alpha_l}, \quad i = 0, 1, \dots, l - 1, l + 1, \dots, n - 1).$$

Тогда тот член-произведение исходного алгоритма, который содержал умножение на линейную форму (7), будет представлять собой умножение на 0 и, следовательно, может быть удален из алгоритма. Алгоритм теперь решает некоторую новую задачу

$$s' = H' d'.$$

Выясним, как выглядит матрица  $H'$ :

$$\begin{pmatrix} h_{00} & h_{01} & \dots & h_{0n-1} \\ h_{10} & h_{11} & \dots & h_{1n-1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{l-10} & h_{l-11} & \dots & h_{l-1n-1} \\ h_{l0} & h_{l1} & \dots & h_{ln-1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{k-10} & h_{k-11} & \dots & h_{k-1n-1} \end{pmatrix} \begin{pmatrix} d_0 \\ d_1 \\ \vdots \\ d_{l-1} \\ -\sum_{\substack{i=0 \\ i \neq l}}^{n-1} \beta_i d_i \\ \vdots \\ d_{n-1} \end{pmatrix} = \begin{pmatrix} \sum_{\substack{j=0 \\ j \neq l}}^{n-1} h_{0j} d_j - h_{0l} \sum_{\substack{j=0 \\ j \neq l}}^{n-1} \beta_j d_j \\ \sum_{\substack{j=0 \\ j \neq l}}^{n-1} h_{1j} d_j - h_{1l} \sum_{\substack{j=0 \\ j \neq l}}^{n-1} \beta_j d_j \\ \vdots \\ \sum_{\substack{j=0 \\ j \neq l}}^{n-1} h_{k-1j} d_j - h_{k-1l} \sum_{\substack{j=0 \\ j \neq l}}^{n-1} \beta_j d_j \end{pmatrix} =$$



$$\begin{pmatrix} \sum_{\substack{j=0 \\ j \neq l}}^{n-1} (h_{0j} - h_{0l} \beta_j) d_j \\ \sum_{\substack{j=0 \\ j \neq l}}^{n-1} (h_{1j} - h_{1l} \beta_j) d_j \\ \vdots \\ \sum_{\substack{j=0 \\ j \neq l}}^{n-1} (h_{k-1j} - h_{k-1l} \beta_j) d_j \end{pmatrix} = \begin{pmatrix} h_{00} - h_{0l} \beta_0 & \dots & h_{0l-1} - h_{0l} \beta_{l-1} & h_{0l+1} - h_{0l} \beta_{l+1} & \dots & h_{0n-1} - h_{0l} \beta_{n-1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ h_{l-10} - h_{l-1l} \beta_0 & \dots & h_{l-1l-1} - h_{l-1l} \beta_{l-1} & h_{l-1l+1} - h_{l-1l} \beta_{l+1} & \dots & h_{l-1n-1} - h_{l-1l} \beta_{n-1} \\ h_{l+10} - h_{l+1l} \beta_0 & \dots & h_{l+1l-1} - h_{l+1l} \beta_{l-1} & h_{l+1l+1} - h_{l+1l} \beta_{l+1} & \dots & h_{l+1n-1} - h_{l+1l} \beta_{n-1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ h_{k-10} - h_{k-1l} \beta_0 & \dots & h_{k-1l-1} - h_{k-1l} \beta_{l-1} & h_{k-1l+1} - h_{k-1l} \beta_{l+1} & \dots & h_{k-1n-1} - h_{k-1l} \beta_{n-1} \end{pmatrix} \begin{pmatrix} d_0 \\ \vdots \\ d_{l-1} \\ d_{l+1} \\ \vdots \\ d_{n-1} \end{pmatrix}$$

Итак, вектор  $\mathbf{d}'$  представляет собой вектор длины  $n-1$ , формируемый из вектора  $\mathbf{d}$  удалением его  $l$ -й компоненты, а матрица  $H'$  получается из матрицы  $H$  заменой каждого столбца  $\mathbf{h}_j$  на столбец  $\mathbf{h}_j - \beta_j \mathbf{h}_l$  ( $j = 0, 1, \dots, l-1, l+1, \dots, n-1$ ) и выбрасыванием  $l$ -го столбца.

Ранг по столбцам матрицы  $H'$ , очевидно, равен  $l-1$ . По предположению индукции любой алгоритм решения задачи  $\mathbf{s}' = H' \mathbf{d}'$  содержит не менее, чем  $l-1$  умножение, значит, исходная задача  $\mathbf{s} = H \mathbf{d}$  данным алгоритмом решается по меньшей мере с  $l$  умножениями.

Рассмотрим задачу вычисления

$$s(x) = d(x) g(x) \pmod{p(x)}, \quad g(x), d(x) \in F[x], \quad (8)$$

где  $p(x) = x^n + p_{n-1}x^{n-1} + \dots + p_1x + p_0$  — унитарный неприводимый многочлен в кольце  $E[x]$ , ( $E \subset F$ ). Запишем вычисление в матричной форме

$$\mathbf{s} = H \mathbf{d},$$

где  $H$  — матрица, элементами которой являются линейные формы коэффициентов многочлена  $g(x)$ . Нас интересует вид матрицы  $H$ .

Обозначим через  $C_p$  сопровождающую матрицу многочлена  $p(x)$ , определяемую  $n \times n$  матрицей вида

$$C_p = \begin{pmatrix} 0 & 0 & \dots & 0 & -p_0 \\ 1 & 0 & \dots & 0 & -p_1 \\ 0 & 1 & \dots & 0 & -p_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & -p_{n-1} \end{pmatrix}.$$

Заметим, что

$$\det(C_p - xE) = \begin{vmatrix} -x & 0 & 0 & \dots & 0 & -p_0 \\ 1 & -x & 0 & \dots & 0 & -p_1 \\ 0 & 1 & -x & \dots & 0 & -p_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & \dots & 1 & -p_{n-1} - x \end{vmatrix} = (-1)^{n+2} p(x). \quad \ominus$$

Утверждается, что матрица  $H$  для задачи (8) имеет вид

$$H = (\mathbf{g} \quad C_p \mathbf{g} \quad C_p^2 \mathbf{g} \quad \dots \quad C_p^{n-1} \mathbf{g}).$$



Проверим, что формула (10) верна для любого натурального  $n$ . Доказательство проведем индукцией по  $n$ . Пусть  $H = (h_{ij})$  ( $i, j = 0, 1, \dots, n-1$ ). Рассмотрим  $s(x)$ :

$$s(x) = g(x) d(x) = g(x) d_0 + x g(x) d_1 + x^2 g(x) d_2 + \dots + x^k g(x) d_k + \dots + x^{n-1} g(x) d_{n-1} =$$

$$(g_0 + g_1 x + \dots + g_{n-1} x^{n-1}) d_0 + (g_0 x + g_1 x^2 + \dots + g_{n-1} x^n) d_1 + \dots + (g_0 x^k + g_1 x^{k+1} + \dots + g_{n-1} x^{n+k-1}) d_k +$$

$$(g_0 x^{k+1} + g_1 x^{k+2} + \dots + g_{n-1} x^{n+k}) d_{k+1} + \dots + (g_0 x^{n-1} + g_1 x^n + g_2 x^{n+1} + \dots + g_{n-1} x^{2n-2}) d_{n-1}.$$

В  $i$ -м столбце матрицы  $H$  стоят, очевидно, коэффициенты многочлена  $x^{i-1} g(x) \pmod{p(x)}$ . В частности, при  $i = 1$  получаем столбец, состоящий из коэффициентов многочлена  $g(x)$ , т. е. вектор-столбец  $\mathbf{g}$ . Посмотрим, какой вид имеет второй столбец матрицы  $H$ . Пусть  $i = 2$ , тогда

$$x g(x) \pmod{p(x)} = x \sum_{l=0}^{n-1} g_l x^l \pmod{p(x)} = \sum_{l=0}^{n-2} g_l x^{l+1} + g_{n-1} x^n \pmod{p(x)} \equiv$$

$$\sum_{l=0}^{n-2} g_l x^{l+1} + g_{n-1} \left( - \sum_{l=0}^{n-1} p_l x^l \right) = \sum_{l=1}^{n-1} g_{l-1} x^l - g_{n-1} \sum_{l=0}^{n-1} p_l x^l =$$

$$-p_0 g_{n-1} + \sum_{l=1}^{n-1} (g_{l-1} - g_{n-1} p_l) x^l = \sum_{l=0}^{n-1} h_{l1} x^l$$

в силу  $x^n \equiv - \sum_{l=0}^{n-1} p_l x^l \pmod{p(x)}$ .

Значит, второй столбец матрицы  $H$  имеет вид

$$\begin{pmatrix} h_{01} \\ h_{11} \\ h_{21} \\ \vdots \\ h_{n-21} \\ h_{n-11} \end{pmatrix} = \begin{pmatrix} -p_0 g_{n-1} \\ g_0 - p_1 g_{n-1} \\ g_1 - p_2 g_{n-1} \\ \vdots \\ g_{n-3} - p_{n-2} g_{n-1} \\ g_{n-2} - p_{n-1} g_{n-1} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & 0 & -p_0 \\ 1 & 0 & 0 & \dots & 0 & 0 & -p_1 \\ 0 & 1 & 0 & \dots & 0 & 0 & -p_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \\ 0 & 0 & 0 & \dots & 1 & 0 & -p_{n-2} \\ 0 & 0 & 0 & \dots & 0 & 1 & -p_{n-1} \end{pmatrix} \begin{pmatrix} g_0 \\ g_1 \\ g_2 \\ \vdots \\ g_{n-2} \\ g_{n-1} \end{pmatrix} = C_p \mathbf{g}.$$

При  $i = 3$  получаем

$$x^2 g(x) \pmod{p(x)} = x \{x g(x) \pmod{p(x)}\} \pmod{p(x)} \equiv x \left\{ \sum_{l=0}^{n-1} h_{l1} x^l \right\} \pmod{p(x)} =$$

$$\sum_{l=0}^{n-1} h_{l1} x^{l+1} \pmod{p(x)} = \sum_{l=0}^{n-2} h_{l1} x^{l+1} + h_{n-11} x^n \pmod{p(x)} \equiv \sum_{l=0}^{n-2} h_{l1} x^{l+1} - h_{n-11} \sum_{l=0}^{n-1} p_l x^l =$$

$$-h_{n-11} p_0 + \sum_{l=1}^{n-1} h_{l-11} x^l - h_{n-11} \sum_{l=1}^{n-1} p_l x^l = -h_{n-11} p_0 - \sum_{l=1}^{n-1} (h_{l-11} - h_{n-11} p_l) x^l,$$

т. е. третий столбец матрицы  $H$  имеет вид

$$\begin{pmatrix} h_{02} \\ h_{12} \\ h_{22} \\ \vdots \\ h_{n-12} \end{pmatrix} = \begin{pmatrix} -p_0 h_{n-11} \\ h_{01} - p_1 h_{n-11} \\ h_{11} - p_2 h_{n-11} \\ \vdots \\ h_{n-21} - p_{n-1} h_{n-11} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & 0 & -p_0 \\ 1 & 0 & 0 & \dots & 0 & 0 & -p_1 \\ 0 & 1 & 0 & \dots & 0 & 0 & -p_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \\ 0 & 0 & 0 & \dots & 1 & 0 & -p_{n-2} \\ 0 & 0 & 0 & \dots & 0 & 1 & -p_{n-1} \end{pmatrix} \begin{pmatrix} h_{01} \\ h_{11} \\ h_{21} \\ \vdots \\ h_{n-11} \end{pmatrix} =$$

$$C_p(C_p \mathbf{g}) = C_p^2 \mathbf{g}.$$

Предположим, что  $k$ -й столбец матрицы  $H$  имеет вид  $C_p^{k-1} \mathbf{g}$ . Найдем  $(k+1)$ -й столбец этой матрицы. В  $k$ -м столбце  $H$  стоят коэффициенты многочлена  $x^{k-1} g(x) \pmod{p(x)}$ , а в  $(k+1)$ -м столбце – коэффициенты многочлена  $x^k g(x) \pmod{p(x)}$ . Допустим, что

$$x^{k-1} g(x) \pmod{p(x)} \equiv \sum_{l=0}^{n-1} h_{l, k-1} x^l.$$

Рассмотрим многочлен  $x^k g(x) \pmod{p(x)}$ . Имеем

$$x^k g(x) \pmod{p(x)} = x \{x^{k-1} g(x) \pmod{p(x)}\} \pmod{p(x)} \equiv x \left\{ \sum_{l=0}^{n-1} h_{l, k-1} x^l \right\} \pmod{p(x)} =$$

$$\sum_{l=0}^{n-1} h_{l, k-1} x^{l+1} \pmod{p(x)} = \sum_{l=0}^{n-2} h_{l, k-1} x^{l+1} + h_{n-1, k-1} x^n \pmod{p(x)} \equiv$$

$$\sum_{l=1}^{n-1} h_{l-1, k-1} x^l - h_{n-1, k-1} \sum_{l=0}^{n-1} p_l x^l = -h_{n-1, k-1} p_0 + \sum_{l=1}^{n-1} (h_{l-1, k-1} - h_{n-1, k-1} p_l) x^l,$$

т. е.  $(k+1)$ -й столбец матрицы  $H$  есть

$$\begin{pmatrix} -p_0 h_{n-1, k-1} \\ h_{0, k-1} - p_1 h_{n-1, k-1} \\ h_{1, k-1} - p_2 h_{n-1, k-1} \\ \vdots \\ h_{n-2, k-1} - p_{n-1} h_{n-1, k-1} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & 0 & -p_0 \\ 1 & 0 & 0 & \dots & 0 & 0 & -p_1 \\ 0 & 1 & 0 & \dots & 0 & 0 & -p_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \\ 0 & 0 & 0 & \dots & 1 & 0 & -p_{n-2} \\ 0 & 0 & 0 & \dots & 0 & 1 & -p_{n-1} \end{pmatrix} \begin{pmatrix} h_{0, k-1} \\ h_{1, k-1} \\ h_{2, k-1} \\ \vdots \\ h_{n-1, k-1} \end{pmatrix} = C_p(C_p^{k-1} \mathbf{g}) = C_p^k \mathbf{g}.$$

Таким образом, индукция завершена и формула (10) доказана.

**Теорема 4.** Пусть  $\mathbf{s} = H \mathbf{d}$  представляет собой матричную запись вычисления коэффициентов многочлена

$$s(x) = d(x) g(x) \pmod{p(x)}, \quad d(x), g(x) \in F[x],$$

где  $p(x)$  – неприводимый унитарный многочлен степени  $n$  над полем  $E \subset F$ , а матрица  $H$  составлена из коэффициентов многочлена  $g(x)$ . Тогда все элементы любой строки матрицы  $H$ , так же как и все элементы любой нетривиальной линейной комбинации строк матрицы  $H$ , линейно независимы.

**Доказательство.** Как следует из формулы (10),  $i$ -й столбец матрицы  $H$  равен  $C_p^{i-1} \mathbf{g}$ , где  $C_p$  – сопровождающая матрица многочлена  $p(x) = x^n + p_{n-1}x^{n-1} + \dots + p_1x + p_0$ , т. е. через свои вектор-столбцы матрица  $H$  записывается в виде

$$H = (\mathbf{g} \ C_p \mathbf{g} \ C_p^2 \mathbf{g} \ \dots \ C_p^{n-1} \mathbf{g}).$$

Пусть  $\omega$  – произвольная ненулевая вектор-строка элементов поля  $E$ , тогда

$$\omega H = (\omega \mathbf{g} \ \omega C_p \mathbf{g} \ \omega C_p^2 \mathbf{g} \ \dots \ \omega C_p^{n-1} \mathbf{g})$$

есть вектор-строка, являющаяся нетривиальной линейной комбинацией строк матрицы  $H$ . Требуется показать, что элементы этой строки (т. е. линейные формы коэффициентов многочлена  $g(x)$ ) линейно независимы как элементы линейного пространства  $E[g_0, g_1, \dots, g_{r-1}]$ . Предположим противное, найдутся такие константы  $a_0, a_1, \dots, a_{n-1}$  из поля  $E$ , среди которых хотя бы одна отлична от нуля, что

$$\sum_{i=0}^{n-1} a_i (\omega C_p^i \mathbf{g}) = 0.$$



Последнее равенство перепишем в виде

$$\mathbf{0} = \sum_{i=0}^{n-1} a_i (\omega C_p^i \mathbf{g}) = \left( \omega \sum_{i=0}^{n-1} a_i C_p^i \right) \mathbf{g}. \quad (11)$$

Равенство (11) должно выполняться для любого вектора  $\mathbf{g}$ , поэтому

$$\omega \left( \sum_{i=0}^{n-1} a_i C_p^i \right) = \mathbf{0}. \quad (12)$$

Пусть

$$a(x) = \sum_{i=0}^{n-1} a_i x^i, \quad \text{тогда} \quad a(C_p) = \sum_{i=0}^{n-1} a_i C_p^i$$

есть матрица размера  $n \times n$ , и (12) принимает вид

$$\omega a(C_p) = \mathbf{0}. \quad (13)$$

Поскольку  $\omega$  – ненулевая вектор-строка, то равенство (13) означает, что матрица  $a(C_p)$  является вырожденной (ее строки линейно зависимы).

Из формулы (9) следует, что

$$\det (C_p - \lambda E) = (-1)^{n+2} p(\lambda),$$

т. е. многочлен  $p(x)$  является характеристическим многочленом матрицы  $C_p$ , следовательно, по теореме Гамильтона – Кэли он аннулирует эту матрицу, т. е.

$$p(C_p) = \mathbf{0},$$

где  $\mathbf{0}$  – нулевая  $n \times n$  - матрица.

Поскольку  $a(C_p)$  – вырожденная матрица, то ее ядро содержит отличные от нуля векторы, пусть  $\mathbf{v} \neq \mathbf{0}$ ,  $\mathbf{v} \in \text{Ker } a(C_p)$ , т. е.  $a(C_p) \mathbf{v} = \mathbf{0}$ .

Рассмотрим многочлены  $a(x)$  и  $p(x)$ . По условию теоремы  $p(x)$  – неприводимый многочлен из кольца  $E[x]$ ,  $a(x)$  по построению многочлен из того же кольца степени, меньшей  $n$ . Значит, многочлены  $a(x)$  и  $p(x)$  взаимно просты, тогда по следствию из алгоритма Евклида (см. расширенный алгоритм Евклида для многочленов § 4 [4]) существуют такие многочлены  $u(x)$ ,  $v(x) \in E[x]$ , что

$$u(x) a(x) + v(x) p(x) = 1.$$

Переходя к многочленам от матриц, перепишем последнее равенство в виде

$$u(C_p) a(C_p) + v(C_p) p(C_p) = E$$

и применим к обеим частям полученного равенства вектор  $\mathbf{v}$ , тогда имеем

$$\{u(C_p) a(C_p) + v(C_p) p(C_p)\} \mathbf{v} = E \mathbf{v} = \mathbf{v}.$$

Но  $p(C_p) = \mathbf{0}$ , поэтому получаем

$$u(C_p) a(C_p) \mathbf{v} = \mathbf{v},$$

что противоречит выбору  $\mathbf{v}$ , так как  $a(C_p) \mathbf{v} = \mathbf{0}$ . Значит, не существует ненулевого многочлена  $a(x)$ ,  $\deg a(x) \leq n - 1$ , такого, что при подстановке в него матрицы  $C_p$  получаем вырожденную матрицу. Следовательно, все коэффициенты  $a_i$  должны быть нулевыми, т. е. из

$$\sum_{i=0}^{n-1} a_i (\omega C_p^i \mathbf{g}) = \mathbf{0}$$

следует, что  $a_i = 0$  ( $i = 0, 1, \dots, n - 1$ ), и линейные формы  $\omega \mathbf{g}$ ,  $\omega C_p \mathbf{g}$ ,  $\dots$ ,  $\omega C_p^{n-1} \mathbf{g}$  линейно независимы.



**Теорема 5.** Пусть  $p(x) \in E[x]$  является неприводимым многочленом степени  $n$ . Каждый алгоритм вычисления произведения многочленов  $d(x), g(x) \in F[x]$

$$s(x) = d(x)g(x) \pmod{p(x)}$$

содержит по меньшей мере  $2n - 1$  умножений.

*Доказательство.* Предположим, что алгоритм содержит  $t$  умножений. Тогда на выходе алгоритма получим линейные комбинации этих  $t$  членов-произведений, т. е.

$$\mathbf{s} = A \mathbf{S}, \quad (14)$$

где  $A$  является  $n \times t$ -матрицей над полем  $E$ , а  $\mathbf{S}$  – вектор длины  $t$ , компонентами которого являются члены-произведения.

Алгоритм должен решать задачу для любой пары многочленов  $d(x)$  и  $g(x)$ . Эти многочлены всегда можно выбрать так, чтобы только один коэффициент многочлена  $s(x)$  был ненулевым, а остальные были бы нулевыми. Следовательно, строки матрицы  $A$  должны быть линейно независимыми (иначе, если  $k$ -я строка есть линейная комбинация остальных, то невозможно подобрать  $d(x)$  и  $g(x)$  так, чтобы  $k$ -я компонента вектора  $\mathbf{s}$  была бы ненулевой, а остальные – нулевыми). Так как  $A$  – матрица над полем,  $n$  строк которой линейно независимы, то у этой матрицы должно быть  $n$  линейно независимых столбцов. Без ограничения общности будем считать, что первые  $n$  столбцов матрицы  $A$  линейно независимы, так что матрица  $A$  может быть разбита на блоки вида

$$A = (A' | A''),$$

где  $A'$  – невырожденная  $n \times n$ -матрица, а  $A''$  – матрица размера  $n \times (t - n)$ . Тогда (14) можно записать в виде

$$\mathbf{s} = (A' | A'') \mathbf{S}. \quad (15)$$

Пусть  $C = (A')^{-1}$ . Умножим обе части (15) слева на  $C$ :

$$C \mathbf{s} = (E | P) \mathbf{S},$$

где  $P = C A''$ . Так как  $\mathbf{s} = H \mathbf{d}$ , то  $C \mathbf{s} = C H \mathbf{d}$ , откуда

$$(E | P) \mathbf{S} = C H \mathbf{d}.$$

Многочлен  $p(x)$  неприводим в кольце  $E[x]$ , значит, по теореме 4 все элементы любой строки матрицы  $H$  линейно независимы, равно как и все элементы любой нетривиальной линейной комбинации строк матрицы  $H$ . Следовательно, все элементы первой строки матрицы  $C H$  линейно независимы (значит, среди них не может быть ни одного нулевого), поэтому для вычисления первой компоненты вектора  $C \mathbf{s} = C H \mathbf{d}$  требуется по меньшей мере  $n$  умножений.

С другой стороны, первая строка вычисления

$$C \mathbf{s} = (E | P) \mathbf{S}$$

содержит самое большее  $1 + (t - n)$  умножений, так как матрица  $P$  содержит  $t - n$  столбцов. Следовательно,

$$1 + (t - n) \geq n,$$

откуда получаем  $t \geq 2n - 1$ , что и требовалось доказать.

Из этой теоремы, в частности, следует, что для выполнения комплексного умножения необходимо не меньше трех вещественных умножений. Действительно, на комплексное умножение можно смотреть как на умножение по модулю многочлена  $x^2 + 1$ . Это неприводимый над  $\mathbb{R}$  (и над  $\mathbb{Q}$ ) многочлен второй степени.

Рассмотрим случай, когда многочлен  $p(x)$  раскладывается в произведение  $k$  попарно взаимно простых неприводимых многочленов над кольцом  $E[x]$

$$p(x) = p_1(x) p_2(x) \dots p_k(x).$$



В этом случае китайская теорема об остатках позволяет разбить задачу на  $k$  подзадач

$$s_i(x) = d(x) g(x) \pmod{p_i(x)}, \quad i = 1, 2, \dots, k.$$

Каждая такая подзадача по теореме 5 требует не менее, чем  $2n_i - 1$  умножений, где  $n_i = \deg p_i(x)$ . Дополнительных умножений при разбиении на подзадачи не возникает, поэтому для полной задачи вычисления произведения многочленов по модулю многочлена  $p(x)$  потребуется умножений не меньше, чем

$$\sum_{i=1}^k (2n_i - 1) = 2 \sum_{i=1}^k n_i - k = 2n - k,$$

где  $n = \sum_{i=1}^k n_i = \deg p(x)$ .

Следующая теорема утверждает, что ни один алгоритм решения этой задачи не может быть лучше такого использования китайской теоремы об остатках.

**Теорема 6.** Пусть многочлен  $p(x)$  степени  $n$  раскладывается в произведение  $k$  попарно взаимно простых неприводимых многочленов из кольца  $E[x]$ . Каждый алгоритм вычисления

$$s(x) = d(x) g(x) \pmod{p(x)}$$

содержит не менее, чем  $2n - k$  умножений.

*Доказательство.* Пусть соответствующая китайской теореме об остатках  $i$ -я подзадача ( $i = 1, 2, \dots, k$ ) задается вычислением

$$\tilde{s}_i = H_i \tilde{d}_i,$$

где  $\tilde{d}_i$  — вектор коэффициентов многочлена  $d(x) \pmod{p_i(x)}$ . Далее выходной вектор  $s$  исходной задачи собирается из компонент выходных векторов подзадач, т. е. компоненты вектора  $s$  равны линейным комбинациям компонент векторов  $\tilde{s}_i$  ( $i = 1, 2, \dots, k$ ):

$$\begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{n-1} \end{pmatrix} = B \begin{pmatrix} \tilde{s}_1 \\ \tilde{s}_2 \\ \vdots \\ \tilde{s}_k \end{pmatrix},$$

где  $B$  — матрица над полем  $E$ . Этот этап не содержит умножений, вычисления векторов  $\tilde{d}_i$  также не содержат умножений. Поэтому достаточно рассмотреть вычисление выходных векторов подзадач

$$\tilde{s} = H \tilde{d}, \tag{16}$$

где  $H$  — блочно-диагональная матрица, т. е.

$$\begin{pmatrix} \tilde{s}_1 \\ \tilde{s}_2 \\ \vdots \\ \tilde{s}_k \end{pmatrix} = \begin{pmatrix} H_1 & O & \dots & O \\ O & H_2 & \dots & O \\ \vdots & \vdots & \ddots & \vdots \\ O & O & \dots & H_k \end{pmatrix} \begin{pmatrix} \tilde{d}_1 \\ \tilde{d}_2 \\ \vdots \\ \tilde{d}_k \end{pmatrix}.$$

Теперь повторим доказательство теоремы 5. Предположим, что алгоритм решения задачи (16) содержит  $t$  умножений. Тогда

$$\tilde{s} = A S,$$

где  $S$  — вектор длины  $t$ , содержащий все члены-произведения, и  $A$  — матрица размера  $n \times t$  над полем  $E$ . Так как  $A$  содержит  $n$  линейно независимых строк, то она должна содержать и  $n$  линейно независимых столбцов, в качестве которых можно выбрать первые  $n$  столбцов. Тогда

$$\tilde{s} = (A' / A'') S, \tag{17}$$

где  $A'$  – невырожденная  $n \times n$ -матрица,  $A''$  – матрица размера  $n \times (t - n)$ . Пусть  $C = (A')^{-1}$ , тогда (17) можно переписать в виде

$$C\tilde{s} = (E | CA'') S = (E | P) S.$$

Следовательно, для вычисления первой компоненты вектора  $C\tilde{s}$  требуется самое большое  $1 + (t - n)$  умножений. Кроме того, из (16) следует, что

$$C\tilde{s} = C H \tilde{d}.$$

Рассмотрим теперь некоторую линейную комбинацию строк матрицы  $H$ . Любая нетривиальная линейная комбинация строк каждой из матриц  $H_i$  содержит лишь линейно независимые столбцы. Поэтому любая линейно независимая комбинация строк матрицы  $H$  содержит по меньшей мере  $n - (k - 1)$  линейно независимых столбцов, т. е. ранг по столбцам матрицы  $CH$  не меньше, чем  $n - (k - 1)$ .

Действительно, пусть в разложении  $p(x)$  на множители имеем

$$\deg p_1(x) = n - (k - 1), \quad \deg p_2(x) = \dots = \deg p_k(x) = 1,$$

т. е.  $H_1$  есть квадратная матрица порядка  $n - (k - 1)$ , тогда в первой строке  $CH$  получаем не меньше, чем  $n - (k - 1)$  линейно независимых элементов.

Тогда вычисление первой компоненты вектора  $CH \tilde{d}$  требует не меньше, чем  $n - (k - 1)$  умножений. Итак, получены верхняя  $1 + (t - n)$  и нижняя  $n - (k - 1)$  границы числа умножений, необходимых для вычисления первой компоненты вектора  $C\tilde{s}$ , значит,

$$1 + (t - n) \geq n - (k - 1),$$

откуда  $t \geq 2n - k$ .

## ЛИТЕРАТУРА

1. Ноцен, П. Алгебраическая алгоритмика/П. Ноцен, К. Китте.– М.: Мир, 1999.
2. Акритас, А. Основы компьютерной алгебры с приложениями/А. Акритас.– М.: Мир, 1994.
3. Яблокова, С. И. Основы алгебраической алгоритмики: Учеб. пособие. Ч.1/С. И. Яблокова.– Ярославль: ЯрГУ, 2007.– 120с.
4. Яблокова, С. И. Основы алгебраической алгоритмики: Учеб. пособие. Ч.2/С. И. Яблокова.– Ярославль: ЯрГУ, 2008. – 126с.
5. Блейхут, Р. Э. Быстрые алгоритмы цифровой обработки сигналов/Р. Э. Блейхут.– М.: Мир, 1979.



## ОГЛАВЛЕНИЕ

Предисловие	3
История быстрых алгоритмов и их приложения	4
§ 1. Понятие быстрого алгоритма. Цифровые фильтры	6
§ 2. Линейная и циклическая свертки	9
§ 3. Алгоритм Кука – Тоома	13
§ 4. Алгоритмы Винограда вычисления коротких сверток	17
§ 5. Примеры построения алгоритмов коротких линейных сверток	25
§ 6. Построение алгоритмов коротких циклических сверток	31
§ 7. Теорема об обмене матриц	40
§ 8. Дискретное преобразование Фурье. Теорема о свертке	43
§ 9. Дискретное преобразование Фурье и циклическая свертка	46
§ 10. Дискретное преобразование Фурье и многочлены	49
§ 11. Алгоритм Кули – Тьюки	52
§ 12. Алгоритм Кули – Тьюки по основанию два	55
§ 13. Алгоритм Кули – Тьюки по основанию четыре	60
§ 14. Алгоритм Гуда – Томаса быстрого преобразования Фурье	65
§ 15. Алгоритм Рейдера в случае, когда длина преобразования равна простому числу	69
§ 16. Алгоритм Рейдера в случае, когда длина преобразования равна степени простого нечетного числа	72
§ 17. Алгоритм Рейдера в случае, когда длина преобразования равна степени двойки	74
§ 18. Алгоритм Винограда для быстрого преобразования Фурье в случае, когда длина преобразования есть простое число	79
§ 19. Алгоритм Винограда для быстрого преобразования Фурье в случае, когда длина преобразования есть степень простого числа	82
§ 20. Алгоритм Винограда для быстрого преобразования Фурье в случае, когда длина преобразования есть степень двойки	87
§ 21. Свертка в конечном поле	96
§ 22. Преобразования Фурье в полях $GF(2^m + 1)$ (преобразования Ферма)	107
§ 23. Преобразования Фурье в полях $GF(2^m - 1)$ (преобразования Мерсенна)	110
§ 24. Алгоритм Агарвала – Кули вычисления свертки	113
§ 25. Сложность алгоритмов свертки	125
ЛИТЕРАТУРА	135