Обработка сигналов. Лекция 5. Быстрое преобразование Фурье.

9 ноября 2021 г.

1 Введение

Хотя ДПФ представляет собой наиболее простую математическую процедуру определения частотного состава временных последовательностей, оно ужасно неэффективно. При увеличении количества точек ДПФ до сотен и тысяч количество выполняемых операций превышает все разумные пределы. В 1965 году была опубликована статья Кули и Тьюки, описывающая очень эффективный алгоритм реализации ДПФ. Этот алгоритм сегодня известен как быстрое преобразование Фурье (БПФ).

В действительности БПФ имеет интересную историю. Исследуя рассеяние рентгеновских лучей, два физика в 1940-х годах воспользовались преимуществами симметрии синусов и косинусов, применив математический метод, основанный на опубликованном в начале 1900-х приеме. Прошло более 20 лет, прежде чем алгоритм БПФ был открыт заново.

До появления БПФ ДПФ длиной в несколько тысяч точек требовало так много времени, что его применение ограничивалось крупными исследовательскими и университетскими вычислительными центрами. Благодаря Кули и Тьюки, а также полупроводниковой промышленности сегодня 1024-точечное ДПФ может быть вычислено за несколько секунд на домашнем компьютере.

О БПФ написаны тома, и, как никакое другое изобретение, разработка этого алгоритма изменила цифровую обработку сигналов, сделав доступной всю мощь анализа Фурье. В этой лекции мы покажем, почему наиболее популярные алгоритмы БПФ (которые называют БПФ по основанию 2) имеют преимущества над классическим алгоритмом ДПФ, дадим ряд рекомендаций, позволяющих получить больше практической пользы от БПФ.

2 Связь БПФ с ДПФ

Существует ряд различных алгоритмов БПФ. В этом разделе мы увидим, почему алгоритм БПФ по основанию 2 так популярен, и узнаем, как он связан с классическим ДПФ. Алгоритм БПФ по основанию 2 — это очень

эффективный алгоритм вычисления ДПФ, когда длина ДПФ равна целой степени двух. (То есть количество точек преобразования равно $N = 2^k$, где k - некоторое положительное число.) Посмотрим, почему практикующие специалисты по ЦОС предпочитают БПФ по основанию 2.

Вспомните пример из предыдущей лекции, в котором мы отметили некоторое количество избыточных операций при выполнении 8-точечного ДПФ. (Например, мы вычисляли произведение $1.0607 \cdot 0.707$ четыре раза.) БПФ по основанию 2 устраняет эту избыточность и существенно снижает количество необходимых арифметических операций. Чтобы оценить эффективность БПФ, рассмотрим количество комплексных умножений, необходимых для выполнения уже знакомого нам N-точечного ДПФ.

$$X(m) = \sum_{n=0}^{N-1} x(n)e^{-2\pi nm/N} \tag{1}$$

В случае 8-точечного ДПФ из (1) следует, что нам необходимо выполнить N^2, или 64, комплексных умножений. (Потому что для каждого из восьми отсчетов $X(m)$ мы должны просуммировать восемь комплексных произведений при изменении n от 0 до 7.) Количество комплексных умножений для N-точечного БПФ равно примерно

$$(N/2) \cdot \log_2 N \tag{2}$$

(Мы говорим "примерно", потому что некоторые умножения выполняются на $+1$ или -1 и сводятся к простой перемене знака.) Величина $(N/2)\log_2 N$ указывает на значительное снижение количества комплексных умножений по сравнению с N^2 операций, необходимых для реализации (1), особенно при больших N. Насколько значительно это снижение, показывает рисунок 1, на котором сравнивается количество комплексных умножений, необходимых для реализации ДПФ и БПФ по основанию 2 при разных значениях N. При $N = 512$, например, ДПФ требует в 200 раз больше комплексных умножений, чем БПФ. При $N = 8192$ ДПФ требует 1000 комплексных умножений на *каждое* комплексное умножение в алгоритме БПФ! Публикация и распространение алгоритма БПФ по основанию 2 были, наверное, самыми важными событиями в цифровой обработке сигналов.

Здесь уместно подчеркнуть, что БПФ не является аппроксимацией ДПФ. Это в точности ДПФ. Более того, все характеристики ДПФ, описанные в предыдущей лекции: симметрия, линейность, гребешковые искажения и прочие, — присущи также и БПФ.

3 Советы по практическому использованию БПФ

Учитывая ценность БПФ для практики, мы приводим здесь ряд практических указаний по накоплению данных и использованию БПФ по основанию 2 для анализа сигналов реального мира.

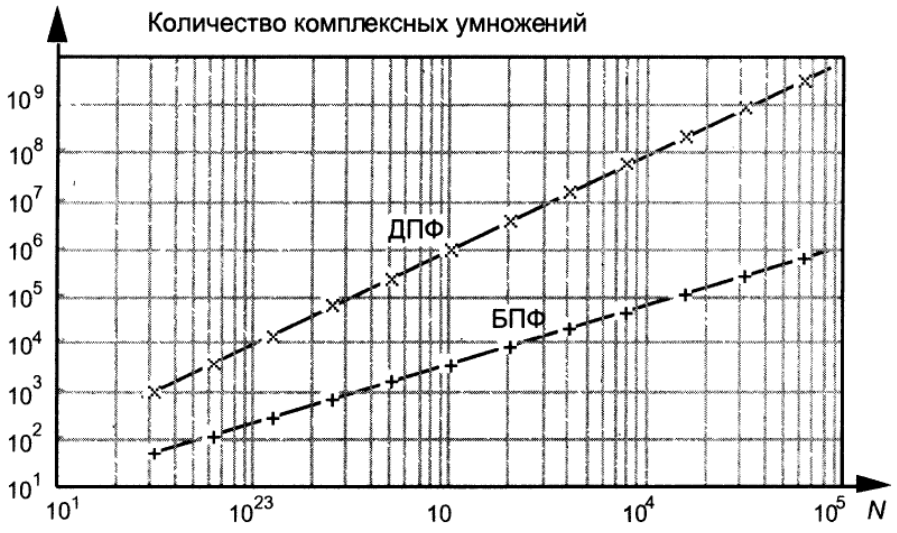

Рис. 1: Количество комплексных умножений, как функция N, при реализации ДПФ и БПФ по основанию 2

3.1 Дискретизируйте достаточно часто и достаточно долго

Как мы знаем, при преобразовании непрерывных сигналов в цифровую форму с помощью АЦП частота дискретизации должна быть не менее чем в два раза больше ширины спектра непрерывного сигнала, чтобы предотвратить наложения в частотной области. На практике в зависимости от приложения используют частоту дискретизации, которая в два с половиной — четыре раза превышает ширину спектра сигнала. Если мы знаем, что ширина спектра преобразуемого сигнала не очень велика по сравнению с максимальной частотой преобразования нашего АЦП, от наложений легко избавиться. Если мы не знаем, какова ширина спектра непрерывного сигнала, как нам определить, имеются ли наложения или нет? В этом случае нам следует с недоверием относиться к результатам БПФ, если имеются спектральные компоненты значительного уровня на частотах вблизи половины частоты дискретизации. В идеале нам хотелось бы работать с сигналами, спектр которых убывает с ростом частоты. Будьте очень осторожны, когда в спектре обнаруживаются компоненты, частоты которых зависят от частоты дискретизации. Если у нас есть подозрение, что возникает наложение или что непрерывный сигнал содержит широкополосный шум, необходимо перед АЦП включить аналоговый фильтр низких частот (ФНЧ). Частота среза ФНЧ должна быть несколько выше максимальной интересующей нас частоты и ниже половины частоты дискретизации.

Хотя мы знаем, что БПФ по основанию 2 требует $N = 2^k$ входных

отсчётов, сколько же именно отсчетов должны мы накопить для анализа? Ответ состоит в том, что интервал времени, на котором выполняется накопление, должен соответствовать требуемой разрешающей способности БПФ при заданной частоте дискретизации f_s. Длительность интервала накопления данных является величиной, обратной требуемой разрешающей способности, и чем больше мы берем отсчётов с фиксированной частотой дискретизации f_s, тем выше разрешение по частоте; т. е. общий интервал накопления равен N/f_s секунд, а разрешающая способность N-точечного БПФ равна f_s/N Гц. Следовательно, если нам необходима разрешающая способность 5 Гц, то $f_s/N = 5$ Гц и

$$N = f_s/(\text{требуемое разрешение}) = f_s/5 = 0.2f_s. \tag{3}$$

В этом случае, если f_s равна, скажем, 10 кГц, то N должно быть не меньше 2000, и нам придется взять N = 2048, потому что это ближайшее число, равное целой степени 2.

3.2 Предварительная обработка данных

Если при использовании БПФ по основанию 2 мы не можем повлиять на количество получаемых отсчетов, и длина последовательности не равна целой степени двойки, можно применить один из двух подходов. Мы можем отбросить часть отсчётов данных так, чтобы длина оставшейся последовательности была равна целой степени двойки. Использование этого варианта не рекомендуется, т. к. игнорирование части отсчетов данных приведет к ухудшению разрешения по частоте. (Чем больше N, тем лучше разрешение, не правда ли?) Лучше дополнить последовательность нулями так, чтобы получить необходимое общее количество отсчётов. Например, если у нас есть 1000 отсчетов, которые необходимо преобразовать, вместо того, чтобы анализировать 512 из этих отсчётов, нам следует добавить в конец последовательности 24 нулевых отсчета и использовать 1024-точечное БПФ.

3.3 Улучшение результатов БПФ

Если мы используем БПФ для обнаружения энергии сигнала в присутствии шума, и у нас есть достаточно данных, мы можем улучшить чувствительность нашего обнаружителя, применяя усреднение результатов нескольких БПФ. Этот метод может быть использован для обнаружения сигналов, мощность которых ниже среднего уровня шума. Другими словами, имея достаточно данных во временной области, мы можем обнаруживать компоненты сигнала при отрицательном отношении сигнал/шум.

Если данные во временной области имеют действительные значения, мы можем воспользоваться преимуществами метода $2N$-точечного действительного БПФ для ускорения обработки; т. е. $2N$-точечная действительная последовательность может быть преобразована с помощью одного N-точечного комплексного БПФ по основанию 2. Таким образом, мы можем

получить частотное разрешение $2N$-точечного БПФ по цене стандартного N-точечного БПФ. Другая возможность ускорения БПФ состоит в применении взвешивания окном в частотной области. Если нам нужно БПФ невзвешенных данных и одновременно мы хотим иметь БПФ тех же данных, взвешенных окном, нам нет необходимости выполнять два разных БПФ. Мы можем вычислить БПФ невзвешенных данных, а затем выполнить взвешивание в частотной области, чтобы уменьшить утечку на всех или на некоторых бинах БПФ.

3.4 Интерпретация результатов БПФ

Первый шаг при интерпретации результатов БПФ состоит в вычислении абсолютных частот центров бинов. Как и в случае ДПФ, расстояние по частоте между бинами БПФ равно частоте дискретизации f_s, деленной на количество точек БПФ, или f_s/N. Обозначим результат БПФ как $X(m)$, где $m = 0, 1, 2, \ldots, N-1$, при этом абсолютная частота центра m-го бина равна mf_s/N. Если отсчеты входной последовательности БПФ действительны, только отсчеты $X(m)$ с индексами от $m = 0$ до $m = N/2$ независимы. Следовательно, в этом случае нам необходимо определить только частоты бинов для m в диапазоне $0 \leq m \leq N/2$. Если же входные отсчеты комплексные, все N отсчетов БПФ независимы, и нам потребуется вычислить частоты бинов для m во всем диапазоне $0 \leq m \leq N-1$.

Если необходимо, мы можем определить истинные амплитуды сигналов по их спектрам. Для этого мы должны помнить, что выходные отсчеты БПФ по основанию 2 - комплексные величины вида

$$X(m) = X_{\text{real}}(m) + jX_{\text{imag}}(m). \tag{4}$$

А все модули отсчетов БПФ

$$X_{\text{mag}}(m) = |X(m)| = \sqrt{X_{\text{real}}(m)^2 + jX_{\text{imag}}(m)^2}. \tag{5}$$

в результате преобразования, когда входные отсчеты действительны, оказываются умноженными на $N/2$, как сообщалось в предыдущей лекции. Если же входные отсчеты комплексные, масштабирующий коэффициент равен N. Следовательно, чтобы определить правильные амплитуды синусоидальных составляющих, нам необходимо разделить модули отсчетов БПФ на соответствующий коэффициент: $N/2$ для действительных последовательностей и N для комплексных последовательностей.

В случае, когда мы хотим определить спектр мощности $X_{\text{ps}}(m)$, нам необходимо вычислить квадрат модуля БПФ, используя соотношение

$$X_{\text{ps}}(m) = |X(m)|^2 = X_{\text{real}}^2 + X_{\text{imag}}(m)^2. \tag{6}$$

Это позволит нам вычислить спектр мощности в логарифмическом масштабе

$$X_{\text{dB}}(m) = 10 \cdot \log_{10}(|X(m)|^2) \text{ дБ} \tag{7}$$

Нормированный спектр мощности в логарифмическом масштабе можно вычислить с помощью выражения

$$\text{норм.}X_{\text{dB}}(m) = 10 \cdot \log_{10}(|X(m)|^2)/(|X(m)|_{\max})^2 \text{ дБ} \tag{8}$$

или

$$\text{норм.}X_{\text{dB}}(m) = 20 \cdot \log_{10}(|X(m)|)/(|X(m)|_{\max}) \text{ дБ} \tag{9}$$

В (8) и (9) член $|X(m)|_{\max}$ представляет собой максимальный модуль отсчета. На практике график $X_{\text{dB}}(m)$ очень информативен благодаря улучшенному разрешению составляющих низкого уровня при использовании логарифмического масштаба.

Зная, что фазы $X_\phi(m)$ отдельных отсчетов БПФ вычисляются по формуле

$$X_\phi(m) = tan^{-1}(X_{\text{imag}}(m)/X_{\text{real}}(m)), \tag{10}$$

мы должны особое внимание обратить на те отсчеты, для которых $X_{\text{real}}(m)$ равны нулю. В этом случае вычисление фазы по (10) невозможно из-за деления на ноль. На практике необходимо, чтобы программа расчета обнаруживала отсчеты, для которых $X_{\text{real}}(m) = 0$, и устанавливала $X_\phi(m) = 90°$ если $X_{\text{imag}}(m) > 0$, $X_\phi(m) = 0°$ если $X_{\text{imag}}(m) = 0$, и $X_\phi(m) = -90°$ если $X_{\text{imag}}(m) < 0$. Рассматривая фазовые углы отсчетов БПФ, имейте ввиду, что шумовые компоненты высокого уровня могут вызвать значительные флуктуации вычисленных $X_\phi(m)$. Это значит, что отсчеты $X_\phi(m)$ имеют смысл только тогда, когда соответствующий $|X(m)|$ намного превышает средний уровень шума.

4 Разработка алгоритма БПФ по основанию 2

Этот раздел содержит подробное описание внутренних структур данных и операций БПФ по основанию 2 для тех студентов, которые интересуются разработкой программ БПФ или проектированием аппаратурных процессоров БПФ.

Чтобы увидеть, как БПФ вырастает из ДПФ, напомним формулу N-точечного ДПФ:

$$X(m) = \sum_{n=0}^{N-1} x(n)e^{-j2\pi nm/N}. \tag{11}$$

Простой и понятный способ получения алгоритма БПФ состоит в том, что исходная последовательность данных $x(n)$ делится на две части. Выделив отсчеты $x(n)$ с четными и нечетными индексами в две отдельные последовательности, мы можем разбить (11) на две части вида

$$X(m) = \sum_{n=0}^{(N/2)-1} x(2n)e^{-j2\pi(2n)m/N} + \sum_{n=0}^{(N/2)-1} x(2n+1)e^{-j2\pi(2n+1)m/N}. \tag{12}$$

Вынося за знак второй суммы экспоненту с постоянным показателем степени, получаем

$$X(m) = \sum_{n=0}^{(N/2)-1} x(2n)e^{-j2\pi(2n)m/N} + e^{-j2\pi m/N} \sum_{n=0}^{(N/2)-1} x(2n+1)e^{-j2\pi(2n)m/N} \tag{13}$$

Для упрощения этих длинных и сложных выражений мы используем стандартные обозначения. Пусть $W_N = e^{-j2\pi/N}$ обозначает комплексный поворачивающий множитель, который постоянен для заданного N. Тогда (13) приобретает вид

$$X(m) = \sum_{n=0}^{(N/2)-1} x(2n)(W_N)^{2nm} + (W_N)^m \sum_{n=0}^{(N/2)-1} x(2n+1)(W_N)^{2nm}. \tag{14}$$

Поскольку $(W_N)^2 = e^{-j2\pi 2/N} = e^{-j2\pi/(N/2)}$ мы можем в формулу (14) подставить $W_{N/2}$ вместо $(W_N)^2$.

$$X(m) = \sum_{n=0}^{(N/2)-1} x(2n)(W_{N/2})^{nm} + (W_N)^m \sum_{n=0}^{(N/2)-1} x(2n+1)(W_{N/2})^{nm}. \tag{15}$$

Итак, мы имеем две суммы $N/2$ слагаемых, результаты которых можно объединить в N-точечное ДПФ. В (15) мы сократили количество операций над данными по сравнению с (11), потому что множители W в обеих суммах уравнения (15) идентичны. Дополнительный выигрыш от разбиения N-точечного ДПФ на две части мы получаем благодаря тому, что вторая половина отсчетов ДПФ вычисляется очень просто. Рассмотрим отсчет $X(m+N/2)$. Если мы подставим $m+N/2$ вместо m в (15), то

$$\begin{aligned} X(m+N/2) = &\sum_{n=0}^{(N/2)-1} x(2n)(W_{N/2})^{n(m+N/2)} + \\ &+(W_N)^{(m+N/2)} \sum_{n=0}^{(N/2)-1} x(2n+1)(W_{N/2})^{n(m+N/2)}. \end{aligned} \tag{16}$$

Кажется, мы усложнили нашу задачу? Ничего, осталось недолго. Мы можем упростить поворачивающие множители под знаками сумм, потому что

$$\begin{aligned} (W_{N/2})^{n(m+N/2)} = (W_{N/2})^{nm}(W_{N/2})^{nN/2} = (W_{N/2})^{nm}(e^{-j2\pi n2N/2N}) = \\ = (W_{N/2})^{nm}(1) = (W_{N/2})^{nm} \end{aligned} \tag{17}$$

для любого целого n. Внимательно проанализировав *поворачивающий множитель* перед второй суммой в (16), мы можем упростить его как

$$\begin{aligned} (W_N)^{(m+N/2)} = (W_N)^m(W_N)^{N/2} = (W_N)^m(e^{-j2\pi N/2N}) = \\ (W_N)^m(-1) = -(W_N)^m \end{aligned} \tag{18}$$

Далее, используя (17) и (18), мы представляем $X(m+N/2)$ из (16) как

$$X(m+N/2) = \sum_{n=0}^{(N/2)-1} x(2n)(W_{N/2})^{nm} - (W_N)^m \sum_{n=0}^{(N/2)-1} x(2n+1)(W_{N/2})^{nm}. \tag{19}$$

Теперь взгляните на (15) и (19), чтобы увидеть, насколько они схожи.

Вот мы и получили то, что хотели. Для вычисления $X(m+N/2)$ нам не нужно выполнять никаких умножений на синусы и косинусы. Мы просто изменяем знак поворачивающего множителя $(W_N)^m$ и используем две суммы, полученные для $X(m)$, для вычисления $X(m+N/2)$. Конечно, в этих выражениях m принимает значения от 0 до $(N/2)-1$, а это значит, что для N-точечного ДПФ мы выполняем два $N/2$-точечных ДПФ, получая первые $N/2$ отсчетов, а затем используем уже полученные результаты для вычисления последних $N/2$ отсчетов. При $N = 8$ (15) и (19) реализуются так, как показано на рисунке 2.

Если мы запишем (15) и (19) в более простой форме

$$X(m) = A(m) + (W_N)^m B(m), \tag{20}$$

$$X(m+N/2) = A(m) - (W_N)^m B(m), \tag{21}$$

мы можем пойти дальше и разбить два 4-точечных ДПФ на четыре 2-точечных ДПФ. Посмотрим, как можно разделить верхнее 4-точечное ДПФ на рисунке 2, четыре выхода которого равны $A(m)$ в (20) и (21). Мы можем разделить четные и нечетные входные отсчеты верхнего 4-точечного ДПФ:

$$A(m) = \sum_{n=0}^{(N/2)-1} x(2n)(W_{N/2})^{nm} = \tag{22}$$

$$= \sum_{n=0}^{(N/4)-1} x(4n)(W_{N/2})^{2nm} + \sum_{n=0}^{(N/4)-1} x(4n+2)(W_{N/2})^{(2n+1)m}$$

Поскольку $(W_{N/2})^{2nm} = (W_{N/4})^{nm}$, мы можем выразить $A(m)$ через два $N/4$-точечных ДПФ как

$$A(m) = \sum_{n=0}^{(N/4)-1} x(4n)(W_{N/4})^{nm} + (W_{N/2})^m \sum_{n=0}^{(N/4)-1} x(4n+2)(W_{N/4})^{nm} \tag{23}$$

Обратите внимание на сходство выражений (23) и (15). Эта возможность деления $N/2$-точечного ДПФ на два $N/4$-точечных ДПФ и сообщает алгоритму БПФ способность существенно уменьшать количество необходимых умножений при реализации ДПФ. (Мы это вскоре продемонстрируем на примере.) Выполняя те же шаги, которые мы выполняли для получения $A(m)$, можно показать, что $B(m)$ в (20) имеет вид

$$B(m) = \sum_{n=0}^{(N/4)-1} x(4n+1)(W_{N/4})^{nm} + (W_{N/2})^m \sum_{n=0}^{(N/4)-1} x(4n+3)(W_{N/4})^{nm}. \tag{24}$$

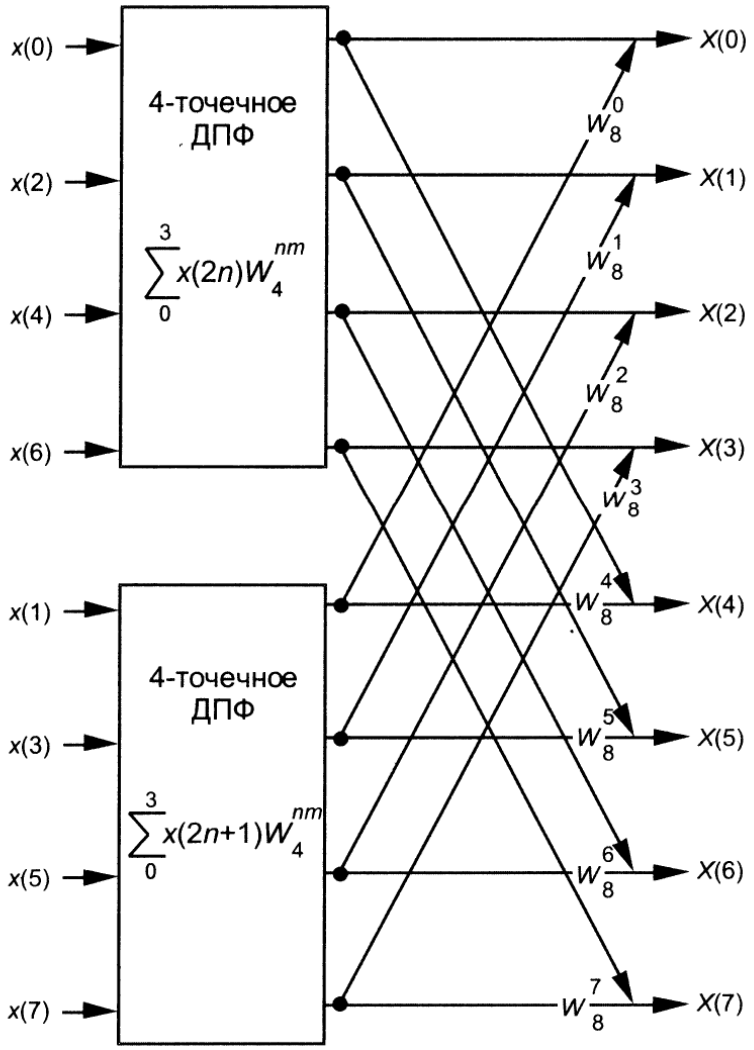

Рис. 2: Реализация БПФ для 8-точечного ДПФ с использованием двух 4-точечных ДПФ.

При $N = 8$ (23) и (24) реализуются так, как показано на рисунке 3. Здесь отчетливо видна структура сигнального графа, содержащая хорошо известные *бабочки*, и мы видим дальнейшую перетасовку входных данных. Поворачивающий множитель $(W_{N/2})^m$ в (23) и (24) в нашем примере с $N = 8$ изменяется от $(W_4)^0$ до $(W_4)^3$, потому что индекс m для $A(m)$ и $B(m)$ изменяется от 0 до 3. Для любого N-точечного ДПФ мы можем разбить каждое из $N/2$-точечных ДПФ на два $N/4$-точечных ДПФ с целью дальнейшего уменьшения количества умножений на синусы и косинусы. В конце концов мы дойдем до ряда 2-точечных ДПФ, после чего дальнейшее сокращение количества операций уже невозможно. Именно по этой причине количество точек БПФ может быть равно только целой степени двойки, а этот алгоритм БПФ называется БПФ по основанию 2.

Двигаясь в том же направлении, сделаем еще один шаг вперед и закончим с рассматриваемым $N = 8$-точечным ДПФ. Двухточечные ДПФ на рисунке 3 нельзя разделить на более мелкие части — мы дошли до конца в процессе уменьшения длины последовательности и пришли к бабочке одного 2-точечного ДПФ, показанной на рисунке 4. Из определения следует, что $(W_N)^0 = e^{-j2\pi 0/N} = 1$, и $(W_N)^{N/2} = e^{-j2\pi N/2N} = e^{-j\pi} = -1$. Следовательно, блоки 2-точечного ДПФ на рисунке 3 можно заменить бабочкой, показанной на рисунке 4, получив, таким образом, полную реализацию 8-точечного БПФ, показанную на рисунке 5.

Итак, мы прошли через изрядное количество алгебраических манипуляций. Чтобы проверить правильность нашего вывода алгоритма БПФ, мы можем взять 8-точечную последовательности из примера предыдущей лекции и обработать ее в соответствии с алгоритмом 8-точечного БПФ, представленным на рисунке 5. Последовательность данных, представляющая сигнал $x(n) = \sin(2\pi 1000nt_s) + 0.5\sin(2\pi 2000nt_s + 3\pi/4)$, имеет вид:

$$\begin{aligned} x(0) &= 0.3535 & x(1) &= 0.3535 \\ x(2) &= 0.6464 & x(3) &= 1.0607 \\ x(4) &= 0.3535 & x(5) &= -1.0607 \\ x(6) &= -1.3535 & x(7) &= -0.3535. \end{aligned} \tag{25}$$

Начнем перемалывать эти данные, наложив входные значения из (25) на рисунок 5, в результате чего получим значения, показанные в левой части рисунка 6. Выходные значения второго каскада БПФ равны

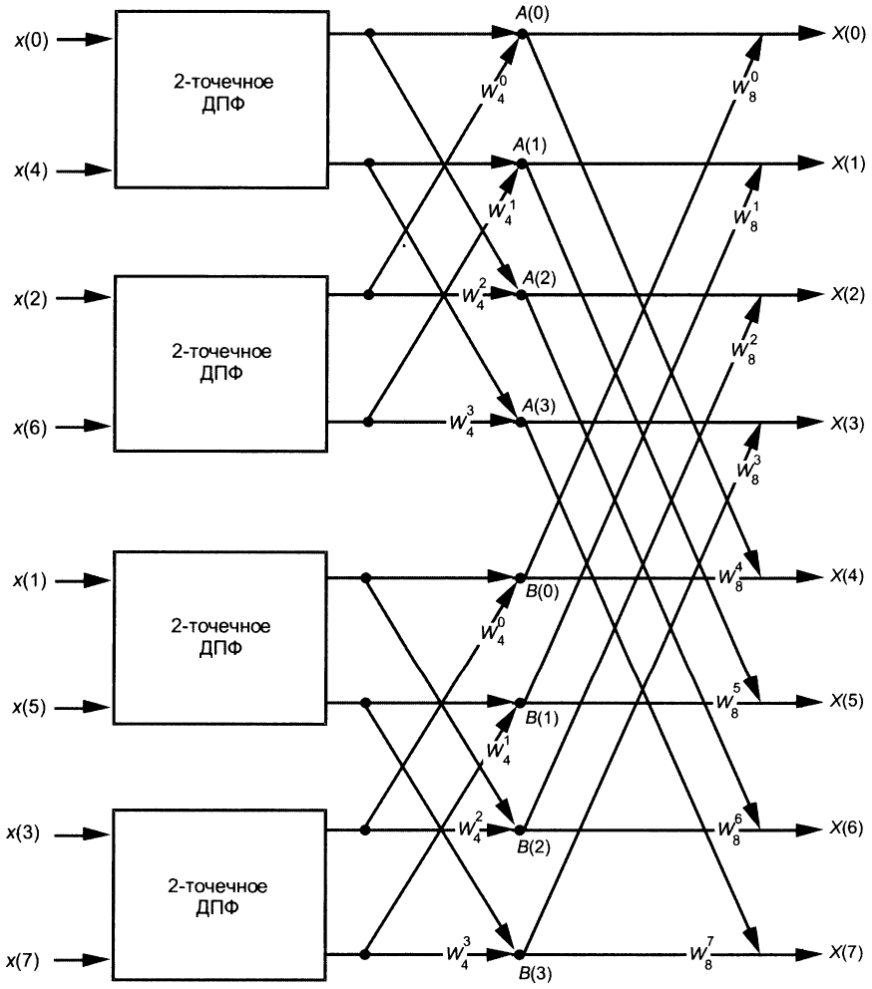

Рис. 3: Быстрая реализация 8-точечного ДПФ в виде двух 4-точечных ДПФ и четырех 2-точечных ДПФ

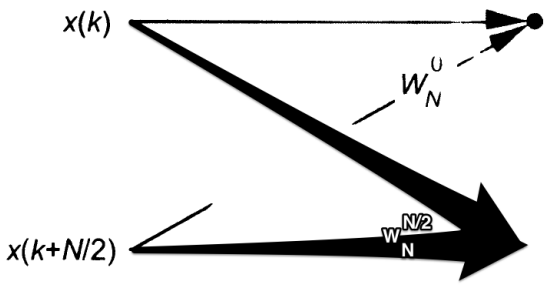

Рис. 4: Бабочка для одного 2-точечного ДПФ.

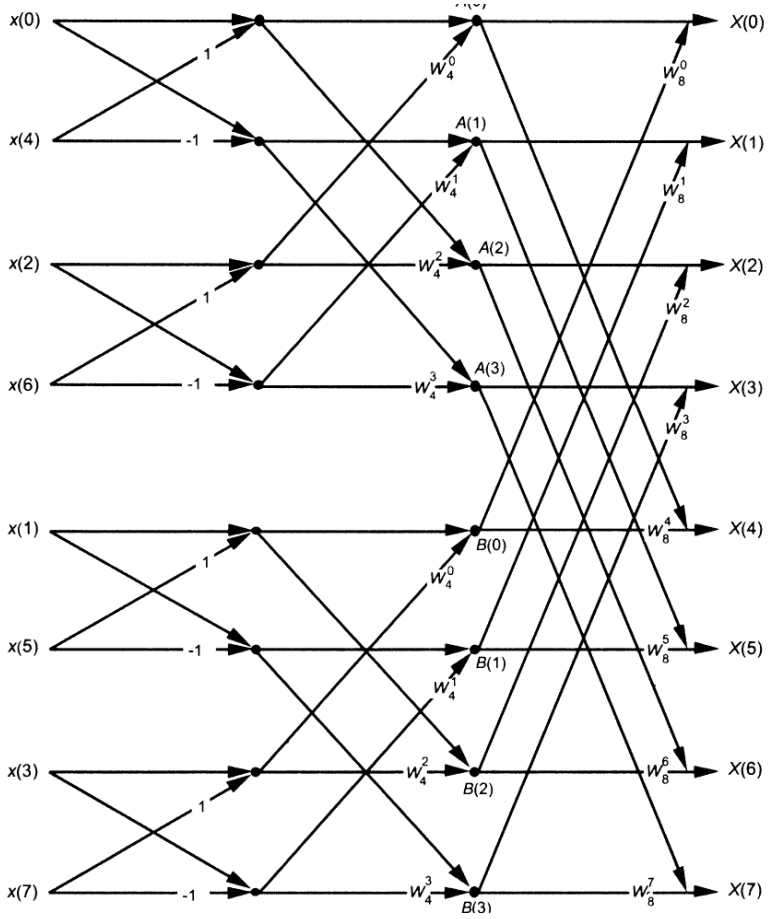

Рис. 5: Полная реализация алгоритма БПФ с прореживанием во времени для 8-точечного ДПФ.

$$A(0) = 0.707 + (W_4)^0(-0.707) = 0.707 + (1 + j0)(-0.707) = \ + j0,$$
$$A(1) = 0.0 + (W_4)^1 1.999) = 0.0 + (0 - j1)(1.999) = 0 - j1.999,$$
$$A(2) = 0.707 + (W_4)^2(-0.707) = 0.707 + (-1 + j0)(-0.707) = 1.414 + j0,$$
$$A(3) = 0.0 + (W_4)^3(1.999) = 0.0 + (0 + j1)(1.999) = \ + j1.999,$$
$$B(0) = 0.707 + (W_4)^0(0.707) = -0.707 + (1 + j0)(0.707) = 0 + j0,$$
$$B(1) = 1.414 + (W_4)^1(1.414) = 1.414 + (0 - j1)(1.414) = 1.414 - j1.414,$$
$$B(2) = -0.707 + (W_4)^2(0.707) = -0.707 + (-1 + j0)(0.707) = -1.414 + j0,$$
$$B(3) = 1.414 + (W_4)^3(1.414) = 1.414 + (0 + j1)(1.414) = 1.414 + j1.414.$$

Вычисляя результат третьего каскада БПФ, получаем окончательный ответ

$$X(0) = A(0) + (W_8)^0 B(0) = 0 + j0 + (1 + j0)(0 + j0) = 0 + j0 + 0 + j0 = 0\angle 0^\circ$$
$$X(1) = A(1) + (W_8)^1 B(1) = 0 - j1.999 + (0.707 - j0.707)(1.414 - j1.414) =$$
$$= 0 - j1.999 + 0 - j1.999 = 0 - j4 = 4\angle -90^\circ$$
$$X(2) = A(2) + (W_8)^2 B(2) = 1.414 + j0 + (0 - 1j)(-1.414 + j0) =$$
$$= 1.414 + j0 + 0 + j1.4242 = 1.414 + j1.414 = 2\angle 45^\circ$$
$$X(3) = A(3) + (W_8)^3 B(3) = 0 + j1.999 + (-0.707 - j0.707)(1.414 + j1.414) =$$
$$= 0 + j1.999 + 0 - j1.999 = 0\angle 0^\circ$$
$$X(4) = A(0) + (W_8)^4 B(0) = 0 + j0 + (-1 + j0)(0 + j0) =$$
$$= 0 + j0 + 0 + j0 = 0\angle 0^\circ$$
$$X(5) = A(1) + (W_8)^5 B(1) = 0 - j1.999 + (-0.707 + j0.707)(1.414 - j1.414) =$$
$$= 0 - j1.999 + 0 + j1.999 = 0\angle 0^\circ$$
$$X(6) = A(2) + (W_8)^6 B(2) = 1.414 + j0 + (0 + j1)(-1.414 + j0) =$$
$$= 1.414 + j0 + 0 - j1.414 = 1.414 - j1.414 = 2\angle -45^\circ$$
$$X(7) = A(3) + (W_8)^7 B(3) = 0 + j1.999 + (0.707 + j0.707)(1.414 + j1.414) =$$
$$= 0 + j1.999 + 0 + j1.999 = 0 + j4 = 4\angle 90^\circ.$$

Итак, к счастью, БПФ дает правильный результат, и это позволяет нам снова напомнить, что БПФ не является аппроксимацией ДПФ, а представляет собой ДПФ с уменьшенным количеством арифметический операций. В приведенном выше примере вы могли видеть, что вычисление 8-точечного БПФ требует меньших затрат, чем 8-точечное ДПФ из примера в прошлой лекции. Некоторые специалисты предпочитают объяснять это уменьшение количества арифметических операций избыточностью поворачивающих множителей . Они показывают это с помощью звездной диаграммы, приведенной на рисунке 7, которая демонстрирует эквивалентность некоторых поворачивающих множителей в 8-точечном ДПФ.

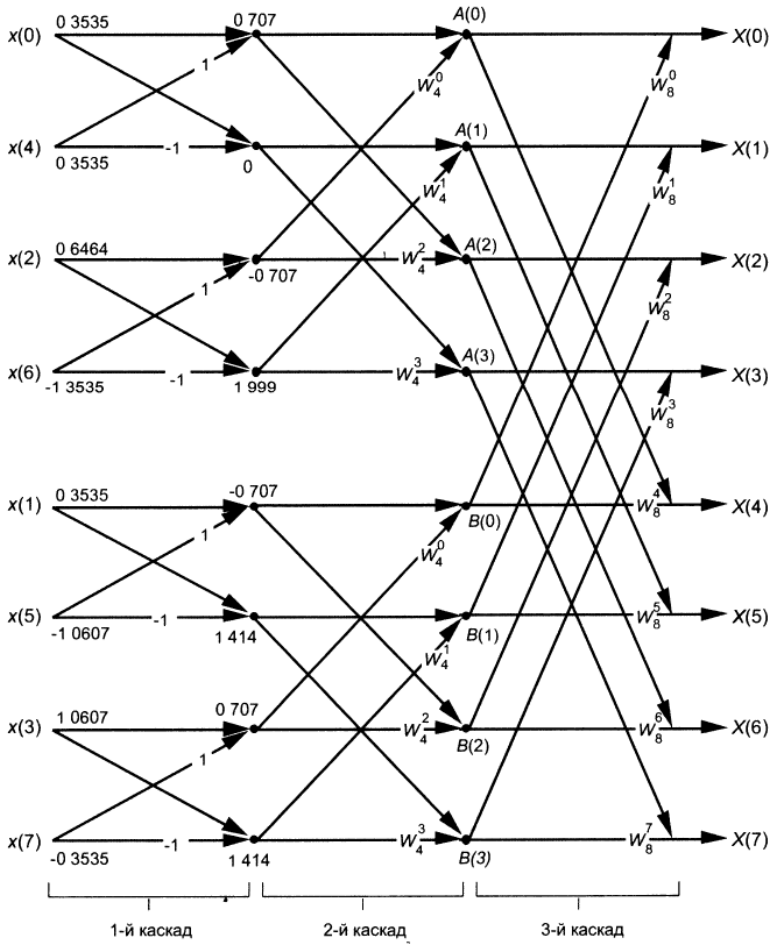

Рис. 6: 8-точечное БПФ последовательности примера предыдущей лекции.

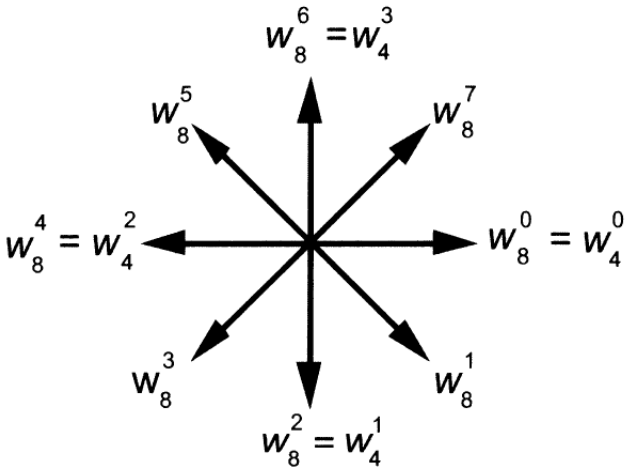

Рис. 7: Избыточность поворачивающих множителей для 8-точечного БПФ.

5 БИТ-реверсивная перестановка входных и выходных данных БПФ

Рассмотрим некоторые особые свойства БПФ, которые имеют большое значение для разработчиков программ и аппаратурных процессоров БПФ. Заметим, что рисунок 5 называется “Полная реализация БПФ с прореживанием по времени для 8-точечного ДПФ”. Слова с прореживанием по времени относятся к способу разбиения входной последовательности отсчетов на четные и нечетные, который мы использовали при выводе соотношений 15, (23) и (24). Такое прореживание приводит к перемешиванию входных отсчетов на рисунке 5. Принцип перемешивания можно понять с помощью таблицы 8. Это перемешивание называют бит-реверсивным, потому что порядок следования отсчетов в перемешанной последовательности можно получить путем изменения порядка следования битов в двоичном представлении индексов отсчётов в неперемешанной последовательности на обратный.

Это звучит несколько запутанно, но на самом деле все просто — таблица 8 иллюстрирует реверс битов для рассматриваемого примера 8-точечного БПФ. Обратите внимание на то, что индексы в левом столбце таблицы 8 расположены в нормальном порядке, а в правом столбце индексы приведены в перемешанном порядке, который совпадает с окончательным порядком следования прореженных входных отсчетов на рисунке 5. Мы перевернули биты двоичного представления нормальных индексов, поменяв их порядок следования на обратный. Старший бит стал младшим, а младший стал старшим, следующий за старшим бит стал следующим за младшим, и т. д.

Нормальный порядок индексов *n*	Двоичное представление индекса *n*	Перестановка битов индекса *n*	Бит-реверсивный порядок индекса *n*
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

Рис. 8: Бит-реверсивная перестановка входных отсчетов 8-точечного БПФ.

6 Структуры бабочек БПФ по основанию 2

Исследуем подробнее сигнальный граф бабочки БПФ с прореживанием по времени. Чтобы упростить сигнальный граф, заменим на рисунке 5 поворачивающие множители их эквивалентными значениями, обозначенными как $(W_N)^m$, где $N = 8$. Мы можем показывать только показатели степени m множителей $(W_N)^m$, чтобы получить структуру БПФ, показанную на рисунке 9. Например, $(W_4)^1$ на рисунке 5 равен $(W_8)^2$ и на рисунке 9 показан как 2, $(W_4)^2$ на рисунке 5 равен $(W_8)^4$ и на рисунке 9 показан как 4 и т. д. Значения 1 и -1 в первом каскаде рисунка 5 на рисунке 9 заменены на 0 и 4 соответственно. За исключением обозначения поворачивающих множителей, рисунок 9 идентичен рисунку 5. Мы можем перетасовать на рисунке 5 сигнальные узлы и получить 8-точечное БПФ с прореживанием по времени, показанное на рисунке 10. Заметьте, что входные данные на рисунке 10 расположены в нормальном порядке, а биты индексов выходных отсчетов переставлены в обратном порядке. В этом случае необходимо выполнять операцию реверсии порядка битов над выходными отсчетами БПФ, чтобы представить частотные отсчеты в нормальном порядке.

На рисунке 11 показана структура сигнального графа БПФ, которая не требует перестановки вообще, но элегантное переплетение линий обычной бабочки превратилось в запутанную, хоть и эффективную, конфигурацию.

Не так давно большая часть времени при аппаратурной реализации БПФ тратилась на выполнение умножений, а процесс бит-реверсивной перестановки, необходимый для доступа к данным в памяти, не составлял значительной части в общем объеме вычислений. Теперь, когда аппаратурные умножители-аккумуляторы в интегральном исполнении могут умножать два числа за один такт синхросигнала, мультиплексирование и адресация данных БПФ приобретают большее значение. Это стимулировало разработку ряда эффективных алгоритмов бит-реверсивной перестановки.

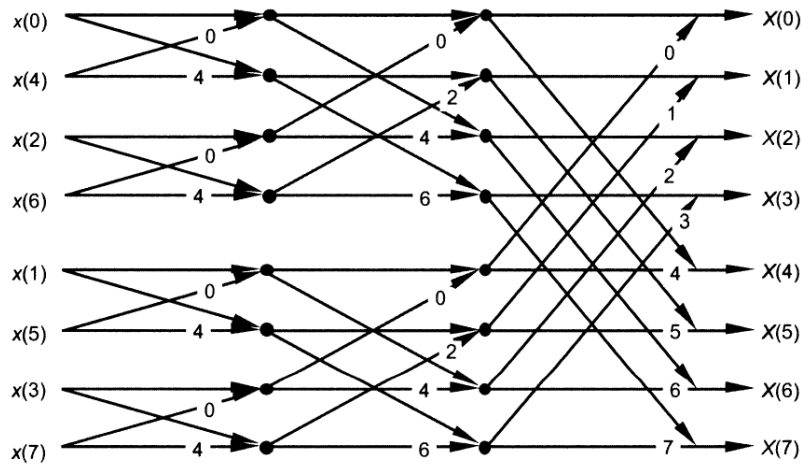

Рис. 9: 8-точечное БПФ с прореживанием по времени, входные отсчеты которого подвергнуты бит-реверсивной перестановке.

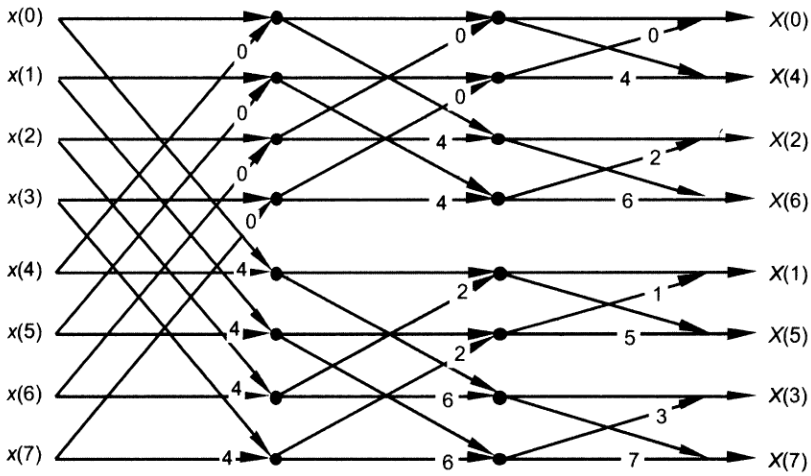

Рис. 10: 8-точечное БПФ с прореживанием по времени с бит-реверсивным порядком выходных отсчетов.

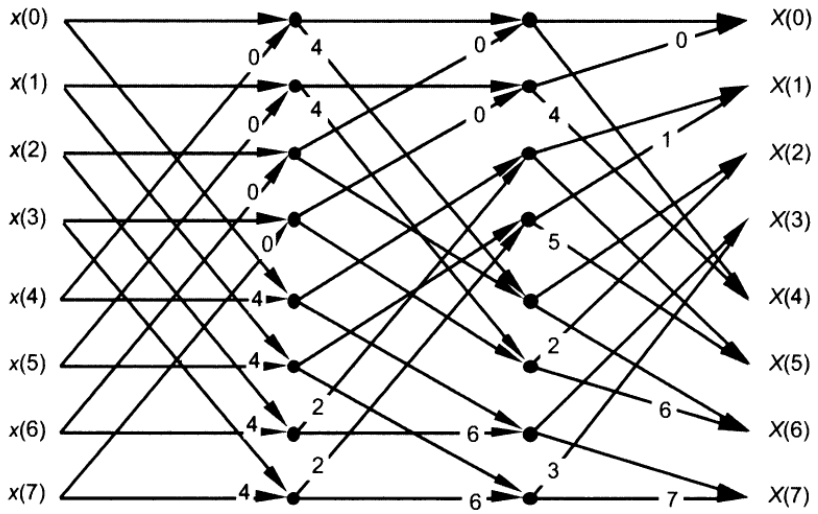

Рис. 11: 8-точечное БПФ с прореживанием по времени, входная и выходная последовательность которого расположены в нормальном порядке.

Существует другой способ получения алгоритма БПФ, который приводит к похожим структурам бабочек, но с другими поворачивающими множителями. Эта разновидность алгоритмов БПФ известна как БПФ с прореживанием по частоте. Если алгоритм БПФ с прореживанием по времени основан на разделении входных данных на четные и нечетные отсчеты, алгоритм БПФ с прореживанием по частоте основан на раздельном вычислении четных и нечетных отсчетов в частотной области. Вывод алгоритма с прореживанием по частоте достаточно прост и включен во многие статьи и учебники, поэтому здесь мы не будем заниматься этим. Но мы приводим структуры бабочек с прореживанием по частоте на рисунках 12 - 14 (аналогичные структурам, показанным на рисунках 9 - 11).

Для каждой структуры БПФ с прореживанием по времени существует эквивалентная структура с прореживанием по частоте. Важно отметить, что количество умножений при реализации алгоритма БПФ с прореживанием по частоте совпадает с количеством умножений при реализации БПФ с прореживанием по времени. Существует так много разных структур бабочек БПФ, описанных в литературе, что легко можно спутать бабочки с прореживанием по частоте и с прореживанием по времени. В зависимости от того, как подается материал, легко можно прийти к выводу, что БПФ с прореживанием по времени всегда требует бит-реверсивной перестановки входных отсчетов, а БПФ с прореживанием по частоте всегда выдает результат в бит-реверсивном порядке. Как показывают приведенные структуры, это не так. Прореживание по времени или частоте определяется тем, какая последовательность — входная или выходная, подвергается разделению при выводе той или иной структуры бабочек БПФ из формулы ДПФ.

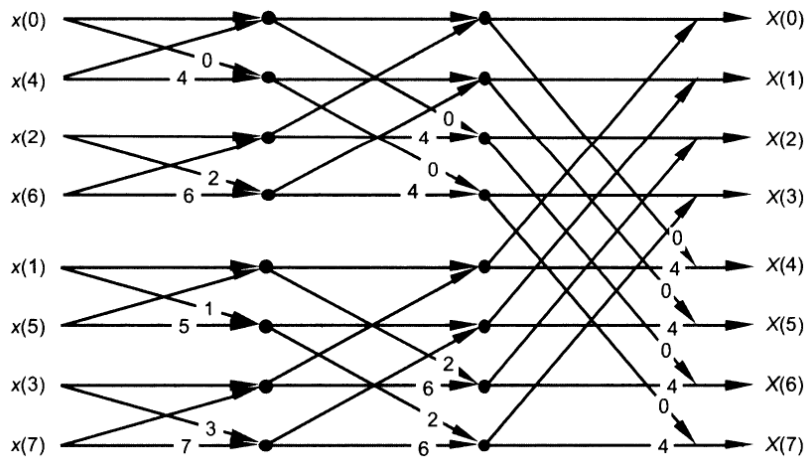

Рис. 12: 8-точечтное БПФ с прореживанием по частоте и бит-реверсивным порядком входных отсчетов.

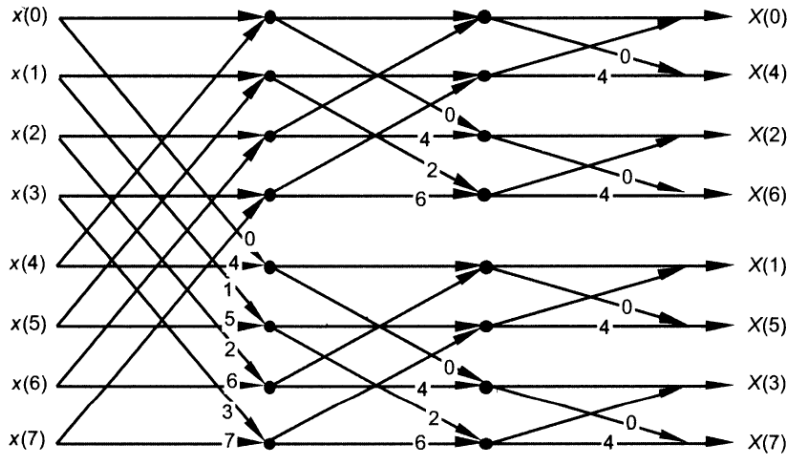

Рис. 13: 8-точечное БПФ с прореживанием по частоте и бит-реверсивным порядком выходной последовательности.

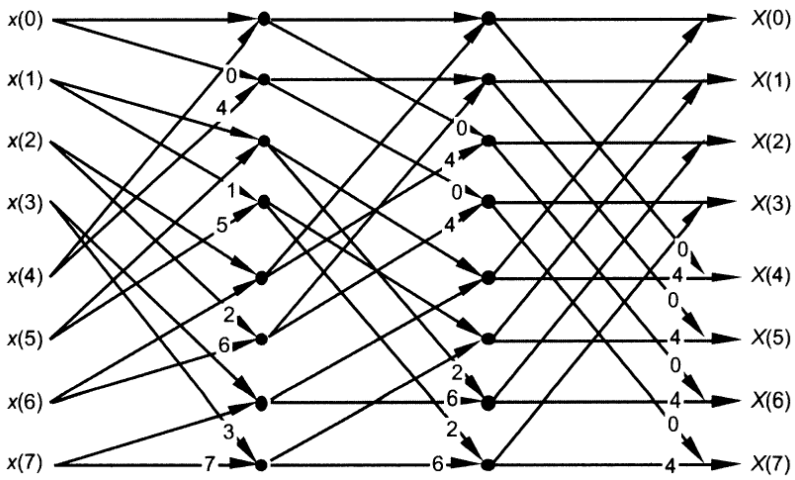

Рис. 14: 8-точечное БПФ с прореживанием по частоте и нормальным порядком входных и выходных отсчетов.

Рассмотрим еще раз отдельную бабочку. Структуры БПФ на рисунках 9, 10, 12 и 13 являются прямым результатом вывода алгоритмов с прореживанием по времени и по частоте. Хотя сначала это не очевидно, но показатели степени поворачивающих множителей, приведенные на этих рисунках, распределены по определенной системе. Обратите внимание на то, что они всегда принимают обобщенные формы, показанные на рисунке 15(а) [помните, что для простоты в структурах бабочек, показанных на рисунках 9 — 14, приведены только показатели степени поворачивающих множителей, k и $k+N/2$, а не полные комплексные поворачивающие множители. Для реализации бабочки с прореживанием по времени, показанной на рисунке 15(а), нам потребуется выполнить два комплексных умножения и два комплексных сложения. Конечно же, есть лучший способ реализации. Рассмотрим бабочку с прореживанием по времени на рисунке 15(а). Если обозначить верхний вход как x, а нижний — как y, верхний выход бабочки будет равен

$$x' = x + (W_N)^k y, \tag{26}$$

а нижний выход бабочки будет равен

$$y' = x + (W_N)^{k+N/2} y. \tag{27}$$

К счастью, операции в (26) и (27) можно упростить, потому что два поворачивающих множителя связаны соотношением

$$(W_N)^{k+N/2} = (W_N)^k (W_N)^{N/2} = (W_N)^k (e^{-j\pi N/2N}) = (W_N)^k (-1) = -(W_N)^k. \tag{28}$$

Следовательно, мы можем заменить поворачивающие множители $(W_N)^{k+N/2}$ на рисунке 15Р(а) множителями $-(W_N)^k$, что дает нам упрощенную форму

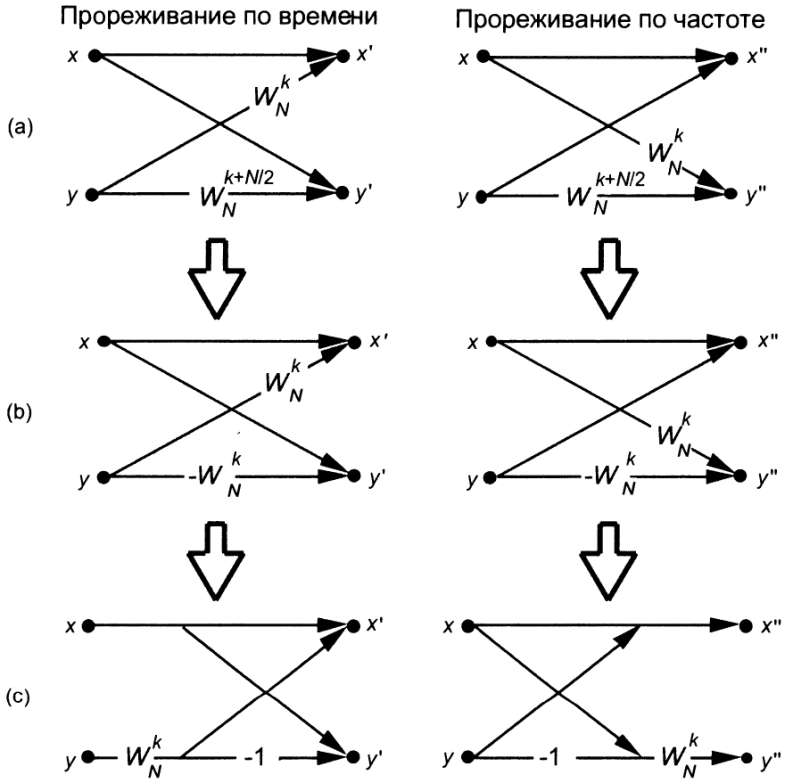

Рис. 15: Структуры бабочек с прореживанием по времени и частоте: (a) исходная форма; (Ь) упрощенная форма; (c) оптимизированная форма.

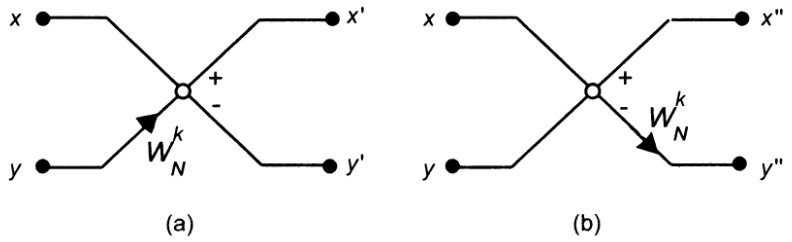

Рис. 16: Другой способ изображения бабочки: (a) с прореживанием по времени; (Ь) с прореживанием по частоте.

бабочек, показанную на рисунке 15(b). Поняв, что поворачивающие множители на рисунке 15(b) отличаются только знаками, мы приходим к оптимизированной форме бабочек, показанной на рисунке 15(c). Заметьте, что такие оптимизированные бабочки требуют два комплексных сложения, но всего одно комплексное умножение, что уменьшает общее количество операций [Мы говорим, что количество комплексных умножений при реализации БПФ равно $(N/2)\log_2 N$ именно потому, что N-точечное БПФ содержит $(N/2)\log_2 N$ бабочек.].

В литературе вы будете часто встречать оптимизированные структуры бабочек, приведенные на рисунке 15(c), вместо структур, показанных на рисунке 15(a). Эти оптимизированные бабочки позволяют нам легко распознавать алгоритмы с прореживанием по времени и по частоте. Если мы видим оптимизированные бабочки, подобные приведенным на рисунке 15(c), мы знаем, что это алгоритм с прореживанием по времени, если поворачивающий множитель стоит перед -1, в противном случае, если поворачивающий множитель стоит после -1, это алгоритм с прореживанием по частоте.

Иногда мы будем встречать в литературе структуры БПФ, в которых используются обозначения, показанные на рисунке 16. Эти бескрылые бабочки эквивалентны показанным на рисунке 15(c). В этих сигнальных графах используется соглашение о том, что выход узла, обозначенного кружочком, помеченный плюсом, представляет собой сумму отсчетов, входящих в узел слева, а выход, помеченным минусом, представляет собой разность этих отсчетов. Таким образом, выходные отсчеты бабочек с прореживанием по времени, доказанных на рисунках 15(c) и 16(a) вычисляются как

$$x' = x + (W_N)^k y \text{ и } y' = x - (W_N)^k y. \tag{29}$$

Выходы бабочек с прореживанием по частоте, показанных на рисунках 15(c) и 16(b) равны

$$x'' = x + y \text{ и } y'' = (W_N)^k (x - y) = (W_N)^k x - (W_N)^k y. \tag{30}$$

Так какая же структура лучше? Это зависит от приложения, аппаратурной реализации и соображений удобства. Если для выполнения БПФ

мы используем программу на компьютере общего назначения, выбор у нас обычно невелик. Большинство специалистов просто используют существующие подпрограммы БПФ, включенные в коммерческие пакеты программ. Их код может быть оптимизирован по скорости выполнения, но вы этого никогда не узнаете. Чтобы разобраться в том, как они реализуют БПФ, может потребоваться изучение этого кода. Если для нас важна скорость работы, первым делом нужно проверить, вычисляются ли синусы и косинусы каждый раз, когда требуется поворачивающий множитель. Обычно вычисление тригонометрических функций занимает множество машинных тактов. Скорость вычисления БПФ можно повысить, вычислив поворачивающие множители заранее и сохранив их таблицу в памяти. В этом случае вычисление поворачивающего множителя заменяется выборкой из таблицы. Если мы пишем собственную процедуру выполнения БПФ, мы можем реализовать алгоритм в целочисленной арифметике, которая на большинстве машин работает быстрее, но при этом мы должны принять меры против возможных переполнений на выходах бабочек и внимательно отнестись к масштабированию результатов. При использовании целочисленной арифметики следует быть внимательным: некоторые RISC процессоры в действительности выполняют целочисленные операции медленнее, т. к. они оптимизированы для выполнения операций над числами с плавающей запятой.

Если мы используем для вычислений векторный процессор, программы для этих процессоров всегда оптимизированы, т. к. их главная цель — высокая скорость вычислений. Изготовители векторных процессоров обычно рекламируют свои изделия, указывая скорость выполнения на них 1024-точечного БПФ.

Посмотрим теперь, какие возможности есть у нас при выборе структуры БПФ для реализации в виде специализированной аппаратуры.

Обсуждавшиеся ранее структуры бабочек БПФ обычно попадают в одну из двух категорий: алгоритмы БПФ с замещением и алгоритмы БПФ с удвоенной памятью. Алгоритм с замещением показан на рисунке 5. Выход бабочки запоминается в той же ячейке памяти, в которой хранились входные отсчеты. Память для хранения промежуточных результатов не нужна. В этом случае для N-точечного БПФ требуется только $2N$ ячеек памяти (коэффициент 2 учитывает то, что операция бабочки выполняется над комплексными числами, имеющими действительную и мнимую части). Камнем преткновения при реализации алгоритмов с замещением является довольно сложная адресация памяти. Структура БПФ с двойной памятью изображена на рисунке 11. В этом случае необходима промежуточная память, т. к. в структуре уже нет стандартной бабочки, и требуется дополнительно $4N$ ячеек памяти. Но доступ к данным и адресация памяти в этих структурах намного проще, чем в случае алгоритма с замещением. Высокоскоростные интегральные схемы с плавающей запятой, реализующие конвейерные структуры БПФ, более полно используют преимущества конвейерной архитектуры при использовании алгоритмов с двойной памятью.

Существует еще один класс структур БПФ, известный как алгоритмы с постоянной геометрией, которые делают адресацию памяти не только про-

стой, но и одинаковой для всех каскадов БПФ. Эти структуры особенно интересны тем, кто проектирует специализированные аппаратурные процессоры БПФ. С точки зрения аппаратуры в общем случае алгоритмы с прореживанием по времени являются оптимальными для действительных входных последовательностей, а алгоритмы с прореживанием по частоте больше подходят для обработки комплексных входных последовательностей. Когда входная последовательность БПФ симметрична во времени, для устранения ненужных операций используются специальные структуры БПФ.

В случае двумерного БПФ, например, при обработке изображений, алгоритмы с прореживанием по частоте считаются оптимальными. Ваше приложение может оказаться таким, что бит-реверсивная перестановка входной и выходной последовательности не важна. Некоторые приложения позволяют манипулировать выходной последовательностью БПФ в бит-реверсивном порядке без восстановления нормального порядка отсчетов. Затем обратное преобразование, которое использует входную последовательность в бит-реверсивном порядке, даст результат во временной области в нормальном порядке. В этой ситуации бит-реверсивная перестановка не нужна вообще. Примером подобного использования БПФ является перемножение двух преобразований Фурье в частотной области для получения свертки или корреляции во временной области. Как можно видеть, выбор оптимального алгоритма и аппаратурной архитектуры БПФ представляет собой достаточно сложную проблему, но, к счастью, литература может дать нам указания по ее решению.