Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ «САРАТОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра информатики и программирования

**ДИСКРЕТНОЕ ПРЕОБРАЗОВАНИЕ ФУРЬЕ И ЕГО РЕАЛИЗАЦИЯ В ПАРАЛЛЕЛЬНЫХ ВЕРСИЯХ БИБЛИОТЕК ПОДДЕРЖКИ ВЫСОКОПРОИЗВОДИТЕЛЬНЫХ ВЫЧИСЛЕНИЙ FFTW**

КУРСОВАЯ РАБОТА

студента 1 курса 141 группы  
направления 050100.62 Математическое обеспечение и администрирование информационных систем

факультета компьютерных наук и информационных технологий

Голикова Артема Олеговича

Научный руководитель

Профессор, кандидат физ.-мат. наук \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д. К. Андрейченко

подпись, дата

Зав. кафедрой

Профессор, кандидат физ.-мат. наук \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д. К. Андрейченко

подпись, дата

Саратов 2013 г.

# ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ](#_Toc357048318) 3

[ГЛАВА 1 ПРЕОБРАЗОВАНИЯ ФУРЬЕ 4](#_Toc357048319)

[1.1. Преобразование Фурье 4](#_Toc357048320)

[1.2. Дискретное преобразование Фурье 5](#_Toc357048321)

[1.3. Одномерное дискретное преобразование Фурье 6](#_Toc357048322)

[1.4. Двумерное дискретное преобразование Фурье 7](#_Toc357048323)

[1.5. Многомерное дискретное преобразование Фурье 8](#_Toc357048324)

[ГЛАВА 2 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ 9](#_Toc357048325)

[2.1. Обработка цифровых сигналов 9](#_Toc357048328)

[2.2. Библиотека FFTW 13](#_Toc357048329)

[2.2.1 Одномерное преобразование Фурье для комплексных чисел 15](#_Toc357048330)

[2.2.2 Многомерные преобразования для комплексных чисел 17](#_Toc357048331)

[2.2.3 Модуль параллельной обработки БПФ для MPI 18](#_Toc357048332)

[2.3. Тесты FFTW 21](#_Toc357048333)

[ЗАКЛЮЧЕНИЕ 27](#_Toc357048334)

Список использованной литературы [2](#_Toc357048334)8

# ВВЕДЕНИЕ

Одним из мощных инструментов обработки данных, непрерывных функций f(x) или некоторого набора дискретных данных f(xi), является спектральный анализ, имеющий в своей основе различные интегральные преобразования. Спектральный анализ используется в различных целях, таких как: подавление шума, фильтрация, построения амплитудно-частотной характеристики, так и для решения других проблем обработки данных.

Примерами спектров является преобразование Фурье.

Преобразование Фурье имеет огромное значение для различных математических приложений и широко применяется в науке и технике. Очень часто приходится иметь дело с дискретным набором данных f(xi), вследствие чего был разработан так называемый спектральный анализ при помощи дискретного преобразования Фурье (ДПФ). Но в большинстве случаев скорость расчета ДПФ, оставляет желать лучшего, и для него был разработан очень эффективный алгоритм, называемый БПФ (быстрое преобразование Фурье). Одной из самых известных на данный момент библиотек для подсчета БПФ и ДПФ является библиотека FFTW. Мы рассмотрим, что собой представляет ДПФ, и его реализацию в библиотеке поддержки высокопроизводительных вычислений FFTW. Посмотрим тесты выполнения различных преобразований на двух разных платформах. Так же сравним быстродействие различных библиотек по вычислению ДПФ и БПФ.

# ГЛАВА 1 ПРЕОБРАЗОВАНИЯ ФУРЬЕ

## Преобразование Фурье

В 1807 году французский математик и физик Жан Батист Жозеф Фурье представил во Французский Институт (Institut de France) доклад о синусоидальном представлении температурных распределений. Доклад содержал спорное утверждение о том, что любой непрерывный периодический сигнал может быть представлен суммой выбранных должным образом сигналов синусоидальной формы. Среди членов комитета, занимавшихся обзором публикаций, были два известных математика – Жозеф Луи Лагранж и Пьер Симон де Лаплас. Лагранж категорически возразил против публикации на основании того, что подход Фурье неприменим к разрывным функциям, таким как сигналы прямоугольной формы. Работа Фурье была отклонена, прежде всего из-за возражения Лагранжа, и была издана после смерти Лагранжа, приблизительно пятнадцатью годами позже.

Преобразование Фурье — это интегральное преобразование, которое раскладывает исходную функцию на базисные функции, в качестве которых выступают синусоидальные функции, то есть представляет исходную функцию в виде интеграла синусоид различной частоты, амплитуды и фазы.

Общий вид преобразования Фурье для непрерывной функции f(t), имеет вид:



Где функция φ(s) и есть преобразование Фурье, для функции f(t). Обратное преобразование Фурье (ОПФ) имеет вид:



Недостаток преобразования Фурье в том, что он «работает» только с непрерывной функцией, что не всегда возможно и целесообразно. В большинстве случаев, мы имеем дело лишь с дискретной функцией (данными) или вообще ограниченным набором данных, для которых мы не можем применить преобразование Фурье, для этого мы должны провести интерполяцию данных. Однако для дискретного набора данных, было придумано так называемое дискретное преобразование Фурье (ДПФ).

## Дискретное преобразование Фурье

Дискретное преобразование Фурье (в англоязычной литературе DFT, Discrete Fourier Transform) — это одно из преобразований Фурье, широко применяемых в алгоритмах цифровой обработки сигналов (его модификации применяются в сжатии звука в MP3, сжатии изображений в JPEG и др.), а также в других областях, связанных с анализом частот в дискретном (к примеру, оцифрованном аналоговом) сигнале. Дискретное преобразование Фурье требует в качестве входа дискретную функцию. Такие функции часто создаются путём дискретизации (выборки значений из непрерывных функций). Дискретные преобразования Фурье помогают решать частные дифференциальные уравнения и выполнять такие операции, как свёртки. В математике, свёртка — это математическая операция двух функций f и g, порождающая третью функцию, которая обычно может рассматриваться как модифицированная версия одной из первоначальных. Дискретные преобразования Фурье также активно используются в статистике, при анализе временных рядов. Существуют одномерные, двумерные и многомерные дискретные преобразования Фурье.

## Одномерное дискретное преобразование Фурье

Дискретное преобразование Фурье является аналогом рядов Фурье T-периодических функций времени t



и получающихся из рядов Фурье при  прямого и обратного преобразования Фурье абсолютно интегрируемых на числовой оси функций:



Пусть - некоторый N-мерный комплексный вектор. Прямое и обратное преобразование Фурье определены следующим образом:



Очевидно, величина  представляет собой один из корней степени  из единицы.

В смысле нумерации, дискретное преобразование Фурье несколько отличается от рядов Фурье. Так, величина  аналогична Фурье - коэффициенту ,  аналогична ,  аналогична ,  аналогична ,  аналогична  и так далее.

В общем случае, выполнение дискретного преобразования Фурье требует порядка  действий. Однако, если , где M – целое число, то число действий резко сокращается до , и говорят, что имеет место быстрое преобразование Фурье.

## Двумерное дискретное преобразование Фурье

Двумерное дискретное преобразование Фурье является аналогов двойных рядов Фурье функций двух переменных  и  с периодом  и .

, 

И получающегося из них при  и  двукратного прямого и обратного преобразования Фурье абсолютно интегрируемых в числовой плоскости  функций:





Пусть

 - некоторая матрица. Прямое и обратное дискретное преобразование Фурье (двумерное) суть:

, 



## Многомерное дискретное преобразование Фурье

Пусть - n мерный массив. Дискретное прямое и обратное n-кратное преобразование Фурье имеет вид:







# ГЛАВА 2 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ



## Обработка цифровых сигналов

Цифровая обработка, в отличие от аналоговой, традиционно используемой во многих радиотехнических устройствах, является более дешевым способом достижения результата, обеспечивает более высокую точность, миниатюрность и технологичность устройства, температурную стабильность.

Наиболее жесткие требования к аппаратной части цифровой обработки предъявляют радиолокационные системы. Основным содержанием цифровой обработки здесь является фильтрация входных сигналов антенны, частоты сигналов от 10 МГц до 10 ГГц. Размеры преобразований могут достигать до 214 комплексных точек, требования по быстродействию составляют 109 умножений в секунду.

При обработке цифровых сигналов радиолокатора используются алгоритмы цифровой фильтрации и спектрального анализа (вычисление дискретного и быстрого преобразования Фурье - ДПФ и БПФ), алгоритмы корреляционного анализа, обратной свертки, специальные алгоритмы линейного предсказания. В системах обработки звука цифровые процессоры обработки сигнала решают задачи анализа, распознавания и синтеза речи, сжатия речи в системах телекоммуникации. Для систем обработки изображений типовыми задачами являются улучшение изображений, сжатие информации для передачи и хранения, распознавание образов. При обработке цифровых звуковых сигналов используются алгоритмы цифровой фильтрации и спектрального анализа (вычисление ДПФ и БПФ), алгоритмы корреляционного анализа, обратной свертки, специальные алгоритмы линейного предсказания. В большинстве случаев удовлетворительные результаты обеспечивает формат данных с фиксированной запятой, длина слова 16 бит, частоты сигналов от 4 до 20 кГц (до 40 кГц в случае обработки музыки), требуемая производительность - до 10x106 операций в секунду - 10 MIPS по компьютерной терминологии.

Характерным для систем обработки изображений является восстановление и улучшение изображений с помощью инверсной свертки, обработка массивов отсчетов с помощью алгоритмов быстрого преобразования Фурье. При восстановлении трехмерной структуры объектов, получаемых методами проникающего излучения в дефектоскопии и медицинской интраскопии, применяются методы пространственно-частотной фильтрации. Другой класс алгоритмов - преобразование контрастности, выделение контуров, статистическая обработка изображений. Для сжатия информации наиболее эффективны ортогональные преобразования Фурье, Адамара и Уолша. Требуемая производительность оценивается величинами 100-1000 MIPS, массивы данных - 105-106 отсчетов.

Так сложилось, что хорошо разработанным (традиционным) приемом анализа и обработки сигналов (и аналоговых, и цифровых) стало разложение по базису синусов-косинусов - преобразование Фурье. В условиях линейности и стационарности это очень удобный инструмент как для обработки детерминированных, так и стохастических сигналов. Но если линза выполняет преобразование Фурье мгновенно, то выполнение преобразования Фурье для цифрового сигнала на классическом последовательном процессоре требует времени. Кули и Тьюки, разработав в свое время алгоритм быстрого преобразования Фурье (БПФ) и приведя вычислительные затраты к соотношению O(Nlog2N), сделали революцию в цифровой обработке сигналов. Но аппетиты растут, и умножать все равно надо много и быстро.

В свое время даже делали специализированные устройства для выполнения БПФ. Потом появились специализированные одночиповые DSP-процессоры. Они приспособлены для быстрого выполнения примитивов цифровой обработки сигналов, часто используется конвейерная обработка. Характерная их особенность - специализированный набор команд, что в ряде случаев вызывает потребность в универсальном процессоре для взаимодействия с внешним миром. Бывает, что DSP-процессоры содержат встроенные АЦП и ЦАП. Специализированные устройства, как правило, имеют более высокое быстродействие, чем универсальные. Все это хорошо, но, как правило, дорого, требует специальных навыков и инструментов разработки для создания законченного изделия и оправданно лишь при массовом серийном производстве или для специальных применений.

С другой стороны, технология изготовления универсальных процессоров тоже не стоит на месте. Набор команд универсальных процессоров дополняется подмножествами команд, правильно, цифровой обработки сигна лов. Для процессоров Intel эти расширения называются MMX, Streaming SIMD Extensions, SSE2. Мультимедиа, Win-модемы, DVD- и MPEG-проигрыватели, распознавание речи и многие другие технологии широко используют эти расширения.

Конечно, уже написано множество программ для обработки сигналов и изображений. Но бывает, что возникают задачи, которые еще не решены, либо требуется встраивание средств обработки в программную систему, либо нужно создать специализированный инструмент (опять возвращаемся от универсального к специализированному, но на программном уровне). Что для этого требуется? Приобретать специализированный DSP-процессор плюс средства разработки для него? Не обязательно. Многие задачи можно успешно решать на уже имеющемся компьютере с процессором Intel inside.

В качестве одного из решений можно назвать FFTW (The Fastest Fourier Transform in the West) - разработку, выполненную Маттео Фриго и Стивеном Джонсоном из Массачусетского технологического института. Это переносимая (способная работать на множестве платформ) библиотека, написанная на C, свободная и доступная позволяющая вычислять одно- и многомерные дискретные преобразования Фурье по алгоритму Кули-Тьюки. Изюминка FFTW заключается в очень хитрой технологии оптимизации вычисления БПФ в зависимости от особенностей процессора и длины реализации. На этапе компиляции библиотеки генерируются вычислительные примитивы (codelets) на С, учитывающие особенности процессора. Затем на этапе выполнения для заданного размера выборки планировщик (planer) соединяет эти примитивы оптимальным образом (генерируя байт-код алгоритма). А исполнительный блок (executor) этот план реализует, обращаясь к вычислительным примитивам. Получается, что программа сама приспосабливается к условиям и работает оптимально на любом процессоре без переписывания исходника. Авторы утверждают, что интерпретация байт-кода не слишком накладна («наши результаты противоречат обывательской теореме, что байт-код медленный»).

Вот так схематично выглядит программа на С, выполняющая БПФ с использованием FFTW:

fftw\_plan plan;

int n = 1024;

COMPLEX A[n], B[n];

/\* создание плана \*/

plan = fftw\_create\_plan(n);

/\* исполнение плана \*/

fftw(plan, A);

/\* повторное использование плана \*/

fftw(plan, B);

## Библиотека FFTW

FFTW - это быстрая библиотека на С, реализующая преобразования Фурье. Содержит комплексные, натуральные, симметричные, многомерные и параллельные преобразования и может эффективно обрабатывать массивы произвольного размера. FFTW обычно работает быстрее других бесплатных реализаций преобразования Фурье, и даже может конкурировать с тонко отлаженными библиотеками производителей процессоров. Для достижения такой производительности используются нестандартные программные ходы и техника самооптимизации во время выполнения (вкупе со многими другими приемами).

FFTW 3.3.3 это последняя официальная версия FFTW. Ниже представлен список некоторых интересных особенностей библиотеки:

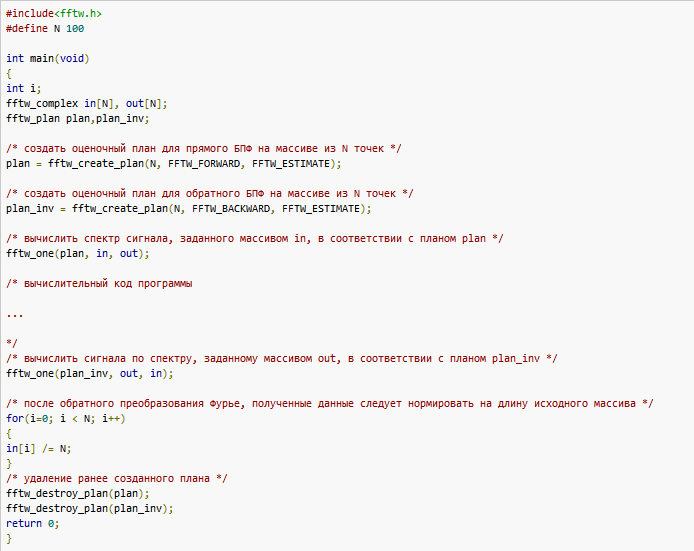
* Скорость. (Поддерживает SSE/SSE2/Altivec, начиная с версии 3.0. Версия 3.3.1 поддерживает AVX и ARM Neon.)
* Выполняет одно- и многомерные преобразования.
* Преобразует данные произвольного размера.
* Быстрые преобразования без погрешности ввода и вывода данных.
* Параллельно трансформирует: распараллелен код для платформ с SMP-машинами или OpenMP.
* Портативность на любой платформе с компилятором C.
* Документация в HTML и в других форматах.
* Задействованы интерфейсы C и Fortran.
* Бесплатное программное обеспечение, выпущенное под лицензией GNU General Public License.

FFTW состоит из четырех различных вариантов вычисления БПФ:

* Одномерное преобразование Фурье для комплексных чисел
* Многомерное преобразование Фурье для комплексных чисел
* Одномерное преобразование Фурье для действительных чисел
* Многомерное преобразование Фурье для действительных чисел

Сначала происходит построение "плана", который оптимизирует время вычислений для данной задачи. Затем построенный план передается в качестве параметра функциям, которые непосредственно отвечают за вычисление БПФ.

### Одномерное преобразование Фурье для комплексных чисел



Пояснения к функции fftw\_create\_plan:

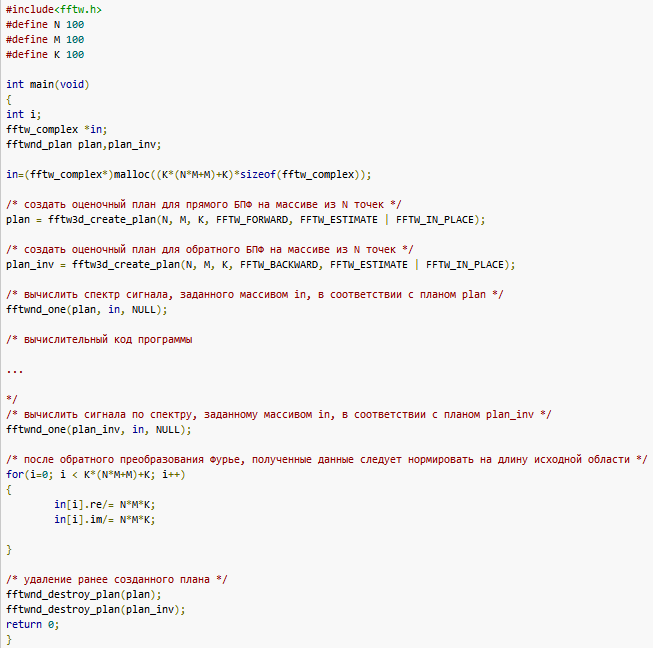
**fftw\_create\_plan(int size, fftw\_direction dir, int flags)**

* size - количество точек, используемых для вычисления БПФ.
* dir - флаг прямого FFTW\_FORWARD или обратного FFTW\_BACKWARD преобразования Фурье.
* flag - FFTW\_ESTIMATE создает план, который не является полностью оптимальным для данной конфигурации; FFTW\_MEASURE - используется для создания оптимального плана, в этом случае значительно увеличивается время работы по вычислению плана и БПФ.

Также можно использовать флаг FFTW\_IN\_PLACE:

plan = fftw\_create\_plan(N, FFTW\_FORWARD, FFTW\_ESTIMATE | FFTW\_IN\_PLACE) тогда в качестве out указывается значение NULL: **fftw\_one(plan, in, NULL)**

### Многомерные преобразования для комплексных чисел



Для построения многомерного плана (вместо приставки fftw\_ используется fftwnd\_, где n размерность) используются функции:

* fftwnd\_plan fftw2d\_create\_plan(int nx, int ny, fftw\_direction dir, int flags) - для двумерного БПФ;
* fftwnd\_plan fftw3d\_create\_plan(int nx, int ny, int nz, fftw\_direction dir, int flags) - для трехмерного БПФ.

Здесь nx, ny, nz - количество точек по соответствующим декартовым координатам (x, y, z); Флаги dir и flag имеют те же значения, что и в случае одномерного БПФ.

Функция преобразования сигнал/спектр:

**void fftwnd\_one(fftwnd\_plan plan, fftw\_complex \*in, fftw\_complex \*out)**

Функция удаления созданного плана:

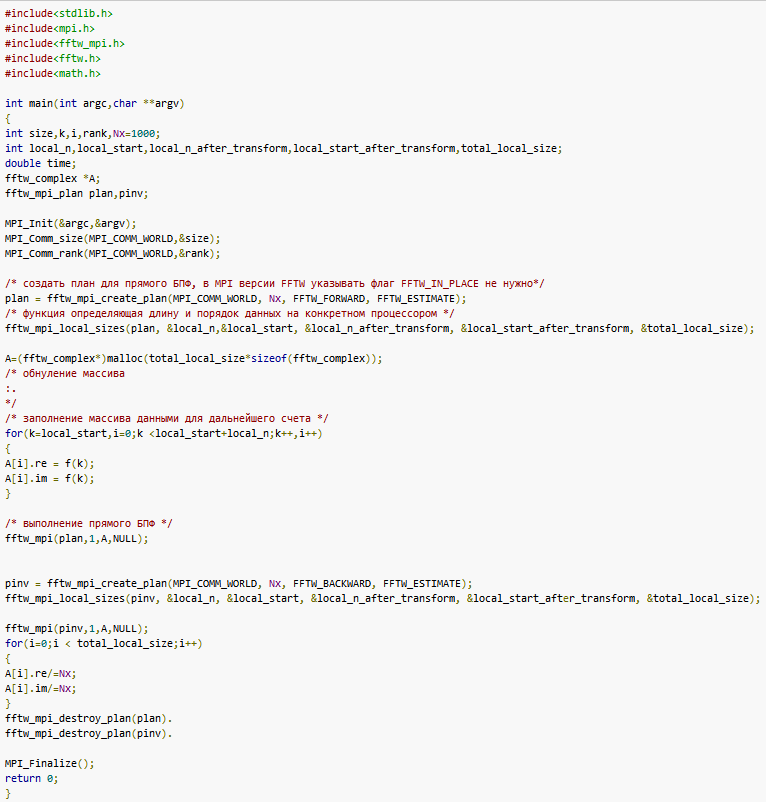
**fftwnd\_destroy\_plan(fftwnd\_plan plan)**

### Модуль параллельной обработки БПФ для MPI

Модуль параллельной обработки БПФ для MPI позволяет работать на многопроцессорных машинах с общей и распределенной памятью. Основное его отличие от однопроцессорного модуля FFTW заключается в том, что на каждом процессоре обрабатывается свое подмножество точек. Функция создания плана для параллельной обработки принимает, в дополнение к стандартным параметрам, значение коммуникатора для набора процессов (MPI\_COMM\_WORLD или другой коммуникатор). Обмен данными производиться с помощью функций MPI\_Alltoall или MPI\_Alltoallv в зависимости от алгоритма распределения данных по процессорам. Надо отметить, что наилучшее распределение по процессорам происходит при условии, что число точек кратно P2, где P - число процессов.

В исходный текст программы нужно включить заголовочный файл в дополнение к . При сборке программы нужно добавить специальную библиотеку с помощью флага -lfftw\_mpi, в дополнение к общей библиотеке FFTW (флаг -lfftw).

Пример выполнения одномерного комплексного БПФ с помощью параллельной версии FFTW:



Функция создания плана для одномерного комплексного БПФ:

**fftw\_mpi\_plan fftw\_mpi\_create\_plan(MPI\_Comm comm, int nx, fftw\_direction dir, int flags);**

Параметры:

* comm - коммуникатор группы процессоров, на которых будет производиться вычисление БПФ;
* nx - длина массива данных на всех процессорах;
* dir - флаг прямого FFTW\_FORWARD или обратного FFTW\_BACKWARD преобразования Фурье.
* flag - FFTW\_ESTIMATE создает план, который не является полностью оптимальным для данной конфигурации; FFTW\_MEASURE - используется для создания оптимального плана, в этом случае значительно увеличивается время работы по вычислению плана и БПФ. FFTW\_IN\_PLACE не указывается, т.к. в этом случае он стоит по умолчанию. Если не важен порядок выходных данных можно использовать FFTW\_SCRAMBLER\_OUTPUT для прямого и FFTW\_SCRAMBLER\_INPUT для обратного БПФ.

Вычисление количества точек, обрабатываемых на каждом из процессоров: **void fftw\_mpi\_local\_sizes(fftw\_mpi\_plan plan,int \*local\_n,int \*local\_start, int \*local\_n\_after\_transform, int \*local\_start\_after\_transform, int \*total\_local\_size);**

* plan - ранее созданный план
* local\_n - количество точек на процессоре перед преобразованием;
* local\_start - указывает с какой точки начинается нумерация на данном процессоре перед преобразованием; (таким образом, на каждом процессоре перед преобразованием размещаются точки от local\_start до local\_start+local\_n-1)
* local\_n\_after\_transform - количество точек на процессоре после преобразования;
* local\_start\_after\_transform - указывает с какой точки начинается нумерация на данном процессоре после преобразования;
* total\_local\_size - размер рабочей области на процессоре

Функция параллельного выполнения БПФ:

**fftw\_mpi(fftw\_mpi\_plan pinv, n\_fields, fftw\_complex \*local\_data, fftw\_complex \*work)**

Здесь n\_fields - количество одинаковых векторов, которые надо преобразовать (в простейшем случае равен 1). Для выполнения многомерных преобразований в параллельной версии используются аналогичные функции с префиксом fftwnd\_mpi.

## Тесты FFTW

Мною были проведены тесты библиотеки FFTW версии 3.3.3. Я собрал две версии библиотек: первая на платформе Win32, вторая на х64. Обе собраны набором инструментов Visual Studio 2012(v110), с поддержкой OpenMP. Всего было проведено 12 различных преобразований Фурье: одномерные, двумерные, трехмерные преобразования с использованием одного потока и двух, с исходными данными кратных и не кратных степени 2. При выполнении преобразований использовал параметр «i», чтобы выделяемая память не выходила за пределы оперативной памяти. Данные компьютера, на котором проводились тесты библиотеки Win32:

* Процессор: Intel Core 2 Duo E8500 3.16GHz
* Оперативная память: DDR3 4GB
* Операционная система: Windows 7

Данные компьютера, на котором проводилось тестирование библиотеки х64:

* Процессор: Intel Core i7 3610QM 2.6GHz
* Оперативная память: DDR3 6GB
* Операционная система: Windows 7

Ниже в таблицах приведены результаты тестов.

Результаты на платформе Win32 с использованием однопоточности:

Таблица 1

|  |  |  |  |
| --- | --- | --- | --- |
|  | Время построения | Время преобразования | Mflops |
| 67108864 | 36.80 s | 4.67 s | 1867.7 |
| 2130880 | 619.88 s | 288.41 ms | 776.64 |
| 8192x8192 | 26.20 s | 2.92 s | 2989.7 |
| 8001x8003 | 47.40 s | 9.82 s | 845.22 |
| 512x512x256 | 11.33 s | 2.63 s | 3319.8 |
| 501x551x247 | 19.21 s | 10.31 s | 860.82 |

Результаты на платформе Win32 с использованием многопоточности:

Таблица 2

|  |  |  |  |
| --- | --- | --- | --- |
|  | Время построения | Время преобразования | Mflops |
| 67108864 | 46.21 s | 2.95 s | 2957.7 |
| 2130880 | 649.51 s | 190.14 ms | 1178 |
| 8192x8192 | 47.27 s | 1.67 s | 5218.4 |
| 8001x8003 | 111.60 s | 5.20 s | 1597.9 |
| 512x512x256 | 19.90 s | 1.57 s | 5543.6 |
| 501x551x247 | 28.32 s | 6.48 s | 1368.8 |

Результаты на платформе x64 с использованием однопоточности:

Таблица 3

|  |  |  |  |
| --- | --- | --- | --- |
|  | Время построения | Время преобразования | Mflops |
| 134217728 | 296.78 s | 5.05 s | 3590.3 |
| 120000001 | 187.70 s | 16.38 s | 983.18 |
| 8192x16384 | 23.41 s | 3.92 s | 4625.8 |
| 8001x16401 | 331.86 s | 12.42 s | 1424.3 |
| 512x512x512 | 14.80 s | 3.52 s | 5146.2 |
| 501x551x501 | 13.47 s | 14.56 s | 1284.1 |

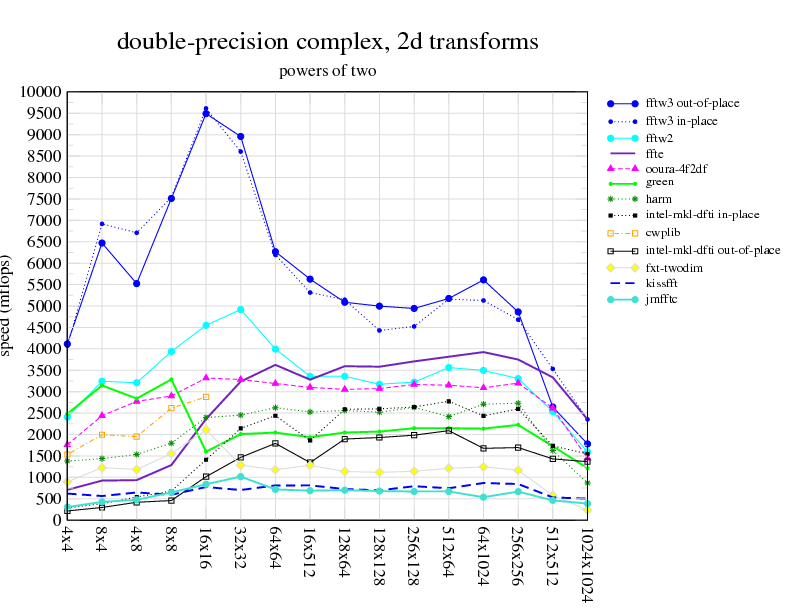
Результаты на платформе х64 с использованием многопоточности:

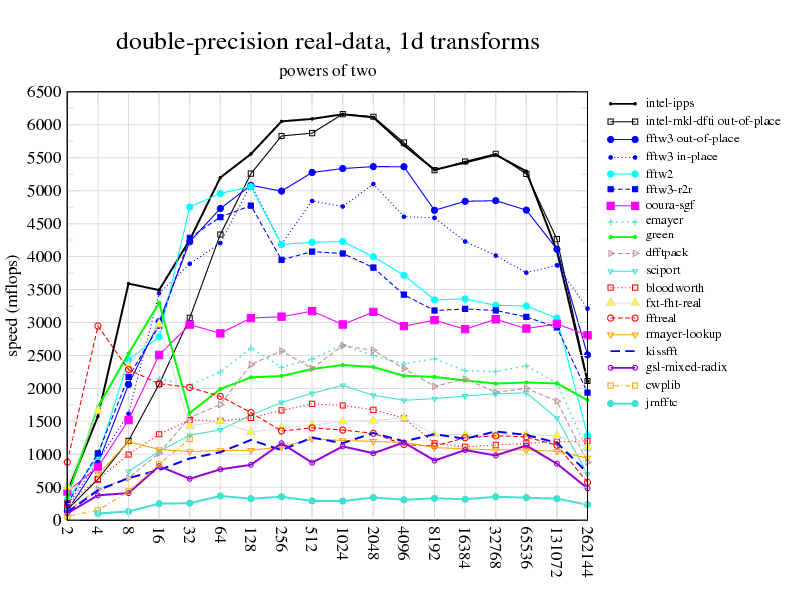
Таблица 4

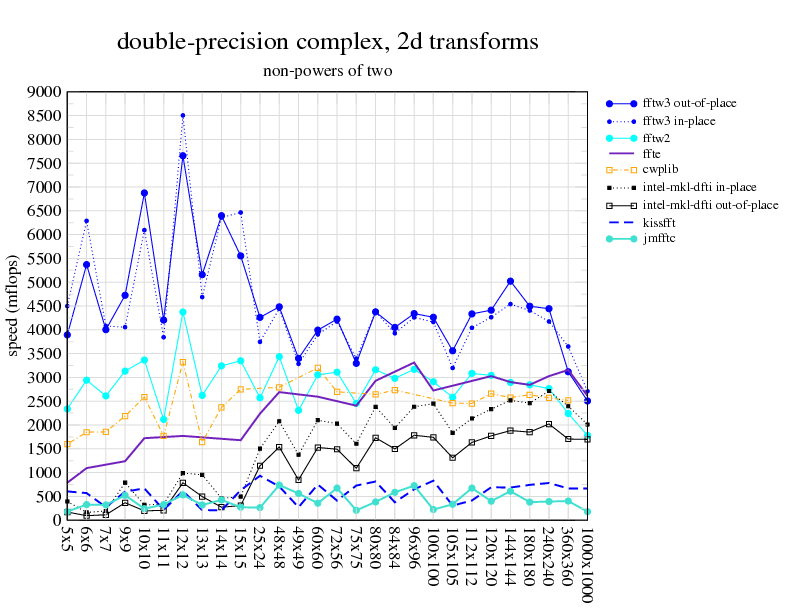
|  |  |  |  |
| --- | --- | --- | --- |
|  | Время построения | Время преобразования | Mflops |
| 134217728 | 283.68 s | 1.95 s | 9270.3 |
| 120000001 | 196.25 s | 8.26 s | 1949.8 |
| 8192x16384 | 43.17 s | 1.91 s | 9505.7 |
| 8001x16401 | 866.83 s | 4.92 s | 3595.9 |
| 512x512x512 | 22.64 s | 1.43 s | 12675 |
| 501x551x501 | 27.36 s | 4.44 s | 4208.9 |

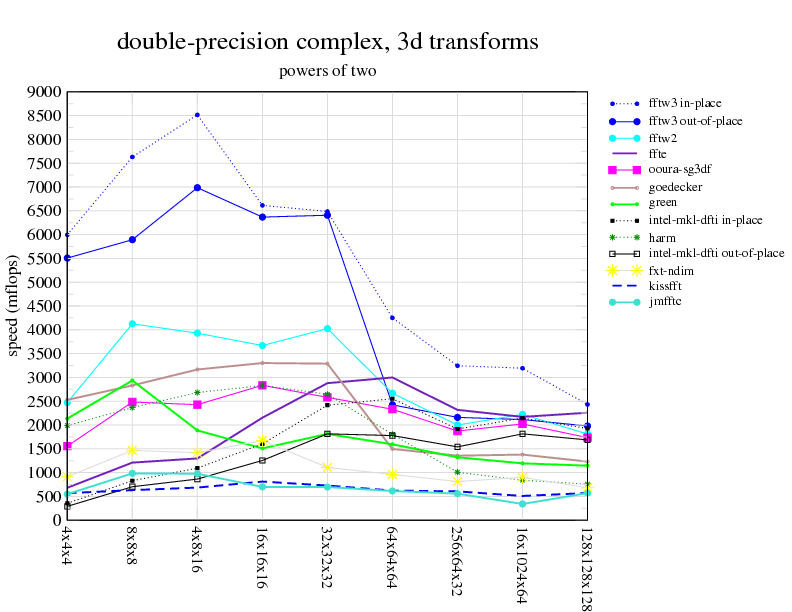
В итоге, если сравнивать между собой Таблицу 1 и Таблицу 2, то можно проследить увеличение скорости выполнения преобразований примерно в 1.5-1.9 раза, за счет разбиения на два потока. Если же сравнивать результаты Таблицы 3 и Таблицы 4, то можно проследить увеличение скорости выполнения преобразований примерно в 2-3 раза, за счет разбиения на 4 потока. Так же отчетливо видно превосходство в скорости БПФ.

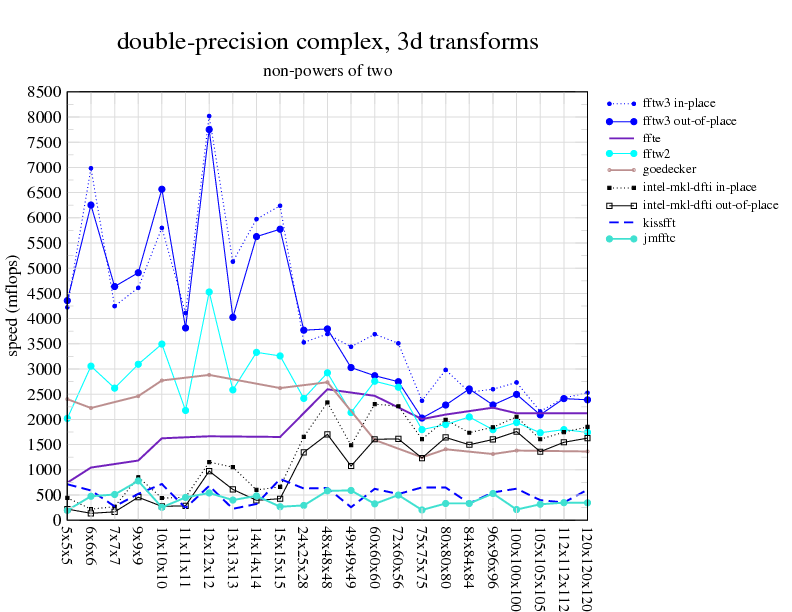
Ниже представлены графики производительности различных библиотек по вычислению ДПФ:

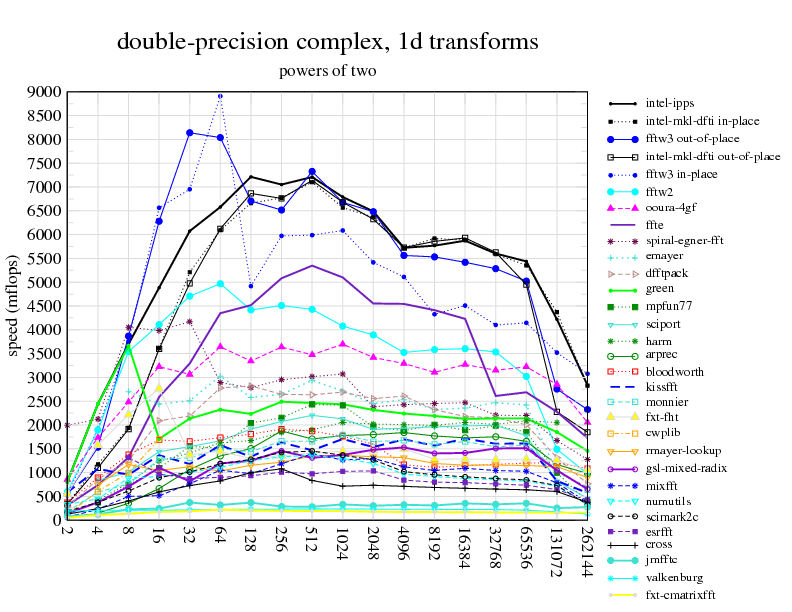












# ЗАКЛЮЧЕНИЕ

Компьютерные технологии не стоят на месте. Одним из главных аспектов «гонки» технологий является быстродействие и производительность. Так как дискретное преобразование Фурье часто встречаемо, то естественно нужны действенные средства для его вычисления. Тому примером стала библиотека FFTW. Нестандартные подходы и методы само оптимизации позволяют этой библиотеке быть одним из лидеров среди себе подобных. Еще один плюс FFTW – это бесплатная библиотека, так еще и исходные коды доступны на официальном сайте. В плане производительности данная библиотека очень хороша, она может конкурировать с тонко отлаженными библиотеками производителей процессоров. Проводя тесты, сам убедился в эффективности многопоточности и преимуществе ДПФ в скорости выполнения.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ**

1. Сергиенко А. Б. Цифровая обработка сигналов. — 2-е. — Спб: Питер, 2006. — С. 751. — ISBN 5-469-00816-9
2. FFTW: [Электронный ресурс]. , 2003-2013. URL: http://www.fftw.org. (Дата обращения: 22.02.2013).
3. Библиотека FFTW для выполнения быстрых преобразований Фурье (БПФ):[Электронный ресурс].,URL: http://www.parallel.ru/cluster/fftw. (Дата обращения: 25.02.2013)
4. OpenMP FFTW: [Электронный ресурс] //ProgCyberForum.ru., URL: http://www.progcyberforum.ru/openmp\_fftw /. (Дата обращения: 18.01.2013).