傅里叶变换是连续周期

DFS（）是离散周期

DFT是离散有限展开为周期

DFT即离散傅里叶变换，是时间复杂度为O(n2)的朴素算法

FFT是DFT的快速算法，将时间复杂度转化为O(nlogn)

所谓傅里叶级数

若f（t）是以T为周期的周期函数，并且在（-T/2，T/2）上满足Dirichlet条件（大多数都满足该条件），则在（-T/2，T/2）上f（t）可以展开成傅氏级数。

我们认为，一个周期函数可以由多个三角函数叠加而成，因此我们可以表示为：

将三角函数展开为

记，将上式表示

此时我们隐式认为基频率，我们可以表达为更一般的方式，令，此时我们有

此时我们得到了在f(t)的连续点t处级数的三角形式

其中对两边积分得到直流成分

对积分，因为三角函数系的正交性，只有频率为的余弦分量不为0，所以

同理

此时我们就可以将时域t转换为频域A了

随后我们引入欧拉公式

其中e是自然对数的底

将转化为

令，，

有

其中

我们可以发现，当n取0时，，同时有，上式可合并为

化简为（去掉了指数上的负号）

这就是我们所求的频域，该式为傅里叶变换（此处的1/T在傅里叶变换或反傅里叶变换前均可，习惯性放在反傅里叶变换前）

同时根据

得到傅里叶级数的复数形式，该式为反傅里叶变换

有了以上基础，我们可以讨论DFS（离散傅里叶级数）, 由于计算机无法处理无限的或连续的输入，因此我们必须要将连续的积分处理转化为离散的数据加和

对于一个周期为N的正弦序列，基频分量为（应为，这里省略T），即角频率

基频成分为

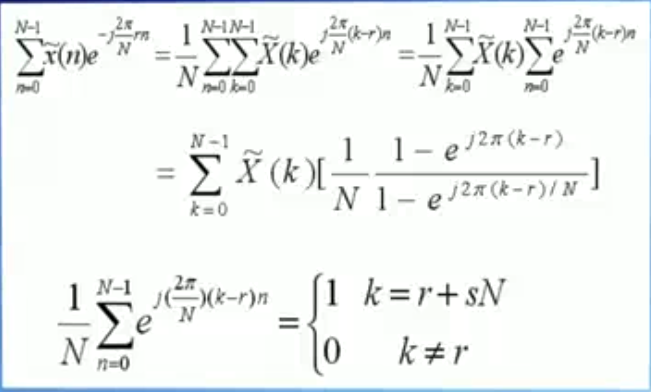
K次谐波序列为

基于采样定理，信号最高的频率分量，不会超过1/2的采样频率（采样频率为2π），由于离散级数所有谐波成分中只有N个是独立的，所以

即

将周期序列展为离散傅里叶级数时，只需取k=0到N-1这N个独立的谐波分量，所以一个周期的离散傅里叶级数只需包含这N个复指数。（）

上式中n表示时间，k表示频率，k=0为直流，k=1为基频，为时域，

为求，对两侧同乘（用r表示，区分于k），并对一个周期求和

根据不知道什么意思的上式，我们得到

习惯上记

整理为

此时我们可以开始讨论DFT

因此对于一个长为N的有限长序列x(n)，有

假定一个周期序列有

但是在公式上是完全一样的

在计算时，我们可以根据欧拉公式将DFT展开为

因为我们的输入只有实数,此时我们可以这样分别计算实数部分与虚数部分

接下来我们讨论二维DFT

我们可以将二维的DFT视为一个的矩阵，为求 依据二维傅里叶变换的可分离性,先按一个轴向做一次一维离散傅里叶变换，再将计算结果按另一个轴向做一次一维离散傅里叶变换就可以得到该图像的二维傅里叶变换结果，也可以对全部数据一起进行离散傅里叶变换，即（同样我们最终将会把放在逆运算）

分解步骤如下

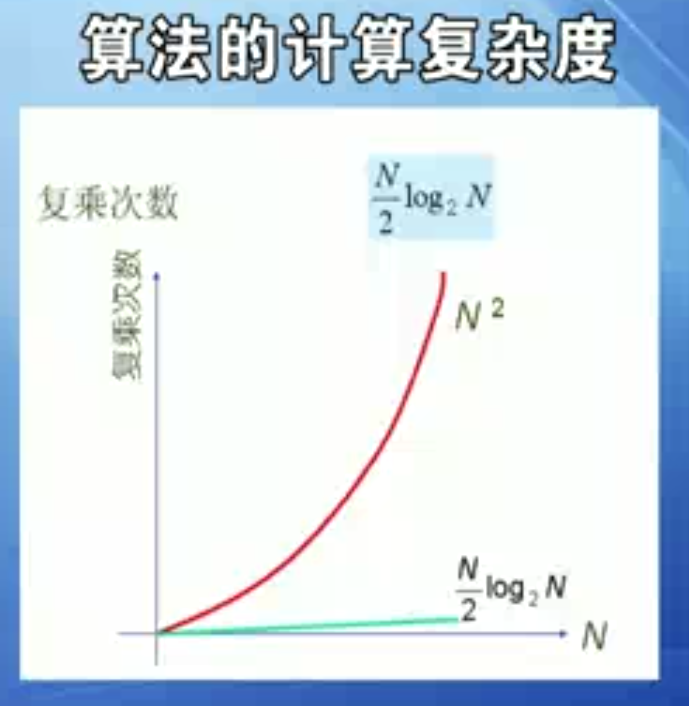
整理如下

特别说明一下，这里X与X共轭

我们现在可以开始着手制作盲水印了，

好吧并不能，这玩意的速度太慢了，我们必须求助于更快的FFT

由于DFT的时间复杂度为O(N²)，数据提升一倍时间提升四倍，所以FFT采取分治的方法将时间复杂度降低到O(NlogN)



首先我们从一维开始

注意其中系数有周期性和对称性，有，以及由得出

对于按时间抽取的FFT，假定N是2的整数次幂，将序列x(n)分为奇数项与偶数项两组

所以

因为

带回上式

其中

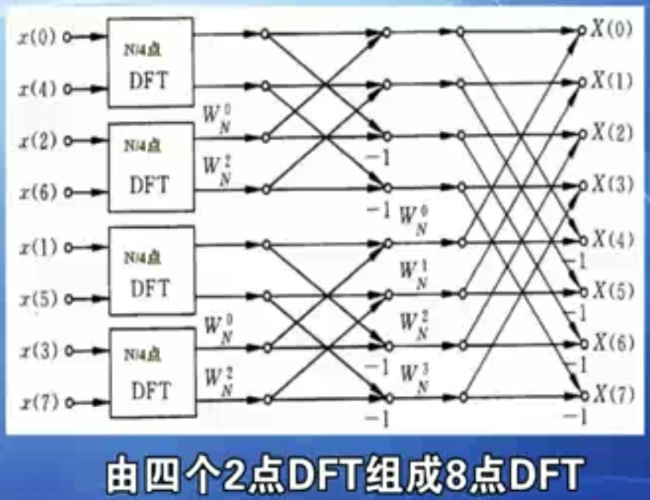
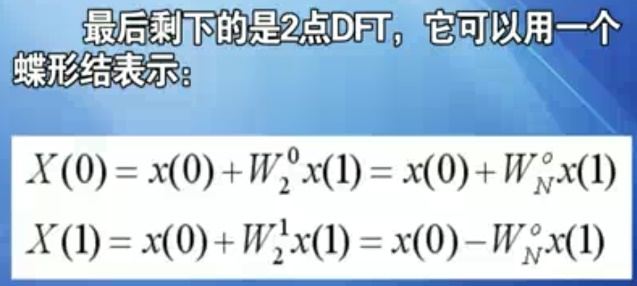
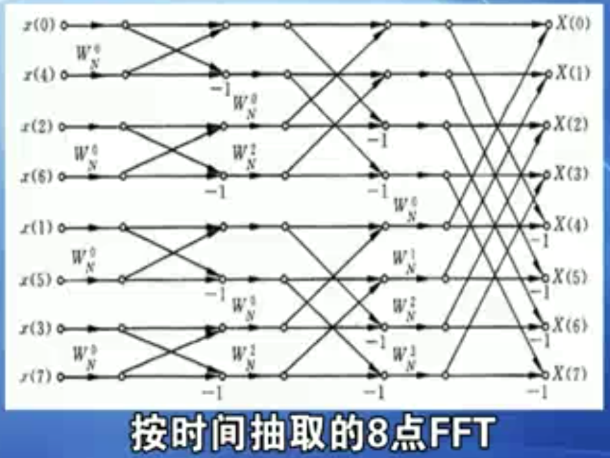
变为了两个N/2的DFT

但现在这个式子只能表示0到N/2-1,

由的特性，我们得到以及

得

整理如下

例子如下

我们可以看出：

蝶形运算：时间抽取FFT总共会运算排，每排有N/2个蝶形单元，每个蝶形进行一次复数乘法，两次加减法

原位计算：一次只有当前位置的两个数据参与运算，每个蝶形单元的输入输出可以保存在相同位置，避免了寻找内存的支出。

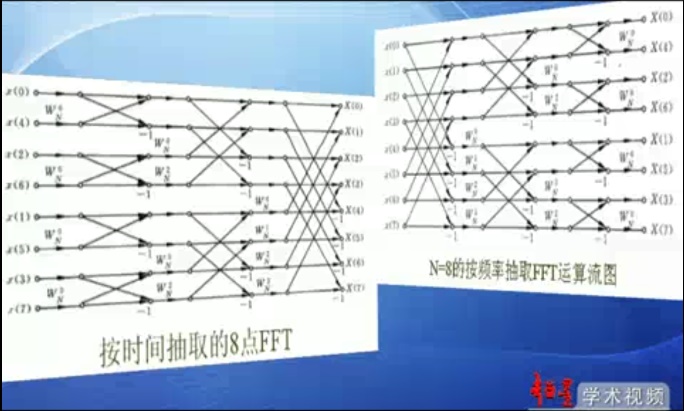
序数重排：原地址按二进制位反转即为新地址

蝶形类型随迭代次数成倍增加：每个蝶形单元取的两个数之间的间隔为,W因子的个数也成倍增加

我们可以采取递归的形式编写代码，每次只处理一个DFT，只负责合并两个低一层的DFT，

递归的边界条件是N=2。而在IFFT中，只需要让每个IDFT返回x/2即可，因为IDFT中每一个蝶形单元是相互独立的，而每一个蝶形单元的N都为2.

为了节省内存消耗与时间，我们可以用用迭代的方式模拟递归，我们一共需要三层循环，最外层循环表示当前层数，中层循环表示该层的当前节点，内层循环表示节点中的蝴蝶单元

按频率抽取的FFT抽取的是频域而不是时域，事实上并没有太大区别，只是加减与乘法顺序相反，可以在初始位置省略重排，但是频域是乱序，对于盲水印或许可以考虑先用频率抽取FFT，再用时间抽取IFFT节省重排开支，因为我们并不需要真正获取频域 。

到了现在，从一维的FFT变化到二维的FFT就非常容易了，虽然从二维的DFT到二维的FFT的二分不容易理解，但是我们可以依照从一维DFT到二维DFT的方式，将二维数据的行列各自进行一次一维FFT，就完成了二维FFT

参考：《复变函数与积分变换》，《数字信号处理》