FWT及其应用

PRODUCED BY XU SHU

28TH JAN 2018

一、引入动机

二、FWT简介

三、FWT应用

一、引入动机

问题引入

- 给出2组二进制数。要求从两组中各选择一个，使得两者或运算后为询问给定的数(i.e. x|y==k)。问有多少种组合?
  - 1001 0011 1100 1000
  - 1110 1001 0010 0100
    - Q1: 1100　A1: 2
    - Q2: 0111　A2: 1

问题分析

- N^2暴力枚举对数?
  - n<=100000   Kidding me ?
- 引入数组A[i] B[i]： 数字i分别在第一二组中的频数

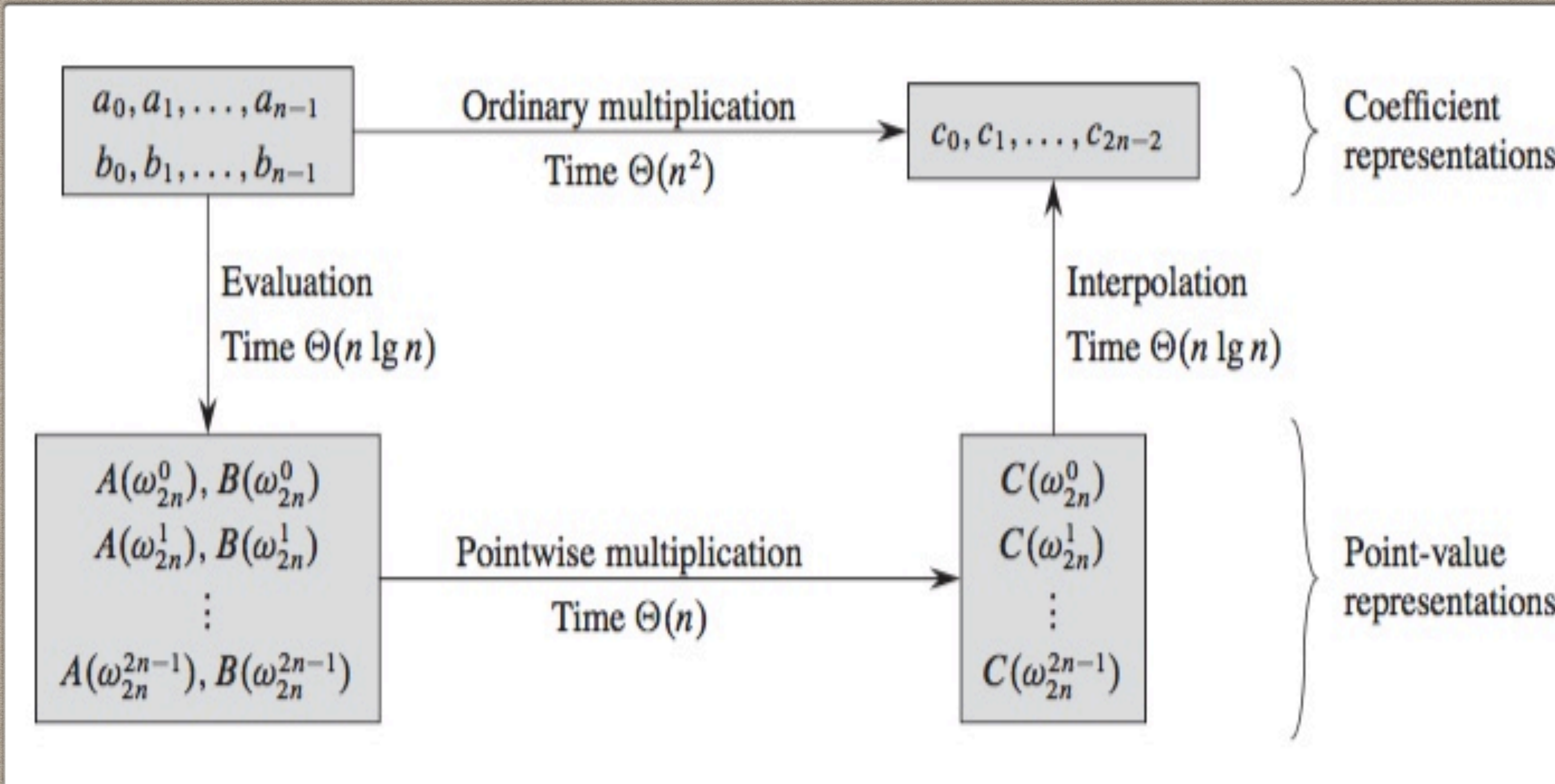
$$Ans = \sum_{i|j=k} A[i] * B[j]$$

- 卷积? FFT?
  - No. It should be FWT!

二、FWT简介

FWT解决的问题

- 快速沃尔什变换（Fast Walsh Transform），简称FWT。是快速完成集合卷积运算的一种算法。
- 主要功能是求：$\mathrm{C[i]}=\sum_{j \bigotimes k=i} \mathrm{A[j]*B[k]}$，j与k中间为集合运算符。



FWT的本质机制

类比FFT三步走：FWT->MULTIPLY->UFWT

FWT的封装实现

[cpp]

```cpp
void FWT(int a[],int n)
{
    for(int d=1;d<n;d<<=1)
        for(int m=d<<1,i=0;i<n;i+=m)
            for(int j=0;j<d;j++)
            {
                int x=a[i+j],y=a[i+j+d];
                a[i+j]=(x+y)%mod,a[i+j+d]=(x-y+mod)%mod;
                //xor:a[i+j]=x+y,a[i+j+d]=(x-y+mod)%mod;
                //and:a[i+j]=x+y;
                //or:a[i+j+d]=x+y;
            }
}

void UFWT(int a[],int n)
{
    for(int d=1;d<n;d<<=1)
        for(int m=d<<1,i=0;i<n;i+=m)
            for(int j=0;j<d;j++)
            {
                int x=a[i+j],y=a[i+j+d];
                a[i+j]=1LL*(x+y)*rev%mod,a[i+j+d]=(1LL*(x-y)*rev%mod+mod)%mod;
                //xor:a[i+j]=(x+y)/2,a[i+j+d]=(x-y)/2;
                //and:a[i+j]=x-y;
                //or:a[i+j+d]=y-x;
            }
}
```

FWT的封装实现

```
void solve(int a[],int b[],int n)
{
    FWT(a,n);
    FWT(b,n);
    for(int i=0;i<n;i++) a[i]=1LL*a[i]*b[i]%mod;
    UFWT(a,n);
}
```

- 不同的位运算只需在FWT和UFWT中相应替换算式即可
- 注意到FWT做的是二进制上的位运算，所以一定要把A和B补到2的整次幂次（即不足的地方填上0）传入参数n

三、FWT应用

HDU5909 TREE CUTTING
DP加速

- **问题描述**
- 一棵N个节点的树，每个节点有一个权值vi，子树的权值定义为子树内所有节点权值的异或和。问各个权值的子树分别有多少种?
- $N \le 1000$，$0 \le v < 1024$

问题分析

- 先序遍历子树，递归处理
- 定义状态s[u][i]: 以u为树根的所有子树中权值为i的种类数
- 初始时$s[u][v_u]=1$，其余为0
- 依次考虑u的各个儿子v，计算出s[v][]后，考虑新增它后对s[u][]的影响

$$Inc[k] = \sum_{i \bigoplus j=k} s[u][i] * s[v][j]$$

- 更新：s[u][k]+=Inc[k]

HDU6057 KANADE'S CONVOLUTION 等价变形

Give you two arrays $A[0..2^m - 1]$ and $B[0..2^m - 1]$.

Please calculate array $C[0..2^m - 1]$:

$$C[k] = \sum_{i \ and \ j = k} A[i \ xor \ j] * B[i \ or \ j]$$

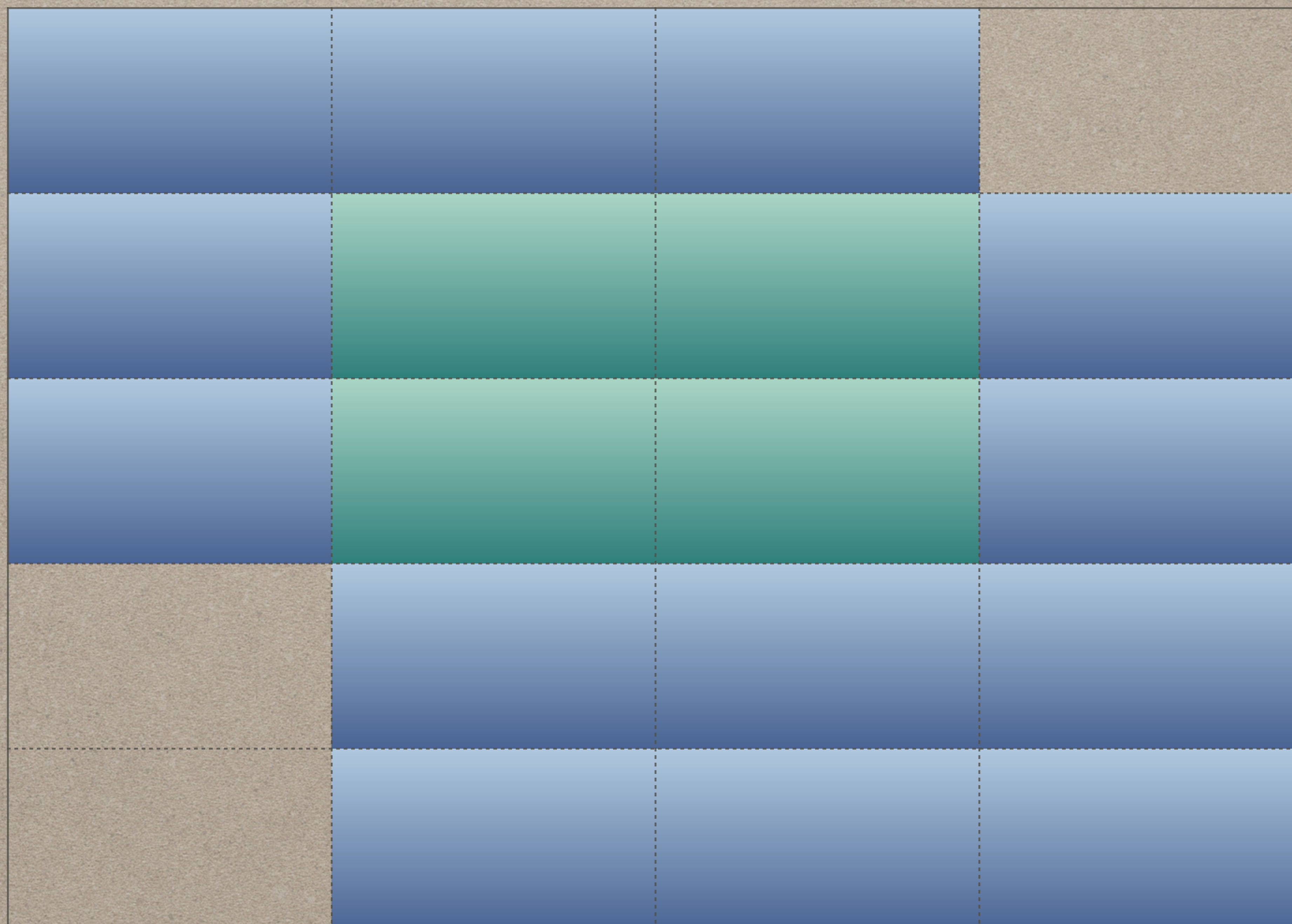
You just need to print $\sum_{i=0}^{2^m-1} C[i] * 1526^i \ mod \ 998244353$

$m <= 19$

$0 \leq A[i], B[i] < 998244353$

位运算的集合含义

&:∩ |:∪ ^:△(对称差)



问题分析

$$C[k] = \sum_{i\ and\ j=k} A[i\ xor\ j] * B[i\ or\ j]$$

- 换元

$$C[k] = \sum_{p\ xor\ q=k} A[p] * B[q]$$

- 等价吗?

$$C[k] = \sum_{p\ xor\ q=k} A[p] * 2^{bit(p)} * B[q],\ p\ and\ q = p$$

$$C[k] = \sum_{p\ xor\ q=k} [bit(q) - bit(p) = bit(k)] * A[p] * 2^{bit(p)} * B[q]$$

- 直接套用FWT?

改造FWT

$$C[k] = \sum_{p\ xor\ q=k} A[p] * 2^{bit(p)} * B[q]$$

$$= \sum_{p\ xor\ q=k} (\sum_{i=1}^{m} A[p][i] * 2^{bit(p)}) * (\sum_{j=1}^{m} B[q][j])$$

$$= \sum_{i=1}^{m} \sum_{j=1}^{m} \sum_{p\ xor\ q=k} A[p][i] * 2^{bit(p)} * B[q][j]$$

$$= \sum_{i=1}^{m} \sum_{j=1}^{m} D[i][j][k]$$

$$C[k] = \sum_{i=1}^{m} \sum_{j=1}^{m} [j - i = bit(k)] D[i][j][k]$$


Let’s practice!