Bluestein's Algorithm

Quack

Chongqing No.8 Secondary School

2016 年 5 月 18 日

扯点别的

这玩意还没进入OI吧，反正无所谓，弄懂之后觉得还是挺简单的东西，就讲一下吧。

扯点别的

这玩意还没进入OI吧，反正无所谓，弄懂之后觉得还是挺简单的东西，就讲一下吧。

Bluestein's Algorithm的作用是$O(n \log n)$算任意长度的DFT和IDFT。

DFT

DFT的实质就是拿一些单位根的幂带入系数表示的多项式中得到点值表示。由于单位根的性质原因这样的DFT只能做长度为2的幂。

DFT的实质就是拿一些单位根的幂带入系数表示的多项式中得到点值表示。 由于单位根的性质原因这样的DFT只能做长度为2的幂。

显然DFT的公式是

DFT

DFT的实质就是拿一些单位根的幂带入系数表示的多项式中得到点值表示。 由于单位根的性质原因这样的DFT只能做长度为2的幂。

显然DFT的公式是

$$X_k = \sum_{n=0}^{N-1} x_n e^{\frac{2\pi i}{N} nk}$$

DFT

DFT的实质就是拿一些单位根的幂带入系数表示的多项式中得到点值表示。由于单位根的性质原因这样的DFT只能做长度为2的幂。

显然DFT的公式是

$$X_k = \sum_{n=0}^{N-1} x_n e^{\frac{2\pi i}{N} nk}$$

其中X_k表示带入单位根的k次幂得到的点值表示，N是多项式长度，x_n表示第n项的系数，$e^{\frac{2\pi i}{N}}$是单位根，指数上的k表示是带入的单位根的k次幂，指数上的n表示第n项的次数是n。

考虑对上式进行变形。

考虑对上式进行变形。由$nk = \frac{-(k-n)^2}{2} + \frac{n^2}{2} + \frac{k^2}{2}$，可得

考虑对上式进行变形。由$nk = \frac{-(k-n)^2}{2} + \frac{n^2}{2} + \frac{k^2}{2}$，可得

$$X_k = e^{\frac{\pi i}{N}k^2} \sum_{n=0}^{N-1} (x_n e^{\frac{\pi i}{N}n^2}) e^{\frac{-\pi i}{N}(k-n)^2}$$

考虑对上式进行变形。由$nk = \frac{-(k-n)^2}{2} + \frac{n^2}{2} + \frac{k^2}{2}$，可得

$$X_k = e^{\frac{\pi i}{N}k^2} \sum_{n=0}^{N-1} (x_n e^{\frac{\pi i}{N}n^2}) e^{\frac{-\pi i}{N}(k-n)^2}$$

设$a_n = x_n e^{\frac{\pi i}{N}n^2}, b_n = e^{\frac{-\pi i}{N}n^2}$，有

考虑对上式进行变形。由$nk = \frac{-(k-n)^2}{2} + \frac{n^2}{2} + \frac{k^2}{2}$，可得

$$X_k = e^{\frac{\pi i}{N}k^2} \sum_{n=0}^{N-1} (x_n e^{\frac{\pi i}{N}n^2}) e^{\frac{-\pi i}{N}(k-n)^2}$$

设$a_n = x_n e^{\frac{\pi i}{N}n^2}, b_n = e^{\frac{-\pi i}{N}n^2}$，有

$$X_k = e^{\frac{\pi i}{N}k^2} \sum_{n=0}^{N-1} a_n b_{k-n}$$

考虑对上式进行变形。由$nk = \frac{-(k-n)^2}{2} + \frac{n^2}{2} + \frac{k^2}{2}$，可得

$$X_k = e^{\frac{\pi i}{N}k^2}\sum_{n=0}^{N-1}(x_n e^{\frac{\pi i}{N}n^2})e^{\frac{-\pi i}{N}(k-n)^2}$$

设$a_n = x_n e^{\frac{\pi i}{N}n^2}, b_n = e^{\frac{-\pi i}{N}n^2}$，有

$$X_k = e^{\frac{\pi i}{N}k^2}\sum_{n=0}^{N-1}a_n b_{k-n}$$

发现这是一个卷积的形式，所以可以用FFT优化计算卷积的过程来求DFT，这样就可以做任意长度的DFT了。时间复杂度$O(n\log n)$。

实现细节

不妨设我们要做的DFT的长度是m，发现$k \in [0, m-1]$。

实现细节

不妨设我们要做的DFT的长度是m，发现$k \in [0, m-1]$。

观察$X_k = e^{\frac{\pi i}{N}k^2} \sum_{n=0}^{N-1} a_n b_{k-n}$，得$k - n \in [-N+1, m-1]$。

实现细节

不妨设我们要做的DFT的长度是m，发现$k \in [0, m-1]$。

观察$X_k = e^{\frac{\pi i}{N} k^2} \sum_{n=0}^{N-1} a_n b_{k-n}$，得$k - n \in [-N+1, m-1]$。

所以这里就有一个问题，卷积下标可能为负数，直接FFT是不行的。

实现细节

不妨设我们要做的DFT的长度是m，发现$k \in [0, m-1]$。

观察$X_k = e^{\frac{\pi i}{N}k^2} \sum_{n=0}^{N-1} a_n b_{k-n}$，得$k - n \in [-N+1, m-1]$。

所以这里就有一个问题，卷积下标可能为负数，直接FFT是不行的。

解决方法是把b_n整体移动N位，即$b_n \to b_{n+N}$，这样有$k - n + N \in [1, m + N - 1]$。

实现细节

不妨设我们要做的DFT的长度是m，发现$k \in [0, m-1]$。

观察$X_k = e^{\frac{\pi i}{N}k^2} \sum_{n=0}^{N-1} a_n b_{k-n}$，得$k - n \in [-N+1, m-1]$。

所以这里就有一个问题，卷积下标可能为负数，直接FFT是不行的。

解决方法是把b_n整体移动N位，即$b_n \to b_{n+N}$，这样有$k - n + N \in [1, m+N-1]$。

显然X_k也会随之移动N位，那么$k + N \in [N, m+N-1]$，应该在这个位置找卷积结果。

实现细节

不妨设我们要做的DFT的长度是m，发现$k \in [0, m-1]$。

观察$X_k = e^{\frac{\pi i}{N}k^2} \sum_{n=0}^{N-1} a_n b_{k-n}$，得$k-n \in [-N+1, m-1]$。

所以这里就有一个问题，卷积下标可能为负数，直接FFT是不行的。

解决方法是把b_n整体移动N位，即$b_n \to b_{n+N}$，这样有$k-n+N \in [1, m+N-1]$。

显然X_k也会随之移动N位，那么$k+N \in [N, m+N-1]$，应该在这个位置找卷积结果。

另外，之前FFT的板子往往是处理实数卷积的，逆变换后只对实部除以N，现在卷积的值域扩大到复数，显然要对实部和虚部一起除以N。

IDFT

刚才只介绍了Bluestein's Algorithm做任意长的DFT，但由于IDFT形式和DFT非常相似，现在来探究IDFT在Bluestein's Algorithm下的实现。

IDFT

刚才只介绍了Bluestein's Algorithm做任意长的DFT，但由于IDFT形式和DFT非常相似，现在来探究IDFT在Bluestein's Algorithm下的实现。

给出IDFT的公式：

IDFT

刚才只介绍了Bluestein's Algorithm做任意长的DFT，但由于IDFT形式和DFT非常相似，现在来探究IDFT在Bluestein's Algorithm下的实现。

给出IDFT的公式：

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} X_k e^{\frac{-2\pi i}{N} nk}$$

IDFT

刚才只介绍了Bluestein's Algorithm做任意长的DFT，但由于IDFT形式和DFT非常相似，现在来探究IDFT在Bluestein's Algorithm下的实现。

给出IDFT的公式：

$$x_n = \frac{1}{N}\sum_{k=0}^{N-1} X_k e^{\frac{-2\pi i}{N}nk}$$

发现就是在DFT的基础上变一下符号，最后除以N就完了。

有两个长度为N的实数序列α, β。然后按照下面的步骤计算序列γ：

TN's Kingdom III - Assassination

有两个长度为N的实数序列α, β。然后按照下面的步骤计算序列γ：

1. 把β倒过来。

有两个长度为N的实数序列α, β。然后按照下面的步骤计算序列γ：

1. 把β倒过来。
2. 把β向右平移一个元素。最右侧的元素补到左边。

有两个长度为N的实数序列α,β。然后按照下面的步骤计算序列γ：

1. 把β倒过来。
2. 把β向右平移一个元素。最右侧的元素补到左边。
3. 计算此时α和β对应位置上元素的积之和，并加到γ的末尾。

有两个长度为N的实数序列α, β。然后按照下面的步骤计算序列γ：

1. 把β倒过来。
2. 把β向右平移一个元素。最右侧的元素补到左边。
3. 计算此时α和β对应位置上元素的积之和，并加到γ的末尾。
4. 如果γ还不足N个元素，重复步骤2和3。

有两个长度为N的实数序列α, β。然后按照下面的步骤计算序列γ：

1. 把β倒过来。
2. 把β向右平移一个元素。最右侧的元素补到左边。
3. 计算此时α和β对应位置上元素的积之和，并加到γ的末尾。
4. 如果γ还不足N个元素，重复步骤2和3。

已知β和γ，求α。

有两个长度为N的实数序列α, β。然后按照下面的步骤计算序列γ:

1. 把β倒过来。
2. 把β向右平移一个元素。最右侧的元素补到左边。
3. 计算此时α和β对应位置上元素的积之和,并加到γ的末尾。
4. 如果γ还不足N个元素,重复步骤2和3。

   已知β和γ,求α。

- N的最大质因子不超过19,$2 \leq N \leq 2^{17}$

有两个长度为N的实数序列α, β。然后按照下面的步骤计算序列γ：

1. 把β倒过来。
2. 把β向右平移一个元素。最右侧的元素补到左边。
3. 计算此时α和β对应位置上元素的积之和，并加到γ的末尾。
4. 如果γ还不足N个元素，重复步骤2和3。

已知β和γ，求α。

- N的最大质因子不超过19，$2 \le N \le 2^{17}$
- source：POJ 2821

显然γ是α和β的循环卷积。那么把γ和β分别DFT后相除得到α的点值表示然后做IDFT就完了。

显然γ是α和β的循环卷积。那么把γ和β分别DFT后相除得到α的点值表示然后做IDFT就完了。

现在有一个问题，就是不能直接做长度为2的幂的DFT，IDFT时会有问题。也不能写多项式除法，因为这是循环卷积。

显然γ是α和β的循环卷积。那么把γ和β分别DFT后相除得到α的点值表示然后做IDFT就完了。

现在有一个问题，就是不能直接做长度为2的幂的DFT，IDFT时会有问题。也不能写多项式除法，因为这是循环卷积。

所以就来一个定长的DFT吧。题目给了信息，N的最大质因子不超过19，标算是一个混合基FFT，实际上没必要，写Bluestein's Algorithm也是可以的。

显然γ是α和β的循环卷积。那么把γ和β分别DFT后相除得到α的点值表示然后做IDFT就完了。

现在有一个问题，就是不能直接做长度为2的幂的DFT，IDFT时会有问题。也不能写多项式除法，因为这是循环卷积。

所以就来一个定长的DFT吧。题目给了信息，N的最大质因子不超过19，标算是一个混合基FFT，实际上没必要，写Bluestein's Algorithm也是可以的。

混合基FFT不能处理大质数长度的DFT，也不能随意更换单位根，但Bluestein's Algorithm可以处理任意长度的DFT，代值可以不代入单位根。但混合基FFT的时空效率要优于Bluestein's Algorithm。