「学习总结」基本卷积算法

Jiayi Su (ShuYuMo)

2021-01-17 15:35:06

「做多项式题就像嗑药，出多项式题就像贩毒。」——某福建知名 OI 选手

学了学红日 bn 一年前玩剩下的东西。

倒是扩展了 FWT 的一种思路吧。

基本卷积算法

加法卷积

用于解决形如以下问题：给出两个序列 A, B。求两个序列的卷积序列 C，其中序列 C 的定义如下

$$C_k = \sum_{i+j=k} A_i \times B_j$$

{#eq:QWQ}

序列可以看作是函数的系数表示法，所以我们可以定义函数 $F(x) = _{i 0} A_i x^i $ 和函数 $G(x) = \sum_{i\geq 0} B_i x^i$ 分别对应序列 A, B 的两个函数。

类似的，还可以定义序列 C 对应函数 $T(x) = \sum_{i\geq 0} C_i x^i$。

注意到这种定义就是将序列的第 i 个数当作多项式函数的第 i 次系数。

注意到

$$T(x) = F(x) \times G(x)$$

{#eq:QAQ}

这个是后面卷积方法的核心工作原理。本质上是 建立了 序列运算 与其对应 函数（多项式）运算 的一种联系。

注意到 (eq.~@eq:QAQ) 也可以理解为对于任意常数 a，总有 $T(a) = F(a) \times G(a)$。

也就是说，如果我们知道了 $G(x)$ 和 $F(x)$ 在 a 处的函数值，那么他们相乘就能够得到 $T(x)$ 在 a 处的函数值。

考虑如果存在一种变换，能够快速计算 $G(x)$ 和 $F(x)$ 在某些特定自变量
$a_1, a_2, a_3, \cdots a_k$ 处的函数值，那么就能快速算出 $T(x)$ 在这些自变量处的函数值。

如果存在某种逆变换，能够快速将函数在某些自变量处的函数值转换成每一次项的系数，那么就能够求出 $T(x)$ 的系数（即序列 C）了。

总的来说，希望存在一种序列上变换 $\text{FFT}(T)$，使得

$$\text{FFT}(C) = \text{FFT}(A \times B) = \text{FFT}(A) \cdot \text{FFT}(B)$$

其中 $\times$ 表示序列加法卷积 (eq.~@eq:QWQ)，$\cdot$ 表示对应位置相乘。

并且这种变换存在逆变换，能够将 $\mathrm{FFT}(C)$ 还原为 C 。

通过上述描述可以发现，对于一个函数 $G(x)$ 来说，其表示成序列的方式有两种：一种是构造序列，使得序列第 i 位为多项式函数 $G(x)$ 的第 i 次项系数。另一种是指定一系列取值
$a_1, a_2, a_3, \cdots a_k$ ，分别带入函数中，得到的函数值依次排开，成为一个序列。我们称前者为函数 $G(x)$ 的系数表示法，称后者为其点值表示法。而我们所希望的变换就是能够实现函数的系数表示法 与 点值表示法 相互转化。

因为 两函数相乘时，其对应的系数表示法的序列就是在做卷积操作，对应了我们希望的运算，但是这个并不能快速计算。而两函数相乘，对于点值表示法，就仅仅是对应位置相乘，这个可以快速计算。所以可以考虑如果能够快速实现 函数的系数表示法 与 点值表示法 相互转化。就能够解决上述问题。可以先转化为点值表示法，然后对点相乘后，再转换为系数表示法，就相当于对序列做卷积。

快速傅里叶变换（FFT）

快速傅里叶变换 就是一种满足上述条件的变换。她可以使得函数在系数表示法与点值表示法之间转换。时间复杂度为 $\mathcal{O}(n \log n)$。

考虑上述过程中，并没有限制点值表示法中的点值应该取哪些值，所以可以考虑取一些有丰富性质的数字，利用这些性质加速运算。

对于 FFT 我们取 **n** 次单位根 来加速运算。n 为待变换序列长度。

单位根 n 次单位根记作 ω_n。其定义为 $\omega_n = \cos\frac{2\pi}{n} + \mathrm{i}\sin\frac{2\pi}{n}$，这是一个虚数。虚数可以通过向量来表示，而 ω_n 就可以考虑为一个模长为 1 ，与 x 轴夹角为 $\frac{2\pi}{n}$ 的向量。易知，$\omega_n^k = \cos\frac{2\pi k}{n} + \mathrm{i}\sin\frac{2\pi k}{n}$。

他有如下性质

- $\omega_n^k = \omega_{2n}^{2k}$
- $\omega_n^{k+\frac{n}{2}} = -\omega_n^k$
- $\omega_n^0 = \omega_n^n = 1$

这些性质都可以通过其定义得知。

考虑将 $\omega_n^0, \omega_n^1, \omega_n^2, \cdots, \omega_n^{n-1}$ 带入函数，得到点值即可。

蝴蝶操作 分别抽取函数 $F(x)$ 的奇、偶次项系数构成两个新函数 $G(x), H(x)$。易知:

$$F(x) = G(x^2) + xH(x^2)$$

即

$$F(\omega_n^k) = G(\omega_{n/2}^k) + \omega_n^k H(\omega_{n/2}^k)$$
$$F(\omega_n^{k+n/2}) = G(\omega_{n/2}^k) - \omega_n^k H(\omega_{n/2}^k)$$

划分了子问题，分治即可。这里必须保证 $n = 2^k$，$k \in \mathbb{N}$

逆变换 单位根还有如下性质

$$\sum_{i=0}^{n-1} \left(\omega_n^k\right)^i = \begin{cases} 0 & k \neq 0 \\ \\ n & k = 0 \end{cases}$$

第二种情况显然，第一种情况根据等比数列求和公式可以得证。

根据上述性质，我们只需要取单位根为之前的倒数，然后跑一遍 FFT 对结果除以 n 即可。不会推 QAQ。

快速数论变换（**NTT**）

FFT 需要实数运算，对精度要求较高。且无法解决常见的取模要求。

注意到 FFT 中所依赖的是一个复平面上的单位圆，其实剩余系本身就可以看作一个环。可以考虑在剩余系下寻找具有与单位根性质类似的数字。

设质数 P 的原根为 g。那么 $\forall k \in [0, \varphi(P)-1], k \in \mathbb{N}, g^k$，可以表示 P 的剩余系中除 0 以外的任何数字，显然可表示的数字有 $P-1$ 个。

以下关于剩余系类比复平面单位圆的描述由笔者口胡，不保证语言严谨。

在 FFT 中取单位根的方式本质上实在均分复平面上单位圆。而 NTT 中，如果将 g^k 依次排成一个环，即剩余系，我们一样可以通过均分这个环，来取剩余系下的“单位根”，来获得和之前复平面单位根 类似性质的一些数。显然，这里需要保证在我们均分单位圆的过程中，每个值都能取到，也就是 $n|\varphi(P)$，即 $n|(P-1)$。

所以这里的剩余系取值有一定要求，变换中所要求的 n 都是 2 的次幂，需要保证 $P-1$ 中 2 的幂次应该足够大。(文末附质数取值表)

设 $P-1 = q \times n$。

考虑将 $g_n = g^q$ 当作 ω_n 即可，根据上面的描述，ω_n 所具有的性质，g_n 显然成立。这里的 q 可以想成等分环时的单位角度。

关于 蝴蝶操作 和 逆变换 的手法和 FFT 是一样的。

位运算卷积

类似地，定义位运算序列卷积。给出两个序列 A, B。求两个序列的卷积序列 C，其中序列 C 的定义如下

$$C_k = \sum_{i \oplus j = k} A_i \times B_j$$

{#eq:QWQWQ}

可以仿照上述思路，构造一种作用在序列上的变换 FWT，使得其满足

$$\mathrm{FWT}(C) = \mathrm{FWT}(A \oplus B) = \mathrm{FWT}(A) \cdot \mathrm{FWT}(B)$$

{#eq:FWT}

$A \oplus B$ 中的 $\oplus$ 指某种位运算。指下标的运算方式。即 (eq.~@eq:QWQWQ) 中的 $i \oplus j = k$。

快速沃尔什变换（**FWT**）

考虑分治处理，令 A_0 为 A 的前一半，A_1 为 A 的后一半。可以考虑如果已经求出了 $\mathrm{FWT}(A_0)$ 和 $\mathrm{FWT}(A_1)$，如何求出 $\mathrm{FWT}(A)$。

顺便定义函数 $\mathrm{merge}(A, B)$ 表示将序列 A, B 直接前后拼接，返回拼接后的大序列。例如 $A = \mathrm{merge}(A_0, A_1)$。

或运算　考虑如何构造 FWT 的方式，使得：

$$\mathrm{FWT}(A|B) = \mathrm{FWT}(A) \cdot \mathrm{FWT}(B)$$

{#eq:FWTORBASE}

对于或运算卷积，直接给出结论。我们定义当前情况下的 FWT 的运算规则如下。

$$\mathrm{FWT}(A) = \mathrm{merge}(\mathrm{FWT}(A_0), \mathrm{FWT}(A_0) + \mathrm{FWT}(A_1))$$

{#eq:FWTOR}

其中 $+$ 为序列对应位置相加。

现在证明：已知 (eq.~@eq:FWTOR)，(eq.~@eq:FWTORBASE) 成立。

将 FWT 简记为 F，将 $\mathrm{merge}(A, B)$ 简记为 $[A, B]$

首先试图从等式左边开始推导：

$$\begin{aligned}
\mathrm{F}(A) \cdot \mathrm{F}(B) &= [\mathrm{F}(A_0),\ \mathrm{F}(A_0) + \mathrm{F}(A_1)] \cdot [\mathrm{F}(B_0),\ \mathrm{F}(B_0) + \mathrm{F}(B_1)] \\
&= [\mathrm{F}(A_0) \cdot \mathrm{F}(B_0),\ (\mathrm{F}(A_0) + \mathrm{F}(A_1)) \cdot (\mathrm{F}(B_0) + \mathrm{F}(B_1))] \\
&= [\mathrm{F}(A_0) \cdot \mathrm{F}(B_0),\ \mathrm{F}(A_0) \cdot \mathrm{F}(B_0) + \mathrm{F}(A_0) \cdot \mathrm{F}(B_1) \\
&\quad + \mathrm{F}(A_1) \cdot \mathrm{F}(B_0) + \mathrm{F}(A_1) \cdot \mathrm{F}(B_1)]
\end{aligned}$$

再从等式右边开始推导，先假设当序列长度为 $\frac{|A|}{2}$ 时，根据 (eq.~@eq:FWTOR) 能够使得 (eq.~@eq:FWTORBASE) 成立。

$$\begin{aligned}
\mathrm{F}(A \mid B) &= \mathrm{F}([A_0|B_0, A_0|B_1 + A_1|B_0 + A_1|B_1]) \\
&= [\mathrm{F}(A_0|B_0),\ \mathrm{F}(A_0|B_1) + \mathrm{F}(A_1|B_0) + \mathrm{F}(A_1|B_1) + \mathrm{F}(A_0|B_0)] \\
&= [\mathrm{F}(A_0) \cdot \mathrm{F}(B_0),\ \mathrm{F}(A_0) \cdot \mathrm{F}(B_0) + \mathrm{F}(A_0) \cdot \mathrm{F}(B_1) + \mathrm{F}(A_1) \cdot \mathrm{F}(B_0) + \mathrm{F}(A_1) \cdot \mathrm{F}(B_1)]
\end{aligned}$$

我们假设结论在 序列长度为 $\frac{|A|}{2}$ 时 成立，能够推出 序列长度为 $|A|$ 时 成立，就能够推出上述结论在任何情况下均适用（数学归纳法）。

考虑如何构造逆变换 iFWT 的运算规则。相当于每一步都反向操作。

因为：

$$\mathrm{FWT}(A) = \mathrm{merge}(\mathrm{FWT}(A_0), \mathrm{FWT}(A_0) + \mathrm{FWT}(A_1))$$

相当于，现在已经得到了 $A'_0 = A_0$ ，$A'_1 = A_0 + A_1$ 那么易知：

$$A_0 = A'_0$$

$$A_1 = A'_1 - A'_0$$

因此，逆变换就是：

$$\mathrm{iFWT}(A) = \mathrm{merge}(\mathrm{iFWT}(A_0), \mathrm{iFWT}(A_1) - \mathrm{iFWT}(A_0))$$

需要注意的是：

$$\mathrm{FWT}(A)_i = \sum_{j \subseteq i} A_j$$

与运算

$$\mathrm{FWT}(A) = \mathrm{merge}(\mathrm{FWT}(A_0) + \mathrm{FWT}(A_1), \mathrm{FWT}(A_1))$$

$$\mathrm{iFWT}(A) = \mathrm{merge}(\mathrm{iFWT}(A_0) - \mathrm{iFWT}(A_1), \mathrm{iFWT}(A_1))$$

需要注意的是：

$$\mathrm{FWT}(A)_i = \sum_{i \subseteq j} A_j$$

一般方法　构造的逆运算为解方程。

考虑如何根据一种位运算，求出一种合法的 FWT / iFWT 构造方式。这里的合法指这种构造方式满足 (eq.~@eq:FWT)

考虑某种位运算 $\oplus$ 的 FWT　应该是什么样子。根据上面的两个例子，可以设出如下式子：

$$\mathrm{F}(A) = \mathrm{merge}(a \cdot \mathrm{F}(A_0) + b \cdot \mathrm{F}(A_1)\ , c \cdot \mathrm{F}(A_0) + d \cdot \mathrm{F}(A_1))$$

这里的 a, b, c, d 为常数，“$\cdot$” 为普通乘法。

为了方便，设 $U = \mathrm{F}(A_0)$，$V = \mathrm{F}(A_1)$，$W = \mathrm{F}(B_0)$，$X = \mathrm{F}(B_1)$

则：

$$\mathrm{F}(A) \cdot \mathrm{F}(B) = [aU + bV, cU + dV] \cdot [aW + bX, cW + dX]$$

$$= [a^2UW + 2ab(UX + VW) + b^2VX, c^2UW + 2cd(UX + VW) + d^2VX \ \ ]$$

{#eq:FWTCOM}

这里以异或为例，因为序列是中间分成两个序列，不妨设序列长度均为 2 的若干次方，那么分开之后，其下标的最高位一定是前一半为 0 后一半为 1，根据异或的运算规则，哪些元素组合起来能够什么样的最高位即可。

假设上式在 序列长度为 $\frac{|A|}{2}$ 时是成立的。则：

$$\mathrm{F}(A \oplus B) = \mathrm{F}([A_0 \oplus B_0 + A_1 \oplus B_1, A_0 \oplus B_1 + A_1 \oplus B_0])$$

$$= [a\,\mathrm{F}(A_0 \oplus B_0 + A_1 \oplus B_1) + b\,\mathrm{F}(A_0 \oplus B_1 + A_1 \oplus B_0), c\,\mathrm{F}(A_0 \oplus B_0 + A_1 \oplus B_1) + d\,\mathrm{F}(A_0 \oplus B_1 + A_1 \oplus B_0)]$$

$$= [a\,\mathrm{F}(A_0 \oplus B_0) + a\,\mathrm{F}(A_1 \oplus B_1) + b\,\mathrm{F}(A_0 \oplus B_1) + b\,\mathrm{F}(A_1 \oplus B_0), c\,\mathrm{F}(A_0 \oplus B_0) + c\,\mathrm{F}(A_1 \oplus B_1) + d\,\mathrm{F}(A_0 \oplus B_1) + d$$

$$= [a\,\mathrm{F}(A_0)\,\mathrm{F}(B_0) + a\,\mathrm{F}(A_1)\,\mathrm{F}(B_1) + b\,\mathrm{F}(A_0)\,\mathrm{F}(B_1) + b\,\mathrm{F}(A_1)\,\mathrm{F}(B_0), c\,\mathrm{F}(A_0)\,\mathrm{F}(B_0) + c\,\mathrm{F}(A_1)\,\mathrm{F}(B_1) + d\,\mathrm{F}(A_0)\,\mathrm{F}(B$$

$$= [aUW + aVX + bUX + bVW\ ,\ cUW + cVX + dUX + dVW]$$

和 (eq.~@eq:FWTCOM) 对齐系数可以得到如下方程组：

$$\begin{cases} a = a^2 \\ a = b^2 \\ b = 2ab \\ c = c^2 \\ c = d^2 \\ d = 2cd \end{cases}$$

解上面的方程组即可，显然有许多解。但是考虑到不仅需要 FWT ，还需要 iFWT。有些解无法保证 iFWT 能够存在解。

由：

$$\mathrm{F}(A) = \mathrm{merge}(a \cdot \mathrm{F}(A_0) + b \cdot \mathrm{F}(A_1)\ , c \cdot \mathrm{F}(A_0) + d \cdot \mathrm{F}(A_1))$$

设 $X = a \cdot \mathrm{F}(A_0) + b \cdot \mathrm{F}(A_1)$，$Y = c \cdot \mathrm{F}(A_0) + d \cdot \mathrm{F}(A_1)$。问题变成了已知 X, Y，求出 $\mathrm{F}(A_0), \mathrm{F}(A_1)$，可以解得：

$$\mathrm{F}(A_1) = \frac{aY - cX}{da - bc}$$

$$\mathrm{F}(A_0) = \frac{dX - bY}{da - bc}$$

因此上述方程组中，解还需要需要保证 $ad \neq bc$。

可以解出一如下两组解：

$$\begin{cases} a = 1 \\ b = -1 \\ c = 1 \\ d = 1 \end{cases}$$

$$\begin{cases} a = 1 \\ b = 1 \\ c = 1 \\ d = -1 \end{cases}$$

上述两组解带入后都可以实现异或卷积。

质数表　来自 min_25 的博客 Orz

最长周期 n	质数	原根	$z(z^n=1)$	$p-1$ 的因数分解
2^{26}	469762049	3	2187	$2^{26}\times 7$
2^{25}	167772161	3	243	$2^{25}\times 5$
2^{24}	754974721	11	739831874	$2^{24}\times 3^2\times 5$
2^{23}	377487361	7	48510621	$2^{23}\times 3^2\times 5$
2^{23}	595591169	3	361399025	$2^{23}\times 71$
2^{23}	645922817	3	224270701	$2^{23}\times 7\times 11$
2^{23}	880803841	26	273508579	$2^{23}\times 3\times 5\times 7$
2^{23}	897581057	3	872686320	$2^{23}\times 107$
2^{23}	998244353	3	15311432	$2^{23}\times 7\times 17$
2^{22}	104857601	3	39193363	$2^{22}\times 5^2$
2^{22}	113246209	7	58671006	$2^{22}\times 3^3$
2^{22}	138412033	5	99040867	$2^{22}\times 3\times 11$
2^{22}	155189249	6	14921912	$2^{22}\times 37$
2^{22}	163577857	23	121532577	$2^{22}\times 3\times 13$
2^{22}	230686721	6	71750113	$2^{22}\times 5\times 11$
2^{22}	415236097	5	73362476	$2^{22}\times 3^2\times 11$
2^{22}	666894337	5	147340140	$2^{22}\times 3\times 53$
2^{22}	683671553	3	236932120	$2^{22}\times 163$
2^{22}	918552577	5	86995699	$2^{22}\times 3\times 73$
2^{22}	935329793	3	86363943	$2^{22}\times 223$
2^{22}	943718401	7	754500478	$2^{22}\times 3^2\times 5^2$
2^{22}	985661441	3	79986183	$2^{22}\times 5\times 47$
2^{21}	111149057	3	60767546	$2^{21}\times 53$
2^{21}	132120577	5	102376994	$2^{21}\times 3^2\times 7$
2^{21}	136314881	3	2981173	$2^{21}\times 5\times 13$
2^{21}	169869313	5	143354861	$2^{21}\times 3^4$
2^{21}	186646529	3	88383805	$2^{21}\times 89$
2^{21}	199229441	3	174670364	$2^{21}\times 5\times 19$
2^{21}	211812353	3	113852926	$2^{21}\times 101$
2^{21}	249561089	3	61724276	$2^{21}\times 7\times 17$
2^{21}	257949697	5	186470816	$2^{21}\times 3\times 41$
2^{21}	270532609	22	74891632	$2^{21}\times 3\times 43$
2^{21}	274726913	3	255478716	$2^{21}\times 131$
2^{21}	383778817	5	324881819	$2^{21}\times 3\times 61$
2^{21}	387973121	6	124477810	$2^{21}\times 5\times 37$
2^{21}	459276289	11	238723101	$2^{21}\times 3\times 73$
2^{21}	463470593	3	428228038	$2^{21}\times 13\times 17$
2^{21}	576716801	6	153098993	$2^{21}\times 5^2\times 11$
2^{21}	597688321	11	395834143	$2^{21}\times 3\times 5\times 19$
2^{21}	635437057	11	171402456	$2^{21}\times 3\times 101$
2^{21}	639631361	6	432237000	$2^{21}\times 5\times 61$
2^{21}	648019969	17	592437138	$2^{21}\times 3\times 103$

最长周期 n	质数	原根	$z(z^n=1)$	$p-1$ 的因数分解
2^{21}	710934529	17	69533131	$2^{21}\times 3\times 113$
2^{21}	715128833	3	355872337	$2^{21}\times 11\times 31$
2^{21}	740294657	3	237508734	$2^{21}\times 353$
2^{21}	786432001	7	228383098	$2^{21}\times 3\times 5^3$
2^{21}	799014913	13	374051146	$2^{21}\times 3\times 127$
2^{21}	824180737	5	133412682	$2^{21}\times 3\times 131$
2^{21}	899678209	7	118485495	$2^{21}\times 3\times 11\times 13$
2^{21}	924844033	5	44009197	$2^{21}\times 3^2\times 7^2$
2^{21}	950009857	7	741494216	$2^{21}\times 3\times 151$
2^{21}	962592769	7	695637473	$2^{21}\times 3^3\times 17$
2^{21}	975175681	17	518451017	$2^{21}\times 3\times 5\times 31$
2^{21}	1004535809	3	702606812	$2^{21}\times 479$
2^{21}	1012924417	5	673144645	$2^{21}\times 3\times 7\times 23$