Relaxed Multiplication

using

the Middle Product

BY

Joris van der Hoeven

ISSAC '03

Philadelphia

Outline

Multiplication of formal power series

Input : $\begin{cases} f = f_0 + \cdots + f_{n-1}\, z^{n-1} \\ g = g_0 + \cdots + g_{n-1}\, z^{n-1} \end{cases}$

Output : $h = h_0 + \cdots + h_{n-1}\, z^{n-1} = f\, g + O(z^n)$.

Classical multiplication algorithms

- Naive multiplication: $O(n^2)$.
- Divide & conquer: $O(n^{\log_2 3})$.
- F.F.T. multiplication: $O(n \log n \log \log n)$.

2. The lazy approach

Principle

Consider the power series as flows of coefficients. The coefficients are computed one by one and at each step we only perform the strictly necessary operations.

Implementation

A foral power series f is an algorithm which takes nothing input and outputs its first coefficient f_0 and the "remainder" $(f - f_0)/z$.

Important consequence

We compute $(f\,g)_n$ as soon as $f_0, \ldots, f_n$ and $g_0, \ldots, g_n$ are known. In particular, f_{n+1} and g_{n+1} may depend on $(f\,g)_0, \ldots, (f\,g)_n$.

The lazy approach II

Application

Computing the exponential $g = e^f$ of a series f by

$$g = \int f' g.$$

Disadvantage

Impossible to use F.F.T. or divide & conquer multiplication.

Idea : anticipation $\longrightarrow$ acceleration

Naive algorithm

g_2			
g_1			
g_0	0		
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = f_0\, g_1 + f_1\, g_0$.

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

Relaxed algorithm

g_2			
g_1			
g_0	0		
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = (f_0 + f_1)\,(g_0 + g_1) - f_0\, g_0 - f_1\, g_1$

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

3. The relaxed approach

Idea : anticipation $\longrightarrow$ acceleration

Naive algorithm

g_2			
g_1	1		
g_0	0	1	
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = f_0\, g_1 + f_1\, g_0$.

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

Relaxed algorithm

g_2			
g_1			
g_0	0		
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = (f_0 + f_1)\,(g_0 + g_1) - f_0\, g_0 - f_1\, g_1$

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

Idea : anticipation $\longrightarrow$ acceleration

Naive algorithm

g_2	2		
g_1	1	2	
g_0	0	1	2
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = f_0\, g_1 + f_1\, g_0$.

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

Relaxed algorithm

g_2			
g_1			
g_0	0		
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = (f_0 + f_1)\,(g_0 + g_1) - f_0\, g_0 - f_1\, g_1$

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

Idea : anticipation $\longrightarrow$ acceleration

Naive algorithm

g_2	2		
g_1	1	2	
g_0	0	1	2
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = f_0\, g_1 + f_1\, g_0$.

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

Relaxed algorithm

g_2			
g_1			
g_0	0		
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = (f_0 + f_1)\,(g_0 + g_1) - f_0\, g_0 - f_1\, g_1$

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

3. The relaxed approach

Idea : anticipation $\longrightarrow$ acceleration

Naive algorithm

g_2	2		
g_1	1	2	
g_0	0	1	2
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = f_0\, g_1 + f_1\, g_0$.

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

Relaxed algorithm

g_2			
g_1	1	1	
g_0	0	1	
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = (f_0 + f_1)\,(g_0 + g_1) - f_0\, g_0 - f_1\, g_1$

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

3. The relaxed approach

Idea : anticipation $\longrightarrow$ acceleration

Naive algorithm

g_2	2		
g_1	1	2	
g_0	0	1	2
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = f_0\, g_1 + f_1\, g_0$.

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

Relaxed algorithm

g_2	2		
g_1	1	1	
g_0	0	1	2
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = (f_0 + f_1)\,(g_0 + g_1) - f_0\, g_0 - f_1\, g_1$

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

Idea : anticipation $\longrightarrow$ acceleration

Naive algorithm

g_2	2		
g_1	1	2	
g_0	0	1	2
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = f_0\, g_1 + f_1\, g_0$.

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

Relaxed algorithm

g_2	2		
g_1	1	1	
g_0	0	1	2
$\times$	f_0	f_1	f_2

0 $h_0 = f_0\, g_0$.

1 $h_1 = (f_0 + f_1)\,(g_0 + g_1) - f_0\, g_0 - f_1\, g_1$

2 $h_2 = f_0\, g_2 + f_1\, g_1 + f_2\, g_0$.

Divide & conquer algorithm

The divide & conquer algorithm is "essentially relaxed": the formula for h_k only depends on $f_0, \ldots, f_k$ and $g_0, \ldots, g_k$.

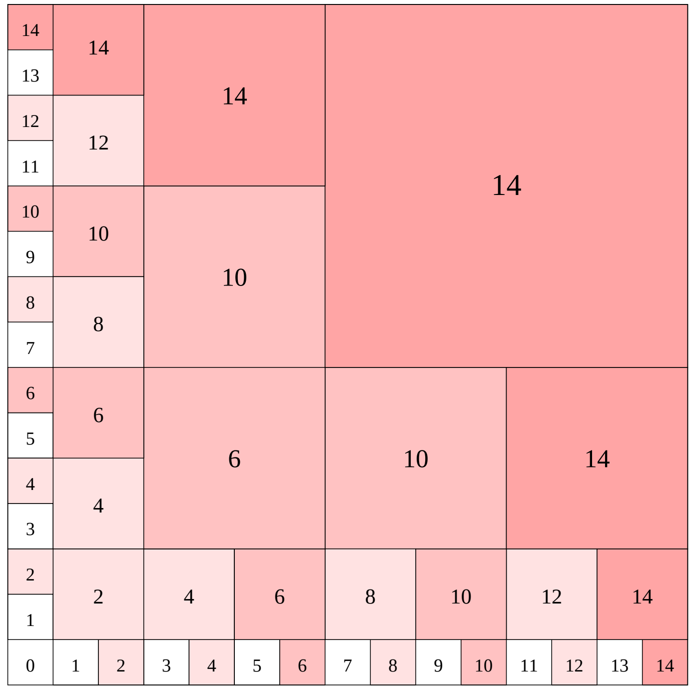
Example : multiplication at order 4

- $h_0 = f_0\, g_0$;
- $h_1 = (f_0 + f_1)\,(g_0 + g_1) - f_0\, g_0 - f_1\, g_1$;
- $h_2 = (f_0 + f_2)\,(g_0 + g_2) - f_0\, g_0 - f_2\, g_2$;
- $h_3 = (f_0 + f_1 + f_2 + f_3)\,(g_0 + g_1 + g_2 + g_3) - (f_0 + f_1)\,(g_0 + g_1) - (f_2 + f_3)\,(g_2 + g_3) + f_0\, g_0 + f_1\, g_1 + f_2\, g_2 + f_3\, g_3$;
- $h_4 = (f_1 + f_3)\,(g_1 + g_3) - f_1\, g_1 - f_3\, g_3$;
- $h_5 = (f_2 + f_3)\,(g_2 + g_3) - f_2\, g_3 - f_2\, g_3$;
- $h_6 = f_3\, g_3$.

g_3	3	3	3	3
g_2	2	3	2	3
g_1	1	1	3	3
g_0	0	1	2	3
$\times$	f_0	f_1	f_2	f_3

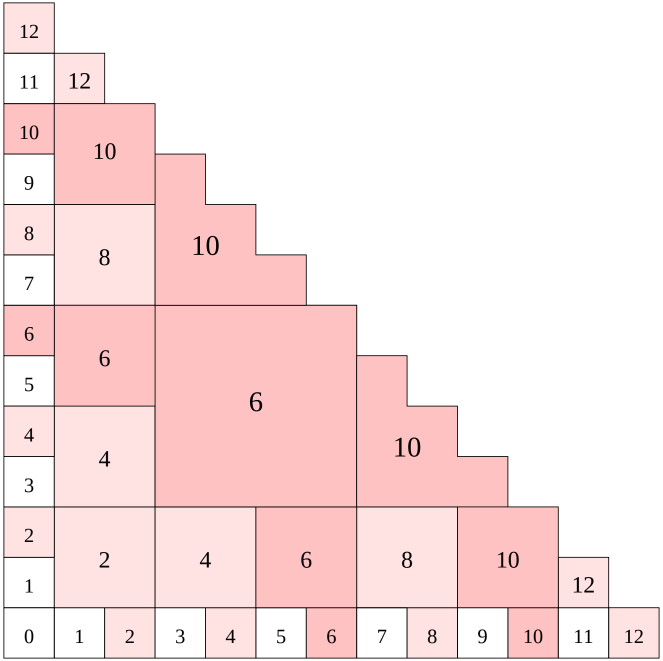
Algorithm in time $K(n)$ and space $O(n \log n)$.

Fast relaxed multiplication



$\longrightarrow$ Relaxed algorithm in time $O(M(n) \log n)$ and space $O(n)$.

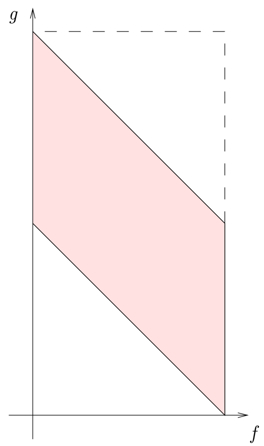
Truncated multiplication



4. The middle product (HQZ)

Given $f = f_0 + \ldots + f_{n-1}\, z^{n-1}$ and $g = g_0 + \cdots + g_{2n-2}\, z^{2n-2}$, compute $h = f * g = h_0 + \cdots + h_{n-1}\, z^{n-1}$ with

$$h_i = \sum_{j=0}^{n-1} f_j\, g_{n-1+i-j}$$

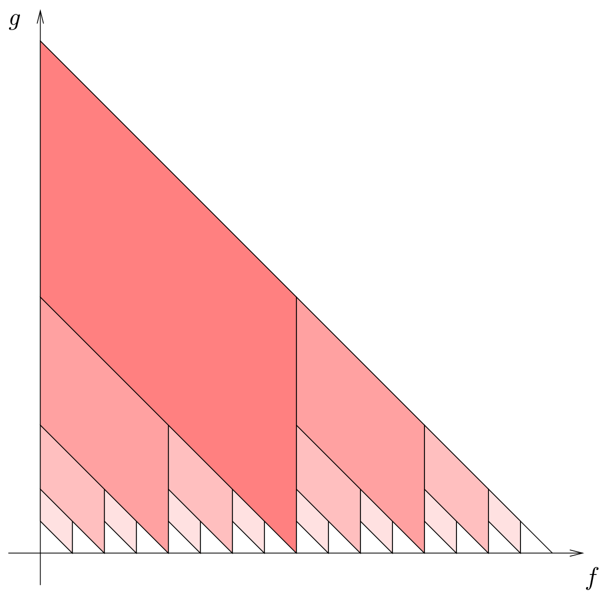


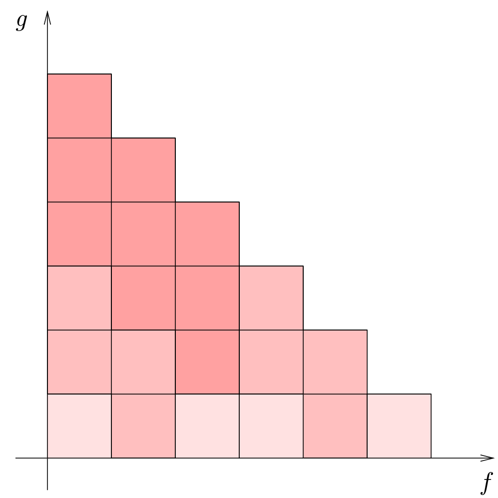
Case $n = 2$

$$\begin{aligned} \alpha &= f_1\,(g_0 + g_1) \\ \beta &= (f_1 - f_0)\, g_1 \\ \gamma &= f_0\,(g_1 + g_2) \\ h_0 &= \alpha - \beta \\ h_1 &= \gamma + \beta \end{aligned}$$

5. Relaxed multiplication with fixed series

For *fixed* $g = g_0 + \cdots + g_{n-1} z^{n-1}$ and *relaxed* $f = f_0 + \cdots + f_{n-1} z^{n-1}$, compute the *relaxed* product $h = f g = h_0 + \cdots + h_{n-1} z^{n-1}$.





Complexity results

Theorem 1. *There exists a (truncated) relaxed multiplication algorithm with one fixed argument, with the same time and space complexity as divide & conquer multiplication.*

Theorem 2. *There exists a (truncated) relaxed multiplication algorithm with one fixed argument, which is twice as efficient as the standard fast relaxed multiplication algorithm from an asymptotic point of view.*

Linear differential equations

Consider

$$L_r f^{(r)} + \cdots + L_0 f = 0,$$

with $L_0, \ldots, L_r \in C[[z]]$, $L_{r,0} = 1$, and given $f_0, \ldots, f_{r-1}$. Then the unique solution can be computed using the formula

$$f = L_r^{-1} \int \overset{r\times}{\cdots} \int (L_r f^{(r)} + \cdots + L_0 f)$$

in time $\sim (r+1) K(n)$.

General implicit linear equations