L7

shelat
16f-4800
sep 30 2016

FFT, DP

Fast Fourier Transform

Your name:

What does the FFT take as input?

polynomial in coeff form

What does the FFT do?

changes to point-wise form

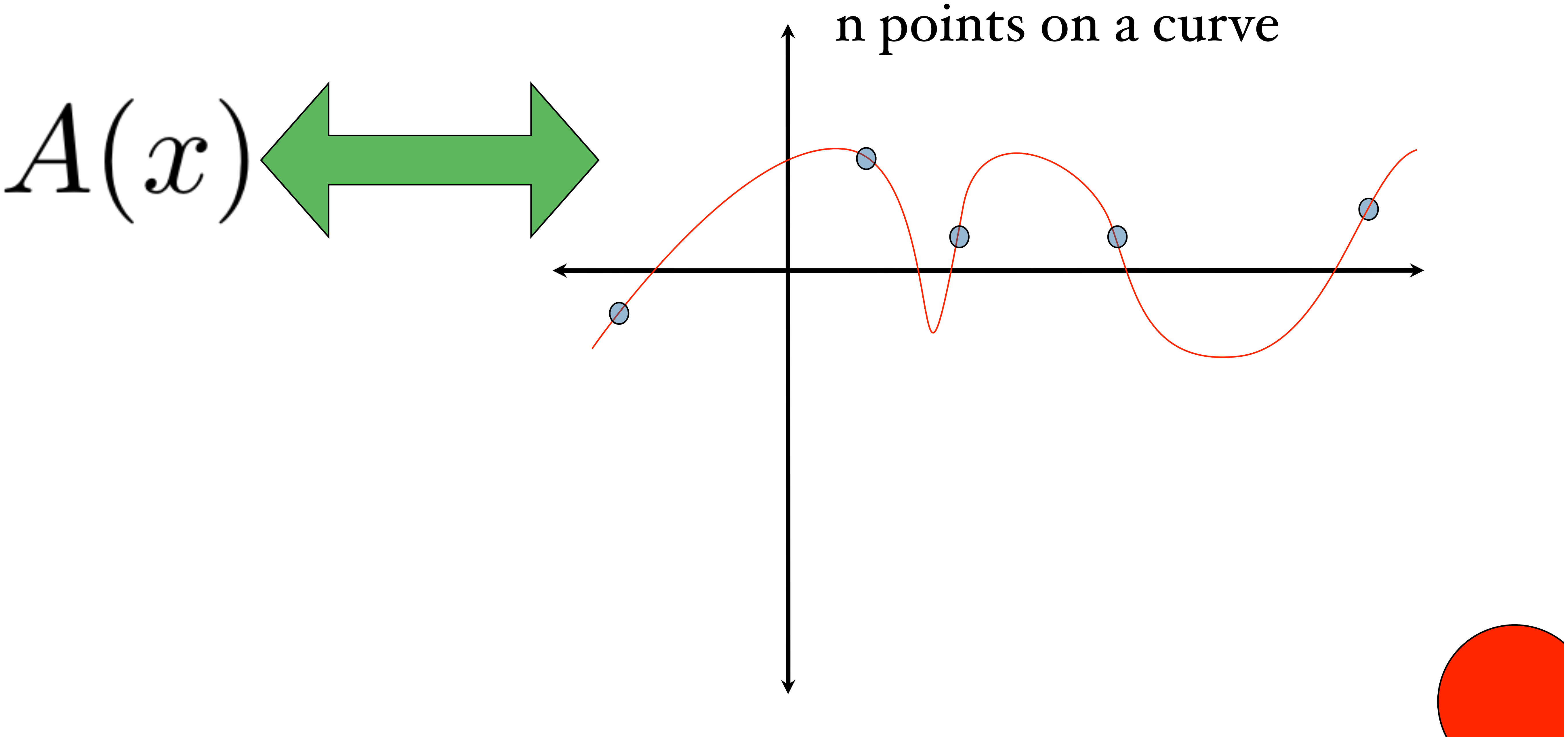
Evaluate, the polynomial of degree $n-1$ at n points

FFT

input: $a_0, a_1, a_2, \ldots, a_{n-1}$

$$A(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$$

output: evaluate polynomial A at (any) n different points. → roots of unity



$$A(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$$

$$A(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$$

$$= a_0 + a_2 x^2 + a_4 x^4 + \cdots + a_{n-2} x^{n-2}$$

$$+ a_1 x + a_3 x^3 + a_5 x^5 + \cdots + a_{n-1} x^{n-1}$$

$$A(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$$

$$= a_0 + a_2 x^2 + \underline{a_4 x}^4 + \cdots + a_{n-2} x^{n-2}$$

$$+ a_1 x + a_3 x^3 + a_5 x^5 + \cdots + a_{n-1} x^{n-1}$$

$$\underline{A_e}(x) = a_0 + a_2 x + a_4 x^2 + \cdots + a_n x^{\underline{(n-2)/2}} \longrightarrow \text{degree } \frac{n}{2} - 1$$

$$\underline{A_o}(x) = a_1 + a_3 x + a_5 x^2 + \cdots + a_{n-1} x^{(n-2)/2}$$

$$
\begin{aligned}
A(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1} \\
= a_0 + a_2 x^2 + a_4 x^4 + \cdots + a_{n-2} x^{n-2} \\
+ a_1 x + a_3 x^3 + a_5 x^5 + \cdots + a_{n-1} x^{n-1}
\end{aligned}
$$

$$
\boxed{
\begin{aligned}
A_e(x) &= a_0 + a_2 x + a_4 x^2 + \cdots + a_n x^{(n-2)/2} \\
A_o(x) &= a_1 + a_3 x + a_5 x^2 + \cdots + a_{n-1} x^{(n-2)/2}
\end{aligned}
}
$$

$$
A(x) = A_e(x^2) + x A_o(x^2)
$$

$$A(x) = A_e(x^2) + xA_o(x^2)$$

suppose we had already had eval of Ae,Ao on {4,9,16,25}

$A_e(4)$	$A_0(4)$
$A_e(9)$	$A_0(9)$
$A_e(16)$	$A_0(16)$
$A_e(25)$	$A_0(25)$

$$A(2) = A_e(2^2) + 2 \cdot A_o(2^2)$$

$$= A_e(4) + 2 \cdot A_o(4)$$

$$A(-2) = A_e(4) + (-2) \cdot A_o(4)$$

$$A(x) = A_e(x^2) + xA_o(x^2)$$

suppose we had already had eval of Ae,Ao on {4,9,16,25}

$A_e(4)$ $A_0(4)$
$A_e(9)$ $A_0(9)$
$A_e(16)$ $A_0(16)$
$A_e(25)$ $A_0(25)$

Then we could compute 8 terms:

$$A(2) = A_e(4) + 2A_o(4)$$

$$A(-2) = A_e(4) + (-2)A_o(4)$$

$$A(3) = A_e(9) + 3A_o(9)$$

$$A(-3) = A_e(9) + (-3)A_o(9)$$

...A(4), A(-4), A(5), A(-5)

FFT(f=a{1,...,n})

Evaluates degree n poly on the n^{th} roots of unity

Roots of unity

$$x^n = 1$$

should have n solutions

what are they?

Remember this?

Euler's identity

$$e^{2\pi i} = 1$$

$j = 0, 1, 2, \ldots n-1$

$$\left\{ e^{2\pi i \cdot j/n} \right\}$$

roots of unity

$$x^n = 1$$

the n solutions are:

consider $\left\{\underset{j=0}{1}, e^{2\pi i/n}, e^{2\pi i 2/n}, e^{2\pi i 3/n}, \ldots, e^{2\pi i(n-1)/n}\right\}$

$$\left[e^{2\pi i \cdot j/n}\right]^n = \left(e^{2\pi i}\right)^{(j/n)\cdot n} = 1^j = 1$$

$$x^n = 1$$

the n solutions are:

consider $e^{2\pi ij/n}$ for j=0,1,2,3,...,n-1

$$\left[e^{(2\pi i/n)j}\right]^n = \left[e^{(2\pi i/n)n}\right]^j = \left[e^{2\pi i}\right]^j = 1^j$$

$e^{2\pi ij/n}$ = $\omega_{j,n}$ is an n^th root of unity

$$\omega_{0,n}, \omega_{2,n}, \ldots, \omega_{n-1,n}$$

What is this number?

$e^{2\pi i j/n}$ = $\omega_{j,n}$ is an n^{th} root of unity

What is this number?

$e^{2\pi ij/n} = \omega_{j,n}$ is an n^{th} root of unity

$$e^{ix} = \cos(x) + i\sin(x)$$

$$e^{2\pi ij/n} = \cos(2\pi j/n) + i\sin(2\pi j/n)$$

$e^{2\pi i j/n}$ = $\omega_{j,n}$ is an n^th root of unity

$$\omega_{0,n}, \omega_{2,n}, \ldots, \omega_{n-1,n}$$

Lets compute $\omega_{1,8}$

$$\omega_{1,8} = \underline{e^{2\pi i \cdot 1/8}} = \cos(2\pi \, 1/8) + i \sin(2\pi \, 1/8)$$

$$= \cos(\pi/4) + i \cdot \sin(\pi/4)$$

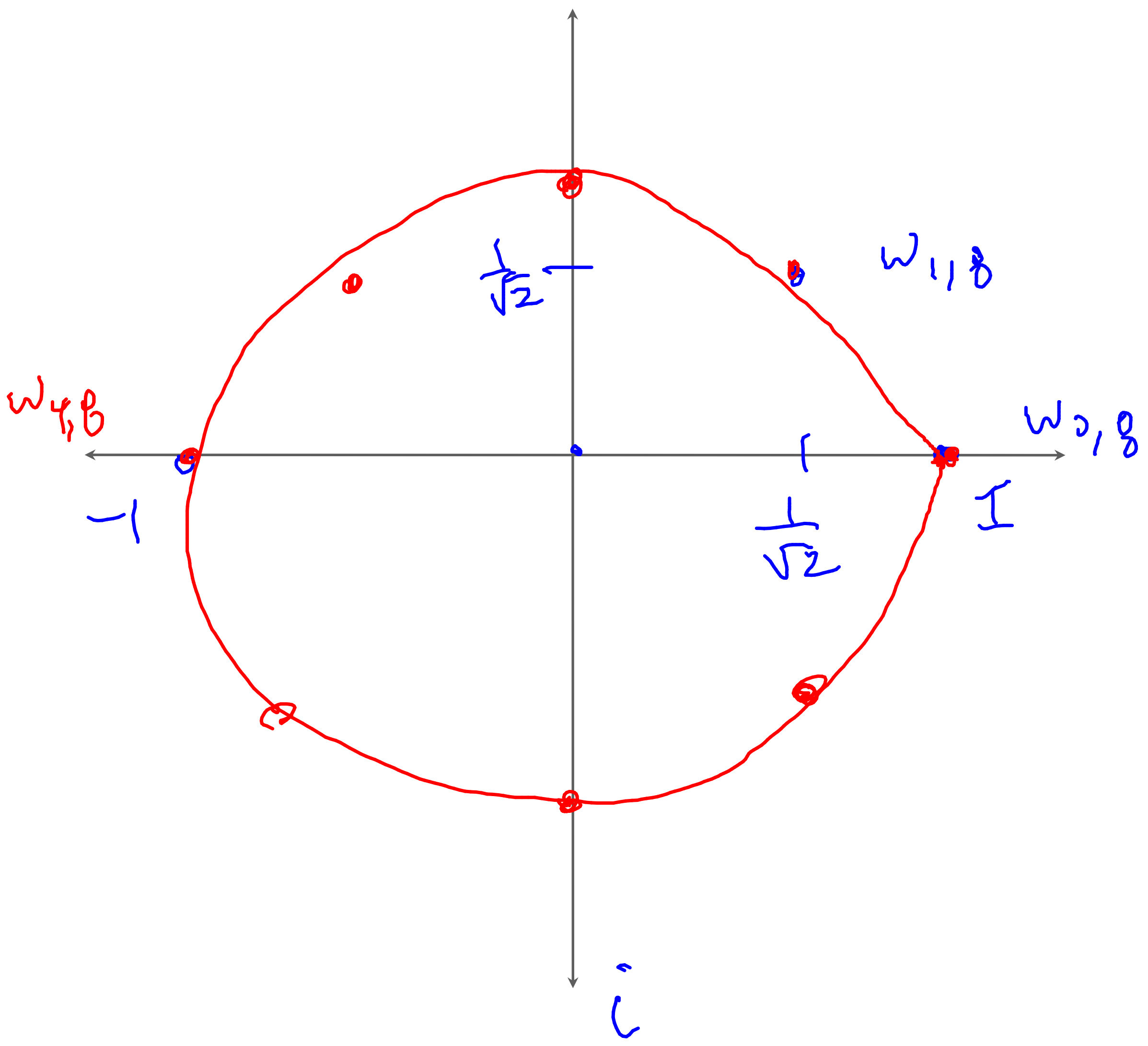
$$\frac{\sqrt{2}}{2} = \frac{1}{\sqrt{2}}$$

$$= \frac{1}{\sqrt{2}} + \frac{i}{\sqrt{2}}$$

Compute all 8 roots of unity

ω_0	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7
1	$\frac{1}{\sqrt{2}} + \frac{i}{\sqrt{2}}$	i	$-\frac{1}{\sqrt{2}} + \frac{i}{\sqrt{2}}$	-1			

$$\frac{\sqrt{2}}{2} = \frac{1}{\sqrt{2}}$$



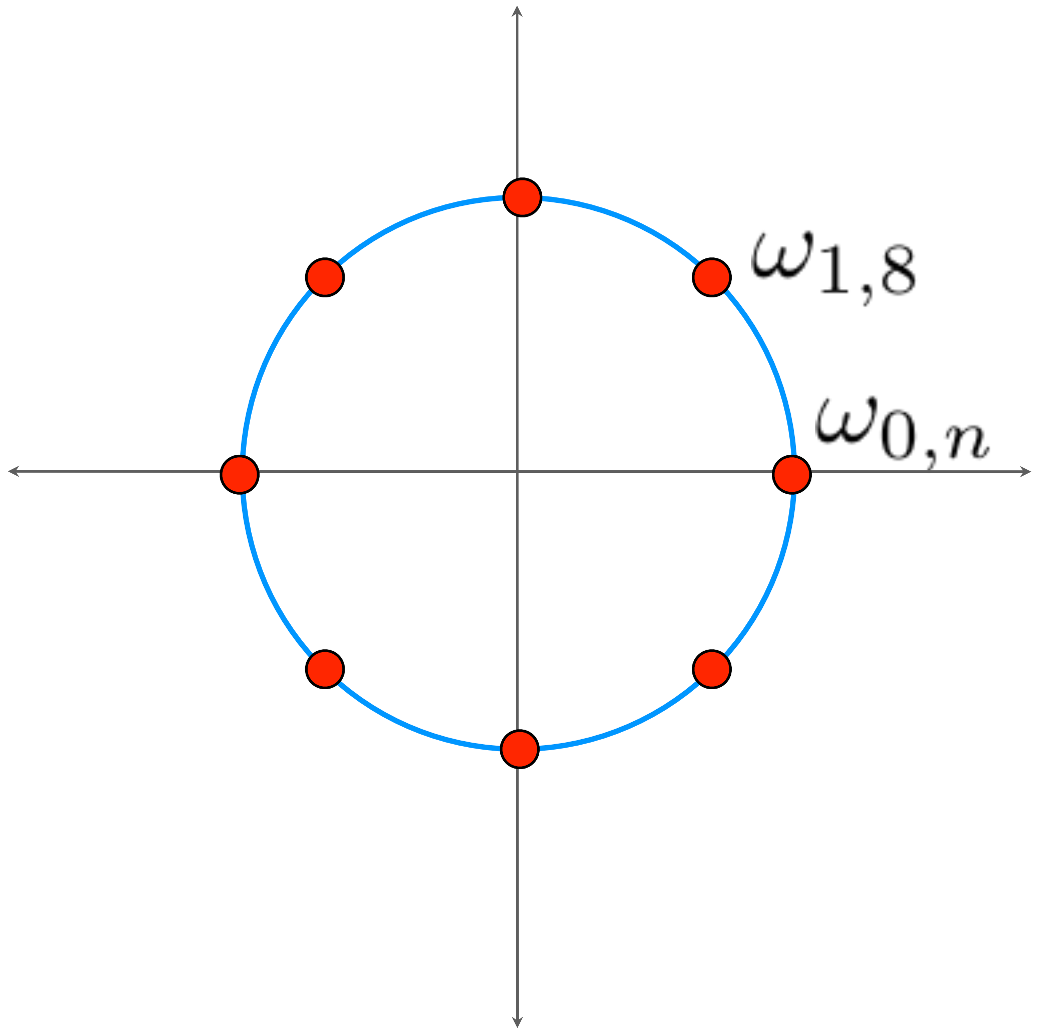
Then graph them

roots of unity

$$x^n = 1$$

should have n solutions

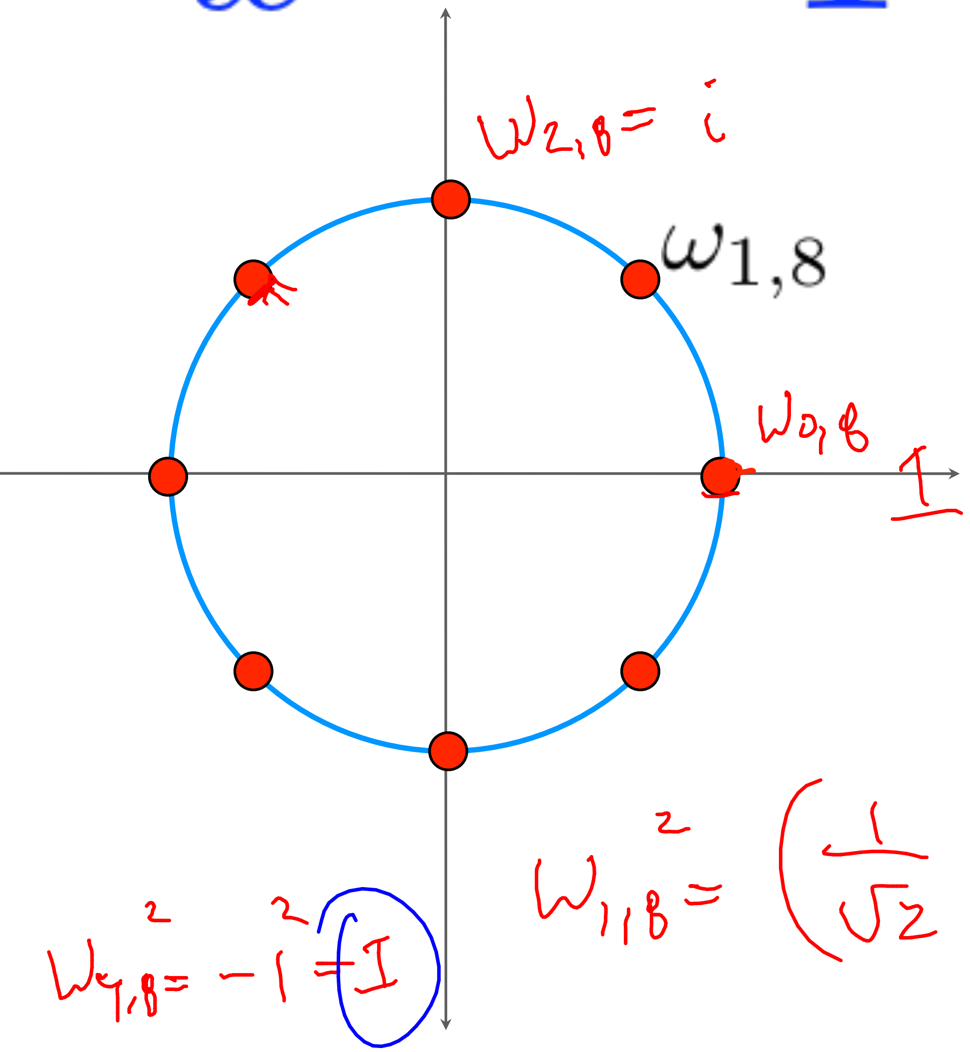
$$e^{2\pi ij/n} = \cos(2\pi j/n) + i\sin(2\pi j/n)$$

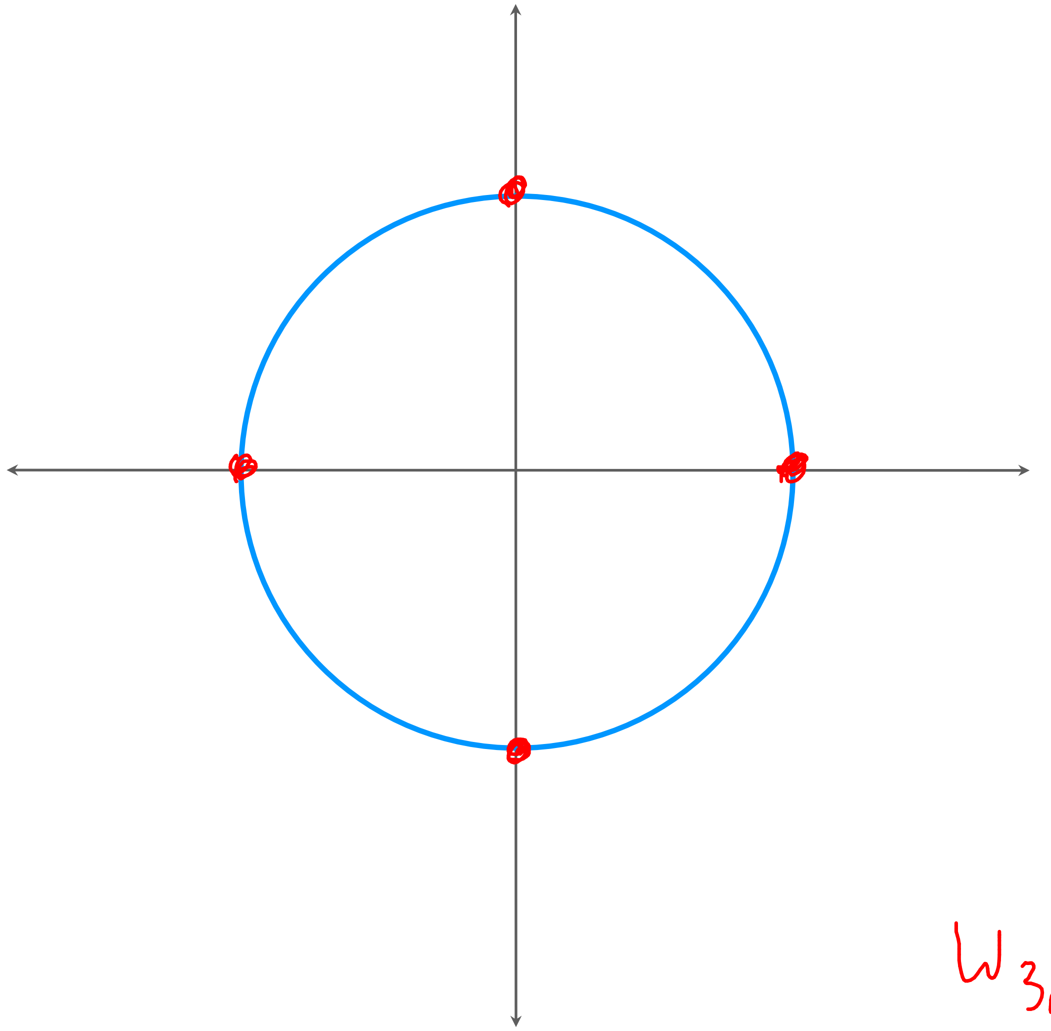


Squaring the n^{th} roots of unity

yields the n/2th roots of unity

$$x^n = 1$$





$W_{4,8}^2 = -1^2 = 1$

$W_{1,8}^2 = \left(\frac{1}{\sqrt{2}} + \frac{i}{\sqrt{2}}\right)^2 = \frac{1}{2} + 2\frac{i}{2} + \frac{i^2}{2} = \frac{1}{2} + i - \frac{1}{2} = i$

$W_{3,8} = \left(-\frac{1}{\sqrt{2}} + \frac{i}{\sqrt{2}}\right)^2 = -i$

$(w_{2,8})^2 = i^2 = -1$

Thm: Squaring an n^th^ root produces an n/2^th^ root.

example: $$\omega_{1,8} = \left(\frac{1}{\sqrt{2}} + \frac{i}{\sqrt{2}}\right)$$

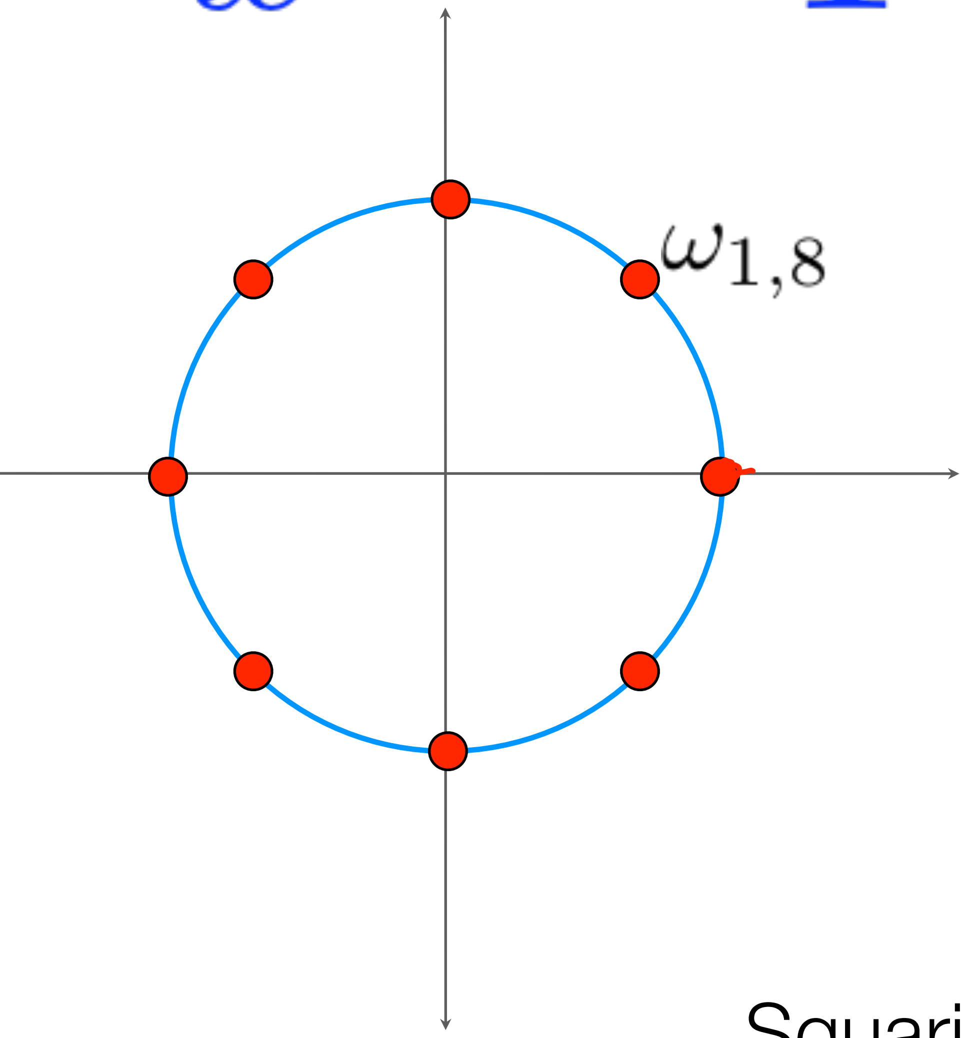
$$\omega_{1,8}^2 = \left(\frac{1}{\sqrt{2}} + \frac{i}{\sqrt{2}}\right)^2 = \left(\frac{1}{\sqrt{2}}\right)^2 + 2\left(\frac{1}{\sqrt{2}}\frac{i}{\sqrt{2}}\right) + \left(\frac{i}{\sqrt{2}}\right)^2$$

$$= 1/2 + i - 1/2$$

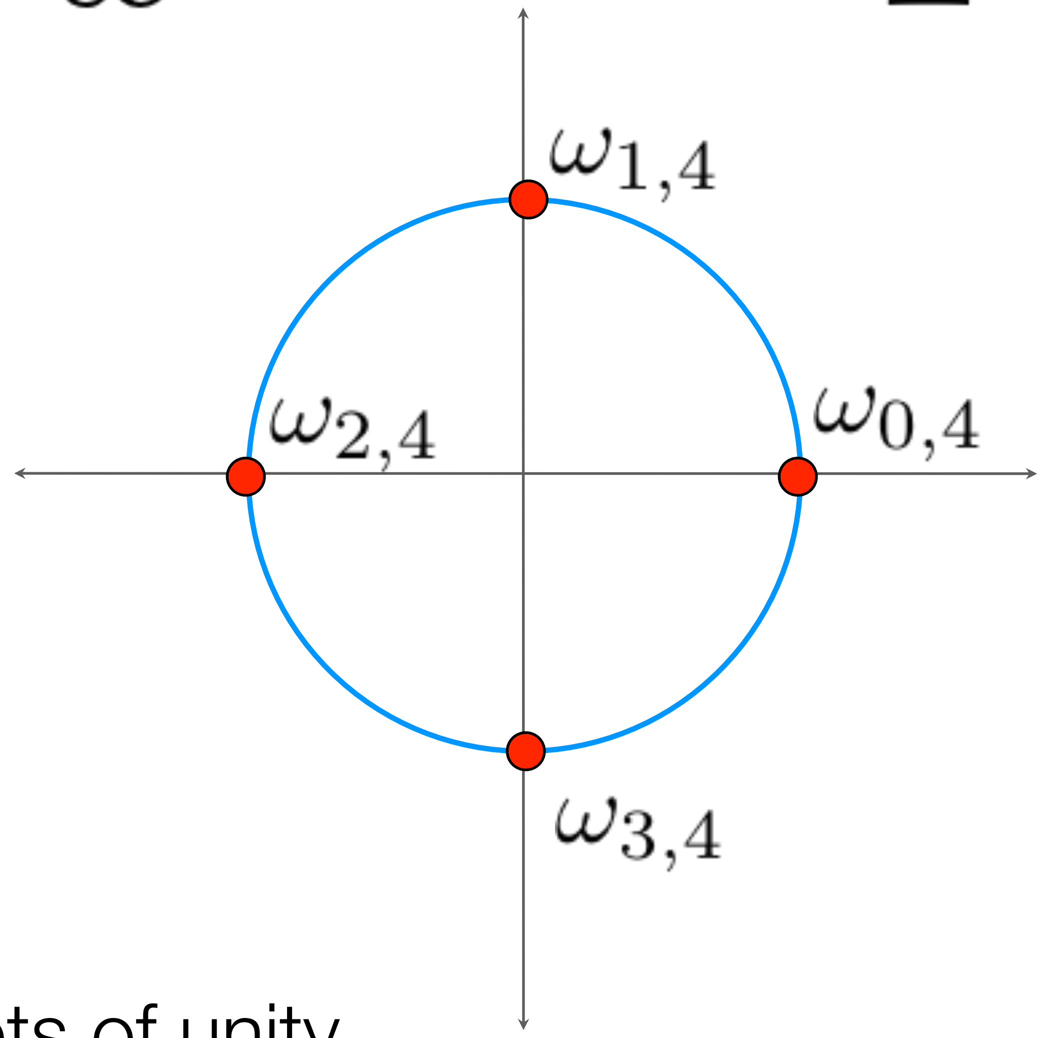
$$= i$$

Squaring the nth roots of unity

$$x^n = 1$$



$$x^{n/2} = 1$$



Squaring all of the nth roots of unity
produces the n/2th roots of unity

Thm: Squaring an nth root produces an n/2th root.

$$\left\{1^2, \left(e^{2\pi i(1/n)}\right)^2, \left(e^{2\pi i(2/n)}\right)^2, \left(e^{2\pi i(3/n)}\right)^2, \ldots, \left(e^{2\pi i(n/2)/n}\right)^2, \left(e^{2\pi i(n/2+1)/n}\right)^2, \ldots, e^{2\pi i(n-1)/n}\right\} \quad n$$

$$\left\{1 \quad e^{2\pi i \, 1/n/2} \quad e^{2\pi i \, 2/n/2} \quad e^{2\pi i \, 3/n/2}\right\} \quad 1$$

$$\left[e^{2\pi i \, 1/n}\right]^2 = e^{2\pi i \, 2/n} = e^{2\pi i \, 1/(n/2)}$$

$$(w_{3,8})^2 = \left(\frac{-1}{\sqrt{2}} + \frac{i}{\sqrt{2}}\right)^2 = \frac{1}{2} - \frac{2i}{2} + \frac{i^2}{2}$$

$$= -i$$

Thm: Squaring an n^th root produces an n/2^th root.

$$\left\{1, e^{2\pi i(1/n)}, e^{2\pi i(2/n)}, e^{2\pi i(3/n)}, \ldots, e^{2\pi i(n/2)/n}, e^{2\pi i(n/2+1)/n}, \ldots, e^{2\pi i(n-1)/n}\right\}$$

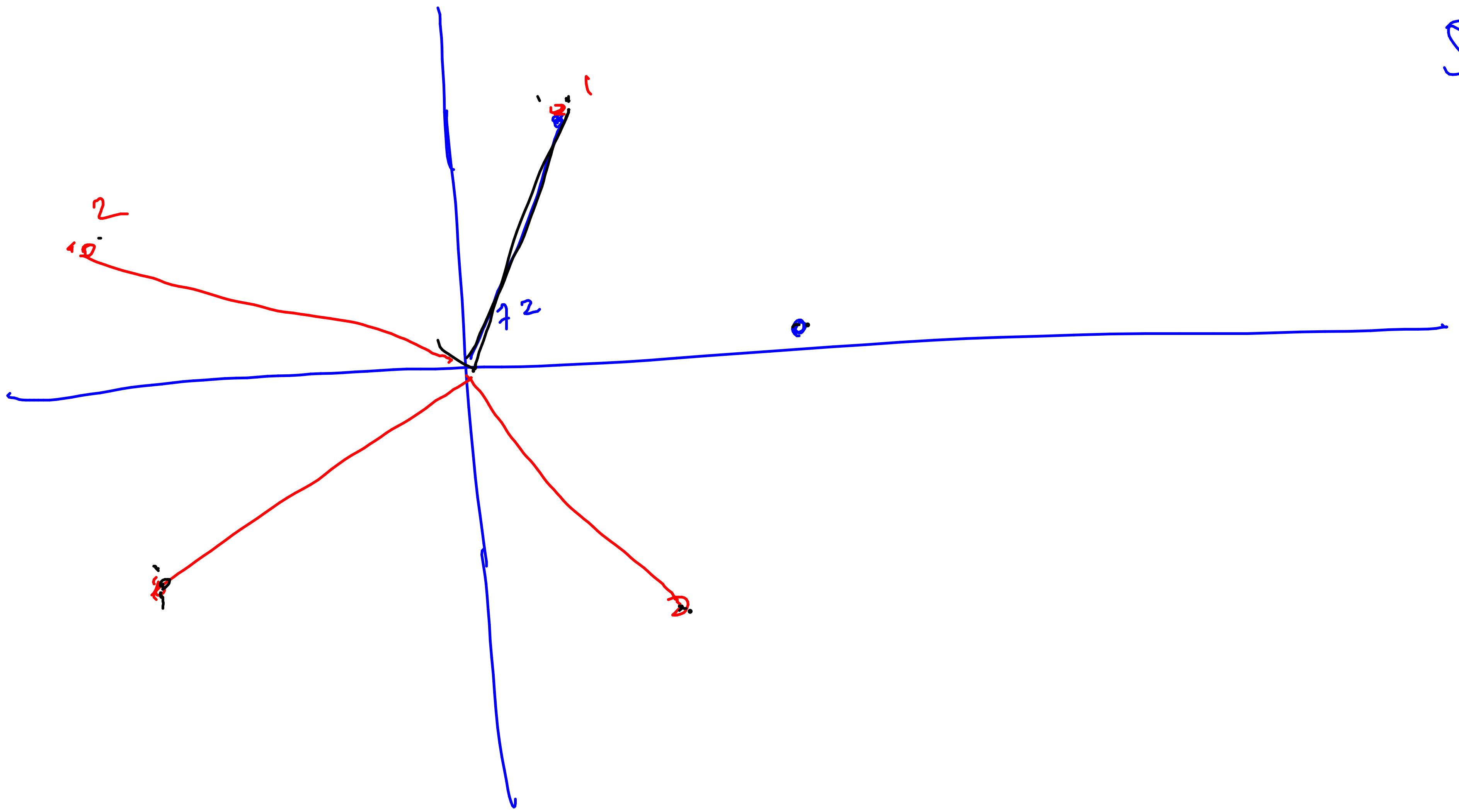
$1 \quad e^{2\pi i(1/(n/2))} \quad e^{2\pi i(2/(n/2))} \quad e^{2\pi i(3/(n/2))} \qquad 1$

$$e^{2\pi i((n/2)+1/(n/2))}$$

$$= e^{2\pi i(1+1/(n/2))}$$

$$= 1 \cdot e^{2\pi i(1/(n/2))}$$

5th roots of unity



$0 \to 0$

$1 \to 2$

$2 \to 4$

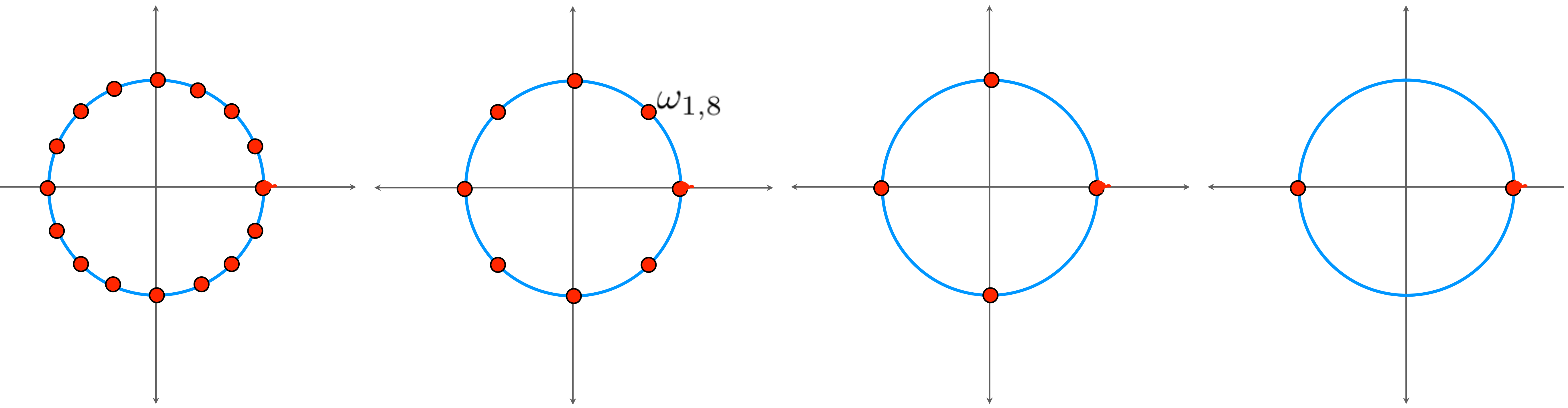
$3 \to 1$

$4 \to 3$

$$\left(w_{1/5}\right)^2 = \left(\cos\left(2\pi/5\right) + i \cdot \sin\left(2\pi/5\right)\right)^2$$

$$=$$

If n=16



$$A(x) = A_e(x^2) + xA_o(x^2)$$

evaluate at a root of unity

$$A(x) = A_e(x^2) + xA_o(x^2)$$

evaluate at a root of unity

$$A(\omega_{i,n}) = \boxed{A_e(\omega_{i,n}^2)} + \omega_{i,n} A_o(\omega_{i,n}^2)$$

n^{th} root of unity

$n/2^{th}$ root of unity

$n/2^{th}$ root of unity

recursive version of the fft

FFT(f=a{1,...,n})

Evaluates degree n poly on the n^{th} roots of unity

FFT(f=a[1,...,n])

Base case if n<=2

E[...] <- FFT(A_e) // eval Ae on n/2 roots of unity
O[...] <- FFT(A_o) // eval Ao on n/2 roots of unity

combine results using equation:

$$A(\omega_{i,n}) = A_e(\omega_{i,n}^2) + \omega_{i,n} A_o(\omega_{i,n}^2)$$

$$A(\omega_{i,n}) = A_e(\omega_{i \mod n/2, \frac{n}{2}}) + \omega_{i,n} A_o(\omega_{i \mod n/2, \frac{n}{2}})$$

Return n points.

Example

a_0 a_{n-1}

FFT(4, 1, 3, 2, 2, 3, 1, 4)

What does this function compute?

Example

FFT(4, 1, 3, 2, 2, 3, 1, 4)

What does this function compute?

$A(x) =$

It evaluates $4 + 1x + 3x^2 + 2x^3 + 2x^4 + 3x^5 + 1x^6 + 4x^7$

on the 8th roots of unity, which are

Example

$A(1) =$

FFT(4, 1, 3, 2, 2, 3, 1, 4)

What does this function compute?

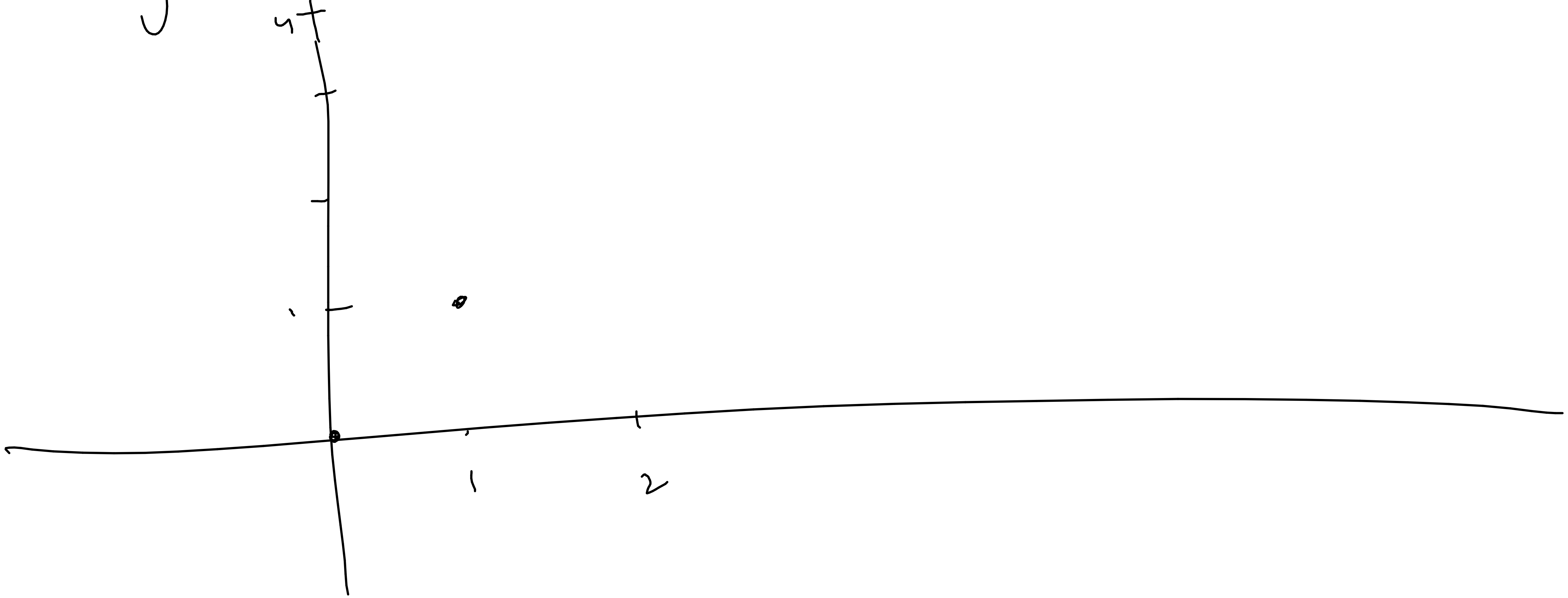
It evaluates $4 + 1x + 3x^2 + 2x^3 + 2x^4 + 3x^5 + 1x^6 + 4x^7$

on the 8th roots of unity, which are

ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7	ω_8
1	$\frac{1}{\sqrt{2}} + \frac{i}{\sqrt{2}}$	i	$\frac{-1}{\sqrt{2}} + \frac{i}{\sqrt{2}}$	-1	$\frac{-1}{\sqrt{2}} + \frac{-i}{\sqrt{2}}$	$-i$	$\frac{1}{\sqrt{2}} + \frac{-i}{\sqrt{2}}$

$f(x) = x^2 + 0x + 0$

$\underline{\begin{pmatrix} 0 & 0 & 1 \end{pmatrix}}$



$f(0) = 0$

$f(1) = 1$

$f(2) = 4$

FFT on $A(x) = 4 + 1x + 3x^2 + 2x^3 + 2x^4 + 3x^5 + 1x^6 + 4x^7$

$$A_e(x) = 4 + 3x + 2x^2 + 1x^3$$

$$A_o(x) = 1 + 2x + 3x^2 + 4x^3$$

$$FFT(A_e) \stackrel{\text{returns}}{=} \left\{ \begin{array}{cccc} 1 & i & -1 & -i \\ 10 & 2+2i & 2 & 2-2i \end{array} \right\}$$

4th roots of unity are $\{1, i, -1, -i\}$

$$FFT(A_o) \stackrel{\text{returns}}{=} \left\{ \begin{array}{cccc} 1 & i & -1 & -i \\ 10 & -2-2i & -2 & -2+2i \end{array} \right\}$$

Last step of FFT:

$T(n) = 2T(\frac{n}{2}) + \Theta(n) = \Theta(n \log n)$

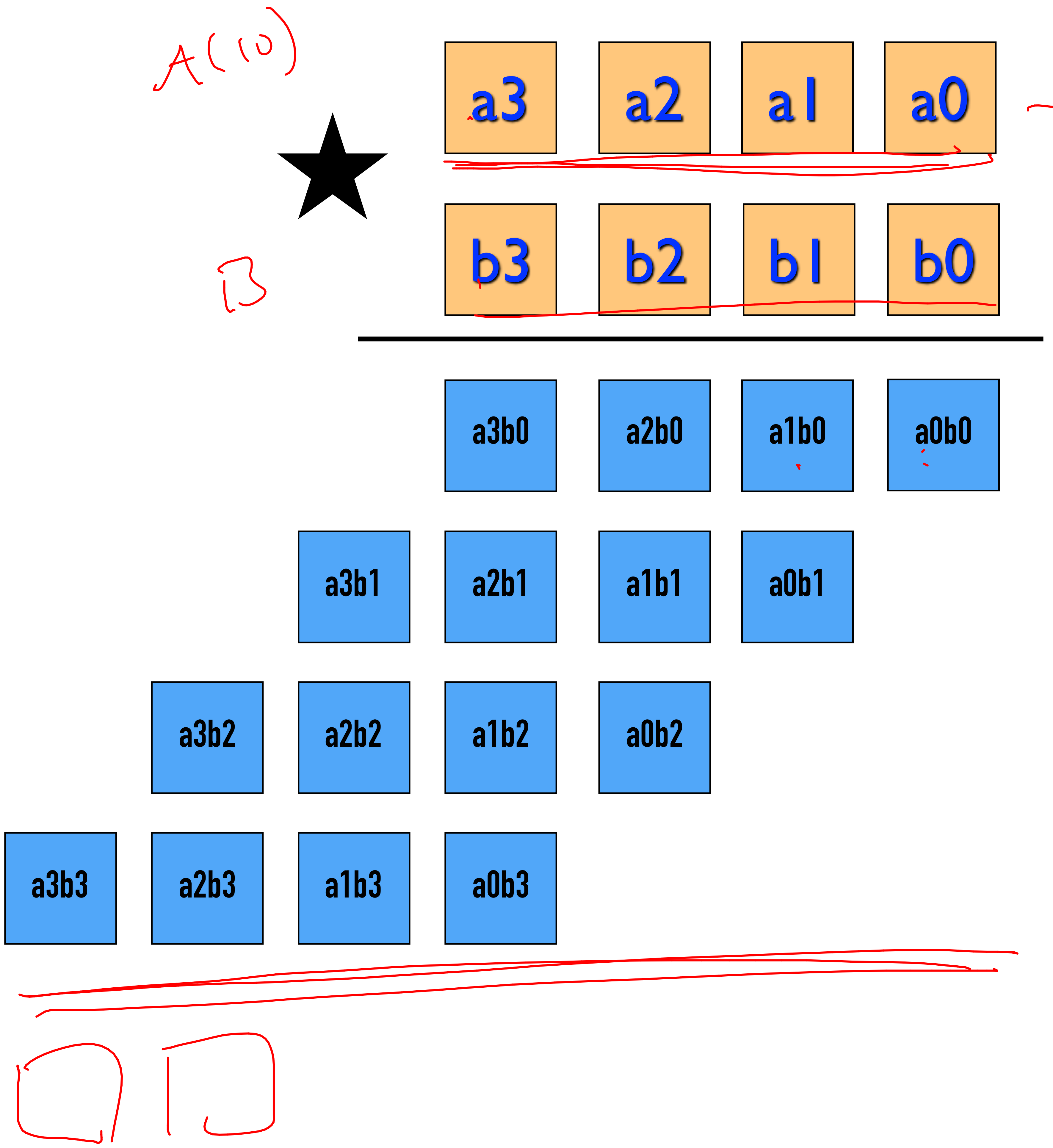
FFT(A) now returns

$$\left\{ \begin{array}{cccccc} w_{0,8} & w_{1,8} & w_{2,8} & w_{3,8} & \cdots & w_{7,8} \\ \downarrow & \downarrow & \downarrow & & & \\ A(1) = & A(w_{1,8}) = & & & & \end{array} \right\}$$

$A(1) = A_e(1) + 1 \cdot A_o(1) = 20$

$A(w_{1,8}) = \underline{A_e(w_{1,4})} + w_{1,8} \cdot A_o(w_{1,4}) = (2+2i) + w_{1,8}(-2-2i)$

$A(w_{2,8}) = A_e(w_{2,4}) + w_{2,8} \cdot A_o(w_{2,4}) = 2 + i(-2)$



$\rightarrow A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$

$$A(x) = a_3x^3 + a_2x^2 + a_1x + a_0$$

$A(10)$

$$B(x) = b_3x^3 + b_2x^2 + b_1x + b_0$$

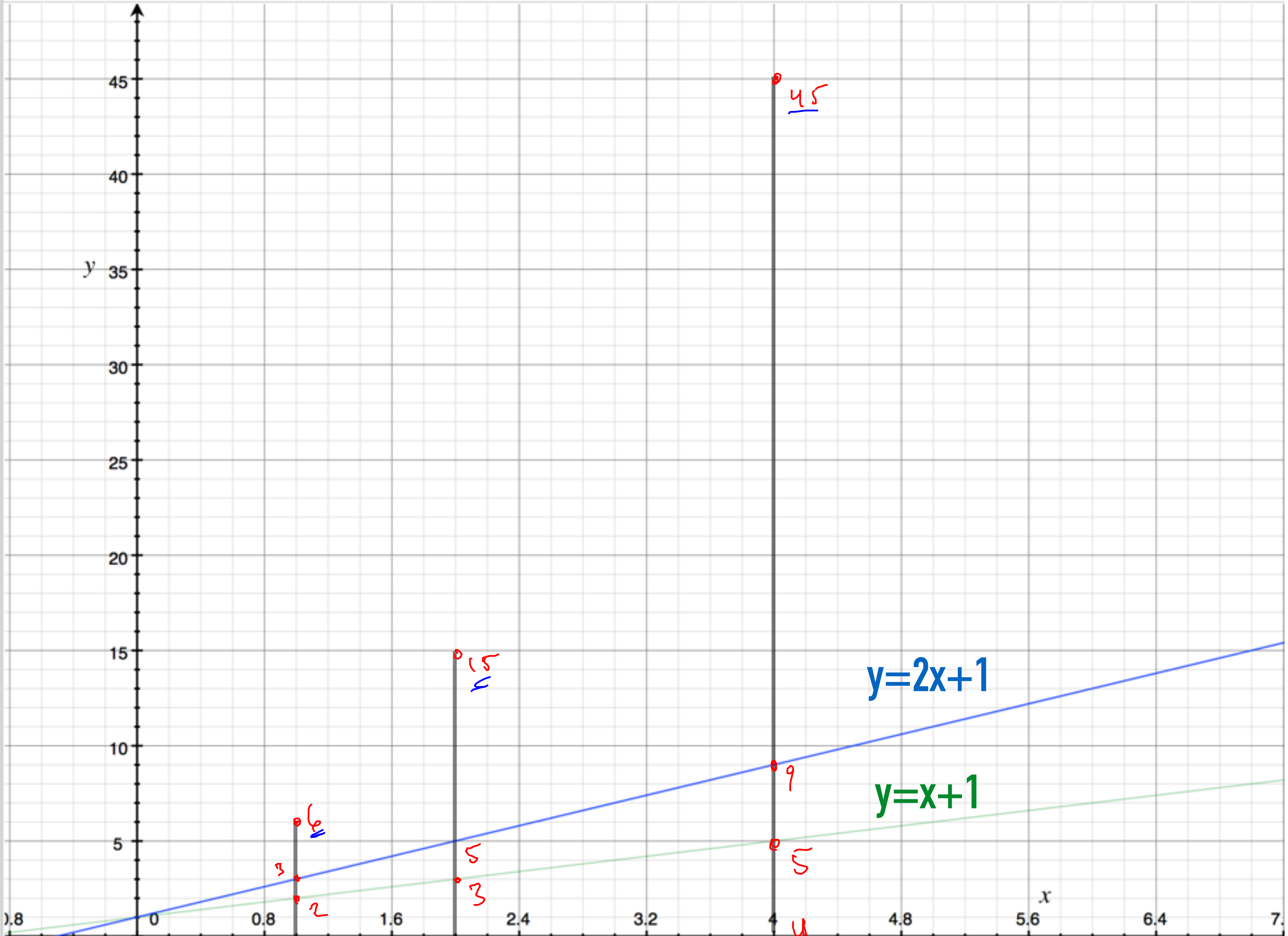
$B(10)$

$C = A \cdot B$

$$C(x) = \begin{cases} a_3b_3x^6 + \\ (a_3b_2 + a_2b_3)x^5 + \\ (a_3b_1 + a_2b_2 + a_1b_3)x^4 + \\ (a_3b_0 + a_2b_1 + a_1b_2 + a_0b_3)x^3 + \\ (a_2b_0 + a_1b_1 + a_0b_2)x^2 + \\ (a_1b_0 + a_0b_1)x + \\ a_0b_0 \end{cases}$$

$C(10)$

$y=2x+1$

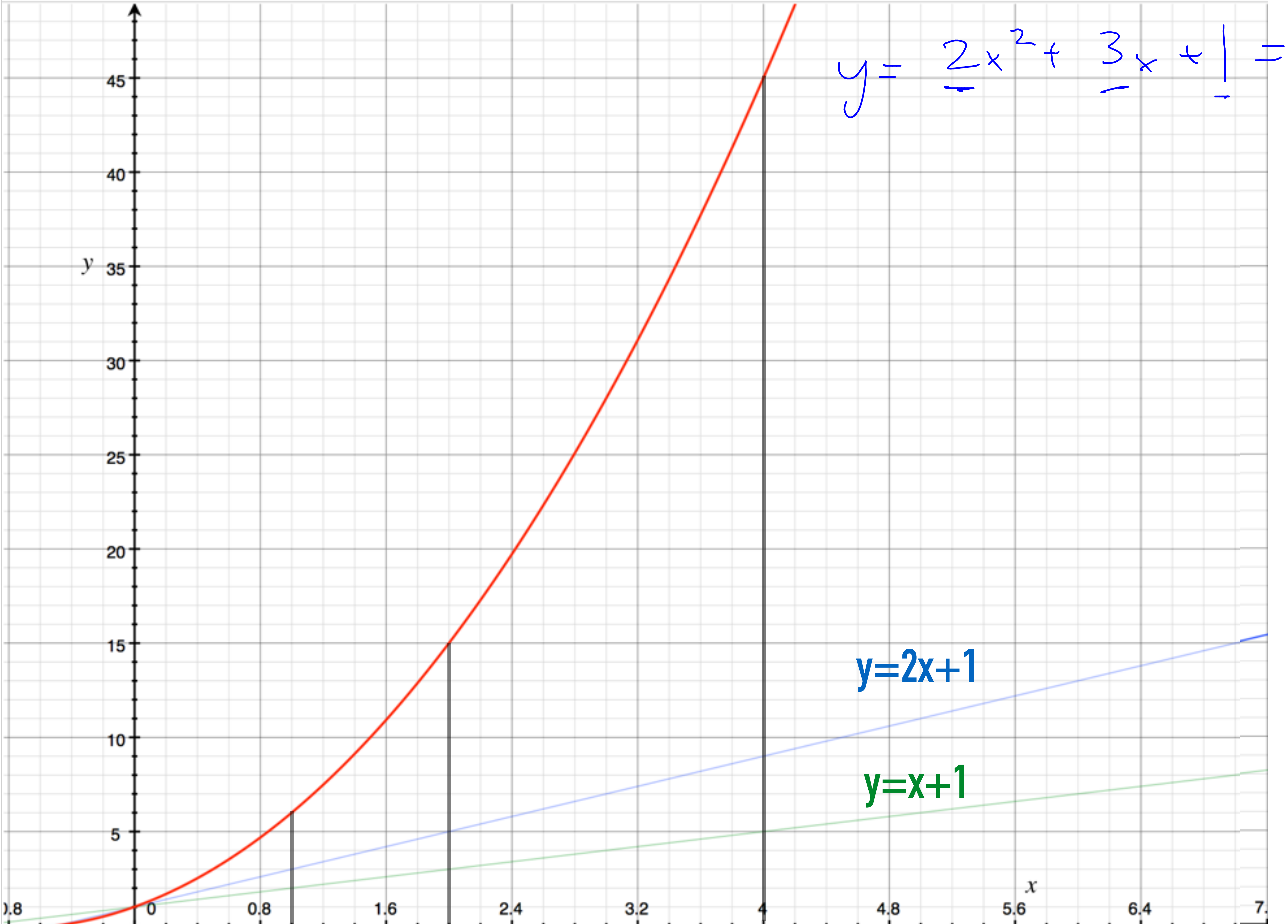


$(6, 15, 45)$ are the point-wise representations of C

$\underline{21}$ $\underline{B(x)=2x+1}$

$\underline{11}$ $\underline{A(x)=x+1}$

$y=2x^2+3x+1$



$y = 2x^2 + 3x + 1 = 231$

21

11

a_3 a_2 a_1 a_0 ★ b_3 b_2 b_1 b_0

$$A(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + 0x^4 + 0x^5 + 0x^6 + 0x^7$$

$$B(x) = b_0 + b_1x + b_2x^2 + b_3x^3 + 0x^4 + 0x^5 + 0x^6 + 0x^7$$

a_3 a_2 a_1 a_0 ★ b_3 b_2 b_1 b_0

$$A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + 0x^4 + 0x^5 + 0x^6 + 0x^7$$

$$B(x) = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + 0x^4 + 0x^5 + 0x^6 + 0x^7$$

$A(\omega_0)$ $\quad$ $A(\omega_1)$ $\quad$ $A(\omega_2)$ $\quad$ .... $\quad$ $A(\omega_7)$

a3 a2 a1 a0 ★ b3 b2 b1 b0

$$A(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + 0x^4 + 0x^5 + 0x^6 + 0x^7$$

$$B(x) = b_0 + b_1x + b_2x^2 + b_3x^3 + 0x^4 + 0x^5 + 0x^6 + 0x^7$$

$A(\omega_0)$ $A(\omega_1)$ $A(\omega_2)$ .... $A(\omega_7)$ **FFT**

$B(\omega_0)$ $B(\omega_1)$ $B(\omega_2)$ .... $B(\omega_7)$ **FFT**

a_3 a_2 a_1 a_0 $\star$ b_3 b_2 b_1 b_0

$$A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + 0x^4 + 0x^5 + 0x^6 + 0x^7$$

$$B(x) = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + 0x^4 + 0x^5 + 0x^6 + 0x^7$$

$A(\omega_0)$ $A(\omega_1)$ $A(\omega_2)$ .... $A(\omega_7)$ **FFT**

$B(\omega_0)$ $B(\omega_1)$ $B(\omega_2)$ .... $B(\omega_7)$ **FFT**

$C(\omega_0)$ $C(\omega_1)$ $C(\omega_2)$ .... $C(\omega_7)$

a3 a2 a1 a0 ★ b3 b2 b1 b0

$$A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + 0x^4 + 0x^5 + 0x^6 + 0x^7$$

$$B(x) = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + 0x^4 + 0x^5 + 0x^6 + 0x^7$$

$A(\omega_0)$	$A(\omega_1)$	$A(\omega_2)$	....	$A(\omega_7)$	**FFT**	$O(n\log n)$
$B(\omega_0)$	$B(\omega_1)$	$B(\omega_2)$	....	$B(\omega_7)$	**FFT**	
$C(\omega_0)$	$C(\omega_1)$	$C(\omega_2)$	....	$C(\omega_7)$		

$$C(x) = c_0 + c_1 x + c_2 x^2 + \cdots c_7 x^7$$

IFFT

application to mult

karatsuba

1 7 8 9 ★ 1 4 3 2

a b ★ c d

$$\underline{\Theta\left(n^{\log_2 3}\right)}$$

application to mult

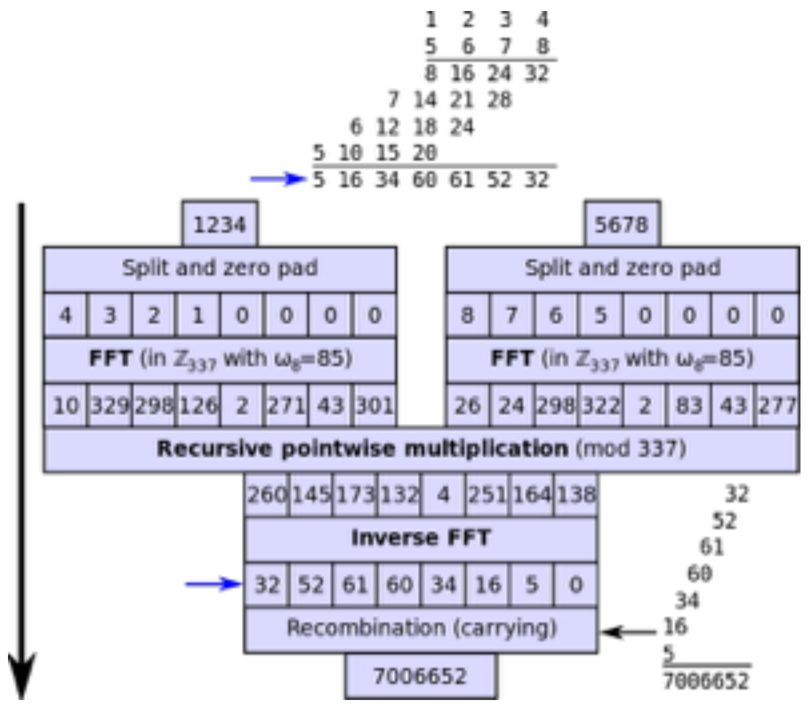
karatsuba

1	7	8	9	★	1	4	3	2
a		b			c		d	

$$T(n) = 3T(n/2) + 6O(n)$$

$$\Theta(n^{\log_2 3})$$

Multiplying n-bit numbers



https://en.wikipedia.org/wiki/File:Integer_multiplication_by_FFT.svg

Schönhage–Strassen '71 $O(n \log n \log \log n)$

Fürer '07 $O(n \log(n) 2^{\log^*(n)})$

$$\log^*(2^{512}) = \underline{\underline{5}}$$

$$2^{2^{512}}$$

$$\log(2^{512}) = 512$$

$$\log(512) = 9$$

$$\log(9) = 3....$$

$$\log(3....) \leq 2.$$

$$\log(2) = 1$$

A GMP-BASED IMPLEMENTATION OF SCHÖNHAGE-STRASSEN'S LARGE INTEGER MULTIPLICATION ALGORITHM

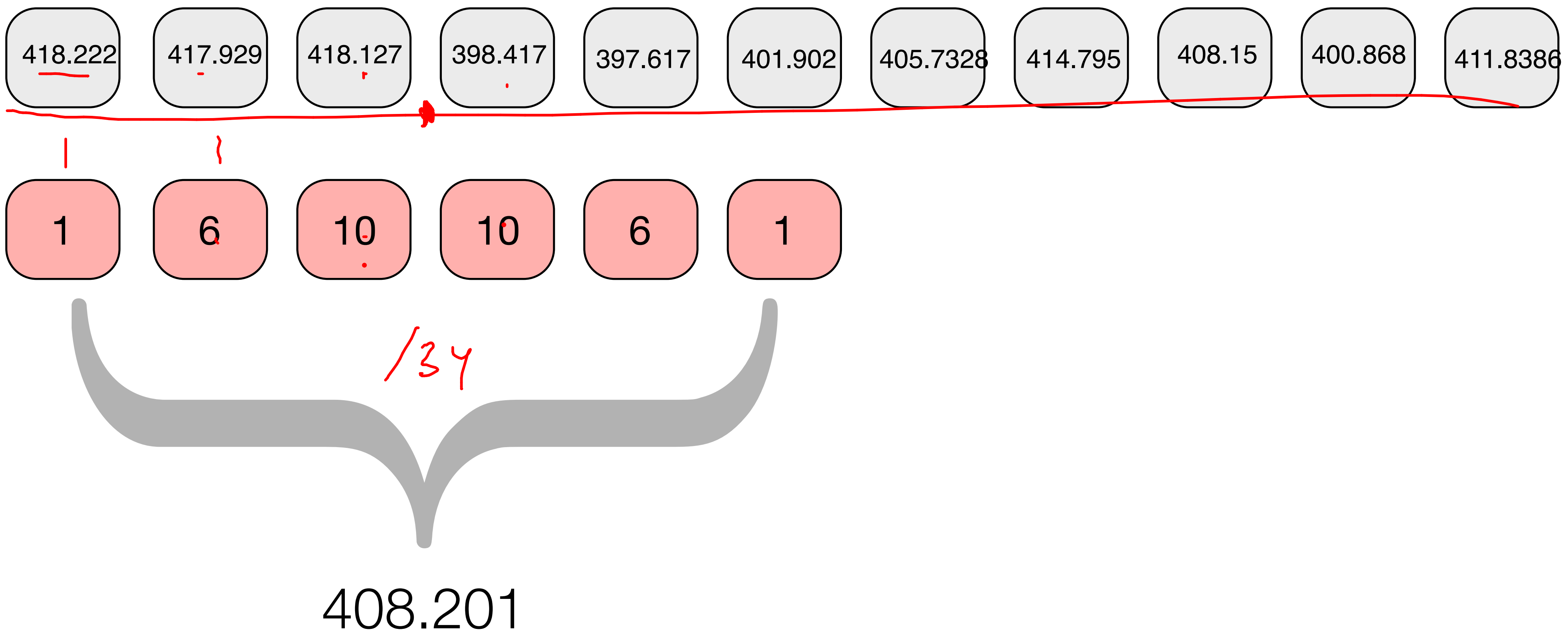
PIERRICK GAUDRY, ALEXANDER KRUPPA, AND PAUL ZIMMERMANN

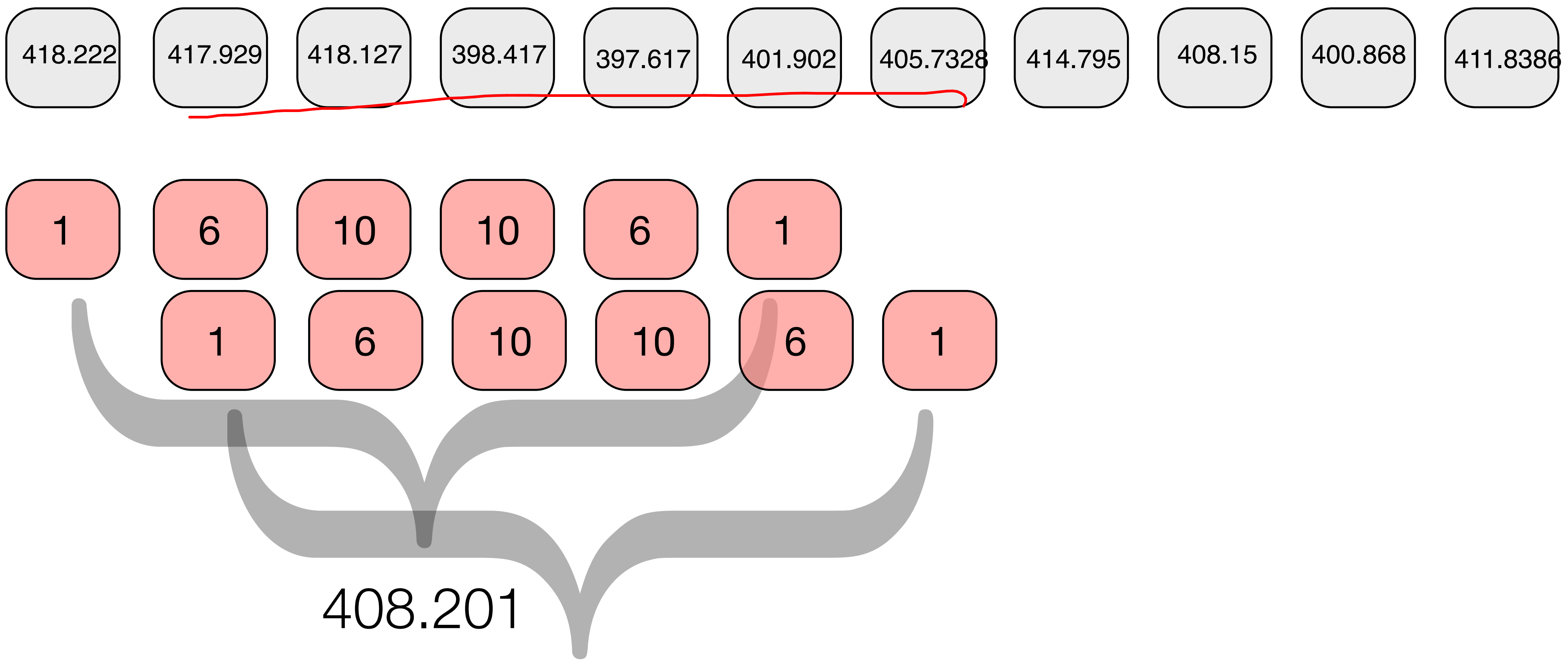
Abstract. Schönhage-Strassen's algorithm is one of the best known algorithms for multiplying large integers. Implementing it efficiently is of utmost importance, since many other algorithms rely on it as a subroutine. We present here an improved implementation, based on the one distributed within the GMP library. The following ideas and techniques were used or tried: faster arithmetic modulo $2^n + 1$, improved cache locality, Mersenne transforms, Chinese Remainder Reconstruction, the $\sqrt{2}$ trick, Harley's and Granlund's tricks, improved tuning. We also discuss some ideas we plan to try in the future.

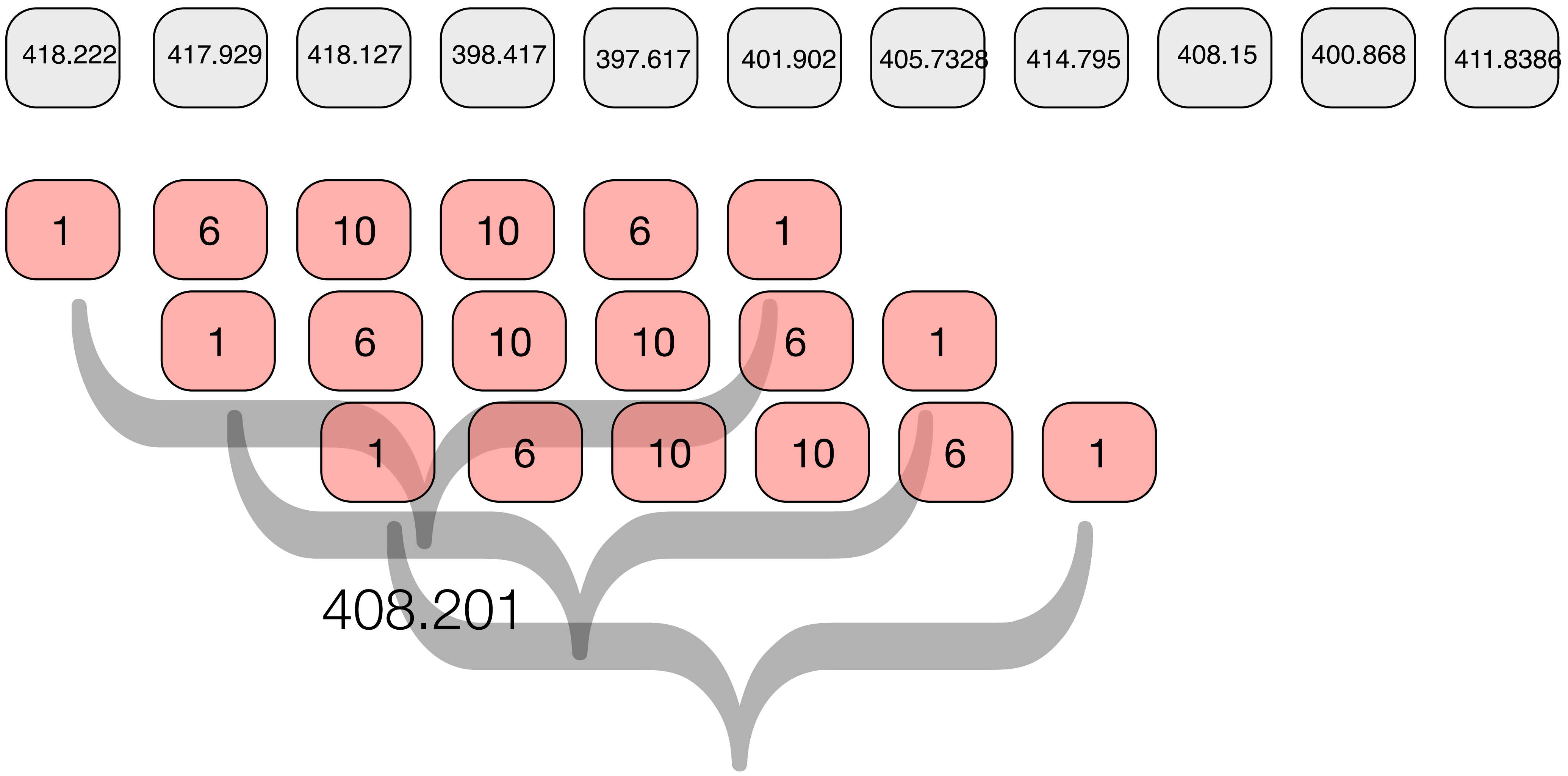
Introduction

Since Schönhage and Strassen have shown in 1971 how to multiply two N-bit integers in $O(N \log N \log\log N)$ time [21], several authors showed how to reduce other operations — inverse, division, square root, gcd, base conversion, elementary functions — to multiplication, possibly with $\log N$ multiplicative factors [5, 8, 17, 18, 20, 23]. It has now become common practice to express complexities in terms of the cost $M(N)$ to multiply two N-bit numbers, and many researchers tried hard to get the best possible constants in front of $M(N)$ for the above-mentioned operations (see for example [6, 16]).

Strangely, much less effort was made for decreasing the implicit constant in $M(N)$ itself, although any gain on that constant will give a similar gain on all multiplication-based operations. Some authors reported on implementations of large integer arithmetic for specific hardware or as part of a number-theoretic project [2, 10]. In this article we concentrate on the question of an optimized implementation of Schönhage-Strassen's algorithm on a classical workstation.


418.222
417.929
418.127
398.417
397.617
401.902
405.7328
414.795
408.15
400.868
411.8386
1
6
10
10
6
1
/34
408.201

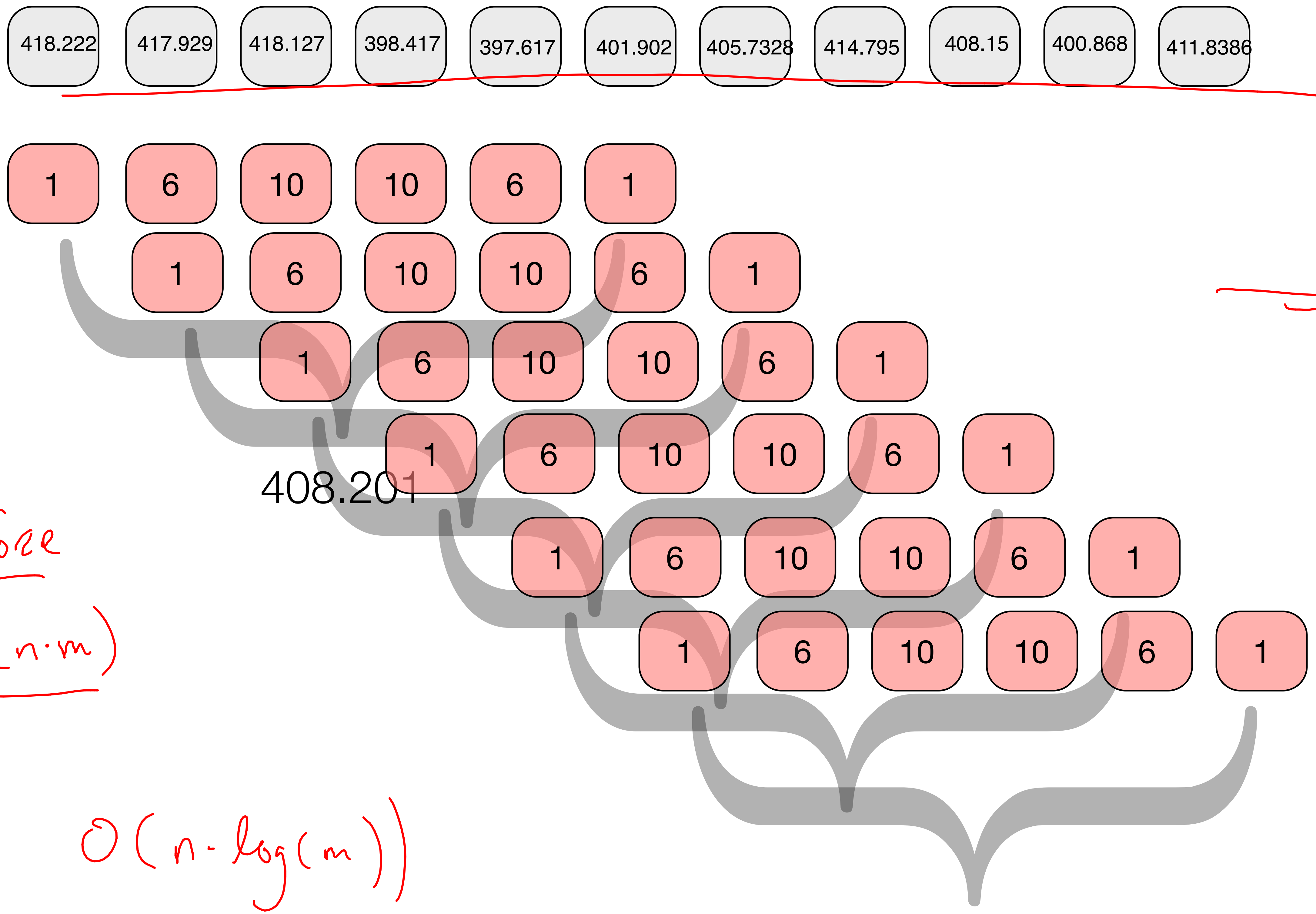

418.222
417.929
418.127
398.417
397.617
401.902
405.7328
414.795
408.15
400.868
411.8386
1
6
10
10
6
1
1
6
10
10
6
1
408.201

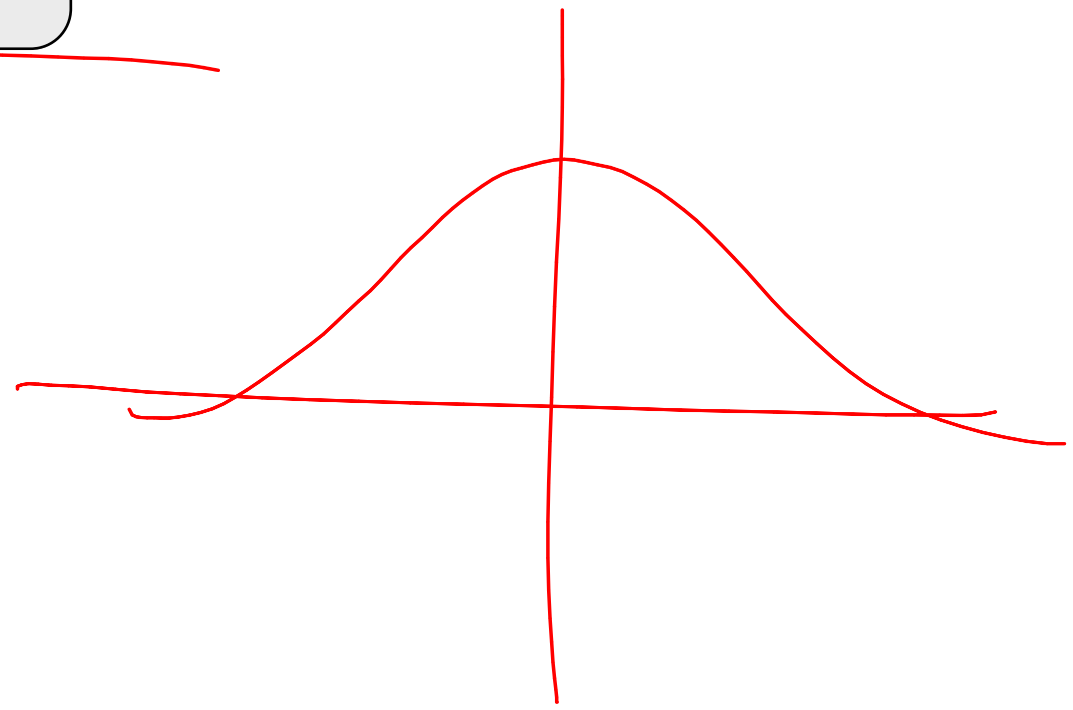


418.222
417.929
418.127
398.417
397.617
401.902
405.7328
414.795
408.15
400.868
411.8386
1
6
10
10
6
1
1
6
10
10
6
1
1
6
10
10
6
1
408.201

n

M





Brute force

$O(n \cdot m)$

FFT: $O(n \cdot \log(m))$

$n \cdot \log m \cdot \frac{\Delta}{m}$

String matching with *

ACAAGATGCCATTGTCCCCCGGCCTCCTGCTGCTGCTGCTCTCCGGGGCCACGGCCACCGCTGCCCTGCC n
CCTGGAGGGTGGCCCCACCGGCCGAGACAGCGAGCATATGCAGGAAGCGGCAGGAATAAGGAAAAGCAGC
CTCCTGACTTTCCTCGCTTGGTGGTTTGAGTGGACCTCCCAGGCCAGTGCCGGGCCCCTCATAGGAGAGG
AAGCTCGGGAGGTGGCCAGGCGGCAGGAAGGCGCACCCCCCCAGCAATCCGCGCGCCGGGACAGAATGCC
CTGCAGGAACTTCTTCTGGAAGACCTTCTCCTCCTGCAAATAAAACCTCACCCATGAATGCTCACGCAAG
TTTAATTACAGACCTGAA

Looking for all occurrences of

GGC*GAG*C*GC m

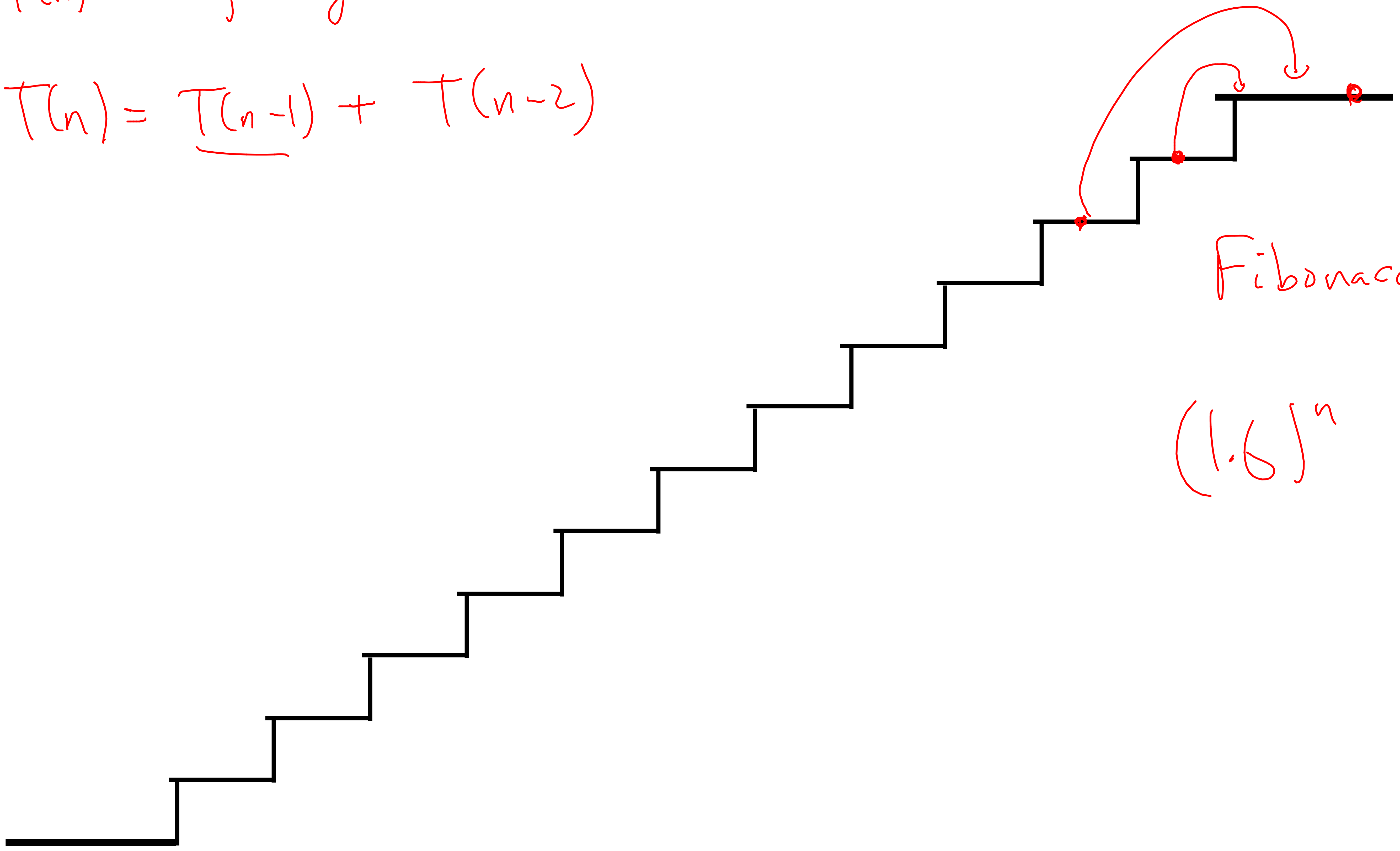
where I don't care what the * symbol is.


West 81st Street, New York, ...
Add a photo


MIGROS

$T(n) =$ # of different ways to climb n stairs using hops of 1 or 2

$$T(n) = \underline{T(n-1)} + T(n-2)$$



$\underline{n}$

Fibonacci recurrence

$$(1.6)^n$$

```
Stairs(n)
    if n<=1 return 1

    return Stairs(n-1) + Stairs(n-2)
```

Stairs(n)

if n<=1 return 1
ret Stairs(n-1) + Stairs(n-2

Stairs(5)

Stairs(4) Stairs(3)

Stairs(3) Stairs(2) Stairs(2) Stairs(1)

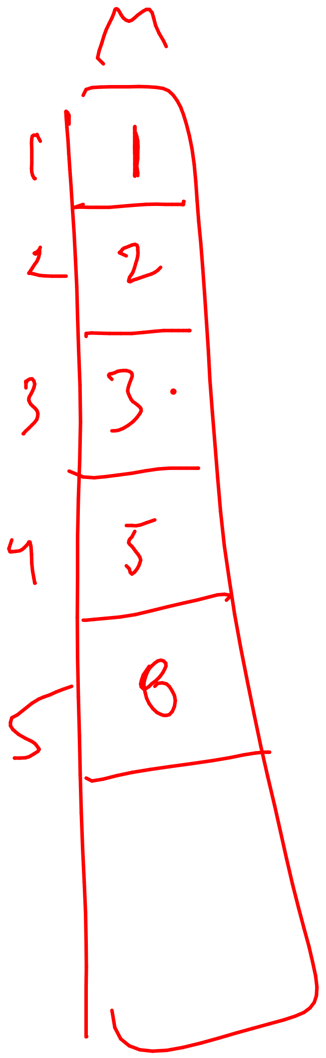
Stairs(2) Stairs(1) Stairs(1) Stairs(0) Stairs(1) Stairs(0)

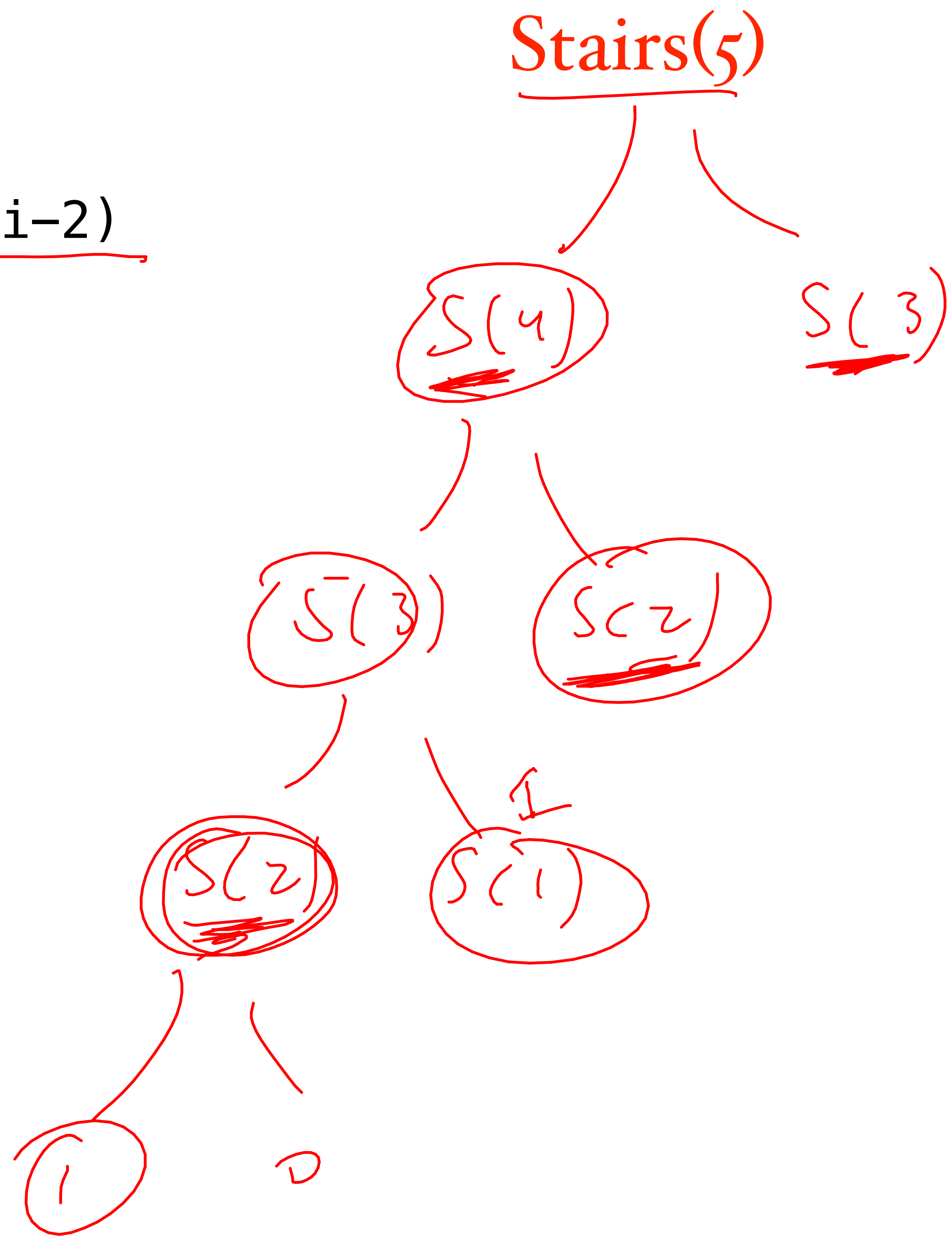
```
initialize memory M

Stairs(n)
```

Stairs(n)

```
    if n<=1 then return 1
    if n is in M, return M[n]
    answer = Stairs(i-1)+ Stairs(i-2)
    M[n] = answer
    return answer
```





Stairs(n)

```
stair[0]=1
stair[1]=1
```

Stairs(n)

```
stair[0]=1
stair[1]=1
for i=2 to n
  stair[i] = stair[i−1]+stair[i−2]
return stair[i]
```

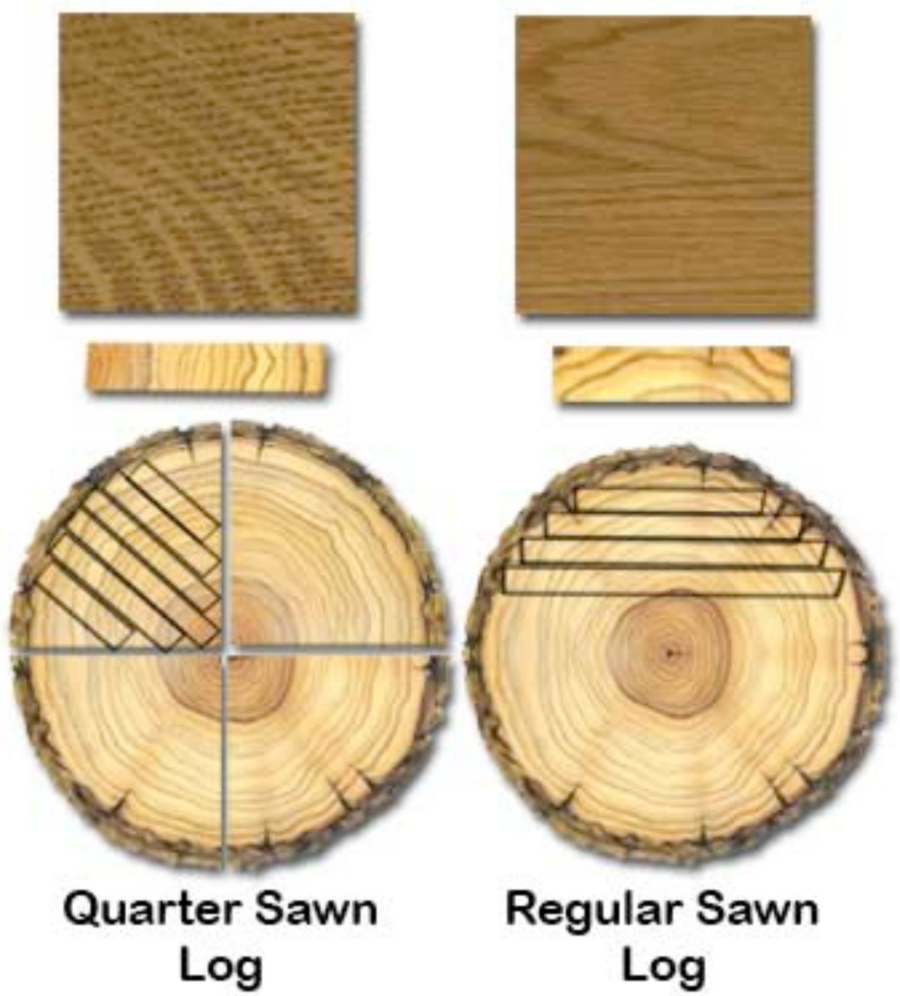
Dynamic Programming

two ideas

recursive structure

memoizing

wood cutting



http://www.amishhandcraftedheirlooms.com/quarter-sawn-oak.htm

http://snlm.files.wordpress.com/2008/08/bill-wakefield-and-carl-fie.gif

Spot price for lumber

1” 2” 3” 4” 5” 6” 7” 8”

Log cutter dilemna

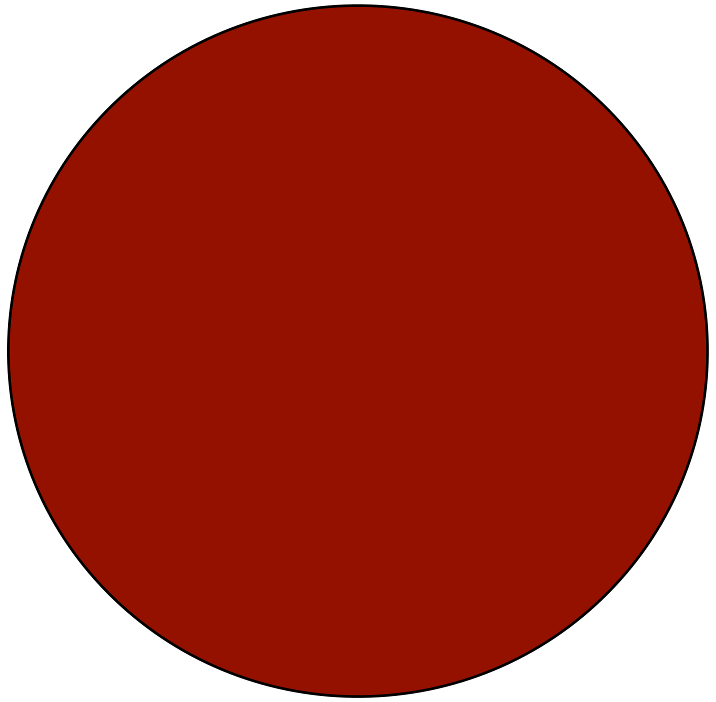
input to the problem: $n, (p_1, \ldots, p_n)$

goal:

Greedy fails

1”	2”	3”	4”	5”
1$	6$	7$	8$	10$

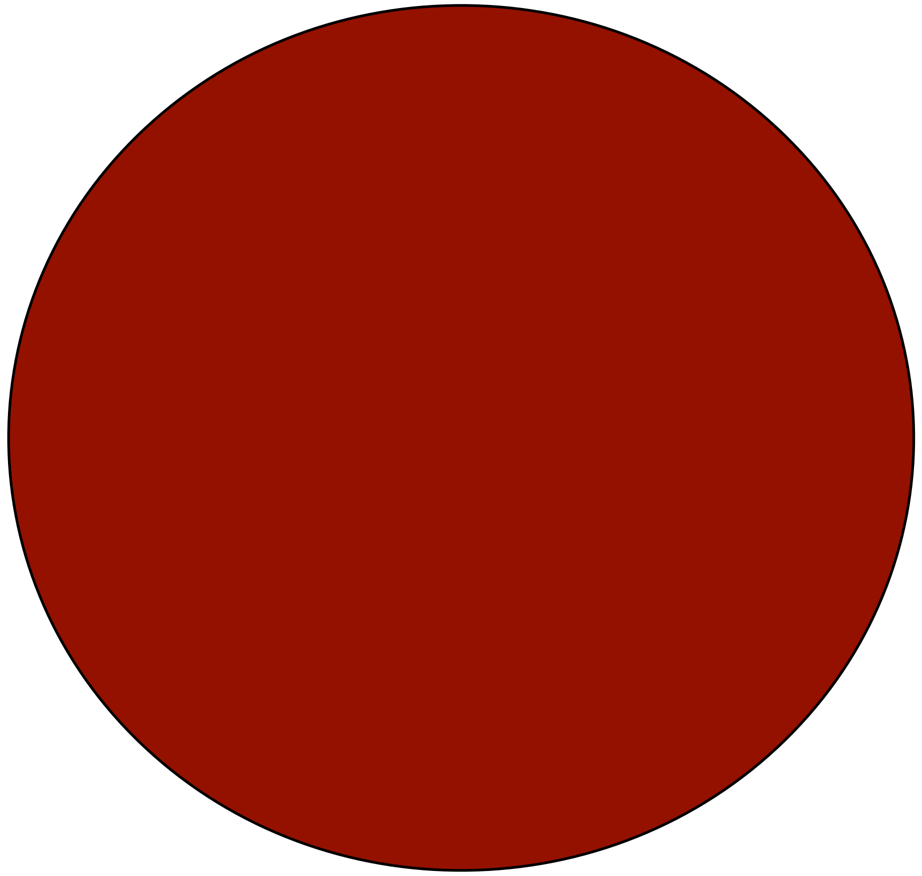
5” log



Greedy “Avg” fails

1”	2”	3”	4”	5”	6”
1$	18$	24$	36$	50$	50$

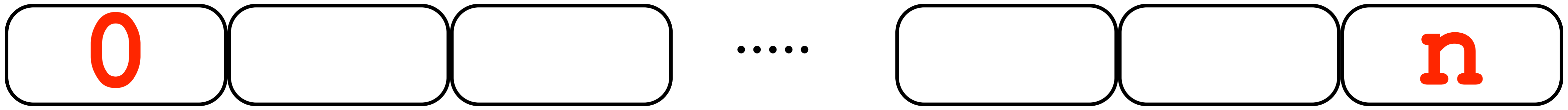
6” log



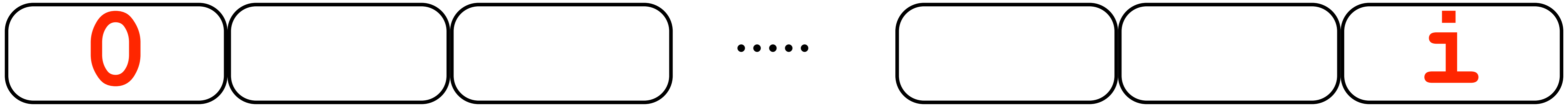
Observation

Solution equation

Approach



Approach



BestLogs($n, (p_1, \ldots, p_n)$)

```
    if n<=0 return 0
```

BestLogs($n, (p_1, \ldots, p_n)$)

```
if n<=0 return 0
for i=1 to n
  Best[i] = max_{k=1...i} {p_k + Best[i-k]}

return Best[n]
```

$$\text{Best}[i] = \max_{k=1\ldots i}\{p_k + \text{Best}[i-k]\}$$

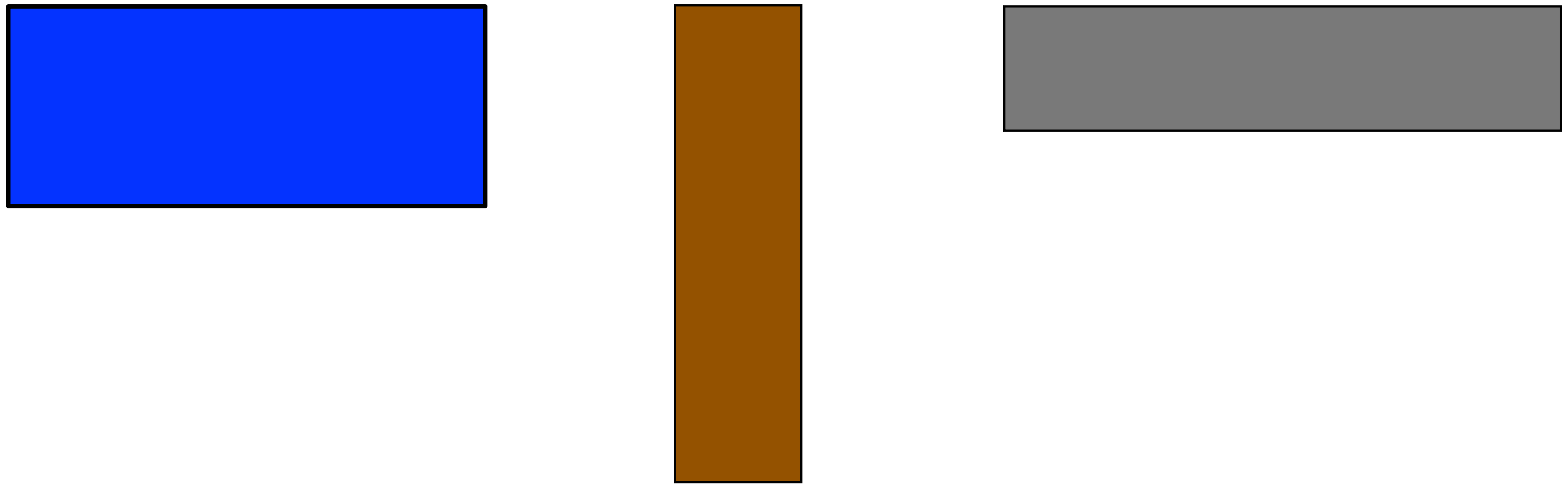
The actual cuts?

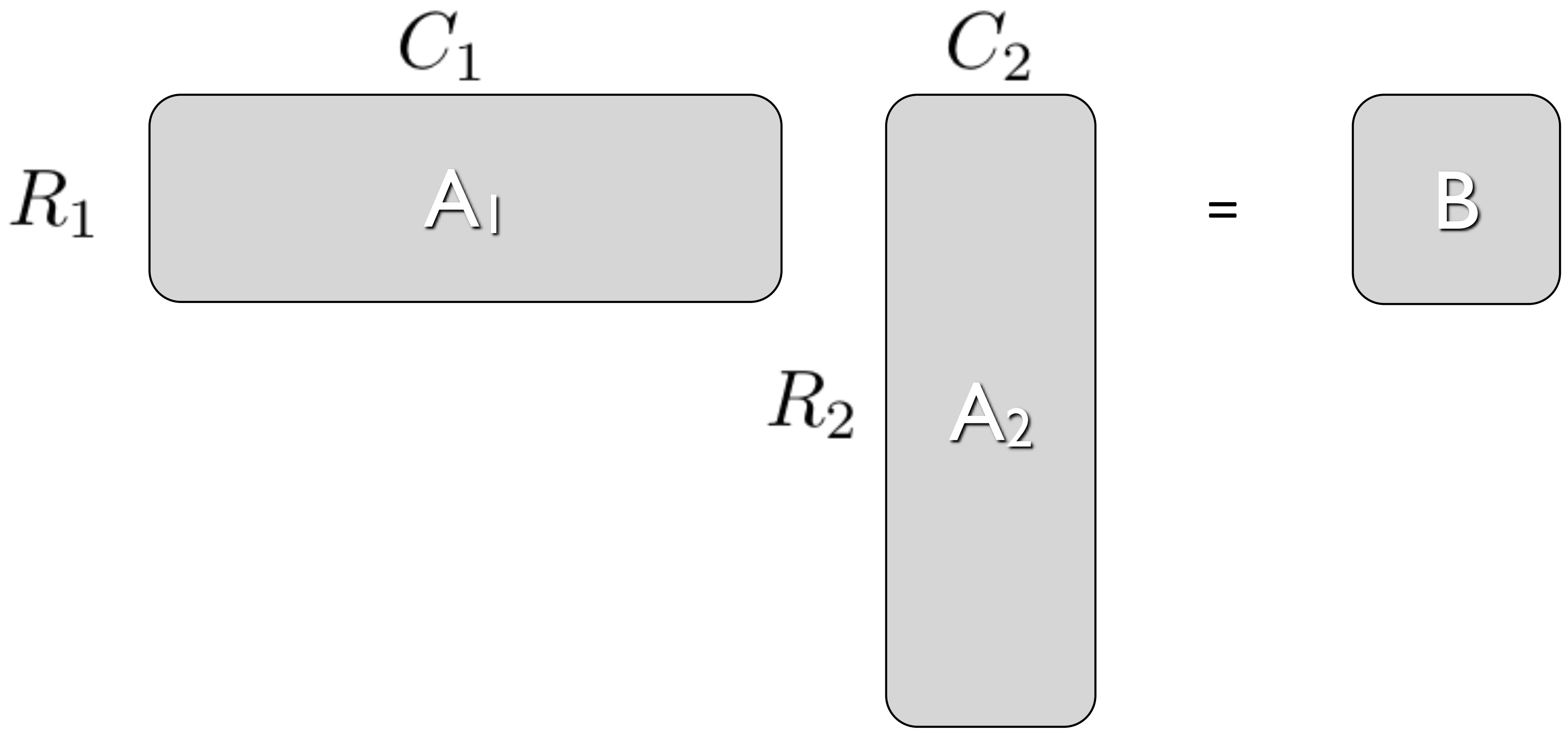
BestLogs($n, (p_1, \ldots, p_n)$)

```
if n<=0 return 0
for i=1 to n
  Best[i] = max_{k=1...i} {p_k + Best[i-k]}
  choice[i] = k*
return Best[n]
```

$$\text{Best}[i] = \max_{k=1\ldots i} \{p_k + \text{Best}[i-k]\}$$

Matrix

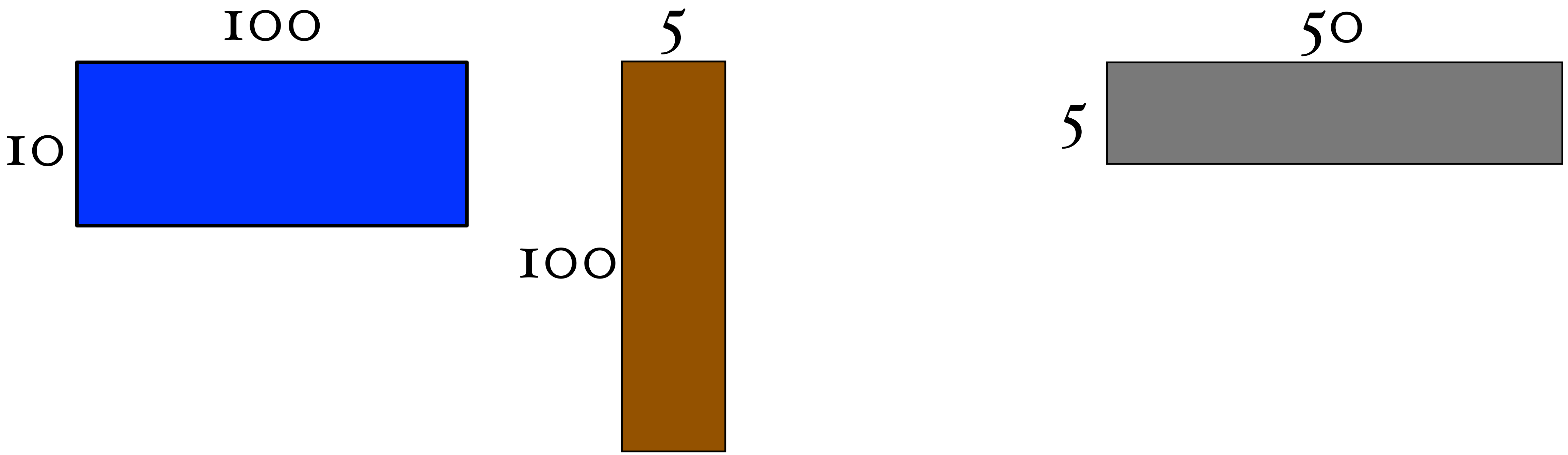



C_1
C_2
R_1
A_1
=
B
R_2
A_2

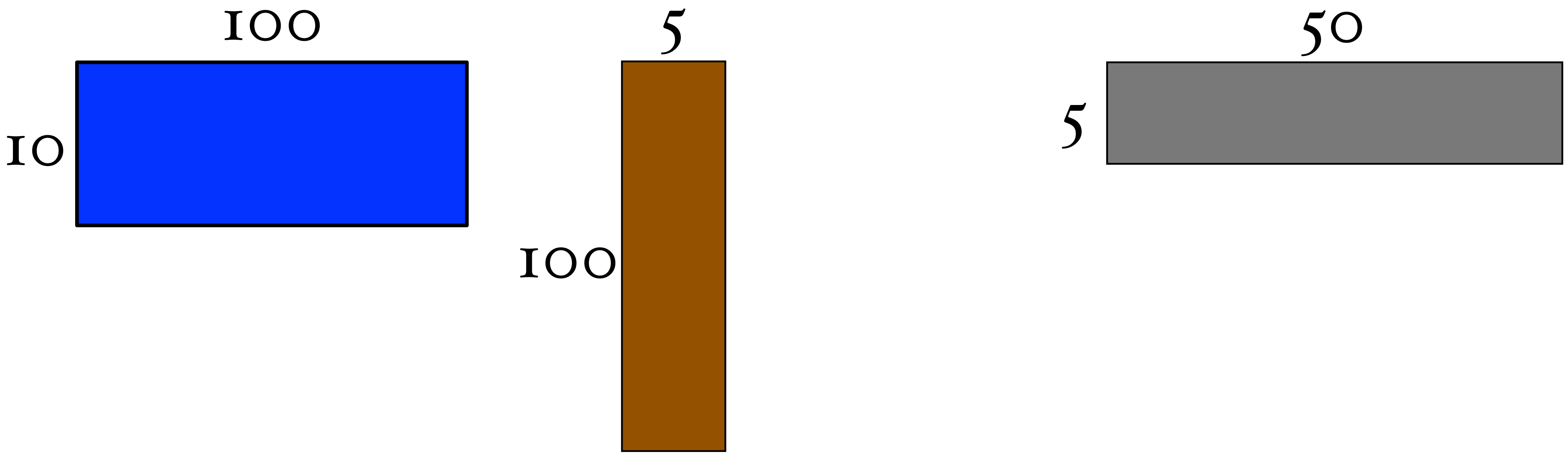
$$A_1 \cdot A_2 \cdot A_3$$

$$(A_1 \cdot A_2) \cdot A_3 \qquad A_1 \cdot (A_2 \cdot A_3)$$

$$(A_1 \cdot A_2) \cdot A_3$$



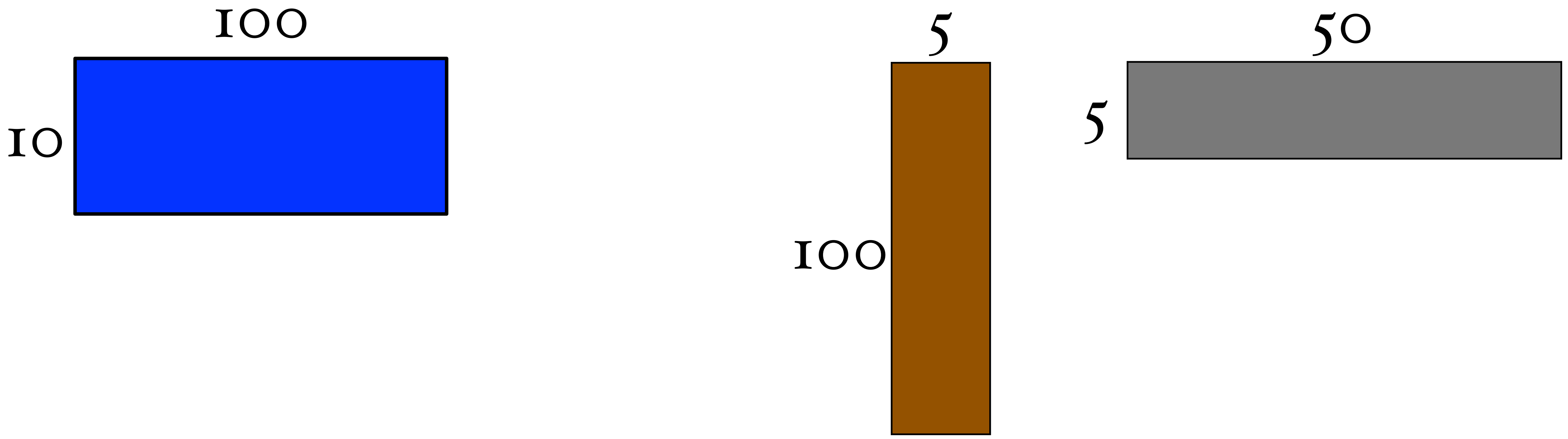
$$(A_1 \cdot A_2) \cdot A_3$$



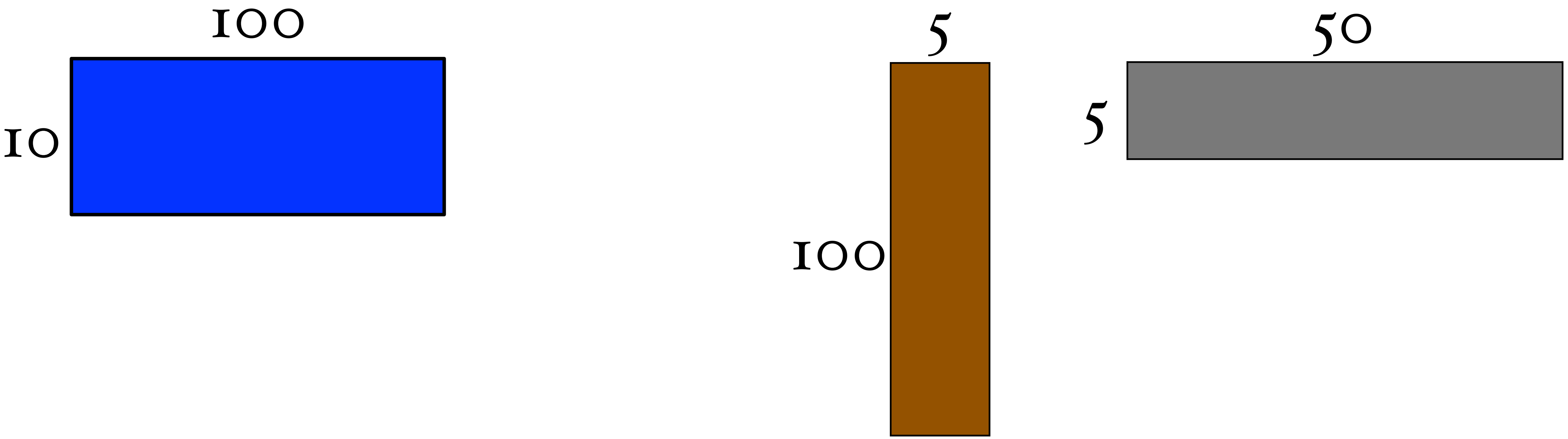
$$10 \cdot 100 \cdot 5 + 10 \cdot 5 \cdot 50$$

operations

$$A_1 \cdot A_2 \cdot A_3$$



$$A_1 \cdot A_2 \cdot A_3$$



$$100 \cdot 5 \cdot 50 + 10 \cdot 100 \cdot 50$$

operations

order matters

(for efficiency)

how many ways to compute?

$$A_1 A_2 A_3 \ldots A_n$$

how many ways to compute?

$$\boxed{A_1}\boxed{A_2 A_3 \ldots A_n}$$

$$A_1 A_2 A_3 \ldots A_n$$

how many ways to compute?

$$\boxed{A_1}\boxed{A_2 A_3 \ldots A_n}$$

$$\boxed{A_1 A_2}\boxed{A_3 \ldots A_n}$$

$$A_1 A_2 A_3 \ldots A_n$$

how many ways to compute?

$$\boxed{A_1}\boxed{A_2 A_3 \ldots A_n}$$

$$\boxed{A_1 A_2}\boxed{A_3 \ldots A_n}$$

$$\boxed{A_1 A_2 A_3}\boxed{\ldots A_n}$$

how do we solve it?

identify smaller instances of the problem

devise method to combine solutions

small # of different subproblems

solved them in the right order

optimal way to compute

$$A_1 A_2 A_3 A_4 \ldots A_n$$

optimal way to compute

$$A_1 A_2 A_3 A_4 \cdots A_n$$

B[1,n]

optimal way to compute

$$A_1 A_2 A_3 A_4 \ldots A_n$$

B[1,n]

B[1,1]
B[2,n]

$R_1 C_1 C_n$

optimal way to compute

$$A_1 A_2 A_3 A_4 \ldots A_n$$

B[1,n]

B[1,1]	B[1,2]	...	B[1,n-2]	B[1,n-1]
B[2,n]	B[3,n]	...	B[n-1,n]	B[n,n]
$R_1 C_1 C_n$	$R_1 C_2 C_n$		$R_1 C_{n-2} C_n$	$R_1 C_{n-1} C_n$

$$B(i, i) = 1$$

$$B(1, n) = \min \left\{ \vphantom{\begin{array}{c} \\ \\ \\ \\ \end{array}} \right.$$

$$B(i,i) = 1$$

$$B(1,n) = \min \begin{cases} B(1,1) + B(2,n) + r_1 c_1 c_n \\ B(1,2) + B(3,n) + r_1 c_2 c_n \\ \vdots \\ B(1,n-1) + B(n,n) + r_1 c_{n-1} c_n \end{cases}$$

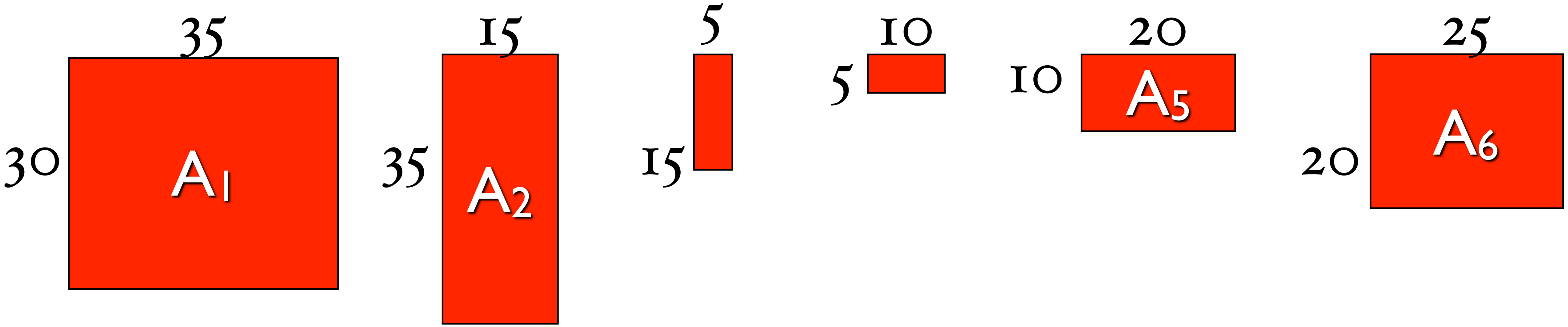
$$B(i, j) =$$

$$\begin{cases} 0 \text{ if } i = j \\ \min_k \{B(i, k) + B(k+1, j) + r_i c_k c_j \end{cases}$$

$$B(i,j) =$$

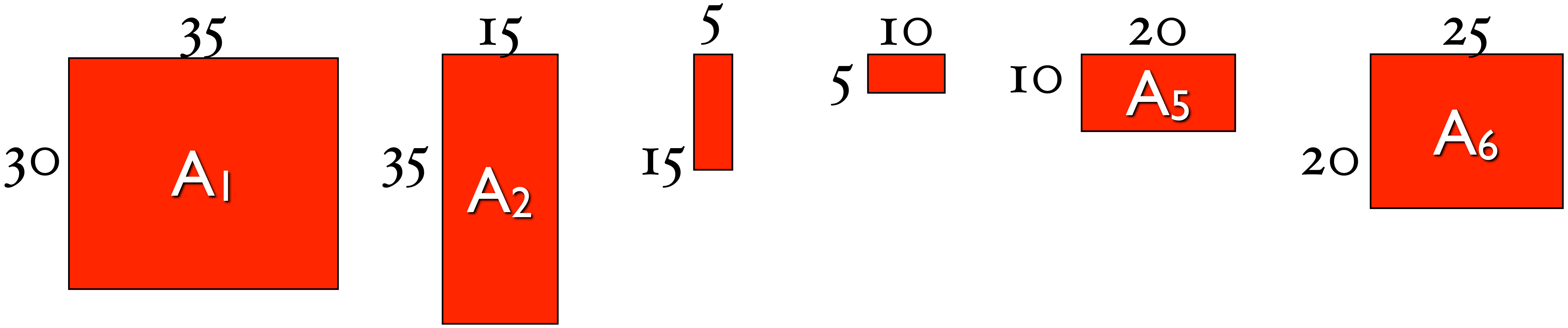
$$\begin{cases} 0 \text{ if } i = j \\ \min_k \{B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$

which order to solve?

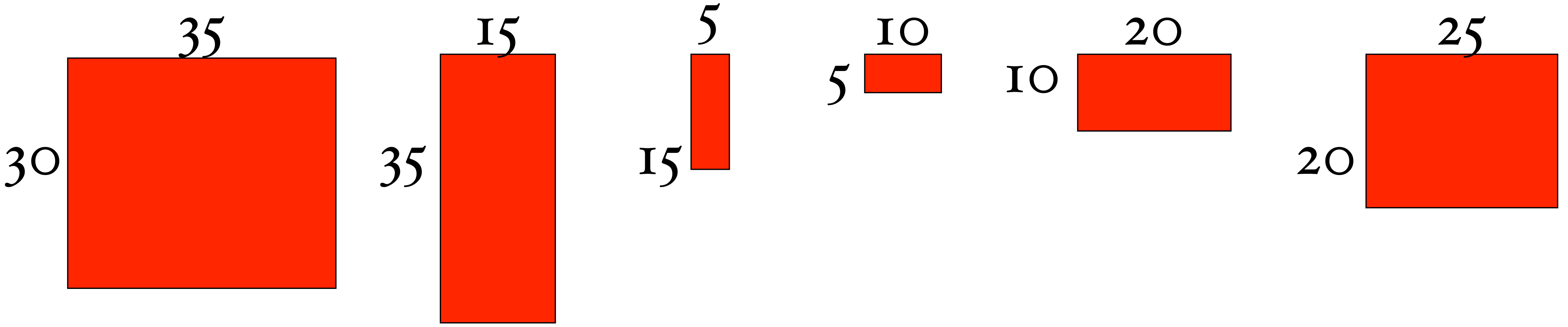


6						0
5					0	
4				0		
3			0			
2		0				
1	0					
	1	2	3	4	5	6

$$B(i,j) = \begin{cases} 0 \text{ if } i = j \\ \min_k\{B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$

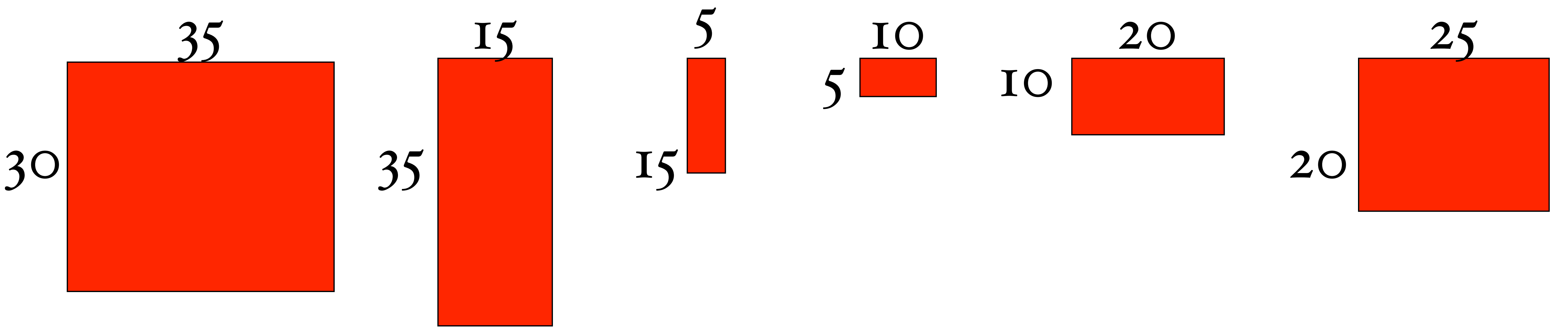


$$B(1,2) =$$



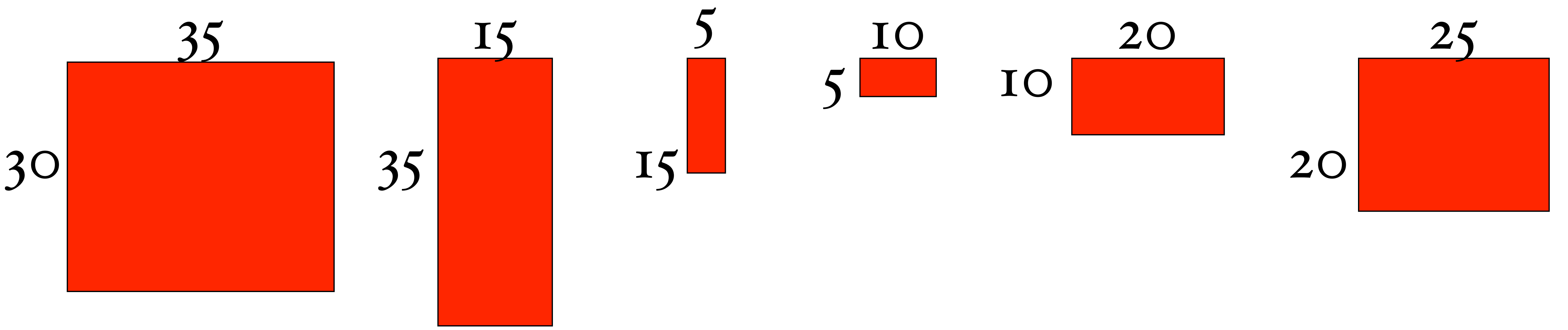
6					10*20*25 = 5000	0
5				5*10*20 = 1000	0	
4			15*5*10 = 750	0		
3		35*15*5 = 2625	0			
2	30*35*15 = 15750	0				
1	0					
	1	2	3	4	5	6

$$B(i,j) = \begin{cases} 0 \text{ if } i = j \\ \min_k \{ B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$



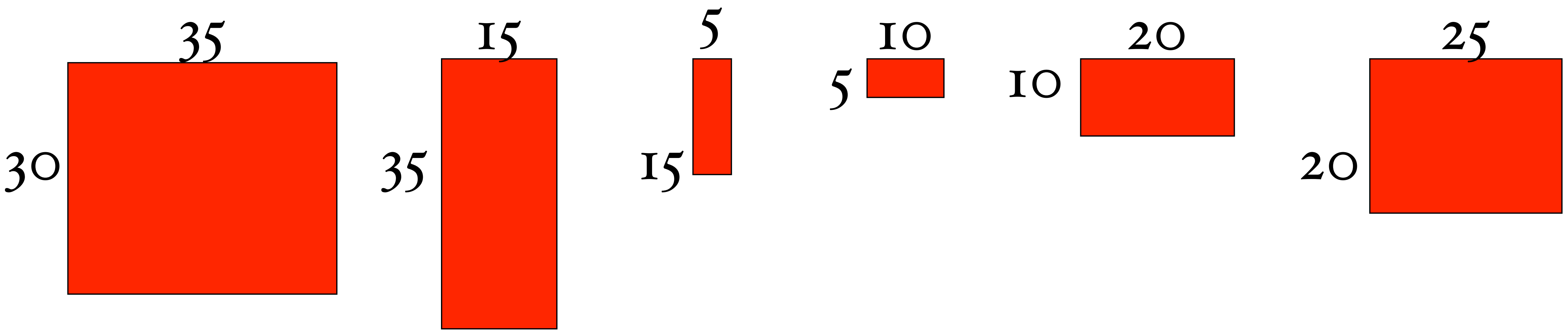
3		35*15*5 = 2625	0			
2	30*35*15 = 15750	0				
1	0					
	1	2	3	4	5	6

$$B(i,j) = \begin{cases} 0 \text{ if } i = j \\ \min_k\{B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$



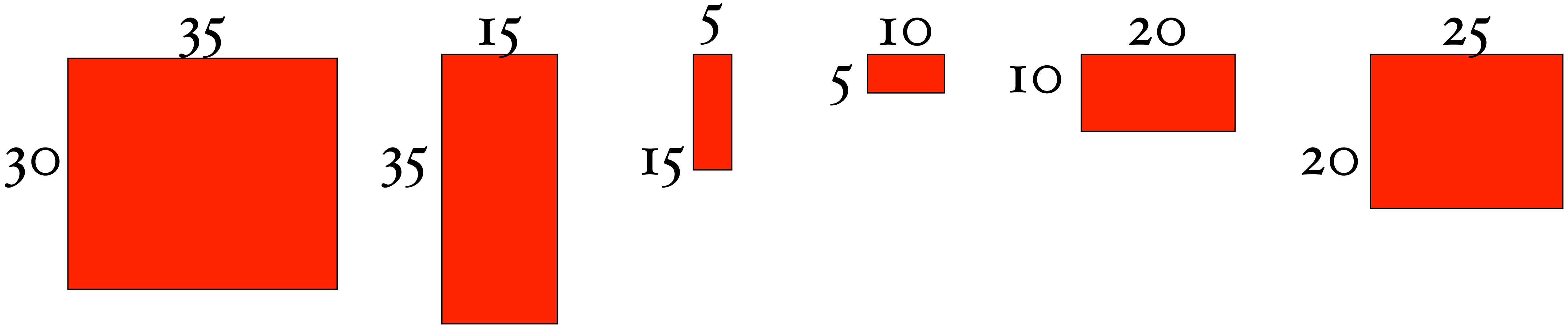
	1	2	3	4	5	6
6		10500	5375	3500	10*20*25 = 5000	0
5	11875	7125	2500	5*10*20 = 1000	0	
4	9375	4375	15*5*10 = 750	0		
3	7875	35*15*5 = 2625	0			
2	30*35*15 = 15750	0				
1	0					

$$B(i,j) = \begin{cases} 0 \text{ if } i = j \\ \min_k\{B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$



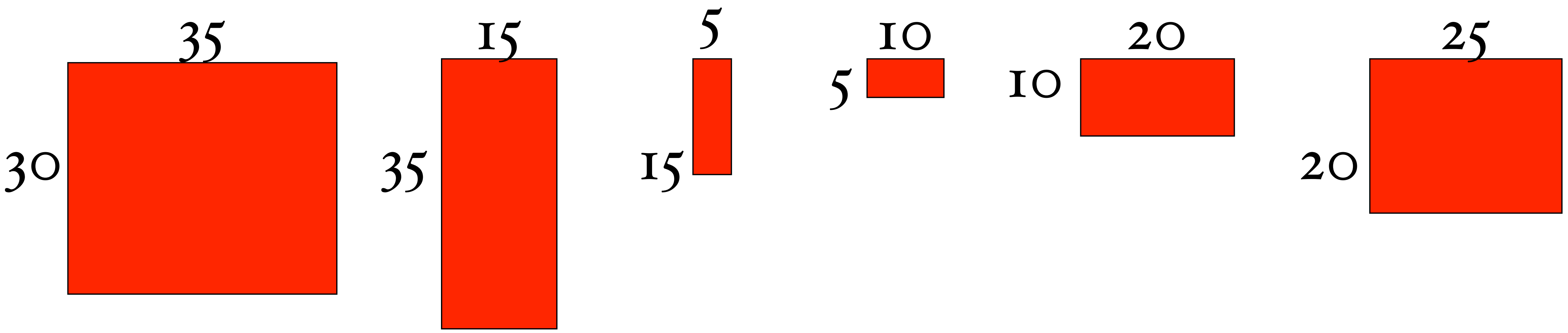
6 

$$C(1,6) = \min \begin{cases} k=1 & C(1,1) + C(2,6) + r_1 c_1 c_6 \\ k=2 & C(1,2) + C(3,6) + r_1 c_2 c_6 \\ k=3 & C(1,3) + C(4,6) + r_1 c_3 c_6 \\ k=4 & C(1,4) + C(5,6) + r_1 c_4 c_6 \\ k=5 & C(1,5) + C(6,6) + r_1 c_5 c_6 \end{cases}$$



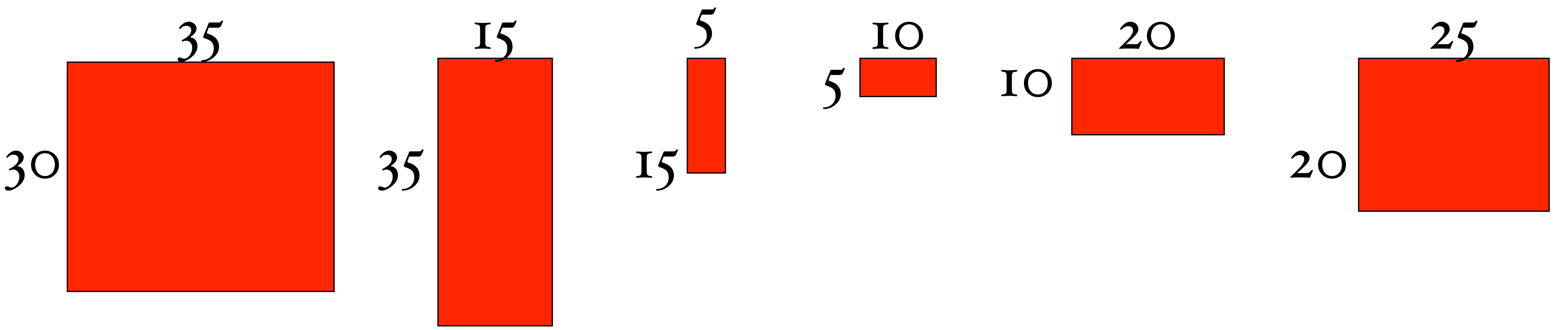
6

$$C(1,6) = \min \begin{cases} k=1 & 0 + 10500 + 30 \cdot 35 \cdot 25 \\ k=2 & 15750 + 5375 + 30 \cdot 15 \cdot 25 \\ k=3 & 7875 + 3500 + 30 \cdot 5 \cdot 25 \\ k=4 & 9375 + 5000 + 30 \cdot 10 \cdot 25 \\ k=5 & 11875 + 0 + 30 \cdot 20 \cdot 25 \end{cases}$$

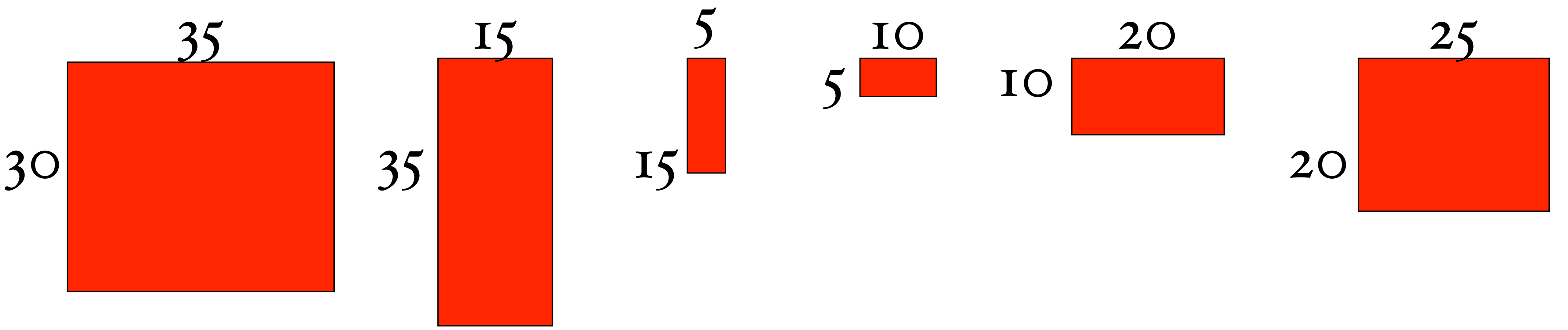


6

$$C(1,6) = \min \begin{cases} k=1 & 0 + 10500 + 26250 \\ k=2 & 15750 + 5375 + 11250 \\ k=3 & 7875 + 3500 + 3750 \\ k=4 & 9375 + 5000 + 7500 \\ k=5 & 11875 + 0 + 15000 \end{cases}$$



6	15125 3	10500	5375	3500	10*20*25 = 5000	0
5	11875	7125	2500	5*10*20 = 1000	0	
4	9375	4375	15*5*10 = 750	0		
3	7875	35*15*5 = 2625	0			
2	30*35*15 = 15750	0				
1	0					
	1	2	3	4	5	6



6	15125 3	10500	5375	3500	10*20*25 = 5000	0
5	11875	7125	2500	5*10*20 = 1000	0	
4	9375	4375	15*5*10 = 750	0		
3	7875	35*15*5 = 2625	0			
2	30*35*15 = 15750	0				
1	0					
	1	2	3	4	5	6

matrix-chain-mult(p)

initialize array m{x,y} to zero

matrix-chain-mult(p)

initialize array m[x,y] to zero

starting at diagonal, working towards upper-left

compute m[i,j] according to

$$\begin{cases} 0 \text{ if } i = j \\ \min_k\{B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$

running time?

initialize array m[x,y] to zero

starting at diagonal, working towards upper-left

compute m[i,j] according to

$$\begin{cases} 0 \text{ if } i = j \\ \min_k\{B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$