Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

School of Computer Science and En
University of New South Wales

4. FAST LARGE INTEGER MULTIPLICATION - part A

Basics revisited: how do we multiply two numbers?

- The primary school algorithm:

Assignment Project Exam Help

```
          X X X X  <- first input integer
```

https://eduassistpro.github.i

```
    X X X X       / O(n^2) elementary  multiplications
  X X X X         + O(n^2) elementary additions
---------------
X X X X X X X X  <-  result of length 2n
```

Add WeChat edu_assist_pr

Basics revisited: how do we multiply two numbers?

- The primary school algorithm:

Assignment Project Exam Help

```
          X X X X  <- first input integer
```

https://eduassistpro.github.i

```
            X X X X       / O(n^2) elementary  multiplications
          X X X X         / + O(n^2) elementary additions
-----------------
X X X X X X X X  <-  result of length 2n
```

Add WeChat edu_assist_pr

- **Can we do it faster than in n^2 many steps??**

The Karatsuba trick

- Take the two input numbers A and B, and split them into two halves:

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick

- Take the two input numbers A and B, and split them into two halves:

$$A = A_1 2^{\frac{n}{2}} + A_0 \qquad A = \overbrace{X\,X\,\ldots\,X}^{A_1}\,\overbrace{X\,X\,\ldots\,X}^{A_0}$$

$n/2$ *bits* $n/2$ *bits*

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick

- Take the two input numbers A and B, and split them into two halves:

$$A = A_1 2^{\frac{n}{2}} + A_0 \qquad A = \overbrace{X\,X\,\ldots\,X}^{A_1}\,\overbrace{X\,X\,\ldots\,X}^{A_0}$$

$n/2$ *bits* $n/2$ *bits*

Assignment Project Exam Help

https://eduassistpro.github.i

- AB can now be calculated as follows:

Add WeChat edu_assist_pr

The Karatsuba trick

- Take the two input numbers A and B, and split them into two halves:

$$A = A_1 2^{\frac{n}{2}} + A_0 \qquad A = \overbrace{X\,X\,\ldots\,X}^{A_1}\,\overbrace{X\,X\,\ldots\,X}^{A_0}$$

$n/2$ *bits* $n/2$ *bits*

Assignment Project Exam Help

https://eduassistpro.github.i

- AB can now be calculated as follows:

$$AB = A_1 B_1 2^n + (A_1 B_0 + A_0 B_1) 2^{\frac{n}{2}} + A_0 B_0$$

Add WeChat edu_assist_pr

The Karatsuba trick

- Take the two input numbers A and B, and split them into two halves:

$$A = A_1 2^{\frac{n}{2}} + A_0 \qquad A = \overbrace{X\,X\,\ldots\,X}^{A_1}\,\overbrace{X\,X\,\ldots\,X}^{A_0}$$

$n/2$ *bits* $n/2$ *bits*

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

- AB can now be calculated as follows:

$$AB = A_1 B_1 2^n + (A_1 B_0 + A_0 B_1) 2^{\frac{n}{2}} + A_0 B_0$$

$$= A_1 B_1 2^n + ((A_1 + A_0)(B_1 + B_0) - A_1 B_1 - A_0 B_0) 2^{\frac{n}{2}} + A_0 B_0$$

- We have saved one multiplication, now we have only three: $A_0 B_0$, $A_1 B_1$ and $(A_1 + A_0)(B_1 + B_0)$.

$$AB = A_1B_1 2^n + ((A_1 + A_0)(B_1 + B_0) - A_1B_1 - A_0B_0)2^{\frac{n}{2}} + A_0B_0$$

```
1:  function MULT(A, B)
2:      if |A| = |B| = 1 then return AB
3:      else
4:
5:
6:
7:
8:          U ← A_0 + A_1;
9:          V ← B_0 + B_1;
10:         X ← MULT(A_0, B_0);
11:         W ← MULT(A_1, B_1);
12:         Y ← MULT(U, V);
13:         return W 2^n + (Y − X − W) 2^{n/2} + X
14:     end if
15: end function
```

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick

- How many steps does this algorithm take? (remember, addition is in linear time!)

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick

- How many steps does this algorithm take? (remember, addition is in linear time!)
- Recurrence: $T(n) = 3T(\frac{n}{2}) + cn$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick

- How many steps does this algorithm take? (remember, addition is in linear time!)
- Recurrence: $T(n) = 3T(\frac{n}{2}) + cn$

Assignment Project Exam Help

$\log_2 3$

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick

- How many steps does this algorithm take? (remember, addition is in linear time!)
- Recurrence: $T(n) = 3T(\frac{n}{2}) + cn$
- since $1 < \log_2 3$

$$f(n) = cn = O(n^{\log_2 3 - \varepsilon}) \quad \text{for}$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick

- How many steps does this algorithm take? (remember, addition is in linear time!)
- Recurrence: $T(n) = 3T(\frac{n}{2}) + cn$
- since $1 < \log_2 3$
- $f(n) = cn = O(n^{\log_2 3 - \varepsilon})$ for
- Thus, the first case of the Master Theorem applies.

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick

- How many steps does this algorithm take? (remember, addition is in linear time!)
- Recurrence: $T(n) = 3T\left(\frac{n}{2}\right) + cn$
- since $1 < \log_2 3$

$$f(n) = cn = O(n^{\log_2 3 - \varepsilon}) \quad \text{for}$$

- Thus, the first case of the Master Theorem applies.
- Consequently,

$$T(n) = \Theta(n^{\log_2 3}) < \Theta(n^{1.585})$$

without going through the messy calculations!

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- Can we do better if we break the numbers in more than two pieces?

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- Can we do better if we break the numbers in more than two pieces?
- Lets try breaking the numbers A, B into 3 pieces; then with $k = n/3$ we obtain

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- Can we do better if we break the numbers in more than two pieces?
- Lets try breaking the numbers A, B into 3 pieces; then with $k = n/3$ we obtain

$$A = \underbrace{XXX \ldots XX}_{0}\,\underbrace{XXX \ldots XX}_{1}\,\underbrace{XXX \ldots XX}_{2}$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- Can we do better if we break the numbers in more than two pieces?
- Lets try breaking the numbers A, B into 3 pieces; then with $k = n/3$ we obtain

$$A = \underbrace{XXX \ldots XX}_{0}\,\underbrace{XXX \ldots XX}_{0}\,\underbrace{XXX \ldots XX}_{0}$$

i.e.,

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- Can we do better if we break the numbers in more than two pieces?
- Lets try breaking the numbers A, B into 3 pieces; then with $k = n/3$ we obtain

$$A = \underbrace{XXX \ldots XX}_{0}\ \underbrace{XXX \ldots XX}_{}\ \underbrace{XXX \ldots XX}_{}$$

i.e.,

$$B = B_2 2^{2k} + B_1 2^k + B_0$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- Can we do better if we break the numbers in more than two pieces?
- Lets try breaking the numbers A, B into 3 pieces; then with $k = n/3$ we obtain

$$A = \underbrace{XXX \ldots XX}_{2} \underbrace{XXX \ldots XX}_{1} \underbrace{XXX \ldots XX}_{0}$$

i.e.,

$$B = B_2 2^{2k} + B_1 2^k + B_0$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

- So,

$$AB = A_2B_2\,2^{4k} + (A_2B_1 + A_1B_2)2^{3k} + (A_2B_0 + A_1B_1 + A_0B_2)2^{2k} + \\ + (A_1B_0 + A_0B_1)2^k + A_0B_0$$

The Karatsuba trick

$$AB = \underbrace{A_2B_2}_{C_4}\, 2^{4k} + \underbrace{(A_2B_1 + A_1B_2)}_{C_3}2^{3k} + \underbrace{(A_2B_0 + A_1B_1 + A_0B_2)}_{C_2}2^{2k} +$$
$$+ \underbrace{(A_1B_0 + A_0B_1)}_{C_1}2^{k} + \underbrace{A_0B_0}_{C_0}$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick

$$AB = \underbrace{A_2B_2}_{C_4}\, 2^{4k} + \underbrace{(A_2B_1 + A_1B_2)}_{C_3}2^{3k} + \underbrace{(A_2B_0 + A_1B_1 + A_0B_2)}_{C_2}2^{2k} +$$
$$+ \underbrace{(A_1B_0 + A_0B_1)}_{C_1}2^{k} + \underbrace{A_0B_0}_{C_0}$$

Assignment Project Exam Help

- we need only 5 coefficients:

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick

$$AB = \underbrace{A_2B_2}_{C_4}\, 2^{4k} + \underbrace{(A_2B_1 + A_1B_2)}_{C_3}2^{3k} + \underbrace{(A_2B_0 + A_1B_1 + A_0B_2)}_{C_2}2^{2k} +$$

$$+ \underbrace{(A_1B_0 + A_0B_1)}_{C_1}2^{k} + \underbrace{A_0B_0}_{C_0}$$

Assignment Project Exam Help

- we need only 5 coefficients:

https://eduassistpro.github.i

$$C_1 = A_1B_0 + A_0$$

$$C_0 = A_0B_0$$

Add WeChat edu_assist_pr

The Karatsuba trick

$$AB = \underbrace{A_2B_2}_{C_4}\, 2^{4k} + \underbrace{(A_2B_1 + A_1B_2)}_{C_3}2^{3k} + \underbrace{(A_2B_0 + A_1B_1 + A_0B_2)}_{C_2}2^{2k} +$$

$$+ \underbrace{(A_1B_0 + A_0B_1)}_{C_1}2^{k} + \underbrace{A_0B_0}_{C_0}$$

Assignment Project Exam Help

- we need only 5 coefficients:

https://eduassistpro.github.i

$$C_1 = A_1B_0 + A_0$$

$$C_0 = A_0B_0$$

Add WeChat edu_assist_pr

- Can we get these with 5 multiplications only?

The Karatsuba trick

$$AB = \underbrace{A_2B_2}_{C_4}\,2^{4k} + \underbrace{(A_2B_1 + A_1B_2)}_{C_3}2^{3k} + \underbrace{(A_2B_0 + A_1B_1 + A_0B_2)}_{C_2}2^{2k} +$$
$$+ \underbrace{(A_1B_0 + A_0B_1)}_{C_1}2^{k} + \underbrace{A_0B_0}_{C_0}$$

Assignment Project Exam Help

- we need only 5 coefficients:

https://eduassistpro.github.i

$$C_1 = A_1B_0 + A_0$$
$$C_0 = A_0B_0$$

Add WeChat edu_assist_pr

- Can we get these with 5 multiplications only?
- Should we perhaps look at

$$(A_2 + A_1 + A_0)(B_2 + B_1 + B_0) =$$
$$A_0B_0 + A_1B_0 + A_2B_0 + A_0B_1 + A_1B_1 + A_2B_1 + A_0B_2 + A_1B_2 + A_2B_2 \quad ???$$

The Karatsuba trick

$$AB = \underbrace{A_2B_2}_{C_4}\,2^{4k} + \underbrace{(A_2B_1 + A_1B_2)}_{C_3}2^{3k} + \underbrace{(A_2B_0 + A_1B_1 + A_0B_2)}_{C_2}2^{2k} +$$
$$+ \underbrace{(A_1B_0 + A_0B_1)}_{C_1}2^{k} + \underbrace{A_0B_0}_{C_0}$$

Assignment Project Exam Help

- we need only 5 coefficients:

https://eduassistpro.github.i

$$C_1 = A_1B_0 + A_0$$
$$C_0 = A_0B_0$$

Add WeChat edu_assist_pr

- Can we get these with 5 multiplications only?
- Should we perhaps look at

$$(A_2 + A_1 + A_0)(B_2 + B_1 + B_0) =$$
$$A_0B_0 + A_1B_0 + A_2B_0 + A_0B_1 + A_1B_1 + A_2B_1 + A_0B_2 + A_1B_2 + A_2B_2 \quad ???$$

- Not clear at all how to get $C_0 - C_4$ with 5 multiplications only ...

The Karatsuba trick: slicing into 3 pieces

- We now look for a method for getting these coefficients without any guesswork!

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- We now look for a method for getting these coefficients without any guesswork!
- Let

$$A = A_2\,2^{2k} + A_1\,2^k + A_0$$

$$B = B_2\,2^{2k} + B_1\,2^k + B_0$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- We now look for a method for getting these coefficients without any guesswork!
- Let

$$A = A_2\, 2^{2k} + A_1\, 2^k + A_0$$

$$B = B_2\, 2^{2k} + B_1\, 2^k + B_0$$

- We f

$$P_A(x) = A_2\, x^2 +$$

$$P_B(x) = B_2\, x^2 +$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- We now look for a method for getting these coefficients without any guesswork!
- Let

$$A = A_2\,2^{2k} + A_1\,2^k + A_0$$
$$B = B_2\,2^{2k} + B_1\,2^k + B_0$$

- We f

$$P_A(x) = A_2\,x^2 +$$
$$P_B(x) = B_2\,x^2 +$$

- Note that

$$A = A_2\,(2^k)^2 + A_1\,2^k + A_0 = P_A(2^k);$$
$$B = B_2\,(2^k)^2 + B_1\,2^k + B_0 = P_B(2^k).$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- If we manage to compute somehow the product polynomial

$$P_C(x) = P_A(x)P_B(x) = C_4\,x^4 + C_3\,x^3 + C_2\,x^2 + C_1\,x + C_0,$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- If we manage to compute somehow the product polynomial

$$P_C(x) = P_A(x)P_B(x) = C_4\,x^4 + C_3\,x^3 + C_2\,x^2 + C_1\,x + C_0,$$

Assignment Project Exam Help

with only 5 multiplications, we can then obtain the product of numbers A and B simply as

$$A \cdot \qquad\qquad + C_1\,2^k + C_0,$$

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- If we manage to compute somehow the product polynomial

$$P_C(x) = P_A(x)P_B(x) = C_4\,x^4 + C_3\,x^3 + C_2\,x^2 + C_1\,x + C_0,$$

Assignment Project Exam Help

with only 5 multiplications, we can then obtain the product of numbers A and B simply as

$$A \cdot \qquad\qquad + C_1\,2^k + C_0,$$

https://eduassistpro.github.i

- Not

Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- If we manage to compute somehow the product polynomial

$$P_C(x) = P_A(x)P_B(x) = C_4\,x^4 + C_3\,x^3 + C_2\,x^2 + C_1\,x + C_0,$$

Assignment Project Exam Help

with only 5 multiplications, we can then obtain the product of numbers A and B simply as

$$A \cdot \qquad\qquad + C_1\,2^k + C_0,$$

https://eduassistpro.github.i

- Not

- Since the product polynomial $P_C(x) =$ need 5 values to **uniquely determine**

Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- If we manage to compute somehow the product polynomial

$$P_C(x) = P_A(x)P_B(x) = C_4\,x^4 + C_3\,x^3 + C_2\,x^2 + C_1\,x + C_0,$$

Assignment Project Exam Help

with only 5 multiplications, we can then obtain the product of numbers A and B simply as

$$A \cdot \qquad\qquad + C_1\,2^k + C_0,$$

https://eduassistpro.github.i

- Not

- Since the product polynomial $P_C(x) =$ need 5 values to **uniquely determine**

Add WeChat edu_assist_pr

- We choose **the smallest possible 5 i** heir absolute value),

The Karatsuba trick: slicing into 3 pieces

- If we manage to compute somehow the product polynomial

$$P_C(x) = P_A(x)P_B(x) = C_4\,x^4 + C_3\,x^3 + C_2\,x^2 + C_1\,x + C_0,$$

Assignment Project Exam Help

with only 5 multiplications, we can then obtain the product of numbers A and B simply as

$A \cdot$ $+ C_1\,2^k + C_0,$

https://eduassistpro.github.i

- Not

- Since the product polynomial $P_C(x) =$ need 5 values to **uniquely determine**

Add WeChat edu_assist_pr

- We choose **the smallest possible 5 i** heir absolute value), i.e., -2, -1, 0, 1, 2.

The Karatsuba trick: slicing into 3 pieces

- If we manage to compute somehow the product polynomial

$$P_C(x) = P_A(x)P_B(x) = C_4\,x^4 + C_3\,x^3 + C_2\,x^2 + C_1\,x + C_0,$$

Assignment Project Exam Help

with only 5 multiplications, we can then obtain the product of numbers A and B simply as

$A \cdot$ $+ C_1\,2^k + C_0,$

https://eduassistpro.github.i

- Not

- Since the product polynomial $P_C(x) =$ need 5 values to **uniquely determine**

Add WeChat edu_assist_pr

- We choose **the smallest possible 5 i** heir absolute value), i.e., -2, -1, 0, 1, 2.

- Thus, we compute

$$P_A(-2), P_A(-1), P_A(0), P_A(1), P_A(2)$$
$$P_B(-2), P_B(-1), P_B(0), P_B(1), P_B(2)$$

The Karatsuba trick: slicing into 3 pieces

- For $P_A(x) = A_2x^2 + A_1x + A_0$ we have

$$P_A(-2) = A_2(-2)^2 + A_1(-2) + A_0 = 4A_2 - 2A_1 + A_0$$

$$P_A(-1) = A_2(-1)^2 + A_1(-1) + A_0 = A_2 - A_1 + A_0$$

$$P_A(0) = A_20^2 + A_10 + A_0 = A_0$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- For $P_A(x) = A_2x^2 + A_1x + A_0$ we have

$$P_A(-2) = A_2(-2)^2 + A_1(-2) + A_0 = 4A_2 - 2A_1 + A_0$$

$$P_A(-1) = A_2(-1)^2 + A_1(-1) + A_0 = A_2 - A_1 + A_0$$

$$P_A(0) = A_20^2 + A_10 + A_0 = A_0$$

Assignment Project Exam Help

https://eduassistpro.github.i

- Simi

$$P_B(-2) = B_2(-2)^2 + B_1(-2)$$

Add WeChat edu_assist_pr

$$P_B(-1) = B_2(-1)^2 + B_1(-1)$$

$$P_B(0) = B_20^2 + B_10 + B_0 = B_0$$

$$P_B(1) = B_21^2 + B_11 + B_0 = B_2 + B_1 + B_0$$

$$P_B(2) = B_22^2 + B_12 + B_0 = 4B_2 + 2B_1 + B_0.$$

The Karatsuba trick: slicing into 3 pieces

- For $P_A(x) = A_2x^2 + A_1x + A_0$ we have

$$P_A(-2) = A_2(-2)^2 + A_1(-2) + A_0 = 4A_2 - 2A_1 + A_0$$

$$P_A(-1) = A_2(-1)^2 + A_1(-1) + A_0 = A_2 - A_1 + A_0$$

$$P_A(0) = A_20^2 + A_10 + A_0 = A_0$$

Assignment Project Exam Help

https://eduassistpro.github.i

- Simi

Add WeChat edu_assist_pr

$$P_B(-2) = B_2(-2)^2 + B_1(-2)$$

$$P_B(-1) = B_2(-1)^2 + B_1(-1)$$

$$P_B(0) = B_20^2 + B_10 + B_0 = B_0$$

$$P_B(1) = B_21^2 + B_11 + B_0 = B_2 + B_1 + B_0$$

$$P_B(2) = B_22^2 + B_12 + B_0 = 4B_2 + 2B_1 + B_0.$$

- These evaluations involve only additions because $2A = A + A$; $4A = 2A + 2A$.

The Karatsuba trick: slicing into 3 pieces

- Having obtained $P_A(-2), P_A(-1), P_A(0), P_A(1), P_A(2)$ and $P_B(-2), P_B(-1), P_B(0), P_B(1), P_B(2)$ we can now obtain $P_C(-2), P_C(-1), P_C(0), P_C(1), P_C(2)$ with only 5 multiplications of large numbers:

Assignment Project Exam Help

$$P_C(-2) = P_A(-2)P_B(-2)$$
$$= (A_0 \quad 2A_1 + 4A_2)(B_0 \quad 2B_1 + 4B_2)$$

https://eduassistpro.github.i

$$P_C(0) = P_A(0)P_B(0)$$
$$= A_0B_0$$

Add WeChat edu_assist_pr

$$P_C(1) = P_A(1)P_B(1)$$
$$= (A_0 + A_1 + A_2)(B_0 + B_1 + B_2)$$

$$P_C(2) = P_A(2)P_B(2)$$
$$= (A_0 + 2A_1 + 4A_2)(B_0 + 2B_1 + 4B_2)$$

The Karatsuba trick: slicing into 3 pieces

- Thus, if we represent the product $C(x) = P_A(x)P_B(x)$ in the coefficient form as $C(x) = C_4x^4 + C_3x^3 + C_2x^2 + C_1x + C_0$ we get

$$C_4(-2)^4 + C_3(-2)^3 + C_2(-2)^2 + C_1(-2) + C_0 = P_C(-2) = P_A(-2)P_B(-2)$$
$$C_4(-1)^4 + C_3(-1)^3 + C_2(-1)^2 + C_1(-1) + C_0 = P_C(-1) = P_A(-1)P_B(-1)$$
$$C_4 + C_3 + C_2 + C_1 + C_0 = P_C(1) = P_A(1)P_B(1)$$
$$16C_4 + 8C_3 + 4C_2 + 2C_1 + C_0 = P_C(2) = P_A(2)P_B(2).$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- Thus, if we represent the product $C(x) = P_A(x)P_B(x)$ in the coefficient form as $C(x) = C_4x^4 + C_3x^3 + C_2x^2 + C_1x + C_0$ we get

$$C_4(-2)^4 + C_3(-2)^3 + C_2(-2)^2 + C_1(-2) + C_0 = P_C(-2) = P_A(-2)P_B(-2)$$
$$C_4(-1)^4 + C_3(-1)^3 + C_2(-1)^2 + C_1(-1) + C_0 = P_C(-1) = P_A(-1)P_B(-1)$$
$$C_0 = P_C(0) = P_A(0)P_B(0)$$
$$C_4 + C_3 + C_2 + C_1 + C_0 = P_C(1) = P_A(1)P_B(1)$$
$$16C_4 + 8C_3 + 4C_2 + 2C_1 + C_0 = P_C(2) = P_A(2)P_B(2).$$

- Simplifying the left side we obtain

$$16C_4 - 8C_3 + 4C_2 - 2C$$
$$C_4 - C_3 + C_2 - C_1 + C_0 = P_C(-1)$$
$$C_0 = P_C(0)$$
$$C_4 + C_3 + C_2 + C_1 + C_0 = P_C(1)$$
$$16C_4 + 8C_3 + 4C_2 + 2C_1 + C_0 = P_C(2)$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- Solving this system of linear equations for C_0, C_1, C_2, C_3, C_4 produces (as an exercise solve this system by hand, using the Gaussian elimination)

$$C_0 = P_C(0)$$

$$C_1 = -\frac{P_C(-2)}{12} - \frac{2P_C(-1)}{3} + \frac{2P_C(1)}{3} - \frac{P_C(2)}{12} + \frac{P_C(2)}{4}$$

$$C_4 = \frac{C}{24} - \frac{C}{6} + \frac{C}{4} \quad \frac{C}{} + \frac{C}{}$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- Solving this system of linear equations for C_0, C_1, C_2, C_3, C_4 produces (as an exercise solve this system by hand, using the Gaussian elimination)

$$C_0 = P_C(0)$$

$$C_1 = \frac{P_C(-2)}{12} - \frac{2P_C(-1)}{3} + \frac{2P_C(1)}{3} - \frac{P_C(2)}{12}$$

$$C_2 = -\frac{P_C(-2)}{24} + \frac{2P_C(-1)}{3} - \frac{5P_C(0)}{4} + \frac{2P_C(1)}{3} - \frac{P_C(2)}{24}$$

$$C_4 = \frac{P_C(-2)}{24} - \frac{P_C(-1)}{6} + \frac{P_C(0)}{4} - \frac{P_C(1)}{6} + \frac{P_C(2)}{24}$$

- Note that these expressions do not involve any multiplications of large numbers and thus can be done in linear time.

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- Solving this system of linear equations for C_0, C_1, C_2, C_3, C_4 produces (as an exercise solve this system by hand, using the Gaussian elimination)

$$C_0 = P_C(0)$$

$$C_1 = -\frac{P_C(-2)}{12} - \frac{P_C(-1)}{3} + \frac{2P_C(1)}{3} - \frac{P_C(2)}{12}$$

$$\frac{P_C(-2)}{} \quad \frac{2P_C(-1)}{} \quad \frac{5P_C(0)}{} \quad \frac{2P_C(1)}{} \quad \frac{P_C(2)}{4}$$

$$C_4 = \frac{C}{24} - \frac{C}{6} + \frac{C}{4} \quad \frac{C}{} + \frac{C}{}$$

- Note that these expressions do not involve any multip numbers and thus can be done in linear time.
- With the coefficients C_0, C_1, C_2, C_3, C_4 ob polynomial $P_C(x) = C_0 + C_1 x + C_2 x^2 + C_3 x^3 + C_4 x^4$.

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- Solving this system of linear equations for C_0, C_1, C_2, C_3, C_4 produces (as an exercise solve this system by hand, using the Gaussian elimination)

$$C_0 = P_C(0)$$

$$C_1 = -\frac{P_C(-2)}{12} - \frac{P_C(-1)}{3} + \frac{2P_C(1)}{3} - \frac{P_C(2)}{12} + \frac{P_C(2)}{4}$$

$$C_4 = \frac{C}{24} - \frac{C}{6} + \frac{C}{4} \quad \frac{C}{} + \frac{C}{}$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

- Note that these expressions do not involve any multip numbers and thus can be done in linear time.
- With the coefficients C_0, C_1, C_2, C_3, C_4 ob polynomial $P_C(x) = C_0 + C_1 x + C_2 x^2 + C_3 x^3 + C_4 x^4$.
- We can now compute $P_C(2^k) = C_0 + C_1 2^k + C_2 2^{2k} + C_3 2^{3k} + C_4 2^{4k}$ in linear time, because computing $P_C(2^k)$ involves only binary shifts of the coefficients plus $O(k)$ additions.

The Karatsuba trick: slicing into 3 pieces

- Solving this system of linear equations for C_0, C_1, C_2, C_3, C_4 produces (as an exercise solve this system by hand, using the Gaussian elimination)

$$C_0 = P_C(0)$$

$$C_1 = \frac{P_C(-2)}{12} - \frac{2P_C(-1)}{3} + \frac{2P_C(1)}{3} - \frac{P_C(2)}{12}$$

$$\frac{P_C(-2)}{} \quad \frac{2P_C(-1)}{} \quad \frac{5P_C(0)}{} \quad \frac{2P_C(1)}{} \quad \frac{P_C(2)}{4}$$

$$C_4 = \frac{C}{24} - \frac{C}{6} + \frac{C}{4} \quad \frac{C}{} + \frac{C}{}$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

- Note that these expressions do not involve any multip numbers and thus can be done in linear time.
- With the coefficients C_0, C_1, C_2, C_3, C_4 ob polynomial $P_C(x) = C_0 + C_1 x + C_2 x^2 + C_3 x^3 + C_4 x^4$.
- We can now compute $P_C(2^k) = C_0 + C_1 2^k + C_2 2^{2k} + C_3 2^{3k} + C_4 2^{4k}$ in linear time, because computing $P_C(2^k)$ involves only binary shifts of the coefficients plus $O(k)$ additions.
- Thus we have obtained $A \cdot B = P_A(2^k) P_B(2^k) = P_C(2^k)$ with only 5 multiplications!

The Karatsuba trick: slicing into 3 pieces

- Solving this system of linear equations for C_0, C_1, C_2, C_3, C_4 produces (as an exercise solve this system by hand, using the Gaussian elimination)

$$C_0 = P_C(0)$$

$$C_1 = -\frac{P_C(-2)}{12} - \frac{2P_C(-1)}{3} + \frac{2P_C(1)}{3} - \frac{P_C(2)}{12}$$

$$\frac{P_C(-2)}{24} \quad \frac{2P_C(-1)}{6} \quad \frac{5P_C(0)}{4} \quad \frac{2P_C(1)}{4} \quad \frac{P_C(2)}{4}$$

$$C_4 = \frac{C}{24} - \frac{C}{6} + \frac{C}{4} \quad \frac{C}{} + \frac{C}{}$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

- Note that these expressions do not involve any multip numbers and thus can be done in linear time.
- With the coefficients C_0, C_1, C_2, C_3, C_4 ob polynomial $P_C(x) = C_0 + C_1 x + C_2 x^2 + C_3 x^3 + C_4 x^4$.
- We can now compute $P_C(2^k) = C_0 + C_1 2^k + C_2 2^{2k} + C_3 2^{3k} + C_4 2^{4k}$ in linear time, because computing $P_C(2^k)$ involves only binary shifts of the coefficients plus $O(k)$ additions.
- Thus we have obtained $A \cdot B = P_A(2^k) P_B(2^k) = P_C(2^k)$ with only 5 multiplications!Here is the complete algorithm:

1: **function** MULT(A, B)

2: obtain A_0, A_1, A_2 and B_0, B_1, B_2 such that $A = A_2\ 2^{2\,k} + A_1\ 2^k + A_0$; $B = B_2\ 2^{2\,k} + B_1\ 2^k + B_0$;

3: form polynomials $P_A(x) = A_2x^2 + A_1x + A_0$; $P_B(x) = B_2x^2 + B_1x + B_0$;

4:

$$P_A(-2) \leftarrow 4A_2 - 2A_1 + A_0 \qquad P_B(-2) \leftarrow 4B_2 - 2B_1 + B_0$$

$$P_A(-1) \leftarrow A_2 - A_1 + A_0 \qquad P_B(-1) \leftarrow B_2 - B_1 + B_0$$

$$P_A(0) \leftarrow A_0 \qquad P_B(0) \leftarrow B_0$$

$$P_A(1) \leftarrow A_2 + A_1 + A_0 \qquad P_B(1) \leftarrow B_2 + B_1 + B_0$$

$$P_A(2) \leftarrow 4A_2 + 2A_1 + A_0 \qquad P_B(2) \leftarrow 4B_2 + 2B_1 + B_0$$

5:

Assignment Project Exam Help
https://eduassistpro.github.i

6:

$$C_0 \leftarrow P_C(0); \qquad C_1 \leftarrow \frac{P_C(-2)}{12} - \frac{2P_C(\ \ 1)}{\ \ } \quad \frac{2P_C(1)}{\ \ } \quad \frac{P_C(2)}{\ \ }$$

Add WeChat edu_assist_pr

$$C_2 \leftarrow -\frac{P_C(-2)}{24} + \frac{2P_C(-1)}{3} - \frac{5P_C(0)}{4} + \frac{\ \ }{\ \ }$$

$$C_3 \leftarrow -\frac{P_C(-2)}{12} + \frac{P_C(-1)}{6} - \frac{P_C(1)}{6} + \frac{\ \ }{12}$$

$$C_4 \leftarrow \frac{P_C(-2)}{24} - \frac{P_C(-1)}{6} + \frac{P_C(0)}{4} - \frac{P_C(1)}{6} + \frac{P_C(2)}{24}$$

7: form $P_C(x) = C_4x^4 + C_3x^3 + C_2x^2 + C_1x + C_0$; compute
$P_C(2^k) = C_4 2^{4k} + C_3 2^{3k} + C_2 2^{2k} + C_1 2^k + C_0$

8: **return** $P_C(2^k) = A \cdot B$.

9: **end function**

The Karatsuba trick: slicing into 3 pieces

- How fast is this algorithm?

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- How fast is this algorithm?
- We have replaced a multiplication of two n bit numbers with 5 mul
and t

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- How fast is this algorithm?
- We have replaced a multiplication of two n bit numbers with 5 mul
and t
- thu

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- How fast is this algorithm?
- We have replaced a multiplication of two n bit numbers with 5
  mul
  and t
- thu
- We now apply the Master Theorem:

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

Assignment Project Exam Help

- How fast is this algorithm?
- We have replaced a multiplication of two n bit numbers with 5 mul
and t
- thu

https://eduassistpro.github.i

- We now apply the Master Theorem:
we have $a = 5$, $b = 3$, so we consider n

Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

Assignment Project Exam Help

- How fast is this algorithm?
- We have replaced a multiplication of two n bit numbers with 5 mul
and t
- thu

https://eduassistpro.github.i

Add WeChat edu_assist_pr

- We now apply the Master Theorem:
we have $a = 5$, $b = 3$, so we consider n
- Clearly, the first case of the MT applies and we get $T(n) = O(n^{\log_3 5}) < O(n^{1.47})$.

The Karatsuba trick: slicing into 3 pieces

- Recall that the original Karatsuba algorithm runs in time

$$n^{\log_2 3} \approx n^{1.58} > n^{1.47}.$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- Recall that the original Karatsuba algorithm runs in time

$$n^{\log_2 3} \approx n^{1.58} > n^{1.47}.$$

Assignment Project Exam Help

- Thu https://eduassistpro.github.i

Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- Recall that the original Karatsuba algorithm runs in time
$$n^{\log_2 3} \approx n^{1.58} > n^{1.47}.$$
- Thu
- Then why not slice numbers A and B i
Maybe we can get even faster algorithm?

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

The Karatsuba trick: slicing into 3 pieces

- Recall that the original Karatsuba algorithm runs in time
$$n^{\log_2 3} \approx n^{1.58} > n^{1.47}.$$
- Thu
- Then why not slice numbers A and B i
Maybe we can get even faster algorithm?
- The answer is, in a sense, BOTH yes and no, so lets see w
we slice numbers into $p+1$ many (approximately) equal slices, where $p = 1, 2, 3, \ldots$...

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

The general case - slicing the input numbers A, B into $p+1$ many slices

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

The general case - slicing the input numbers A, B into $p+1$ many slices

- For simplicity, let us assume A and B have exactly $(p+1)k$ bits (otherwise one of the slices will have to be shorter);

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

The general case - slicing the input numbers A, B into $p+1$ many slices

- For simplicity, let us assume A and B have exactly $(p+1)k$ bits (otherwise one of the slices will have to be shorter);
- Note: p is a fixed (smallish) number, a fixed parameter of our design – $p+1$ is the number of slices we are going to make, but k depends on the inp

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

The general case - slicing the input numbers A, B into $p+1$ many slices

- For simplicity, let us assume A and B have exactly $(p+1)k$ bits (otherwise one of the slices will have to be shorter);
- Note: p is a fixed (smallish) number, a fixed parameter of our design – $p+1$ is the number of slices we are going to make, but k depends on the inp
- Slic

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

$$B = \underset{p}{B_p} 2^{kp} + \underset{p-1}{B_{p-1}} \qquad \underset{0}{\qquad}$$

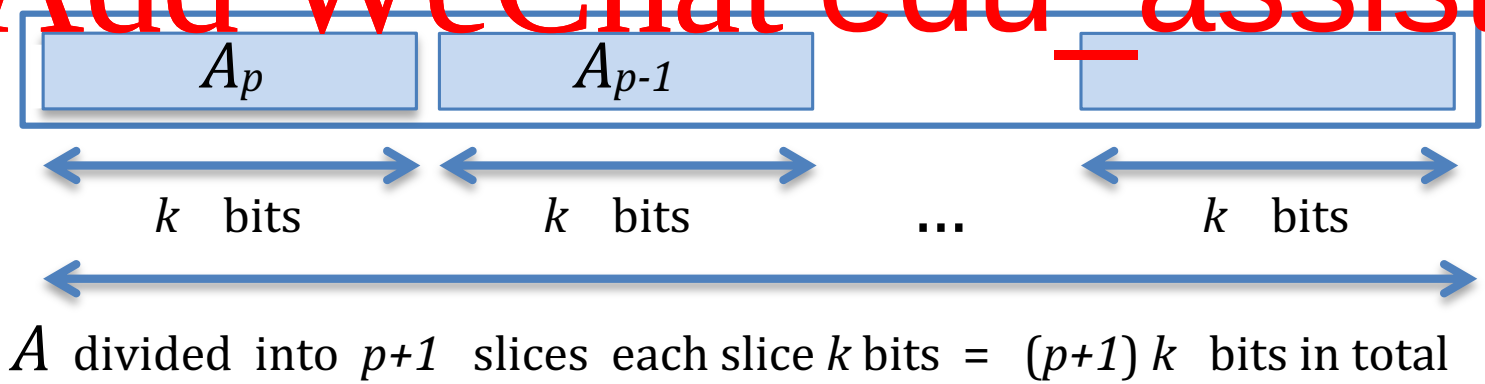
Generalizing Karatsuba's algorithm

The general case - slicing the input numbers A, B into $p+1$ many slices

- For simplicity, let us assume A and B have exactly $(p+1)k$ bits (otherwise one of the slices will have to be shorter);
- Note: p is a fixed (smallish) number, a fixed parameter of our design – $p+1$ is the number of slices we are going to make, but k depends on the inp
- Slic

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

$$B = B_p 2^{kp} + B_{p-1} \quad (p \text{ and } p-1 \text{ and } 0)$$



Generalizing Karatsuba's algorithm

- We form the naturally corresponding polynomials:

$$P_A(x) = A_p x^p + A_{p-1} x^{p-1} + \cdots + A_0$$
$$P_B(x) = B_p x^p + B_{p-1} x^{p-1} + \cdots + B_0$$

- As before, we have:

$$A = P_A(2^k); \quad B = P_B(2^k); \quad AB = P_A(2^k) P_B(2^k) = (P_A(x) \quad P_B(x))\,|_{x=2^k}$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- We form the naturally corresponding polynomials:

$$P_A(x) = A_p x^p + A_{p-1} x^{p-1} + \cdots + A_0$$
$$P_B(x) = B_p x^p + B_{p-1} x^{p-1} + \cdots + B_0$$

- As before, we have:
  $A = P_A(2^k)$; $B = P_B(2^k)$; $AB = P_A(2^k)P_B(2^k) = (P_A(x) \quad P_B(x))\,|_{x=2^k}$
- Sinc

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- We form the naturally corresponding polynomials:

$$P_A(x) = A_p x^p + A_{p-1} x^{p-1} + \cdots + A_0$$
$$P_B(x) = B_p x^p + B_{p-1} x^{p-1} + \cdots + B_0$$

- As before, we have:
$A = P_A(2^k)$; $B = P_B(2^k)$; $AB = P_A(2^k)P_B(2^k) = (P_A(x) \quad P_B(x))\,|_{x=2^k}$

- Sinc
we a

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- We form the naturally corresponding polynomials:

$$P_A(x) = A_p x^p + A_{p-1} x^{p-1} + \cdots + A_0$$
$$P_B(x) = B_p x^p + B_{p-1} x^{p-1} + \cdots + B_0$$

- As before, we have:
  $A = P_A(2^k)$; $B = P_B(2^k)$; $AB = P_A(2^k)P_B(2^k) = (P_A(x) \quad P_B(x))\,|_{x=2^k}$
- Sinc
  we a
  - we will first figure out how to multiply polyno
    $P_C(x) = P_A$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- We form the naturally corresponding polynomials:

$$P_A(x) = A_p x^p + A_{p-1} x^{p-1} + \cdots + A_0$$
$$P_B(x) = B_p x^p + B_{p-1} x^{p-1} + \cdots + B_0$$

- As before, we have:
$A = P_A(2^k)$; $B = P_B(2^k)$; $AB = P_A(2^k)P_B(2^k) = (P_A(x) \quad P_B(x))\,|_{x=2^k}$

- Sinc
we a
  - we will first figure out how to multiply polyno
  - then we evaluate $P_C(2^k)$.

$P_C(x) = P_A$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- We form the naturally corresponding polynomials:

$$P_A(x) = A_p x^p + A_{p-1} x^{p-1} + \cdots + A_0$$
$$P_B(x) = B_p x^p + B_{p-1} x^{p-1} + \cdots + B_0$$

- As before, we have:
$A = P_A(2^k)$; $B = P_B(2^k)$; $AB = P_A(2^k)P_B(2^k) = (P_A(x) \quad P_B(x))\,|_{x=2^k}$

- Sinc
we a
  - we will first figure out how to multiply polyno
$P_C(x) = P_A$
  - then we evaluate $P_C(2^k)$.

- Note that $P_C(x) = P_A(x) \cdot P_B(x)$ is of degree $2p$:

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- We form the naturally corresponding polynomials:

$$P_A(x) = A_p x^p + A_{p-1} x^{p-1} + \cdots + A_0$$
$$P_B(x) = B_p x^p + B_{p-1} x^{p-1} + \cdots + B_0$$

- As before, we have:
$A = P_A(2^k)$; $B = P_B(2^k)$; $AB = P_A(2^k)P_B(2^k) = (P_A(x) \quad P_B(x))\,|_{x=2^k}$

- Sinc
we a
  - we will first figure out how to multiply polyno
$P_C(x) = P_A$
  - then we evaluate $P_C(2^k)$.

- Note that $P_C(x) = P_A(x) \cdot P_B(x)$ is of degree $2p$:

$$P_C(x) = \sum_{j=0}^{2p} C_j x^j$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- Example:

$$(A_3x^3 + A_2x^2 + A_1x + A_0)(B_3x^3 + B_2x^2 + B_1x + B_0) =$$
$$A_3B_3x^6 + (A_2B_3 + A_3B_2)x^5 + (A_1B_3 + A_2B_2 + A_3B_1)x^4$$
$$+(A_0B_3 + A_1B_2 + A_2B_1 + A_3B_0)x^3 + (A_0B_2 + A_1B_1 + A_2B_0)x^2$$
$$+(A_0B_1 + A_1B_0)x + A_0B_0$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- Example:

$$(A_3x^3 + A_2x^2 + A_1x + A_0)(B_3x^3 + B_2x^2 + B_1x + B_0) =$$
$$A_3B_3x^6 + (A_2B_3 + A_3B_2)x^5 + (A_1B_3 + A_2B_2 + A_3B_1)x^4$$
$$+(A_0B_3 + A_1B_2 + A_2B_1 + A_3B_0)x^3 + (A_0B_2 + A_1B_1 + A_2B_0)x^2$$
$$+(A_0B_1 + A_1B_0)x + A_0B_0$$

- In ge

$$P_B(x) = B_px \quad + B_{p-1}x \qquad + \qquad + B_0$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- Example:

$$(A_3x^3 + A_2x^2 + A_1x + A_0)(B_3x^3 + B_2x^2 + B_1x + B_0) =$$
$$A_3B_3x^6 + (A_2B_3 + A_3B_2)x^5 + (A_1B_3 + A_2B_2 + A_3B_1)x^4$$
$$+(A_0B_3 + A_1B_2 + A_2B_1 + A_3B_0)x^3 + (A_0B_2 + A_1B_1 + A_2B_0)x^2$$
$$+(A_0B_1 + A_1B_0)x + A_0B_0$$

- In ge

$$P_B(x) = B_px \quad + B_{p-1}x \qquad + \qquad + B_0$$

we have

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Generalizing Karatsuba's algorithm

- Example:

$$(A_3x^3 + A_2x^2 + A_1x + A_0)(B_3x^3 + B_2x^2 + B_1x + B_0) =$$
$$A_3B_3x^6 + (A_2B_3 + A_3B_2)x^5 + (A_1B_3 + A_2B_2 + A_3B_1)x^4$$
$$+(A_0B_3 + A_1B_2 + A_2B_1 + A_3B_0)x^3 + (A_0B_2 + A_1B_1 + A_2B_0)x^2$$
$$+(A_0B_1 + A_1B_0)x + A_0B_0$$

Assignment Project Exam Help

- In ge

https://eduassistpro.github.i

$$P_B(x) = B_p x \quad + B_{p-1}x \qquad + \qquad + B_0$$

we have

Add WeChat edu_assist_pr

$$P_A(x) \cdot P_B(x) = \sum_{j=0}^{2p} \left(\sum_{i+k=j} A_i B_k \right) x^j = \sum_{j=0} C_j \, x^j$$

Generalizing Karatsuba's algorithm

- Example:

$$(A_3x^3 + A_2x^2 + A_1x + A_0)(B_3x^3 + B_2x^2 + B_1x + B_0) =$$
$$A_3B_3x^6 + (A_2B_3 + A_3B_2)x^5 + (A_1B_3 + A_2B_2 + A_3B_1)x^4$$
$$+(A_0B_3 + A_1B_2 + A_2B_1 + A_3B_0)x^3 + (A_0B_2 + A_1B_1 + A_2B_0)x^2$$
$$+(A_0B_1 + A_1B_0)x + A_0B_0$$

Assignment Project Exam Help

- In ge

https://eduassistpro.github.i

$$P_B(x) = B_p x \ + B_{p-1}x \quad + \quad + B_0$$

we have

Add WeChat edu_assist_pr

$$P_A(x) \cdot P_B(x) = \sum_{j=0}^{2p} \left(\sum_{i+k=j} A_i B_k \right) x^j = \sum_{j=0} C_j \, x^j$$

- We need to find the coefficients $C_j = \sum_{i+k=j} A_i B_k$ without performing $(p+1)^2$ many multiplications necessary to get all products of the form $A_i B_k$.

A VERY IMPORTANT DIGRESSION:

If you have two sequences $\vec{A} = (A_0, A_1, \ldots, A_{p-1}, A_p)$ and $\vec{B} = (B_0, B_1, \ldots, B_{m-1}, B_m)$, and if you form the two corresponding polynomials

$$P_A(x) = A_0 + A_1 x + \ldots + A_{p-1} x^{p-1} + A_p x^p$$

$$P_B(x) = B_0 + B_1 x + \ldots + B_{m-1} x^{m-1} + B_m x^m$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

A VERY IMPORTANT DIGRESSION:

If you have two sequences $\vec{A} = (A_0, A_1, \ldots, A_{p-1}, A_p)$ and $\vec{B} = (B_0, B_1, \ldots, B_{m-1}, B_m)$, and if you form the two corresponding polynomials

$$P_A(x) = A_0 + A_1 x + \ldots + A_{p-1} x^{p-1} + A_p x^p$$

$$P_B(x) = B_0 + B_1 x + \ldots + B_{m-1} x^{m-1} + B_m x^m$$

and if you mu

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

$$j=0 \qquad i+k=j \qquad\qquad j=0$$

A VERY IMPORTANT DIGRESSION:

If you have two sequences $\vec{A} = (A_0, A_1, \ldots, A_{p-1}, A_p)$ and $\vec{B} = (B_0, B_1, \ldots, B_{m-1}, B_m)$, and if you form the two corresponding polynomials

$$P_A(x) = A_0 + A_1 x + \ldots + A_{p-1} x^{p-1} + A_p x^p$$
$$P_B(x) = B_0 + B_1 x + \ldots + B_{m-1} x^{m-1} + B_m x^m$$

and if you mu

$$\sum_{j=0} \quad \sum_{i+k=j} \qquad \sum_{j=0}$$

then the sequence $\vec{C} = (C_0, C_1, \ldots, C_{p+m})$ of the coe polynomial, with these coefficients given by

$$C_j = \sum_{i+k=j} A_i B_k, \quad \text{for} \quad 0 \le j \le p+m,$$

is **extremely important** and is called the **LINEAR CONVOLUTION** of sequences $\vec{A}$ and $\vec{B}$ and is denoted by $\vec{C} = \vec{A} \star \vec{B}$.

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

AN IMPORTANT DIGRESSION:

- For example, if you have an audio signal and you want to emphasise the bass sounds, you would pass the sequence of discrete samples of the signal through a digital filter which amplifies the low frequencies more than the medium and the high audio frequencies.

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

AN IMPORTANT DIGRESSION:

- For example, if you have an audio signal and you want to emphasise the bass sounds, you would pass the sequence of discrete samples of the signal through a digital filter which amplifies the low frequencies more than the medium and the high audio frequencies.
- This is accomplished by computing the linear convolution of the sequence of discrete samples of the signal with a sequence of values which correspond to that filter, called *the impulse response* of the filter.

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

AN IMPORTANT DIGRESSION:

- For example, if you have an audio signal and you want to emphasise the bass sounds, you would pass the sequence of discrete samples of the signal through a digital filter which amplifies the low frequencies more than the medium and the high audio frequencies.
- This is accomplished by computing the linear convolution of the sequence of discrete samples of the signal with a sequence of values which correspond to that filter, called *the impulse response* of the filter.
- This
  the pr

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

AN IMPORTANT DIGRESSION:

- For example, if you have an audio signal and you want to emphasise the bass sounds, you would pass the sequence of discrete samples of the signal through a digital filter which amplifies the low frequencies more than the medium and the high audio frequencies.
- This is accomplished by computing the linear convolution of the sequence of discrete samples of the signal with a sequence of values which correspond to that filter, called *the impulse response* of the filter.
- This
  the pr
  1. e input

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

AN IMPORTANT DIGRESSION:

- For example, if you have an audio signal and you want to emphasise the bass sounds, you would pass the sequence of discrete samples of the signal through a digital filter which amplifies the low frequencies more than the medium and the high audio frequencies.
- This is accomplished by computing the linear convolution of the sequence of discrete samples of the signal with a sequence of values which correspond to that filter, called *the impulse response* of the filter.
- This
  the pr
  1. e input
  2. polynomial $P_B(x)$ whose coefficients
     called impulse response of the filter (they de
     filtering you want to do).

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

AN IMPORTANT DIGRESSION:

- For example, if you have an audio signal and you want to emphasise the bass sounds, you would pass the sequence of discrete samples of the signal through a digital filter which amplifies the low frequencies more than the medium and the high audio frequencies.
- This is accomplished by computing the linear convolution of the sequence of discrete samples of the signal with a sequence of values which correspond to that filter, called *the impulse response* of the filter.
- This
  the pr
  1. e input
  2. polynomial $P_B(x)$ whose coefficients
     called impulse response of the filter (they de
     filtering you want to do).
- Convolutions are bread-and-butter of signal proce
  is **extremely important** to find fast ways of multiplying two polynomials of possibly very large degrees.

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

AN IMPORTANT DIGRESSION:

- For example, if you have an audio signal and you want to emphasise the bass sounds, you would pass the sequence of discrete samples of the signal through a digital filter which amplifies the low frequencies more than the medium and the high audio frequencies.
- This is accomplished by computing the linear convolution of the sequence of discrete samples of the signal with a sequence of values which correspond to that filter, called *the impulse response* of the filter.
- This
  the pr
  1. e input
  2. polynomial $P_B(x)$ whose coefficients
     called impulse response of the filter (they de
     filtering you want to do).
- Convolutions are bread-and-butter of signal proce
  is **extremely important** to find fast ways of multiplying two polynomials of possibly very large degrees.
- In signal processing these degrees can be greater than 1000.

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

AN IMPORTANT DIGRESSION:

- For example, if you have an audio signal and you want to emphasise the bass sounds, you would pass the sequence of discrete samples of the signal through a digital filter which amplifies the low frequencies more than the medium and the high audio frequencies.
- This is accomplished by computing the linear convolution of the sequence of discrete samples of the signal with a sequence of values which correspond to that filter, called *the impulse response* of the filter.
- This
  the pr
  1. e input
  2. polynomial $P_B(x)$ whose coefficients
     called impulse response of the filter (they de
     filtering you want to do).
- Convolutions are bread-and-butter of signal proce
  is **extremely important** to find fast ways of multiplying two polynomials of possibly very large degrees.
- In signal processing these degrees can be greater than 1000.
- This is the main reason for us to study methods of fast computation of convolutions (aside of finding products of large integers, which is what we are doing at the moment).

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Coefficient vs value representation of polynomials

- Every polynomial $P_A(x)$ of degree p is uniquely determined by its values at any $p+1$ distinct input values $x_0, x_1, \ldots, x_p$:

$$P_A(x) \leftrightarrow \{(x_0, P_A(x_0)), (x_1, P_A(x_1)), \ldots, (x_p, P_A(x_p))\}$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Coefficient vs value representation of polynomials

- Every polynomial $P_A(x)$ of degree p is uniquely determined by its values at any $p+1$ distinct input values $x_0, x_1, \ldots, x_p$:

$$P_A(x) \leftrightarrow \{(x_0, P_A(x_0)), (x_1, P_A(x_1)), \ldots, (x_p, P_A(x_p))\}$$

- For ia a mat

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^p \\ 1 & x_1 & x_1^2 & \ldots & x_1^p \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_p & x_p^2 & \ldots & x_p^p \end{pmatrix} \begin{pmatrix} A & & P(x) \end{pmatrix} \quad (1)$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Coefficient vs value representation of polynomials

- Every polynomial $P_A(x)$ of degree p is uniquely determined by its values at any $p+1$ distinct input values $x_0, x_1, \ldots, x_p$:

$$P_A(x) \leftrightarrow \{(x_0, P_A(x_0)), (x_1, P_A(x_1)), \ldots, (x_p, P_A(x_p))\}$$

- For
mat
ia a

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^p \\ 1 & x_1 & x_1^2 & \ldots & x_1^p \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_p & x_p^2 & \ldots & x_p^p \end{pmatrix} \begin{pmatrix} A & & P(x) \end{pmatrix} \quad (1)$$

- It can be shown that if x_i are all distinct then this matrix is invertible.

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Coefficient vs value representation of polynomials

- Every polynomial $P_A(x)$ of degree p is uniquely determined by its values at any $p+1$ distinct input values $x_0, x_1, \ldots, x_p$:

$$P_A(x) \leftrightarrow \{(x_0, P_A(x_0)), (x_1, P_A(x_1)), \ldots, (x_p, P_A(x_p))\}$$

- For
mat
ia a

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^p \\ 1 & x_1 & x_1^2 & \ldots & x_1^p \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_p & x_p^2 & \ldots & x_p^p \end{pmatrix} \begin{pmatrix} A & & P(x) \end{pmatrix} \quad (1)$$

- It can be shown that if x_i are all distinct then this matrix is invertible.
- Such a matrix is called *the Vandermonde matrix*.

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Coefficient vs value representation of polynomials - ctd.

- Thus, if all x_i are all distinct, given any values $P_A(x_0), P_A(x_1), \ldots, P_A(x_p)$ the coefficients $A_0, A_1, \ldots, A_p$ of the polynomial $P_A(\cdot)$ are uniquely determined:

Assignment Project Exam Help

$$\begin{array}{c} A_0 \\ A_1 \end{array} \begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^p \\ 1 & x_1 & x_1^2 & \ldots & x_1^p \end{pmatrix} \begin{array}{c} \\ \\ \end{array} \begin{array}{c} \\ \\ \end{array} \begin{pmatrix} P_A(x_0) \\ P_A(x_1) \end{pmatrix} \tag{2}$$

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Coefficient vs value representation of polynomials - ctd.

- Thus, if all x_i are all distinct, given any values $P_A(x_0), P_A(x_1), \ldots, P_A(x_p)$ the coefficients $A_0, A_1, \ldots, A_p$ of the polynomial $P_A(\cdot)$ are uniquely determined:

Assignment Project Exam Help

$$\begin{pmatrix} A_0 \\ A_1 \end{pmatrix} \begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^p \\ 1 & x_1 & x_1^2 & \ldots & x_1^p \end{pmatrix} = \begin{pmatrix} P_A(x_0) \\ P_A(x_1) \end{pmatrix} \quad (2)$$

https://eduassistpro.github.i

- Equations (1) and (2) show how we can commute be

Add WeChat edu_assist_pr

Coefficient vs value representation of polynomials - ctd.

- Thus, if all x_i are all distinct, given any values $P_A(x_0), P_A(x_1), \ldots, P_A(x_p)$ the coefficients $A_0, A_1, \ldots, A_p$ of the polynomial $P_A(x)$ are uniquely determined:

$$\begin{pmatrix} A_0 \\ A_1 \end{pmatrix} \begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^p \\ 1 & x_1 & x_1^2 & \ldots & x_1^p \end{pmatrix} = \begin{pmatrix} P_A(x_0) \\ P_A(x_1) \end{pmatrix} \tag{2}$$

- Equations (1) and (2) show how we can commute be
  1. a representation of a polynomial P $A_p, A_{p-1}, \ldots, A_0$, i.e. $P_A(x) = A_p$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Coefficient vs value representation of polynomials - ctd.

- Thus, if all x_i are all distinct, given any values $P_A(x_0), P_A(x_1), \ldots, P_A(x_p)$ the coefficients $A_0, A_1, \ldots, A_p$ of the polynomial $P_A(x)$ are uniquely determined:

Assignment Project Exam Help

$$\begin{pmatrix} A_0 \\ A_1 \end{pmatrix} \begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^p \\ 1 & x_1 & x_1^2 & \ldots & x_1^p \end{pmatrix} \begin{pmatrix} P_A(x_0) \\ P_A(x_1) \end{pmatrix} \tag{2}$$

https://eduassistpro.github.i

- Equations (1) and (2) show how we can commute be
  1. a representation of a polynomial P $A_p, A_{p-1}, \ldots, A_0$, i.e. $P_A(x) = A_p$
  2. a representation of a polynomial $P_A(x)$ via its values

Add WeChat edu_assist_pr

$$P_A(x) \leftrightarrow \{(x_0, P_A(x_0)), (x_1, P_A(x_1)), \ldots, (x_p, P_A(x_p))\}$$

Coefficient vs value representation of polynomials- ctd.

- If we fix the inputs $x_0, x_1, \ldots, x_p$ then commuting between a representation of a polynomial $P_A(x)$ via its coefficients and a representation via its values at these points is done via the following two matrix multiplications, with matrices made up from constants:

Assignment Project Exam Help

$$P_A(x_0) \quad \begin{matrix} 1 & x_0 & x_0^2 & \ldots & x_0^p \end{matrix} \quad A_0$$

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Coefficient vs value representation of polynomials- ctd.

- If we fix the inputs $x_0, x_1, \ldots, x_p$ then commuting between a representation of a polynomial $P_A(x)$ via its coefficients and a representation via its values at these points is done via the following two matrix multiplications, with matrices made up from constants:

$$\begin{array}{ccc} P_A(x_0) & & \begin{matrix} 1 & x_0 & x_0^2 & \ldots & x_0^p \end{matrix} & & A_0 \\ & & \vdots & & \vdots \end{array}$$

$$\begin{pmatrix} A_0 \\ A_1 \\ \vdots \\ A_p \end{pmatrix} = \begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^p \\ 1 & x_1 & x_1^2 & \ldots & x_1^p \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_p & x_p^2 & \ldots & x_p^p \end{pmatrix} \begin{pmatrix} \\ \\ \\ P_A(x_p) \end{pmatrix}$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Coefficient vs value representation of polynomials- ctd.

- If we fix the inputs $x_0, x_1, \ldots, x_p$ then commuting between a representation of a polynomial $P_A(x)$ via its coefficients and a representation via its values at these points is done via the following two matrix multiplications, with matrices made up from **constants**:

$$\begin{array}{ccc} P_A(x_0) & & \begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^p \end{pmatrix} \\ & & \vdots \\ & & \end{array} \qquad A_0$$

$$\begin{pmatrix} A_0 \\ A_1 \\ \vdots \\ A_p \end{pmatrix} = \begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^p \\ 1 & x_1 & x_1^2 & \ldots & x_1^p \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_p & x_p^2 & \ldots & x_p^p \end{pmatrix} \begin{pmatrix} \\ \\ \\ P_A(x_p) \end{pmatrix}$$

- Thus, for fixed input values $x_0, \ldots, x_p$ this switch between the two kinds of representations is done in **linear time**!

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Our strategy to multiply polynomials fast:

1. Given two polynomials of degree at most p,

$$P_A(x) = A_p x^p + \ldots + A_0; \qquad P_B(x) = B_p x^p + \ldots + B_0$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Our strategy to multiply polynomials fast:

1. Given two polynomials of degree at most p,

$$P_A(x) = A_p x^p + \ldots + A_0; \qquad P_B(x) = B_p x^p + \ldots + B_0$$

convert them into value representation at $2p+1$ distinct points $x_0, x_1, \ldots, x_{2p}$.

$$P_A(x) \leftrightarrow \{(x_0, P_A(x_0)), (x_1, P_A(x_1)), \ldots, (x_{2p}, P_A(x_{2p}))\}$$

$$))\}$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Our strategy to multiply polynomials fast:

1. Given two polynomials of degree at most p,

$$P_A(x) = A_p x^p + \ldots + A_0; \qquad P_B(x) = B_p x^p + \ldots + B_0$$

convert them into value representation at $2p+1$ distinct points $x_0, x_1, \ldots, x_{2p}$.

$$P_A(x) \leftrightarrow \{(x_0, P_A(x_0)), (x_1, P_A(x_1)), \ldots, (x_{2p}, P_A(x_{2p}))\}$$

$))\}$

- p

ther than just $p+1$ points!

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Our strategy to multiply polynomials fast:

1. Given two polynomials of degree at most p,

$$P_A(x) = A_p x^p + \ldots + A_0; \qquad P_B(x) = B_p x^p + \ldots + B_0$$

convert them into value representation at $2p+1$ distinct points $x_0, x_1, \ldots, x_{2p}$.

$$P_A(x) \leftrightarrow \{(x_0, P_A(x_0)), (x_1, P_A(x_1)), \ldots, (x_{2p}, P_A(x_{2p}))\}$$

$))\}$

- p ther than just $p+1$ points!

2. Multiply these two polynomials point-wise, using 2 ly.

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Our strategy to multiply polynomials fast:

1. Given two polynomials of degree at most p,

$$P_A(x) = A_p x^p + \ldots + A_0; \qquad P_B(x) = B_p x^p + \ldots + B_0$$

convert them into value representation at $2p+1$ distinct points $x_0, x_1, \ldots, x_{2p}$.

$$P_A(x) \leftrightarrow \{(x_0, P_A(x_0)), (x_1, P_A(x_1)), \ldots, (x_{2p}, P_A(x_{2p}))\}$$

$$))\}$$

- p

ther than just $p+1$ points!

2. Multiply these two polynomials point-wise, using 2 ly.

$$P_A(x)P_B(x) \leftrightarrow \{(x_0, \underbrace{P_A(x_0)P_B(x_0)}_{P_C(x_0)}), (x_1, \underbrace{P_A(x_1)}_{P_C(x_1)} \qquad \underbrace{(x_{2p})}_{P_C(x_{2p})})\}$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Our strategy to multiply polynomials fast:

1. Given two polynomials of degree at most p,

$$P_A(x) = A_p x^p + \ldots + A_0; \qquad P_B(x) = B_p x^p + \ldots + B_0$$

convert them into value representation at $2p+1$ distinct points $x_0, x_1, \ldots, x_{2p}$.

$$P_A(x) \leftrightarrow \{(x_0, P_A(x_0)), (x_1, P_A(x_1)), \ldots, (x_{2p}, P_A(x_{2p}))\}$$

$$))\}$$

- p

ther than just $p+1$ points!

2. Multiply these two polynomials point-wise, using 2 ly.

$$P_A(x)P_B(x) \leftrightarrow \{(x_0, \underbrace{P_A(x_0)P_B(x_0)}_{P_C(x_0)}), (x_1, \underbrace{P_A(x_1)}_{P_C(x_1)} \qquad \underbrace{(x_{2p})}_{P_C(x_{2p})})\}$$

3. Convert such value representation of $P_C(x) = P_A(x)P_B(x)$ back to coefficient form

$$P_C(x) = C_{2p}x^{2p} + C_{2p-1}x^{2p-1} + \ldots + C_1 x + C_0;$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Fast multiplication of polynomials - continued

- What values should we choose for $x_0, x_1, \ldots, x_{2p}$??

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Fast multiplication of polynomials - continued

- What values should we choose for $x_0, x_1, \ldots, x_{2p}$??
- Key idea: use $2p+1$ smallest possible integer values!

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Fast multiplication of polynomials - continued

- What values should we choose for $x_0, x_1, \ldots, x_{2p}$??
- Key idea: use $2p+1$ smallest possible integer values!

$$\{-p, -p+1, \ldots, -1, 0, 1, \ldots, p-1, p\}$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Fast multiplication of polynomials - continued

- What values should we choose for $x_0, x_1, \ldots, x_{2p}$??
- Key idea: use $2p+1$ smallest possible integer values!

$$\{-p, -p+1, \ldots, -1, 0, 1, \ldots, p-1, p\}$$

- So we $m \leq p$.

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Fast multiplication of polynomials - continued

- What values should we choose for $x_0, x_1, \ldots, x_{2p}$??
- Key idea: use $2p+1$ smallest possible integer values!

$$\{-p, -p+1, \ldots, -1, 0, 1, \ldots, p-1, p\}$$

- So we $m \leq p$.
- Rem A, B.

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Fast multiplication of polynomials - continued

- What values should we choose for $x_0, x_1, \ldots, x_{2p}$??
- Key idea: use $2p+1$ smallest possible integer values!

$$\{-p, -p+1, \ldots, -1, 0, 1, \ldots, p-1, p\}$$

Assignment Project Exam Help

- So we $m \leq p$.
- Rem A, B.
- Mul d can be don i

$$d \cdot A = \underbrace{A + A}$$

https://eduassistpro.github.i
Add WeChat edu_assist_pr

Fast multiplication of polynomials - continued

- What values should we choose for $x_0, x_1, \ldots, x_{2p}$??
- Key idea: use $2p+1$ smallest possible integer values!

$$\{-p, -p+1, \ldots, -1, 0, 1, \ldots, p-1, p\}$$

Assignment Project Exam Help

- So we $m \leq p.$
- Rem $A, B.$
- Mul d can be
don i

$$d \cdot A = \underbrace{A + A \quad \ldots}$$

https://eduassistpro.github.i
Add WeChat edu_assist_pr

- Thus, all the values

$$P_A(m) = A_p m^p + A_{p-1} m^{p-1} + \cdots + A_0 : \quad -p \leq m \leq p,$$

$$P_B(m) = B_p m^p + B_{p-1} m^{p-1} + \cdots + B_0 : \quad -p \leq m \leq p.$$

can be found in time linear in the number of bits of the input numbers!

Fast multiplication of polynomials - ctd.

- We now perform $2p+1$ **multiplications of large numbers** to obtain

$$P_A(-p)P_B(-p),\ \ldots\ ,\ P_A(-1)P_B(-1),\quad P_A(0)P_B(0),\quad P_A(1)P_B(1),\ \ldots\ ,\ P_A(p)P_B(p)$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Fast multiplication of polynomials - ctd.

- We now perform $2p+1$ **multiplications of large numbers** to obtain

$$P_A(-p)P_B(-p),\ \ldots\ ,\ P_A(-1)P_B(-1),\quad P_A(0)P_B(0),\quad P_A(1)P_B(1),\ \ldots\ ,\ P_A(p)P_B(p)$$

- For $P_C(x) = P_A(x)P_B(x)$ these products are $2p+1$ many values of $P_C(x)$:

$$P_C(-p) = P_A(\ \ p)P_B(\ \ p),\ \ldots\ ,\ P_C(0) = P_A(0)P_B(0),\ \ldots\ ,\ P_C(p) = P_A(p)P_B(p)$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Fast multiplication of polynomials - ctd.

- We now perform $2p+1$ **multiplications of large numbers** to obtain

$$P_A(-p)P_B(-p),\ \ldots\ ,\ P_A(-1)P_B(-1),\quad P_A(0)P_B(0),\quad P_A(1)P_B(1),\ \ldots\ ,\ P_A(p)P_B(p)$$

- For $P_C(x) = P_A(x)P_B(x)$ these products are $2p+1$ many values of $P_C(x)$:

$$P_C(-p) = P_A(\ \ p)P_B(\ \ p),\ \ldots\ ,\ P_C(0) = P_A(0)P_B(0),\ \ldots\ ,\ P_C(p) = P_A(p)P_B(p)$$

- Let $C(x)$, i.e., let

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Fast multiplication of polynomials - ctd.

- We now perform $2p+1$ **multiplications of large numbers** to obtain

$$P_A(-p)P_B(-p),\ \ldots,\ P_A(-1)P_B(-1),\quad P_A(0)P_B(0),\quad P_A(1)P_B(1),\ \ldots,\ P_A(p)P_B(p)$$

Assignment Project Exam Help

- For $P_C(x) = P_A(x)P_B(x)$ these products are $2p+1$ many values of $P_C(x)$:

$$P_C(-p) = P_A(-p)P_B(-p),\ \ldots,\ P_C(0) = P_A(0)P_B(0),\ \ldots,\ P_C(p) = P_A(p)P_B(p)$$

https://eduassistpro.github.i

- Let $C(x)$, i.e., let

Add WeChat edu_assist_pr

- We now have:

$$C_{2p}(-p)^{2p} + C_{2p-1}(-p)^{2p-1} + \cdots + C_0 = P_C(-p)$$
$$C_{2p}(-(p-1))^{2p} + C_{2p-1}(-(p-1))^{2p-1} + \cdots + C_0 = P_C(-(p-1))$$
$$\vdots$$
$$C_{2p}(p-1)^{2p} + C_{2p-1}(p-1)^{2p-1} + \cdots + C_0 = P_C(p-1)$$
$$C_{2p}p^{2p} + C_{2p-1}p^{2p-1} + \cdots + C_0 = P_C(p)$$

Fast multiplication of polynomials - ctd.

- This is just a system of linear equations, that can be solved for $C_0, C_1, \ldots, C_{2p}$:

$$\begin{pmatrix} 1 & -p & (-p)^2 & \ldots & (-p)^{2p} \\ 1 & -(p-1) & (-(p-1))^2 & \ldots & (-(p-1))^{2p} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & p & 1 & (p & 1)^2 & \ldots & (p & 1)^{2p} \\ 1 & & & p & & & & \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ C_{2p} & 1 \end{pmatrix} = \begin{pmatrix} P_C(-p) \\ P_C(-(p-1)) \\ \vdots \\ P_C(p-1) \\ P_C(p) \end{pmatrix}$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

- This is just a system of linear equations, that can be solved for $C_0, C_1, \ldots, C_{2p}$:

$$\begin{pmatrix} 1 & -p & (-p)^2 & \ldots & (-p)^{2p} \\ 1 & -(p-1) & (-(p-1))^2 & \ldots & (-(p-1))^{2p} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & p-1 & (p-1)^2 & \ldots & (p-1)^{2p} \\ 1 & p & p^2 & \ldots & p^{2p} \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ C_{2p} \end{pmatrix} = \begin{pmatrix} P_C(-p) \\ P_C(-(p-1)) \\ \vdots \\ P_C(p-1) \\ P_C(p) \end{pmatrix}$$

- i.e., w

$$\begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ C_{2p} \end{pmatrix} = \begin{pmatrix} 1 & -p & (-p)^2 & \ldots & (-p)^{2p} \\ 1 & -(p-1) & (-(p-1))^2 & \ldots & (-(p-1))^{2p} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & p-1 & (p-1)^2 & \ldots & (p-1)^{2p} \\ 1 & p & p^2 & \ldots & p^{2p} \end{pmatrix}^{-1} \begin{pmatrix} P_C(-p) \\ P_C(-(p-1)) \\ \vdots \\ P_C(p-1) \\ P_C(p) \end{pmatrix}$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Fast multiplication of polynomials - ctd.

- This is just a system of linear equations, that can be solved for $C_0, C_1, \ldots, C_{2p}$:

$$\begin{pmatrix} 1 & -p & (-p)^2 & \ldots & (-p)^{2p} \\ 1 & -(p-1) & (-(p-1))^2 & \ldots & (-(p-1))^{2p} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & p & 1 & (p & 1)^{2p} \\ 1 & & p & & \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ C_{2p} & 1 \end{pmatrix} = \begin{pmatrix} P_C(-p) \\ P_C(-(p-1)) \\ \vdots \\ P_C(p-1) \\ P_C(p) \end{pmatrix}$$

- i.e., w

$$\begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ C_{2p} \end{pmatrix} = \begin{pmatrix} 1 & -p & (-p)^2 & \ldots & & 2p & & -1 & & -p) \\ 1 & -(p-1) & (-(p-1))^2 & \ldots & & & & & & -1)) \\ \vdots & \vdots & \vdots & \vdots & & & & & & \vdots \\ 1 & p-1 & (p-1)^2 & \ldots & & & & & & -1) \\ 1 & p & p^2 & \ldots & & p^{2p} & & & & P_C(p) \end{pmatrix}$$

- But the inverse matrix also involves only constants depending on p only;

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Fast multiplication of polynomials - ctd.

- This is just a system of linear equations, that can be solved for $C_0, C_1, \ldots, C_{2p}$:

$$\begin{pmatrix} 1 & -p & (-p)^2 & \ldots & (-p)^{2p} \\ 1 & -(p-1) & (-(p-1))^2 & \ldots & (-(p-1))^{2p} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & p & 1 & (p & 1)^{2p} \\ 1 & & p & & \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ C_{2p} & 1 \end{pmatrix} = \begin{pmatrix} P_C(-p) \\ P_C(-(p-1)) \\ \vdots \\ P_C(p-1) \\ P_C(p) \end{pmatrix}$$

- i.e., w

$$\begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ C_{2p} \end{pmatrix} = \begin{pmatrix} 1 & -p & (-p)^2 & \ldots & & 2p & & -1 & & -p) \\ 1 & -(p-1) & (-(p-1))^2 & \ldots & & & & & & -1)) \\ \vdots & \vdots & \vdots & \vdots & & & & & & \vdots \\ 1 & p-1 & (p-1)^2 & \ldots & & & & & & -1) \\ 1 & p & p^2 & \ldots & & p^{2p} & & & & P_C(p) \end{pmatrix}$$

- But the inverse matrix also involves only constants depending on p only;
- Thus the coefficients C_i can be obtained in linear time.

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Fast multiplication of polynomials - ctd.

- This is just a system of linear equations, that can be solved for $C_0, C_1, \ldots, C_{2p}$:

$$\begin{pmatrix} 1 & -p & (-p)^2 & \ldots & (-p)^{2p} \\ 1 & -(p-1) & (-(p-1))^2 & \ldots & (-(p-1))^{2p} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & p & 1 & (p & 1)^{2p} \\ 1 & & p & & \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ C_{2p} \end{pmatrix} = \begin{pmatrix} P_C(-p) \\ P_C(-(p-1)) \\ \vdots \\ P_C(p-1) \\ P_C(p) \end{pmatrix}$$

Assignment Project Exam Help

- i.e., w

https://eduassistpro.github.i

$$\begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ C_{2p} \end{pmatrix} = \begin{pmatrix} 1 & -p & (-p)^2 & \ldots & & 2p & & -1 & & -p) \\ 1 & -(p-1) & (-(p-1))^2 & \ldots & & & & & & -1)) \\ \vdots & \vdots & \vdots & \vdots & & & & & & \vdots \\ 1 & p-1 & (p-1)^2 & \ldots & & & & & & -1) \\ 1 & p & p^2 & \ldots & & p^{2p} & & & & P_C(p) \end{pmatrix}$$

Add WeChat edu_assist_pr

- But the inverse matrix also involves only constants depending on p only;
- Thus the coefficients C_i can be obtained in linear time.
- So here is the algorithm we have just described:

1: **function** MULT(A, B)
2: **if** $|A| = |B| < p + 1$ **then return** AB
3: **else**
4: obtain $p + 1$ slices $A_0, A_1, \ldots, A_p$ and $B_0, B_1, \ldots, B_p$ such that

$$A = A_p 2^{p\,k} + A_{p-1} 2^{(p-1)\,k} + \ldots + A_0$$

$$B = B_p 2^{p\,k} + B_{p-1} 2^{(p-1)\,k} + \ldots + B_0$$

5: form polynomials

$$P_A(x) = A_p x^p + A_{p-1} x^{(p-1)} + \ldots + A_0$$

$$P_B(x) = B_p x^p + B_{p-1} x^{(p-1)} + \ldots + B_0$$

6: **f**
7:
8:
9: **end for**
10: compute $C_0,\ C_1, \ldots\ C_{2p}$ via

$$\begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ C_{2p} \end{pmatrix} = \begin{pmatrix} 1 & -A & (-p)^2 & \ldots & \\ 1 & -(p-1) & (-(p-1))^2 & \ldots & 1)) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & p & p^2 & \ldots & p^{2p} \end{pmatrix} \begin{pmatrix} \\ \\ \vdots \\ P_C(p) \end{pmatrix}.$$

11: form $P_C(x) = C_{2p} x^{2p} + \ldots + C_0$ and compute $P_C(2^k)$
12: **return** $P_C(2^k) = A \cdot B$
13: **end if**
14: **end function**

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

How fast is our algorithm?

- it is easy to see that the values of the two polynomials we are multiplying have at most $k + s$ bits where s is a constant which depends on p but does NOT depend on k:

$$P_A(m) = A_p m^p + A_{p-1} m^{p-1} + \cdots + A_0 ; \quad -p \leq m \leq p.$$

This is because each A_i is smaller than 2^k because each A_k has k bits; thus

$$P_A(m) < p^p(p+1) \quad 2^k \qquad \log_2 P_A(m) < \log_2(p^p(p+1)) + k = s + k$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

How fast is our algorithm?

- it is easy to see that the values of the two polynomials we are multiplying have at most $k + s$ bits where s is a constant which depends on p but does NOT depend on k:

$$P_A(m) = A_p m^p + A_{p-1} m^{p-1} + \cdots + A_0 ; \quad -p \leq m \leq p.$$

This is because each A_i is smaller than 2^k because each A_k has k bits; thus

$$P_A(m) < p^p(p+1) \quad 2^k \qquad \log_2 P_A(m) < \log_2(p^p(p+1)) + k = s + k$$

- Thu mult the nu $p+1$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

How fast is our algorithm?

- it is easy to see that the values of the two polynomials we are multiplying have at most $k + s$ bits where s is a constant which depends on p but does NOT depend on k:

$$P_A(m) = A_p m^p + A_{p-1} m^{p-1} + \cdots + A_0 ; \quad -p \leq m \leq p.$$

This is because each A_i is smaller than 2^k because each A_k has k bits; thus

$$P_A(m) < p^p(p+1) \quad 2^k \qquad \log_2 P_A(m) < \log_2(p^p(p+1)) + k = s + k$$

- Thu $p+1$ mult the nu
- So we get the following recurrence for the complexity of

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

How fast is our algorithm?

- it is easy to see that the values of the two polynomials we are multiplying have at most $k + s$ bits where s is a constant which depends on p but does NOT depend on k:

$$P_A(m) = A_p m^p + A_{p-1} m^{p-1} + \cdots + A_0 ; \quad -p \leq m \leq p.$$

This is because each A_i is smaller than 2^k because each A_k has k bits; thus

$$P_A(m) < p^p(p+1) \quad 2^k \qquad \log_2 P_A(m) < \log_2(p^p(p+1)) + k = s + k$$

- Thu $p+1$ mult the nu
- So we get the following recurrence for the complexity of

$$T((p+1)k) = (2p+1)$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

How fast is our algorithm?

- it is easy to see that the values of the two polynomials we are multiplying have at most $k+s$ bits where s is a constant which depends on p but does NOT depend on k:

$$P_A(m) = A_p m^p + A_{p-1} m^{p-1} + \cdots + A_0 \; ; \quad -p \le m \le p.$$

This is because each A_i is smaller than 2^k because each A_k has k bits; thus

$$P_A(m) \; < p^p(p+1) \quad 2^k \qquad \log_2 \; P_A(m) \; < \log_2(p^p(p+1)) + k = s + k$$

- Thu $p+1$ mult the nu
- So we get the following recurrence for the complexity of

$$T((p+1)k) = (2p+1)$$

- Let $n = (p+1)k$. Then

$$T(n) = \underbrace{(2p+1)}_{a} \; T\Big(\underbrace{\frac{n}{p+1}}_{b} + s\Big) + \frac{c}{p+1}\, n$$

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

How fast is our algorithm?

- it is easy to see that the values of the two polynomials we are multiplying have at most $k+s$ bits where s is a constant which depends on p but does NOT depend on k:

$$P_A(m) = A_p m^p + A_{p-1} m^{p-1} + \cdots + A_0 \; ; \quad -p \le m \le p.$$

This is because each A_i is smaller than 2^k because each A_k has k bits; thus

$$P_A(m) \; < p^p(p+1) \quad 2^k \qquad \log_2 \; P_A(m) \; < \log_2(p^p(p+1)) + k = s + k$$

- Thu $p+1$ mult the nu
- So we get the following recurrence for the complexity of

$$T((p+1)k) = (2p+1)$$

- Let $n = (p+1)k$. Then

$$T(n) = \underbrace{(2p+1)}_{a} \; T\Big(\underbrace{\frac{n}{p+1}}_{b} + s\Big) + \frac{c}{p+1}\, n$$

- Since s is constant, its impact can be neglected.

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

How fast is our algorithm?

$$T(n) = \underbrace{(2p+1)}_{a} T\left(\frac{n}{\underbrace{p+1}_{b}} + s\right) + \frac{c}{p+1} n$$

Assignment Project Exam Help

- Sinc
  $\log_b$

https://eduassistpro.github.i

Add WeChat edu_assist_pr

How fast is our algorithm?

$$T(n) = \underbrace{(2p+1)}_{a} T\left(\frac{n}{\underbrace{p+1}_{b}} + s\right) + \frac{c}{p+1} n$$

Assignment Project Exam Help

- Sinc
  $\log_b$
- Cons

https://eduassistpro.github.i

$\log_b a - \varepsilon$).

Add WeChat edu_assist_pr

How fast is our algorithm?

$$T(n) = \underbrace{(2p+1)}_{a}\, T\left(\frac{n}{\underbrace{p+1}_{b}} + s\right) + \frac{c}{p+1}\, n$$

Assignment Project Exam Help

- Sinc
  $\log_b$
- Cons
  $\log_b a - \varepsilon$).
- Thus, with $a = 2p+1$ and $b = p+1$ the first case of t

https://eduassistpro.github.i

Add WeChat edu_assist_pr

How fast is our algorithm?

$$T(n) = \underbrace{(2p+1)}_{a} T\left(\frac{n}{\underbrace{p+1}_{b}} + s\right) + \frac{c}{p+1} n$$

Assignment Project Exam Help

- Sinc
  $\log_b$
- Cons
- Thus, with $a = 2p+1$ and $b = p+1$ the first case of t
- so we get:

https://eduassistpro.github.i

$\log_b a - \varepsilon$).

Add WeChat edu_assist_pr

$$T(n) = \Theta\left(n^{\log_b a}\right) = \Theta\left(n^{\log_{p+1}(2p+1)}\right)$$

- Note that

$$n^{\log_{p+1}(2p+1)} < n^{\log_{p+1} 2(p+1)} = n^{\log_{p+1} 2 + \log_{p+1}(p+1)}$$
$$= n^{1+\log_{p+1} 2} = n^{1+\frac{1}{\log_2(p+1)}}$$

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

- Note that

$$n^{\log_{p+1}(2p+1)} < n^{\log_{p+1} 2(p+1)} = n^{\log_{p+1} 2 + \log_{p+1}(p+1)}$$
$$= n^{1+\log_{p+1} 2} = n^{1+\frac{1}{\log_2(p+1)}}$$

Assignment Project Exam Help

- Thu ily clos

https://eduassistpro.github.i

Add WeChat edu_assist_pr

- Note that

$$n^{\log_{p+1}(2p+1)} < n^{\log_{p+1} 2(p+1)} = n^{\log_{p+1} 2 + \log_{p+1}(p+1)}$$
$$= n^{1+\log_{p+1} 2} = n^{1+\frac{1}{\log_2(p+1)}}$$

Assignment Project Exam Help

- Thu ily clos
- Ho in time $n^{1.1}$?

https://eduassistpro.github.i

Add WeChat edu_assist_pr

- Note that

$$n^{\log_{p+1}(2p+1)} < n^{\log_{p+1} 2(p+1)} = n^{\log_{p+1} 2 + \log_{p+1}(p+1)}$$

$$= n^{1+\log_{p+1} 2} = n^{1+\frac{1}{\log_2(p+1)}}$$

Assignment Project Exam Help

- Thu ily clos

https://eduassistpro.github.i

- Ho in time $n^{1.1}$?

Add WeChat edu_assist_pr

$$n^{1.1} = n^{1+\frac{1}{\log_2(p+1)}} \to \frac{1}{\log_2(p} \to 0$$

- Note that

$$n^{\log_{p+1}(2p+1)} < n^{\log_{p+1} 2(p+1)} = n^{\log_{p+1} 2 + \log_{p+1}(p+1)}$$

$$= n^{1+\log_{p+1} 2} = n^{1+\frac{1}{\log_2(p+1)}}$$

Assignment Project Exam Help

- Thu ily clos

https://eduassistpro.github.i

- Ho in time $n^{1.1}$?

Add WeChat edu_assist_pr

$$n^{1.1} = n^{1+\frac{1}{\log_2(p+1)}} \to \frac{1}{\log_2(p} \to 0$$

- Thus, we would have to slice the input numbers into $2^{10} = 1024$ pieces!!

- We would have to evaluate polynomials $P_A(x)$ and $P_B(x)$ both of degree p at values up to p.

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

- We would have to evaluate polynomials $P_A(x)$ and $P_B(x)$ both of degree p at values up to p.
- However, $p = 2^{10}$, so evaluating $P_A(p) = A_p p^p + \ldots + A_0$ involves multiplication of A_p with $p^p = (2^{10})^{2^{10}} \simeq 1.27 \times 10^{3077}$.

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

- We would have to evaluate polynomials $P_A(x)$ and $P_B(x)$ both of degree p at values up to p.
- However, $p = 2^{10}$, so evaluating $P_A(p) = A_p p^p + \ldots + A_0$ involves multiplication of A_p with $p^p = (2^{10})^{2^{10}} \simeq 1.27 \times 10^{3077}$.
- Thu can
  the nt c is
  abs

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

- We would have to evaluate polynomials $P_A(x)$ and $P_B(x)$ both of degree p at values up to p.
- However, $p = 2^{10}$, so evaluating $P_A(p) = A_p p^p + \ldots + A_0$ involves multiplication of A_p with $p^p = (2^{10})^{2^{10}} \simeq 1.27 \times 10^{3077}$.
- Thu can **the** nt c is abs
- Consequently, slicing the input numbers in more than just a few slices results in a hopelessly slow algorithm, despite the f asymptotic bounds improve as we increase the nu

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

- We would have to evaluate polynomials $P_A(x)$ and $P_B(x)$ both of degree p at values up to p.
- However, $p = 2^{10}$, so evaluating $P_A(p) = A_p p^p + \ldots + A_0$ involves multiplication of A_p with $p^p = (2^{10})^{2^{10}} \simeq 1.27 \times 10^{3077}$.
- Thu can
  the nt c is
  abs
- Consequently, slicing the input numbers in more than just a few slices results in a hopelessly slow algorithm, despite the f
  asymptotic bounds improve as we increase the nu
- The moral is: **In practice, asymptotic es**
  size of the constants hidden by the O-notation are not estimated and found to be reasonably small!!!

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr

- **Crucial question:** Are there numbers $x_0, x_1, \ldots, x_p$ such that the size of x_i^p does not grow uncontrollably?

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

- **Crucial question:** Are there numbers $x_0, x_1, \ldots, x_p$ such that the size of x_i^p does not grow uncontrollably?

- Answer: YES; they are the complex numbers z_i lying on the unit circle, i.e., such that $|z_i| = 1$!

Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

- **Crucial question:** Are there numbers $x_0, x_1, \ldots, x_p$ such that the size of x_i^p does not grow uncontrollably?

- Answer: YES; they are the complex numbers z_i lying on the unit circle, i.e., such that $|z_i| = 1$!

Assignment Project Exam Help

- This motivates us to consider values of polynomials at inputs which are equally spaced complex numbers all lying on the unit circle.

https://eduassistpro.github.i

Add WeChat edu_assist_pr

- **Crucial question:** Are there numbers $x_0, x_1, \ldots, x_p$ such that the size of x_i^p does not grow uncontrollably?

- Answer: YES; they are the complex numbers z_i lying on the unit circle, i.e., such that $|z_i| = 1$!

Assignment Project Exam Help

- This motivates us to consider values of polynomials at inputs which are equally spaced complex numbers all lying on the unit circle.

- The s **sform**
(D
eval

https://eduassistpro.github.i

Add WeChat edu_assist_pr

- **Crucial question:** Are there numbers $x_0, x_1, \ldots, x_p$ such that the size of x_i^p does not grow uncontrollably?

- Answer: YES; they are the complex numbers z_i lying on the unit circle, i.e., such that $|z_i| = 1$!

Assignment Project Exam Help

- This motivates us to consider values of polynomials at inputs which are equally spaced complex numbers all lying on the unit circle.

- The s **sform**
(D
eval https://eduassistpro.github.i

- We will present a very fast algorithm for computin
the Fast Fourier Transform, abbrevi

Add WeChat edu_assist_pr

- **Crucial question:** Are there numbers $x_0, x_1, \ldots, x_p$ such that the size of x_i^p does not grow uncontrollably?

- Answer: YES; they are the complex numbers z_i lying on the unit circle, i.e., such that $|z_i| = 1$!

Assignment Project Exam Help

- This motivates us to consider values of polynomials at inputs which are equally spaced complex numbers all lying on the unit circle.

- The s **sform**
(D
eval https://eduassistpro.github.i

- We will present a very fast algorithm for computin
the Fast Fourier Transform, abbrevi

Add WeChat edu_assist_pr

- The Fast Fourier Transform is **the most**
and is thus arguably **the most important algorithm of all**.

- **Crucial question:** Are there numbers $x_0, x_1, \ldots, x_p$ such that the size of x_i^p does not grow uncontrollably?

- Answer: YES; they are the complex numbers z_i lying on the unit circle, i.e., such that $|z_i| = 1$!

Assignment Project Exam Help

- This motivates us to consider values of polynomials at inputs which are equally spaced complex numbers all lying on the unit circle.

- The s **sform**
(D
eval https://eduassistpro.github.i

- We will present a very fast algorithm for computin
the Fast Fourier Transform, abbrevi

Add WeChat edu_assist_pr

- The Fast Fourier Transform is **the most**
and is thus arguably **the most important algorithm of all**.

- Every mobile phone performs thousands of FFT runs each second, for example to compress your speech signal or to compress images taken by your camera, to mention just a few uses of the FFT.

PUZZLE!

The warden meets with 23 new prisoners when they arrive. He tells them, "You may meet today and plan a strategy. But after today, you will be in isolated cells and will have no communication with one another. In the prison there is a switch room, which contains two light switches labeled A and B, each of which can be in either the on or the off position. I am not telling you their present positions. The switches are not conn
inclined, I wi
This prison
move one, b
either. The
until I lead the next prisoner there, and he'll be instructed to do the same thing. I'm going to choose prisoners at random. I may choose the same g
a row, or I may jump around and come back. But, given enough ti
would eventually visit the switch room many times. At any ti
declare to me: "We have all visited the switch room. If it is true, the
be set free. If it is false, and somebody has not yet visited the switch room, you will be fed to the alligators."

What is the strategy the prisoners can devise to gain their freedom?

Assignment Project Exam Help
https://eduassistpro.github.i
Add WeChat edu_assist_pr