Memory-efficient polynomial arithmetic

Pascal Giorgi[1] **Bruno Grenet**[1] Daniel S. Roche[2]

Séminaire LACL — 26 nov. 2018

[1] LIRMM, Université de Montpellier

[2] CS Department, US Naval Academy

Polynomial arithmetic

- Multiplication: $\mathrm{M}(n)$
  - Naïve: $2n^2 + 2n - 1$
  - Karatsuba: $< 6.5n^{\log_2 3}$
  - Toom-3: $< 18.75n^{\log_3 5}$
  - FFT-based: $4.5n \log n + O(n)$ or $O(n \log n \log \log n)$

Polynomial arithmetic

- Multiplication: $\mathsf{M}(n)$
  - Naïve: $2n^2 + 2n - 1$
  - Karatsuba: $< 6.5n^{\log_2 3}$
  - Toom-3: $< 18.75n^{\log_3 5}$
  - FFT-based: $4.5n \log n + O(n)$ or $O(n \log n \log\log n)$
- Other tasks:
  - Euclidean division: $5\mathsf{M}(n) + o(\mathsf{M}(n))$
  - GCD: $O(\mathsf{M}(n) \log n)$
  - Evaluation & interpolation: $O(\mathsf{M}(n) \log n)$
  - Power series computations: $O(\mathsf{M}(n))$ or $O(\mathsf{M}(n) \log n)$
  - ...

Polynomial arithmetic

- Multiplication: $\mathsf{M}(n)$
  - Naïve: $2n^2 + 2n - 1$
  - Karatsuba: $< 6.5n^{\log_2 3}$
  - Toom-3: $< 18.75n^{\log_3 5}$
  - FFT-based: $4.5n \log n + O(n)$ or $O(n \log n \log \log n)$
- Other tasks:
  - Euclidean division: $5\mathsf{M}(n) + o(\mathsf{M}(n))$
  - GCD: $O(\mathsf{M}(n) \log n)$
  - Evaluation & interpolation: $O(\mathsf{M}(n) \log n)$
  - Power series computations: $O(\mathsf{M}(n))$ or $O(\mathsf{M}(n) \log n)$
  - ...

What about space complexity?

Space complexity of polynomial arithmetic

- Quadratic multiplication algorithm: $O(1)$ [1]
- Karatsuba, Toom-3, FFT: $O(n)$
- Other tasks: often $O(n)$

1. Models to be defined later.

Space complexity of polynomial arithmetic

- Quadratic multiplication algorithm: $O(1)$ [1]
- Karatsuba, Toom-3, FFT: $O(n)$
- Other tasks: often $O(n)$

- Improvements on Karatsuba's algorithm:
  - Thomé (2002): $n + O(\log n)$
  - Roche (2009): $O(\log n)$

  $\rightarrow$ time complexity multiplied by a constant

1. Models to be defined later.

Space complexity of polynomial arithmetic

- Quadratic multiplication algorithm: $O(1)$ [1]
- Karatsuba, Toom-3, FFT: $O(n)$
- Other tasks: often $O(n)$

- Improvements on Karatsuba's algorithm:
  - Thomé (2002): $n + O(\log n)$
  - Roche (2009): $O(\log n)$

  $\rightarrow$ time complexity multiplied by a constant

- Improvements on FFT-based algorithms:
  - Roche (2009): $O(1)$ if $n = 2^k$
  - Harvey & Roche (2010): $O(1)$

  $\rightarrow$ time complexity multiplied by a constant

1. Models to be defined later.

Space-complexity models

Algebraic-RAM machine:

$\rightarrow$ *Standard* registers of size $O(\log n)$

$\rightarrow$ *Algebraic* registers containing one coefficient

Space-complexity models

Algebraic-RAM machine:

$\rightarrow$ *Standard* registers of size $O(\log n)$

$\rightarrow$ *Algebraic* registers containing one coefficient

- Read-only input / write-only output
  - (Close to) classical complexity theory
  - Lower bound $\Omega(n^2)$ on time $\times$ space for multiplication

Space-complexity models

Algebraic-RAM machine:

$\rightarrow$ *Standard* registers of size $O(\log n)$

$\rightarrow$ *Algebraic* registers containing one coefficient

- Read-only input / write-only output
  - (Close to) classical complexity theory
  - Lower bound $\Omega(n^2)$ on time $\times$ space for multiplication
- Read-only input / read-write output
  - Thomé (2002), Roche (2009) and Harvey & Roche (2010)
  - *Reasonable* from a programmer's viewpoint

Space-complexity models

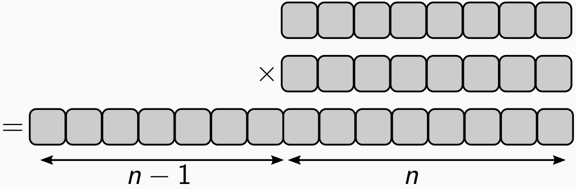
Algebraic-RAM machine:

$\rightarrow$ *Standard* registers of size $O(\log n)$

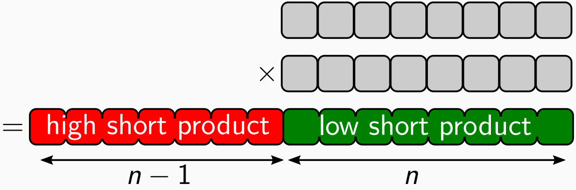
$\rightarrow$ *Algebraic* registers containing one coefficient

- Read-only input / write-only output
  - (Close to) classical complexity theory
  - Lower bound $\Omega(n^2)$ on time $\times$ space for multiplication
- Read-only input / read-write output
  - Thomé (2002), Roche (2009) and Harvey & Roche (2010)
  - *Reasonable* from a programmer's viewpoint
- Read-write input and output
  - Too permissive in general
  - Special case: inputs must be restored at the end

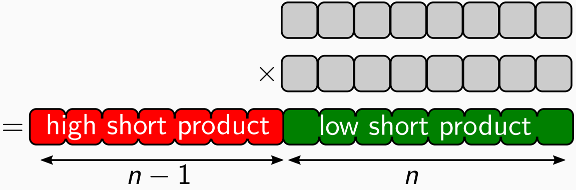
Short product



Short product

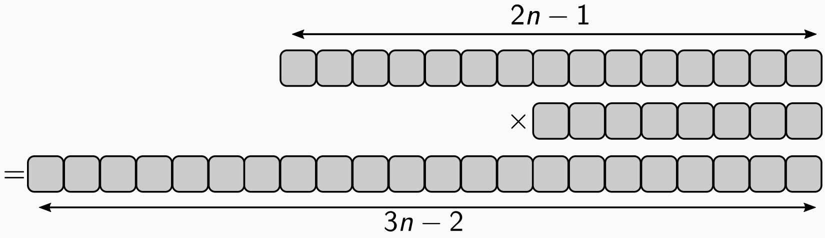


Short product

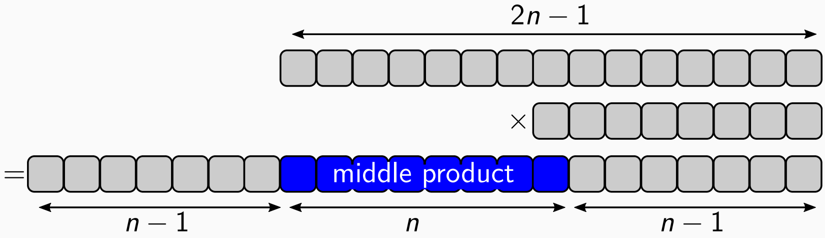


- Low short product: product of truncated power series
- Useful in other algorithms
- Time complexity: $\mathrm{M}(n)$
- Space complexity: $O(n)$

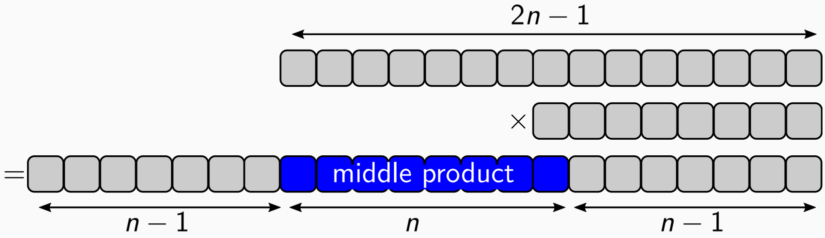
Middle product



Middle product

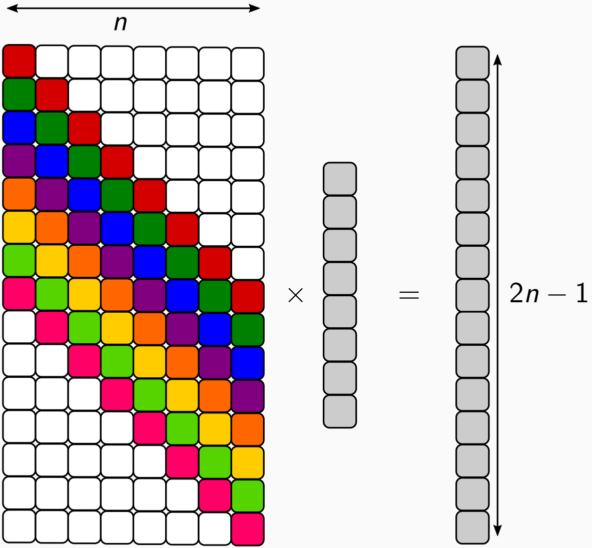


Middle product

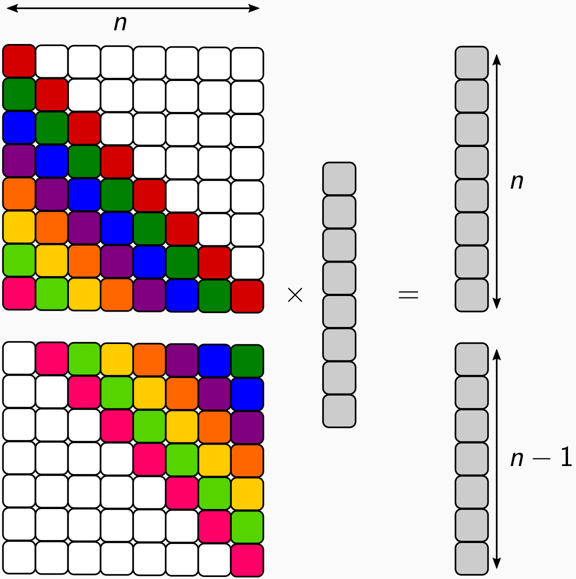


- Useful for Newton iteration
  - $G \leftarrow G(1 - GF) \bmod X^{2n}$ with $GF = 1 + X^n H$
  - division, square root, ...
- Time complexity: $\mathsf{M}(n) \rightarrow$ Tellegen's transposition
- Space complexity: $O(n)$

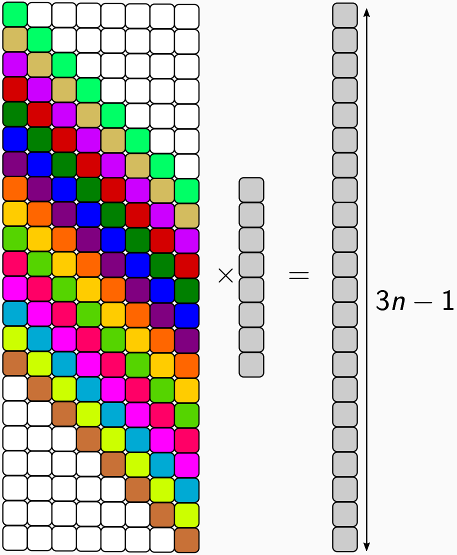
Multiplications as linear maps



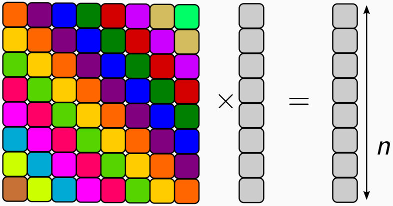
Multiplications as linear maps



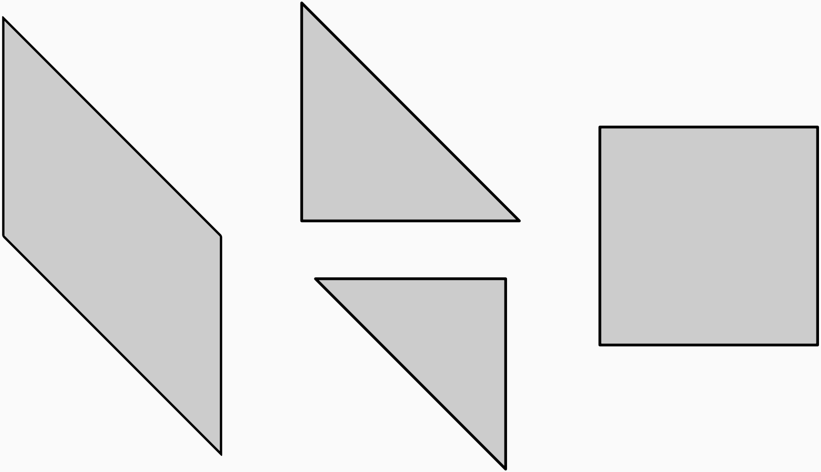
Multiplications as linear maps



Multiplications as linear maps



Multiplications as linear maps



Full product | Short products | Middle product

Outline

Space-preserving reductions

In-place algorithms from out-of-place algorithms

Space-preserving reductions

Reductions

Definitions.

- $\text{TISP}(t(n), s(n))$: decidable in time $t(n)$ and space $s(n)$
- $A \leq B$: A decidable with oracle B
  - constant number of calls to oracle
  - negligible extra time
  - without extra space ($O(1)$)
- $A \equiv B$: $A \leq B$ and $B \leq A$

Reductions

Definitions.

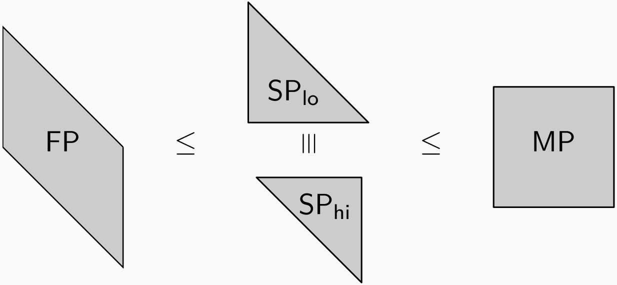
- $\mathrm{TISP}(t(n), s(n))$: decidable in time $t(n)$ and space $s(n)$
- $A \leq B$: A decidable with oracle B
  - constant number of calls to oracle
  - negligible extra time
  - without extra space ($O(1)$)
- $A \equiv B$: $A \leq B$ and $B \leq A$

Proposition.
If $B \in \mathrm{TISP}(t(n), s(n))$ and $A \leq B$, then

$$A \in \mathrm{TISP}(O(t(n)), s(n) + O(1))$$

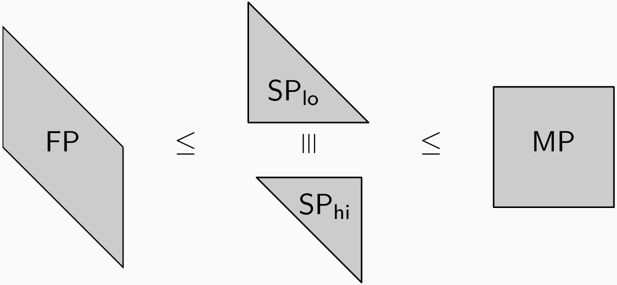
Results

Theorem.



Results

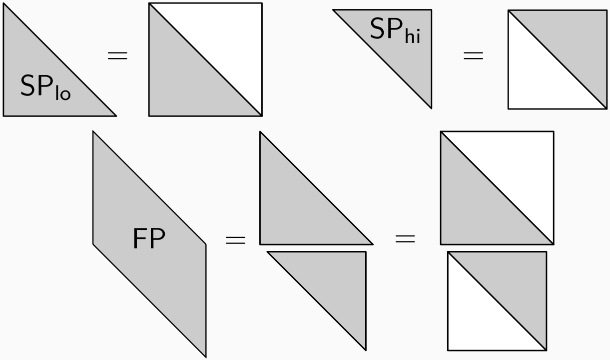
Theorem.



Remark.

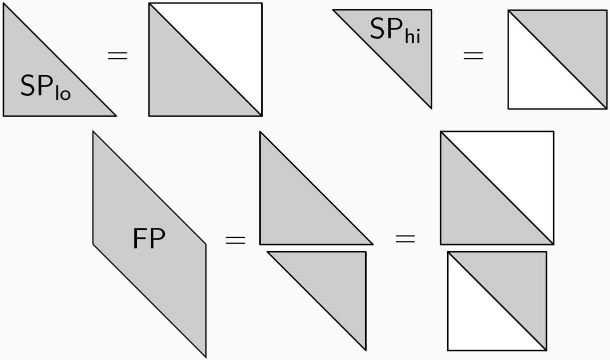
- FP : $n \times n \to 2n - 1$
- SP_{lo} : $n \times n \to n$; SP_{hi} : $n - 1 \times n - 1 \to n - 1$;
- MP : $2n - 1 \times n \to n$

Visual proof



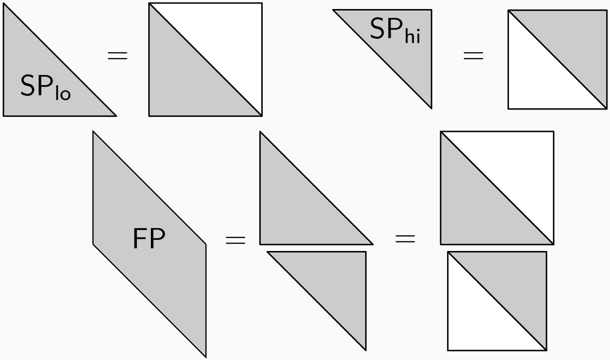
- Use of *fake padding* (in input, **not** in output!)

Visual proof



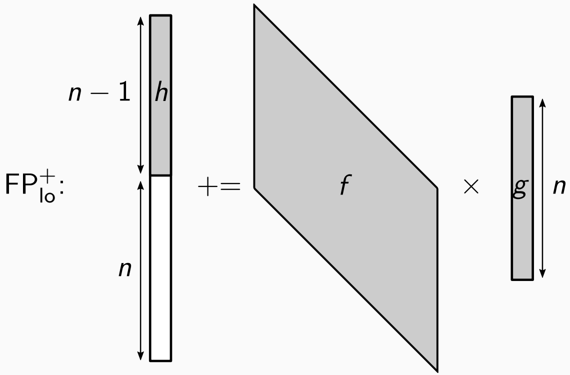
- Use of *fake padding* (in input, **not** in output!)
- $SP_{lo}(n) \leq MP(n)$; $SP_{hi}(n) \leq MP(n-1)$

Visual proof



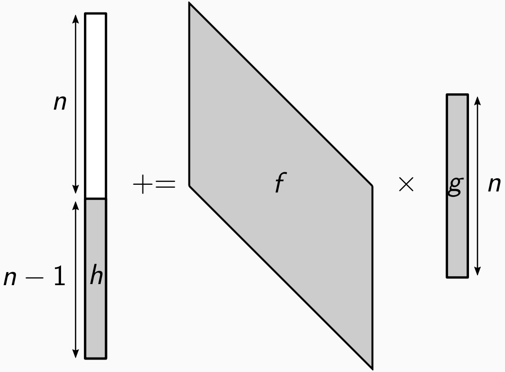
- Use of *fake padding* (in input, **not** in output!)
- $\mathrm{SP_{lo}}(n) \leq \mathrm{MP}(n)$; $\mathrm{SP_{hi}}(n) \leq \mathrm{MP}(n-1)$
- $\mathrm{FP}(n) \leq \mathrm{SP_{hi}}(n) + \mathrm{SP_{lo}}(n) \leq \mathrm{MP}(n) + \mathrm{MP}(n-1)$

Half-additive full product: $h \leftarrow h + f \cdot g$

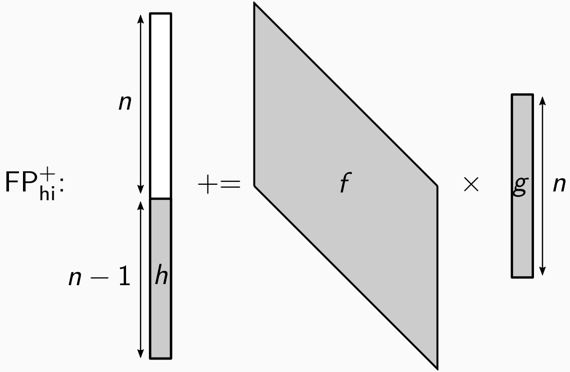


Half-additive full product: $h \leftarrow h + f \cdot g$

$\mathrm{FP}^{+}_{\mathrm{hi}}$:



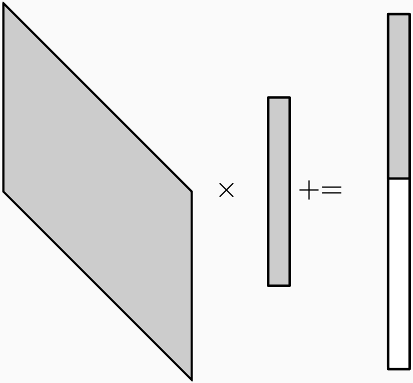
Half-additive full product: $h \leftarrow h + f \cdot g$



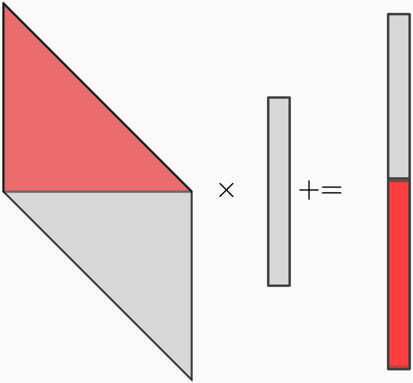
Remark. $\mathsf{FP}^+_{\mathsf{lo}} \equiv \mathsf{FP}^+_{\mathsf{hi}}$

Theorem. $\mathsf{FP}^+ \equiv \mathsf{SP}$

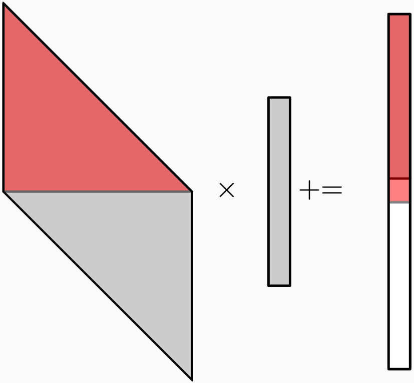
SP to FP^+



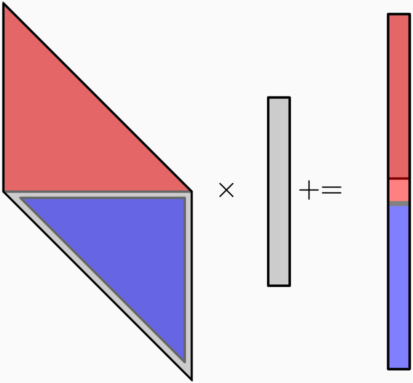
SP to FP$^+$



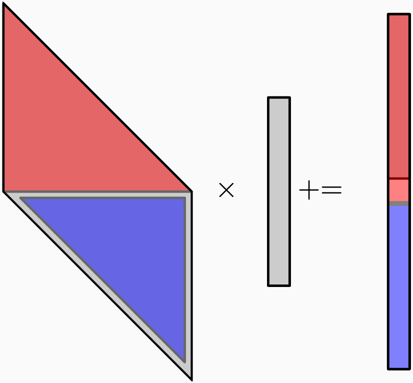
SP to FP$^+$



SP to FP$^+$



SP to FP^+



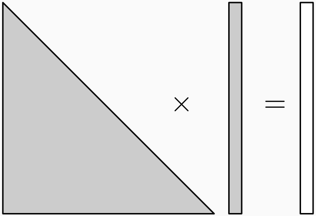
$$FP^+_{lo}(n) \leq SP_{lo}(n) + SP_{hi}(n) + n - 1$$

FP^+ to SP

$$\left(f_0 + X^{\lceil n/2 \rceil} f_1\right) \cdot \left(g_0 + X^{\lceil n/2 \rceil} g_1\right) = f_0 g_0 + X^{\lceil n/2 \rceil}(f_0 g_1 + f_1 g_0) \mod X^n$$

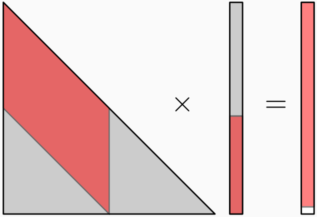
FP$^+$ to SP

$$\left(f_0 + X^{\lceil n/2\rceil} f_1\right)\cdot\left(g_0 + X^{\lceil n/2\rceil} g_1\right) = f_0 g_0 + X^{\lceil n/2\rceil}(f_0 g_1 + f_1 g_0) \mod X^n$$



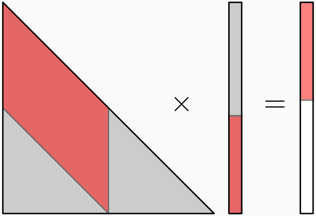
FP^+ to SP

$$\left(f_0 + X^{\lceil n/2 \rceil} f_1\right) \cdot \left(g_0 + X^{\lceil n/2 \rceil} g_1\right) = f_0 g_0 + X^{\lceil n/2 \rceil} (f_0 g_1 + f_1 g_0) \mod X^n$$



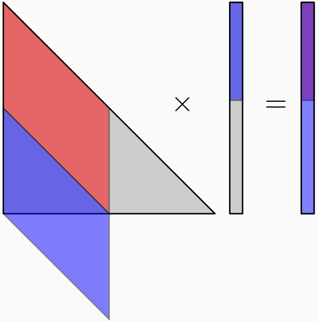
FP$^+$ to SP

$$\left(f_0 + X^{\lceil n/2 \rceil} f_1\right) \cdot \left(g_0 + X^{\lceil n/2 \rceil} g_1\right) = f_0 g_0 + X^{\lceil n/2 \rceil}(f_0 g_1 + f_1 g_0) \mod X^n$$



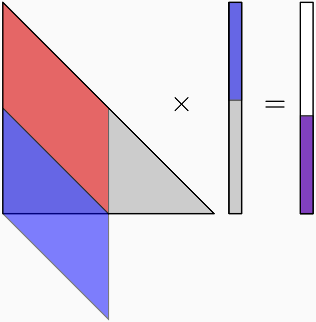
FP$^+$ to SP

$$\left(f_0 + X^{\lceil n/2 \rceil} f_1\right) \cdot \left(g_0 + X^{\lceil n/2 \rceil} g_1\right) = f_0 g_0 + X^{\lceil n/2 \rceil}(f_0 g_1 + f_1 g_0) \mod X^n$$



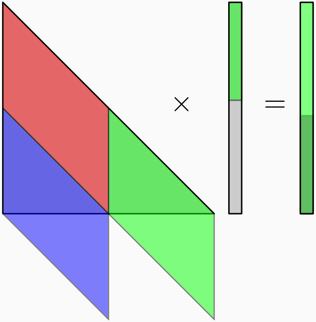
FP$^+$ to SP

$$\left(f_0 + X^{\lceil n/2 \rceil} f_1\right) \cdot \left(g_0 + X^{\lceil n/2 \rceil} g_1\right) = f_0 g_0 + X^{\lceil n/2 \rceil} ({\color{red} f_0 g_1} + {\color{blue} f_1 g_0}) \mod X^n$$



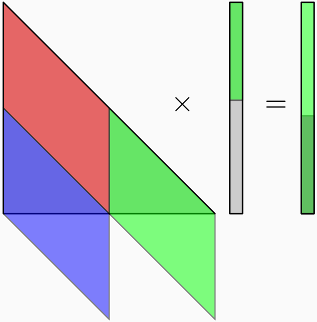
FP$^+$ to SP

$$\left(f_0 + X^{\lceil n/2 \rceil} f_1\right) \cdot \left(g_0 + X^{\lceil n/2 \rceil} g_1\right) = f_0 g_0 + X^{\lceil n/2 \rceil}(f_0 g_1 + f_1 g_0) \mod X^n$$



FP$^+$ to SP

$$\left(f_0 + X^{\lceil n/2 \rceil} f_1\right) \cdot \left(g_0 + X^{\lceil n/2 \rceil} g_1\right) = f_0 g_0 + X^{\lceil n/2 \rceil}(f_0 g_1 + f_1 g_0) \mod X^n$$



$$\mathrm{SP}_{\mathrm{lo}}(n) \leq \mathrm{FP}(\lfloor n/2 \rfloor) + \mathrm{FP}^+_{\mathrm{lo}}(\lfloor n/2 \rfloor) + \mathrm{FP}^+_{\mathrm{hi}}(\lceil n/2 \rceil)$$

Converse directions?

- From FP to SP:
  - problem with the output size
  - without space restriction: is $\mathsf{SP}(n) \simeq \mathsf{FP}(n/2)$?

Converse directions?

- From FP to SP:
  - problem with the output size
  - without space restriction: is $\mathsf{SP}(n) \simeq \mathsf{FP}(n/2)$?
- From FP to MP:
  - partial result: $\log(n)$ increase in time complexity
  - without space restriction: Tellegen's transposition principle

In-place algorithms from out-of-place algorithms

Framework

- In-place algorithms parametrized by out-of-place algorithm
  - Out-of-place: Uses cn extra space
  - Constant c known in the algorithm

Framework

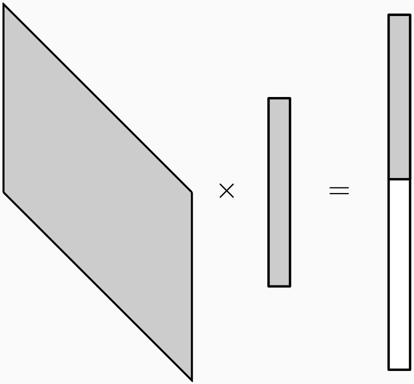
- In-place algorithms parametrized by out-of-place algorithm
  - Out-of-place: Uses cn extra space
  - Constant c known in the algorithm
- Goal:
  - Space complexity: $O(1)$
  - Time complexity: closest to the out-of-place algorithm

Framework

- In-place algorithms parametrized by out-of-place algorithm
  - Out-of-place: Uses cn extra space
  - Constant c known in the algorithm
- Goal:
  - Space complexity: $O(1)$
  - Time complexity: closest to the out-of-place algorithm
- Technique:
  - Oracle calls in smaller size
  - Recursive call

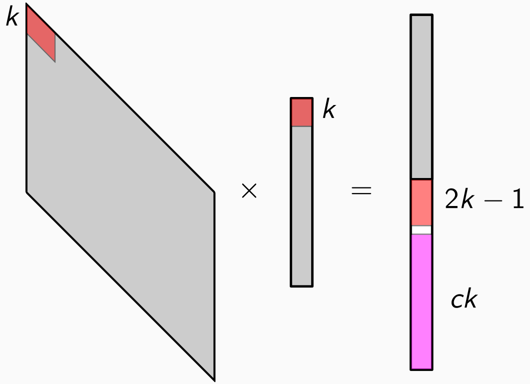
In-place FP$^+$

$$(f_0 + X^k \hat{f}) \cdot (g_0 + X^k \hat{g}) = f_0 g_0 + X^k (f_0 \hat{g} + \hat{f} g_0) + X^{2k} \hat{f} \hat{g}$$



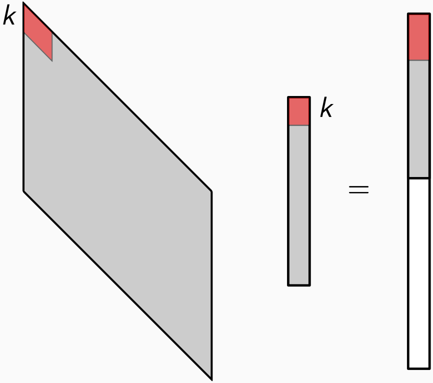
In-place FP$^+$

$$(f_0 + X^k \hat{f}) \cdot (g_0 + X^k \hat{g}) = f_0 g_0 + X^k (f_0 \hat{g} + \hat{f} g_0) + X^{2k} \hat{f} \hat{g}$$



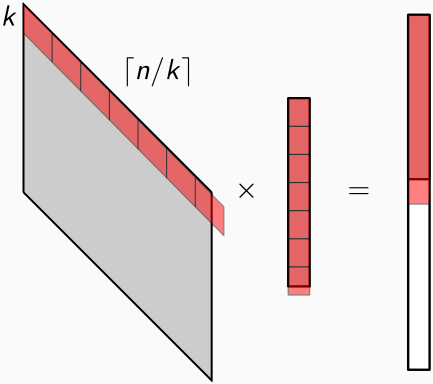
In-place FP$^+$

$$(f_0 + X^k \hat{f}) \cdot (g_0 + X^k \hat{g}) = f_0 g_0 + X^k (f_0 \hat{g} + \hat{f} g_0) + X^{2k} \hat{f} \hat{g}$$



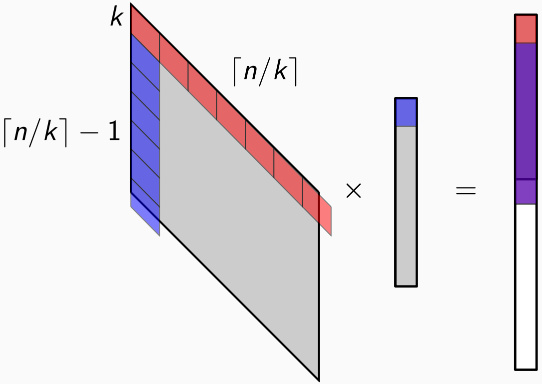
In-place FP$^+$

$$(f_0 + X^k \hat{f}) \cdot (g_0 + X^k \hat{g}) = f_0 g_0 + X^k(f_0 \hat{g} + \hat{f} g_0) + X^{2k} \hat{f} \hat{g}$$



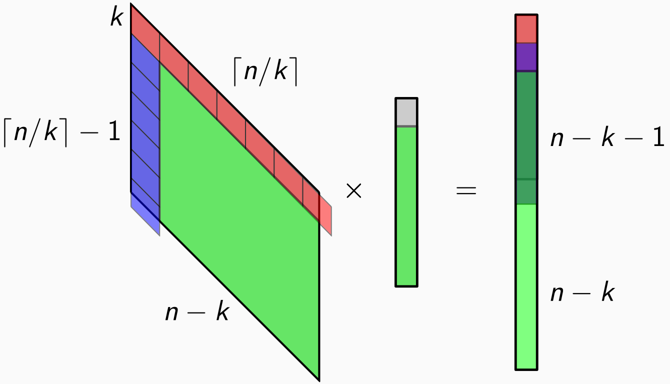
In-place FP$^+$

$$(f_0 + X^k \hat{f}) \cdot (g_0 + X^k \hat{g}) = f_0 g_0 + X^k (f_0 \hat{g} + \hat{f} g_0) + X^{2k} \hat{f} \hat{g}$$

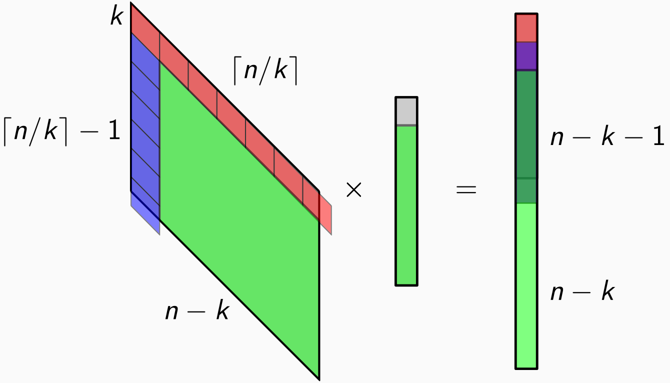


In-place FP$^+$

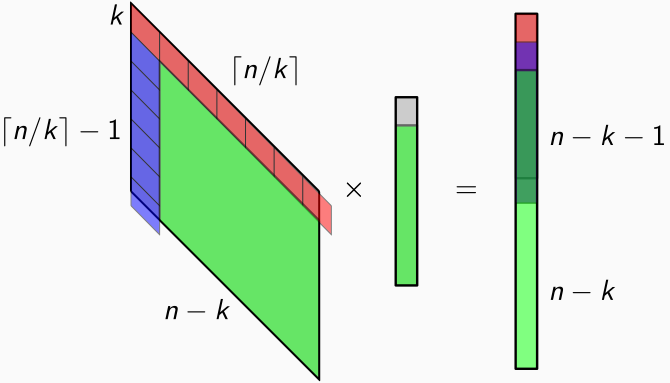
$$(f_0 + X^k \hat{f}) \cdot (g_0 + X^k \hat{g}) = f_0 g_0 + X^k(f_0 \hat{g} + \hat{f} g_0) + X^{2k} \hat{f} \hat{g}$$



Analysis

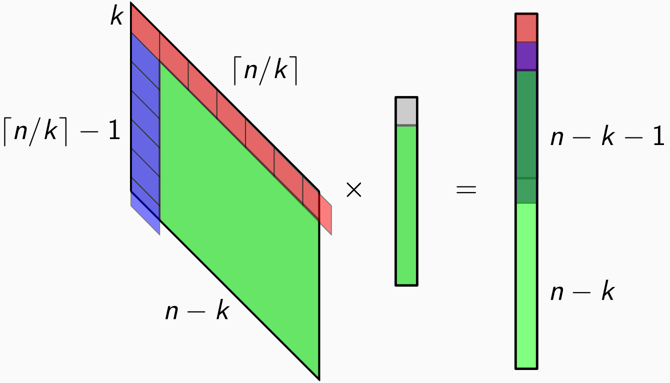


Analysis



- $ck + 2k - 1 \leq n - k \rightarrow k \leq \frac{n+1}{c+3}$
- $T(n) = (2\lceil n/k \rceil - 1)(\mathrm{M}(k) + 2k - 1) + T(n-k)$

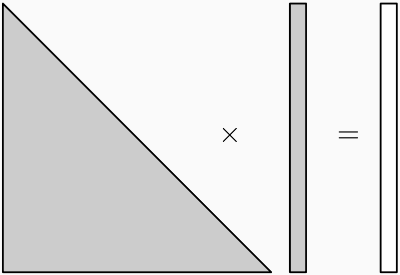
Analysis



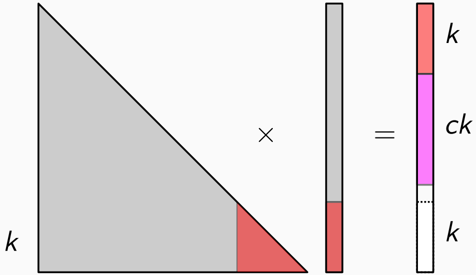
- $ck + 2k - 1 \leq n - k \rightarrow k \leq \frac{n+1}{c+3}$
- $T(n) = (2\lceil n/k \rceil - 1)(\mathrm{M}(k) + 2k - 1) + T(n - k)$

$$T(n) \leq (2c + 7)\mathrm{M}(n) + o(\mathrm{M}(n))$$

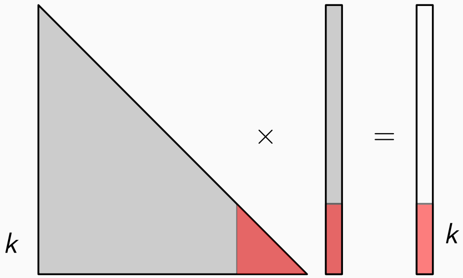
In-place short product



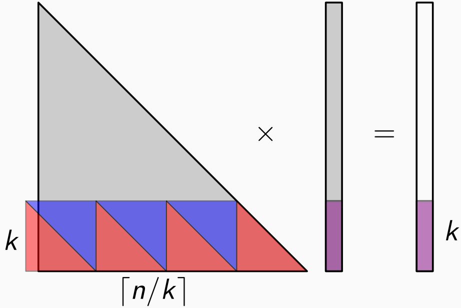
In-place short product



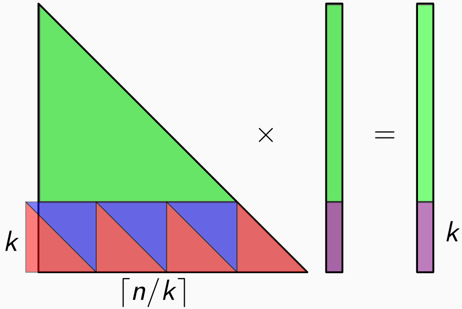
In-place short product



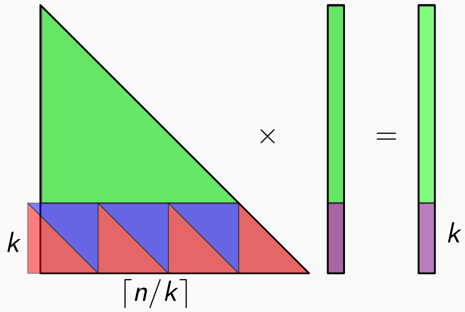
In-place short product



In-place short product

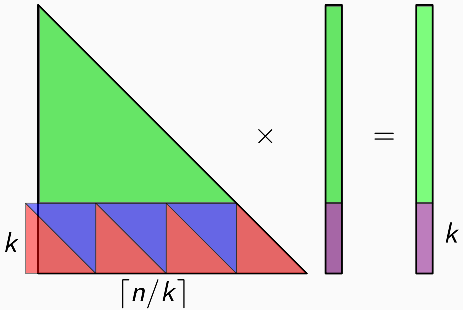


In-place short product



- $k \leq n/(c+2)$
- $T(n) = \lceil n/k \rceil \mathsf{M}(k) + (\lceil n/k \rceil - 1)\mathsf{M}(k-1) + 2k(\lceil n/k \rceil - 1) + T(n-k)$

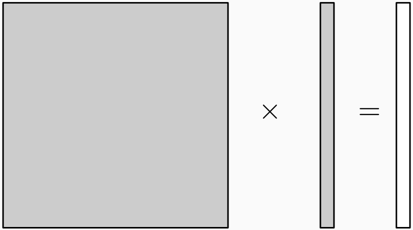
In-place short product



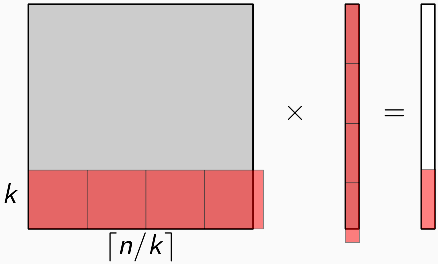
- $k \leq n/(c+2)$
- $T(n) = \lceil n/k \rceil \mathrm{M}(k) + (\lceil n/k \rceil - 1)\mathrm{M}(k-1) + 2k(\lceil n/k \rceil - 1) + T(n-k)$

$$T(n) \leq (2c+5)\mathrm{M}(n) + o(\mathrm{M}(n))$$

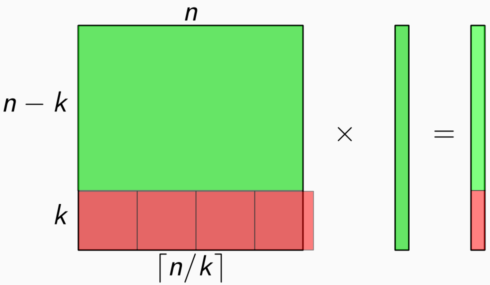
In-place middle product



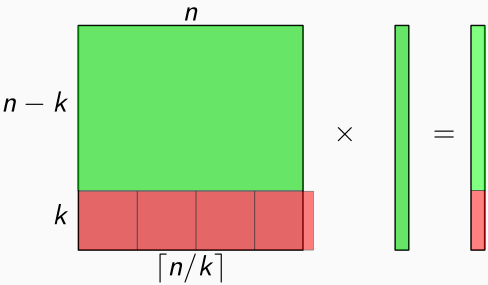
In-place middle product



In-place middle product

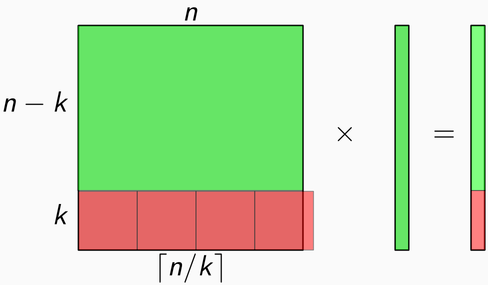


In-place middle product



- Only f's size decreases, not g!
- $T(n, m) = \lceil n/k \rceil \mathsf{M}(k) + T(n, m - k)$

In-place middle product



- Only f's size decreases, not g!
- $T(n,m) = \lceil n/k \rceil \mathsf{M}(k) + T(n, m-k)$

$$T(n,m) \le \mathsf{M}(n)\log_{1+1/c+1}(m) + o(\mathsf{M}(n)\log m)$$

Other operations

Work in progress!

Other operations

Work in progress!

- Use our in-place algorithms as building blocks
  - Newton iteration: division, square root, ...
  - Evaluation & interpolation

$\rightarrow$ (at most) $\log(n)$ increase in complexity

Other operations

Work in progress!

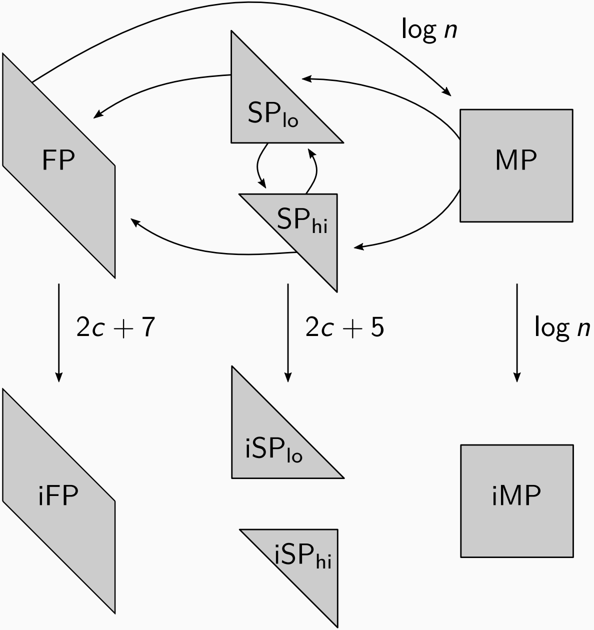
- Use our in-place algorithms as building blocks
  - Newton iteration: division, square root, ...
  - Evaluation & interpolation

$\rightarrow$ (at most) $\log(n)$ increase in complexity

Remark.

- In place: division with remainder
- Only quotient or only remainder: not clear
- Main difficulty: size of the output

Summary



Conclusion

- TISP-reductions between polynomial products
- Self-reductions to obtain in-place algorithms

Conclusion

- TISP-reductions between polynomial products
- Self-reductions to obtain in-place algorithms

Comparisons

- Better use specialized in-place algorithms. . .
- . . . when they exist!

Conclusion

- TISP-reductions between polynomial products
- Self-reductions to obtain in-place algorithms

Comparisons

- Better use specialized in-place algorithms...
- ... when they exist!

Main open problems

- Remove the $\log(n)$ for middle product or prove a lower bound
  - Karatsuba's algorithm with read-write restorable inputs
- General result on Tellegen's transposition principle

Conclusion

- TISP-reductions between polynomial products
- Self-reductions to obtain in-place algorithms

Comparisons

- Better use specialized in-place algorithms. . .
- . . . when they exist!

Main open problems

- Remove the $\log(n)$ for middle product or prove a lower bound
  - Karatsuba's algorithm with read-write restorable inputs
- General result on Tellegen's transposition principle

Thank you!