Memory Efficient Scheduling of Strassen-Winograd's Matrix Multiplication Algorithm

Brice Boyer[1] Jean-Guillaume Dumas[1]
Clément Pernet[2] Wei Zhou[3]

[1]LJK, Université de Grenoble, France
[2]INRIA-MOAIS, Université de Grenoble, France
[3]University of Waterloo, Canada

ISSAC '09

July 30, 2009

Motivation

Fact

Multiplying two $10\,000 \times 10\,000$ matrices on $\mathbf{F}_{65521}$ can be done in *only* $200s$ on a Xeon 3.6Ghz.

However, memory limits larger matrices multiplications.

Outline

1. Previous work about keeping memory requirements low
   - Existing Schedulings
   - Extra memory usage
   - Our Contribution

Outline

1. Previous work about keeping memory requirements low
   - Existing Schedulings
   - Extra memory usage
   - Our Contribution
2. A dynamic program generating schedules : a pebble game.
   - The dependency graph

Outline

1. Previous work about keeping memory requirements low
   - Existing Schedulings
   - Extra memory usage
   - Our Contribution
2. A dynamic program generating schedules : a pebble game.
   - The dependency graph
3. Building New Algorithms from the schedules
   - Building Blocks
   - A fully in place algorithm with constant inputs

Outline

1. Previous work about keeping memory requirements low
   - Existing Schedulings
   - Extra memory usage
   - Our Contribution
2. A dynamic program generating schedules : a pebble game.
   - The dependency graph
3. Building New Algorithms from the schedules
   - Building Blocks
   - A fully in place algorithm with constant inputs
4. Conclusion.

Outline

1. Previous work about keeping memory requirements low
   - Existing Schedulings
   - Extra memory usage
   - Our Contribution
2. A dynamic program generating schedules : a pebble game.
   - The dependency graph
3. Building New Algorithms from the schedules
   - Building Blocks
   - A fully in place algorithm with constant inputs
4. Conclusion.

Strassen-Winograd Algorithm

$$\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \times \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$

Strassen-Winograd Algorithm

$$\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \times \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$

Algorithm

$$\begin{array}{lll} S_1 \leftarrow A_{21} + A_{22} & & T_1 \leftarrow B_{12} - B_{11} \\ S_2 \leftarrow S_1 - A_{11} & & T_2 \leftarrow B_{22} - T_1 \\ S_3 \leftarrow A_{11} - A_{22} & & T_3 \leftarrow B_{22} - B_{12} \\ S_4 \leftarrow A_{12} - S_2 & & T_4 \leftarrow T_2 - B_{21} \end{array}$$

$$\begin{array}{lll} P_1 \leftarrow A_{11} \times B_{11} & & P_5 \leftarrow S_1 \times T_1 \\ P_2 \leftarrow A_{12} \times B_{21} & & P_6 \leftarrow S_2 \times T_2 \\ P_3 \leftarrow S_4 \times B_{22} & & P_7 \leftarrow S_3 \times T_3 \\ P_4 \leftarrow A_{22} \times T_4 & & \end{array}$$

$$\begin{array}{lll} \mathbf{U_1} \leftarrow P_1 + P_2 & & \mathbf{U_5} \leftarrow U_4 + P_3 \\ U_2 \leftarrow P_1 + P_6 & & \mathbf{U_6} \leftarrow U_3 - P_4 \\ U_3 \leftarrow U_2 + P_7 & & \mathbf{U_7} \leftarrow U_3 + P_5 \\ U_4 \leftarrow U_2 + P_5 & & \end{array}$$

Product $C \leftarrow A \times B$

A Strassen-Winograd scheduling (Douglas et al. '94)

#	operation	$loc.$	#	operation	$loc.$
1	$S_3 \leftarrow A_{11} - A_{21}$	X	12	$P_1 \leftarrow A_{11}B_{11}$	X
2	$T_3 \leftarrow B_{22} - B_{12}$	Y	13	$U_2 \leftarrow P_6 + P_1$	C_{12}
3	$P_7 \leftarrow S_3T_3$	C_{21}	14	$U_3 \leftarrow U_2 + P_7$	C_{21}
4	$S_1 \leftarrow A_{21} + A_{22}$	X	15	$U_4 \leftarrow U_2 + P_5$	C_{12}
5	$T_1 \leftarrow B_{12} - B_{11}$	Y	16	$\mathbf{U_7} \leftarrow U_3 + P_5$	C_{22}
6	$P_5 \leftarrow S_1T_1$	C_{22}	17	$\mathbf{U_5} \leftarrow U_4 + P_3$	C_{12}
7	$S_2 \leftarrow S_1 - A_{11}$	X	18	$T_4 \leftarrow T_2 - B_{21}$	Y
8	$T_2 \leftarrow B_{22} - T_1$	Y	19	$P_4 \leftarrow A_{22}T_4$	C_{11}
9	$P_6 \leftarrow S_2T_2$	C_{12}	20	$\mathbf{U_6} \leftarrow U_3 - P_4$	C_{21}
10	$S_4 \leftarrow A_{12} - S_2$	X	21	$P_2 \leftarrow A_{12}B_{21}$	C_{11}
11	$P_3 \leftarrow S_4B_{22}$	C_{11}	22	$\mathbf{U_1} \leftarrow -P_1 + P_2$	C_{11}

Product $C \leftarrow A \times B$

A Strassen-Winograd scheduling (Douglas et al. '94)

#	operation	*loc.*	#	operation	*loc.*
1	$S_3 \leftarrow A_{11} - A_{21}$	X	12	$P_1 \leftarrow A_{11}B_{11}$	X
2	$T_3 \leftarrow B_{22} - B_{12}$	Y	13	$U_2 \leftarrow P_6 + P_1$	C_{12}
3	$P_7 \leftarrow S_3T_3$	C_{21}	14	$U_3 \leftarrow U_2 + P_7$	C_{21}
4	$S_1 \leftarrow A_{21} + A_{22}$	X	15	$U_4 \leftarrow U_2 + P_5$	C_{12}
5	$T_1 \leftarrow B_{12} - B_{11}$	Y	16	$\mathbf{U_7} \leftarrow U_3 + P_5$	C_{22}
6	$P_5 \leftarrow S_1T_1$	C_{22}	17	$\mathbf{U_5} \leftarrow U_4 + P_3$	C_{12}
7	$S_2 \leftarrow S_1 - A_{11}$	X	18	$T_4 \leftarrow T_2 - B_{21}$	Y
8	$T_2 \leftarrow B_{22} - T_1$	Y	19	$P_4 \leftarrow A_{22}T_4$	C_{11}
9	$P_6 \leftarrow S_2T_2$	C_{12}	20	$\mathbf{U_6} \leftarrow U_3 - P_4$	C_{21}
10	$S_4 \leftarrow A_{12} - S_2$	X	21	$P_2 \leftarrow A_{12}B_{21}$	C_{11}
11	$P_3 \leftarrow S_4B_{22}$	C_{11}	22	$\mathbf{U_1} \leftarrow -P_1 + P_2$	C_{11}

2 temporaries

Product with accumulation

$$C \leftarrow \alpha A \times B + \beta C$$

A product with accumulation scheduling (Huss-Lederman et al. '96)

#	operation	$loc.$	#	operation	$loc.$
1	$S_1 \leftarrow A_{21} + A_{22}$	X	12	$S_4 \leftarrow A_{12} - S_2$	X
2	$T_1 \leftarrow B_{12} - B_{11}$	Y	13	$T_4 \leftarrow T_2 - B_{21}$	Y
3	$P_5 \leftarrow \alpha S_1 T_1$	Z	14	$C_{12} \leftarrow \alpha S_4 B_{22} + C_{12}$	C_{12}
4	$C_{22} \leftarrow P_5 + \beta C_{22}$	C_{22}	15	$\mathbf{U_5} \leftarrow U_2 + C_{12}$	C_{12}
5	$C_{12} \leftarrow P_5 + \beta C_{12}$	C_{12}	16	$P_4 \leftarrow \alpha A_{22} T_4 - \beta C_{21}$	C_{21}
6	$S_2 \leftarrow S_1 - A_{11}$	X	17	$S_3 \leftarrow A_{11} - A_{21}$	X
7	$T_2 \leftarrow B_{22} - T_1$	Y	18	$T_3 \leftarrow B_{22} - B_{12}$	Y
8	$P_1 \leftarrow \alpha A_{11} B_{11}$	Z	19	$U_3 \leftarrow \alpha S_3 T_3 + U_2$	Z
9	$C_{11} \leftarrow P_1 + \beta C_{11}$	C_{11}	20	$\mathbf{U_7} \leftarrow U_3 + C_{22}$	C_{22}
10	$U_2 \leftarrow \alpha S_2 T_2 + P_1$	Z	21	$\mathbf{U_6} \leftarrow U_3 - C_{21}$	C_{21}
11	$\mathbf{U_1} \leftarrow \alpha A_{12} B_{21} + C_{11}$	C_{11}			

Product with accumulation

$$C \leftarrow \alpha A \times B + \beta C$$

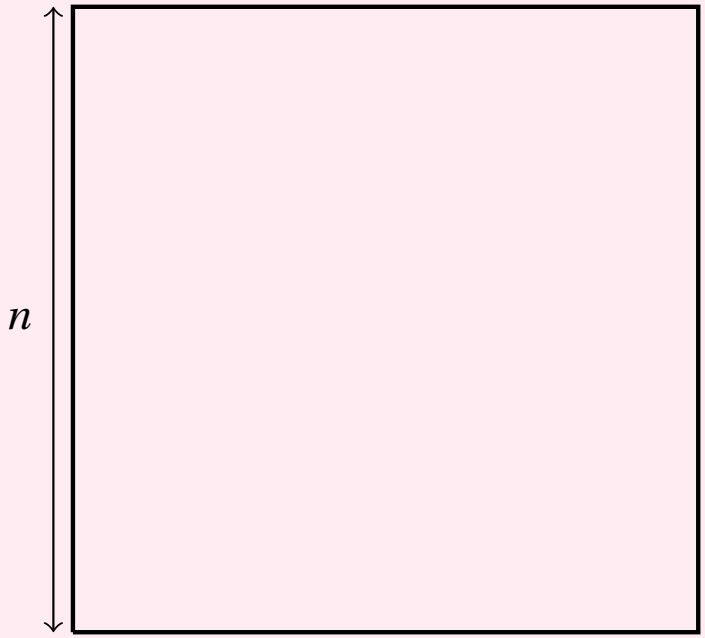
A product with accumulation scheduling (Huss-Lederman et al. '96)

#	operation	*loc.*	#	operation	*loc.*
1	$S_1 \leftarrow A_{21} + A_{22}$	**X**	12	$S_4 \leftarrow A_{12} - S_2$	**X**
2	$T_1 \leftarrow B_{12} - B_{11}$	**Y**	13	$T_4 \leftarrow T_2 - B_{21}$	**Y**
3	$P_5 \leftarrow \alpha S_1 T_1$	**Z**	14	$C_{12} \leftarrow \alpha S_4 B_{22} + C_{12}$	C_{12}
4	$C_{22} \leftarrow P_5 + \beta C_{22}$	C_{22}	15	$\mathbf{U_5} \leftarrow U_2 + C_{12}$	C_{12}
5	$C_{12} \leftarrow P_5 + \beta C_{12}$	C_{12}	16	$P_4 \leftarrow \alpha A_{22} T_4 - \beta C_{21}$	C_{21}
6	$S_2 \leftarrow S_1 - A_{11}$	**X**	17	$S_3 \leftarrow A_{11} - A_{21}$	**X**
7	$T_2 \leftarrow B_{22} - T_1$	**Y**	18	$T_3 \leftarrow B_{22} - B_{12}$	**Y**
8	$P_1 \leftarrow \alpha A_{11} B_{11}$	**Z**	19	$U_3 \leftarrow \alpha S_3 T_3 + U_2$	**Z**
9	$C_{11} \leftarrow P_1 + \beta C_{11}$	C_{11}	20	$\mathbf{U_7} \leftarrow U_3 + C_{22}$	C_{22}
10	$U_2 \leftarrow \alpha S_2 T_2 + P_1$	**Z**	21	$\mathbf{U_6} \leftarrow U_3 - C_{21}$	C_{21}
11	$\mathbf{U_1} \leftarrow \alpha A_{12} B_{21} + C_{11}$	C_{11}			

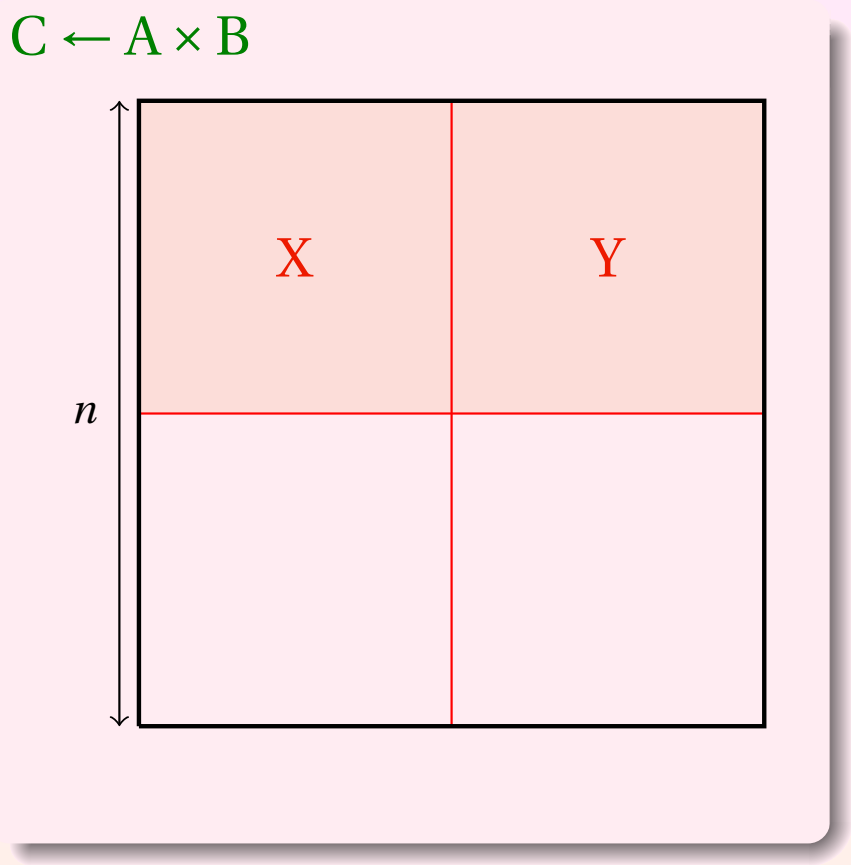
3 temporaries

Extra memory usage: the square case

$C \leftarrow A \times B$



Extra memory usage: the square case



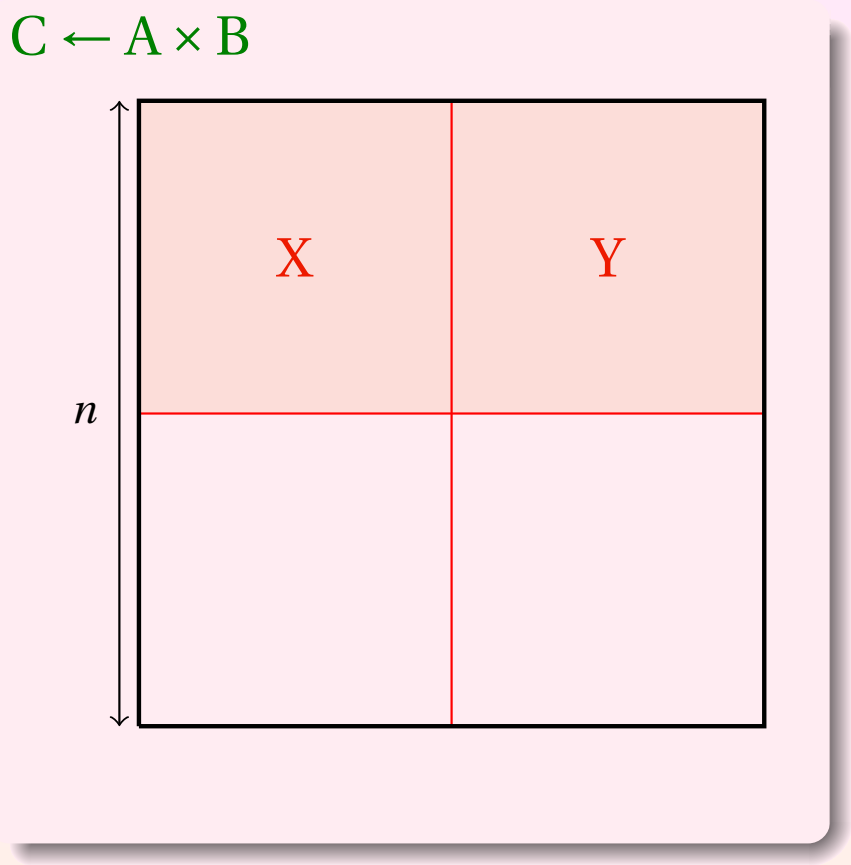
Product $C \leftarrow A \times B$

A Strassen-Winograd scheduling (Douglas et al. '94)

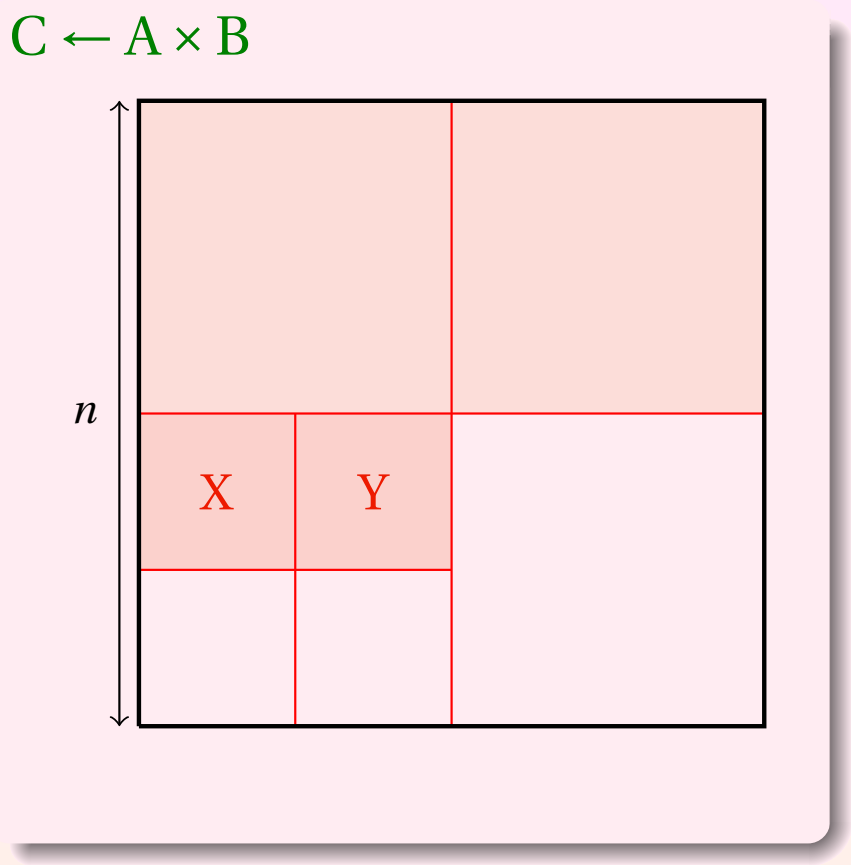
#	operation	*loc.*	#	operation	*loc.*
1	$S_3 \leftarrow A_{11} - A_{21}$	X	12	$P_1 \leftarrow A_{11}B_{11}$	X
2	$T_3 \leftarrow B_{22} - B_{12}$	Y	13	$U_2 \leftarrow P_6 + P_1$	C_{12}
3	$P_7 \leftarrow S_3T_3$	C_{21}	14	$U_3 \leftarrow U_2 + P_7$	C_{21}
4	$S_1 \leftarrow A_{21} + A_{22}$	X	15	$U_4 \leftarrow U_2 + P_5$	C_{12}
5	$T_1 \leftarrow B_{12} - B_{11}$	Y	16	$\mathbf{U_7} \leftarrow U_3 + P_5$	C_{22}
6	$P_5 \leftarrow S_1T_1$	C_{22}	17	$\mathbf{U_5} \leftarrow U_4 + P_3$	C_{12}
7	$S_2 \leftarrow S_1 - A_{11}$	X	18	$T_4 \leftarrow T_2 - B_{21}$	Y
8	$T_2 \leftarrow B_{22} - T_1$	Y	19	$P_4 \leftarrow A_{22}T_4$	C_{11}
9	$P_6 \leftarrow S_2T_2$	C_{12}	20	$\mathbf{U_6} \leftarrow U_3 - P_4$	C_{21}
10	$S_4 \leftarrow A_{12} - S_2$	X	21	$P_2 \leftarrow A_{12}B_{21}$	C_{11}
11	$P_3 \leftarrow S_4B_{22}$	C_{11}	22	$\mathbf{U_1} \leftarrow -P_1 + P_2$	C_{11}

2 temporaries

Extra memory usage: the square case

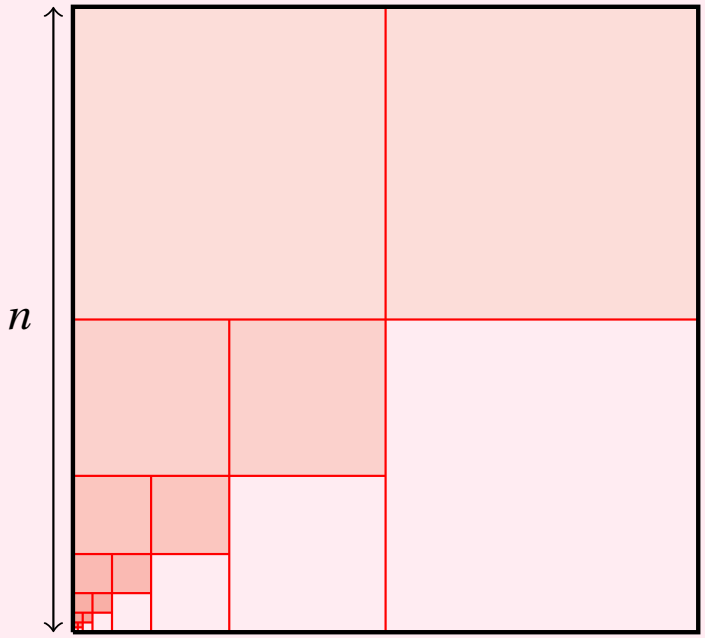


Extra memory usage: the square case



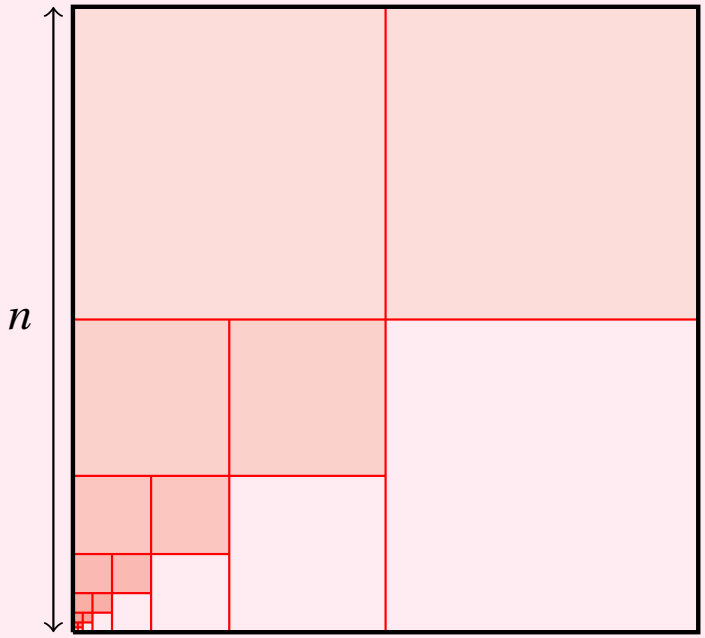
Extra memory usage: the square case

$C \leftarrow A \times B$



Extra memory usage: the square case

$C \leftarrow A \times B$

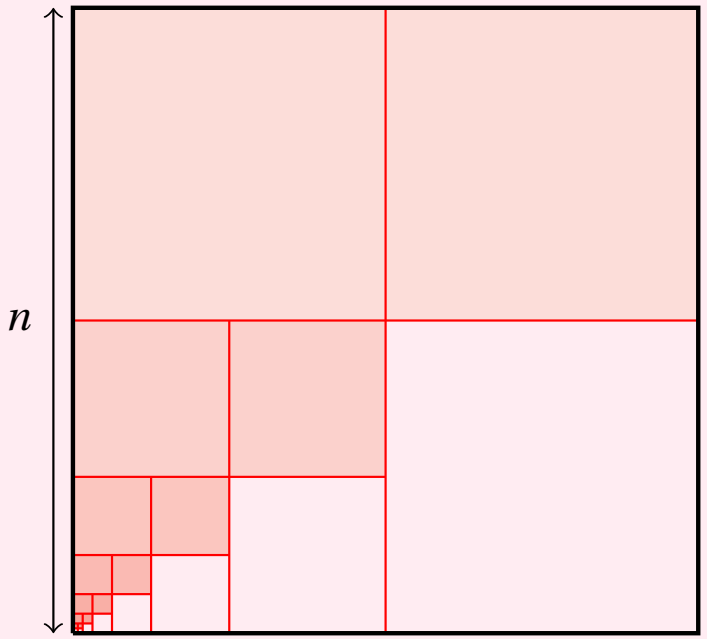


Extra memory needed $< \frac{2}{3} n^2$

(Douglas '94)

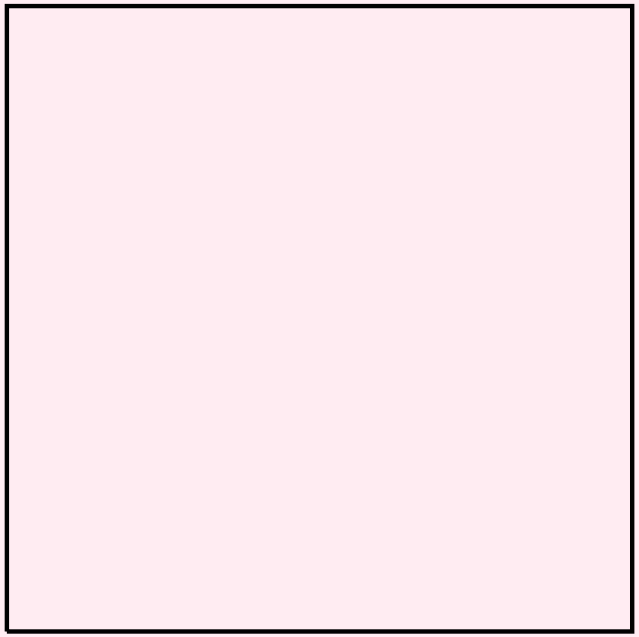
Extra memory usage: the square case

$C \leftarrow A \times B$



Extra memory needed $< \frac{2}{3}n^2$

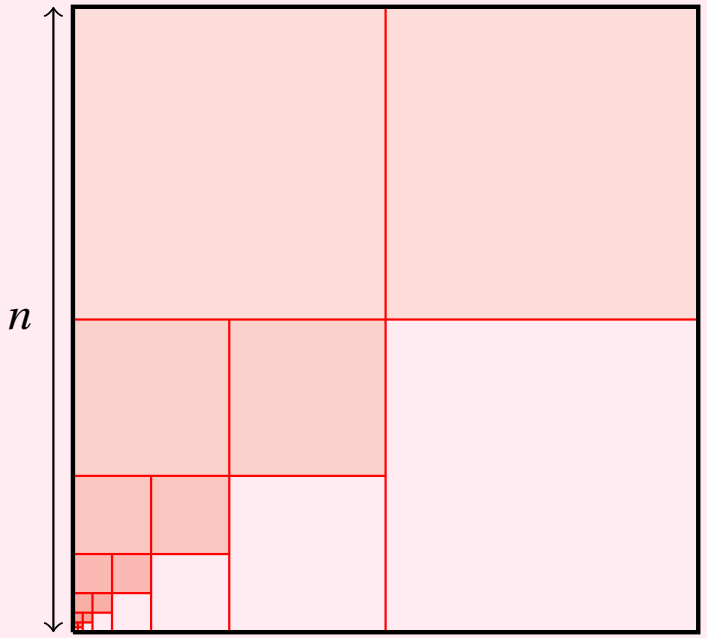
$C \leftarrow A \times B + C$



(Douglas '94)

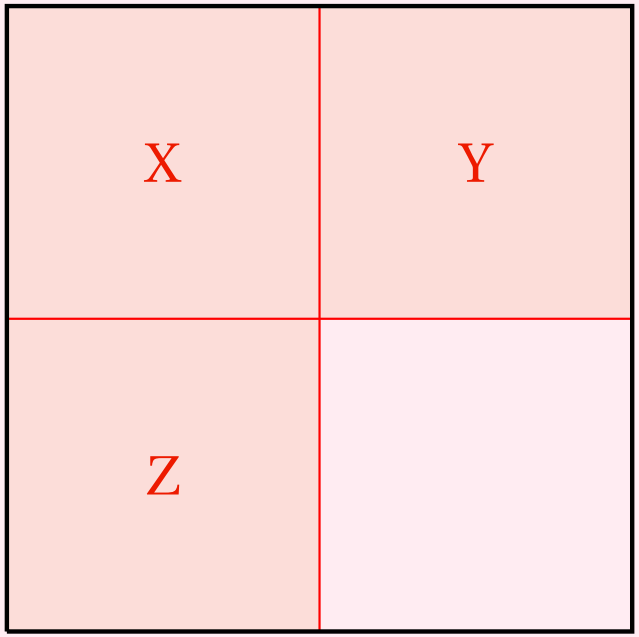
Extra memory usage: the square case

$C \leftarrow A \times B$



Extra memory needed $< \frac{2}{3}n^2$

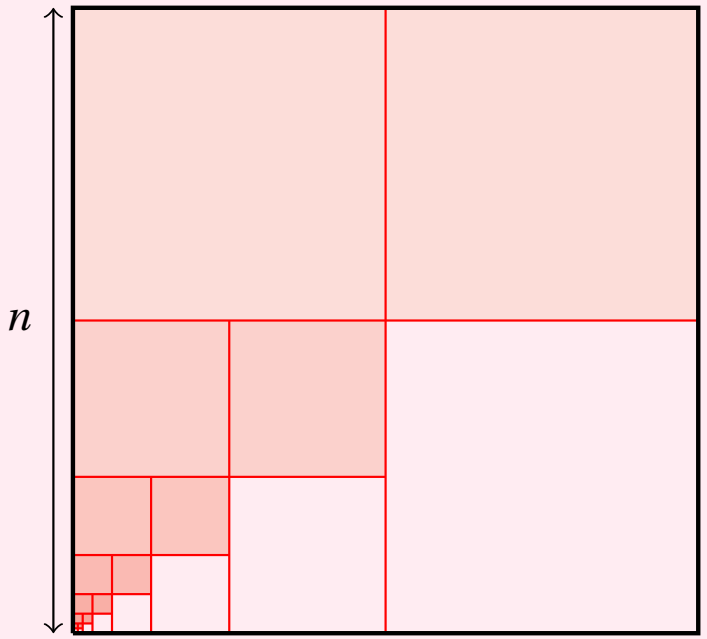
$C \leftarrow A \times B + C$



(Douglas '94)

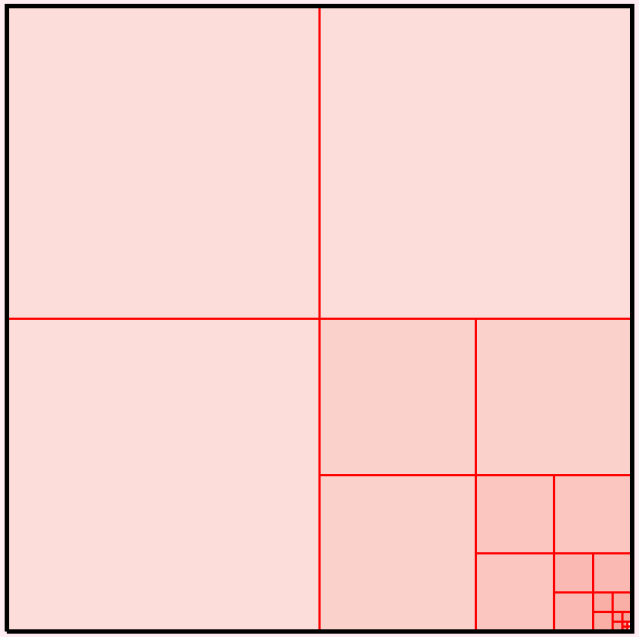
Extra memory usage: the square case

$C \leftarrow A \times B$



Extra memory needed $< \frac{2}{3}n^2$

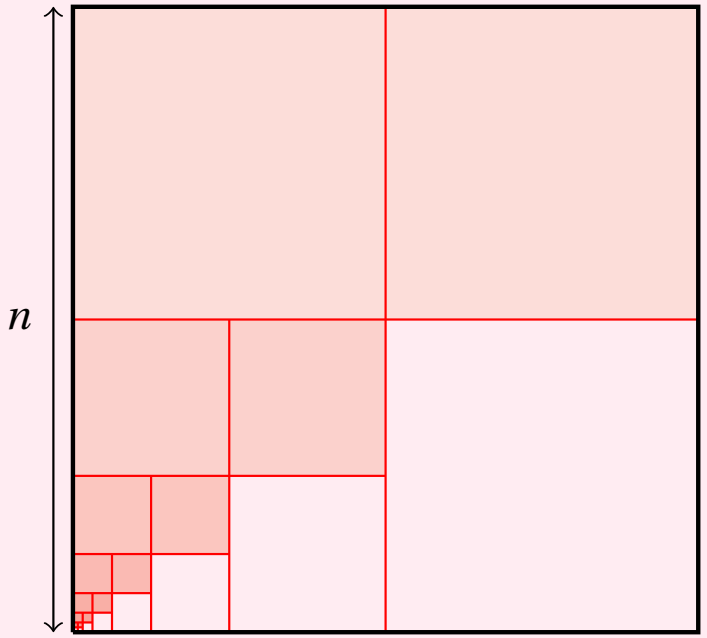
$C \leftarrow A \times B + C$



(Douglas '94)

Extra memory usage: the square case

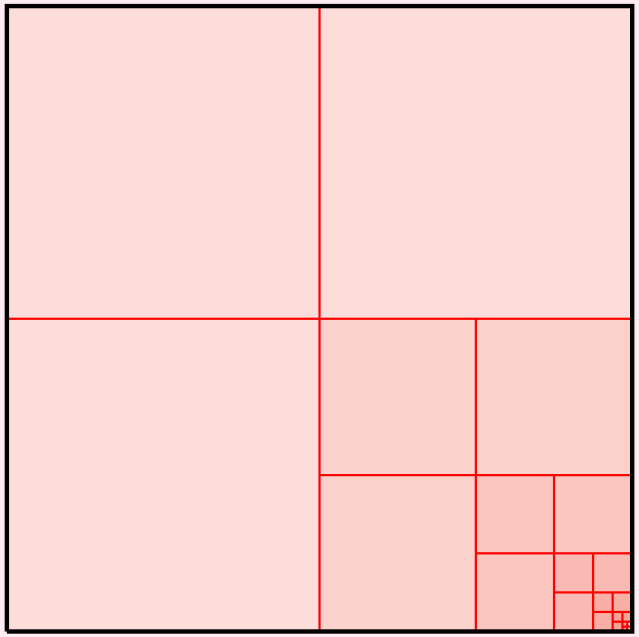
$C \leftarrow A \times B$



Extra memory needed $< \frac{2}{3}n^2$

(Douglas '94)

$C \leftarrow A \times B + C$



Extra memory needed $< n^2$

(Huss-Lederman '96)

Our contribution

Extra memory requirements

$C \leftarrow A \times B$	$C \leftarrow A \times B + C$
$2/3n^2$	n^2
(Douglas)	(Huss-Lederman)

Our contribution

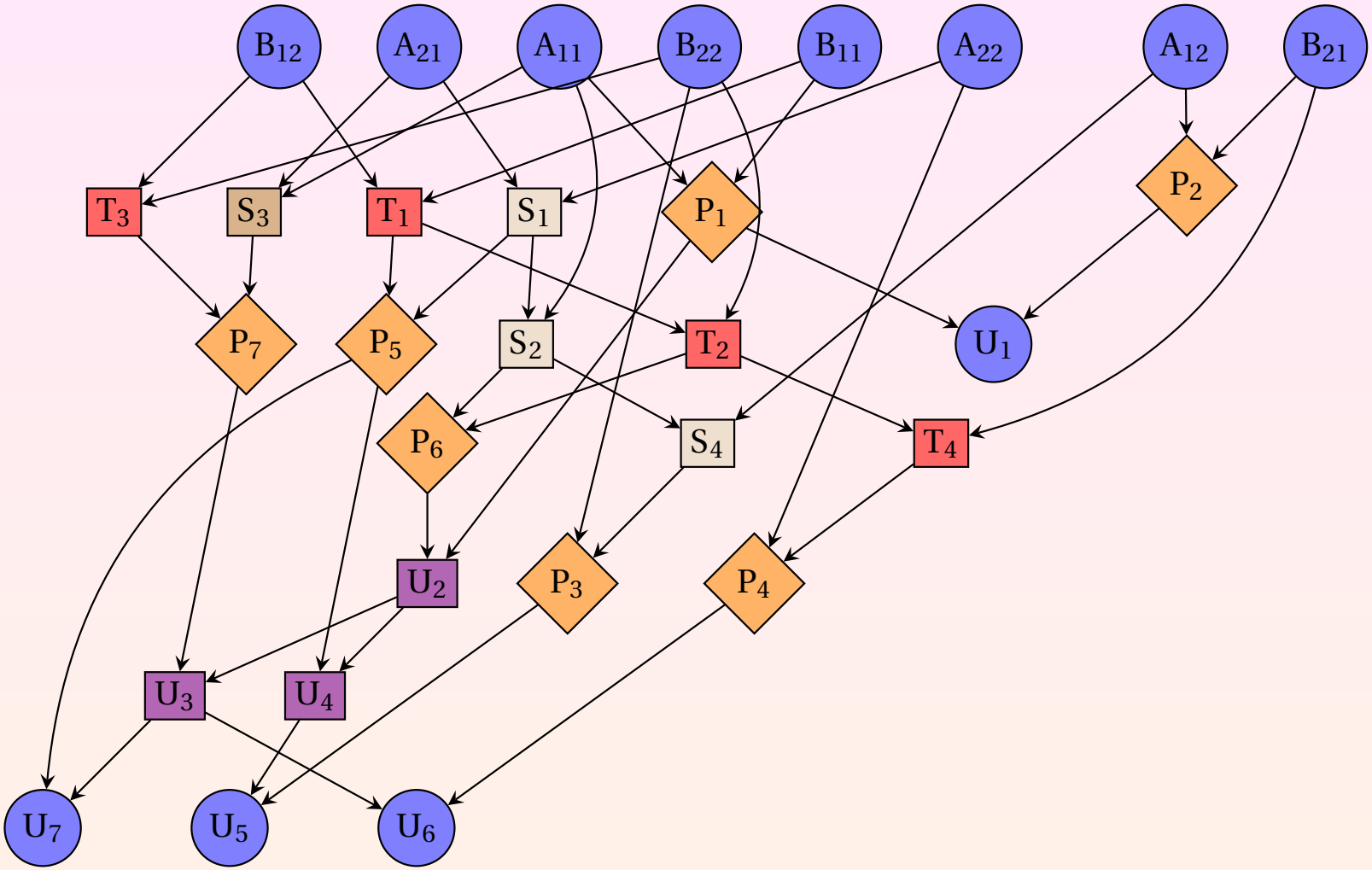
Extra memory requirements

$C \leftarrow A \times B$	$C \leftarrow A \times B + C$
$2/3n^2$	n^2
(Douglas)	(Huss-Lederman)
0	$2/3n^2$
(if overwritable inputs)	(with square inputs)
0	$\rightarrow 0$
(and slightly more operations)	(with more operations)

Outline

1. Previous work about keeping memory requirements low
   - Existing Schedulings
   - Extra memory usage
   - Our Contribution
2. A dynamic program generating schedules : a pebble game.
   - The dependency graph
3. Building New Algorithms from the schedules
   - Building Blocks
   - A fully in place algorithm with constant inputs
4. Conclusion.

Dependency graph for Strassen-Winograd Alogrithm

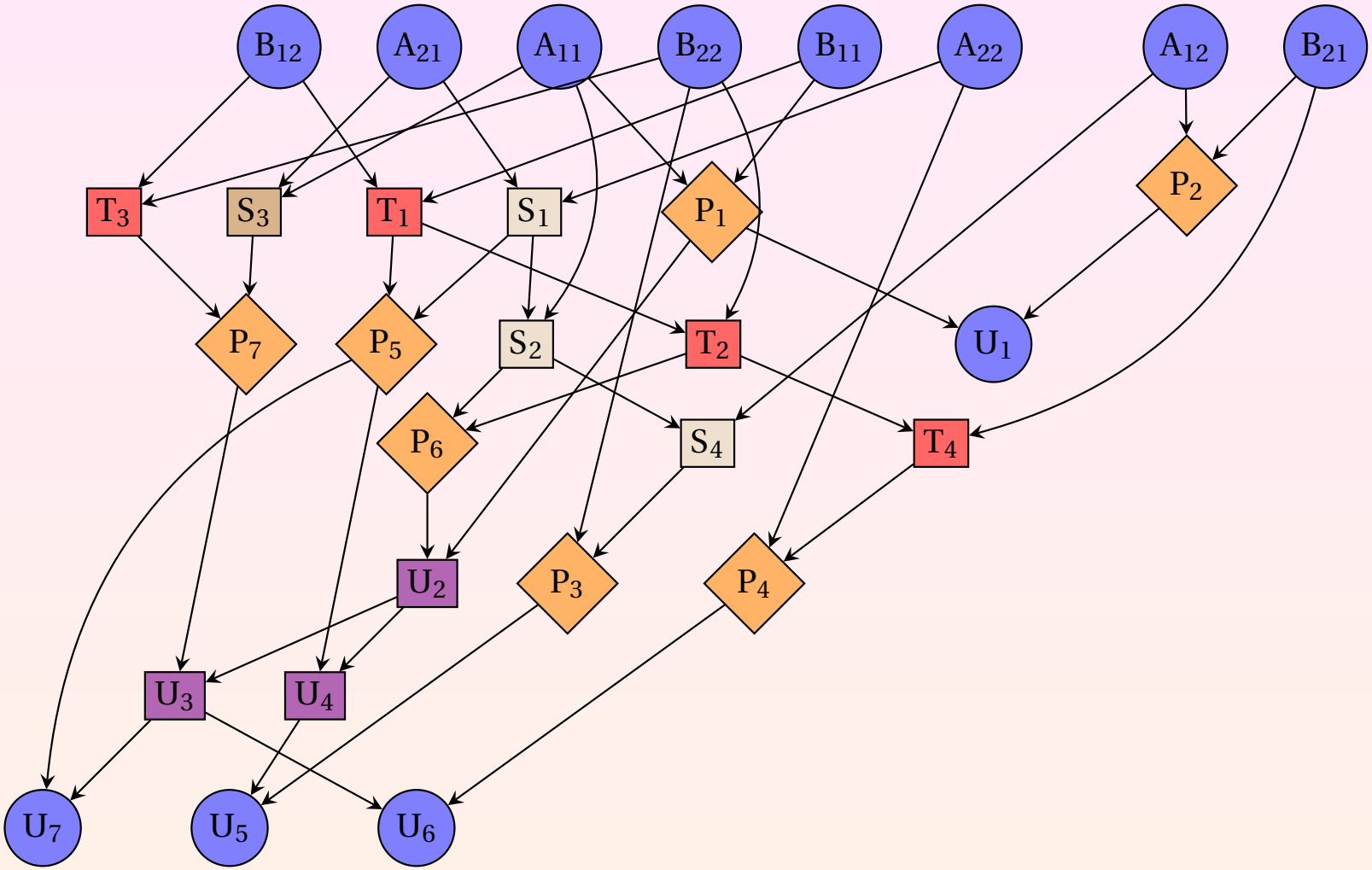


Ideas

What we do

- We consider one dependency graph.
- Every initial node gets a pebble (representing memory space).
- With a set of rules and possibly more pebbles (i.e. extra memory), we try to get 4 of them down to the final nodes.

Dependency graph for Strassen-Winograd Alogrithm



Ideas

What we do

- We consider one dependency graph.
- Every initial node gets a pebble (representing memory space).
- With a set of rules and possibly more pebbles (i.e. extra memory), we try to get 4 of them down to the final nodes.

What we get

- Either new schedules if they exist;
- Or a certification no schedule exists with the prescribed conditions.

Outline

1. Previous work about keeping memory requirements low
   - Existing Schedulings
   - Extra memory usage
   - Our Contribution
2. A dynamic program generating schedules : a pebble game.
   - The dependency graph
3. Building New Algorithms from the schedules
   - Building Blocks
   - A fully in place algorithm with constant inputs
4. Conclusion.

Techniques used

- Overwriting input matrices.
- Overwriting temporary matrices.

Techniques used

- Overwriting input matrices.
- Overwriting temporary matrices.
- Introducing pre-additions (variants of Strassen-Winograd dependency graph/algorithm).

Techniques used

- Overwriting input matrices.
- Overwriting temporary matrices.
- Introducing pre-additions (variants of Strassen-Winograd dependency graph/algorithm).
- Using a hybrid scheduling (classic + Strassen).

Example 1

Acc schedule $C \leftarrow \alpha A \times B + \beta C$

#	operation	loc.	#	operation	loc.
1	$Z_1 = C_{22} - C_{12}$	C_{22}	14	$P_2 = \mathtt{Acc}(\alpha A_{12}B_{21} + \beta C_{11})$	C_{11}
2	$Z_3 = C_{12} - C_{21}$	C_{12}	15	$\mathbf{U_1} = P_1 + P_2$	C_{11}
3	$S_1 = A_{21} + A_{22}$	X	16	$\mathbf{U_5} = U_2 + P_3$	C_{12}
4	$T_1 = B_{12} - B_{11}$	Y	17	$S_3 = A_{11} - A_{21}$	X
5	$P_5 = \mathtt{Acc}(\alpha S_1 T_1 + \beta Z_3)$	C_{12}	18	$T_3 = B_{22} - B_{12}$	Y
6	$S_2 = S_1 - A_{11}$	X	19	$U_3 = P_7 + U_2$	C_{21}
7	$T_2 = B_{22} - T_1$	Y		$= \alpha\mathtt{AcLR}(S_3T_3 + U_2)$	
8	$P_6 = \mathtt{Acc}(\alpha S_2 T_2 + \beta C_{21})$	C_{21}	20	$\mathbf{U_7} = U_3 + W_1$	C_{22}
9	$S_4 = A_{12} - S_2$	X	21	$T'_1 = B_{12} - B_{11}$	Y
10	$W_1 = P_5 + \beta Z_1$	C_{22}	22	$T'_2 = B_{22} - T'_1$	Y
11	$P_3 = \mathtt{Acc}(\alpha S_4 B_{22} + P_5)$	C_{12}	23	$T_4 = T'_2 - B_{21}$	Y
12	$P_1 = \alpha A_{11} B_{11}$	X	24	$\mathbf{U_6} = U_3 - P_4$	C_{21}
13	$U_2 = P_6 + P_1$	C_{21}		$= -\alpha\mathtt{AccR}(A_{22}T_4 - U_3)$	

Example 1

Acc schedule $C \leftarrow \alpha A \times B + \beta C$

#	operation	loc.	#	operation	loc.
1	$\mathbf{Z_1} = C_{22} - C_{12}$	C_{22}	14	$P_2 = \mathtt{Acc}(\alpha A_{12}B_{21} + \beta C_{11})$	C_{11}
2	$\mathbf{Z_3} = C_{12} - C_{21}$	C_{12}	15	$\mathbf{U_1} = P_1 + P_2$	C_{11}
3	$S_1 = A_{21} + A_{22}$	X	16	$\mathbf{U_5} = U_2 + P_3$	C_{12}
4	$T_1 = B_{12} - B_{11}$	Y	17	$S_3 = A_{11} - A_{21}$	X
5	$P_5 = \mathtt{Acc}(\alpha S_1 T_1 + \beta Z_3)$	C_{12}	18	$T_3 = B_{22} - B_{12}$	Y
6	$S_2 = S_1 - A_{11}$	X	19	$U_3 = P_7 + U_2$	C_{21}
7	$T_2 = B_{22} - T_1$	Y		$= \alpha\mathtt{AcLR}(S_3T_3 + U_2)$	
8	$P_6 = \mathtt{Acc}(\alpha S_2 T_2 + \beta C_{21})$	C_{21}	20	$\mathbf{U_7} = U_3 + W_1$	C_{22}
9	$S_4 = A_{12} - S_2$	X	21	$T'_1 = B_{12} - B_{11}$	Y
10	$W_1 = P_5 + \beta Z_1$	C_{22}	22	$T'_2 = B_{22} - T'_1$	Y
11	$P_3 = \mathtt{Acc}(\alpha S_4 B_{22} + P_5)$	C_{12}	23	$T_4 = T'_2 - B_{21}$	Y
12	$P_1 = \alpha A_{11} B_{11}$	X	24	$\mathbf{U_6} = U_3 - P_4$	C_{21}
13	$U_2 = P_6 + P_1$	C_{21}		$= -\alpha\mathtt{AccR}(A_{22}T_4 - U_3)$	

2 pre-additions

Example 1

Acc schedule $C \leftarrow \alpha A \times B + \beta C$

#	operation	loc.	#	operation	loc.
1	$Z_1 = C_{22} - C_{12}$	C_{22}	14	$P_2 = \mathtt{Acc}(\alpha A_{12}B_{21} + \beta C_{11})$	C_{11}
2	$Z_3 = C_{12} - C_{21}$	C_{12}	15	$\mathbf{U_1} = P_1 + P_2$	C_{11}
3	$S_1 = A_{21} + A_{22}$	X	16	$\mathbf{U_5} = U_2 + P_3$	C_{12}
4	$T_1 = B_{12} - B_{11}$	Y	17	$S_3 = A_{11} - A_{21}$	X
5	$P_5 = \mathtt{Acc}(\alpha S_1 T_1 + \beta Z_3)$	C_{12}	18	$T_3 = B_{22} - B_{12}$	Y
6	$S_2 = S_1 - A_{11}$	X	19	$U_3 = P_7 + U_2$	C_{21}
7	$T_2 = B_{22} - T_1$	Y		$= \alpha\mathbf{AcLR}(S_3T_3 + U_2)$	
8	$P_6 = \mathtt{Acc}(\alpha S_2 T_2 + \beta C_{21})$	C_{21}	20	$\mathbf{U_7} = U_3 + W_1$	C_{22}
9	$S_4 = A_{12} - S_2$	X	21	$T'_1 = B_{12} - B_{11}$	Y
10	$W_1 = P_5 + \beta Z_1$	C_{22}	22	$T'_2 = B_{22} - T'_1$	Y
11	$P_3 = \mathtt{Acc}(\alpha S_4 B_{22} + P_5)$	C_{12}	23	$T_4 = T'_2 - B_{21}$	Y
12	$P_1 = \alpha A_{11} B_{11}$	X	24	$\mathbf{U_6} = U_3 - P_4$	C_{21}
13	$U_2 = P_6 + P_1$	C_{21}		$= -\alpha\mathbf{AccR}(A_{22}T_4 - U_3)$	

Overwriting temporary matrices

Example 1

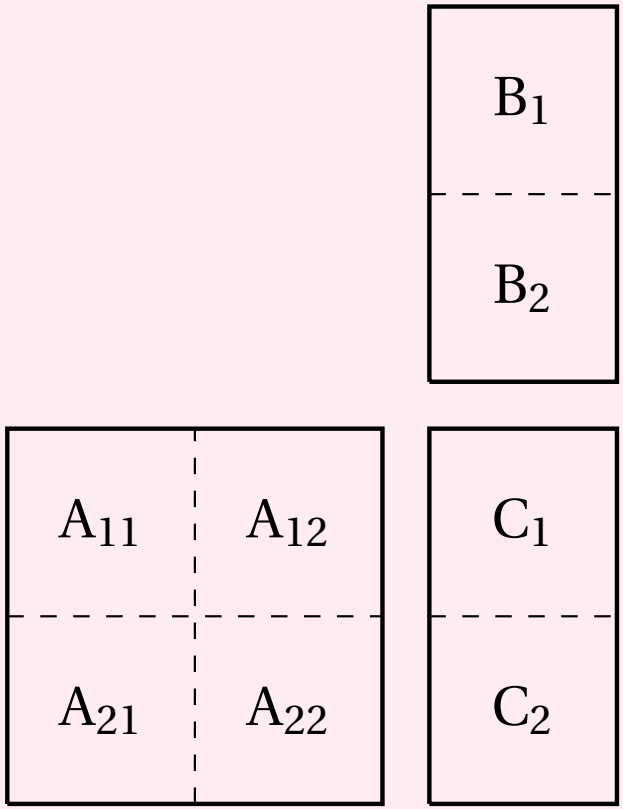
Acc schedule $C \leftarrow \alpha A \times B + \beta C$

#	operation	loc.	#	operation	loc.
1	$Z_1 = C_{22} - C_{12}$	C_{22}	14	$P_2 = \mathtt{Acc}(\alpha A_{12}B_{21} + \beta C_{11})$	C_{11}
2	$Z_3 = C_{12} - C_{21}$	C_{12}	15	$\mathbf{U_1} = P_1 + P_2$	C_{11}
3	$S_1 = A_{21} + A_{22}$	X	16	$\mathbf{U_5} = U_2 + P_3$	C_{12}
4	$T_1 = B_{12} - B_{11}$	Y	17	$S_3 = A_{11} - A_{21}$	X
5	$P_5 = \mathtt{Acc}(\alpha S_1 T_1 + \beta Z_3)$	C_{12}	18	$T_3 = B_{22} - B_{12}$	Y
6	$S_2 = S_1 - A_{11}$	X	19	$U_3 = P_7 + U_2$	C_{21}
7	$T_2 = B_{22} - T_1$	Y		$= \alpha\mathtt{AcLR}(S_3T_3 + U_2)$	
8	$P_6 = \mathtt{Acc}(\alpha S_2 T_2 + \beta C_{21})$	C_{21}	20	$\mathbf{U_7} = U_3 + W_1$	C_{22}
9	$S_4 = A_{12} - S_2$	X	21	$T'_1 = B_{12} - B_{11}$	Y
10	$W_1 = P_5 + \beta Z_1$	C_{22}	22	$T'_2 = B_{22} - T'_1$	Y
11	$P_3 = \mathtt{Acc}(\alpha S_4 B_{22} + P_5)$	C_{12}	23	$T_4 = T'_2 - B_{21}$	Y
12	$P_1 = \alpha A_{11} B_{11}$	X	24	$\mathbf{U_6} = U_3 - P_4$	C_{21}
13	$U_2 = P_6 + P_1$	C_{21}		$= -\alpha\mathtt{AccR}(A_{22}T_4 - U_3)$	

Only 2 temporaries

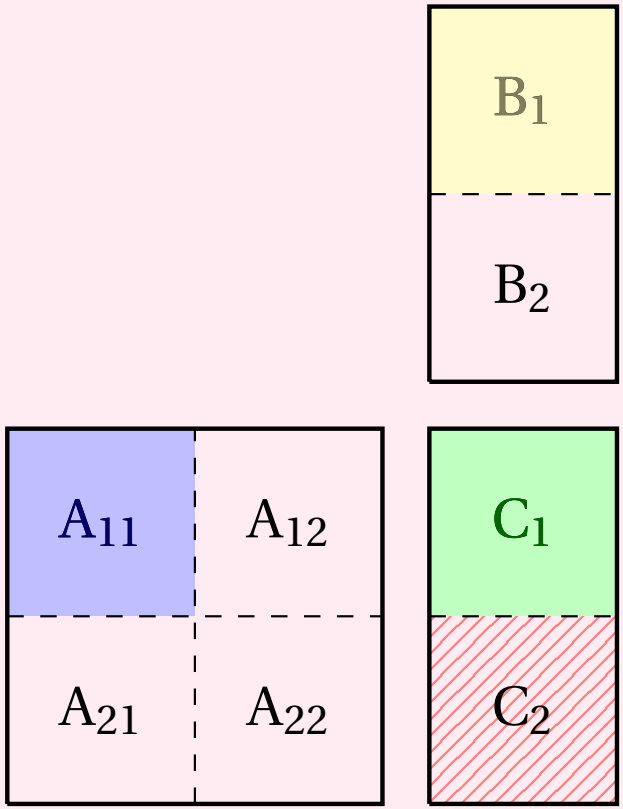
Example 2

In place matrix multiply overwriting inputs



Example 2

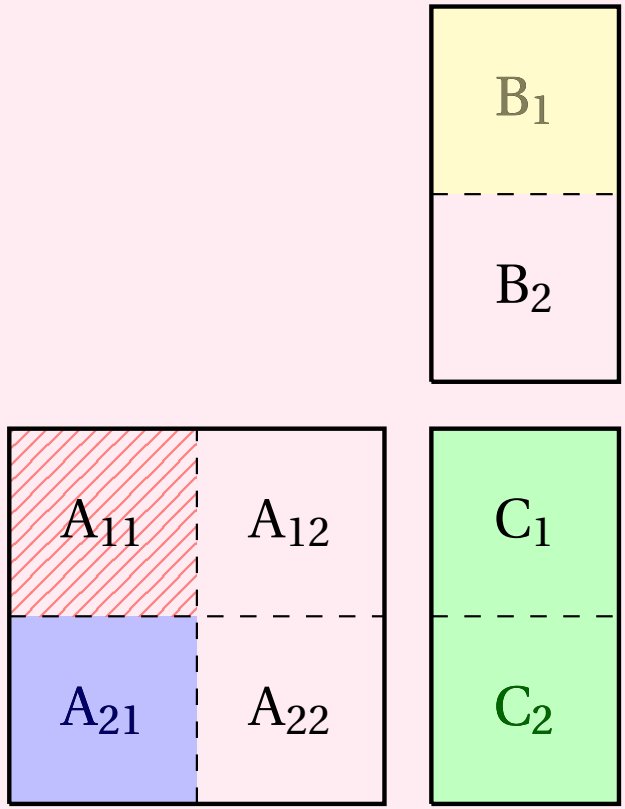
In place matrix multiply overwriting inputs



Strassen-Winograd with C_2 as temp. space

Example 2

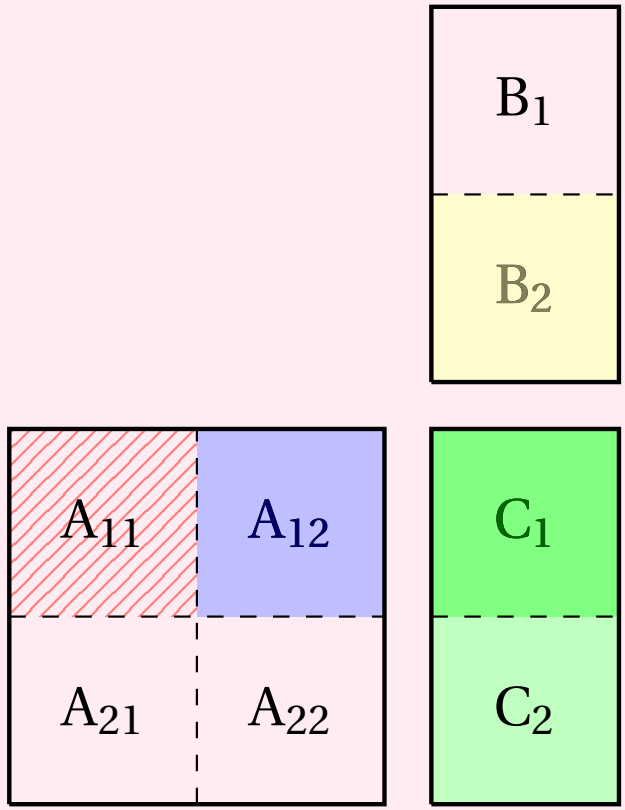
In place matrix multiply overwriting inputs



Strassen-Winograd with A_{11} as temp. space

Example 2

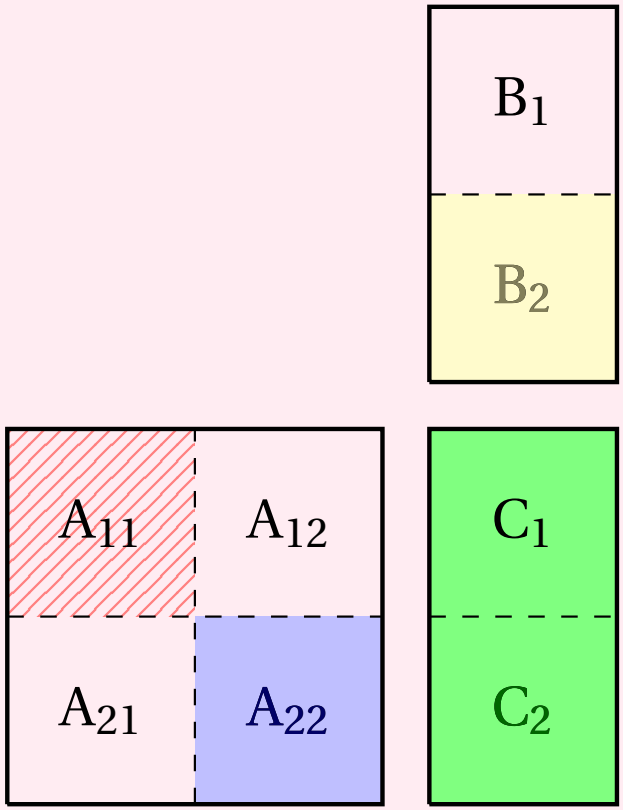
In place matrix multiply overwriting inputs



Product with accumulation with A_{11} as temp. space

Example 2

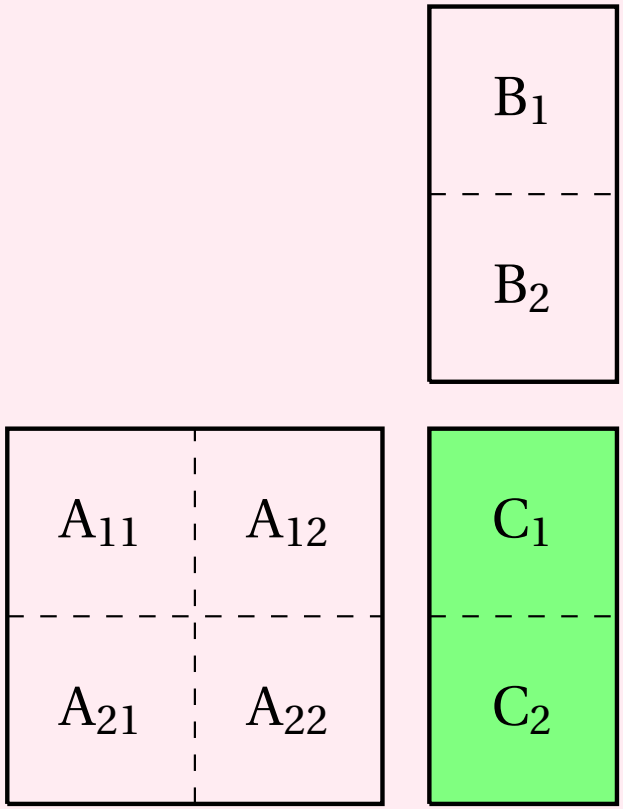
In place matrix multiply overwriting inputs



Product with accumulation with A_{11} as temp. space

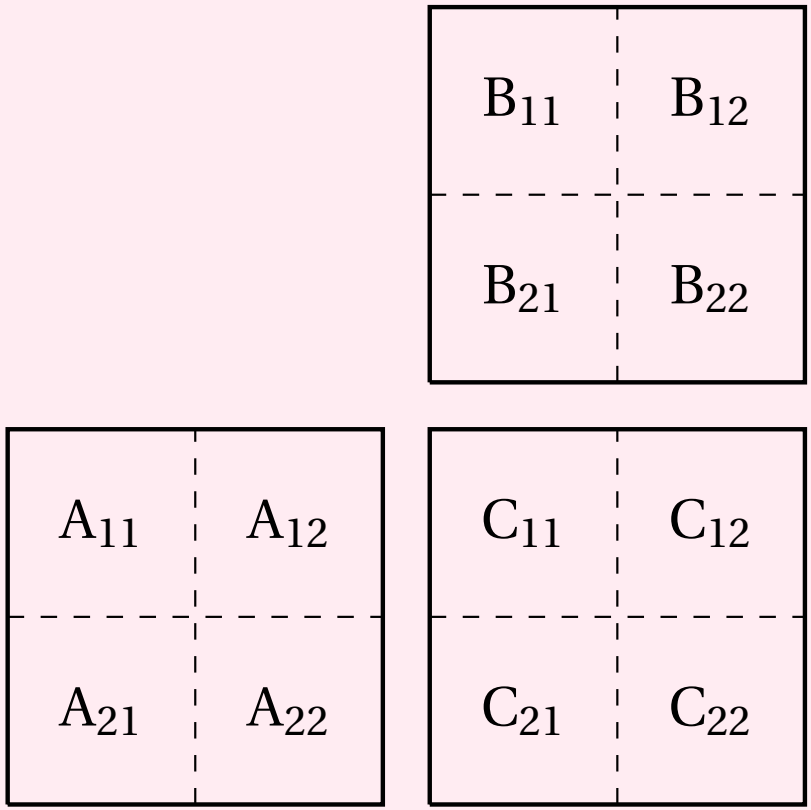
Example 2

In place matrix multiply overwriting inputs



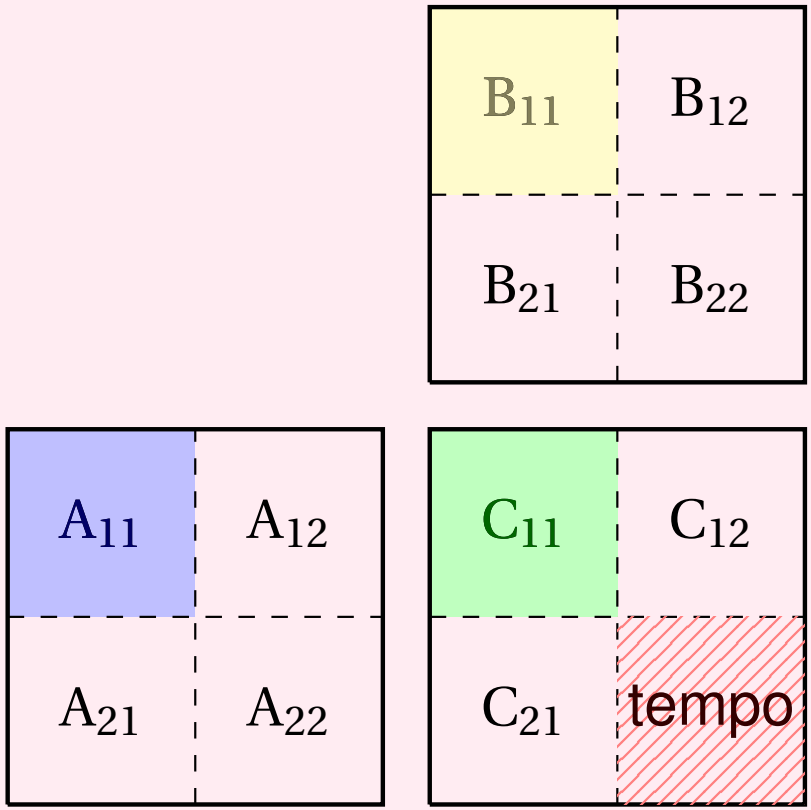
Principle

In place Matrix Multiply



Principle

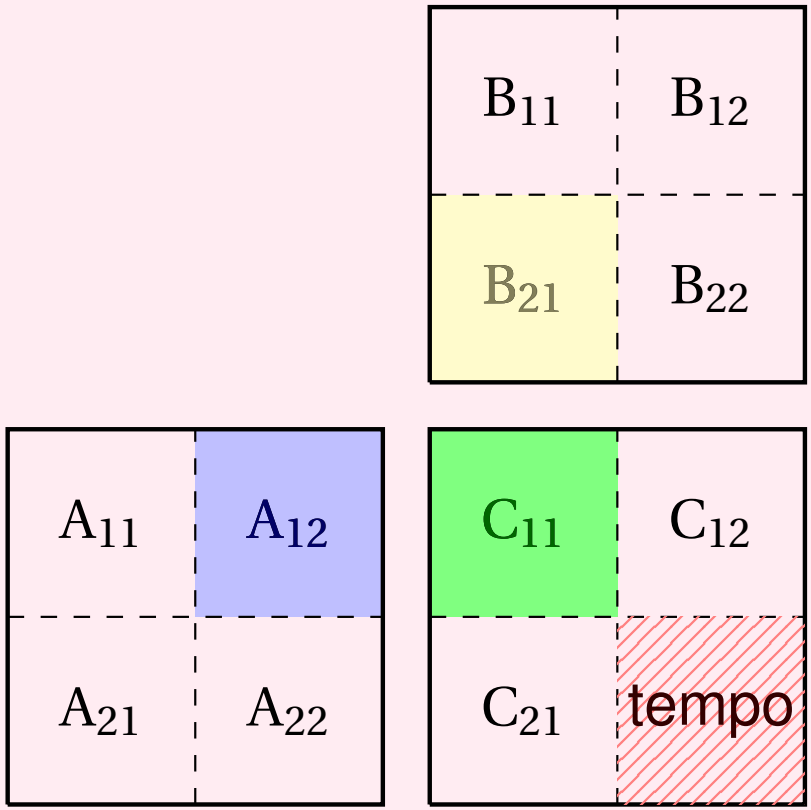
In place Matrix Multiply



Strassen algorithm with C_{22} as temp. space.

Principle

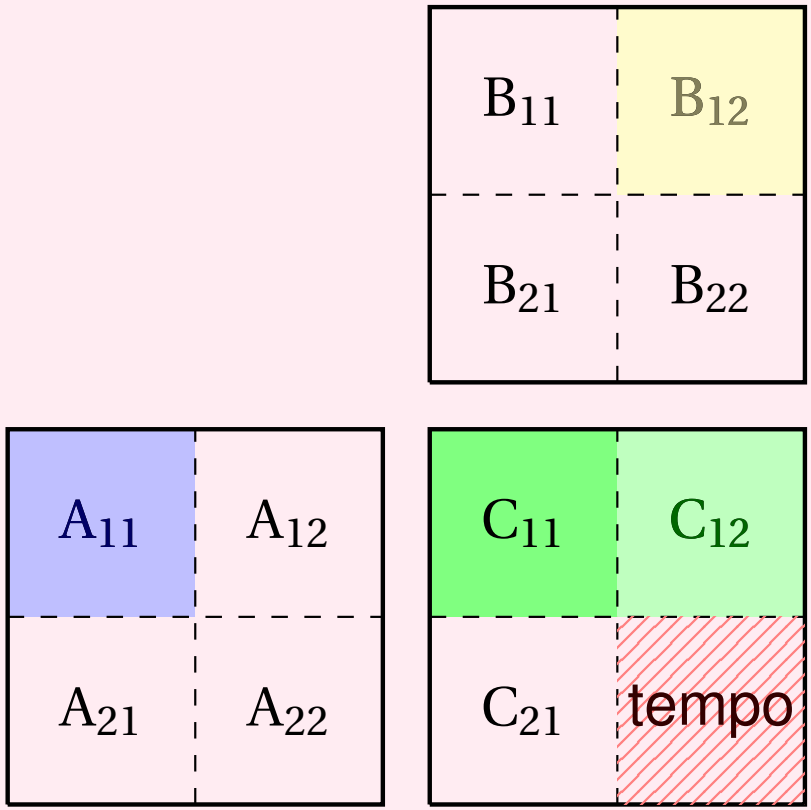
In place Matrix Multiply



Accumulation algorithm with C_{22} as temp. space.

Principle

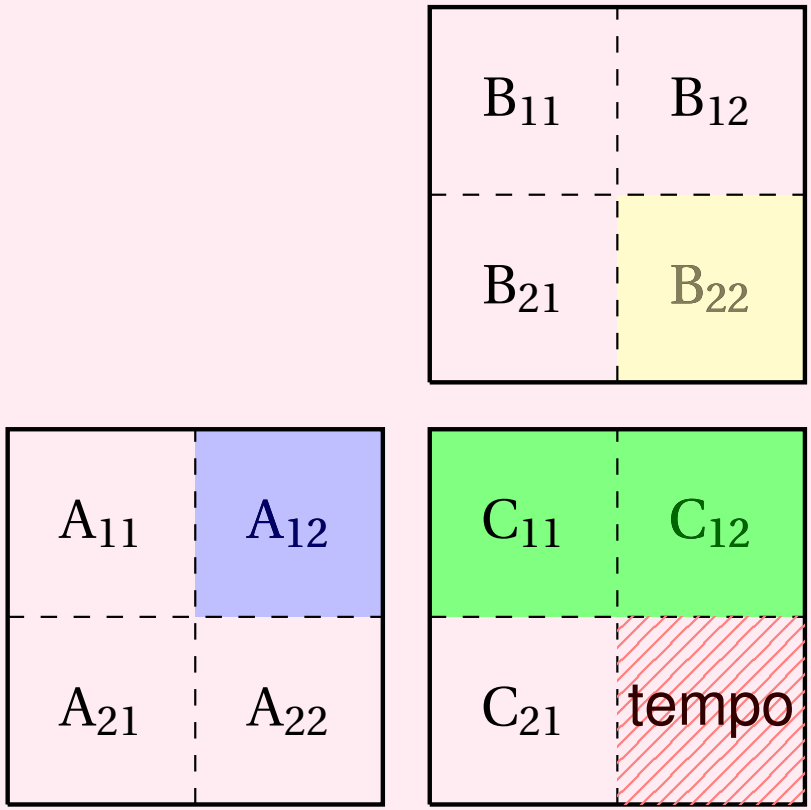
In place Matrix Multiply



Strassen algorithm with C_{22} as temp. space.

Principle

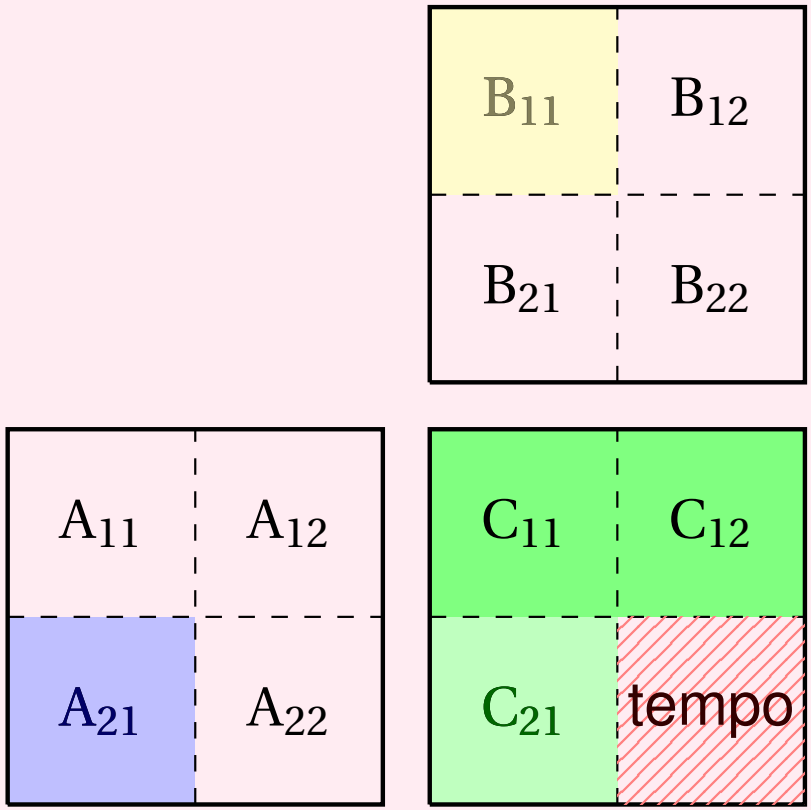
In place Matrix Multiply



Accumulation algorithm with C_{22} as temp. space.

Principle

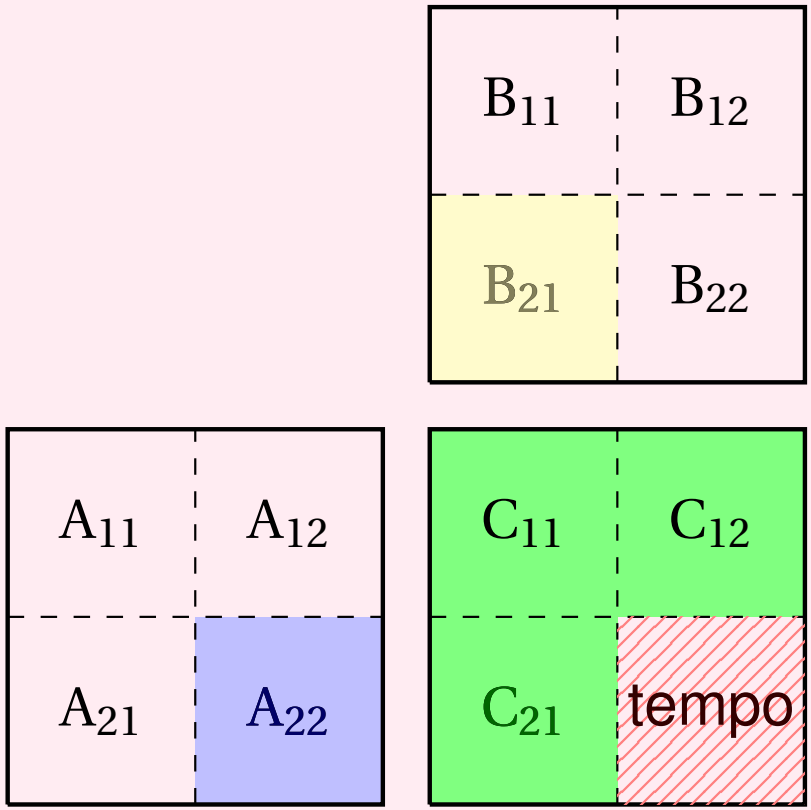
In place Matrix Multiply



Strassen algorithm with C_{22} as temp. space.

Principle

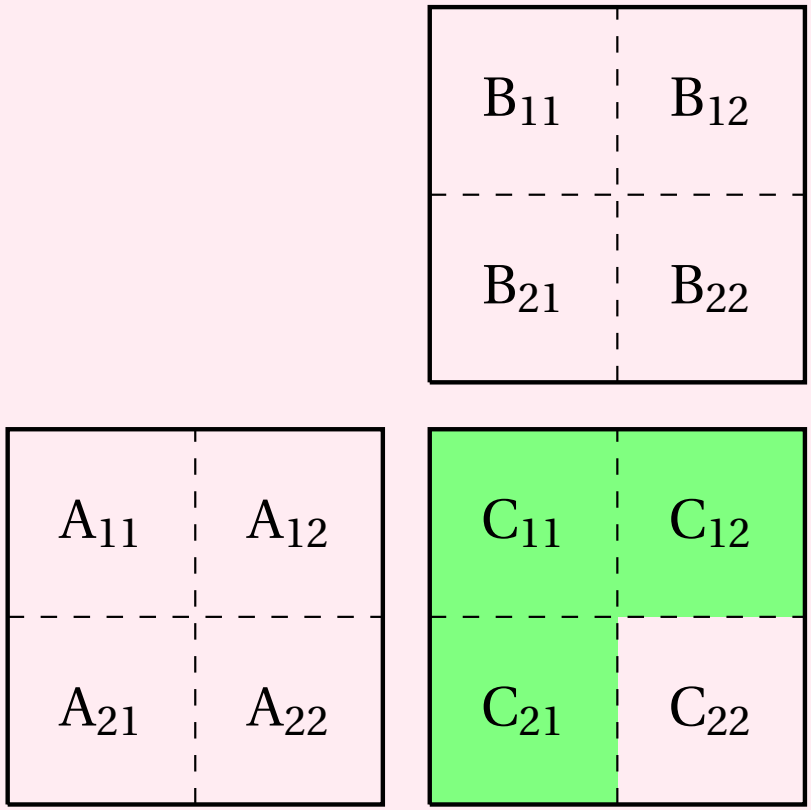
In place Matrix Multiply



Accumulation algorithm with C_{22} as temp. space.

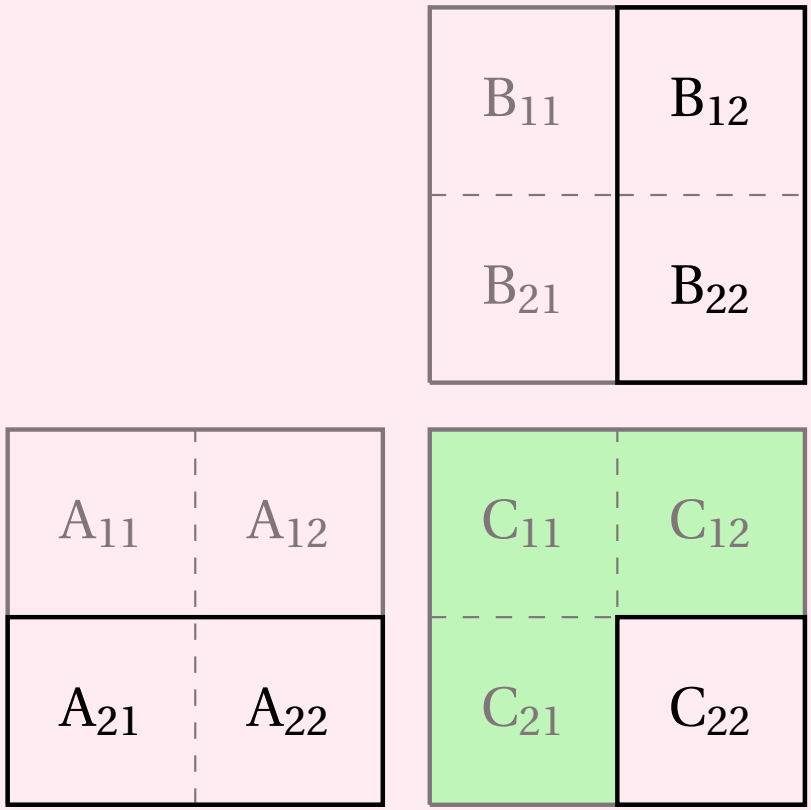
Principle

In place Matrix Multiply



Principle

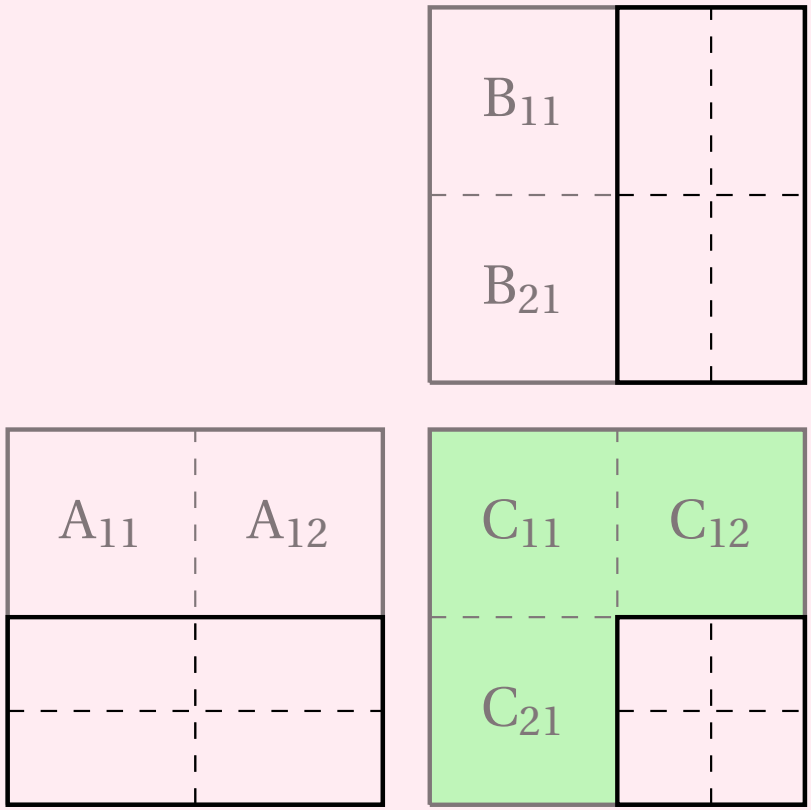
In place Matrix Multiply



C_{22} computed recursively.

Principle

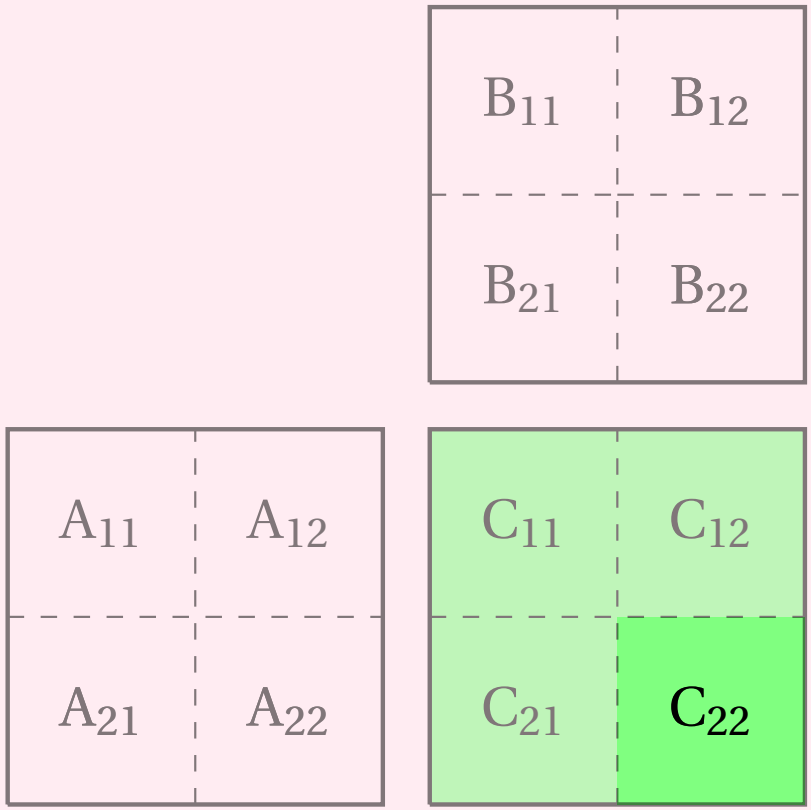
In place Matrix Multiply



C_{22} computed recursively.

Principle

In place Matrix Multiply



Complexity

Propriety

This is a sub-cubic in place algorithm with complexity $\mathcal{O}(n^{\omega})$.

First mesurements

Multiplication of rectangular matrices with dimensions $m \times k$ and $k \times n$: computation time in seconds on a core2 duo, 3.0GHz, $2\times$2Gb RAM

Dims. (m,k,n)	Classic	Douglas	IPMM	IP0vMM
(4096,4096,4096)	14.03	**11.93**	13.59	11.98
(4096,8192,4096)	28.29	**23.39**	27.16	23.88
(8192,8192,8192)	113.07	85.97	98.75	**85.02**
(8192,16384,8192)	231.86	MT	197.24	**170.72**

Outline

1. Previous work about keeping memory requirements low
   - Existing Schedulings
   - Extra memory usage
   - Our Contribution
2. A dynamic program generating schedules : a pebble game.
   - The dependency graph
3. Building New Algorithms from the schedules
   - Building Blocks
   - A fully in place algorithm with constant inputs
4. Conclusion.

Complexities of various schedules

	Algorithm	Input overwritten?	# of extra temp	total extra mem	arithmetic complexity
$A \times B$	Douglas ('94)		2	$\frac{2}{3}n^2$	$6n^{2.807} - 5n^2$
	`IP`	L,R	0	0	$6n^{2.807} - 5n^2$
	`OvR` or `OvL`	L/R	1	$\frac{1}{3}n^2$	$6n^{2.807} - 5n^2$
	`IPMM`		0	0	$7.2n^{2.807} - 13n^2$
$\alpha A \times B + \beta C$	Huss-L ('96)		3	n^2	$6n^{2.807} - 4n^2$
	`AcLR`	L,R	2	$\frac{2}{3}n^2$	$6n^{2.807} - 4n^2 + \frac{1}{2}n^2 \log_2(n)$
	`AccR`	R	2	$\frac{2}{3}n^2$	$6n^{2.807} - 4n^2 + \frac{1}{2}n^2 \log_2(n)$
	`Acc`		2	$\frac{2}{3}n^2$	$6n^{2.807} - 4n^2 + \frac{4}{3}n^2 \log_2(n)$
	Reduced Acc		N/A	$\frac{1}{4}n^2$	$6.857n^{2.807} - 8n^2$
	Reduced Acc		N/A	$\frac{1}{9}n^2$	$7.414n^{2.807} - 12n^2$

Complexities of various schedules

	Algorithm	Input overwritten?	# of extra temp	total extra mem	arithmetic complexity
$A \times B$	Douglas ('94)		2	$\frac{2}{3}n^2$	$6n^{2.807} - 5n^2$
	`IP`	L,R	0	0	$6n^{2.807} - 5n^2$
	`OvR` or `OvL`	L/R	1	$\frac{1}{3}n^2$	$6n^{2.807} - 5n^2$
	`IPMM`		0	0	$\mathbf{7.2}n^{2.807} - 13n^2$
$\alpha A \times B + \beta C$	Huss-L ('96)		3	n^2	$6n^{2.807} - 4n^2$
	`AcLR`	L,R	2	$\frac{2}{3}n^2$	$6n^{2.807} - 4n^2 + \frac{1}{2}n^2 \log_2(n)$
	`AccR`	R	2	$\frac{2}{3}n^2$	$6n^{2.807} - 4n^2 + \frac{1}{2}n^2 \log_2(n)$
	`Acc`		2	$\frac{2}{3}n^2$	$6n^{2.807} - 4n^2 + \mathbf{\frac{4}{3}n^2 \log_2(n)}$
	Reduced Acc		N/A	$\frac{1}{4}n^2$	$6.857n^{2.807} - 8n^2$
	Reduced Acc		N/A	$\frac{1}{9}n^2$	$7.414n^{2.807} - 12n^2$

Thank you!

Memory Efficient Scheduling of Strassen-Winograd's Matrix Multiplication Algorithm

Brice Boyer[1] Jean-Guillaume Dumas[1]
Clément Pernet[2] Wei Zhou[3]

[1]LJK, Université de Grenoble, France
[2]INRIA-MOAIS, Université de Grenoble, France
[3]University of Waterloo, Canada

ISSAC '09

July 30, 2009