SuperMatrix: A Multithreaded Runtime Scheduling System for Algorithms-by-Blocks

Ernie Chan, Field G. Van Zee, Robert van de Geijn, Paolo Bientinesi, Enrique S. Quintana-Ortí and Gregorio Quintana-Ortí

Software Engineering Seminar – Luc Humair

Motivation

- Multicore architectures demand concurrent algorithms
- Complicated and error prone linear algebra libraries

Motivation

- SuperMatrix offers level of abstraction for algorithms-by-block:
  - Automatic parallelization
  - Straight forward implementation of algorithms-by-block

Motivation

- SuperMatrix offers level of abstraction for algorithms-by-block:
  - Automatic parallelization
  - Straight forward implementation of algorithms-by-block
- Work with blocked matrices (FLAME/FLASH API)
- Dependency analysis
- Out of order scheduling

Inversion of a SPD Matrix

Given symmetric positive definite matrix

$$A \in R^{n \times n}$$

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 3 \end{bmatrix}$$

Inversion of a SPD Matrix

Given symmetric positive definite matrix

$$A \in R^{n \times n}$$

$$A = \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 2 & 1 \\ \hline 1 & 1 & 3 \\ \hline \end{array}$$

1. Cholesky factorization (CHOL)

$$A \rightarrow U^T U$$

$$U = \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 1.4 \\ \hline \end{array}$$

Inversion of a SPD Matrix

Given symmetric positive definite matrix

$$A \in R^{n \times n}$$

$$A = \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 2 & 1 \\ \hline 1 & 1 & 3 \\ \hline \end{array}$$

1. Cholesky factorization (CHOL)

$$A \rightarrow U^T U$$

$$U = \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 1.4 \\ \hline \end{array}$$

2. Inversion of triangular matrix (TRINV)

$$R := U^{-1}$$

$$R = \begin{array}{|c|c|c|} \hline 1 & -1 & -0.7 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0.7 \\ \hline \end{array}$$

Inversion of a SPD Matrix

Given symmetric positive definite matrix

$$A \in R^{n \times n}$$

$$A = \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 2 & 1 \\ \hline 1 & 1 & 3 \\ \hline \end{array}$$

1. Cholesky factorization (CHOL)

$$A \rightarrow U^T U$$

$$U = \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 1.4 \\ \hline \end{array}$$

2. Inversion of triangular matrix (TRINV)

$$R := U^{-1}$$

$$R = \begin{array}{|c|c|c|} \hline 1 & -1 & -0.7 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0.7 \\ \hline \end{array}$$

3. Triangular transpose matrix mult. (TTMM)

$$A^{-1} := RR^T$$

$$A^{-1} = \begin{array}{|c|c|c|} \hline 2.5 & -1 & -0.5 \\ \hline -1 & 1 & 0 \\ \hline -0.5 & 0 & 0.5 \\ \hline \end{array}$$

Inversion of a SPD Matrix – Proof

SPD matrix $A \in R^{n \times n}$

1. (CHOL) $A \rightarrow U^T U$
2. (TRINV) $R := U^{-1}$
3. (TTMM) $A^{-1} := RR^T$

Proof:

AA^{-1}

Inversion of a SPD Matrix – Proof

SPD matrix $A \in R^{n \times n}$

1. (CHOL) $A \rightarrow U^T U$
2. (TRINV) $R := U^{-1}$
3. (TTMM) $A^{-1} := RR^T$

Proof:

$$AA^{-1} \overset{1,3}{=} U^T URR^T$$

Inversion of a SPD Matrix – Proof

SPD matrix $A \in R^{n \times n}$

1. (CHOL) $A \rightarrow U^T U$

2. (TRINV) $R := U^{-1}$

3. (TTMM) $A^{-1} := RR^T$

Proof:

$$AA^{-1} \overset{1,3}{=} U^T URR^T \overset{2}{=} U^T UU^{-1}U^{-T}$$

Inversion of a SPD Matrix – Proof

SPD matrix $A \in R^{n \times n}$

1. (CHOL) $A \rightarrow U^T U$
2. (TRINV) $R := U^{-1}$
3. (TTMM) $A^{-1} := RR^T$

Proof:

$$AA^{-1} \overset{1,3}{=} U^T URR^T \overset{2}{=} U^T UU^{-1}U^{-T} = U^T U^{-T}$$

Inversion of a SPD Matrix – Proof

SPD matrix $A \in R^{n \times n}$

1. (CHOL) $A \rightarrow U^T U$
2. (TRINV) $R := U^{-1}$
3. (TTMM) $A^{-1} := RR^T$

Proof:

$$AA^{-1} \overset{1,3}{=} U^T URR^T \overset{2}{=} U^T UU^{-1}U^{-T} = U^T U^{-T} = I$$

One variant of computing (CHOL)

Algorithm: $A := \text{CHOL\_BLK}(A)$ | $A := \text{TRINV\_BLK}(A)$ | $A := \text{TTMM\_BLK}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right)$

where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Determine block size b

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

where A_{11} **is** $b \times b$

CHOL	TRINV	TTMM
Variant 1:	Variant 1:	Variant 1:
$A_{01} := A_{00}^{-T} A_{01}$	$A_{01} := A_{00} A_{01}$	$A_{00} := A_{00} + A_{01} A_{01}^T$
$A_{11} := A_{11} - A_{01}^T A_{01}$	$A_{01} := -A_{01} A_{11}^{-1}$	$A_{01} := A_{01} A_{11}^T$
$A_{11} := \text{CHOL}(A_{11})$	$A_{11} := A_{11}^{-1}$	$A_{11} := A_{11} A_{11}^T$
Variant 2:	Variant 2:	Variant 2:
$A_{11} := A_{11} - A_{01}^T A_{01}$	$A_{12} := A_{12} A_{22}^{-1}$	$A_{01} := A_{01} A_{11}^T$
$A_{11} := \text{CHOL}(A_{11})$	$A_{12} := -A_{11}^{-1} A_{12}$	$A_{01} := A_{01} + A_{02} A_{12}^T$
$A_{12} := A_{12} - A_{01}^T A_{02}$	$A_{11} := A_{11}^{-1}$	$A_{11} := A_{11} A_{11}^T$
$A_{12} := A_{11}^{-T} A_{12}$		$A_{11} := A_{11} + A_{12} A_{12}^T$
Variant 3:	Variant 3:	Variant 3:
$A_{11} := \text{CHOL}(A_{11})$	$A_{12} := -A_{11}^{-1} A_{12}$	$A_{11} := A_{11} A_{11}^T$
$A_{12} := A_{11}^{-T} A_{12}$	$A_{02} := A_{02} + A_{01} A_{12}$	$A_{11} := A_{11} + A_{12} A_{12}^T$
$A_{22} := A_{22} - A_{12}^T A_{12}$	$A_{01} := A_{01} A_{11}^{-1}$	$A_{12} := A_{12} A_{22}^T$
	$A_{11} := A_{11}^{-1}$	

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

endwhile

Source: Paper

One variant of computing (CHOL)

Algorithm: $A := \text{CHOL\_BLK}(A)$ | $A := \text{TRINV\_BLK}(A)$ | $A := \text{TTMM\_BLK}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right)$
where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**
Determine block size b
Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

where A_{11} **is** $b \times b$

CHOL	TRINV	TTMM
Variant 1:	Variant 1:	Variant 1:
$A_{01} := A_{00}^{-T} A_{01}$	$A_{01} := A_{00} A_{01}$	$A_{00} := A_{00} + A_{01} A_{01}^T$
$A_{11} := A_{11} - A_{01}^T A_{01}$	$A_{01} := -A_{01} A_{11}^{-1}$	$A_{01} := A_{01} A_{11}^T$
$A_{11} := \text{CHOL}(A_{11})$	$A_{11} := A_{11}^{-1}$	$A_{11} := A_{11} A_{11}^T$
Variant 2:	Variant 2:	Variant 2:
$A_{11} := A_{11} - A_{01}^T A_{01}$	$A_{12} := A_{12} A_{22}^{-1}$	$A_{01} := A_{01} A_{11}^T$
$A_{11} := \text{CHOL}(A_{11})$	$A_{12} := -A_{11}^{-1} A_{12}$	$A_{01} := A_{01} + A_{02} A_{12}^T$
$A_{12} := A_{12} - A_{01}^T A_{02}$	$A_{11} := A_{11}^{-1}$	$A_{11} := A_{11} A_{11}^T$
$A_{12} := A_{11}^{-T} A_{12}$		$A_{11} := A_{11} + A_{12} A_{12}^T$
Variant 3:	Variant 3:	Variant 3:
$A_{11} := \text{CHOL}(A_{11})$	$A_{12} := -A_{11}^{-1} A_{12}$	$A_{11} := A_{11} A_{11}^T$
$A_{12} := A_{11}^{-T} A_{12}$	$A_{02} := A_{02} + A_{01} A_{12}$	$A_{11} := A_{11} + A_{12} A_{12}^T$
$A_{22} := A_{22} - A_{12}^T A_{12}$	$A_{01} := A_{01} A_{11}^{-1}$	$A_{12} := A_{12} A_{22}^T$
	$A_{11} := A_{11}^{-1}$	

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

endwhile

Algorithm: $A := \text{CHOL\_BLK}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right)$
where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**
Determine block size b
Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

where A_{11} **is** $b \times b$

$A_{11} := \text{CHOL}(A_{11})$
$A_{12} := A_{11}^{-T} A_{12}$
$A_{22} := A_{22} - A_{12}^T A_{12}$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

endwhile

Source: Paper

First iteration (4x4 matrix blocks)

Algorithm: $A := \text{CHOL\_BLK}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right)$

where A_{TL} **is** 0×0

while $m(A_{TL}) < m(A)$ **do**

Determine block size b

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

where A_{11} **is** $b \times b$

$A_{11} := \text{CHOL}(A_{11})$

$A_{12} := A_{11}^{-T} A_{12}$

$A_{22} := A_{22} - A_{12}^{T} A_{12}$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

endwhile

$A_{1,1}$	$A_{1,2}$	$A_{1,3}$	$A_{1,4}$
$(A_{2,1})$	$A_{2,2}$	$A_{2,3}$	$A_{2,4}$
$(A_{3,1})$	$(A_{3,2})$	$A_{3,3}$	$A_{3,4}$
$(A_{4,1})$	$(A_{4,2})$	$(A_{4,3})$	$A_{4,4}$

Source: Paper

First iteration (4x4 matrix blocks)

Algorithm: $A := \text{CHOL\_BLK}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right)$

where A_{TL} **is** 0×0

while $m(A_{TL}) < m(A)$ **do**

Determine block size b

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

where A_{11} **is** $b \times b$

$A_{11} := \text{CHOL}(A_{11})$

$A_{12} := A_{11}^{-T} A_{12}$

$A_{22} := A_{22} - A_{12}^{T} A_{12}$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

endwhile

$A_{1,1}$	$A_{1,2}$	$A_{1,3}$	$A_{1,4}$
$(A_{2,1})$	$A_{2,2}$	$A_{2,3}$	$A_{2,4}$
$(A_{3,1})$	$(A_{3,2})$	$A_{3,3}$	$A_{3,4}$
$(A_{4,1})$	$(A_{4,2})$	$(A_{4,3})$	$A_{4,4}$

Source: Paper

First iteration (4x4 matrix blocks)

Computations:

$CHOL_0$ $CHOL(A_{1,1})$	$TRSM_1$ $Inv(A_{1,1})\ A_{1,2}$	$TRSM_2$ $Inv(A_{1,1})\ A_{1,3}$	$TRSM_3$ $Inv(A_{1,1})\ A_{1,4}$
$(A_{2,1})$	$SYRK_4$ $A_{2,2} - A^T_{1,2}A_{1,2}$	$GEMM_5$ $A_{2,3} - A^T_{1,2}A_{1,3}$	$GEMM_6$ $A_{2,4} - A^T_{1,2}A_{1,4}$
$(A_{3,1})$	$(A_{3,2})$	$SYRK_7$ $A_{3,3} - A^T_{1,3}A_{1,3}$	$GEMM_8$ $A_{3,4} - A^T_{1,3}A_{1,4}$
$(A_{4,1})$	$(A_{4,2})$	$(A_{4,3})$	$SYRK_9$ $A_{4,4} - A^T_{1,4}A_{1,4}$

$A_{11} := \mathrm{CHOL}(A_{11})$
$A_{12} := A_{11}^{-T} A_{12}$
$A_{22} := A_{22} - A_{12}^T A_{12}$

Source: Paper

First iteration (4x4 matrix blocks)

Computations:

$CHOL_0$ $CHOL(A_{1,1})$	$TRSM_1$ $Inv(A_{1,1})\ A_{1,2}$	$TRSM_2$ $Inv(A_{1,1})\ A_{1,3}$	$TRSM_3$ $Inv(A_{1,1})\ A_{1,4}$
$(A_{2,1})$	$SYRK_4$ $A_{2,2} - A^T_{1,2}A_{1,2}$	$GEMM_5$ $A_{2,3} - A^T_{1,2}A_{1,3}$	$GEMM_6$ $A_{2,4} - A^T_{1,2}A_{1,4}$
$(A_{3,1})$	$(A_{3,2})$	$SYRK_7$ $A_{3,3} - A^T_{1,3}A_{1,3}$	$GEMM_8$ $A_{3,4} - A^T_{1,3}A_{1,4}$
$(A_{4,1})$	$(A_{4,2})$	$(A_{4,3})$	$SYRK_9$ $A_{4,4} - A^T_{1,4}A_{1,4}$

$A_{11} := \mathrm{CHOL}(A_{11})$
$A_{12} := A_{11}^{-T} A_{12}$
$A_{22} := A_{22} - A_{12}^T A_{12}$

- CHOL
  Cholesky
  factorization

Source: Paper

First iteration (4x4 matrix blocks)

Computations:

$CHOL_0$ $CHOL(A_{1,1})$	$TRSM_1$ $Inv(A_{1,1})\ A_{1,2}$	$TRSM_2$ $Inv(A_{1,1})\ A_{1,3}$	$TRSM_3$ $Inv(A_{1,1})\ A_{1,4}$
$(A_{2,1})$	$SYRK_4$ $A_{2,2} - A^T_{1,2}A_{1,2}$	$GEMM_5$ $A_{2,3} - A^T_{1,2}A_{1,3}$	$GEMM_6$ $A_{2,4} - A^T_{1,2}A_{1,4}$
$(A_{3,1})$	$(A_{3,2})$	$SYRK_7$ $A_{3,3} - A^T_{1,3}A_{1,3}$	$GEMM_8$ $A_{3,4} - A^T_{1,3}A_{1,4}$
$(A_{4,1})$	$(A_{4,2})$	$(A_{4,3})$	$SYRK_9$ $A_{4,4} - A^T_{1,4}A_{1,4}$

$A_{11} := \mathrm{CHOL}(A_{11})$
$A_{12} := A_{11}^{-T} A_{12}$
$A_{22} := A_{22} - A_{12}^T A_{12}$

- CHOL
  Cholesky factorization
- TRSM
  Triangular solves with multiple right hand sides

Source: Paper

First iteration (4x4 matrix blocks)

Computations:

$CHOL_0$ $CHOL(A_{1,1})$	$TRSM_1$ $Inv(A_{1,1})\ A_{1,2}$	$TRSM_2$ $Inv(A_{1,1})\ A_{1,3}$	$TRSM_3$ $Inv(A_{1,1})\ A_{1,4}$
$(A_{2,1})$	$SYRK_4$ $A_{2,2} - A^T_{1,2}A_{1,2}$	$GEMM_5$ $A_{2,3} - A^T_{1,2}A_{1,3}$	$GEMM_6$ $A_{2,4} - A^T_{1,2}A_{1,4}$
$(A_{3,1})$	$(A_{3,2})$	$SYRK_7$ $A_{3,3} - A^T_{1,3}A_{1,3}$	$GEMM_8$ $A_{3,4} - A^T_{1,3}A_{1,4}$
$(A_{4,1})$	$(A_{4,2})$	$(A_{4,3})$	$SYRK_9$ $A_{4,4} - A^T_{1,4}A_{1,4}$

$A_{11} := \mathrm{CHOL}(A_{11})$
$A_{12} := A_{11}^{-T} A_{12}$
$A_{22} := A_{22} - A_{12}^T A_{12}$

- CHOL
  Cholesky factorization
- TRSM
  Triangular solves with multiple right hand sides
- SYRK
  Symmetric rank-k update

Source: Paper

First iteration (4x4 matrix blocks)

Computations:

$CHOL_0$ $CHOL(A_{1,1})$	$TRSM_1$ $Inv(A_{1,1})\ A_{1,2}$	$TRSM_2$ $Inv(A_{1,1})\ A_{1,3}$	$TRSM_3$ $Inv(A_{1,1})\ A_{1,4}$
$(A_{2,1})$	$SYRK_4$ $A_{2,2} - A^T_{1,2}A_{1,2}$	$GEMM_5$ $A_{2,3} - A^T_{1,2}A_{1,3}$	$GEMM_6$ $A_{2,4} - A^T_{1,2}A_{1,4}$
$(A_{3,1})$	$(A_{3,2})$	$SYRK_7$ $A_{3,3} - A^T_{1,3}A_{1,3}$	$GEMM_8$ $A_{3,4} - A^T_{1,3}A_{1,4}$
$(A_{4,1})$	$(A_{4,2})$	$(A_{4,3})$	$SYRK_9$ $A_{4,4} - A^T_{1,4}A_{1,4}$

$A_{11} := \mathrm{CHOL}(A_{11})$
$A_{12} := A_{11}^{-T} A_{12}$
$A_{22} := A_{22} - A_{12}^T A_{12}$

- CHOL
  Cholesky factorization
- TRSM
  Triangular solves with multiple right hand sides
- SYRK
  Symmetric rank-k update
- GEMM
  Matrix-Matrix multiplication

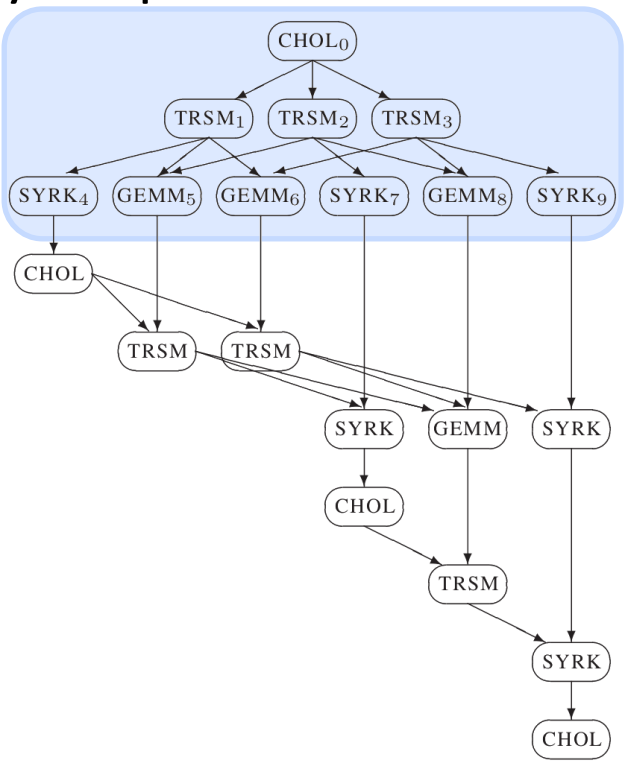
Source: Paper

First iteration (4x4 matrix blocks)

Computations:

$CHOL_0$ $CHOL(A_{1,1})$	$TRSM_1$ $Inv(A_{1,1})\ A_{1,2}$	$TRSM_2$ $Inv(A_{1,1})\ A_{1,3}$	$TRSM_3$ $Inv(A_{1,1})\ A_{1,4}$
$(A_{2,1})$	$SYRK_4$ $A_{2,2} - A^T_{1,2}A_{1,2}$	$GEMM_5$ $A_{2,3} - A^T_{1,2}A_{1,3}$	$GEMM_6$ $A_{2,4} - A^T_{1,2}A_{1,4}$
$(A_{3,1})$	$(A_{3,2})$	$SYRK_7$ $A_{3,3} - A^T_{1,3}A_{1,3}$	$GEMM_8$ $A_{3,4} - A^T_{1,3}A_{1,4}$
$(A_{4,1})$	$(A_{4,2})$	$(A_{4,3})$	$SYRK_9$ $A_{4,4} - A^T_{1,4}A_{1,4}$

Dependency Graph:



- CHOL
  Cholesky factorization
- TRSM
  Triangular solves with multiple right hand sides
- SYRK
  Symmetric rank-k update
- GEMM
  Matrix-Matrix multiplication

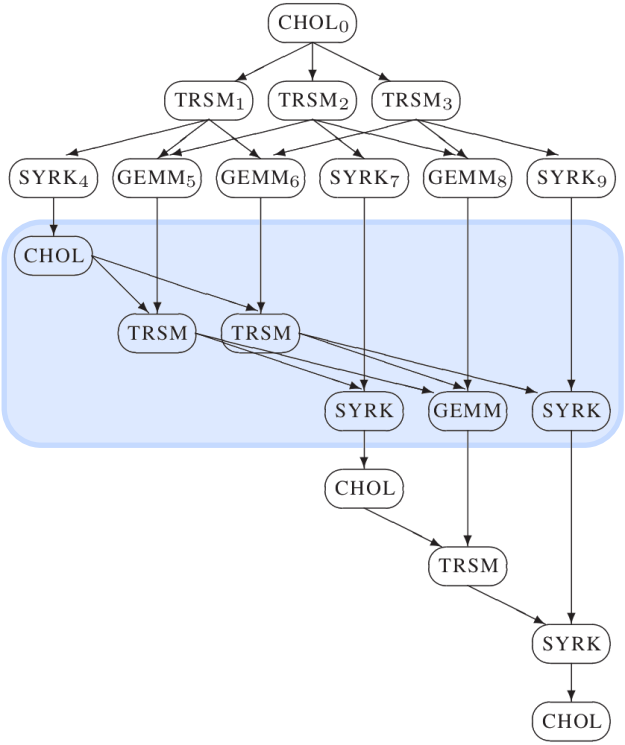
Source: Paper

First iteration (4x4 matrix blocks)

Computations:

Dependency Graph:

$(A_{1,1})$	$(A_{1,2})$	$(A_{1,3})$	$(A_{1,4})$
$(A_{2,1})$	$CHOL_0$ $CHOL(A_{2,2})$	$TRSM_1$ $Inv(A_{2,2})\ A_{2,3}$	$TRSM_2$ $Inv(A_{2,2})\ A_{2,4}$
$(A_{3,1})$	$(A_{3,2})$	$SYRK_3$ $A_{3,3} - A^T_{2,3}A_{2,3}$	$GEMM_4$ $A_{3,4} - A^T_{2,3}A_{2,4}$
$(A_{4,1})$	$(A_{4,2})$	$(A_{3,2})$	$SYRK_7$ $A_{4,4} - A^T_{2,4}A_{2,4}$



- CHOL
  Cholesky factorization
- TRSM
  Triangular solves with multiple right hand sides
- SYRK
  Symmetric rank-k update
- GEMM
  Matrix-Matrix multiplication

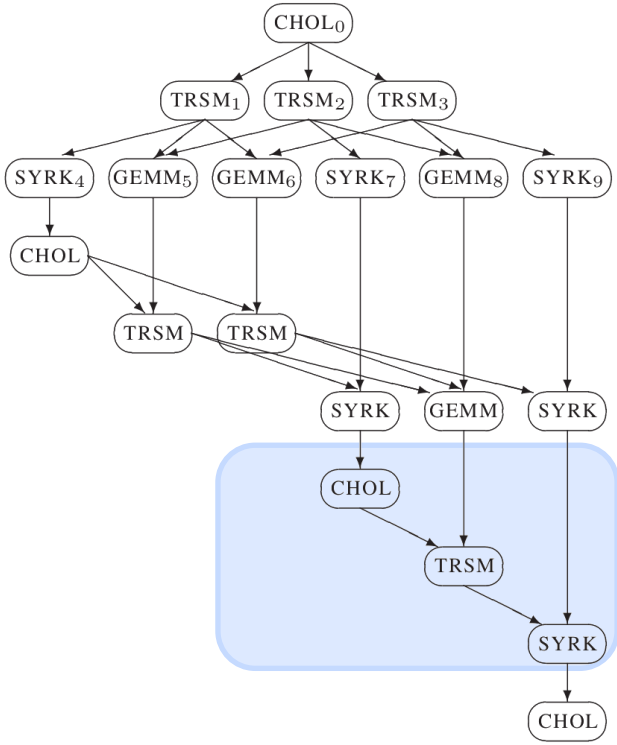
Source: Paper

First iteration (4x4 matrix blocks)

Computations:

$(A_{1,1})$	$(A_{1,2})$	$(A_{1,3})$	$(A_{1,4})$
$(A_{2,1})$	$(A_{2,2})$	$(A_{2,3})$	$(A_{2,4})$
$(A_{3,1})$	$(A_{3,2})$	$CHOL_0$ $CHOL(A_{3,3})$	$TRSM_2$ $Inv(A_{3,3})\ A_{3,4}$
$(A_{4,1})$	$(A_{4,2})$	$(A_{4,3})$	$SYRK_3$ $A_{4,4} - A^T_{3,4}A_{3,4}$

Dependency Graph:



- CHOL
  Cholesky factorization
- TRSM
  Triangular solves with multiple right hand sides
- SYRK
  Symmetric rank-k update
- GEMM
  Matrix-Matrix multiplication

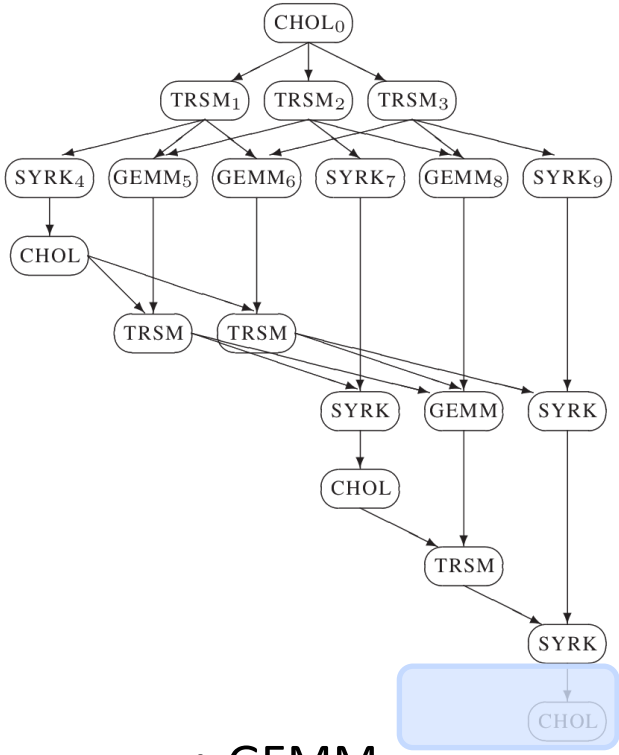
Source: Paper

First iteration (4x4 matrix blocks)

Computations:

Dependency Graph:

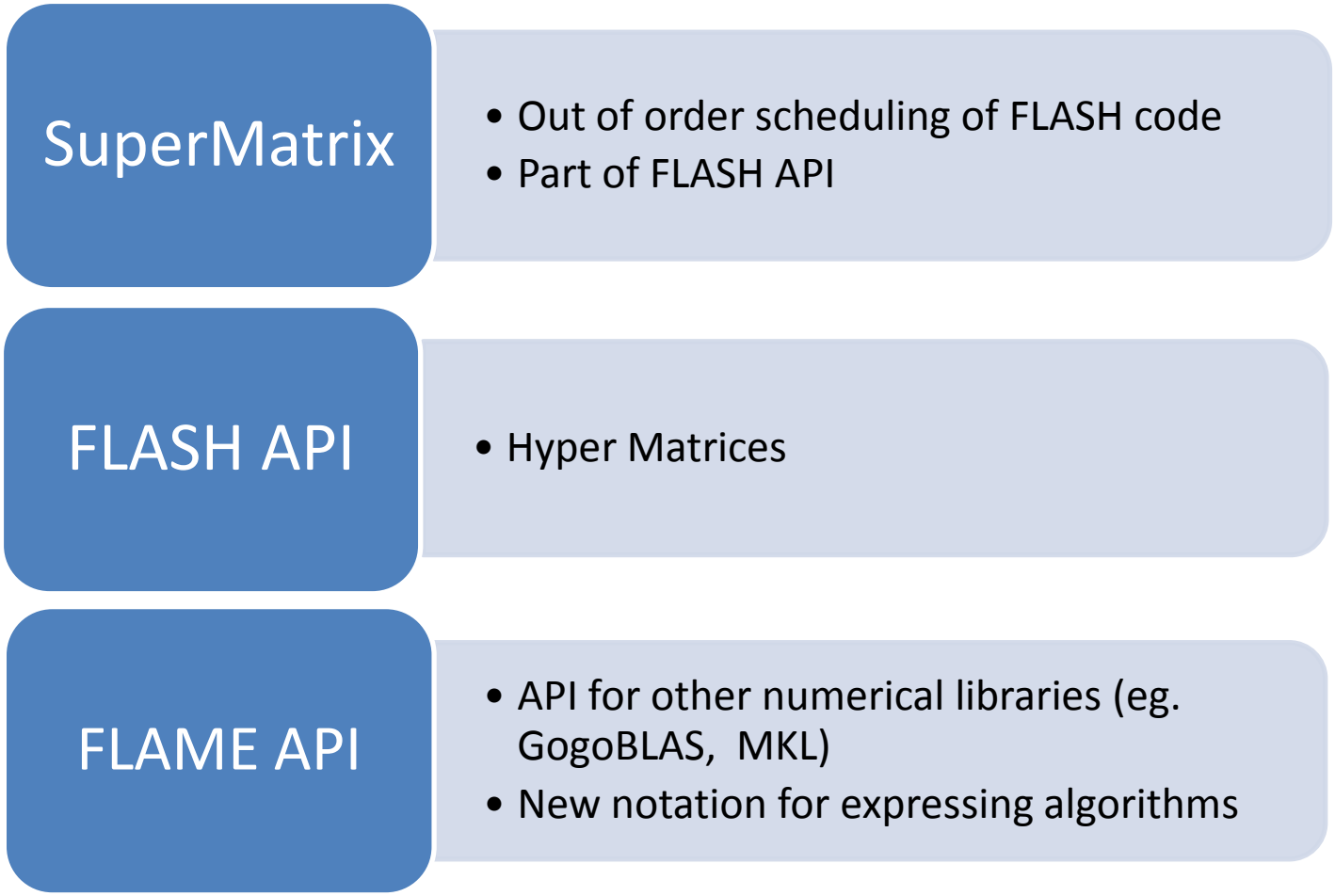
$(A_{1,1})$	$(A_{1,2})$	$(A_{1,3})$	$(A_{1,4})$
$(A_{2,1})$	$(A_{2,2})$	$(A_{2,3})$	$(A_{2,4})$
$(A_{3,1})$	$(A_{3,2})$	$(A_{3,3})$	$(A_{3,4})$
$(A_{4,1})$	$(A_{4,2})$	$(A_{4,3})$	$CHOL_0$ $CHOL(A_{4,4})$



- CHOL
  Cholesky factorization
- TRSM
  Triangular solves with multiple right hand sides
- SYRK
  Symmetric rank-k update
- GEMM
  Matrix-Matrix multiplication

Source: Paper

Layers of SuperMatrix



FLASH implementation of (CHOL)

Implementation:

```
FLA_Error FLASH_Chol( FLA_Obj A )
{
   FLA_Obj ATL, ATR,      A00, A01, A02,
           ABL, ABR,      A10, A11, A12,
                          A20, A21, A22;

   FLA_Part_2x2( A,     &ATL, &ATR,
                        &ABL, &ABR,     0, 0, FLA_TL );

   while ( FLA_Obj_length( ATL ) < FLA_Obj_length( A ) )
   {
      FLA_Repart_2x2_to_3x3(
         ATL, /**/ ATR,       &A00, /**/ &A01, &A02,
      /* ************* */   /* ******************** */
                              &A10, /**/ &A11, &A12,
         ABL, /**/ ABR,       &A20, /**/ &A21, &A22,
         1, 1, FLA_BR );

      /*-----------------------------------------------*/
         FLASH_Chol( FLA_UPPER_TRIANGULAR, A11 );
         FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
                     FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
                     FLA_ONE, A11, A12 );
         FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_TRANSPOSE,
                     FLA_MINUS_ONE, A12, FLA_ONE, A22 );
      /*-----------------------------------------------*/

      FLA_Cont_with_3x3_to_2x2(
         &ATL, /**/ &ATR,        A00, A01, /**/ A02,
                                 A10, A11, /**/ A12,
       /* ************** */   /* ****************** */
         &ABL, /**/ &ABR,        A20, A21, /**/ A22,
         FLA_TL );
   }
   return FLA_SUCCESS;
}
```

Procedural:

Algorithm: $A := \text{CHOL\_BLK}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right)$

where A_{TL} **is** 0×0

while $m(A_{TL}) < m(A)$ **do**

Determine block size b

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

where A_{11} **is** $b \times b$

$$A_{11} := \text{CHOL}(A_{11})$$
$$A_{12} := A_{11}^{-T} A_{12}$$
$$A_{22} := A_{22} - A_{12}^{T} A_{12}$$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

endwhile

Source: Paper

FLASH implementation of (CHOL)

Implementation:

```
FLA_Error FLASH_Chol( FLA_Obj A )
{
   FLA_Obj ATL, ATR,      A00, A01, A02,
           ABL, ABR,      A10, A11, A12,
                          A20, A21, A22;

   Partition A → ( ATL | ATR )  where ATL is 0 × 0
                 ( ★   | ABR )
   while m(ATL) < m(A) do
   {
      Repartition
         ( ATL | ATR )  →  ( A00 | A01 | A02 )
         ( ★   | ABR )     ( ★   | A11 | A12 )
                           ( ★   | ★   | A22 )
         where A11 is b × b

      /*-----------------------------------------*/
      A11 := CHOL(A11)
      A12 := A11^-T A12
      A22 := A22 - A12^T A12
      /*-----------------------------------------*/

      Continue with
         ( ATL | ATR )  ←  ( A00 | A01 | A02 )
         ( ★   | ABR )     ( ★   | A11 | A12 )
                           ( ★   | ★   | A22 )
   endwhile
   return FLA_SUCCESS;
}
```

Procedural:

Algorithm: $A := \text{CHOL\_BLK}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right)$ **where** A_{TL} **is** 0×0

while $m(A_{TL}) < m(A)$ **do**

Determine block size b

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

where A_{11} **is** $b \times b$

$$A_{11} := \text{CHOL}(A_{11})$$
$$A_{12} := A_{11}^{-T} A_{12}$$
$$A_{22} := A_{22} - A_{12}^{T} A_{12}$$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

endwhile

Source: Paper

FLASH implementation of (CHOL)

Implementation:

```
FLA_Error FLASH_Chol( FLA_Obj A )
{
   FLA_Obj ATL, ATR,      A00, A01, A02,
           ABL, ABR,      A10, A11, A12,
                          A20, A21, A22;

   FLA_Part_2x2( A,      &ATL, &ATR,
                         &ABL, &ABR,     0, 0, FLA_TL );

   while ( FLA_Obj_length( ATL ) < FLA_Obj_length( A ) )
   {
      FLA_Repart_2x2_to_3x3(
         ATL, /**/ ATR,       &A00, /**/ &A01, &A02,
      /* ************* */   /* ******************** */
                              &A10, /**/ &A11, &A12,
         ABL, /**/ ABR,       &A20, /**/ &A21, &A22,
         1, 1, FLA_BR );

      /*--------------------------------------------*/
         FLASH_Chol( FLA_UPPER_TRIANGULAR, A11 );
         FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
                     FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
                     FLA_ONE, A11, A12 );
         FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_TRANSPOSE,
                     FLA_MINUS_ONE, A12, FLA_ONE, A22 );
      /*--------------------------------------------*/

      FLA_Cont_with_3x3_to_2x2(
         &ATL, /**/ &ATR,        A00, A01, /**/ A02,
                                 A10, A11, /**/ A12,
       /* ************** */   /* ****************** */
         &ABL, /**/ &ABR,        A20, A21, /**/ A22,
         FLA_TL );
   }
   return FLA_SUCCESS;
}
```

Procedural:

Algorithm: $A := \text{CHOL\_BLK}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right)$
where A_{TL} **is** 0×0

while $m(A_{TL}) < m(A)$ **do**

Determine block size b

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

where A_{11} **is** $b \times b$

$$A_{11} := \text{CHOL}(A_{11})$$
$$A_{12} := A_{11}^{-T} A_{12}$$
$$A_{22} := A_{22} - A_{12}^T A_{12}$$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

endwhile

Source: Paper

Computing SPD^{-1}

Algorithm: $A := \text{CHOL\_BLK}(A)$ | $A := \text{TRINV\_BLK}(A)$ | $A := \text{TTMM\_BLK}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right)$

where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**

Determine block size b

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

where A_{11} **is** $b \times b$

CHOL

Variant 1:

$A_{01} := A_{00}^{-T} A_{01}$
$A_{11} := A_{11} - A_{01}^T A_{01}$
$A_{11} := \text{CHOL}(A_{11})$

Variant 2:

$A_{11} := A_{11} - A_{01}^T A_{01}$
$A_{11} := \text{CHOL}(A_{11})$
$A_{12} := A_{12} - A_{01}^T A_{02}$
$A_{12} := A_{11}^{-T} A_{12}$

Variant 3:

$A_{11} := \text{CHOL}(A_{11})$
$A_{12} := A_{11}^{-T} A_{12}$
$A_{22} := A_{22} - A_{12}^T A_{12}$

TRINV

Variant 1:

$A_{01} := A_{00} A_{01}$
$A_{01} := -A_{01} A_{11}^{-1}$
$A_{11} := A_{11}^{-1}$

Variant 2:

$A_{12} := A_{12} A_{22}^{-1}$
$A_{12} := -A_{11}^{-1} A_{12}$
$A_{11} := A_{11}^{-1}$

Variant 3:

$A_{12} := -A_{11}^{-1} A_{12}$
$A_{02} := A_{02} + A_{01} A_{12}$
$A_{01} := A_{01} A_{11}^{-1}$
$A_{11} := A_{11}^{-1}$

TTMM

Variant 1:

$A_{00} := A_{00} + A_{01} A_{01}^T$
$A_{01} := A_{01} A_{11}^T$
$A_{11} := A_{11} A_{11}^T$

Variant 2:

$A_{01} := A_{01} A_{11}^T$
$A_{01} := A_{01} + A_{02} A_{12}^T$
$A_{11} := A_{11} A_{11}^T$
$A_{11} := A_{11} + A_{12} A_{12}^T$

Variant 3:

$A_{11} := A_{11} A_{11}^T$
$A_{11} := A_{11} + A_{12} A_{12}^T$
$A_{12} := A_{12} A_{22}^T$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

endwhile

Source: Paper

Computing SPD^{-1}

Algorithm: $A := \text{CHOL\_BLK}(A)$ | $A := \text{TRINV\_BLK}(A)$ | $A := \text{TTMM\_BLK}(A)$

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right)$
where A_{TL} is 0×0

while $m(A_{TL}) < m(A)$ **do**
Determine block size b
Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

where A_{11} **is** $b \times b$

CHOL	TRINV	TTMM
Variant 1:	Variant 1:	Variant 1:
$A_{01} := A_{00}^{-T} A_{01}$	$A_{01} := A_{00} A_{01}$	$A_{00} := A_{00} + A_{01} A_{01}^T$
$A_{11} := A_{11} - A_{01}^T A_{01}$	$A_{01} := -A_{01} A_{11}^{-1}$	$A_{01} := A_{01} A_{11}^T$
$A_{11} := \text{CHOL}(A_{11})$	$A_{11} := A_{11}^{-1}$	$A_{11} := A_{11} A_{11}^T$
Variant 2:	Variant 2:	Variant 2:
$A_{11} := A_{11} - A_{01}^T A_{01}$	$A_{12} := A_{12} A_{22}^{-1}$	$A_{01} := A_{01} A_{11}^T$
$A_{11} := \text{CHOL}(A_{11})$	$A_{12} := -A_{11}^{-1} A_{12}$	$A_{01} := A_{01} + A_{02} A_{12}^T$
$A_{12} := A_{12} - A_{01}^T A_{02}$	$A_{11} := A_{11}^{-1}$	$A_{11} := A_{11} A_{11}^T$
$A_{12} := A_{11}^{-T} A_{12}$		$A_{11} := A_{11} + A_{12} A_{12}^T$
Variant 3:	Variant 3:	Variant 3:
$A_{11} := \text{CHOL}(A_{11})$	$A_{12} := -A_{11}^{-1} A_{12}$	$A_{11} := A_{11} A_{11}^T$
$A_{12} := A_{11}^{-T} A_{12}$	$A_{02} := A_{02} + A_{01} A_{12}$	$A_{11} := A_{11} + A_{12} A_{12}^T$
$A_{22} := A_{22} - A_{12}^T A_{12}$	$A_{01} := A_{01} A_{11}^{-1}$	$A_{12} := A_{12} A_{22}^T$
	$A_{11} := A_{11}^{-1}$	

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline \star & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline \star & A_{11} & A_{12} \\ \hline \star & \star & A_{22} \end{array}\right)$$

endwhile

```
FLA_Error FLASH_SPD_inv_u_op( int op, FLA_Obj A )
{
   FLA_Obj ATL, ATR,      A00, A01, A02,
           ABL, ABR,      A10, A11, A12,
                          A20, A21, A22;

   FLA_Part_2x2( A,     &ATL, &ATR,
                        &ABL, &ABR,     0, 0, FLA_TL );

   while ( FLA_Obj_length( ATL ) < FLA_Obj_length( A ) )
   {
      FLA_Repart_2x2_to_3x3(
         ATL, /**/ ATR,       &A00, /**/ &A01, &A02,
      /* ************* */   /* ******************** */
                              &A10, /**/ &A11, &A12,
         ABL, /**/ ABR,       &A20, /**/ &A21, &A22,
         1, 1, FLA_BR );

      /*----------------------------------------------*/
      switch ( op )
      {
      FLASH_Chol_op:   /* Variant 3 */
         FLASH_Chol( FLA_UPPER_TRIANGULAR, A11 );
         FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
                     FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
                     FLA_ONE, A11, A12 );
         FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_TRANSPOSE,
                     FLA_MINUS_ONE, A12, FLA_ONE, A22 );
         break;
      FLASH_Trinv_op:  /* Variant 3 */
         FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
                     FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
                     FLA_MINUS_ONE, A11, A12 );
         FLASH_Gemm( FLA_NO_TRANSPOSE, FLA_NO_TRANSPOSE,
                     FLA_ONE, A01, A12, FLA_ONE, A02 );
         FLASH_Trsm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
                     FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
                     FLA_ONE, A11, A01 );
         FLASH_Trinv( FLA_UPPER_TRIANGULAR,
                      FLA_NONUNIT_DIAG, A11 );
         break;
      FLASH_Ttmm_op:   /* Variant 1 */
         FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_NO_TRANSPOSE,
                     FLA_ONE, A01, FLA_ONE, A00 );
         FLASH_Trmm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
                     FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
                     FLA_ONE, A11, A01 );
         FLASH_Ttmm( FLA_UPPER_TRIANGULAR, A11 );
         break;
      }
      /*----------------------------------------------*/

      FLA_Cont_with_3x3_to_2x2(
         &ATL, /**/ &ATR,        A00, A01, /**/ A02,
                                 A10, A11, /**/ A12,
       /* ************** */   /* ****************** */
         &ABL, /**/ &ABR,        A20, A21, /**/ A22,
         FLA_TL );
   }
   return FLA_SUCCESS;
}
```

Source: Paper

Computing SPD^{-1}

CHOL

$$A \rightarrow U^T U$$

$$\begin{aligned} A_{11} &:= \text{CHOL}(A_{11}) \\ A_{12} &:= A_{11}^{-T} A_{12} \\ A_{22} &:= A_{22} - A_{12}^T A_{12} \end{aligned}$$

```
FLASH_Chol( FLA_UPPER_TRIANGULAR, A11 );
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A12 );
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_TRANSPOSE,
            FLA_MINUS_ONE, A12, FLA_ONE, A22 );
```

TRINV

$$R := U^{-1}$$

$$\begin{aligned} A_{12} &:= -A_{11}^{-1} A_{12} \\ A_{02} &:= A_{02} + A_{01} A_{12} \\ A_{01} &:= A_{01} A_{11}^{-1} \\ A_{11} &:= A_{11}^{-1} \end{aligned}$$

```
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_MINUS_ONE, A11, A12 );
FLASH_Gemm( FLA_NO_TRANSPOSE, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, A12, FLA_ONE, A02 );
FLASH_Trsm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Trinv( FLA_UPPER_TRIANGULAR,
             FLA_NONUNIT_DIAG, A11 );
```

TTMM

$$A^{-1} := RR^T$$

$$\begin{aligned} A_{00} &:= A_{00} + A_{01} A_{01}^T \\ A_{01} &:= A_{01} A_{11}^T \\ A_{11} &:= A_{11} A_{11}^T \end{aligned}$$

```
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, FLA_ONE, A00 );
FLASH_Trmm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Ttmm( FLA_UPPER_TRIANGULAR, A11 );
```

Source: Paper

Computing SPD^{-1}

CHOL

$$A \rightarrow U^T U$$

$$\begin{aligned} A_{11} &:= \text{CHOL}(A_{11}) \\ A_{12} &:= A_{11}^{-T} A_{12} \\ A_{22} &:= A_{22} - A_{12}^T A_{12} \end{aligned}$$

```
FLASH_Chol( FLA_UPPER_TRIANGULAR, A11 );
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A12 );
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_TRANSPOSE,
            FLA_MINUS_ONE, A12, FLA_ONE, A22 );
```

TRINV

$$R := U^{-1}$$

$$\begin{aligned} A_{12} &:= -A_{11}^{-1} A_{12} \\ A_{02} &:= A_{02} + A_{01} A_{12} \\ A_{01} &:= A_{01} A_{11}^{-1} \\ A_{11} &:= A_{11}^{-1} \end{aligned}$$

```
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_MINUS_ONE, A11, A12 );
FLASH_Gemm( FLA_NO_TRANSPOSE, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, A12, FLA_ONE, A02 );
FLASH_Trsm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Trinv( FLA_UPPER_TRIANGULAR,
             FLA_NONUNIT_DIAG, A11 );
```

TTMM

$$A^{-1} := RR^T$$

$$\begin{aligned} A_{00} &:= A_{00} + A_{01} A_{01}^T \\ A_{01} &:= A_{01} A_{11}^T \\ A_{11} &:= A_{11} A_{11}^T \end{aligned}$$

```
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, FLA_ONE, A00 );
FLASH_Trmm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Ttmm( FLA_UPPER_TRIANGULAR, A11 );
```

Source: Paper

How to analyze dependencies

- FLAME/FLASH API: Last operand is overridden
- Preceding ones are strictly inputs
- Distinct between three types of dependencies:
  - Flow dependencies
  - Anti-dependencies
  - Output dependencies

```
FLASH_Chol( FLA_UPPER_TRIANGULAR, A11 );
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A12 );
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_TRANSPOSE,
            FLA_MINUS_ONE, A12, FLA_ONE, A22 );
```

```
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_MINUS_ONE, A11, A12 );
FLASH_Gemm( FLA_NO_TRANSPOSE, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, A12, FLA_ONE, A02 );
FLASH_Trsm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Trinv( FLA_UPPER_TRIANGULAR,
             FLA_NONUNIT_DIAG, A11 );
```

```
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, FLA_ONE, A00 );
FLASH_Trmm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Ttmm( FLA_UPPER_TRIANGULAR, A11 );
```

Source: Paper

Flow dependencies

- Read-after-write dependency

S_1: A = B + C
S_2: D = A + E

S1 must complete the write before
S2 can read.

```
FLASH_Chol( FLA_UPPER_TRIANGULAR, A11 );
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A12 );
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_TRANSPOSE,
            FLA_MINUS_ONE, A12, FLA_ONE, A22 );
```

```
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_MINUS_ONE, A11, A12 );
FLASH_Gemm( FLA_NO_TRANSPOSE, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, A12, FLA_ONE, A02 );
FLASH_Trsm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Trinv( FLA_UPPER_TRIANGULAR,
             FLA_NONUNIT_DIAG, A11 );
```

```
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, FLA_ONE, A00 );
FLASH_Trmm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Ttmm( FLA_UPPER_TRIANGULAR, A11 );
```

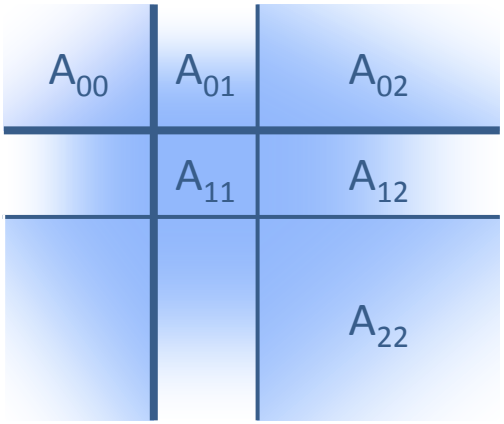
Source: Paper

Flow dependencies

- Read-after-write dependency

S_1: A = B + C
S_2: D = A + E

S1 must complete the write before S2 can read.




```
FLASH_Chol( FLA_UPPER_TRIANGULAR, A11 );
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A12 );
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_TRANSPOSE,
            FLA_MINUS_ONE, A12, FLA_ONE, A22 );
```

```
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_MINUS_ONE, A11, A12 );
FLASH_Gemm( FLA_NO_TRANSPOSE, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, A12, FLA_ONE, A02 );
FLASH_Trsm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Trinv( FLA_UPPER_TRIANGULAR,
             FLA_NONUNIT_DIAG, A11 );
```

```
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, FLA_ONE, A00 );
FLASH_Trmm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Ttmm( FLA_UPPER_TRIANGULAR, A11 );
```

→ Flow dependency

→ Intra-iterational
⇢ Inter-iterational

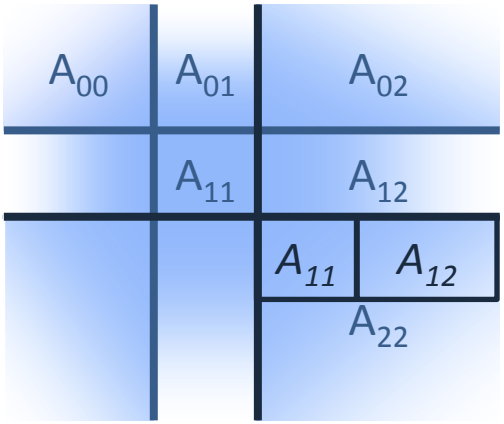
Source: Paper

Flow dependencies

- Read-after-write dependency

S_1: A = B + C
S_2: D = A + E

S1 must complete the write before S2 can read.




```
FLASH_Chol( FLA_UPPER_TRIANGULAR, A11 );
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A12 );
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_TRANSPOSE,
            FLA_MINUS_ONE, A12, FLA_ONE, A22 );
```

```
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_MINUS_ONE, A11, A12 );
FLASH_Gemm( FLA_NO_TRANSPOSE, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, A12, FLA_ONE, A02 );
FLASH_Trsm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Trinv( FLA_UPPER_TRIANGULAR,
             FLA_NONUNIT_DIAG, A11 );
```

```
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, FLA_ONE, A00 );
FLASH_Trmm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Ttmm( FLA_UPPER_TRIANGULAR, A11 );
```

→ Flow dependency

→ Intra-iterational
⇢ Inter-iterational

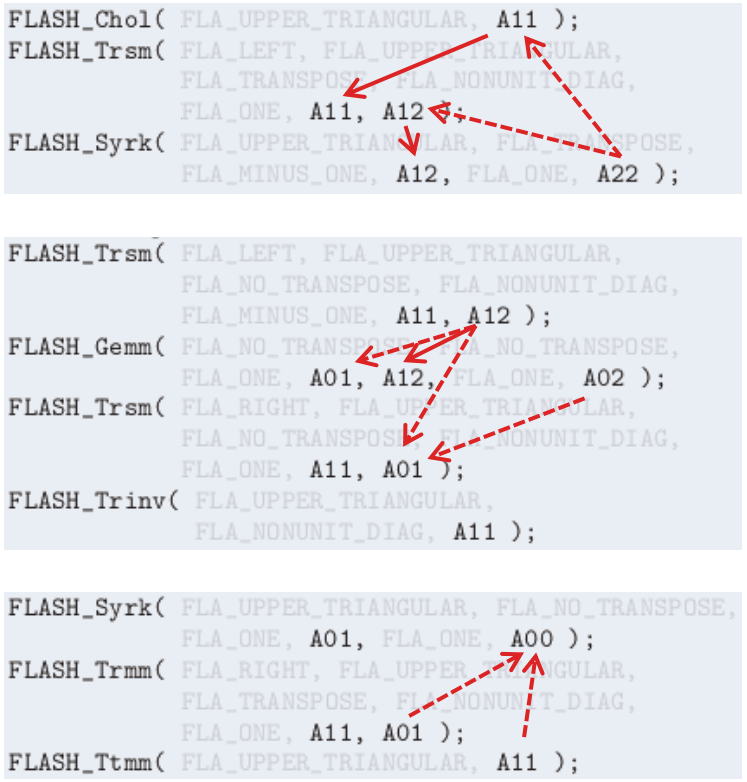
Source: Paper

Anti-dependencies

- Write-after-read dependency

S_3: A = B + C
S_4: C = B + E

S3 must complete the read before
S4 can write.



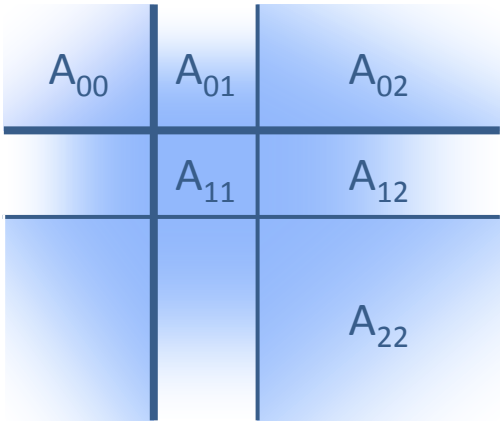
Source: Paper

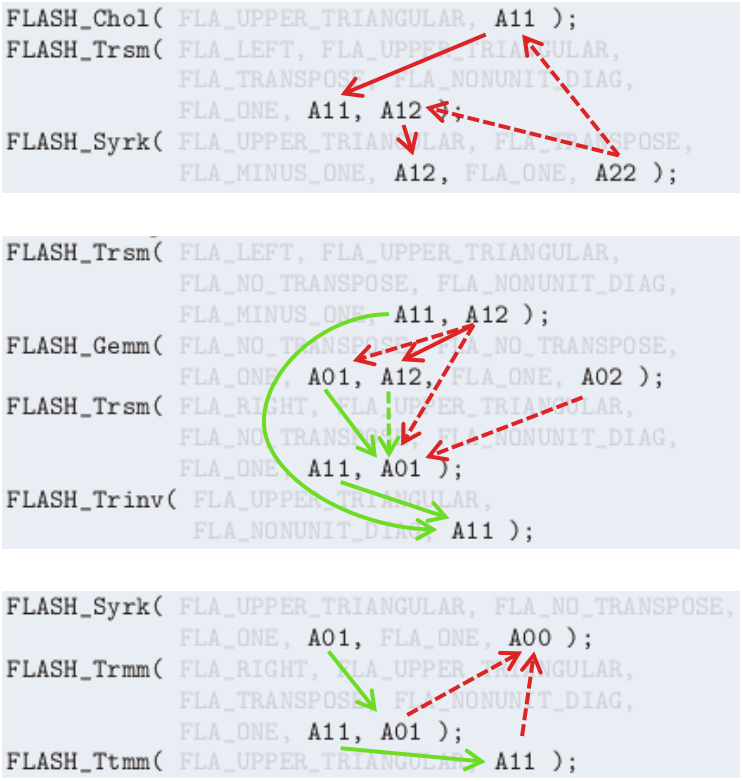
Anti-dependencies

- Write-after-read dependency

S_3: A = B + C
S_4: C = B + E

S3 must complete the read before
S4 can write.





Flow dependency
Anti-dependency
Intra-iterational
Inter-iterational

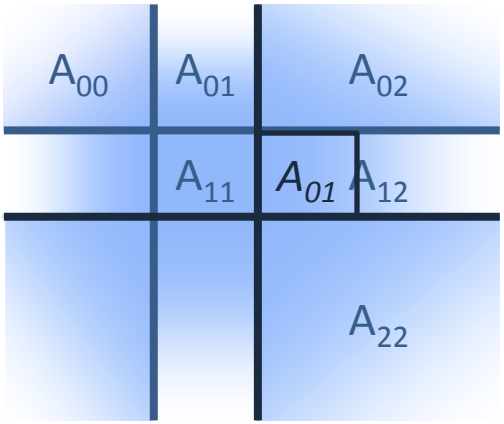
Source: Paper

Anti-dependencies

- Write-after-read dependency

S_3: A = B + C
S_4: C = B + E

S3 must complete the read before
S4 can write.




```
FLASH_Chol( FLA_UPPER_TRIANGULAR, A11 );
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A12 );
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_TRANSPOSE,
            FLA_MINUS_ONE, A12, FLA_ONE, A22 );
```

```
FLASH_Trsm( FLA_LEFT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_MINUS_ONE, A11, A12 );
FLASH_Gemm( FLA_NO_TRANSPOSE, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, A12, FLA_ONE, A02 );
FLASH_Trsm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_NO_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Trinv( FLA_UPPER_TRIANGULAR,
             FLA_NONUNIT_DIAG, A11 );
```

```
FLASH_Syrk( FLA_UPPER_TRIANGULAR, FLA_NO_TRANSPOSE,
            FLA_ONE, A01, FLA_ONE, A00 );
FLASH_Trmm( FLA_RIGHT, FLA_UPPER_TRIANGULAR,
            FLA_TRANSPOSE, FLA_NONUNIT_DIAG,
            FLA_ONE, A11, A01 );
FLASH_Ttmm( FLA_UPPER_TRIANGULAR, A11 );
```

→ Flow dependency
→ Anti-dependency
→ Intra-iterational
⇢ Inter-iterational

Source: Paper

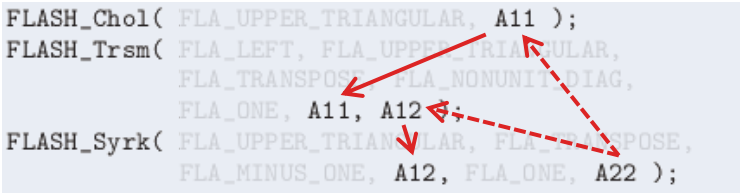
Output dependencies

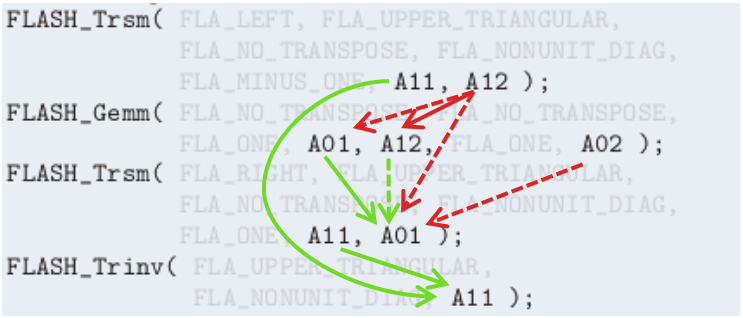
- Write-after-Write dependency

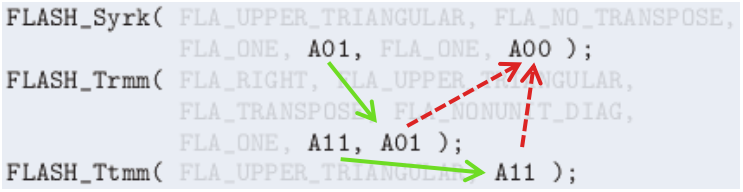
S_5: A = B + C
S_6: A = D + E

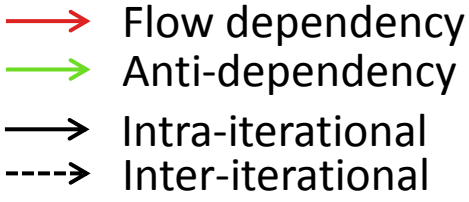
S6 must override A after S5.

- Last operand is also input operand, thus can be interpreted as flow dependency.



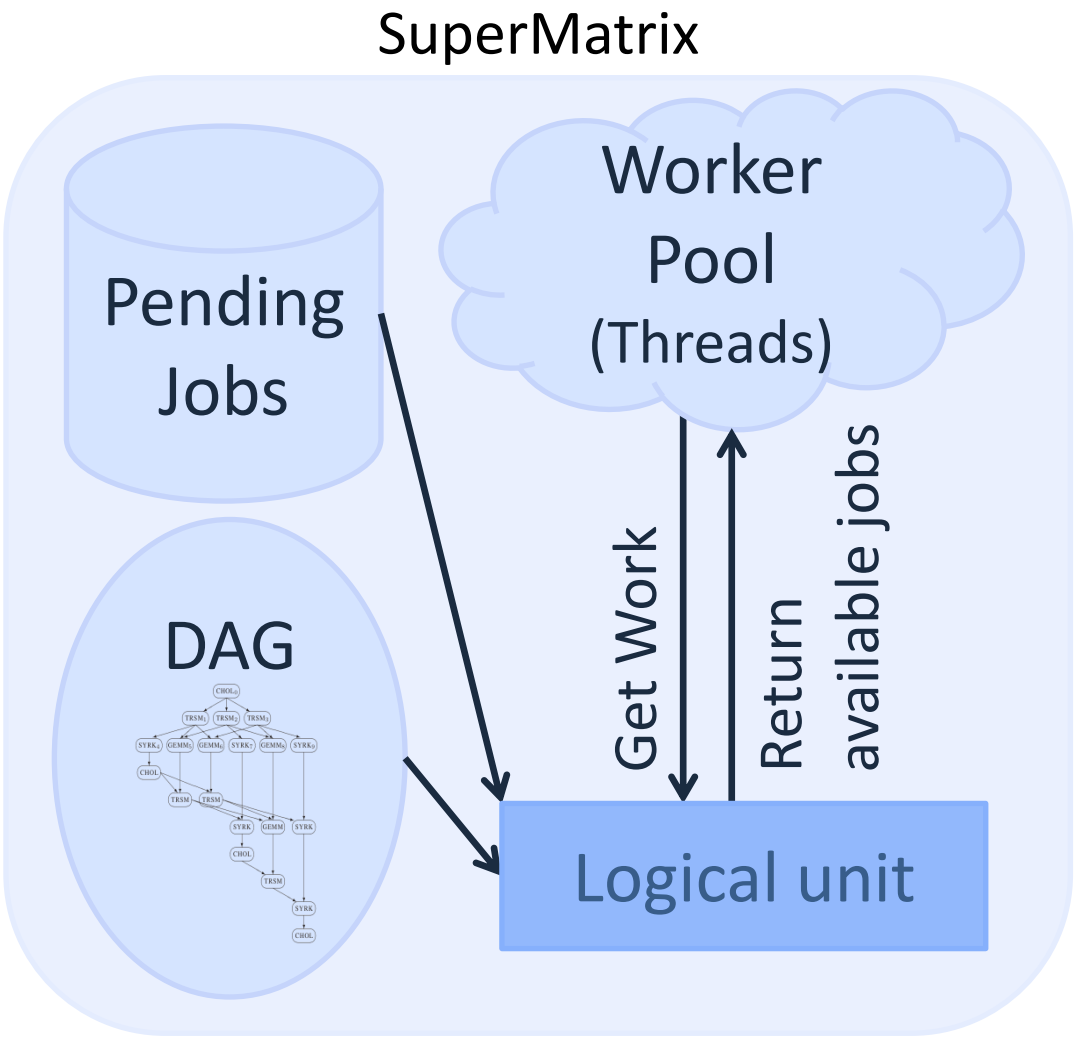




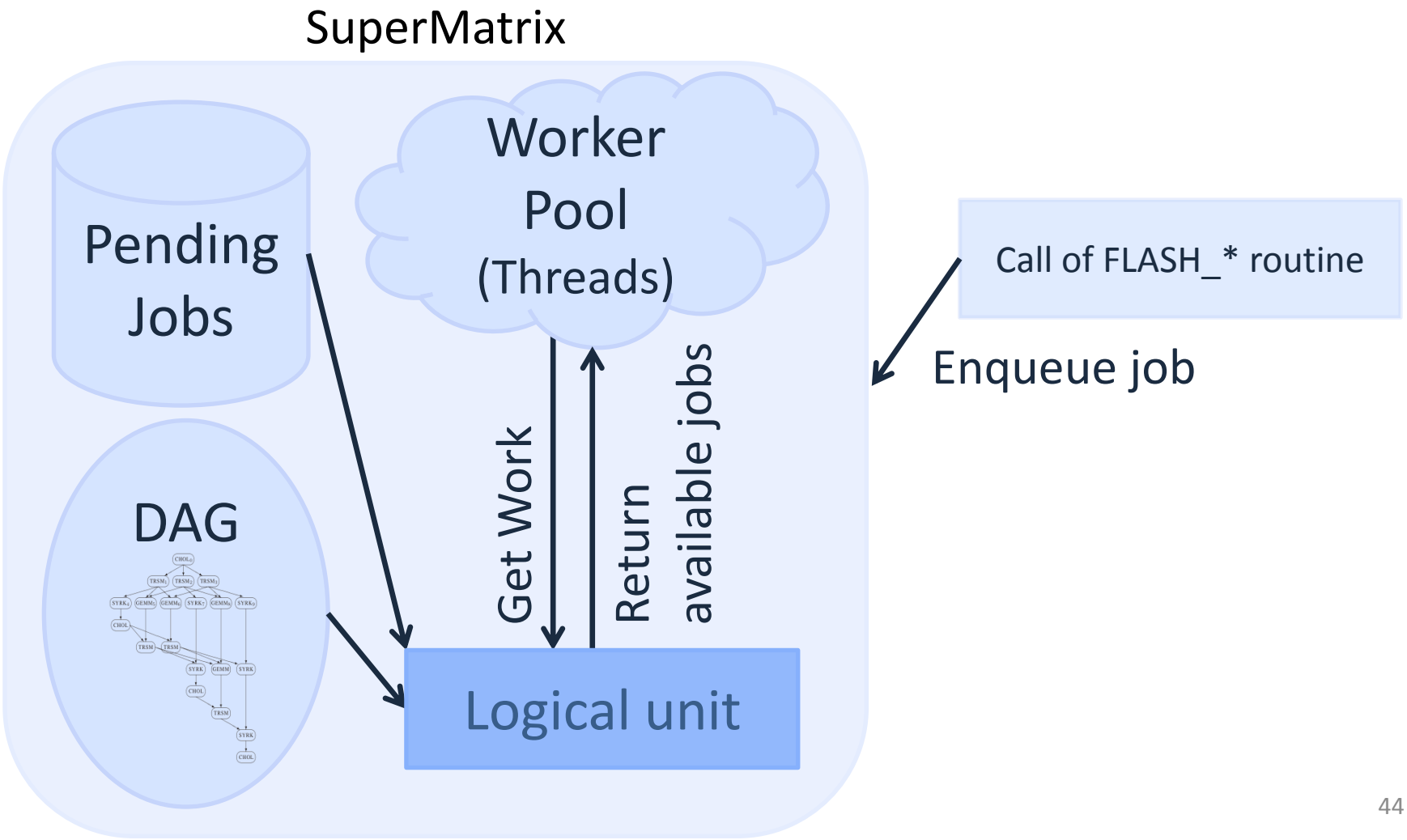


Source: Paper

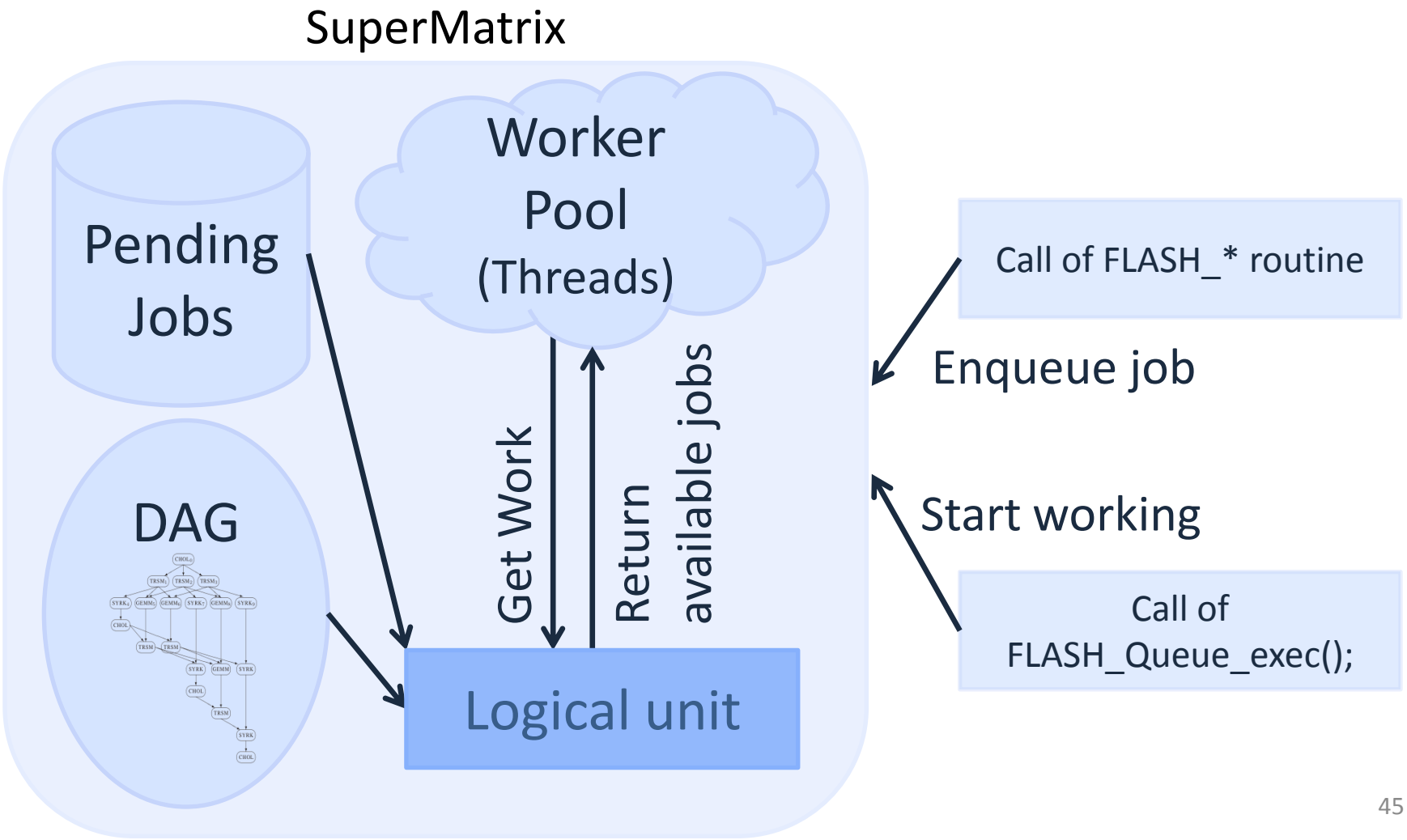
Workflow of SuperMatrix



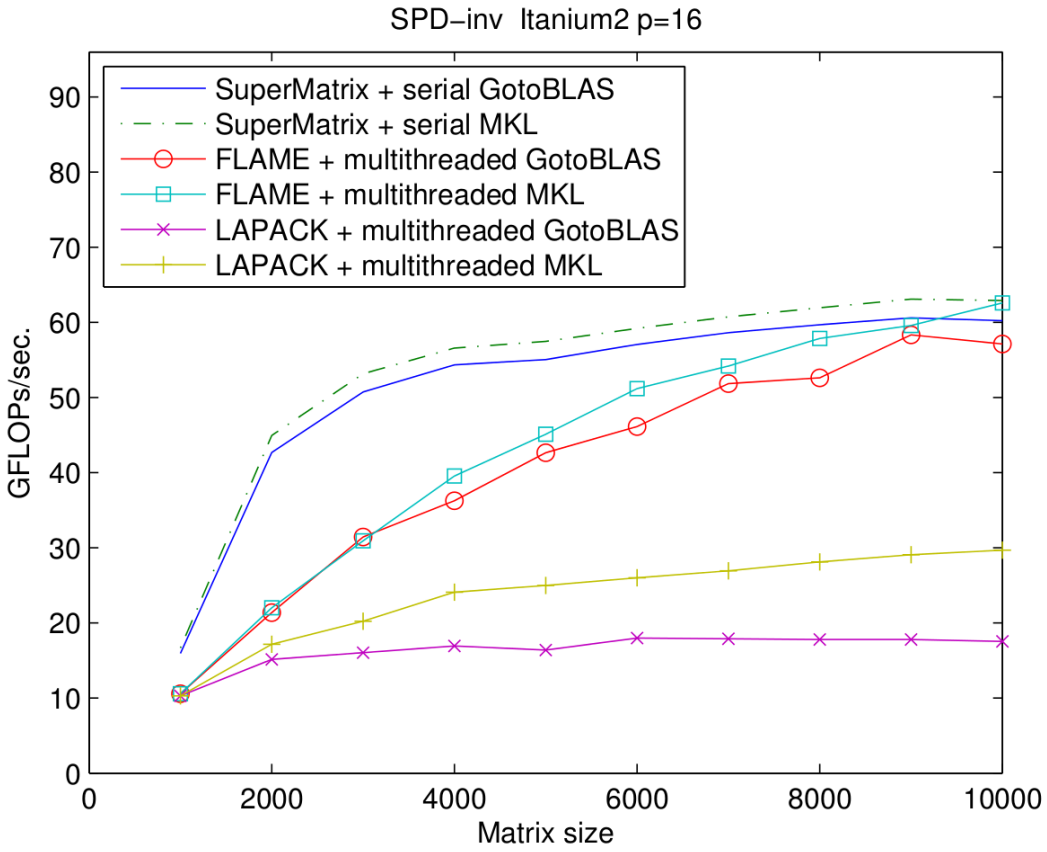
Workflow of SuperMatrix



Workflow of SuperMatrix



Performance of SPD-inv

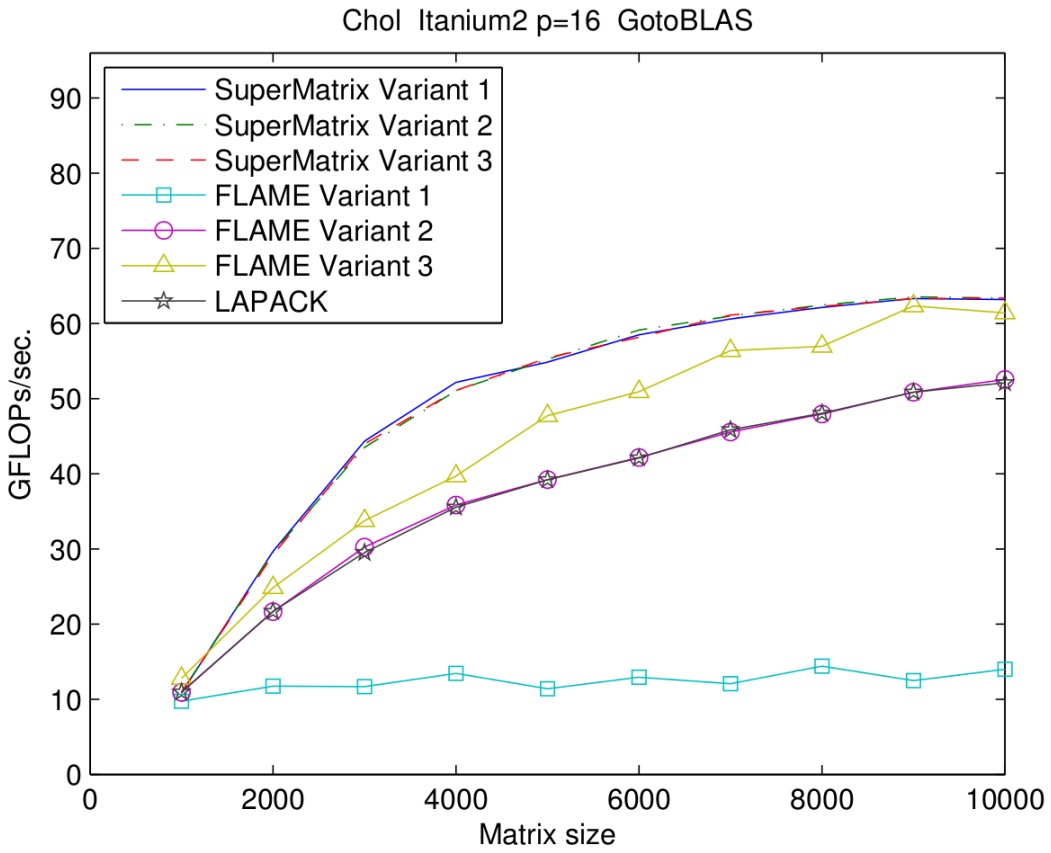


Architecture: Eight nodes, each with two 1.5 Ghz Intel Itanium2 processors. Peak performance is 96 GFLOPs/sec

Block size: 192

Source: Paper

Performance of (CHOL) variants



- SuperMatrix:
  Linked with serial GotoBLAS
- FLAME & LAPACK:
  Linked with multithreaded GotoBLAS

Architecture: Eight nodes, each with two 1.5 Ghz Intel Itanium2 processors. Peak performance is 96 GFLOPs/sec
Block size: 192

Source: Paper

Conclusion

- Easy to implement algorithms
- Less error prone
- Pretty good performance results
- Detailed instructions how to install

Further material

- Homepage of the FLAME project
  http://www.cs.utexas.edu/users/flame/

- Overview over FLASH, FLAME and SuperMatrix:
  High performance computing for computational science:8th international conference, Toulouse, France, June 24-27, 2008. revised selected papers
  Pages 228 – 239
  http://books.google.co.uk/books?id=us_EQmotveYC&lpg=PR3&pg=PR3#v=onepage&q&f=false

Thanks for your attention!

Questions?