Embedding Fault-Tolerance, Exploiting Approximate Computing and Retaining High Performance in the Matrix Multiplication

Tyler M. Smith
Robert A. van de Geijn

Enrique S. Quintana-Ortí

Mikhail Smelyanskiy

Motivation

UNIVERSITAT JAUME•I

- Matrix multiplication (GEMM)

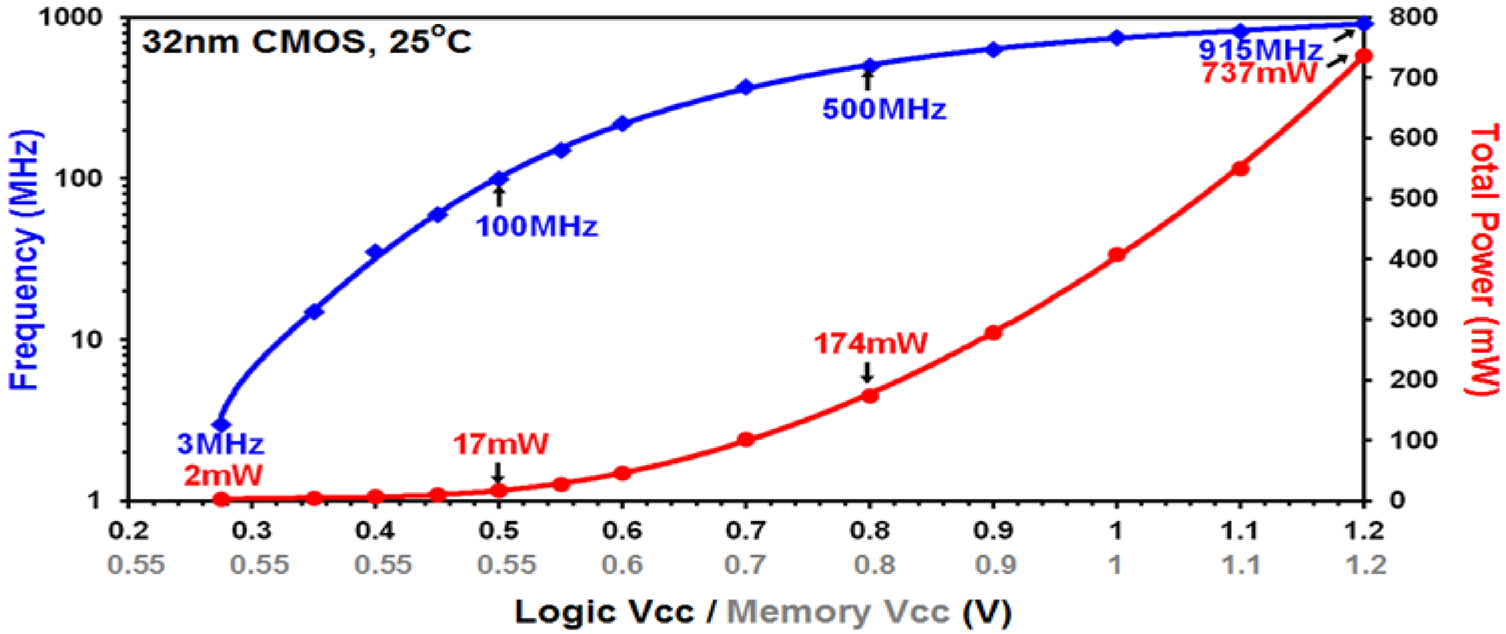
"Fault-tolerant high-performance matrix-matrix multiplication: theory and practice"
John A. Gunnels, Daniel S. Katz, Enrique S. Quintana, Robert van de Geijn
Int. Conference on Dependable Systems and Networks - DSN 2001

Provide a software layer for reliability in numerical libraries for spaceborne missions

NASA JPL
Jet Propulsion Laboratory
California Institute of Technology

Motivation

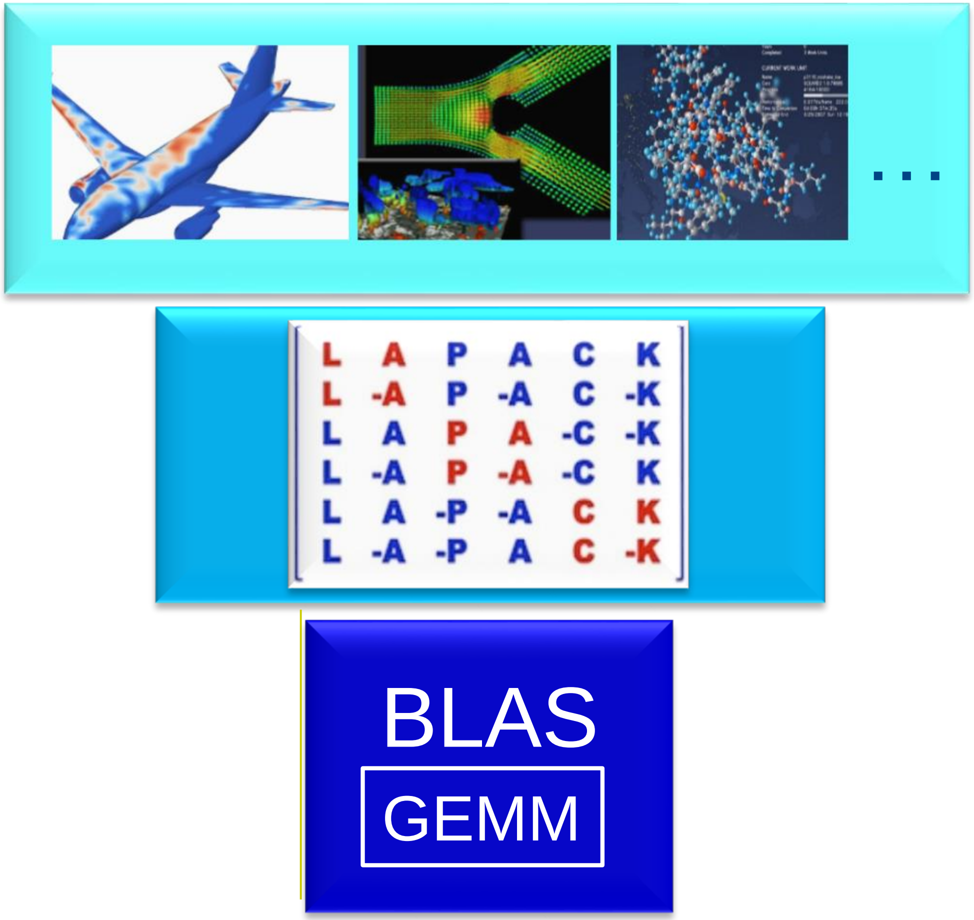
- Fault tolerance for GEMM, revisited
  - *Near-threshold voltage computing* (NTVC) reduces power…



at the cost of increasing error rates

Motivation

- Why GEMM?
  - Many scientific and engineering computations can be decomposed into a reduced number of linear algebra operations
  - Most dense linear algebra operations can be cast in terms of GEMM



Index

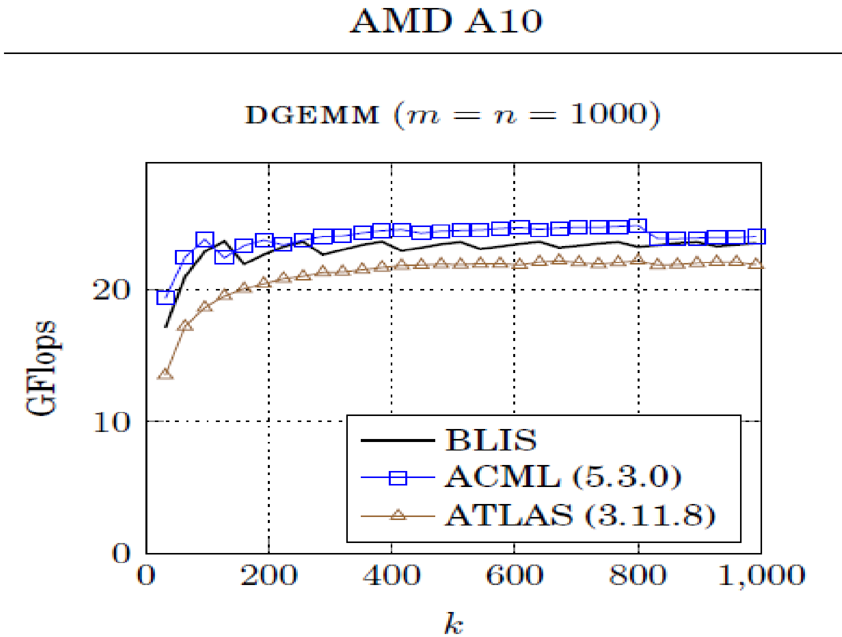
- High performance GEMM
- Fault tolerance (FT) vs approx. computing (AC)
- Embedding FT/AC in high performance GEMM
- Experimental results
- Concluding remarks

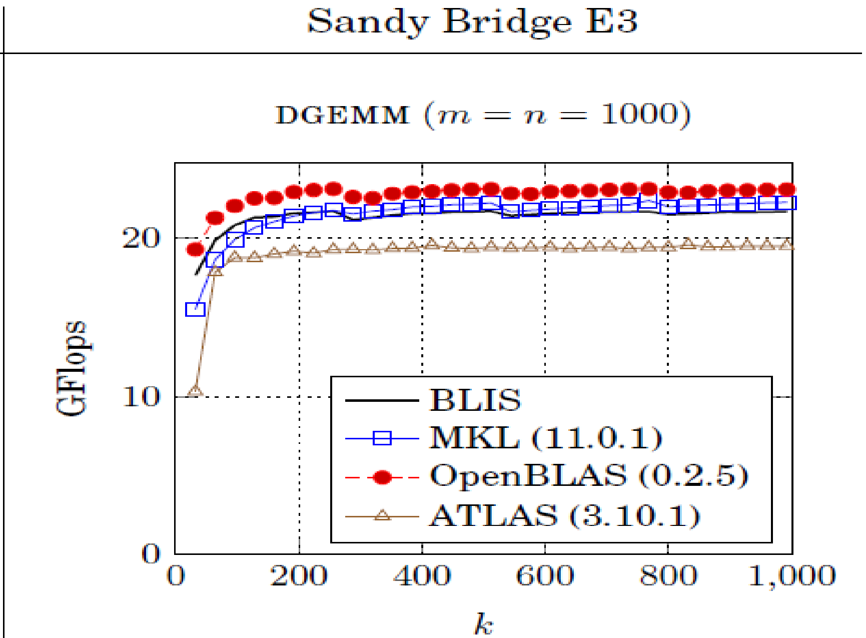
High Performance GEMM

- Commercial libraries for BLAS

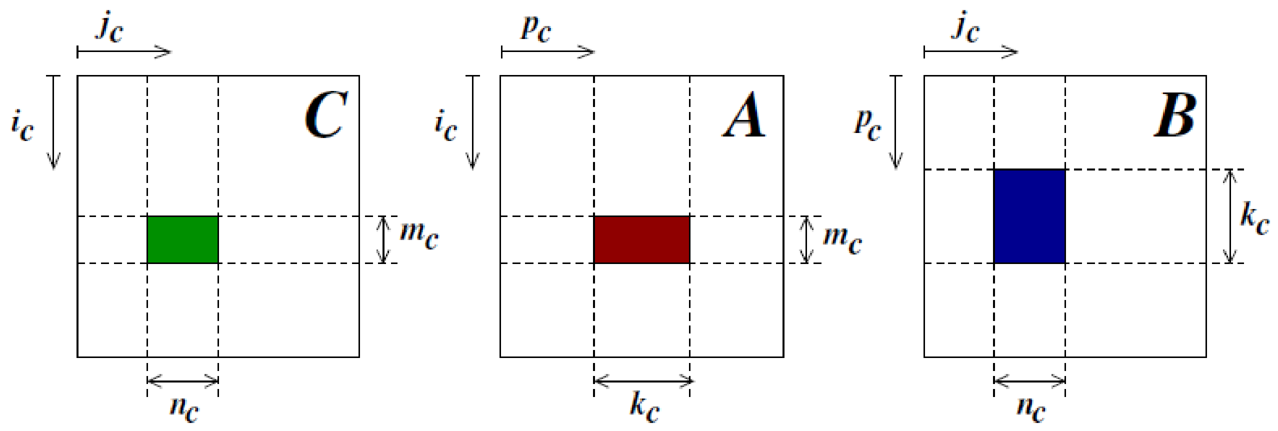
MKL ACML ESSL

- “Open” sw.: GotoBLAS, ATLAS, OpenBLAS, BLIS





High Performance GEMM
BLIS



for $j_c = 0, \ldots, n-1$ in steps of n_c
  for $p_c = 0, \ldots, k-1$ in steps of k_c
    $B(p_c : p_c + k_c - 1, j_c : j_c + n_c - 1) \rightarrow B_c$ // Pack into B_c
    for $i_c = 0, \ldots, m-1$ in steps of m_c
      $A(i_c : i_c + m_c - 1, p_c : p_c + k_c - 1) \rightarrow A_c$ // Pack into A_c

---

      for $j_r = 0, \ldots, n_c - 1$ in steps of n_r // Macro-kernel
        for $i_r = 0, \ldots, m_c - 1$ in steps of m_r

---

          for $p_r = 0, \ldots, k_c - 1$ in steps of 1 // Micro-kernel
            $C_c(i_r : i_r + m_r - 1, j_r : j_r + n_r - 1)$
              $\mathrel{+}= A_c(i_r : i_r + m_r - 1, p_r)$
              $\cdot\ B_c(p_r, j_r : j_r + n_r - 1)$
          endfor

---

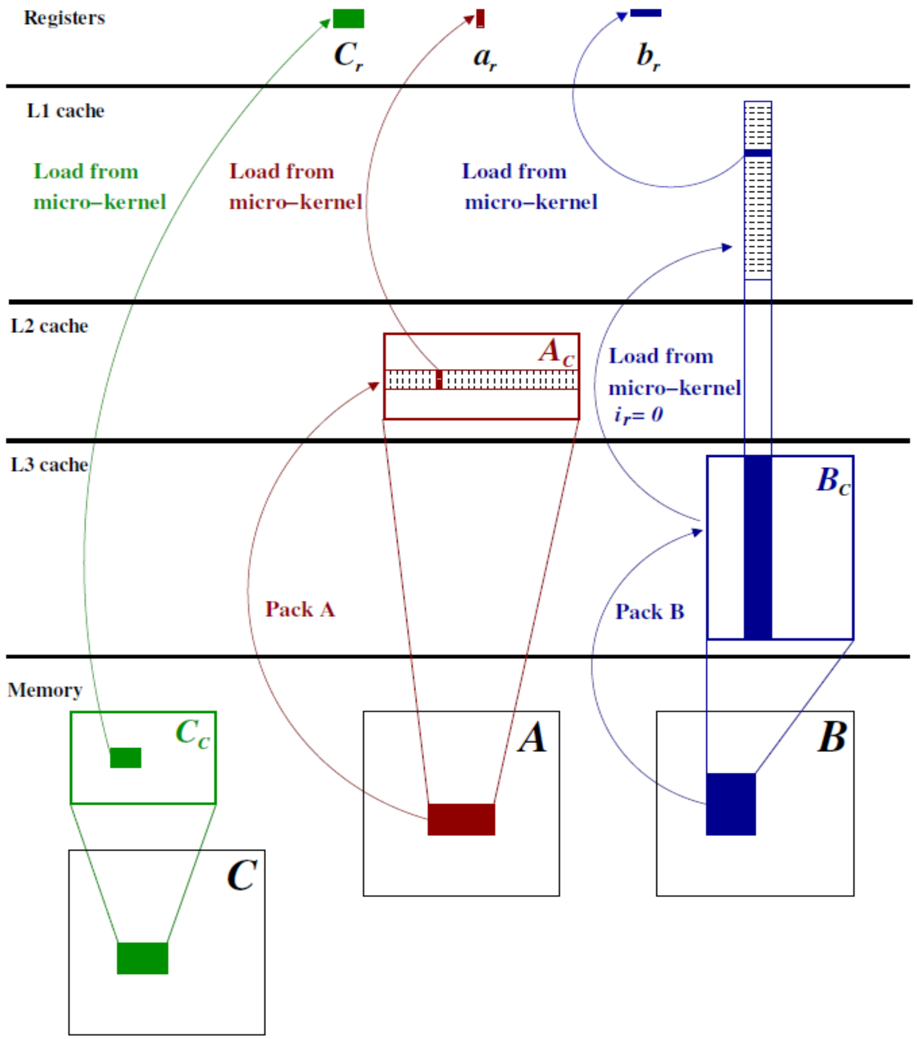
        endfor
      endfor

---

    endfor
  endfor
endfor

High Performance GEMM BLIS

for $j_c = 0, \ldots, n-1$ in steps of n_c
  for $p_c = 0, \ldots, k-1$ in steps of k_c
    $B(p_c : p_c + k_c - 1, j_c : j_c + n_c - 1) \rightarrow B_c$
    for $i_c = 0, \ldots, m-1$ in steps of m_c
      $A(i_c : i_c + m_c - 1, p_c : p_c + k_c - 1) \rightarrow A_c$
      for $j_r = 0, \ldots, n_c - 1$ in steps of n_r
        for $i_r = 0, \ldots, m_c - 1$ in steps of m_r
          for $p_r = 0, \ldots, k_c - 1$ in steps of 1
            $C_c(i_r : i_r + m_r - 1, j_r : j_r + n_r - 1)$
              $\mathrel{+}= A_c(i_r : i_r + m_r - 1, p_r)$
              $\cdot\ B_c(p_r, j_r : j_r + n_r - 1)$
          endfor
        endfor
      endfor
    endfor
  endfor
endfor



FT/AC in GEMM

- Consider $C = A\ B$, and the augmented matrices

$$A^* = \left(\frac{A}{v^T A} \right), \quad B^* = \left(\begin{array}{c|c} B & Bw \end{array} \right), \quad C^* = \left(\begin{array}{c|c} C & Cw \\ \hline v^T C & v^T Cw \end{array} \right),$$

In absence of error, then $C^* = A^* B^*$.

Use *left and right checksum* vectors:

$$\begin{array}{rclcl} \|d\|_\infty & = & \|C \cdot w & - & A \cdot (B \cdot w)\|_\infty & > & 0 & \text{or} \\ \|e^T\|_\infty & = & \|v^T \cdot C & - & (v^T \cdot A) \cdot B\|_\infty & > & 0. \end{array}$$

"Algorithm-based fault tolerance for matrix operations"
K.-H. Huang and J. A. Abraham
IEEE Transactions on Computers, vol. 33, no. 6, pp. 518–528, 1984.

FT/AC in GEMM

- In practice, due to finite precision arithmetic, an error is detected if

$$\begin{aligned} \|d\|_\infty &> \tau \cdot \|A\|_\infty \cdot \|B\|_\infty \\ \|e^T\|_\infty &> \tau \cdot \|A\|_\infty \cdot \|B\|_\infty, \end{aligned}$$

where $\tau = \max(m, n, k) \cdot u$ for FT

"Fault-tolerant high-performance matrix-matrix multiplication: theory and practice"
John A. Gunnels, Daniel S. Katz, Enrique S. Quintana, Robert van de Geijn
Int. Conference on Dependable Systems and Networks - DSN 2001

or higher for AC!

FT/AC in GEMM

- Overhead for full GEMM (detection only)

$$
\begin{aligned}
\|d\|_\infty &= \|C \cdot w &- \quad & A \cdot (B \cdot w)\|_\infty \\
\|e^T\|_\infty &= \|v^T \cdot C &- \quad & (v^T \cdot A) \cdot B\|_\infty
\end{aligned}
$$

$$
\mathcal{O}_d(m,n,k) = \frac{4mn + 5mk + 5kn}{\mathcal{O}_c(m,n,k)} = \frac{4mn + 5mk + 5kn}{2mnk},
$$

  - Has to be applied off-line
  - Requires a copy of the full matrix C
  - Correction is expensive: recompute the full product

FT/AC in GEMM

- Apply with smaller granularity

for $j_c = 0, \ldots, n-1$ in steps of n_c
    for $p_c = 0, \ldots, k-1$ in steps of k_c
        $B(p_c : p_c + k_c - 1, j_c : j_c + n_c - 1) \rightarrow B_c$
        for $i_c = 0, \ldots, m-1$ in steps of m_c
            $A(i_c : i_c + m_c - 1, p_c : p_c + k_c - 1) \rightarrow A_c$
            for $j_r = 0, \ldots, n_c - 1$ in steps of n_r
                for $i_r = 0, \ldots, m_c - 1$ in steps of m_r
                    for $p_r = 0, \ldots, k_c - 1$ in steps of 1
                        $C_c(i_r : i_r + m_r - 1, j_r : j_r + n_r - 1)$
                            $+= A_c(i_r : i_r + m_r - 1, p_r)$
                            $\cdot\ B_c(p_r, j_r : j_r + n_r - 1)$
                    endfor
                endfor
            endfor
        endfor
    endfor
endfor

Loop index	Required workspace	$\mathcal{O}_d$ and $\mathcal{O}_c$ depend on
j_c	$m \times n_c$	(m, n_c, k)
p_c	$m \times n_c$	(m, n_c, k_c)
i_c	$m_c \times n_c$	(m_c, n_c, k_c)
j_r	$m_c \times n_r$	(m_c, n_r, k_c)
i_r	$m_r \times n_r$	(m_r, n_r, k_c)
k_r	$m_r \times n_r$	$(m_r, n_r, 1)$

FT/AC in GEMM

- Intel Xeon E5 (Sandy-Bridge): Macro-kernel

for $j_c = 0, \ldots, n-1$ in steps of n_c
  for $p_c = 0, \ldots, k-1$ in steps of k_c
    $B(p_c : p_c + k_c - 1, j_c : j_c + n_c - 1) \rightarrow B_c$
    for $i_c = 0, \ldots, m-1$ in steps of m_c
      $A(i_c : i_c + m_c - 1, p_c : p_c + k_c - 1) \rightarrow A_c$
      for $j_r = 0, \ldots, n_c - 1$ in steps of n_r
        for $i_r = 0, \ldots, m_c - 1$ in steps of m_r
          for $p_r = 0, \ldots, k_c - 1$ in steps of 1
            $C_c(i_r : i_r + m_r - 1, j_r : j_r + n_r - 1)$
              $\mathrel{+}= A_c(i_r : i_r + m_r - 1, p_r)$
              $\cdot\ B_c(p_r, j_r : j_r + n_r - 1)$
          endfor
        endfor
      endfor
    endfor
  endfor
endfor

Loop index	Required workspace	$\mathcal{O}_d$ and $\mathcal{O}_c$ depend on
j_c	$m \times n_c$	(m, n_c, k)
p_c	$m \times n_c$	(m, n_c, k_c)
i_c	$m_c \times n_c$	(m_c, n_c, k_c)
j_r	$m_c \times n_r$	(m_c, n_r, k_c)
i_r	$m_r \times n_r$	(m_r, n_r, k_c)
k_r	$m_r \times n_r$	$(m_r, n_r, 1)$

Workspace: 96 x 4,096 numbers
Overhead for error detection: 2.6%

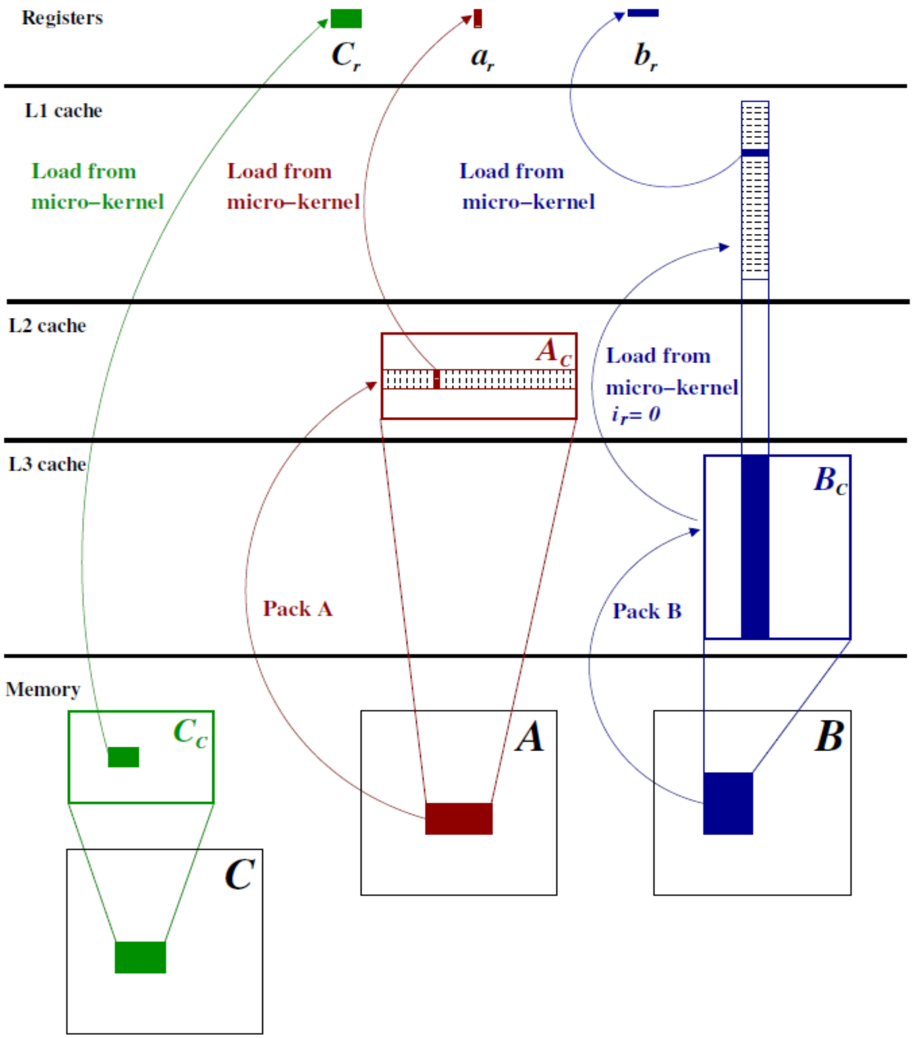
High Performance GEMM
BLIS

$$
\begin{array}{l}
\text{for } j_c = 0, \ldots, n-1 \text{ in steps of } n_c \\
\quad \text{for } p_c = 0, \ldots, k-1 \text{ in steps of } k_c \\
\qquad B(p_c : p_c + k_c - 1, j_c : j_c + n_c - 1) \rightarrow B_c \\
\qquad \text{for } i_c = 0, \ldots, m-1 \text{ in steps of } m_c \\
\qquad\quad A(i_c : i_c + m_c - 1, p_c : p_c + k_c - 1) \rightarrow A_c \\
\hline
\qquad\quad \text{for } j_r = 0, \ldots, n_c - 1 \text{ in steps of } n_r \\
\qquad\qquad \text{for } i_r = 0, \ldots, m_c - 1 \text{ in steps of } m_r \\
\hline
\qquad\qquad\quad \text{for } p_r = 0, \ldots, k_c - 1 \text{ in steps of } 1 \\
\qquad\qquad\qquad C_c(i_r : i_r + m_r - 1, j_r : j_r + n_r - 1) \\
\qquad\qquad\qquad\quad \mathrel{+}= A_c(i_r : i_r + m_r - 1, p_r) \\
\qquad\qquad\qquad\quad \cdot \; B_c(p_r, j_r : j_r + n_r - 1) \\
\qquad\qquad\quad \text{endfor} \\
\hline
\qquad\qquad \text{endfor} \\
\qquad\quad \text{endfor} \\
\hline
\qquad \text{endfor} \\
\quad \text{endfor} \\
\text{endfor}
\end{array}
$$

$$
\begin{array}{rcl}
\|d\|_\infty & = & \|C \cdot w \; - \; A \cdot (B \cdot w)\|_\infty \\
\|e^T\|_\infty & = & \|v^T \cdot C \; - \; (v^T \cdot A) \cdot B\|_\infty
\end{array}
$$

with $C = C_c, A = A_c, B = B_c$
(macro-kernel)

High Performance GEMM
BLIS



$$
\begin{aligned}
\|d\|_\infty &= \|C \cdot w &- A \cdot (B \cdot w)\|_\infty \\
\|e^T\|_\infty &= \|v^T \cdot C &- (v^T \cdot A) \cdot B\|_\infty
\end{aligned}
$$

with $C = C_c, A = A_c, B = B_c$ (macro-kernel)

FT/AC in GEMM

- Left checksum: $d = \hat{C}_c \cdot w - A_c \cdot B_c \cdot w$

for $j_c = 0, \ldots, n-1$ **in steps of** n_c, $\mathcal{J}_c = j_c : j_c + n_c - 1$
  for $p_c = 0, \ldots, k-1$ **in steps of** k_c, $\mathcal{P}_c = p_c : p_c + k_c - 1$
    $B(\mathcal{P}_c, \mathcal{J}_c) \rightarrow B_c$
    $d_b = -B_c \cdot w$
    for $i_c = 0, \ldots, m-1$ **in steps of** m_c, $\mathcal{I}_c = i_c : i_c + m_c - 1$
      $A(\mathcal{I}_c, \mathcal{P}_c) \rightarrow A_c$
      $d = A_c \cdot d_b \ (= A_c \cdot B_c \cdot d_b)$
      $e_a^T = -v^T \cdot A_c$

      for $j_r = 0, \ldots, n_c - 1$ **in steps of** n_r, $\mathcal{J}_r = j_r : j_r + n_r - 1$
        $e^T(\mathcal{J}_r) = e_a^T \cdot B_c(0 : k_c - 1, \mathcal{J}_r)$

        for $i_r = 0, \ldots, m_c - 1$ **in steps of** m_r, $\mathcal{I}_r = i_r : i_r + m_r - 1$

          $\hat{C}_c(\mathcal{I}_r, \mathcal{J}_r) = A_c(\mathcal{I}_r, 0 : k_c - 1) \cdot B_c(0 : k_c - 1, \mathcal{J}_r)$

          $d(\mathcal{I}_r) \mathrel{+}= \hat{C}_c(\mathcal{I}_r, \mathcal{J}_r) \cdot w(\mathcal{J}_r)$
          $e^T(\mathcal{J}_r) \mathrel{+}= v^T(\mathcal{I}_r) \cdot \hat{C}_c(\mathcal{I}_r, \mathcal{J}_r)$
        endfor
      endfor

      if $(\|d\|_\infty > \tau \|A\|_\infty \|B\|_\infty)$ **or** $(\|e_c^T\|_\infty > \tau \|A\|_\infty \|B\|_\infty)$
        recompute macro-kernel
      else
        $C(\mathcal{I}_c, \mathcal{J}_c) += \hat{C}_c$
      endif
    endfor
  endfor
endfor

FT/AC in GEMM

- Right checksum: $e^T = v^T \cdot \hat{C}_c - v^T \cdot A_c \cdot B_c$

```
for j_c = 0,...,n-1 in steps of n_c, J_c = j_c : j_c + n_c - 1
  for p_c = 0,...,k-1 in steps of k_c, P_c = p_c : p_c + k_c - 1
    B(P_c, J_c) → B_c
    d_b = -B_c · w
    for i_c = 0,...,m-1 in steps of m_c, I_c = i_c : i_c + m_c - 1
      A(I_c, P_c) → A_c
      d = A_c · d_b (= A_c · B_c · d_b)
      e_a^T = -v^T · A_c
      for j_r = 0,...,n_c-1 in steps of n_r, J_r = j_r : j_r + n_r - 1
        e^T(J_r) = e_a^T · B_c(0 : k_c-1, J_r)

        for i_r = 0,...,m_c-1 in steps of m_r, I_r = i_r : i_r + m_r - 1
          Ĉ_c(I_r, J_r) = A_c(I_r, 0 : k_c-1) · B_c(0 : k_c-1, J_r)
          d(I_r) += Ĉ_c(I_r, J_r) · w(J_r)
          e^T(J_r) += v^T(I_r) · Ĉ_c(I_r, J_r)
        endfor
      endfor
      if (||d||_∞ > τ||A||_∞||B||_∞) or (||e_c^T||_∞ > τ||A||_∞||B||_∞)
        recompute macro-kernel
      else
        C(I_c, J_c) += Ĉ_c
      endif
    endfor
  endfor
endfor
```

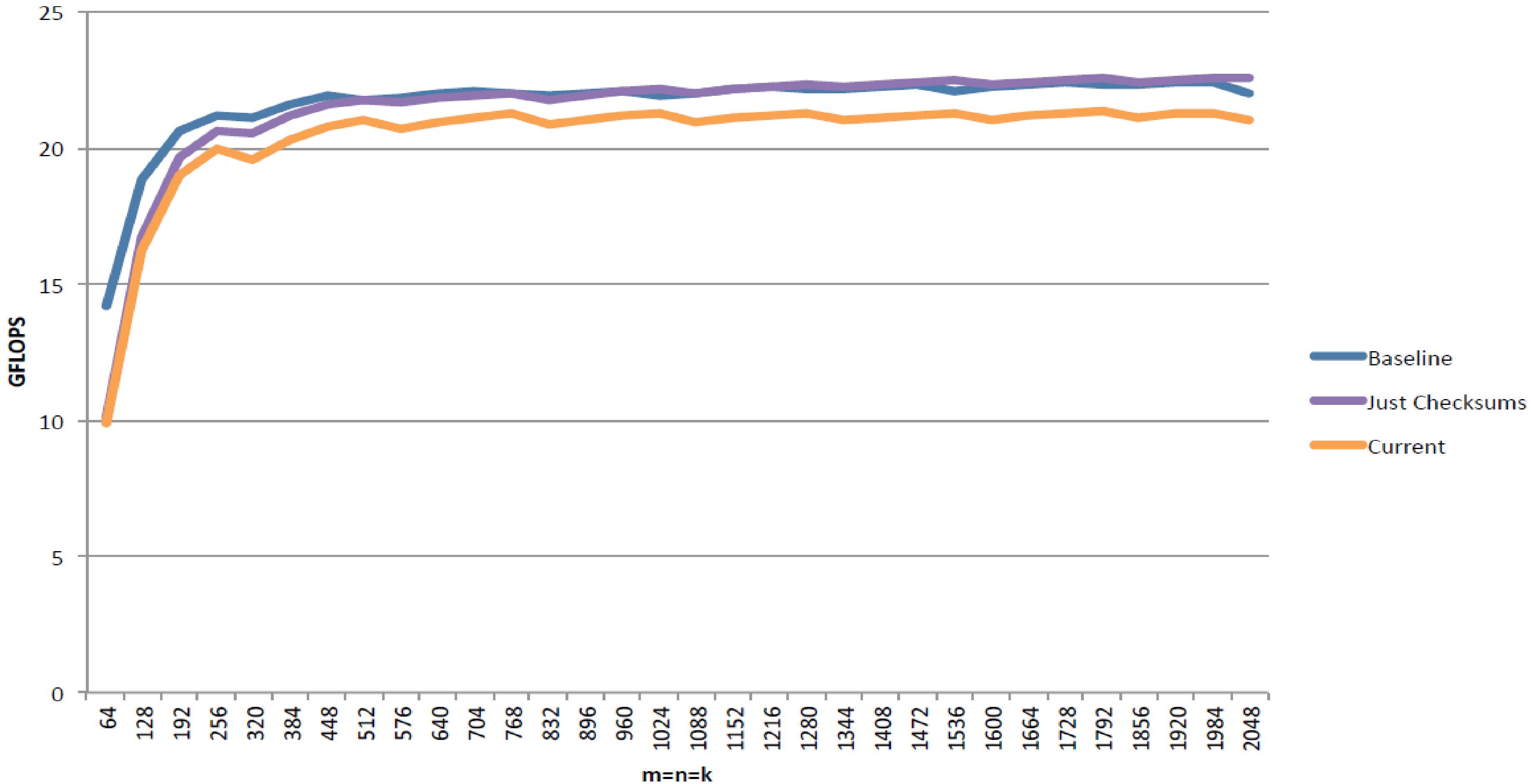
FT/AC in GEMM

- Detect and prevent error: Check $\|d\|_\infty$ and $\|e^T\|_\infty$

$$
\begin{array}{l}
\textbf{for } j_c = 0,\ldots,n-1 \textbf{ in steps of } n_c,\ \mathcal{J}_c = j_c : j_c + n_c - 1 \\
\quad \textbf{for } p_c = 0,\ldots,k-1 \textbf{ in steps of } k_c,\ \mathcal{P}_c = p_c : p_c + k_c - 1 \\
\quad\quad B(\mathcal{P}_c, \mathcal{J}_c) \rightarrow B_c \\
\quad\quad d_b = -B_c \cdot w \\
\quad\quad \textbf{for } i_c = 0,\ldots,m-1 \textbf{ in steps of } m_c,\ \mathcal{I}_c = i_c : i_c + m_c - 1 \\
\quad\quad\quad A(\mathcal{I}_c, \mathcal{P}_c) \rightarrow A_c \\
\quad\quad\quad d = A_c \cdot d_b \ (= A_c \cdot B_c \cdot d_b) \\
\quad\quad\quad e_a^T = -v^T \cdot A_c \\
\hline
\quad\quad\quad \textbf{for } j_r = 0,\ldots,n_c-1 \textbf{ in steps of } n_r,\ \mathcal{J}_r = j_r : j_r + n_r - 1 \\
\quad\quad\quad\quad e^T(\mathcal{J}_r) = e_a^T \cdot B_c(0 : k_c - 1, \mathcal{J}_r) \\
\\
\quad\quad\quad\quad \textbf{for } i_r = 0,\ldots,m_c-1 \textbf{ in steps of } m_r,\ \mathcal{I}_r = i_r : i_r + m_r - 1 \\
\hline
\quad\quad\quad\quad\quad \hat{C}_c(\mathcal{I}_r, \mathcal{J}_r) = A_c(\mathcal{I}_r, 0 : k_c - 1) \cdot B_c(0 : k_c - 1, \mathcal{J}_r) \\
\hline
\quad\quad\quad\quad\quad d(\mathcal{I}_r) \mathrel{+}= \hat{C}_c(\mathcal{I}_r, \mathcal{J}_r) \cdot w(\mathcal{J}_r) \\
\quad\quad\quad\quad\quad e^T(\mathcal{J}_r) \mathrel{+}= v^T(\mathcal{I}_r) \cdot \hat{C}_c(\mathcal{I}_r, \mathcal{J}_r) \\
\quad\quad\quad\quad \textbf{endfor} \\
\quad\quad\quad \textbf{endfor} \\
\hline
\quad\quad\quad \textbf{if } (\|d\|_\infty > \tau \|A\|_\infty \|B\|_\infty) \textbf{ or } (\|e_c^T\|_\infty > \tau \|A\|_\infty \|B\|_\infty) \\
\quad\quad\quad\quad \text{recompute macro-kernel} \\
\quad\quad\quad \textbf{else} \\
\quad\quad\quad\quad C(\mathcal{I}_c, \mathcal{J}_c) + = \hat{C}_c \\
\quad\quad\quad \textbf{endif} \\
\quad\quad \textbf{endfor} \\
\quad \textbf{endfor} \\
\textbf{endfor}
\end{array}
$$

FT/AC in GEMM

- Intel Xeon E5-2680. BLIS (baseline) vs current FT-BLIS



FT/AC in GEMM

- Selective error correction
  Detection at the macro-kernel level:

$$4m_c n_c + 5m_c k_c + 5k_c n_c \qquad \mathcal{O}_d(m_c, n_c, k_c)$$

  but correction can proceed at the micro-kernel level:

$$2m_r n_r k_c \qquad \mathcal{O}_c(m_r, n_r, k_c)$$

  instead of

$$2m_c n_c k_c \qquad \mathcal{O}_c(m_c, n_c, k_c)$$

Concluding Remarks

- Easy to integrate FT and AC into the same framework for BLIS
- Left and right checksums yield acceptable overhead for high performance GEMM
- Much work to be done to turn it practical:
  - Multi-threaded GEMM