Stream Algorithms and Architecture

Henry Hoffmann, Volker Strumpen, Anant Agarwal

MIT

hank@mit.edu

joint work with the Raw Team

Special Thanks to Janice McMahon and Lincoln Lab’s Group 102
The Parallel Processing Dilemma

Common sense about the scalability of parallel processing: efficiency decreases as the number of processors increases.

\[ S = P \]  
\[ \text{(ideal)} \]

\[ S < P \]  
\[ \text{(real)} \]
The actual compute efficiency is even worse when measured in Flops, relative to the peak Flops of a particular machine.
The efficiency of Stream Algorithms increases while adding processors.

Floating-Point Efficiency
\[ E = \frac{\text{Total FLOPs}(P)}{\text{Total Cycles}(P)} \]
How is that possible?

Move load’s and store’s off critical path

1. **Systolic algorithms** [Kung and Leiserson, 1978]
   (a) provide solution when $\text{Problem Size}(N) = \text{Network Size}(R)$
   (b) for $N > R$ simulation common, but inefficient

2. **Decoupled Access Execute Architectures** [Jim Smith, 1982]
   (a) separate memory accesses from computation

3. **Out of Core Algorithms** [Sivan Toledo, 1999]
   (a) work on large problems with limited space

Stream Architecture and Algorithms handle $N > R$
Holistic approach achieves 100% floating-point efficiency
Outline

1. Holistic Approach
   (a) Stream Architecture
   (b) Stream Algorithms

2. Example: Matrix Multiplication

3. General Definition of Stream Algorithms

4. Conclusion
The Stream Architecture

1. **Network**: Register-mapped FIFOs, pipeline-integrated, near-neighbor, programmable

2. **Compute Processors (P)**: Registers, FPU, Control
   FPU has multiply-and-add unit

3. **Memory Processors (M)**: P with local memory

Raw can easily implement the stream architecture!
Stream Algorithms

1. Eliminate load’s and store’s from critical path using register mapped networks and memory tiles
   (a) replace memory ops with network communication

2. Partition computation into subproblems

3. Use systolic design for computation

4. Decouple memory accesses from computation

5. Use $M$ procs for memory ops and $P$ procs for flops
   (a) $M$ is asymptotically smaller than $P$: $M = o(P)$. 
Example: Matrix Multiplication, $C = AB$

$C, A, B$ all $N \times N$

$R \times R$ array of compute processors

1. **Partition** into $(N/R)^2$ problems by recursively applying:

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

2. Use **systolic** matrix multiplication [Leighton, 1992]

   (a) $P(R) = R^2$ compute processors

3. **Decouple** memory access - store $A_{ij}, B_{kl}$ until necessary

   (a) $R$ processors store rows of $A$
   (b) $R$ processors store columns of $B$
   (c) $M(R) = 2R$ memory processors
Decoupled Matrix Multiplication

Memory tiles implement the data access to stream the rows and columns into the systolic array of compute tiles.

Number of memory tiles: $M = 2R$
Number of compute tiles: $P = R^2$  \[ \Rightarrow M = o(P) \]

$M = o(P)$ is necessary, but not sufficient condition for 100% efficiency.
Decoupled Systolic Matrix Multiplication

1. \( \begin{array}{cc}
A(0,:) & B(:,0) \\
A(2,:) & B(:,0) \\
A(1,:) & B(:,2) \\
A(3,:) & B(:,3)
\end{array} \)

2. \( \begin{array}{cc}
A(0,:) & B(:,0) \\
A(2,:) & a_{00} \\
A(1,:) & b_{00} \\
A(3,:) & b_{10}
\end{array} \)

3. \( \begin{array}{cc}
A(0,:) & B(:,0) \\
A(2,:) & a_{01} \\
A(1,:) & b_{00} \\
A(3,:) & a_{10}
\end{array} \)

4. \( \begin{array}{cc}
A(0,:) & B(:,0) \\
A(2,:) & a_{02} \\
A(1,:) & b_{10} \\
A(3,:) & b_{01}
\end{array} \)

5. \( \begin{array}{cc}
A(0,:) & b_{30} \\
A(2,:) & a_{03} \\
A(1,:) & b_{20} \\
A(3,:) & a_{12}
\end{array} \) \( c_{00} \)

6. \( \begin{array}{cc}
A(0,:) & b_{02} \\
A(2,:) & a_{03} \\
A(1,:) & b_{30} \\
A(3,:) & a_{13}
\end{array} \) \( c_{01} \)

7. \( \begin{array}{cc}
A(0,:) & b_{12} \\
A(2,:) & a_{01} \\
A(1,:) & b_{20} \\
A(3,:) & a_{10}
\end{array} \) \( c_{10} \)

8. \( \begin{array}{cc}
A(0,:) & b_{22} \\
A(2,:) & a_{02} \\
A(1,:) & b_{12} \\
A(3,:) & b_{03}
\end{array} \) \( c_{11} \)

9. \( \begin{array}{cc}
A(0,:) & b_{32} \\
A(2,:) & a_{03} \\
A(1,:) & b_{22} \\
A(3,:) & a_{12}
\end{array} \) \( c_{02} \)

10. \( \begin{array}{cc}
A(0,:) & b_{00} \\
A(2,:) & a_{03} \\
A(1,:) & b_{32} \\
A(3,:) & a_{13}
\end{array} \) \( c_{03} \)

11. \( \begin{array}{cc}
A(0,:) & b_{01} \\
A(2,:) & a_{21} \\
A(1,:) & b_{00} \\
A(3,:) & a_{30}
\end{array} \) \( c_{12} \)

12. \( \begin{array}{cc}
A(0,:) & b_{20} \\
A(2,:) & a_{22} \\
A(1,:) & b_{10} \\
A(3,:) & b_{01}
\end{array} \) \( c_{13} \)
### Decoupled Systolic Matrix Multiplication

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(B(:,2))</td>
<td>(B(:,3))</td>
<td>(B(:,0))</td>
<td>(B(:,1))</td>
</tr>
<tr>
<td>(A(0,:))</td>
<td>(b_{32})</td>
<td>(b_{23})</td>
<td>(c_{02})</td>
</tr>
<tr>
<td>(A(2,:))</td>
<td>(a_{20})</td>
<td>(a_{03})</td>
<td>(c_{03})</td>
</tr>
<tr>
<td>(A(1,:))</td>
<td>(b_{22})</td>
<td>(b_{13})</td>
<td>(c_{12})</td>
</tr>
<tr>
<td>(A(3,:))</td>
<td>(a_{32})</td>
<td>(a_{11})</td>
<td>(c_{13})</td>
</tr>
</tbody>
</table>

(9)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(B(:,2))</td>
<td>(B(:,3))</td>
<td>(B(:,0))</td>
<td>(B(:,1))</td>
</tr>
<tr>
<td>(A(0,:))</td>
<td>(b_{20})</td>
<td>(b_{31})</td>
<td>(c_{20})</td>
</tr>
<tr>
<td>(A(2,:))</td>
<td>(a_{20})</td>
<td>(a_{33})</td>
<td>(c_{21})</td>
</tr>
<tr>
<td>(A(1,:))</td>
<td>(b_{32})</td>
<td>(b_{23})</td>
<td>(c_{30})</td>
</tr>
<tr>
<td>(A(3,:))</td>
<td>(a_{33})</td>
<td>(a_{12})</td>
<td>(c_{31})</td>
</tr>
</tbody>
</table>

(10)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(B(:,2))</td>
<td>(B(:,3))</td>
<td>(B(:,0))</td>
<td>(B(:,1))</td>
</tr>
<tr>
<td>(A(0,:))</td>
<td>(b_{12})</td>
<td>(b_{30})</td>
<td>(c_{22})</td>
</tr>
<tr>
<td>(A(2,:))</td>
<td>(a_{22})</td>
<td>(a_{31})</td>
<td>(c_{23})</td>
</tr>
<tr>
<td>(A(1,:))</td>
<td>(b_{22})</td>
<td>(b_{13})</td>
<td>(c_{32})</td>
</tr>
<tr>
<td>(A(3,:))</td>
<td>(a_{32})</td>
<td>(a_{11})</td>
<td>(c_{33})</td>
</tr>
</tbody>
</table>

(11)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(B(:,2))</td>
<td>(B(:,3))</td>
<td>(B(:,0))</td>
<td>(B(:,1))</td>
</tr>
<tr>
<td>(A(0,:))</td>
<td>(b_{00})</td>
<td>(b_{01})</td>
<td>(c_{00})</td>
</tr>
<tr>
<td>(A(2,:))</td>
<td>(a_{21})</td>
<td>(a_{20})</td>
<td>(c_{01})</td>
</tr>
<tr>
<td>(A(1,:))</td>
<td>(b_{30})</td>
<td>(b_{33})</td>
<td>(c_{10})</td>
</tr>
<tr>
<td>(A(3,:))</td>
<td>(a_{30})</td>
<td>(a_{13})</td>
<td>(c_{11})</td>
</tr>
</tbody>
</table>

(12)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(B(:,2))</td>
<td>(B(:,3))</td>
<td>(B(:,0))</td>
<td>(B(:,1))</td>
</tr>
<tr>
<td>(A(0,:))</td>
<td>(b_{00})</td>
<td>(b_{01})</td>
<td>(c_{00})</td>
</tr>
<tr>
<td>(A(2,:))</td>
<td>(a_{21})</td>
<td>(a_{20})</td>
<td>(c_{01})</td>
</tr>
<tr>
<td>(A(1,:))</td>
<td>(b_{30})</td>
<td>(b_{33})</td>
<td>(c_{10})</td>
</tr>
<tr>
<td>(A(3,:))</td>
<td>(a_{30})</td>
<td>(a_{13})</td>
<td>(c_{11})</td>
</tr>
</tbody>
</table>

(13)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(B(:,2))</td>
<td>(B(:,3))</td>
<td>(B(:,0))</td>
<td>(B(:,1))</td>
</tr>
<tr>
<td>(A(0,:))</td>
<td>(b_{02})</td>
<td>(b_{03})</td>
<td>(c_{20})</td>
</tr>
<tr>
<td>(A(2,:))</td>
<td>(a_{22})</td>
<td>(a_{21})</td>
<td>(c_{21})</td>
</tr>
<tr>
<td>(A(1,:))</td>
<td>(b_{32})</td>
<td>(b_{23})</td>
<td>(c_{30})</td>
</tr>
<tr>
<td>(A(3,:))</td>
<td>(a_{32})</td>
<td>(a_{12})</td>
<td>(c_{31})</td>
</tr>
</tbody>
</table>

(14)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(B(:,2))</td>
<td>(B(:,3))</td>
<td>(B(:,0))</td>
<td>(B(:,1))</td>
</tr>
<tr>
<td>(A(0,:))</td>
<td>(b_{12})</td>
<td>(b_{30})</td>
<td>(c_{22})</td>
</tr>
<tr>
<td>(A(2,:))</td>
<td>(a_{22})</td>
<td>(a_{31})</td>
<td>(c_{23})</td>
</tr>
<tr>
<td>(A(1,:))</td>
<td>(b_{22})</td>
<td>(b_{13})</td>
<td>(c_{32})</td>
</tr>
<tr>
<td>(A(3,:))</td>
<td>(a_{32})</td>
<td>(a_{11})</td>
<td>(c_{33})</td>
</tr>
</tbody>
</table>

(15)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(B(:,2))</td>
<td>(B(:,3))</td>
<td>(B(:,0))</td>
<td>(B(:,1))</td>
</tr>
<tr>
<td>(A(0,:))</td>
<td>(b_{10})</td>
<td>(b_{01})</td>
<td>(c_{10})</td>
</tr>
<tr>
<td>(A(2,:))</td>
<td>(a_{21})</td>
<td>(a_{20})</td>
<td>(c_{11})</td>
</tr>
<tr>
<td>(A(1,:))</td>
<td>(b_{00})</td>
<td>(b_{33})</td>
<td>(c_{01})</td>
</tr>
<tr>
<td>(A(3,:))</td>
<td>(a_{33})</td>
<td>(a_{13})</td>
<td>(c_{02})</td>
</tr>
</tbody>
</table>

(16)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(B(:,2))</td>
<td>(B(:,3))</td>
<td>(B(:,0))</td>
<td>(B(:,1))</td>
</tr>
<tr>
<td>(A(0,:))</td>
<td>(b_{20})</td>
<td>(b_{11})</td>
<td>(c_{20})</td>
</tr>
<tr>
<td>(A(2,:))</td>
<td>(a_{21})</td>
<td>(a_{20})</td>
<td>(c_{21})</td>
</tr>
<tr>
<td>(A(1,:))</td>
<td>(b_{10})</td>
<td>(b_{01})</td>
<td>(c_{30})</td>
</tr>
<tr>
<td>(A(3,:))</td>
<td>(a_{31})</td>
<td>(a_{30})</td>
<td>(c_{31})</td>
</tr>
</tbody>
</table>

(17)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(B(:,2))</td>
<td>(B(:,3))</td>
<td>(B(:,0))</td>
<td>(B(:,1))</td>
</tr>
<tr>
<td>(A(0,:))</td>
<td>(b_{32})</td>
<td>(b_{23})</td>
<td>(c_{02})</td>
</tr>
<tr>
<td>(A(2,:))</td>
<td>(a_{23})</td>
<td>(a_{02})</td>
<td>(c_{03})</td>
</tr>
<tr>
<td>(A(1,:))</td>
<td>(b_{22})</td>
<td>(b_{13})</td>
<td>(c_{12})</td>
</tr>
<tr>
<td>(A(3,:))</td>
<td>(a_{32})</td>
<td>(a_{11})</td>
<td>(c_{13})</td>
</tr>
</tbody>
</table>

(18)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(B(:,2))</td>
<td>(B(:,3))</td>
<td>(B(:,0))</td>
<td>(B(:,1))</td>
</tr>
<tr>
<td>(A(0,:))</td>
<td>(b_{33})</td>
<td>(a_{03})</td>
<td>(c_{03})</td>
</tr>
<tr>
<td>(A(2,:))</td>
<td>(a_{23})</td>
<td>(a_{02})</td>
<td>(c_{02})</td>
</tr>
<tr>
<td>(A(1,:))</td>
<td>(b_{22})</td>
<td>(b_{13})</td>
<td>(c_{12})</td>
</tr>
<tr>
<td>(A(3,:))</td>
<td>(a_{33})</td>
<td>(a_{11})</td>
<td>(c_{13})</td>
</tr>
</tbody>
</table>

(19)

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(B(:,2))</td>
<td>(B(:,3))</td>
<td>(B(:,0))</td>
<td>(B(:,1))</td>
</tr>
<tr>
<td>(A(0,:))</td>
<td>(b_{10})</td>
<td>(b_{01})</td>
<td>(c_{10})</td>
</tr>
<tr>
<td>(A(2,:))</td>
<td>(a_{21})</td>
<td>(a_{20})</td>
<td>(c_{11})</td>
</tr>
<tr>
<td>(A(1,:))</td>
<td>(b_{00})</td>
<td>(b_{33})</td>
<td>(c_{01})</td>
</tr>
<tr>
<td>(A(3,:))</td>
<td>(a_{33})</td>
<td>(a_{13})</td>
<td>(c_{02})</td>
</tr>
</tbody>
</table>

(20)
Efficiency of Stream Matrix Multiplication

# of multiply-and-adds, \( F(N) = N^3 \)

# of cycles, \( T(N) = (N/R)^3 R + 3R \)

# of procs, \( P(R) + M(R) = R^2 + 2R \)

ratio of \( N \) to \( R \), \( \sigma \) \( = N/R \)

\[
E(N, R) = \frac{F(N)}{T(N) \cdot (P(R) + M(R))}
\]

\[
E_{mm}(N, R) = \frac{N^3}{((N/R)^3 R + 3) \cdot (R^2 + 2R)}
\]

\[
E_{mm}(\sigma, R) = \frac{\sigma^3}{\sigma^3 + 3} \cdot \frac{R}{R + 2}
\]

\[
\lim_{\sigma, R \to \infty} E(\sigma, R) = 1, \quad E_{mm}(8, 16) > 90\%
\]
Existing Stream Algorithms and Efficiencies

FIR Filter
\[ E_{fir}(\sigma, R) = \frac{\sigma^2}{\sigma^2+1} \cdot \frac{R}{R+2} \]

Matrix Multiplication
\[ E_{mm}(\sigma, R) = \frac{\sigma^3}{\sigma^3+3} \cdot \frac{R}{R+2} \]

DFT
\[ E_{dft}(\sigma, R) = \frac{\sigma^2}{\sigma^2+1} \cdot \frac{R}{R+1} \]

Triangular Solver
\[ E_{ts}(\sigma, R) = \frac{\sigma^3}{\sigma^3+\sigma^2+6\sigma-2} \cdot \frac{R}{R+3} \]

LU Factorization
\[ E_{lu}(\sigma, R) \approx \frac{\sigma^3}{\sigma^3+\frac{3}{2}\sigma^2+\frac{31}{2}\sigma-6} \cdot \frac{R}{R+3} \]

QR Factorization
\[ E_{qr}(\sigma, R) \approx \frac{\sigma^3}{\sigma^3+\frac{3}{5}\sigma^2+\frac{64}{5}\sigma+\frac{3}{5}} \cdot \frac{R}{R+3} \]
Conclusion

1. Introduced Stream Architecture and Algorithms
   Pro: Maximum efficiency for large numbers of processors
   Con: design and implementation are hideous

2. DSP algorithms are Regular Iterative Algorithms
   [Rao and Kailath, 1988]
   Thus, can be structured as stream algorithms

3. Raw can implement the Stream Architecture
   Large Raw fabrics can achieve 100% efficiency

4. Future Work
   (a) Explore more algorithms: SVD, Graph Algorithms
   (b) Consider automatic code generation
   (c) Collect experimental results on Raw