Matrix Vector Multiplication

- Multiply an \( n \times m \) matrix with an \( m \times 1 \) vector in parallel

\[
\begin{bmatrix}
2 & 1 & 0 & 4 \\
3 & 2 & 1 & 1 \\
4 & 3 & 1 & 2 \\
3 & 0 & 2 & 0 \\
\end{bmatrix}
\times
\begin{bmatrix}
1 \\
3 \\
4 \\
1 \\
\end{bmatrix}
= 
\begin{bmatrix}
9 \\
14 \\
19 \\
11 \\
\end{bmatrix}
\]
Row-Block Decomposition

Inner product computation

Row $i$ of $A$

All-gather communication

Row $i$ of $A$
Column-Wise Algorithm Complexity

- **Sequential Runtime**
  - $T_s = \Theta(n^2)$

- **Parallel Runtime**
  - $T_p = \Theta\left(\frac{n^2}{p} + n + \log n\right)$

- **Processor Time Product**
  - $pT_p = \Theta(n^2 + pn + p\log n)$

- **Overhead**
  - $T_o = pT_p - T_s = \Theta(pn + p\log n)$

- **Isoefficiency Relation**
  - $n^2 = \Theta(pn + p\log n)$

- **Isoefficiency Function**
  - $n = \Theta(p)$
Column-Block Decomposition

\[
\begin{align*}
    c_0 &= a_{0,0} b_0 + a_{0,1} b_1 + a_{0,2} b_2 + a_{0,3} b_3 + a_{4,4} b_4 \\
    c_1 &= a_{1,0} b_0 + a_{1,1} b_1 + a_{1,2} b_2 + a_{1,3} b_3 + a_{4,4} b_4 \\
    c_2 &= a_{2,0} b_0 + a_{2,1} b_1 + a_{2,2} b_2 + a_{2,3} b_3 + a_{4,4} b_4 \\
    c_3 &= a_{3,0} b_0 + a_{3,1} b_1 + a_{3,2} b_2 + a_{3,3} b_3 + b_{3,4} b_4 \\
    c_4 &= a_{4,0} b_0 + a_{4,1} b_1 + a_{4,2} b_2 + a_{4,3} b_3 + a_{4,4} b_4
\end{align*}
\]

Processor 0’s initial computation

Processor 1’s initial computation

Proc 2

Proc 3

Proc 4
Column-Block Decomposition

Multiplications

All-to-all exchange

Reduction

Column $i$ of $A$

$b$

Column $i$ of $A$

$b$

~$c$

Column $i$ of $A$

$b$

~$c$

Column $i$ of $A$

$b$

$b$

$c$

Figure Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.
Row-Wise Algorithm Complexity

- **Sequential Runtime**
  - $T_s = \Theta(n^2)$

- **Parallel Runtime**
  - $T_p = \Theta\left(\frac{n^2}{p} + n \log p\right)$

- **Processor Time Product**
  - $pT_p = \Theta(n^2 + pn \log p)$

- **Overhead**
  - $T_o = pT_p - T_s = \Theta(pn \log p)$

- **Isoefficiency Relation**
  - $n^2 = \Theta(pn \log p)$

- **Isoefficiency Function**
  - $n = \Theta(p \log p)$
Checkerboard Decomposition
When $p$ is a square number

When $p$ is not a square number

Figure Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.
# Row-Block Results

<table>
<thead>
<tr>
<th>$p$</th>
<th>Predicted</th>
<th>Actual</th>
<th>Speedup</th>
<th>Mflops</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>63.4</td>
<td>63.4</td>
<td>1.00</td>
<td>31.6</td>
</tr>
<tr>
<td>2</td>
<td>32.4</td>
<td>32.7</td>
<td>1.94</td>
<td>61.2</td>
</tr>
<tr>
<td>3</td>
<td>22.3</td>
<td>22.7</td>
<td>2.79</td>
<td>88.1</td>
</tr>
<tr>
<td>4</td>
<td>17.0</td>
<td>17.8</td>
<td>3.56</td>
<td>112.4</td>
</tr>
<tr>
<td>5</td>
<td>14.1</td>
<td>15.2</td>
<td>4.16</td>
<td>131.6</td>
</tr>
<tr>
<td>6</td>
<td>12.0</td>
<td>13.3</td>
<td>4.76</td>
<td>150.4</td>
</tr>
<tr>
<td>7</td>
<td>10.5</td>
<td>12.2</td>
<td>5.19</td>
<td>163.9</td>
</tr>
<tr>
<td>8</td>
<td>9.4</td>
<td>11.1</td>
<td>5.70</td>
<td>180.2</td>
</tr>
<tr>
<td>16</td>
<td>5.7</td>
<td>7.2</td>
<td>8.79</td>
<td>277.8</td>
</tr>
</tbody>
</table>
## Column-Block Results

<table>
<thead>
<tr>
<th>$p$</th>
<th>Predicted</th>
<th>Actual</th>
<th>Speedup</th>
<th>Mflops</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>63.4</td>
<td>63.8</td>
<td>1.00</td>
<td>31.4</td>
</tr>
<tr>
<td>2</td>
<td>32.4</td>
<td>32.9</td>
<td>1.92</td>
<td>60.8</td>
</tr>
<tr>
<td>3</td>
<td>22.2</td>
<td>22.6</td>
<td>2.80</td>
<td>88.5</td>
</tr>
<tr>
<td>4</td>
<td>17.2</td>
<td>17.5</td>
<td>3.62</td>
<td>114.3</td>
</tr>
<tr>
<td>5</td>
<td>14.3</td>
<td>14.5</td>
<td>4.37</td>
<td>137.9</td>
</tr>
<tr>
<td>6</td>
<td>12.5</td>
<td>12.6</td>
<td>5.02</td>
<td>158.7</td>
</tr>
<tr>
<td>7</td>
<td>11.3</td>
<td>11.2</td>
<td>5.65</td>
<td>178.6</td>
</tr>
<tr>
<td>8</td>
<td>10.4</td>
<td>10.0</td>
<td>6.33</td>
<td>200.0</td>
</tr>
<tr>
<td>16</td>
<td>8.5</td>
<td>7.6</td>
<td>8.33</td>
<td>263.2</td>
</tr>
</tbody>
</table>
## Checkerboard Results

<table>
<thead>
<tr>
<th>Procs</th>
<th>Predicted (msec)</th>
<th>Actual (msec)</th>
<th>Speedup</th>
<th>Megaflops</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>63.4</td>
<td>63.4</td>
<td>1.00</td>
<td>31.6</td>
</tr>
<tr>
<td>4</td>
<td>17.8</td>
<td>17.4</td>
<td>3.64</td>
<td>114.9</td>
</tr>
<tr>
<td>9</td>
<td>9.7</td>
<td>9.7</td>
<td>6.53</td>
<td>206.2</td>
</tr>
<tr>
<td>16</td>
<td>6.2</td>
<td>6.2</td>
<td>10.21</td>
<td>322.6</td>
</tr>
</tbody>
</table>
Comparison

Figure

- Rowwise Block Striped
- Columnwise Block Striped
- Checkerboard Block

Processors

Speedup
Estimate Pi via Monte Carlo

\[ \frac{\text{Circle}}{\text{Square}} = \frac{D^2/4}{D^2} = \frac{1}{4} \]
Randomly Sample Points

\[
\frac{16}{20} \times 4 = 3.2 \approx \pi
\]
Accuracy Increases with Samples

<table>
<thead>
<tr>
<th>$n$</th>
<th>Estimate</th>
<th>Error</th>
<th>$1/(2n^{1/2})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2.40000</td>
<td>0.23606</td>
<td>0.15811</td>
</tr>
<tr>
<td>100</td>
<td>3.36000</td>
<td>0.06952</td>
<td>0.05000</td>
</tr>
<tr>
<td>1,000</td>
<td>3.14400</td>
<td>0.00077</td>
<td>0.01581</td>
</tr>
<tr>
<td>10,000</td>
<td>3.13920</td>
<td>0.00076</td>
<td>0.00500</td>
</tr>
<tr>
<td>100,000</td>
<td>3.14132</td>
<td>0.00009</td>
<td>0.00158</td>
</tr>
<tr>
<td>1,000,000</td>
<td>3.14006</td>
<td>0.00049</td>
<td>0.00050</td>
</tr>
<tr>
<td>10,000,000</td>
<td>3.14136</td>
<td>0.00007</td>
<td>0.00016</td>
</tr>
<tr>
<td>100,000,000</td>
<td>3.14154</td>
<td>0.00002</td>
<td>0.00005</td>
</tr>
<tr>
<td>1,000,000,000</td>
<td>3.14155</td>
<td>0.00001</td>
<td>0.00002</td>
</tr>
</tbody>
</table>
Mean Value Theorem

\[ \int_{a}^{b} f(x) \, dx = (b - a) f(\bar{x}) \]

Figure Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.
The expected value of \((1/n)(f(x_0) + \ldots + f(x_{n-1}))\) is \(f\)
\[
\int_{a}^{b} f(x) \, dx = (b - a) f \approx (b - a) \frac{1}{n} \sum_{i=0}^{n-1} f(x_i)
\]
Random Number Generators

- uniformly distributed
- uncorrelated (short term or long term)
- never cycles
- satisfies all statistical tests for randomness
- reproducible
- machine independent
- changed with a “seed” value
- easily split into independent subsequences
- generated rapidly
- limited memory required
- in practice: never satisfies all the above!
Linear Congruential RNG

\[ X_i = (a \times X_{i-1} + c) \mod M \]

- Multiplier
- Additive constant
- Modulus

Sequence depends on choice of seed, \( X_0 \)
$X_i = X_{i-p} * X_{i-q}$

$p$ and $q$ are lags, $p > q$

* is any binary arithmetic operation

- Addition modulo $M$
- Subtraction modulo $M$
- Multiplication modulo $M$
- Bitwise exclusive or
Parallel RNG Methods

• Manager-Worker Method
  – Master process hands out random samples
  – Not scalable

• Leapfrog Method
  – Task $t$ takes each $u_i$ where $i \mod \text{size} = t$
  – May turn long-range correlation to short-range

• Sequence Splitting Method
  – Each process allocated contiguous sequence
  – Can be slow, long-range correlation is issue

• Parameterization
  – Use different RNG for each process
  – Lagged Fibonacci is especially good for this
Parallel RNG Methods

Leapfrog Method

Process with rank 1 of 4 processes

Sequence Splitting Method

Process with rank 1 of 4 processes
Non-uniform Distributions

cumulative distribution

non-uniform distribution

inverse of cumulative distribution

Figure Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.
\[ F^{-1}(u) = -m \ln u \]

\[ F(x) = 1 - e^{-mx} \]

\[ f(x) = \frac{1}{m} e^{-x/m} \]
Box-Muller Transform (Gaussian)

repeat

\[ v_1 \leftarrow 2u_1 - 1 \]
\[ v_2 \leftarrow 2u_2 - 1 \]
\[ r \leftarrow v_1^2 + v_2^2 \]
until \( r > 0 \) and \( r < 1 \)

\[ f \leftarrow \sqrt{-2 \ln r/r} \]
\[ g_1 \leftarrow sf v_1 + m \]
\[ g_2 \leftarrow sf v_2 + m \]

\( u_1 \) and \( u_2 \) are samples from uniform distribution
\( g_1 \) and \( g_2 \) and samples from gaussian distribution
with mean \( m \) and standard deviation \( s \)
if \( u_i \delta h(x_i) \leq f(x_i) \) then accept. \( u_i \) and \( x_i \) are uniform random
\[ C = C_c + C_s \]

- distance \( L \) before interacting with an atom:
  \[ L = -\frac{1}{C} \ln u \]
- probability of bouncing is \( \frac{C_s}{C} \)
- probability of being absorbed is \( \frac{C_c}{C} \)
- scattering in each direction \( D \) is equally likely
- distance traveled in x direction is \( L \cos D \)
Temperature in a 2D Plate

Random walk
2D Ising Model
**Room Assignment Problem**

```
<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>3</td>
<td>5</td>
<td>9</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>B</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>C</td>
<td>5</td>
<td>2</td>
<td>0</td>
<td>8</td>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>9</td>
<td>6</td>
<td>8</td>
<td>0</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>E</td>
<td>1</td>
<td>4</td>
<td>9</td>
<td>3</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>6</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>0</td>
</tr>
</tbody>
</table>
```

Pairing A-B, C-D, and E-F leads to total conflict value of 32.

"Dislikes" matrix
Convergence in Simulated Annealing

Starting with higher initial temperature leads to more iterations before convergence.
## Parking Lot Simulation

### Times Spaces Are Available

<table>
<thead>
<tr>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>101.2</td>
</tr>
<tr>
<td>142.1</td>
</tr>
<tr>
<td>70.3</td>
</tr>
<tr>
<td>91.7</td>
</tr>
<tr>
<td>223.1</td>
</tr>
</tbody>
</table>

### Current Time

- **Car Count**: 15
- **Cars Rejected**: 2
Traffic Circle Simulation

![Traffic Circle Simulation Diagram]

<table>
<thead>
<tr>
<th></th>
<th>F</th>
<th>D</th>
<th>N</th>
<th>E</th>
<th>S</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>0.33</td>
<td>N</td>
<td>0.1</td>
<td>0.2</td>
<td>0.5</td>
<td>0.2</td>
</tr>
<tr>
<td>E</td>
<td>0.50</td>
<td>E</td>
<td>0.3</td>
<td>0.1</td>
<td>0.2</td>
<td>0.4</td>
</tr>
<tr>
<td>S</td>
<td>0.25</td>
<td>S</td>
<td>0.5</td>
<td>0.3</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>W</td>
<td>0.33</td>
<td>W</td>
<td>0.2</td>
<td>0.4</td>
<td>0.3</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Figure Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.
### Data Structures

<table>
<thead>
<tr>
<th>Iteration</th>
<th>N</th>
<th>W</th>
<th>S</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>4</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>26</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>19</td>
<td>23</td>
<td>22</td>
<td>38</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>13</td>
<td>11</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>49</td>
<td>20</td>
<td>41</td>
<td></td>
</tr>
</tbody>
</table>

**Offset**

**Arrival**

**ArrivalCnt**

**WaitCnt**

**Queue**

**QueueAccum**

<table>
<thead>
<tr>
<th>Circle</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
</tr>
</tbody>
</table>
Back Substitution

\[\begin{align*}
1x_0 &+ 1x_1 - 1x_2 + 4x_3 &= 8 \\
-2x_1 &- 3x_2 + 1x_3 &= 5 \\
2x_2 &- 3x_3 &= 0 \\
2x_3 &= 4
\end{align*}\]
Back Substitution

\[
\begin{align*}
1x_0 + 1x_1 - 1x_2 + 4x_3 &= 8 \\
-2x_1 - 3x_2 + 1x_3 &= 5 \\
2x_2 - 3x_3 &= 0 \\
x_3 &= 2 \\
2x_3 &= 4
\end{align*}
\]
Back Substitution

\[ 1x_0 + 1x_1 - 1x_2 = 0 \]
\[ -2x_1 - 3x_2 = 3 \]
\[ 2x_2 = 6 \]
\[ 2x_3 = 4 \]
Back Substitution

\[ \begin{align*}
1x_0 + 1x_1 - 1x_2 &= 0 \\
-2x_1 - 3x_2 &= 3 \\
2x_2 &= 6 \\
x_2 &= 3 \\
2x_3 &= 4
\end{align*} \]
Back Substitution

\[ 1x_0 + 1x_1 = 3 \]

\[ -2x_1 = 12 \]

\[ 2x_2 = 6 \]

\[ 2x_3 = 4 \]
Back Substitution

\[
\begin{align*}
1x_0 + 1x_1 &= 3 \\
-2x_1 &= 12 \\
2x_2 &= 6 \\
x_1 &= -6 \\
2x_3 &= 4
\end{align*}
\]
Back Substitution

\[
\begin{align*}
1x_0 &= 9 \\
-2x_1 &= 12 \\
2x_2 &= 6 \\
2x_3 &= 4
\end{align*}
\]
Back Substitution

\[ 1x_0 = 9 \]
\[ -2x_1 = 12 \]
\[ 2x_2 = 6 \]
\[ x_0 = 9 \quad 2x_3 = 4 \]
Back Substitution Pseudocode

for $i \leftarrow n - 1$ down to 1 do
  $x[i] \leftarrow b[i] / a[i,i]$
  for $j \leftarrow 0$ to $i - 1$ do
    $b[j] \leftarrow b[j] - x[i] \times a[j,i]$
  endfor
endfor

Time complexity: $\Theta(n^2)$
Back Substitution Task Graph

We cannot execute the outer loop in parallel.
We can execute the inner loop in parallel.
Interleaved Decomposition

Rowwise interleaved striped decomposition

Columnwise interleaved striped decomposition
Comparing Row and Column

Message-passing time dominates
Column-oriented algorithm superior

Computational time dominates
Row-oriented algorithm superior

Figure Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.
Gaussian Elimination

\[
\begin{align*}
4x_0 + 6x_1 + 2x_2 - 2x_3 &= 8 \\
2x_0 + 5x_2 - 2x_3 &= 4 \\
-4x_0 - 3x_1 - 5x_2 + 4x_3 &= 1 \\
8x_0 + 18x_1 - 2x_2 + 3x_3 &= 40
\end{align*}
\]
Gaussian Elimination

\[
\begin{align*}
4x_0 & \quad +6x_1 & \quad +2x_2 & \quad -2x_3 & = & \quad 8 \\
-3x_1 & \quad +4x_2 & \quad -1x_3 & = & \quad 0 \\
+3x_1 & \quad -3x_2 & \quad +2x_3 & = & \quad 9 \\
+6x_1 & \quad -6x_2 & \quad +7x_3 & = & \quad 24
\end{align*}
\]
Gaussian Elimination

\[
\begin{align*}
4x_0 + 6x_1 + 2x_2 - 2x_3 &= 8 \\
-3x_1 + 4x_2 - 1x_3 &= 0 \\
1x_2 + 1x_3 &= 9 \\
2x_2 + 5x_3 &= 24
\end{align*}
\]
Gaussian Elimination

\[
\begin{align*}
4x_0 + 6x_1 + 2x_2 - 2x_3 &= 8 \\
-3x_1 + 4x_2 - 1x_3 &= 0 \\
1x_2 + 1x_3 &= 9 \\
3x_3 &= 6
\end{align*}
\]
Gaussian Elimination Pseudocode

1. procedure GAUSSIAN ELIMINATION (A, b, y)
2. begin
3. for k := 0 to n - 1 do /* Outer loop */
4. begin
5. for j := k + 1 to n - 1 do
7. y[k] := b[k] / A[k, k];
8. A[k, k] := 1;
9. for i := k + 1 to n - 1 do
10. begin
11. for j := k + 1 to n - 1 do
13. b[i] := b[i] - A[i, k] × y[k];
15. endfor; /* Line9 */
16. endfor; /* Line3 */
17. end GAUSSIAN ELIMINATION
Gaussian Elimination Iteration

Figure 8.5 A typical computation in Gaussian elimination.
Pipelined Row-wise Algorithm

![Figure 8.7](image) Pipelined Gaussian elimination on a 5 x 5 matrix partitioned with one row per process.

---

© 2003, Walt Ligon, all rights reserved.
Partial Pivoting

Without partial pivoting  With partial pivoting
Column-oriented Partial Pivoting

\[ a[j][i] \]

\[ a[j][k] \]

\[ a[picked][i] \]

\[ a[picked][k] \]
Iterative Methods

- Jacobi's Method
  - Solving $Ax = b$
  - Guess a series of values of $x^k$
  - Evaluate $Ax^k = b'$
  - Compare $b$ and $b'$
  - Refine $x^k$ into $x^{k+1}$
  - Repeat until $b - b' < \text{threshold}$
  - For each iteration $k$:
    \[
    x_{i}^{k+1} = \frac{1}{a_{ii}} \left( b_{i} - \sum_{j \neq i} a_{ij} x_{j}^{k} \right)
    \]
  - Then Allgather $x^{k+1}$ for next iteration
More Iterative Methods

- Gauss-Seidel Method
  - Same approach as Jacobi Method but ...
  - For each iteration $k$:
    
    $$
x_{i}^{k+1} = \frac{1}{a_{ii}} \left( b_{i} - \sum_{j<i} a_{ij} x_{j}^{k+1} + \sum_{j>i} a_{ij} x_{j}^{k} \right)
    $$

  - Then Allgather $x^{k+1}$ for next iteration
  - This converges faster
Real Iterative Methods

• Conjugate Gradient Method
  – Minimize $q(x) = 1/2x^TAx - x^Tb + c$
  – Iteration computes $x^k = x^{k-1} + s^kd^k$
  – Iteration:

  $g^k = Ax^{k-1} - b$

  $d^k = -g^k + (g^kTg^k)/(g^{k-1T}g^{k-1}) * d^{k-1}$

  $s^k = (d^kTg^k)/(d^kTAd^k)$

  $x^k = x^{k-1} + s^kd^k$