Designing Parallel Programs
Abhinav Bhanetele, Department of Computer Science
Writing parallel programs

- Decide the serial algorithm first
- Data: how to distribute data among threads/processes?
  - Data locality: assignment of data to specific processes to minimize data movement
- Computation: how to divide work among threads/processes?
- Figure out how often communication will be needed
Two-dimensional stencil computation

- Commonly found kernel in computational codes
- Heat diffusion, Jacobi method, Gauss-Seidel method

Serial code
for(int t=0; t<num_steps; t++) {
    ...
    for(i ...) 
        for(j ...) 
    // copy contents of A_new into A 
    ...
}
2D stencil computation in parallel
2D stencil computation in parallel

- 1D decomposition
  - Divide rows (or columns) among processes
2D stencil computation in parallel

• 1D decomposition
  • Divide rows (or columns) among processes
2D stencil computation in parallel

- **1D decomposition**
  - Divide rows (or columns) among processes

- **2D decomposition**
  - Divide both rows and columns (2d blocks) among processes
2D stencil computation in parallel

- 1D decomposition
  - Divide rows (or columns) among processes

- 2D decomposition
  - Divide both rows and columns (2d blocks) among processes
Prefix sum

- Calculate partial sums of elements in array
- Also called a “scan” sometimes

\[ \text{pSum}[0] = A[0] \]

\[
\text{for}(i=1; \ i<N; \ i++) \ { \\
\quad \text{pSum}[i] = \text{pSum}[i-1] + A[i] \\
}
\]

<table>
<thead>
<tr>
<th>A</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>pSum</td>
<td>1</td>
<td>3</td>
<td>6</td>
<td>10</td>
<td>15</td>
<td>21</td>
<td>...</td>
</tr>
</tbody>
</table>
Parallel prefix sum

<table>
<thead>
<tr>
<th>2</th>
<th>8</th>
<th>3</th>
<th>5</th>
<th>7</th>
<th>4</th>
<th>1</th>
<th>6</th>
</tr>
</thead>
</table>
Parallel prefix sum

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>8</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>4</td>
<td>1</td>
<td>6</td>
</tr>
</tbody>
</table>
Parallel prefix sum

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>8</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>4</td>
<td>1</td>
<td>6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>10</td>
<td>11</td>
<td>8</td>
<td>12</td>
<td>11</td>
<td>5</td>
<td>7</td>
</tr>
</tbody>
</table>
Parallel prefix sum

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>8</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>4</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>10</td>
<td>11</td>
<td>8</td>
<td>12</td>
<td>11</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>10</td>
<td>13</td>
<td>18</td>
<td>23</td>
<td>19</td>
<td>17</td>
<td>18</td>
</tr>
</tbody>
</table>
Parallel prefix sum

0 1 2 3 4 5 6 7

2 8 3 5 7 4 1 6

2 10 11 8 12 11 5 7

2 10 13 18 23 19 17 18

2 10 13 18 25 29 30 36
In practice
In practice

- You have $N$ numbers and $P$ processes, $N >> P$
In practice

- You have $N$ numbers and $P$ processes, $N \gg P$
- Assign a $N/P$ block to each process
  - Do calculation for the blocks on each process locally
In practice

- You have $N$ numbers and $P$ processes, $N \gg P$

- Assign a $N/P$ block to each process
  - Do calculation for the blocks on each process locally

- Then do parallel algorithm with partial prefix sums
The $n$-body problem

- Simulate the motion of celestial objects interacting with one another due to gravitational forces
- Naive algorithm: $O(n^2)$
  - Every body calculates forces pair-wise with every other body (particle)

Data distribution in $n$-body problems

- Naive approach: Assign $n/p$ particles to each process
- Other approaches?
Data distribution in $n$-body problems

- Naive approach: Assign $n/p$ particles to each process
- Other approaches?

http://datagenetics.com/blog/march222013/
https://en.wikipedia.org/wiki/Z-order_curve
Data distribution in \( n \)-body problems

- Naive approach: Assign \( n/p \) particles to each process
- Other approaches?

Space-filling curves

http://datagenetics.com/blog/march222013/
https://en.wikipedia.org/wiki/Z-order_curve
Data distribution in $n$-body problems

- Naive approach: Assign $n/p$ particles to each process
- Other approaches?

Space-filling curves

http://datagenetics.com/blog/march22013/
https://en.wikipedia.org/wiki/Z-order_curve

Data distribution in $n$-body problems

- Let us consider a two-dimensional space with bodies/particles in it
Data distribution in $n$-body problems

• Let us consider a two-dimensional space with bodies/particles in it
Data distribution in $n$-body problems

- Let us consider a two-dimensional space with bodies/particles in it
Load balance and grain size

• Load balance: try to balance the amount of work (computation) assigned to different threads/ processes
  • Bring ratio of maximum to average load as close to 1.0 as possible
  • Secondary consideration: also load balance amount of communication

• Grain size: ratio of computation-to-communication
  • Coarse-grained (more computation) vs. fine-grained (more communication)