Withall
=
Nested For + If

Sheena Mathew,
Div. of Computer Engg.,
School of Engineering,
CUSAT,
Kerala, India.

Tomsy Paul,
Dept. Of Computer Applications,
RIT Kottayam,
Kerala, India.
Overview

- Typical Array processing
  - Lower Triangle of a Matrix
    - For(i=0;i<N;i++)
      For(j=0;j<N;j++)
        If (i<=j) A[i,j] = A[i,j] * 0.2;
  - Withall Loop
    - withall (i<=j) A[i,j] = A[i,j] * 0.2;
withall : Syntax

- withall \( (expression) \) \( statement \)
- \( statement \) should contain at least one Array reference
- If there are many, all have same dimensions and size
- \( expression \) involves Index variables in Array reference
- Examples
  - \texttt{withall (i==j) A[i,j] = 1;}
  - \texttt{withall (i+j == N+1) printf("\%d",A[i,j]);}
- \( Expression \) can also be \( (i,j) \) – Applied to all elements
  - \texttt{withall (i,j) A[i,j] = 0;}

Motivation

- *with* loop of SAC
- \( X = [1,2,3]; \)
- *with* ([0] \( \leq k < [2] \)) : 7
- \text{modarray}(x)
- \( X \) becomes [7,7,3]
Current Literature

• List Comprehension

• From Functional Programming (Python – numpy)

• Example (Haskell)

• \[ \{(i,j)|i\in[1..3],j\in[1..3],i\leq j\} \]

• Result

  – \[ (1,1),(1,2),(1,3),(2,2),(2,3),(3,3) \]

• Complexity - \( O(N^2) \)
Current Literature

- Boolean Indexing
- Present in MATLAB, Python, R, etc.
- \(a=[1,2,3,4,5]\)
- \(a>3\)
- Results in \([0,0,0,1,1]\)
- \(a[a>3]=3\)
- print \(a\)
- Output \([1,2,3,3,3]\)
Implementing withall

- Syntactic Sugar
- Efficient Implementation
  - NP Hard
  - Solution?
- Restricted Grammar
Restricted Grammar

- withall_loop
  -> withall withall_expression statement
  | withall ID_LIST statement
  ;
- withall_expression
  -> withall_expression && withall_expression
  | withall_expression || withall_expression
  | (withall_expression)
  | !withall_expression
  | withall_term
  ;
Restricted Grammar

- $\text{withall\_term} \rightarrow \text{ID RELOP expression}$

- RELOP can be any of the six operators $<,\leq,!,=,==,\geq$ and $>$ and ID is any index variable used in statement.

- Example
  - $i \leq 10$
  - $j \leq i$
  - $\neg i+j < N+1$
Implementation

• For One Dimensional Arrays
  – Using List of Ordered Pairs, for each with all_expression
  – Examples
    • int a[1:100]
      – i < 50 represented by {{1,49}}
      – i > 25 represented by {{26,100}}
      – i == 50 represented by {{50,50}}
      – i != 50 represented by {{1,49},{51,100}}
Implementation

• For One Dimensional Arrays
  – Using List of Ordered Pairs, for each withall_expression
  – Advantages
    • Logical And, Or and Negation can be implemented as Union, Intersection and Complement of Ordered Pairs
    • $O(m+n)$ for And and Or
    • $O(n)$ for Negation
Implementation

- int a[1:100]
  - (i < 50) && (i > 20)
    - \{(1,49)\} Intersection \{(21,100)\} \longrightarrow \{(21,49)\}
  - (i < 50) || (i > 75)
    - \{(1,49)\} Union \{(76,100)\} \longrightarrow \{(1,49),(76,100)\}
  - !(i < 30) && (i>20))
    - Complement \{(21,29)\} \longrightarrow \{(1,20),(30,100)\}
Implementation

• Language Used
• Cray Chapel
  – Free & Open Source
  – Excellent Support
  – LR Parser
  – Modules and Iterators
  – Version 1.14
Implementation

• Platform
  – Desktop
    • Intel Xeon 2 GHz, 6 core processor
    • 8 GB RAM
Implementation

• Procedure
  – Modified Chapel Compiler
  – Files modified
    • compiler/parser/chapel.lex
    • compiler/parser/chapel.ypp
    • compiler/AST/ForLoop.cpp
    • compiler/include/ForLoop.h
    • modules/internal/ChapelArray.chpl
    • modules/internal/ChapelIteratorSupport.chpl
Experiment

• Withall Vs Nested For + If
• Array Size (N)
  – $10^3$
  – $10^4$
  – $10^6$
  – $10^8$
Experiment

- Withall Vs Nested For + If
- `withall_expressions`

<table>
<thead>
<tr>
<th>Number</th>
<th><code>withall_expression</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>i</td>
</tr>
<tr>
<td>2</td>
<td>i==N/2</td>
</tr>
<tr>
<td>3</td>
<td>i&lt;=N/2</td>
</tr>
<tr>
<td>4</td>
<td>!(i&gt;N/2)</td>
</tr>
<tr>
<td>5</td>
<td>i&lt;=N/4</td>
</tr>
<tr>
<td>6</td>
<td>i&gt;=N/4 &amp;&amp; i&lt;=N/2</td>
</tr>
<tr>
<td>7</td>
<td>!(i&gt;=N/4 &amp;&amp; i&lt;=N/2)</td>
</tr>
</tbody>
</table>
Results

• Comparison
  – Execution Time
  – Target Code Size
  – Compilation Time
## Execution Time (Seconds)

<table>
<thead>
<tr>
<th>withall_expression</th>
<th>Array Size</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$10^3$</td>
<td>$10^4$</td>
<td>$10^6$</td>
<td>$10^8$</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$w^a$</td>
<td>$2.10 \times 10^{-6}$</td>
<td>$1.93 \times 10^{-5}$</td>
<td>$1.18 \times 10^{-3}$</td>
<td>$1.50 \times 10^{-1}$</td>
</tr>
<tr>
<td>$f+i^b$</td>
<td>$2.50 \times 10^{-6}$</td>
<td>$1.50 \times 10^{-5}$</td>
<td>$1.16 \times 10^{-3}$</td>
<td>$1.61 \times 10^{-1}$</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$w$</td>
<td>$9.00 \times 10^{-7}$</td>
<td>$1.00 \times 10^{-6}$</td>
<td>$1.00 \times 10^{-6}$</td>
<td>$1.80 \times 10^{-6}$</td>
</tr>
<tr>
<td>$f+i$</td>
<td>$2.50 \times 10^{-6}$</td>
<td>$1.75 \times 10^{-5}$</td>
<td>$1.68 \times 10^{-3}$</td>
<td>$1.71 \times 10^{-1}$</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$w$</td>
<td>$1.80 \times 10^{-6}$</td>
<td>$1.14 \times 10^{-5}$</td>
<td>$6.04 \times 10^{-4}$</td>
<td>$7.52 \times 10^{-2}$</td>
</tr>
<tr>
<td>$f+i$</td>
<td>$2.90 \times 10^{-6}$</td>
<td>$1.96 \times 10^{-5}$</td>
<td>$1.83 \times 10^{-3}$</td>
<td>$1.84 \times 10^{-1}$</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$w$</td>
<td>$2.50 \times 10^{-6}$</td>
<td>$1.19 \times 10^{-5}$</td>
<td>$6.07 \times 10^{-4}$</td>
<td>$7.44 \times 10^{-2}$</td>
</tr>
<tr>
<td>$f+i$</td>
<td>$2.70 \times 10^{-6}$</td>
<td>$1.98 \times 10^{-5}$</td>
<td>$1.83 \times 10^{-3}$</td>
<td>$1.84 \times 10^{-1}$</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$w$</td>
<td>$3.40 \times 10^{-6}$</td>
<td>$1.10 \times 10^{-5}$</td>
<td>$5.71 \times 10^{-4}$</td>
<td>$7.23 \times 10^{-2}$</td>
</tr>
<tr>
<td>$f+i$</td>
<td>$3.30 \times 10^{-6}$</td>
<td>$2.64 \times 10^{-5}$</td>
<td>$2.45 \times 10^{-3}$</td>
<td>$2.45 \times 10^{-1}$</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$w$</td>
<td>$2.00 \times 10^{-6}$</td>
<td>$7.00 \times 10^{-6}$</td>
<td>$3.19 \times 10^{-4}$</td>
<td>$3.75 \times 10^{-2}$</td>
</tr>
<tr>
<td>$f+i$</td>
<td>$3.30 \times 10^{-6}$</td>
<td>$2.43 \times 10^{-5}$</td>
<td>$2.36 \times 10^{-3}$</td>
<td>$2.34 \times 10^{-1}$</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$w$</td>
<td>$5.30 \times 10^{-6}$</td>
<td>$1.52 \times 10^{-5}$</td>
<td>$5.96 \times 10^{-4}$</td>
<td>$7.00 \times 10^{-2}$</td>
</tr>
<tr>
<td>$f+i$</td>
<td>$4.20 \times 10^{-6}$</td>
<td>$3.31 \times 10^{-5}$</td>
<td>$3.37 \times 10^{-3}$</td>
<td>$3.13 \times 10^{-1}$</td>
</tr>
</tbody>
</table>

$^a$ $w$ denotes **withall**  
$^b$ $f+i$ denotes for loop + if statement
## Target Code Size (KB)

<table>
<thead>
<tr>
<th>withall_expression</th>
<th>Target Code Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(w)</td>
</tr>
<tr>
<td>1</td>
<td>3667.89</td>
</tr>
<tr>
<td>2</td>
<td>3668.04</td>
</tr>
<tr>
<td>3</td>
<td>3668.04</td>
</tr>
<tr>
<td>4</td>
<td>3668.08</td>
</tr>
<tr>
<td>5</td>
<td>3672.37</td>
</tr>
<tr>
<td>6</td>
<td>3672.27</td>
</tr>
<tr>
<td>7</td>
<td>3672.40</td>
</tr>
</tbody>
</table>
## Time to Compile (Seconds)

<table>
<thead>
<tr>
<th>withall_expression</th>
<th>$10^3$</th>
<th>$10^4$</th>
<th>$10^6$</th>
<th>$10^8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w</td>
<td>10.19</td>
<td>10.68</td>
<td>10.59</td>
<td>10.26</td>
</tr>
<tr>
<td>f+i</td>
<td>10.52</td>
<td>10.42</td>
<td>10.14</td>
<td>9.98</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w</td>
<td>10.60</td>
<td>10.38</td>
<td>10.42</td>
<td>10.36</td>
</tr>
<tr>
<td>f+i</td>
<td>10.41</td>
<td>10.19</td>
<td>10.17</td>
<td>10.18</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w</td>
<td>10.29</td>
<td>10.49</td>
<td>10.40</td>
<td>10.36</td>
</tr>
<tr>
<td>f+i</td>
<td>10.13</td>
<td>10.17</td>
<td>10.21</td>
<td>10.16</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w</td>
<td>10.48</td>
<td>10.45</td>
<td>10.43</td>
<td>10.39</td>
</tr>
<tr>
<td>f+i</td>
<td>10.43</td>
<td>10.23</td>
<td>10.14</td>
<td>10.13</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w</td>
<td>10.85</td>
<td>10.96</td>
<td>10.35</td>
<td>10.71</td>
</tr>
<tr>
<td>f+i</td>
<td>10.64</td>
<td>10.09</td>
<td>10.11</td>
<td>10.11</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w</td>
<td>11.21</td>
<td>10.51</td>
<td>10.43</td>
<td>10.45</td>
</tr>
<tr>
<td>f+i</td>
<td>10.63</td>
<td>10.44</td>
<td>10.10</td>
<td>10.11</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w</td>
<td>10.57</td>
<td>10.60</td>
<td>10.18</td>
<td>10.56</td>
</tr>
<tr>
<td>f+i</td>
<td>10.50</td>
<td>10.14</td>
<td>10.09</td>
<td>10.02</td>
</tr>
</tbody>
</table>
Conclusion

• Withall is concise
• More than a Syntactic Sugar
  – One Dimension – Tuple based
  – Much Faster than For + If
• Two Dimensional
  – List of Polygons
  – Polygon Intersection for And
  – Polygon Union for Or
Reference


Python 3.7.5rc1 documentation, https://docs.python.org/3/tutorial/datastructures.html


Thank You!!