Solving Triangular Systems in Parallel

Summary of Forward Substitution Algorithms

- Sequential Inner Product Version
- Sequential Vector Sum
- Fine-Grain Parallel Pseudocode

```
**if** $i = j$
    recv sum reduction $\sigma_i$
    $x_i = (b_i - \sigma_i)/l_{ii}$
    broadcast $x_i$ to tasks $(k, i)$
    $(k = i + 1, \cdots, n)$
**else**
    recv $x_j$
    $t = l_{i,j} x_j$
    send $t$ for sum reduction across tasks
    $(i, k)$  $(k = 1, \cdots, (i - 1))$ to task $(i, i)$
**end**
```

- Row Partitioning for Vector Sum Algorithm (Fan-out)

  The output loop (for all the columns) is sequential. Consequently the number of sequential steps is $O(n)$. Processors have to wait for information above them before they start updating b_j. Processors *broadcast* the x_j's they are responsible for, as soon as they are calculated.

```
for $j = 1$ to $n$
    if $j \in myrows$ then
        $x_j = b_j / l_{jj}$
        broadcast $x_j$ to other tasks
    else
        recv $x_j$
    end
    for $i \in myrows, \ \ i > j$
        $b_i = b_i - l_{ij} x_j$
    end
end
```

- Column Partitioning for Inner Product Algorithm (Fan-in)

  The output loop (for all the rows) is sequential. Consequently the number of sequential steps is $O(n)$. For each row a partial inner product (t) is calculated and used in a *reduce* accross other processors. Processors have to wait for informationl on their left before calculating x_j's.

```
for $i = 1$ to $n$
    $t = 0$
    for $j \in mycols, \;\; j < i$
        $t = t + l_{ij}x_j$
    end
    if $i \in mycols$ then
        recv sum reduction of $t$
        $x_i = (b_i - t)/l_{ii}$
    else
        send $t$ for sum reduction across tasks
    end
end
```

- Wavefront Vector Sum Algorithm

```
for $j \in mycols$
   for $k = 1$ to (# of $segments$)
      recv $segment$
      if $k = 1$ then
         $x_j = (b_j - z_j)/l_{jj}$
         $segment = segment - \{z_j\}$
      end
      for $z_i \in segment$
         $z_i = z_i + l_{ij}x_j$
      end
      if $|segment| > 0$ then
         send $segment$ to task with column $j + 1$
      end
   end
end
```

- Wavefront Scalar Product Algorithm

```
for $j \in myrows$
    for $k = 1$ to (# of $segments - 1$)
        recv $segment$
        send $segment$ to task owing row $i + 1$
        for $x_j \in segment$
            $b_i = b_i - l_{ij}x_j$
        end
    end
    recv $segment$
    for $x_j \in segment$
        $b_i = b_i - l_{ij}x_j$
    end
    $x_i = b_i/l_{ii}$
    $segment = segment \cup \{x_i\}$
    send $segment$ to task owing row $i + 1$
end
```