Parallel Program Design
Tasks, Critical Path

Ned Nedialkov

McMaster University
Canada

SE 3F03
January 2015
Outline

Foster’s Methodology

Tasks

Degrees of concurrency

Critical path

Examples
Foster’s Methodology

1. *Partitioning*. Divide the computation into small tasks that can be done in parallel

2. *Communication*. Determine the communications between the tasks

3. *Aggregation (or agglomeration)*. Combine tasks and communications into larger tasks

4. *Mapping*. Assign the combined tasks to processes (or threads)

From Ian Foster, *Designing and Building Parallel Programs*, http://www.mcs.anl.gov/~itf/dbpp/
Tasks

- We can decompose a computation into smaller parts or tasks
- The goal is to determine which can be executed in parallel
- **Fine-grained** decomposition: many small tasks
- **Coarse-grained** decomposition: small number of large tasks
- **Task-dependency graph**
  - directed acyclic graph
  - nodes are tasks
  - there is an edge between task $A$ and task $B$ if $B$ must be executed after $A$
Degrees of concurrency

- **Maximum degree of concurrency**: the maximum number of tasks that can be executed in parallel
- **Average degree of concurrency**: the average number of tasks that can be executed in parallel
- The average degree of concurrency is a more useful measure
Critical path

- **Start nodes**: nodes with no incoming edges
- **Finish nodes**: nodes with no outgoing edges
- **Critical path**: the longest directed path between any pair of start and finish nodes
- **Critical path length**: sum of the weights of the nodes on a critical path

\[
\text{ave degree of concurrency} = \frac{\text{total amount of work}}{\text{critical path length}}
\]
Example

Task-dependency graph

- maximum degree of concurrency is 6
- critical path length is 5
- total amount of work is 14 (assuming each task takes one unit of time)
- average degree of concurrency is $14/5 = 2.8$
- An assignment to 2 processes and the order in which the tasks are executed
- First number is the process number
- The speedup is $14/7 = 2$
- An assignment to 3 processes
- The speedup is $14/5 = 2.8$
- An assignment to 4 processes
- The speedup is \( \frac{14}{5} = 2.8 \)

- What is the speedup on 8 processes?
Example: LU decomposition

- We want to compute the LU decomposition of a matrix $A$
- Assume $A$ consists of $3 \times 3$ blocks
- We want

\[
\begin{bmatrix}
A_{11} & A_{12} & A_{13} \\
A_{21} & A_{22} & A_{23} \\
A_{31} & A_{32} & A_{33}
\end{bmatrix} \rightarrow 
\begin{bmatrix}
L_{11} & 0 & 0 \\
L_{21} & L_{22} & 0 \\
L_{31} & L_{32} & L_{33}
\end{bmatrix}
\begin{bmatrix}
U_{11} & U_{12} & U_{13} \\
0 & U_{22} & U_{23} \\
0 & 0 & U_{33}
\end{bmatrix}
\]

\[L \rightarrow U\]
We need to determine the $L_{ij}$ and $U_{ij}$

Consider $L$ times the first column of $U$

We have

\[ A_{11} = L_{11} U_{11} \]
compute $L_{11}$ and $U_{11}$ \hfill (1)

\[ A_{21} = L_{21} U_{11} \]
\[ L_{21} = A_{21} U_{11}^{-1} \] \hfill (2)

\[ A_{31} = L_{31} U_{11} \]
\[ L_{31} = A_{31} U_{11}^{-1} \] \hfill (3)

From multiplying the first row of $L$ times $U$, we obtain

\[ A_{12} = L_{11} U_{12} \]
\[ U_{12} = L_{11}^{-1} A_{21} \] \hfill (4)

\[ A_{13} = L_{11} U_{13} \]
\[ U_{13} = L_{11}^{-1} A_{13} \] \hfill (5)
Then we write

\[ A_{22} = L_{21} U_{12} + L_{22} U_{22} \]

\[ A_{23} = L_{21} U_{13} + L_{22} U_{23} \]

\[ A_{32} = L_{31} U_{12} + L_{32} U_{22} \]

\[ A_{33} = L_{31} U_{13} + L_{32} U_{23} + L_{33} U_{33} \]

\[ A'_{22} = A_{22} - L_{21} U_{12} = L_{22} U_{22} \] \hspace{1cm} (6)

\[ A'_{23} = A_{23} - L_{21} U_{13} = L_{22} U_{23} \] \hspace{1cm} (8)

\[ U_{23} = L_{22}^{-1} A'_{23} \] \hspace{1cm} (9)

\[ A'_{32} = A_{32} - L_{31} U_{12} = L_{32} U_{22} \] \hspace{1cm} (10)

\[ L_{32} = A'_{32} U_{22}^{-1} \] \hspace{1cm} (11)

\[ A'_{33} = A_{33} - L_{31} U_{13} \] \hspace{1cm} (12)

\[ A''_{33} = A'_{33} - L_{32} U_{23} = L_{33} U_{33} \] \hspace{1cm} (13)

\[ \text{compute } L_{33} \text{ and } U_{22} \] \hspace{1cm} (14)
Using the right column in (1–14), we can decompose the computation in the following tasks:

1. \( A_{11} = L_{11} U_{11} \)  
   (compute LU factorization)
2. \( L_{21} = A_{21} U_{11}^{-1} \)
3. \( L_{31} = A_{31} U_{11}^{-1} \)
4. \( U_{12} = L_{11}^{-1} A_{21} \)
5. \( U_{13} = L_{11}^{-1} A_{13} \)
6. \( A'_{22} = A_{22} - L_{21} U_{12} \)
7. \( A'_{32} = A_{32} - L_{31} U_{12} \)
8. \( A'_{23} = A_{23} - L_{21} U_{13} \)
9. \( A'_{33} = A_{33} - L_{31} U_{13} \)
10. \( A''_{22} = L_{22} U_{22} \)  
    (compute LU factorization)
11. \( L_{32} = A'_{32} U_{22}^{-1} \)
12. \( U_{23} = L_{22}^{-1} A'_{23} \)
13. \( A''_{33} = A'_{33} - L_{32} U_{23} \)
14. \( A''_{33} = L_{33} U_{33} \)  
    (compute LU factorization)
**Figure:** Task-dependency graph
Figure: Task distribution on 4 processes. Maximum speedup is 2