HW/SW Codesign

Exercise 3:
Mapping and Partitioning (1/2)

8. October 2014

Devendra Rai

devendra.rai@tik.ee.ethz.ch
Mapping

\[\text{Mapping} = \text{Binding} + \text{Scheduling}\]

Bind software (application) components to hardware (architecture) components

How to specify?
How to specify - Application specification

- Depends on the model of computation
  Ex: task graphs, process network, state charts, ...

- Commonly represented as a graph \(- G_p(V_p, E_p)\)

- Nodes \(V_p\) denote functional and communication units

- Edges \(E_p\) denote data/control dependencies

Wireless LAN 802.11a job represented as a Synchronous Dataflow graph
How to specify – Architecture specification

- Depends on the **model of platform**
- Commonly represented as a **graph** - \( G_A(V_A, E_A) \). Nodes \( V_A \) denote architectural units. Edges \( E_A \) denote links (connected-ness).

Block diagram of a single tile (left) and the multi-tile SHAPEs architecture (right). The availability of six links on the distributed network processor enables the construction of a scalable on-chip/off-chip toroidal network.
What then is binding

- Binding is a function $f: V_P \rightarrow V_A$.

- Some bindings **may not be valid**
  
  - **Functionality constraints**
    - Communication node cannot be bound to a DSP node
    - FFT decomposition node written for a DSP cannot be bound to a RISC processor node
    - Encryption node cannot be bound to a NoC switch
  
  - **Connectedness constraints**
    - Two application nodes requiring communication cannot be bound to two architectural nodes with no link between them
**Definition:** A *specification graph* is a graph $G_S = (V_S, E_S)$ consisting of a problem graph $G_P$, an architecture graph $G_A$, and edges $E_M$. In particular, $V_S = V_P \cup V_A$, $E_S = E_P \cup E_A \cup E_M$. 

![Spec Graph](image)
Problem 1

- Given
  - task graph,
  - architecture graph,
  - table of possible bindings

- To
  - draw application graph with nodes for communication
  - specification graph
  - suggest modifications to architecture

Figure 1: Task graph and target architecture

Table 1: Possible bindings for tasks $T_1 \ldots T_4$. 

<table>
<thead>
<tr>
<th>Component</th>
<th>$T_1$</th>
<th>$T_2$</th>
<th>$T_3$</th>
<th>$T_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIPS</td>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>DSP</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FPGA</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>ASIC</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
</tbody>
</table>
Partitioning

- Partitioning problem is to divide a set of objects into mutually exclusive blocks (see formal definition in lecture slides)

- Several methods – ILP, random, hierarchical clustering, Kernighan-Lin algorithm, simulated annealing, Evolutionary algorithms

- Partitioning is a key step in binding decisions
  - What to run on software (RISC processor) and what to run on hardware (specialized co-processors)?
  - How to bind tasks on a multicore processor?
  - How to implement a given behavior on a FPGA?
Hierarchical clustering

• Define a closeness function between every pair of nodes
  – Designing closeness functions for real problem is quite an art. We will discover more on this in next exercise!

• Nodes that are close are good candidates for clustering into same partition

• Method:
  – in each step we cluster two closest nodes and appropriately modify the graph
  – After all steps, we decide the cut-level and generate the partition
Hierarchical clustering: lecture example

1. Merge two closest nodes
2. Modify the graph by changing the new weights using arithmetic mean
3. Repeat process till done
Hierarchical clustering: lecture example

step 1:

step 2:

step 3:

- Choose cut line and generate partitioning
- Another art for real problems
Problem 2

• Given
  – A graph with closeness functions for each pair of connected nodes

• To
  – Hierarchically cluster the graph by setting closeness functions of new edges using
    – average values
    – minimum values

Figure 2: Graph with objects
Solution

slides next
Solution  Problem 1

Problemgraph

Architekturgraph

T1

T2

T3

T4

MIPS

DSP

FPGA

ASIC

bus 1

bus 2
Solution Problem 1

• Restricted binding because of communication provided by architecture: no way to interconnect FPGA with MIPS

• Can try to improve the for instance interconnect bus1 and bus2 to a common shared bus or a hierarchical bus

Figure 1: Task graph and target architecture
Solution Problem 2

Figure 4: Hierarchical Clustering with "average closeness"
Solution Problem 2

Figure 5: Hierarchical Clustering with "minimum closeness"