Module 3: Behavioral Synthesis
Lecture 3.5: Heuristic Scheduling – ASAP and ALAP

Anand Raghunathan
raghunathan@purdue.edu
Heuristic Scheduling Techniques

- Unconstrained
  - As soon as possible (ASAP)
  - As late as possible (ALAP)
- Scheduling as a graph partitioning problem
- List scheduling
- Force-directed scheduling
Y = ((a*b)+c)+(d*e)-(f+g)

ASAP Schedule: Example

The execution cycle of each operation is the least one allowed by the dependencies.
ASAP Scheduling Algorithm

**ASAP** (DFG $G(V, E)$)

```c
for each $v_i \in V$
  if($v_i$ is driven only by PIs)
    $t_i = 1$;

repeat
  for each $v_i \in V$
    if(predecessors of $v_i$ are all scheduled) {
      Schedule $v_i$ by setting $t_i = \text{MAX}(t_j + d_j)$;
    }

until all the nodes are scheduled;
```

**Terminology:**
- $t_i$: start cycle of $v_i$
- $d_i$: no. of cycles for $v_i$
- $v_j$ is a predecessor of $v_i$
- if $(v_j, v_i) \in E$

Q: What is the worst-case time complexity of this algorithm?
ALAP Schedule: Example

\[ Y = ((a\times b) + c) + (d \times e) - (f + g) \]

The execution cycle of each operation is the latest one allowed by the dependencies and the latency constraint.
ALAP Scheduling Algorithm

**ALAP** (DFG $G(V, E)$, Time $T$) {

  for each $v_i \in V$ {
    if ($v_i$ drives POs)
      $t_i$ (execution cycle of $v_i$) = $T$;
  }

  repeat {
    for each $v_i \in V$ {
      if (successors of $v_i$ are all scheduled) {
        Schedule $v_i$ by setting $t_i = \text{MIN}(t_j) - d_i$;
      }
    }
  }

  until all the nodes are scheduled;
}

**Q:** What is the worst-case time complexity of this algorithm?

**Terminology:**
- $t_i$: start cycle of $v_i$
- $d_i$: no. of cycles for $v_i$
- $v_j$ is a successor of $v_i$ if $(v_i, v_j) \in E$
Mobility (or Slack)

Y = ((a*b)+c)+(d*e)-(f+g)

Mobility is the difference of the execution cycles computed by ALAP and ASAP scheduling.
Scheduling as a Graph Partitioning Problem

- Given an acyclic data-flow graph $\text{DFG} = G(V, E)$
- Partition it into sub-graphs $S_1 \ldots S_k$ such that
  - The reduced graph formed by collapsing each partition into a single vertex is acyclic
    - Constraint ensures causality is maintained
  - Additional constraints
    - Longest path between ops in a single partition (limit clock period)
    - Number of operations of a given type in a single partition (limit area)