Scheduling

Revisited

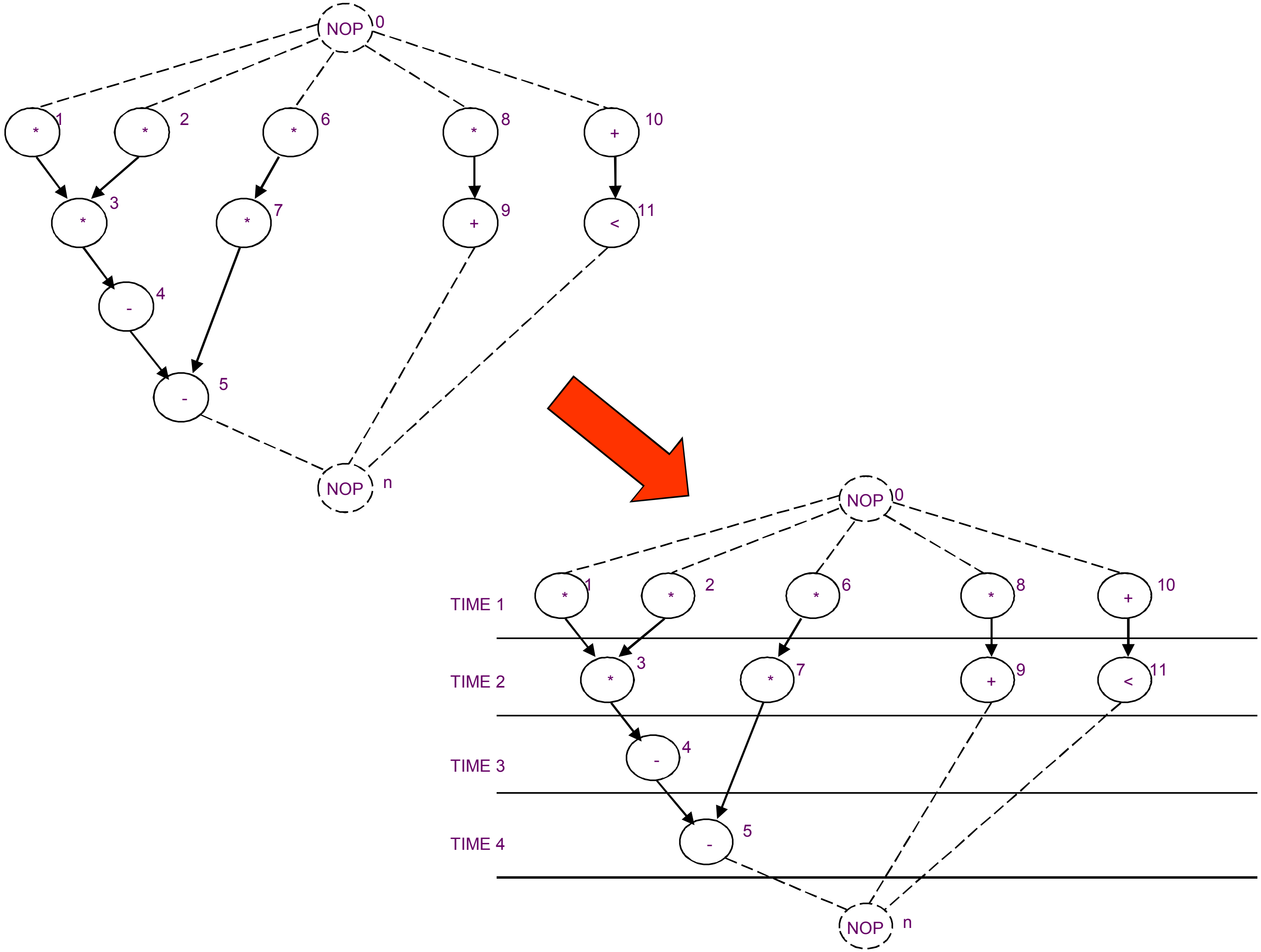
Module 1

- **Objectives:**
  - **The scheduling problem**
    - **Case analysis**
  - **Scheduling without constraints**
  - **Scheduling with timing constraints**

Scheduling

- **Circuit model:**
  - Sequencing graph
  - Cycle-time is given
  - Operation delays expressed in cycles
- **Scheduling:**
  - Determine the start times for the operations
  - Satisfying all the sequencing (timing and resource) constraint
- **Goal:**
  - Determine *area/latency* trade-off

Example



Taxonomy

- Unconstrained scheduling
- Scheduling with timing constraints:
  - Latency
  - Detailed timing constraints
- Scheduling with resource constraints
- Related problems:
  - Chaining
  - Synchronization
  - Pipeline scheduling

Simplest method

- All operations have bounded delays
- All delays are in cycles:
  - Cycle-time is given
- No constraints – no bounds on area
- Goal:
  - Minimize latency

Minimum-latency unconstrained scheduling problem

◆Given a set of ops *V* with integer delays *D* and a partial order on the operations *E:*

◆Find an integer labeling of the operations $\varphi : V \rightarrow Z^+$ such that:

$t_i = \varphi(v_i)$,

$t_i \geq t_j + d_j \qquad \forall\, i, j \text{ s.t. } (v_j, v_i) \in E$

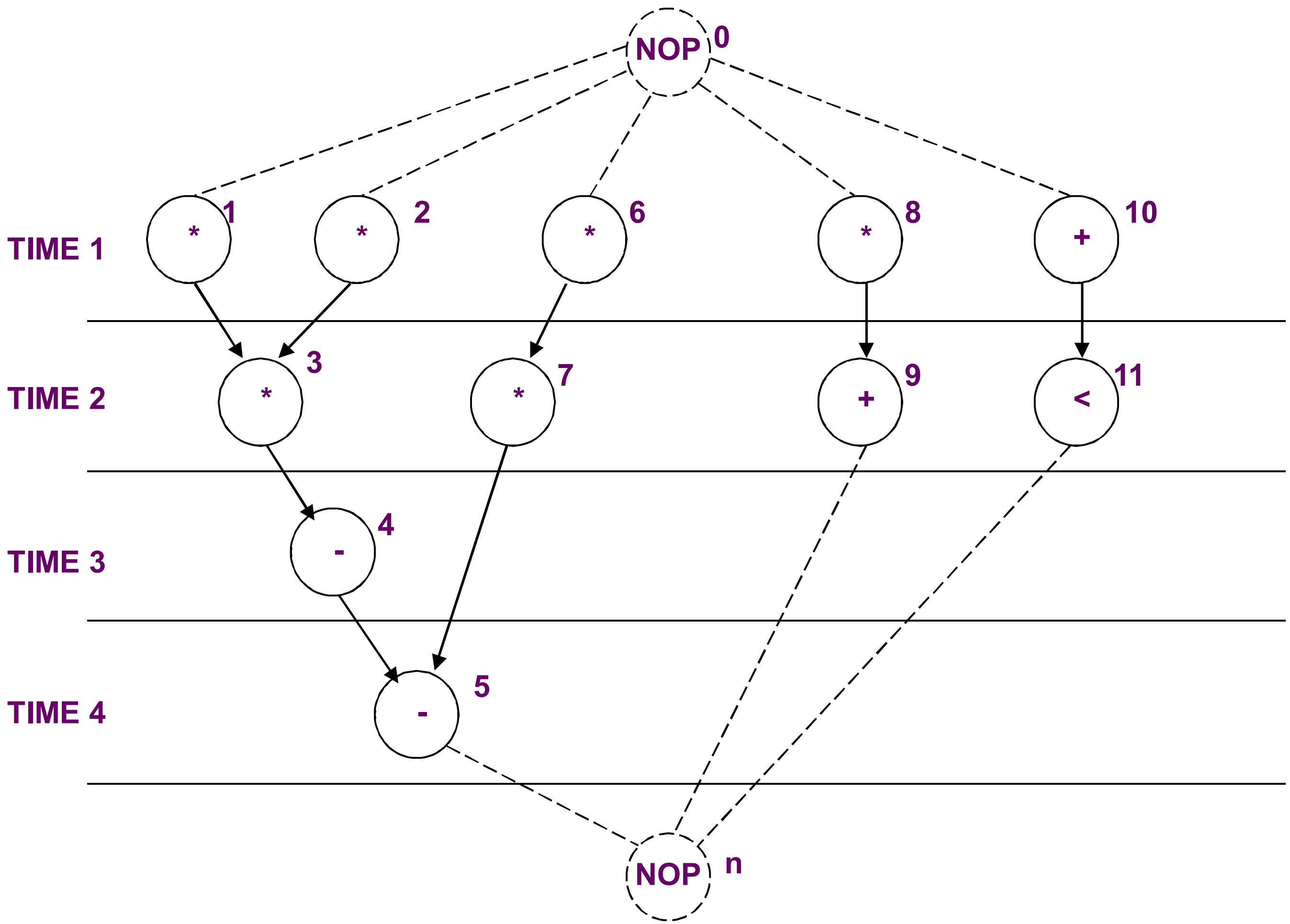
and t_n is minimum

ASAP scheduling algorithm

```
ASAP ( G_s(V,E) ) {
Schedule v_0 by setting t_0^S = 1;
repeat {
        Select a vertex v_i whose predecessors are all scheduled;
        Schedule v_i by setting  t_i^S =   max       t_j^S  + d_j ;
}                                      j:(v_j,v_i) є E
until (v_n is scheduled);
return (t^S);
}
```

$$t_i^S = \max_{j:(v_j,v_i) \in E} t_j^S + d_j$$

Example



ALAP scheduling algorithm

ALAP ($G_s(V,E), \bar{\lambda}$) {

Schedule v_n by setting $t_n^L = \bar{\lambda} + 1$;

repeat {

Select a vertex v_i whose successors are all scheduled;

Schedule v_i by setting $t_i^L = \min_{j:(v_i,v_j) \in E} t_j^L - d_i$;

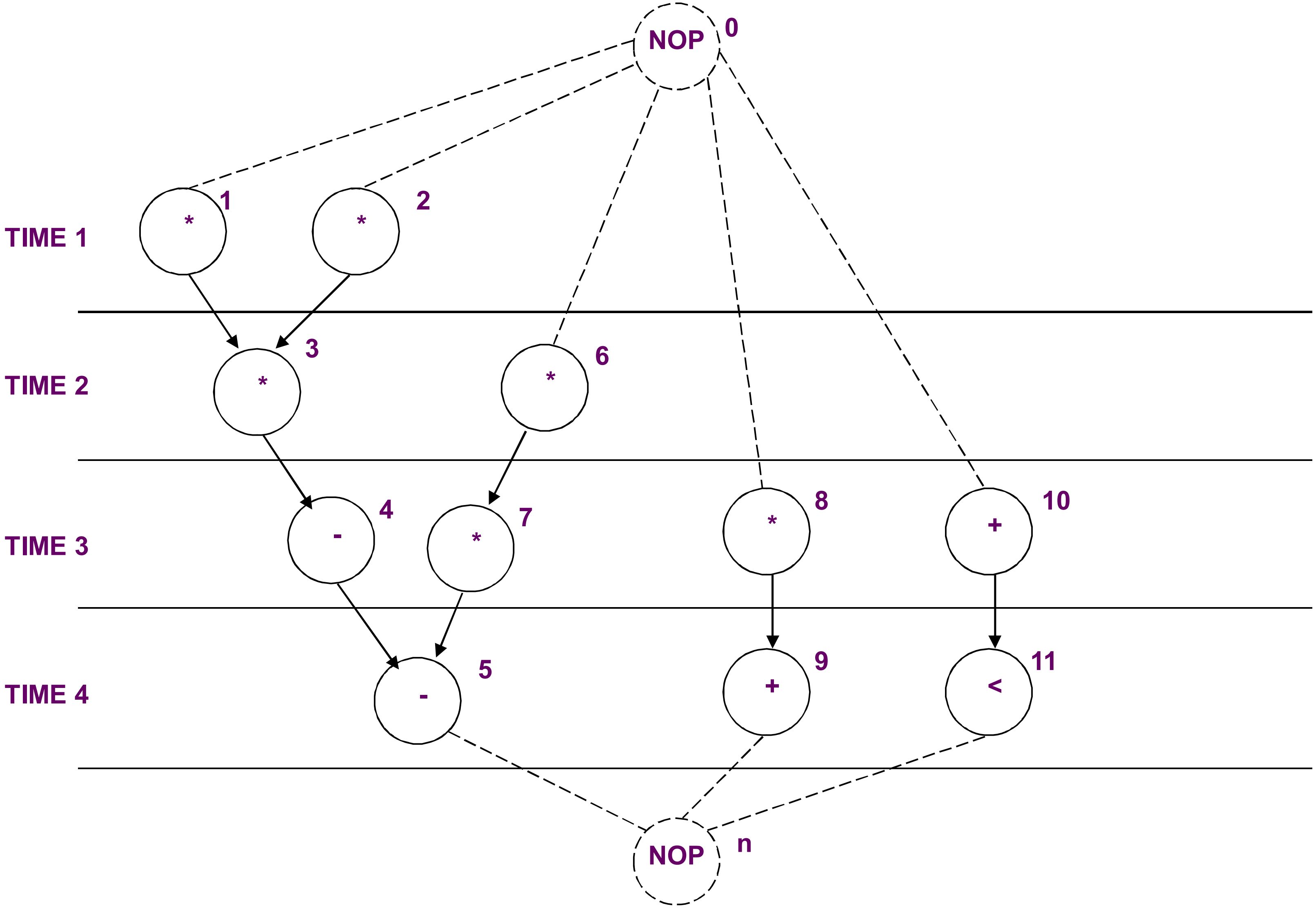
}

until (v_0 is scheduled);

return (t^L);

}

Example

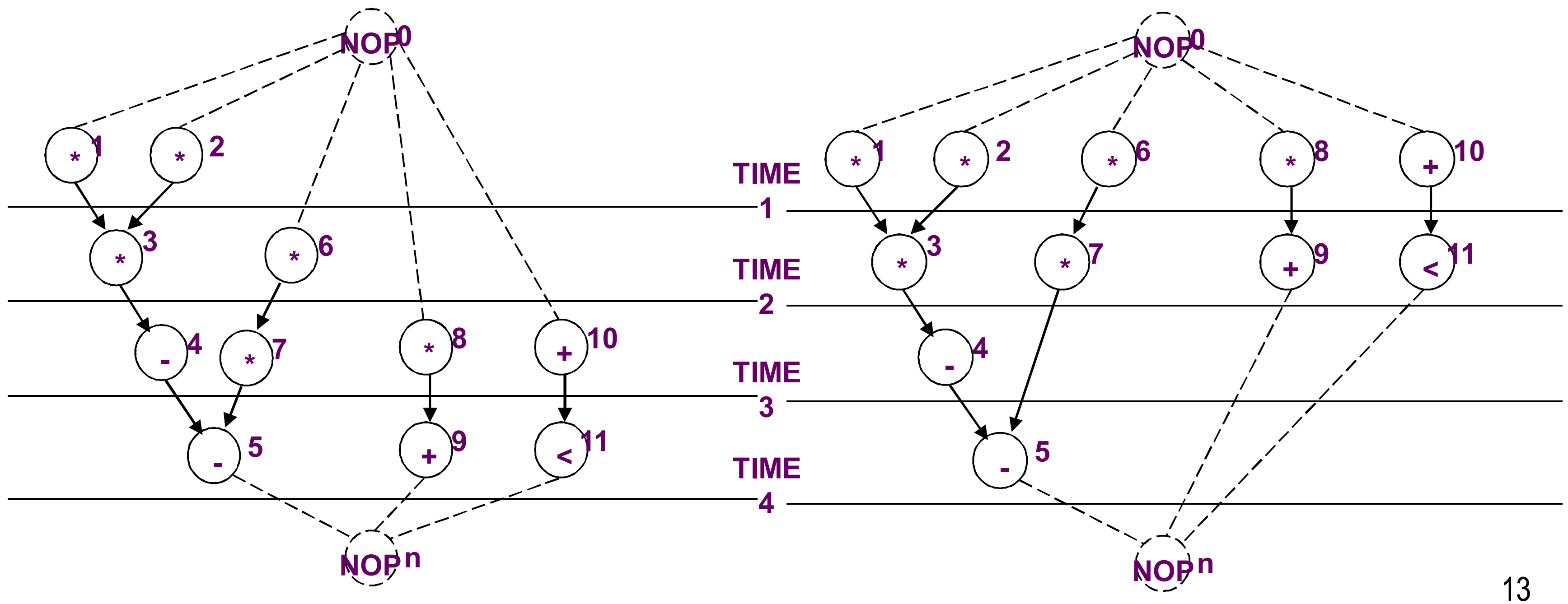


Remarks

- ALAP solves a latency-constrained problem
- Latency bound can be set to latency computed by ASAP algorithm
- Mobility:
  - Defined for each operation
  - Difference between ALAP and ASAP schedule
- Slack on the start time

Example

- **Operations with zero mobility:**
  - { v_1, v_2, v_3, v_4, v_5 }
  - Critical path
- **Operations with mobility one:**
  - { v_6, v_7 }
- **Operations with mobility two:**
  - { v_8, v_9, v_{10}, v_{11} }



Module 2

- **Objectives:**
  - Scheduling with resource constraints
  - Exact formulation:
    - ILP
    - Hu's algorithm
  - Heuristic methods
    - List scheduling
    - Force-directed scheduling

Scheduling under resource constraints

- Intractable problem
- Algorithms:
  - Exact:
    - Integer linear program
    - Hu (restrictive assumptions)
  - Approximate :
    - List scheduling
    - Force-directed scheduling

ILP formulation

- **Binary decision variables:**

  $X = \{ x_{il}, \quad i = 1,2,\dots. n; \; l = 1,2,\dots, \lambda + 1\}$

  x_{il} is TRUE only when operation v_i starts in step l of the schedule (i.e. $l = t_i$)

  λ is an upper bound on latency

- **Start time of operation v_i :** $\Sigma_l \; l \cdot x_{il}$

ILP formulation constraints

- Operations start only once

  $\Sigma\, x_{il} = 1 \quad i = 1, 2, \ldots, n$

- Sequencing relations must be satisfied

  $t_i \geq t_j + d_j \quad \rightarrow \quad t_i - t_j - d_j \geq 0$ *for all* $(v_j, v_i) \in E$

  $\Sigma\, l \bullet x_{il} - \Sigma\, l \bullet x_{jl} - d_j \geq 0$ *for all* $(v_j, v_i) \in E$

- Resource bounds must be satisfied

  Simple case (unit delay)

  $\Sigma^{l}_{i:T(v_i)=k}\, x_{il} \leq a_k \quad k = 1,2,\ldots n_{res}\,;$ *for all* l

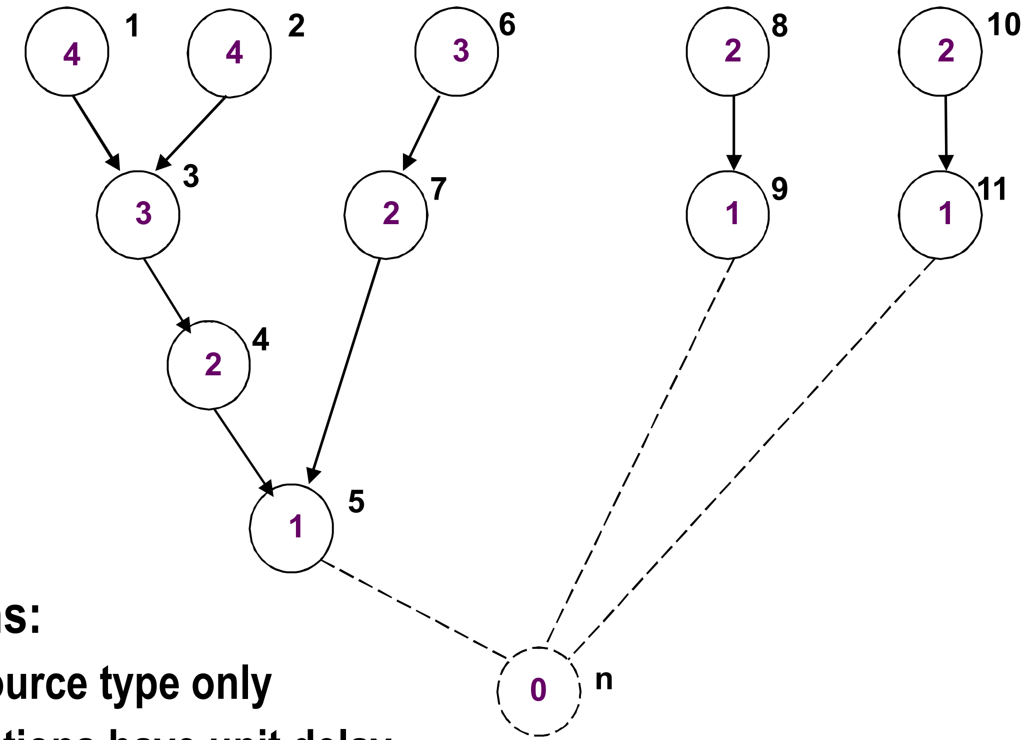
ILP Solution

- Use standard ILP packages
- Transform into LP problem
- Advantages:
  - Exact method
  - Others constraints can be incorporated
- Disadvantages:
  - Works well up to few thousand variables

Hu's algorithm

- **Assumptions:**
  - Graph is a forest
  - All operations have unit delay
  - All operations have the same type
- **Algorithm:**
  - Greedy strategy
  - Exact solution

Example



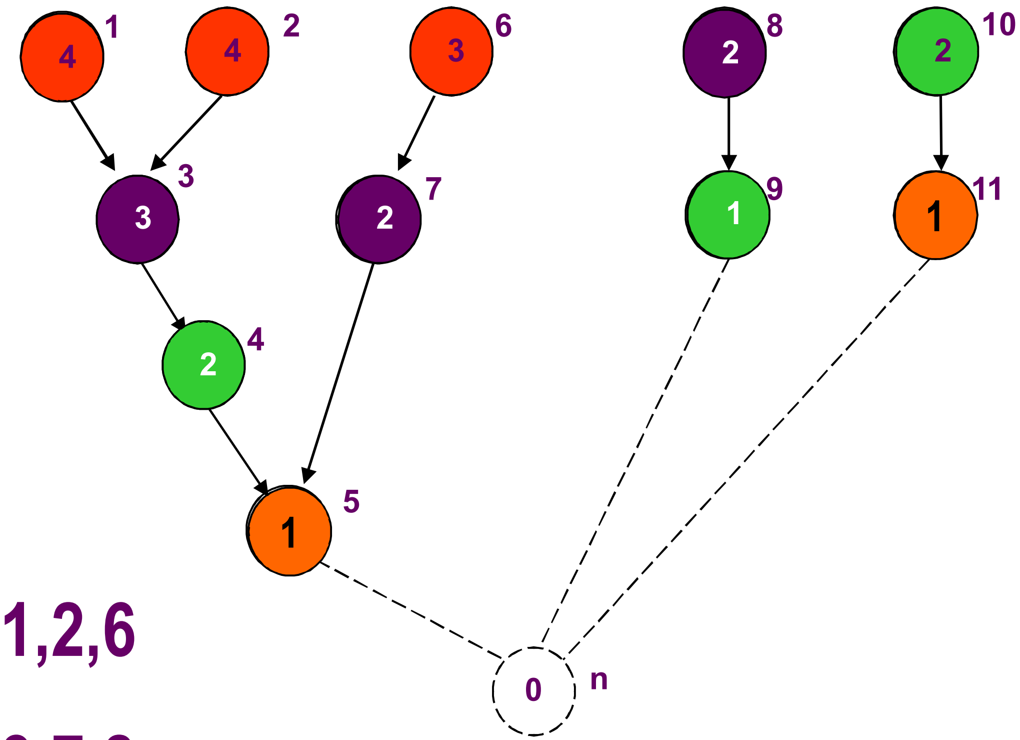
- **Assumptions:**
  - One resource type only
  - All operations have unit delay
- **Labels:**
  - Distance to sink

Algorithm
Hu's schedule with ā resources

- Label operations with distance to sink
- Set step $l = 1$
- Repeat until all ops are scheduled:
  - Select $s \leq \bar{a}$ resources with
    - All predecessors scheduled
    - Maximal labels
  - Schedule the s operations at step l
  - Increment step $l = l + 1$

Example

$\bar{a} = 3$



Step 1: Op 1,2,6

Step 2: Op 3,7,8

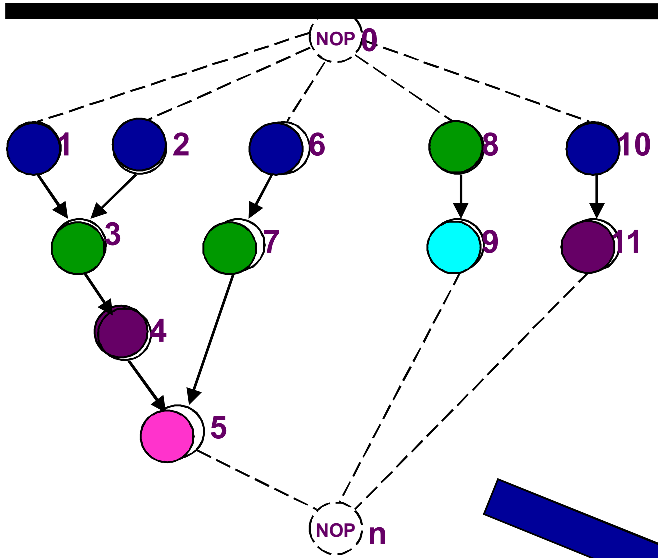
Step 3: Op 4,9,10

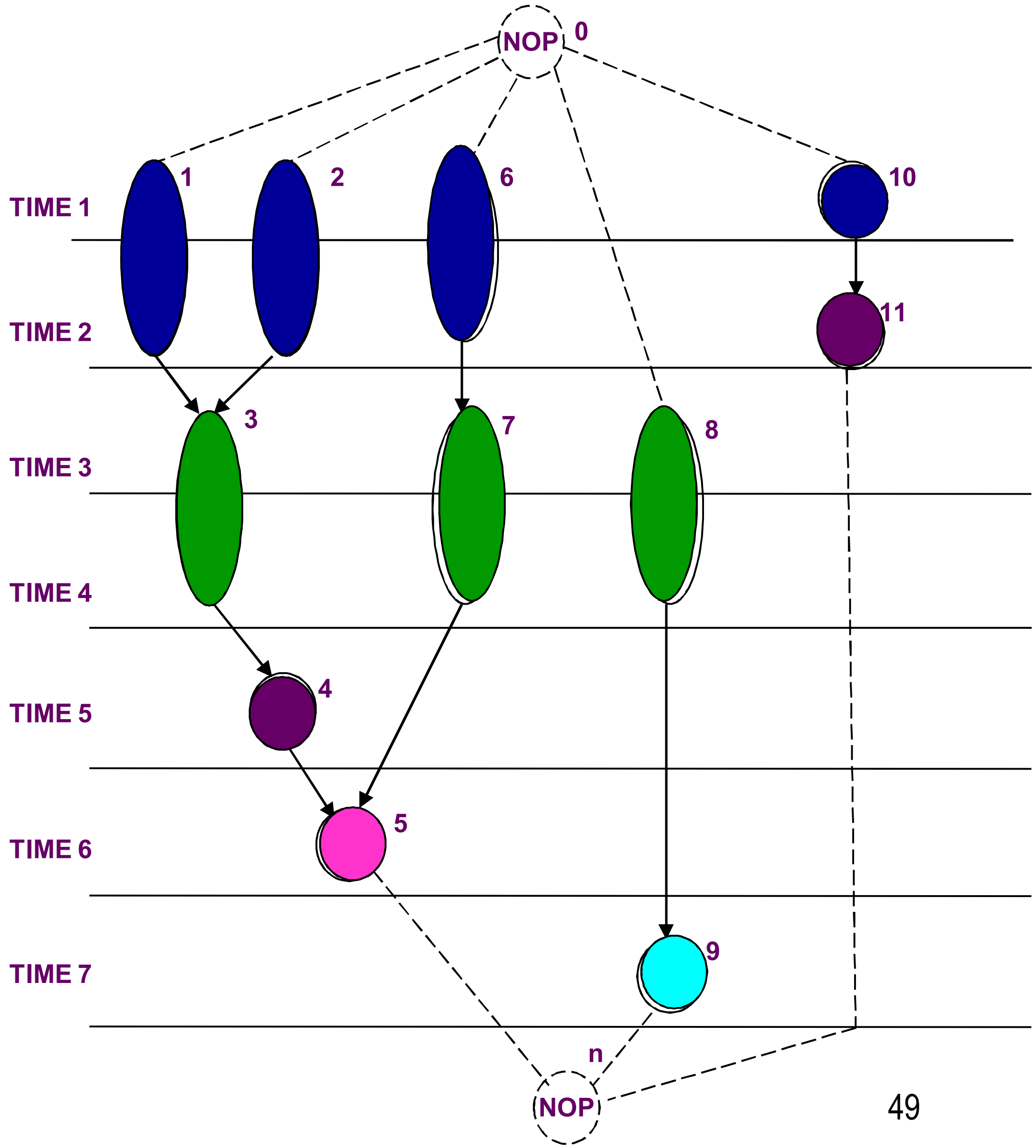
Step 4: Op 5,11

List scheduling algorithms

- Heuristic method for:
  - Min *latency* subject to *resource bound*
  - Min *resource* subject to *latency bound*
- Greedy strategy (like Hu's)
- General graphs (unlike Hu's)
- Priority list heuristics
  - Longest path to sink
  - Longest path to timing constraint

Example





Resource bounds:

3 multipliers with delay 2

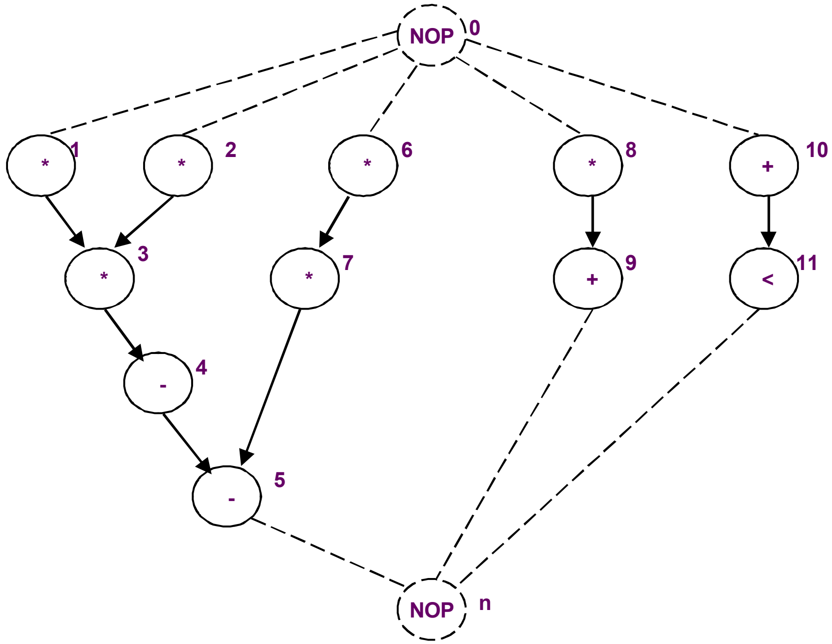
1 ALU with delay 1

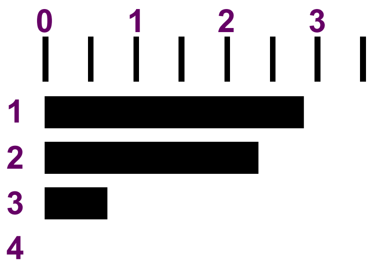
Force-directed scheduling definitions

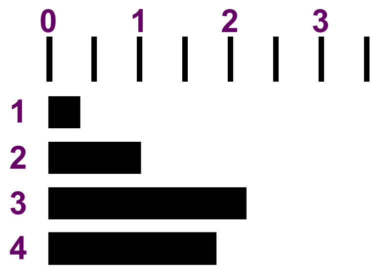
- **Operation *interval*:**
  - **Mobility plus one (μ_i +1)**
  - **Computed by ASAP and ALAP scheduling [t^S , t^L]**
- **Operation *probability* p_i *(l)*:**
  - **Probability of executing in a given step**

    ***1/ (μ_i + 1)* inside interval; *0* elsewhere**
- ***Operation-type* distribution q_k *(l)*:**
  - **Sum of the operation probabilities for each type**

Example







- Distribution graphs for multiplier and ALU

Force

- **Used as *priority* function**
- **Force is related to concurrency:**
  - **Sort operations for least force**
- **Mechanical analogy:**
  - **Force = constant x displacement**
    - **Constant = operation-type distribution**
    - **Displacement = change in probability**

Forces related to the assignment of an operation to a control step

- **Self-force:**
  - Sum of forces to feasible schedule steps
  - Self-force for operation v_i in step l

$$\sum_{m \text{ in interval}} q_k(m)\,(\delta_{lm} - p_i(m))$$

- **Predecessor/successor-force:**
  - Related to the predecessors/successors
    - Fixing an operation timeframe restricts timeframe of predecessors/successors
    - Ex: Delaying an operation implies delaying its successors

Scheduling and chaining

- Consider propagation delays of resources not in terms of cycles
- Use scheduling to *chain* multiple operations in the same control step
- Useful technique to explore effect of *cycle-time* on area/latency trade-off
- Algorithms:
  - ILP, ALAP/ASAP, list scheduling

Summary

- Scheduling determines *area/latency* trade-off
- Intractable problem in general:
  - Heuristic algorithms
  - ILP formulation (small-case problems)
- Several heuristic formulations
  - List scheduling is the fastest and most used
  - Force-directed scheduling tends to yield good results
- Several extensisons
  - Chaining