SCHEDULING II

©*Giovanni De Micheli*

Stanford University

Scheduling under resource constraints

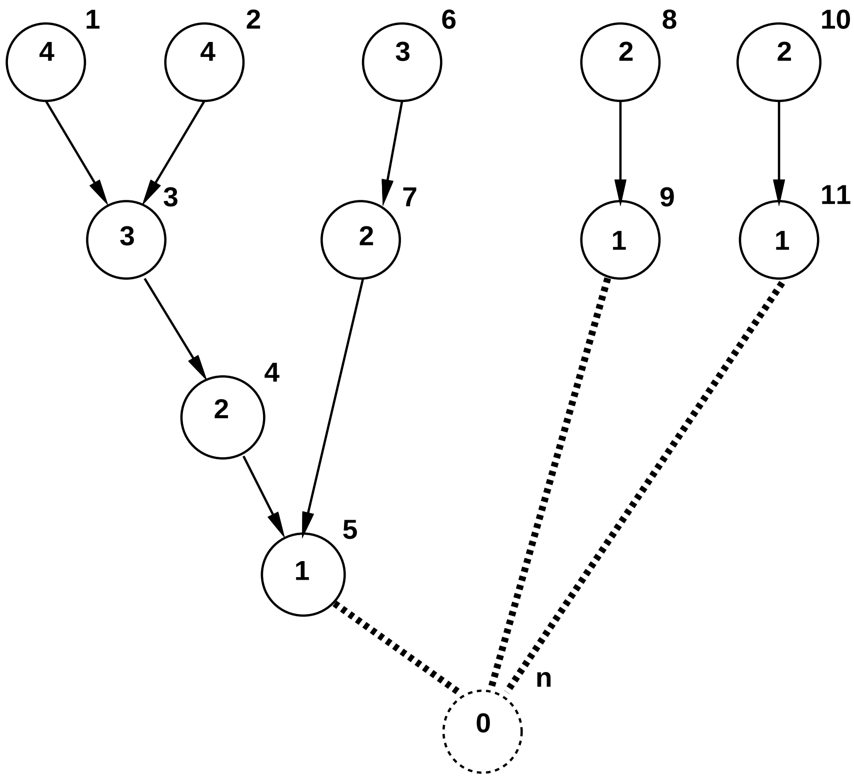
- Simplified models:
  - Hu's algorithm.

- Heuristic algorithms:
  - List scheduling.
  - Force-directed scheduling.

Hu's algorithm

- Assumptions:
  - Graph is a forest.
  - All operations have unit delay.
  - All operations have the same type.
- Algorithm:
  - Label vertices with distance from sink.
  - Greedy strategy.
  - Exact solution.

Example



- Assumptions:
  - One resource type only.
  - All operations have unit delay.

Algorithm
Hu's schedule with $\overline{a}$ resources

- Set step $l = 1$.

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

Example

- Minimum latency with $a = 3$ resources.
- Step 1: Select $\{v_1, v_2, v_6\}$.
- Step 2: Select $\{v_3, v_7, v_8\}$.
- Step 3: Select $\{v_4, v_9, v_{10}\}$.
- Step 4: Select $\{v_5, v_{11}\}$.

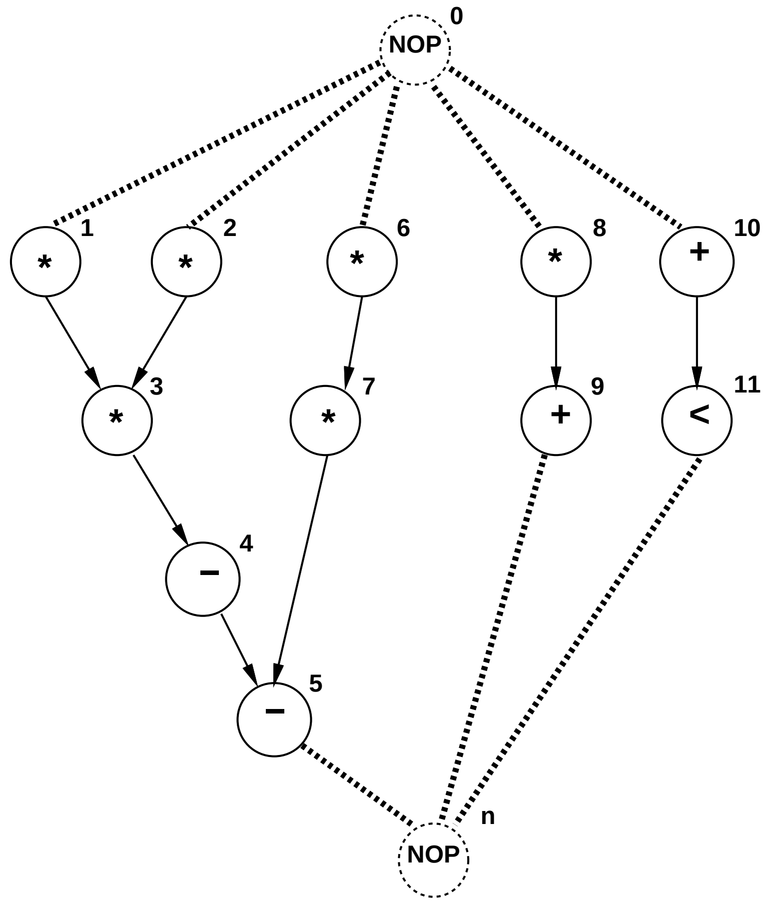
List scheduling algorithms

- 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.

List scheduling algorithm for minimum latency

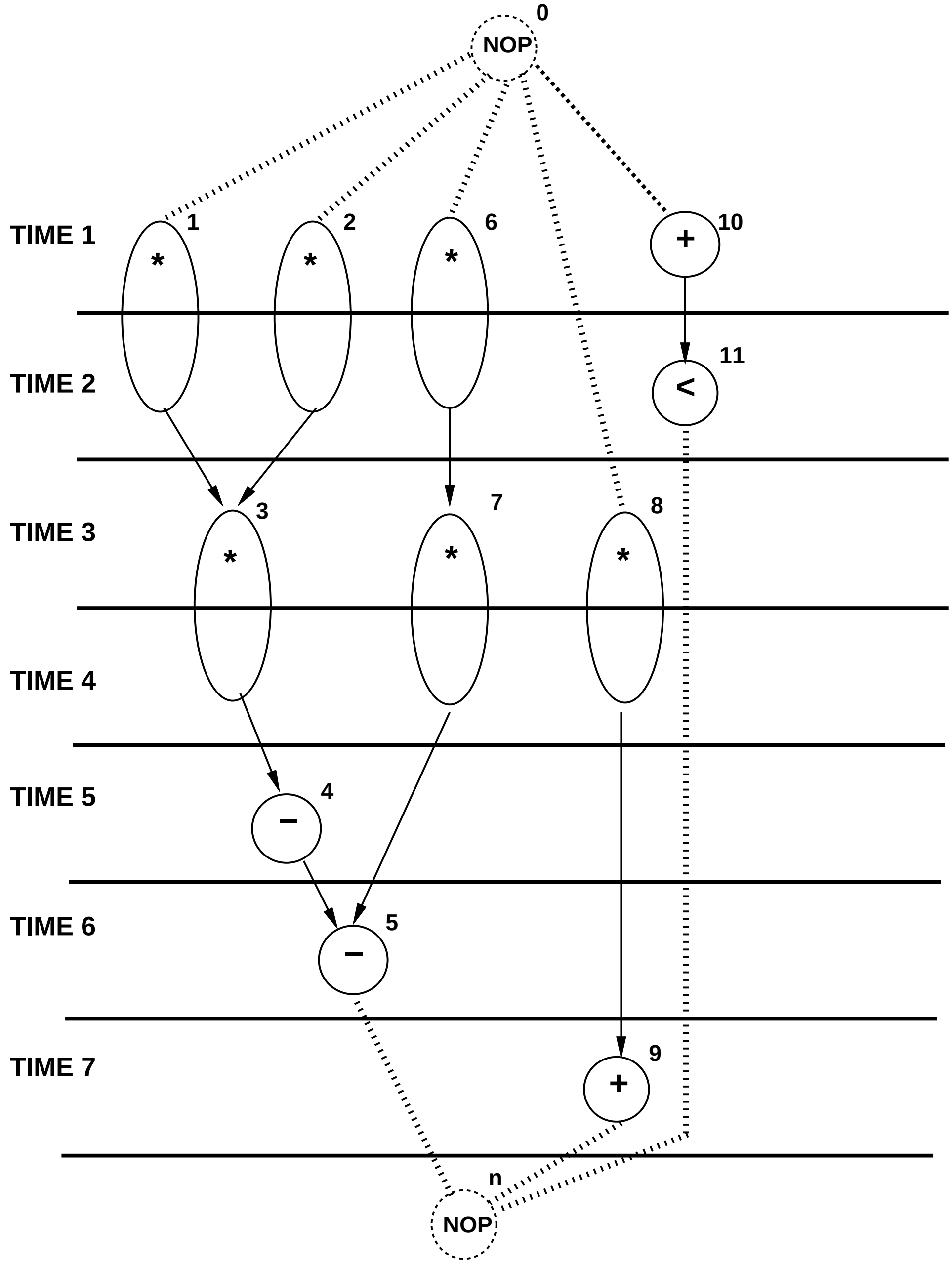

```
LIST_L( G(V,E), a ) {
    l = 1;
    repeat {
        for each resource type k = 1, 2, ... n_res {
            Determine candidate operations U_{l,k};
            Determine unfinished operations T_{l,k};
            Select S_k ⊆ U_{l,k} vertices, s.t. |S_k| + |T_{l,k}| ≤ a_k;
            Schedule the S_k operations at step l;
            }
        l = l + 1;
    }
    until (v_n is scheduled) ;
    return (t);
}
```

Example



- Assumptions:
  - $a_1 = 3$ multipliers with delay 2.
  - $a_2 = 1$ ALUs with delay 1.

Example

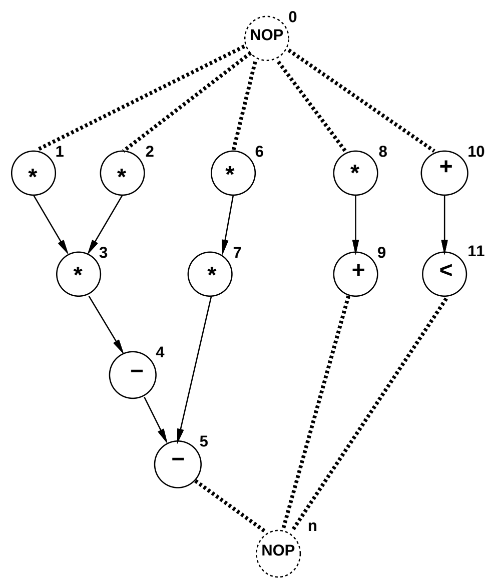

0
NOP
TIME 1
1
*
2
*
6
*
+
10
TIME 2
11
<
TIME 3
3
*
7
*
8
*
TIME 4
TIME 5
4
−
TIME 6
5
−
TIME 7
9
+
n
NOP

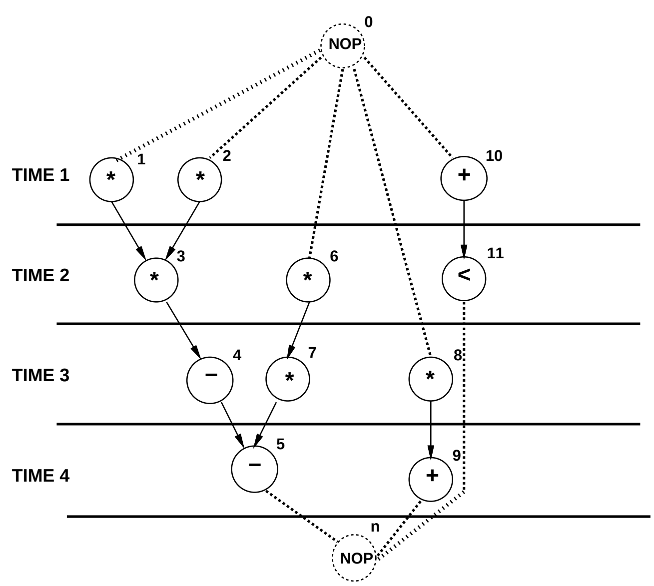
List scheduling algorithm for minimum resource usage

LIST_R($G(V,E),\overline{\lambda}$) {
    $\mathbf{a} = \mathbf{1}$;
    Compute the latest possible start times $\mathbf{t}^L$
    by ALAP ($G(V,E),\overline{\lambda}$);
    if ($t_0^L < 0$)
        return ($\emptyset$);
    $l = 1$;
    repeat {
        for each resource type $k = 1, 2, \ldots n_{res}$ {
            Determine candidate operations U_{lk};
            Compute the slacks $\{s_i = t_i^L - l \ \forall v_i \in U_{lk}\}$;
            Schedule the candidate operations
            with zero slack and update $\mathbf{a}$;
            Schedule the candidate operations
            that do not require additional resources;
        }
        $l = l + 1$;
    }
    until (v_n is scheduled) ;
    return ($\mathbf{t}, \mathbf{a}$);
}

Example

© GDM


NOP
0
1
2
6
8
10
3
7
9
11
4
5
n
NOP


NOP
0
TIME 1
TIME 2
TIME 3
TIME 4
1
2
10
3
6
11
4
7
8
5
9
n
NOP

Summary

- Scheduling determines *area/latency* trade-off.
- Intractable problem in general:
  - Heuristic algorithms.
  - ILP formulation (small-case problems).
- Chaining:
  - Incorporate *cycle-time* considerations.