SCHEDULING

©*Giovanni De Micheli*

Stanford University

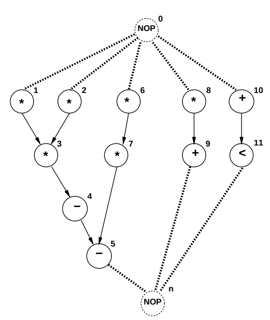
Outline

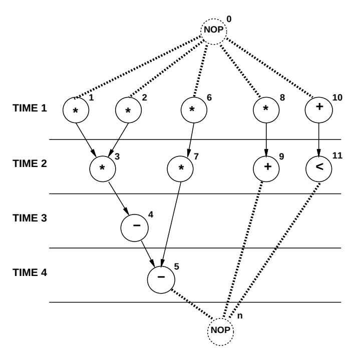
- The scheduling problem.
- Scheduling without constraints.
- Scheduling under timing constraints.
  - Relative scheduling.
- Scheduling under resource constraints.
  - The ILP model.
  - Heuristic methods.

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 model

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

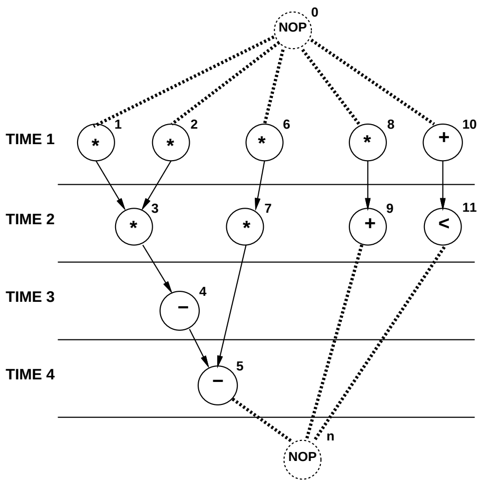
Minimum-latency unconstrained scheduling problem

- 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 $\varphi : V \to Z^+$, such that:
  - $t_i = \varphi(v_i)$,
  - $t_i \geq t_j + d_j \quad \forall\ i, j\ s.t.\ (v_j, v_i) \in E$
  - 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 pred. are all scheduled;
    Schedule v_i by setting $t_i^S = \max_{j:(v_j,v_i)\in E} t_j^S + d_j$;
  }
  until (v_n is scheduled) ;
  return ($\mathbf{t}^S$);
}

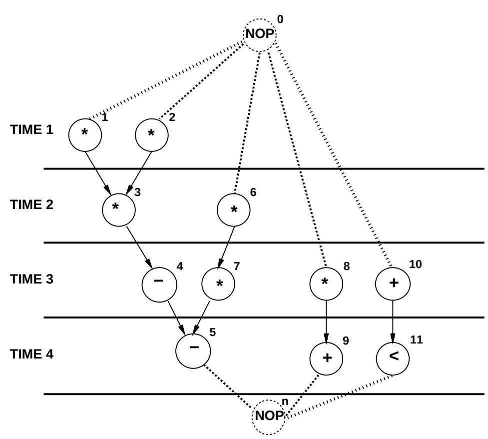
Example



ALAP scheduling algorithm

ALAP($G_s(V,E), \overline{\lambda}$) {
Schedule v_n by setting $t_n^L = \overline{\lambda} + 1$;
repeat {
Select vertex v_i whose succ. 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 ($\mathbf{t}^L$);
}

Example



Remarks

- ALAP solves a latency-constrained problem.
- Latency bound can be set to latency computed by ASAP algorithm.
- *Mobility*:
  - Defined for each operation.
  - Diff. 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:
  - $\{v_8, v_9, v_{10}, v_{11}\}$.

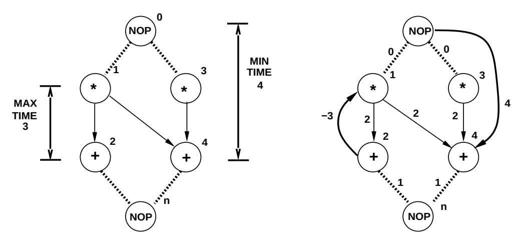
Scheduling under detailed timing constraints

- Motivation:
  - Interface design.
  - Control over operation start time.
- Constraints:
  - Upper/lower bounds on start-time difference of any operation pair.
- Feasibility of a solution.

Constraint graph model

- Start from sequencing graph.
- Model delays as weights on edges.
- Add forward edges for *minimum* constraints.
  - Edge (v_i, v_j) with weight $l_{ij} \Rightarrow t_j \geq t_i + l_{ij}$.
- Add backward edges for *maximum* constraints.
  - Edge (v_j, v_i) with weight:
    - $-u_{ij} \Rightarrow t_j \leq t_i + u_{ij}$
  - because $t_j \leq t_i + u_{ij} \Rightarrow t_i \geq t_j - u_{ij}$.

Example



Vertex	*Start time*
v_0	1
v_1	1
v_2	3
v_3	1
v_4	5
v_n	6

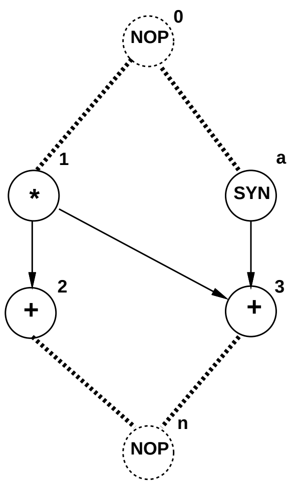
Methods for scheduling under detailed timing constraints

- Assumption:
  - All delays are fixed and known.
- Set of linear inequalities.
- *Longest path* problem.
- Algorithms:
  - Bellman-Ford, Liao-Wong.

Method for scheduling with unbounded-delay operations

- Unbounded delays:
  - Synchronization.
  - Unbounded-delay operations (e.g. loops).
- *Anchors*.
  - Unbounded-delay operations.
- Relative scheduling:
  - Schedule ops w.r. to the anchors.
  - Combine schedules.

Example



- $t_3 = \max\{t_1 + d_1; t_a + d_a\}$

Relative scheduling method

- For each vertex:
  - Determine *relevant anchor set* $R(\cdot)$.
  - Anchors affecting start time.
  - Determine time offset from anchors.
- Start-time:
  - Expressed by: $t_i = \max_{a \in R(v_i)} \{t_a + d_a + t_i^a\}$
  - Computed only at run-time because delays of anchors are unknown.

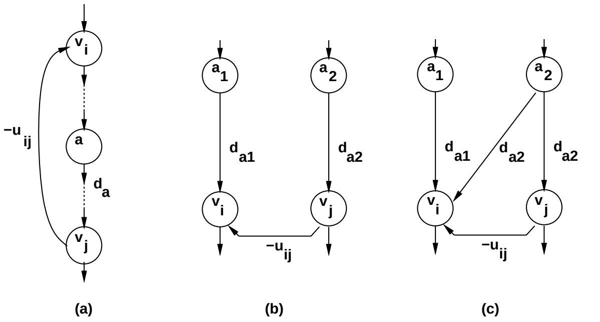
Relative scheduling under timing constraints

- Problem definition:
  - Detailed timing constraints.
  - Unbounded delay operations.
- Solution:
  - May or may not exist.
  - Problem may be ill-specified.

Relative scheduling under timing constraints

- Feasible problem:
  - A solution exists when unknown delays are zero.
- Well-posed problem:
  - A solution exists for any value of the unknown delays.
- Theorem:
  - A constraint graph can be made well-posed iff there are no cycles with unbounded weights.

Example

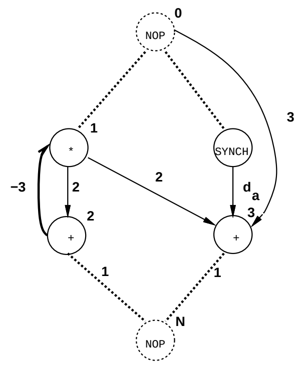


Relative scheduling approach

- Analyze graph:
  - Detect anchors.
  - Well-posedness test.
  - Determine dependencies from anchors.
- Schedule ops with respect to relevant anchors:
  - Bellman-Ford, Liao-Wong, Ku algorithms.
- Combine schedules to determine start times:
  - $t_i = \max_{a \in R(v_i)} \{t_a + d_a + t_i^a\} \quad \forall i$

Example

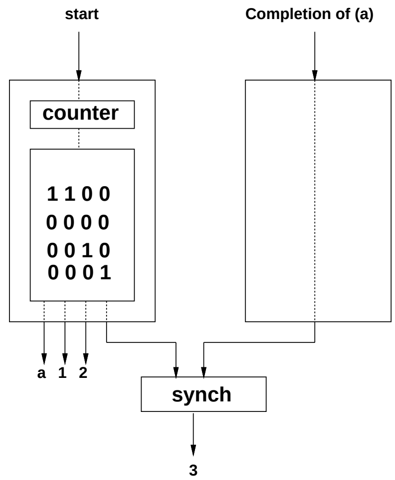
© GDM



Vertex v_i	Relevant Anchor Set $R(v_i)$	Offsets t_0	t_a
a	$\{v_0\}$	0	-
v_1	$\{v_0\}$	0	-
v_2	$\{v_0\}$	2	-
v_3	$\{v_0, a\}$	3	0

Example of control-unit

© GDM



Scheduling under resource constraints

© GDM

- Classical scheduling problem.
  - Fix area bound - minimize latency.
- The amount of available resources affects the achievable latency.
- Dual problem:
  - Fix latency bound - minimize resources.
- Assumption:
  - All delays bounded and known.

Minimum latency resource-constrained scheduling problem

© GDM

- Given a set of ops V with integer delays D a partial order on the operations E, and upper bounds $\{a_k; k = 1, 2, \ldots, n_{res}\}$:
- Find an integer labeling of the operations $\varphi : V \rightarrow Z^+$
- such that :
  - $t_i = \varphi(v_i)$,
  - $t_i \geq t_j + d_j \ \forall \ i, j \ s.t. \ (v_j, v_i) \in E$,
  - $|\{v_i | \mathcal{T}(v_i) = k$ and $t_i \leq l < t_i + d_i\}| \leq a_k$ $\forall$types $k = 1, 2, \ldots, n_{res}$ and $\forall$ steps l
  - and t_n is *minimum*.

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:
  - $X = \{x_{il}; i = 1, 2, \ldots, n; l = 1, 2, \ldots, \overline{\lambda} + 1\}$.
  - x_{il}, is TRUE only when operation v_i starts in step l of the schedule (i.e. $l = t_i$).
  - $\overline{\lambda}$ is an upper bound on latency.
- Start time of operation v_i:
  - $\sum_l l \cdot x_{il}$

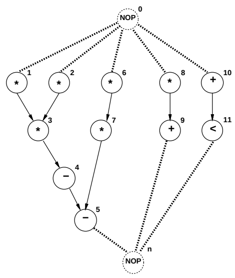
ILP formulation constraints

- Operations start only once.
  - $\sum_l x_{il} = 1 \quad i = 1, 2, \ldots, n$
- Sequencing relations must be satisfied.
  - $t_i \geq t_j + d_j \qquad \forall (v_j, v_i) \in E$
  - $\sum_l l \cdot x_{il} - \sum_l l \cdot x_{jl} - d_j \geq 0 \;\; \forall (v_j, v_i) \in E$
- Resource bounds must be satisfied.
  - Simple case (unit delay)
  - $\sum_{i:\mathcal{T}(v_i)=k} x_{il} \leq a_k \quad k = 1, 2, \ldots, n_{res}; \; \forall l$

ILP Formulation

$$
\begin{aligned}
\min \quad ||\mathbf{t}|| \quad & \textit{such that} \\
\sum_j x_{ij} \quad = \quad & 1 \;\; i = 1, 2, \ldots, n \\
\sum_l l \cdot x_{il} - \sum_l l \cdot x_{jl} - d_j \quad \geq \quad & 0 \;\; i, j = 1, 2, \ldots, n, (v_j, v_i) \in E \\
\sum_{i:\mathcal{T}(v_i)=k} \;\; \sum_{m=l-d_i+1}^{l} x_{im} \quad \leq \quad & a_k \;\; k = 1, 2, \ldots, n_{res}; l = 0, 1, \ldots, t_n
\end{aligned}
$$

Example

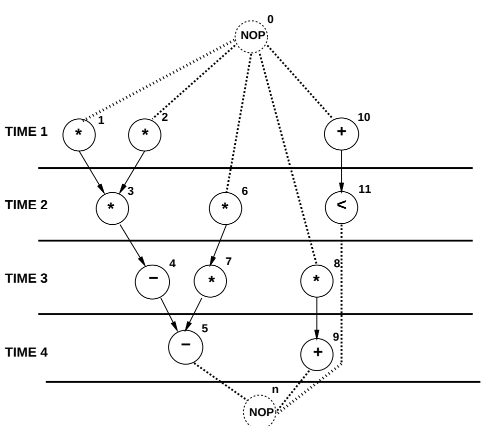


- Resource constraints:
  - 2 ALUs; 2 Multipliers.
  - $a_1 = 2; a_2 = 2.$
- Single-cycle operation.
  - $d_i = 1 \; \forall i.$

Example

- Operations start only once.
  - $x_{11} = 1$
  - $x_{61} + x_{62} = 1$
  - ...
- Sequencing relations must be satisfied.
  - $x_{61} + 2x_{62} - 2x_{72} - 3x_{73} + 1 \leq 0$
  - $2x_{92} + 3x_{93} + 4x_{94} - 5x_{N5} + 1 \leq 0$
  - ...
- Resource bounds must be satisfied.
  - $x_{11} + x_{21} + x_{61} + x_{81} \leq 2$
  - $x_{32} + x_{62} + x_{72} + x_{82} \leq 2$
  - ...

Example

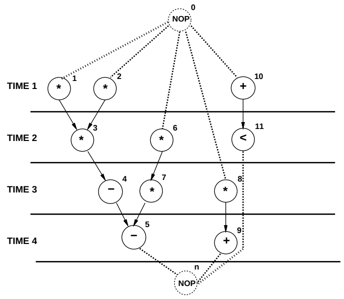


Dual ILP formulation

- Minimize resource usage under latency constraint
- Additional constraint:
  - Latency bound must be satisfied.
  - $\sum_l l \; x_{nl} \leq \overline{\lambda} + 1$
- Resource usage is unknown in the constraints.
- Resource usage is the objective to minimize.

Example

© GDM



- Multiplier area = 5. ALU area = 1.

- Objective function: $5a_1 + a_2$.

ILP Solution

© GDM

- Use standard ILP packages.

- Transform into LP problem [Gebotys].

- Advantages:
  - Exact method.
  - Others constraints can be incorporated.

- Disadvantages:
  - Works well up to few thousand variables.