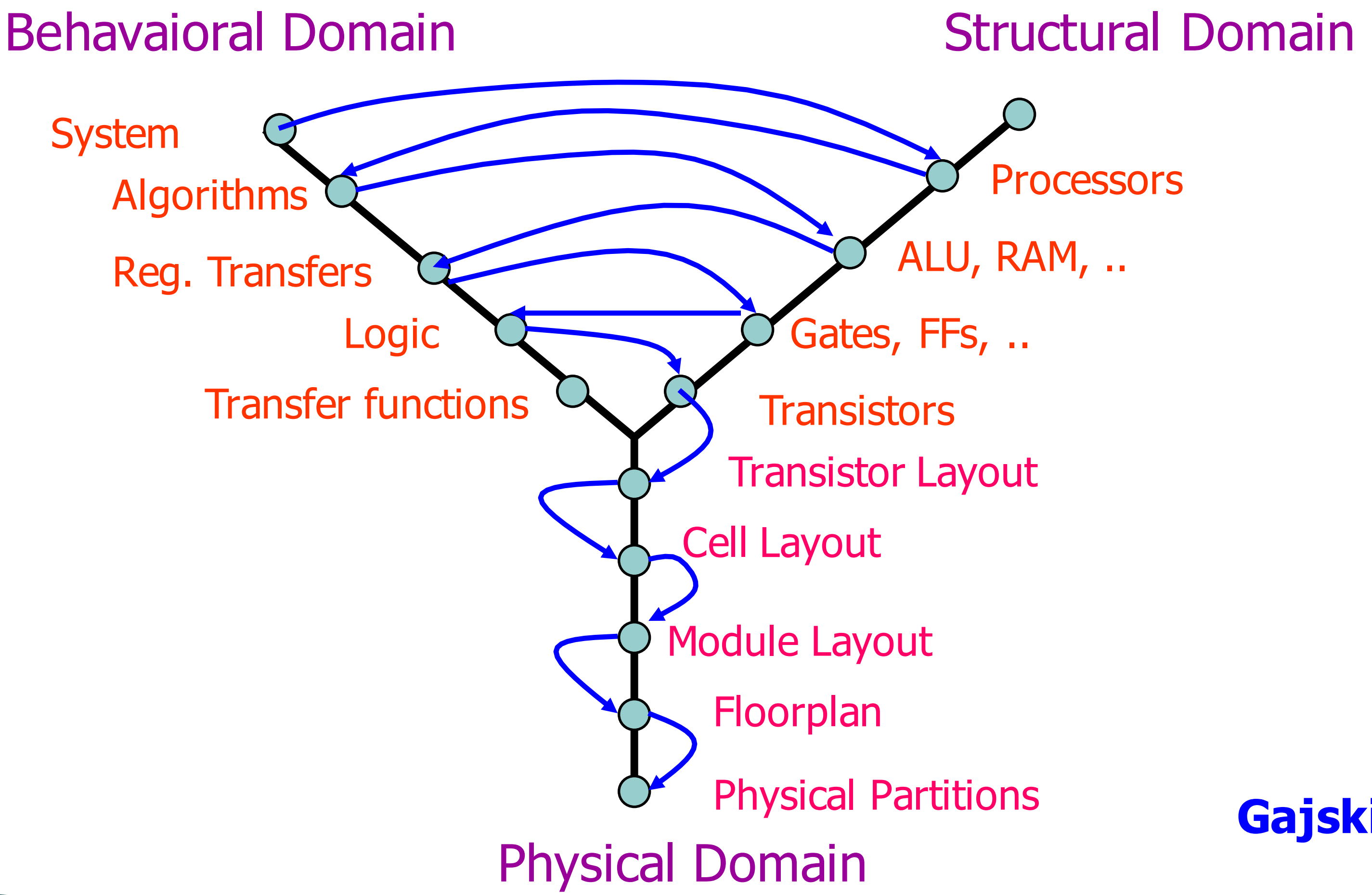
High-Level Synthesis -I

Virendra Singh
Indian Institute of Science
Bangalore
virendra@computer.org

E0-285: CAD of VLSI Systems

Design Domains (Y - Chart)



Gajski, 1982

Architectural Synthesis

Architectural Level Abstraction

- **Datapath**
- **Controller**

Architectural Synthesis

- **Constructing the macroscopic structure of a digital circuit starting from behavioural models that can be captured from Data flow or Sequencing Graph**

Architectural Synthesis

Objective

- Area
- Cycle time
- Latency
- Throughput

- **Worst case bound**
- **Evaluation**
- **Architectural Exploration**

Architectural Synthesis

Architectural synthesis tool can select an appropriate design point according to some user specific criterion and construct corresponding user specific Datapath and Controller

Circuit Specification for Architectural Synthesis

- Behavioural circuit model
- Details about resources being used and constraints
- Capture by Sequencing Graph

Architectural Synthesis

Resources

- Functional Resources
  - Primitive Resources
  - Application Specific Resources
- Memory Resources
- Interface Resources

Architectural Synthesis

Circuit Specification

- Sequencing Graph
- A set of functional resources, fully characterized in terms of area and execution delay
- A set of constraints

Architectural Synthesis

Computation: Differential Equation Solver

$Y'' + 3\,x\,y' + 3y = 0$

$X(0) = 0$

$y(0) = y$

$y'(0) = u$

```
Diffeq{
       read (x, y, u, dx, a)
       repeat{
            xl = x + dx
            ul = u – (3*x*u*dx) –
                        (3*y*dx)
            yl = y + (u*dx);
            c = xl < a
            x = xl; u = ul; y = yl;
        until (c);
        write (y)
  }
```

Architectural Synthesis

```
architecture BEHAVIOUR of DIFFEQ is
begin
process
            variable x, y, u, dx, a, xl, ul, yl: bit8;
begin
      wait until start'event and start = '1';
            x := x_port; y := y_port; a := a_port; u := u_port; dx := dx_port;
            DIFFEQ_LOOP:
            while ( x < a) loop
                         wait until clk'event and clk = '1';
                        xl = x + dx;
                        ul = u – (3*x*u*dx) – (3*y*dx);
                        yl = y + (u*dx);
                        x = xl; u = ul; y = yl;
            end loop DIFFEQ_LOOP;
            y_port := y;
end process
end BEHAVIOUR;
```

Architectural Synthesis

Computation: Differential Equation Solver

xl = x + dx

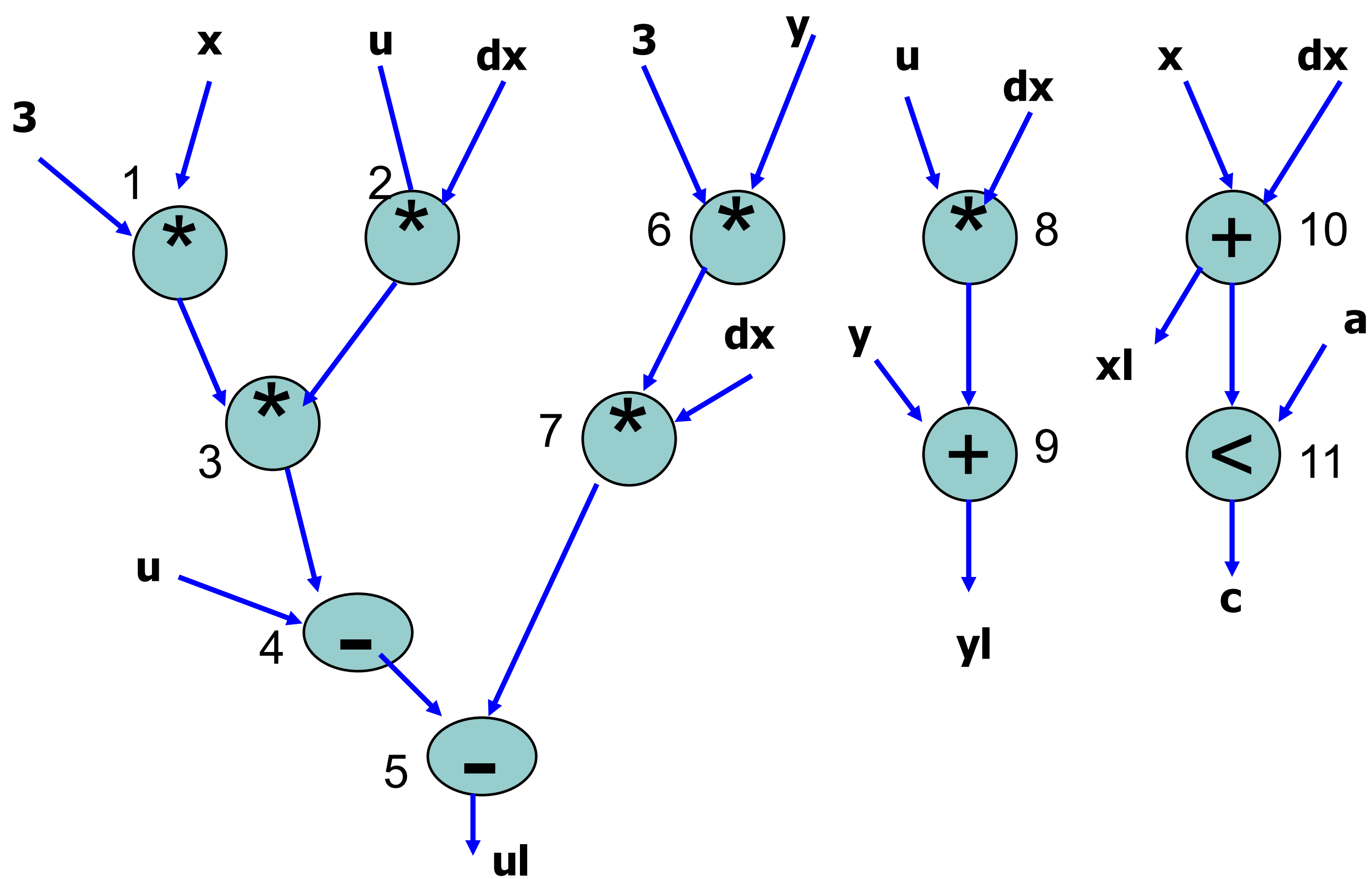
ul = u – (3*x*u*dx) – (3*y*dx)

yl = y + (u*dx);

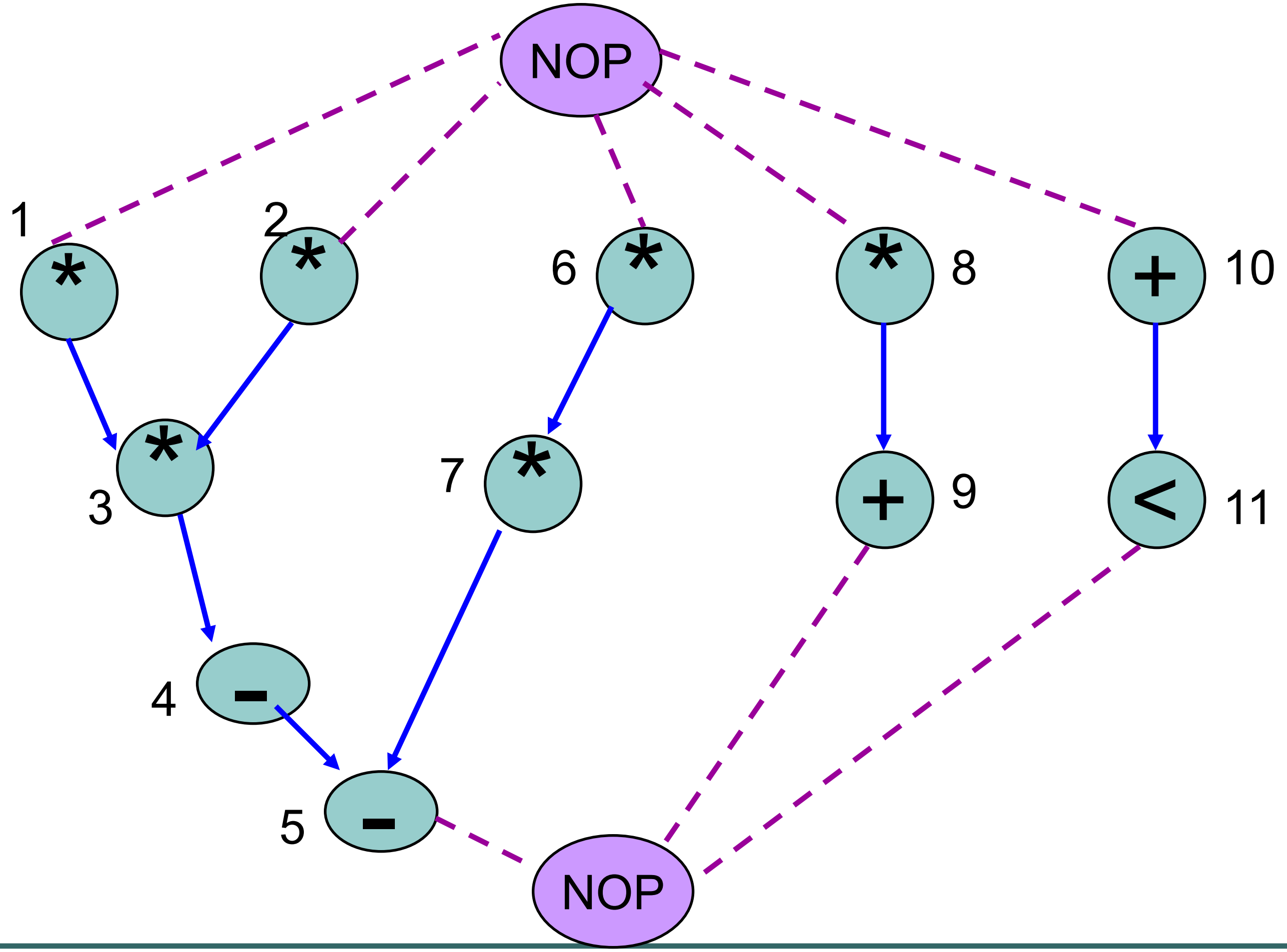
c = xl < a

Data Flow Graph (DFG): represent operation and data dependencies

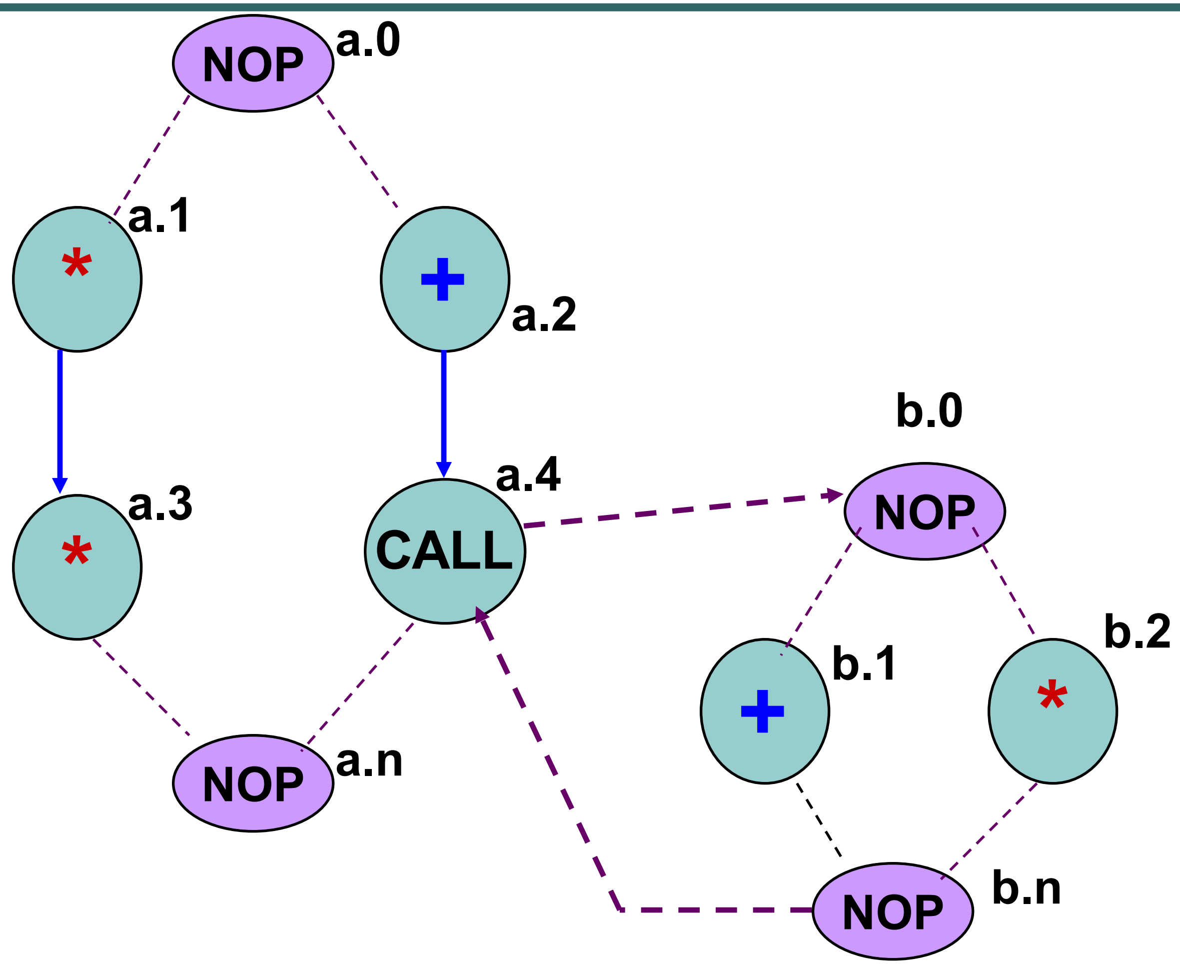
Data Flow Graph



Sequencing Graph



Hierarchical Sequencing Graph



Architectural Synthesis

Architectural Synthesis and optimization consists of two stages

1. Placing the operation in time and in space, i.e., determining their time interval of execution and binding to resources
2. Determining detailed interconnection of the datapath and the logic-level specifications of the control unit

Temporal Domain: Scheduling

Delay **D** = {d_i; i = 0,1, 2, ….. n}

Start time T ={t_i; i= 0, 1, …., n)

Scheduling: Task of determining the start timing, subject to preceding constraints specified by sequencing graph

Latency $\lambda = t_n - t_0$

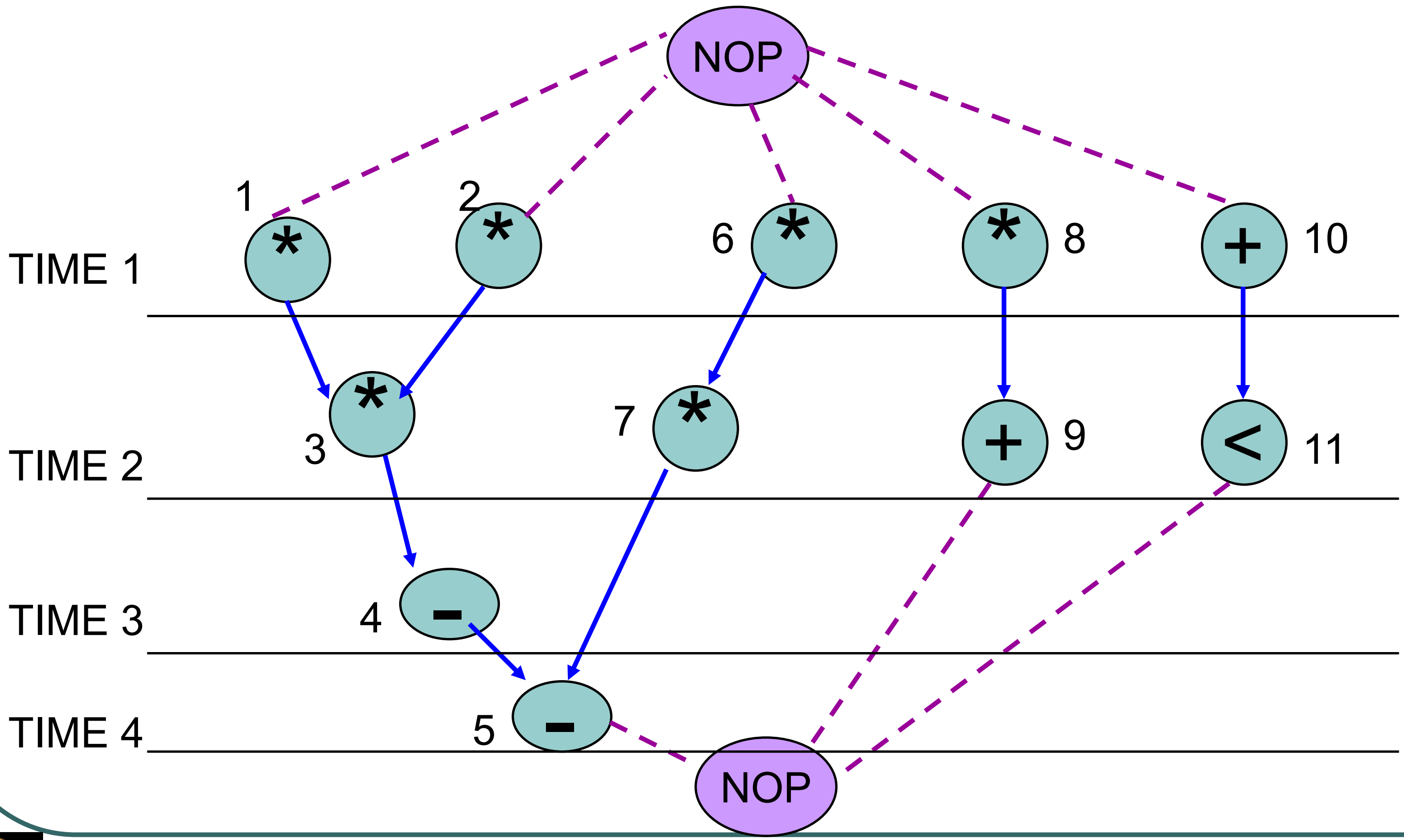
Temporal Domain: Scheduling

A scheduled sequencing graph is a vertex-weighted sequencing graph, where each vertex is labeled by its start time

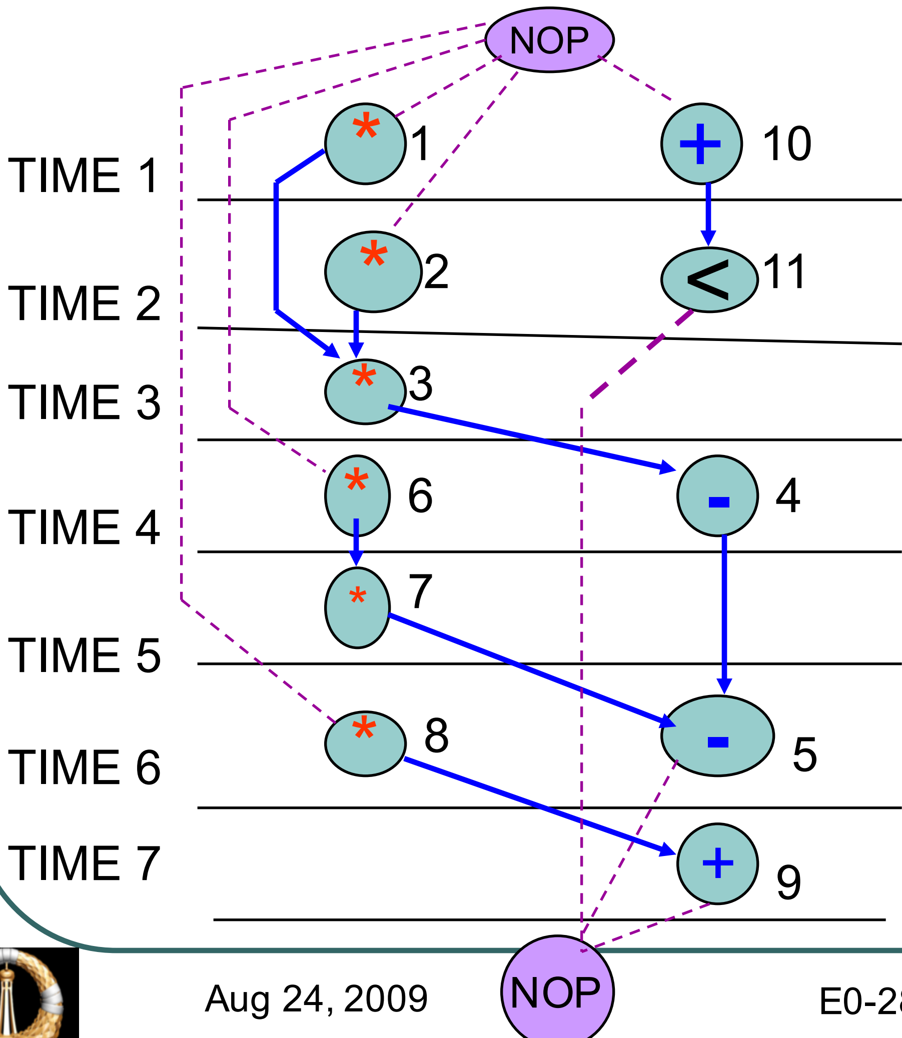
Operation	Start time
V1,V2, v6,v8, v10	1
V3, v7, v9,v11	2
V4	3
V5	4

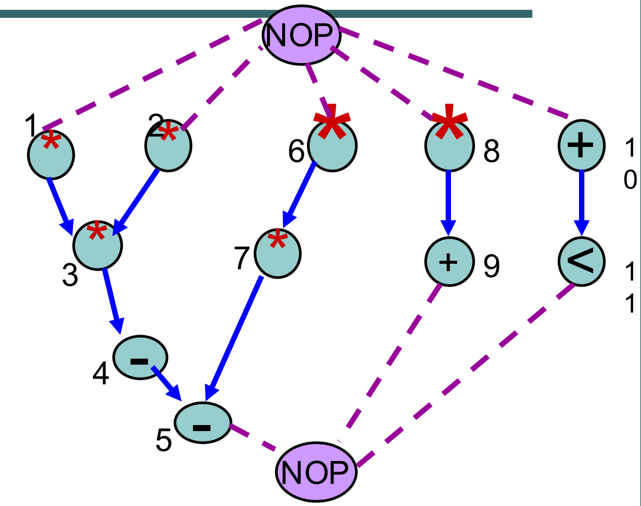
- **Chaining**

Temporal Domain: Scheduling



Temporal Domain: Scheduling





Spatial Domain: Binding

A fundamental concept that relates operation to resources is binding

- Resource types
- Resource sharing

Simple case of binding is a dedicated resources

Spatial Domain: Binding

$\beta(v1) = (1,1)$

$\beta(v2) = (1,2)$

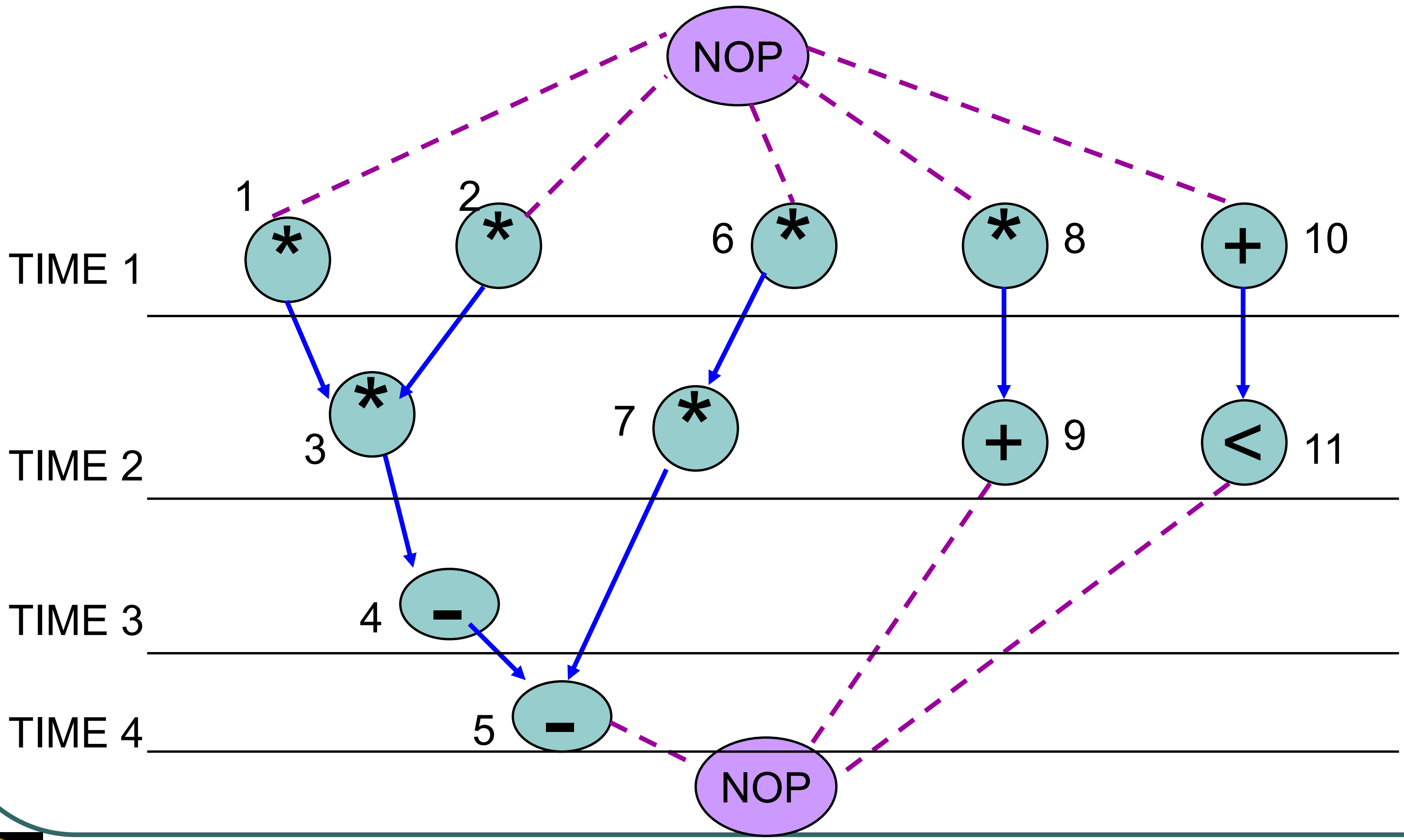
$\beta(v3) = (1,3)$

$\beta(v4) = (2,1)$

$\beta(v5) = (2,2)$

..

Spatial Domain: Binding

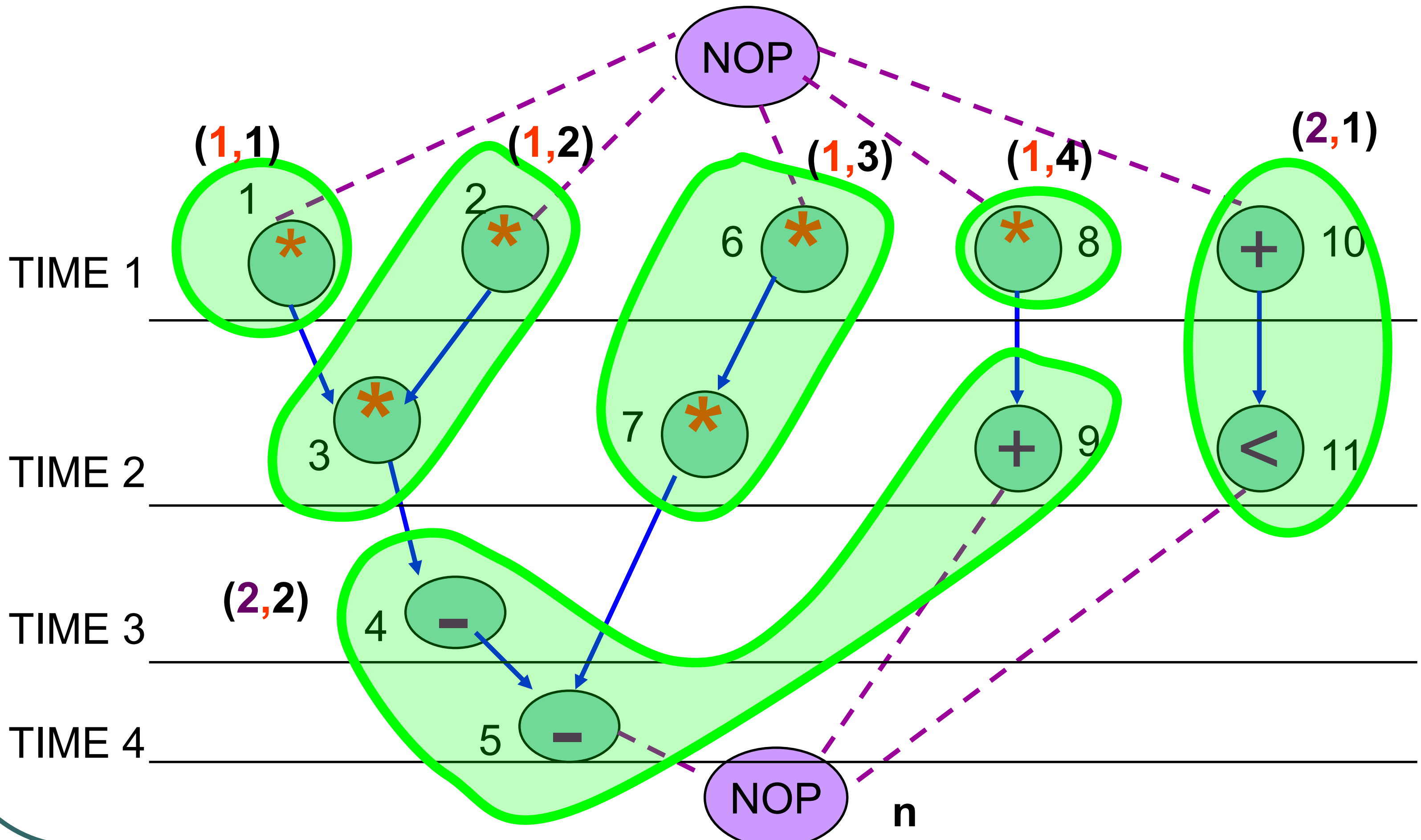


Spatial Domain: Binding

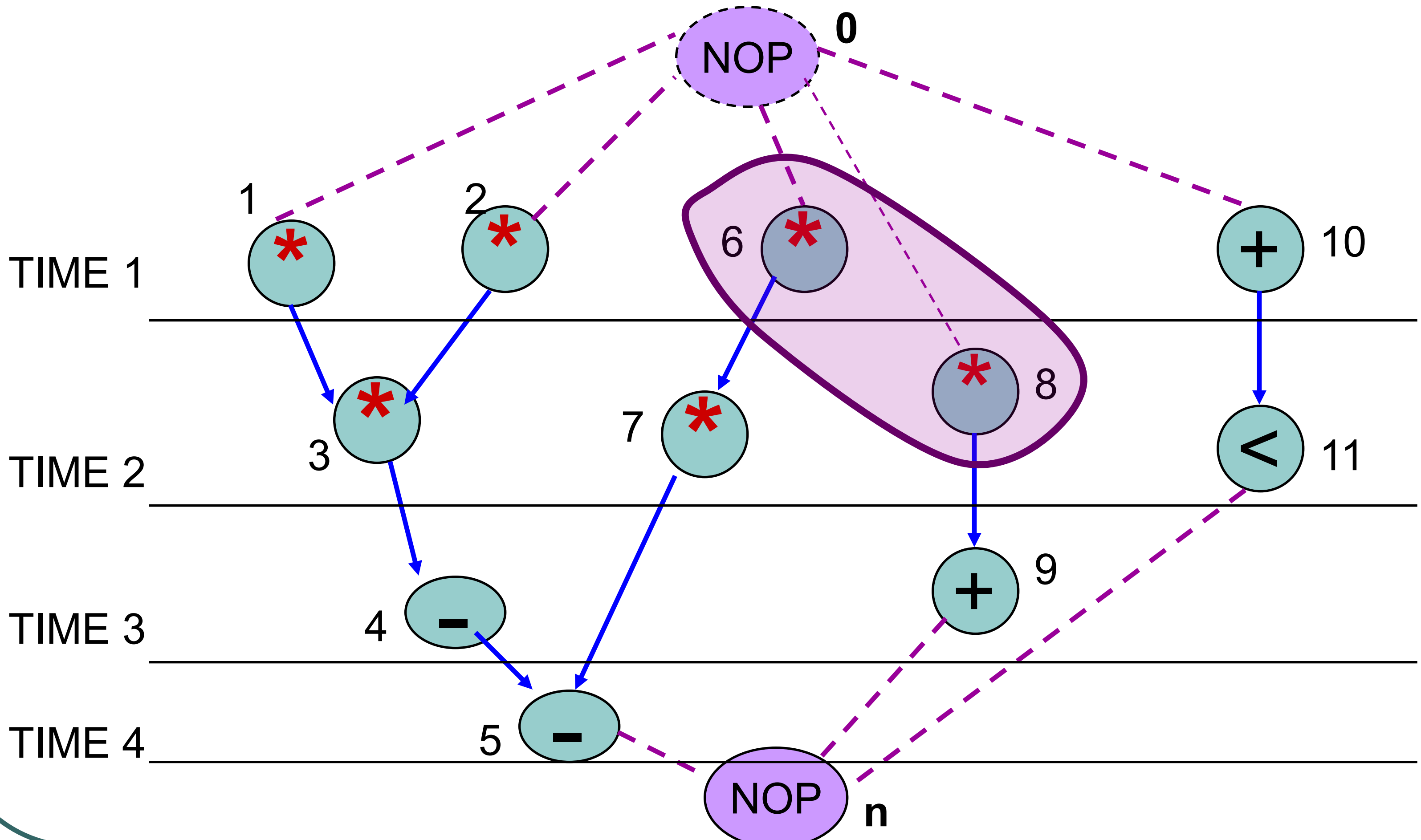
- A necessary condition for resource binding to produce a valid circuit implementation is that operation corresponding to the shared resource do not execute concurrently

- A resource binding can be represented by a labeled hyper-graph, where the vertex set V represents operations and the edge set E_β represents the binding of the operation to the resources

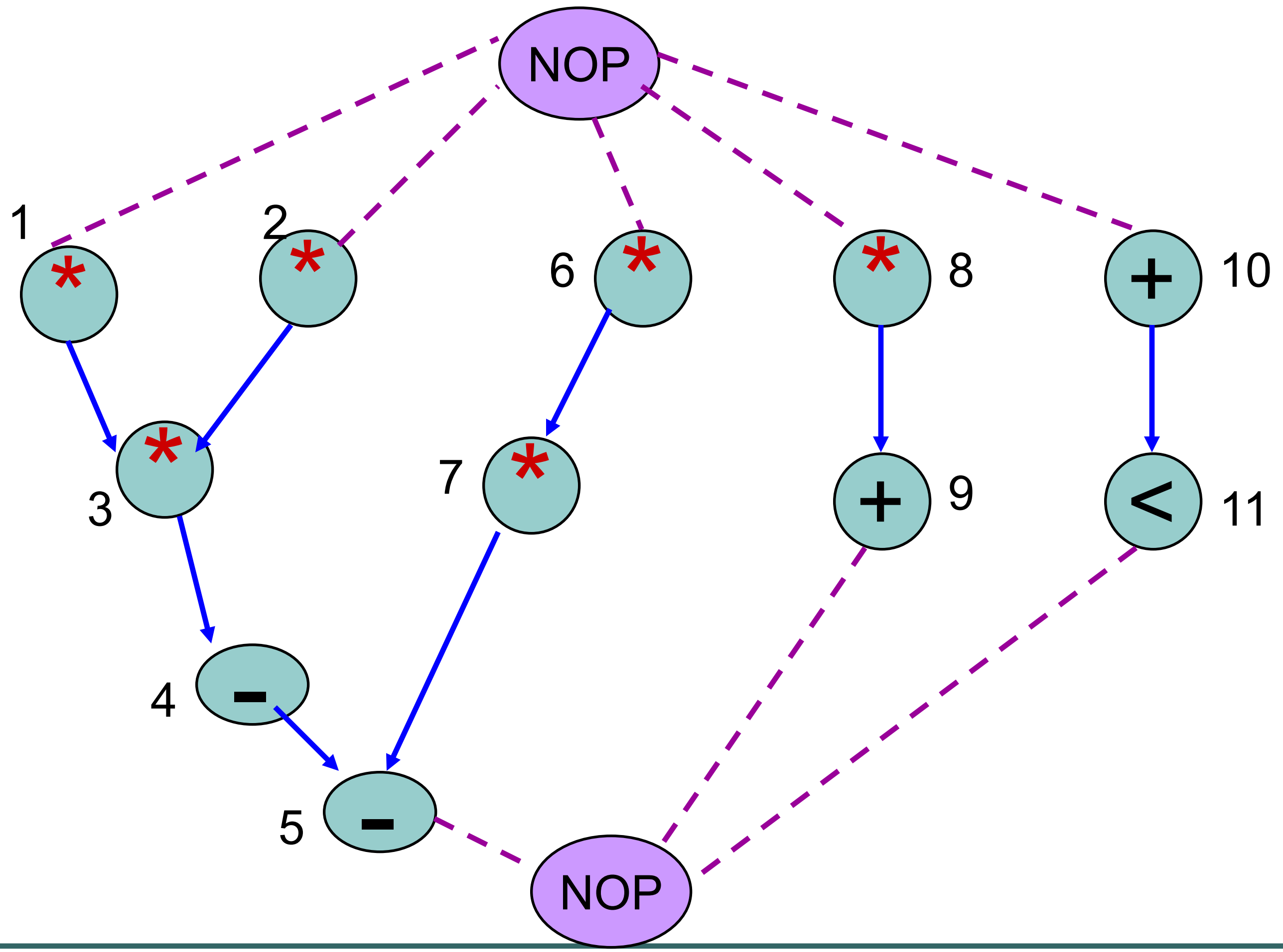
Spatial Domain: Binding



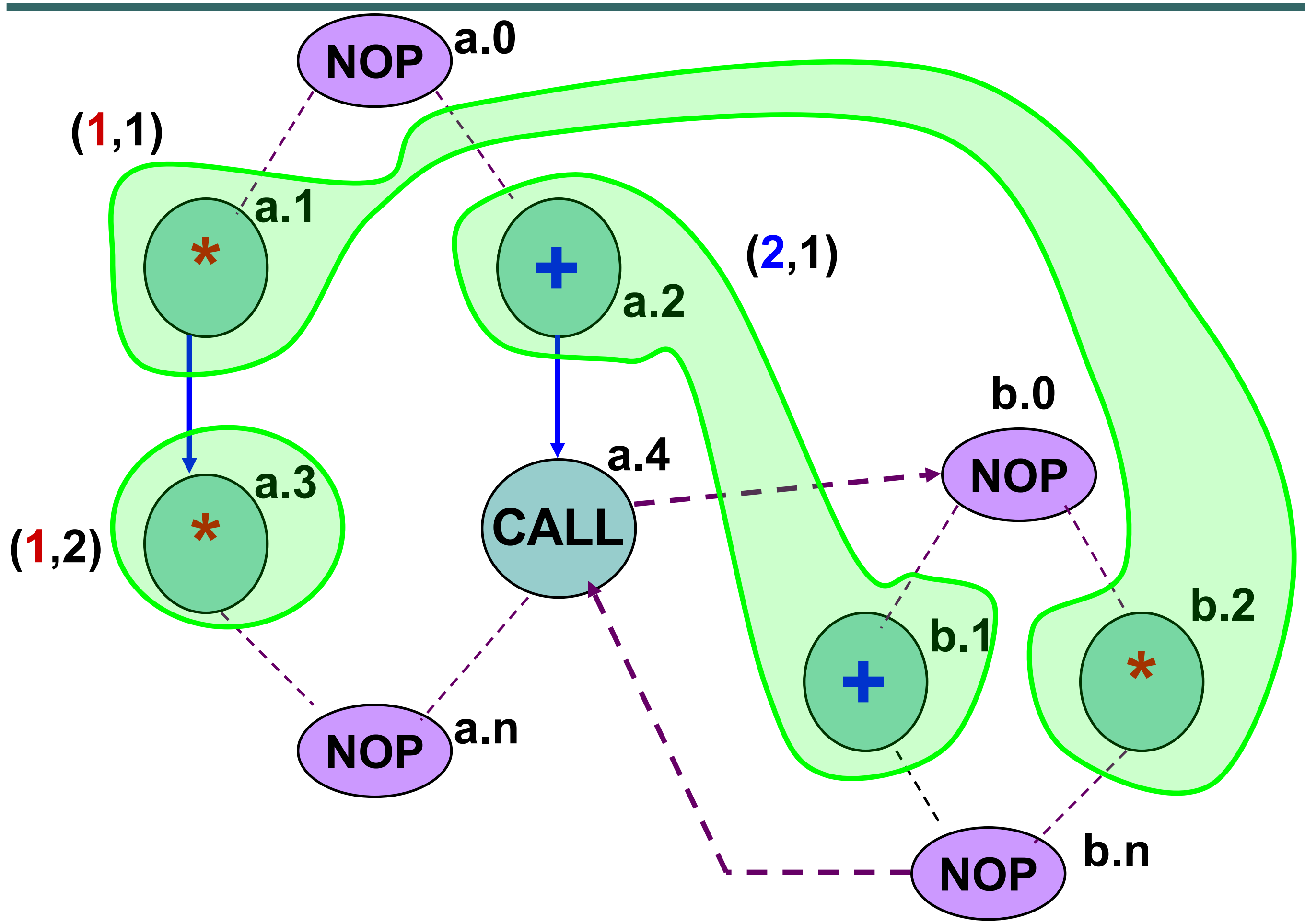
Spatial Domain: Binding



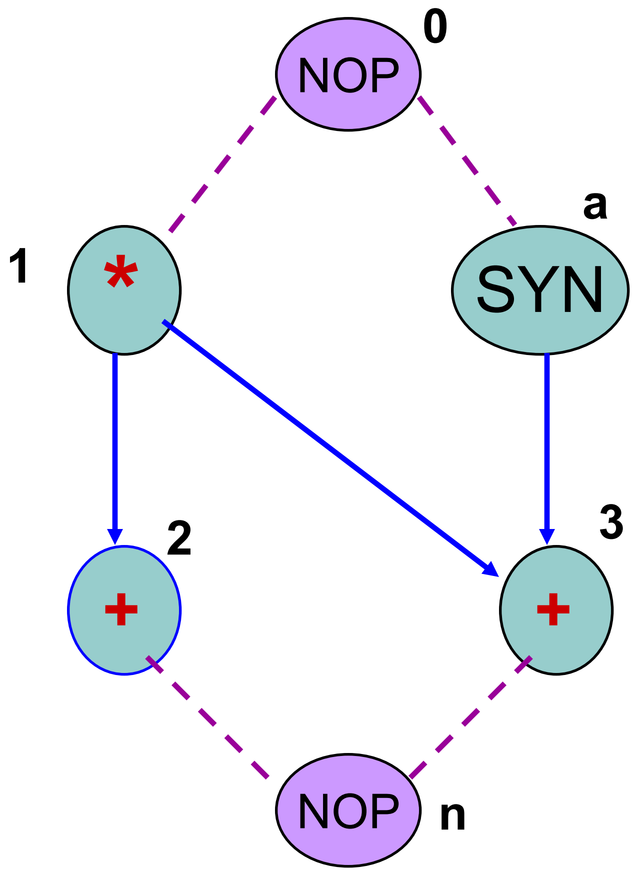
Sequencing Graph



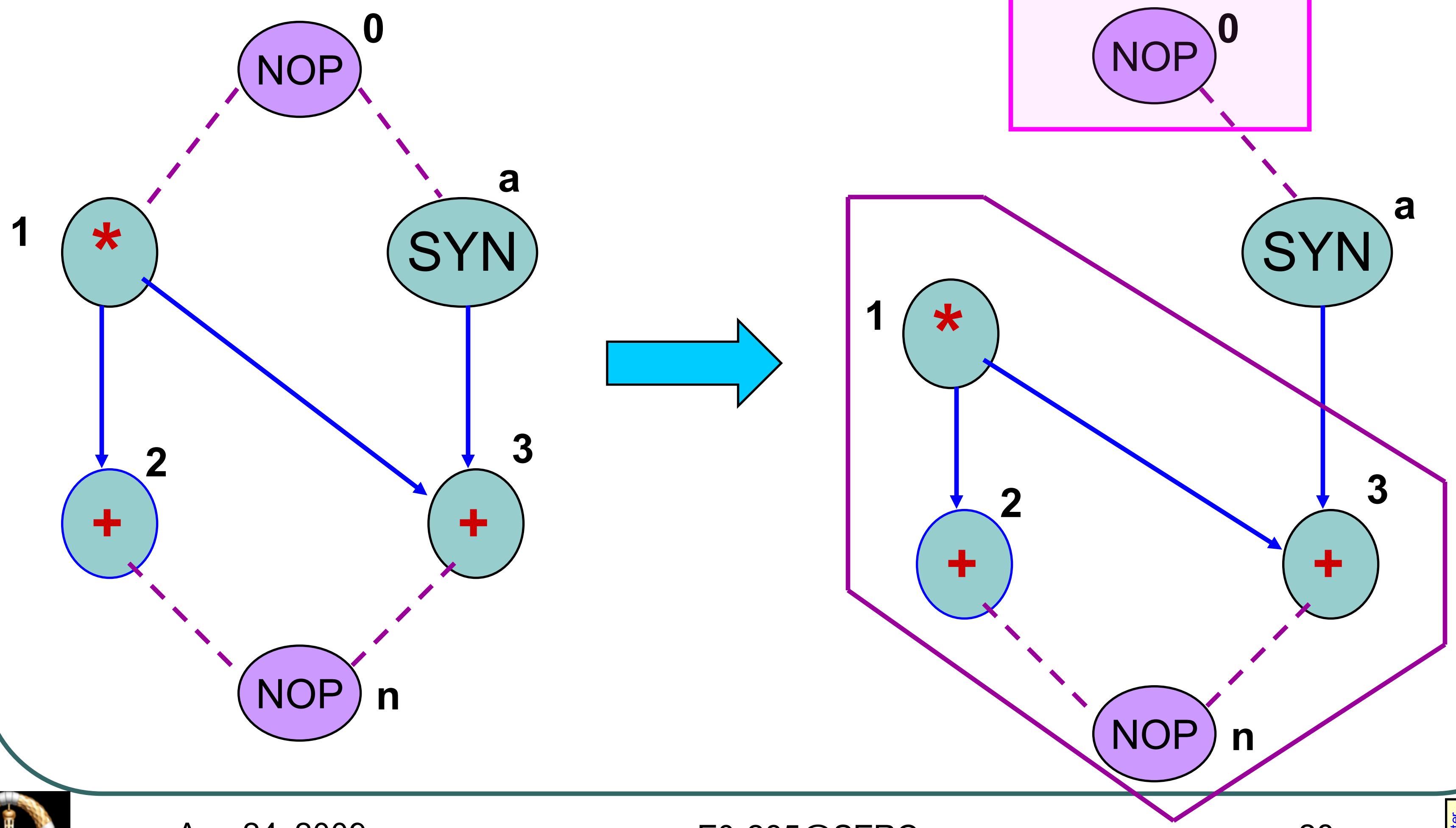
Hierarchical Sequencing Graph



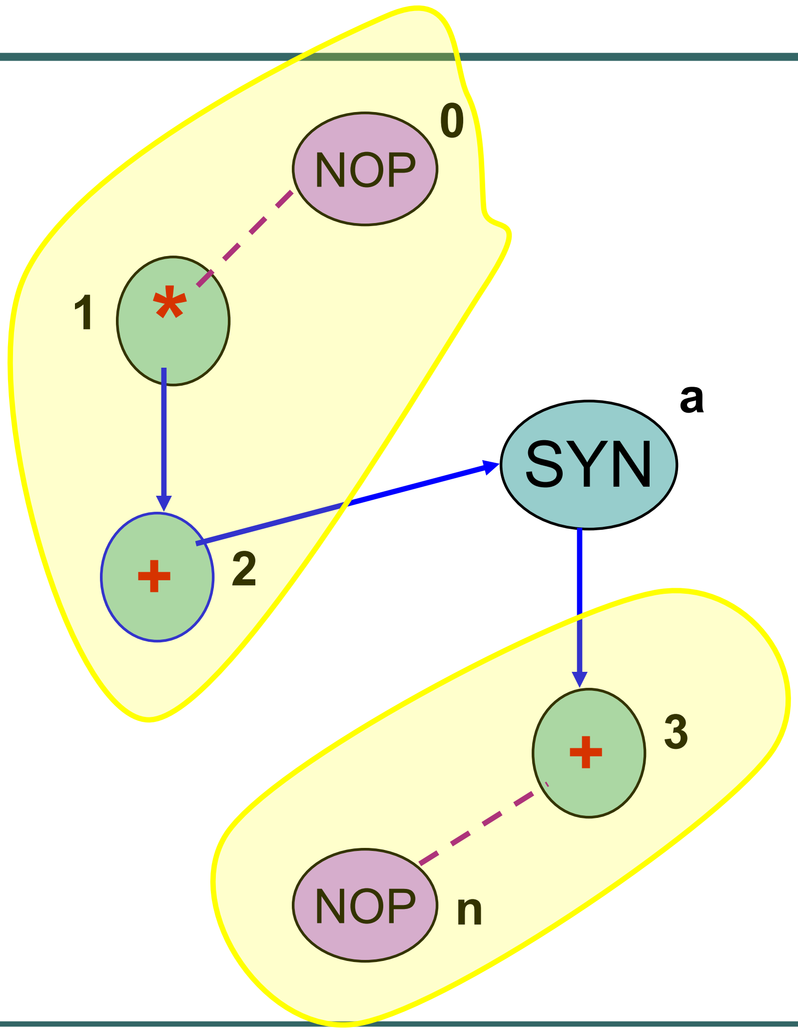
Synchronization



Synchronization



Synchronization



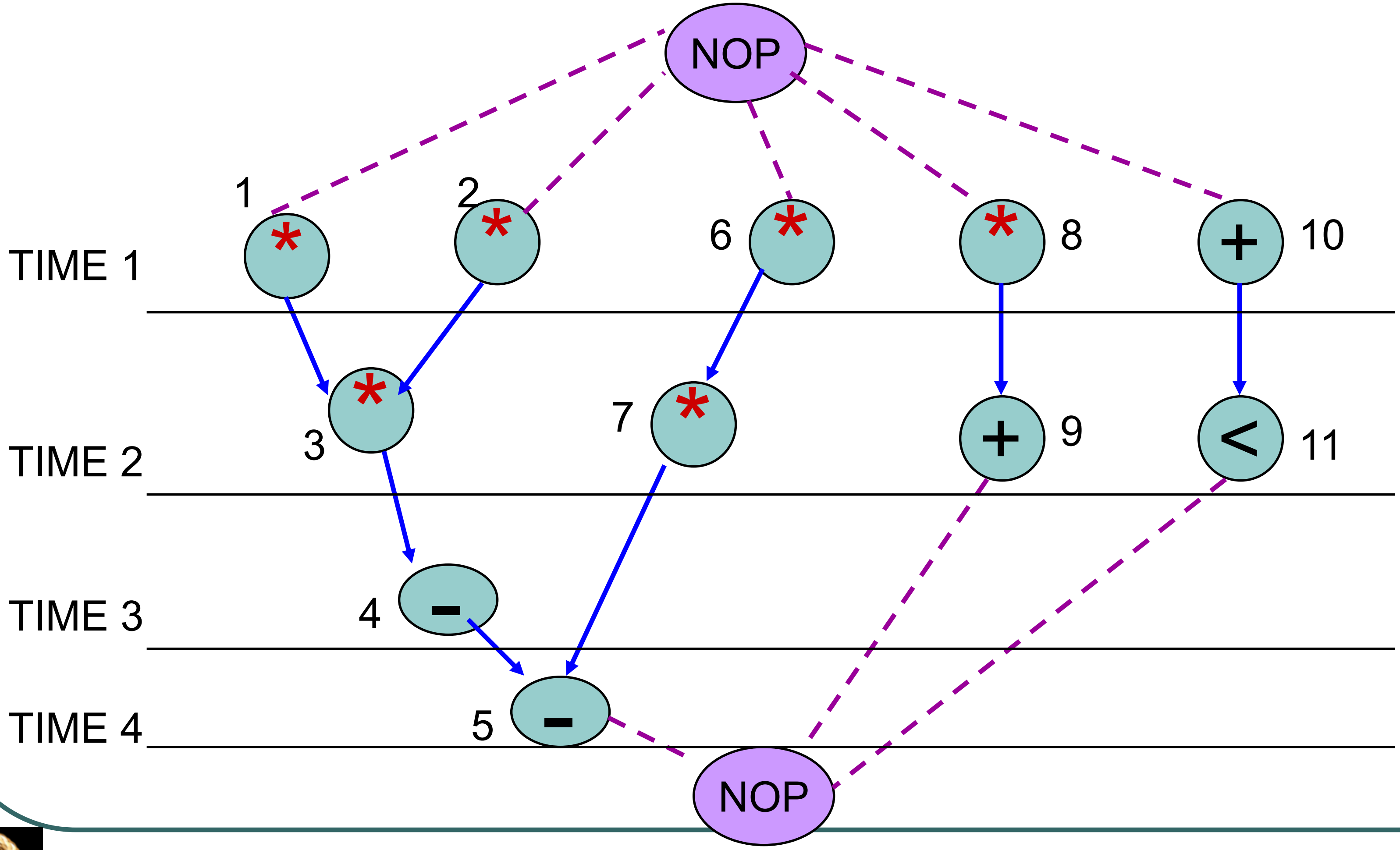
Area/Performance Estimation

Accurate area and performance estimation is not an easy task

Schedule: provides latency

Binding: provides information about the area

ASAP Scheduling



ASAP Scheduling

ASAP(G_s(V,E)){

Schedule v_0 by setting $t_0^s = 1$;

repeat{

select vertex v_i whose predecessors are all scheduled;

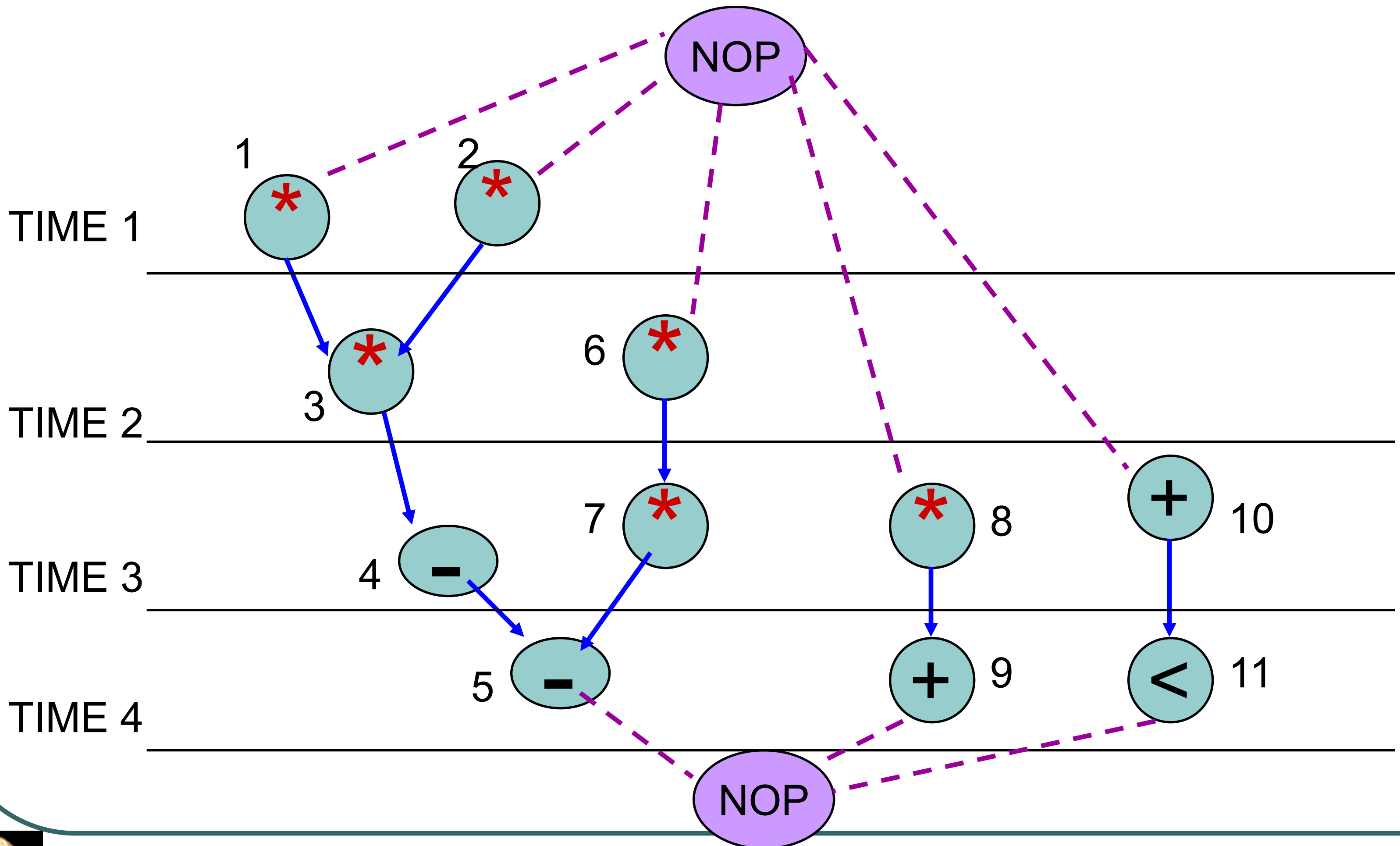
schedule v_i by setting $t_i^s = \max\{t_j^s + d_j\}$

} untill (v_n is scheduled)

return (t^s);

}

ALAP Scheduling



ALAP Scheduling

ALAP(G_s(V,E), λ){

Schedule v_n by setting $t_n^L = \lambda$;

repeat{

select vertex v_i whose successors are all scheduled;

schedule v_i by setting $t_i^L = \min\{t_j^L - d_j\}$

} untill (v_0 is scheduled)

return (t^L);

}

Scheduling with Resource Constraint

Scheduling under resource constraints

- **computing area/latency trade-off points**

Problems

- **Intractable problem**
- **Area-performance trade-off points are affected by the other factors - non-resource dominated circuits**

Scheduling with Resource Constraint

ILP Formulation

Binary decision variable $X = \{x_{il}\}$

1. Start time of each operation is unique

   $\Sigma_l \; x_{il} = 1$

2. Sequencing relations represented by $G_s(V,E)$ must be satisfied

   $\Sigma_l \; x_{il} \geq \Sigma_l \; xjl \; + d_j$

3. Resource bound must be met at every schedule step

   $\Sigma_k \; \Sigma_m \; x_{im} \leq a_k$

ILP Formulation

All operation must start only once

$$x_{0,1} = 1$$

$$x_{1,1} = 1$$

$$x_{2,1} = 1$$

$$x_{3,2} = 1$$

$$x_{4,3} = 1$$

$$x_{5,4} = 1$$

$$x_{6,1} + x_{6,2} = 1$$

$$x_{7,2} + x_{7,3} = 1$$

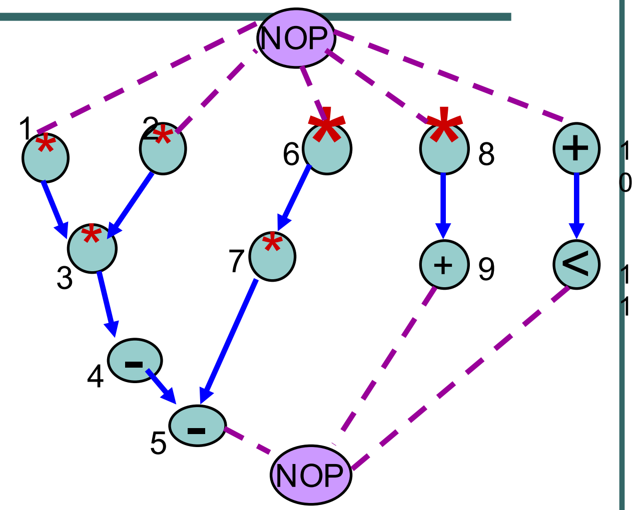
$$x_{8,1} + x_{8,2} + x_{8,3} = 1$$

$$x_{9,2} + x_{9,3} + x_{9,4} = 1$$

$$x_{10,1} + x_{10,2} + x_{10,3} = 1$$

$$x_{11,2} + x_{11,3} + x_{11,4} = 1$$

$$x_{n,5} = 1$$



ILP Formulation

Constraints – based on sequencing

(more than one starting time for at least one operation)

$2\,x_{7,2} + 3\,x_{7,3} - x_{6,1} - 2\,x_{6,2} - 1 \geq 0$

$2\,x_{9,2} + 3\,x_{9,3} + 4\,x_{9,4} - x_{8,1} - 2\,x_{8,2} - 3\,x_{8,3} - 1 \geq 0$

$2\,x_{11,2} + 3\,x_{11,3} + 4\,x_{11,4} - x_{10,1} - 2\,x_{10,2} - 3\,x_{10,3} - 1 \geq 0$

$4\,x_{5,4} - 2\,x_{7,2} - 3\,x_{7,3} - 1 \geq 0$

$5\,x_{n,5} - 2\,x_{9,2} - 3\,x_{9,3} - 4\,x_{9,4} - 1 \geq 0$

$5\,x_{n,5} - 2\,x_{11,2} - 3\,x_{11,3} - 4\,x_{11,4} - 1 \geq 0$

ILP Formulation

Resource Constraints

$x_{1,1} + x_{2,2} + x_{6,1} + x_{8,1} \leq 2$

$x_{3,2} + x_{6,2} + x_{7,2} + x_{8,2} \leq 2$

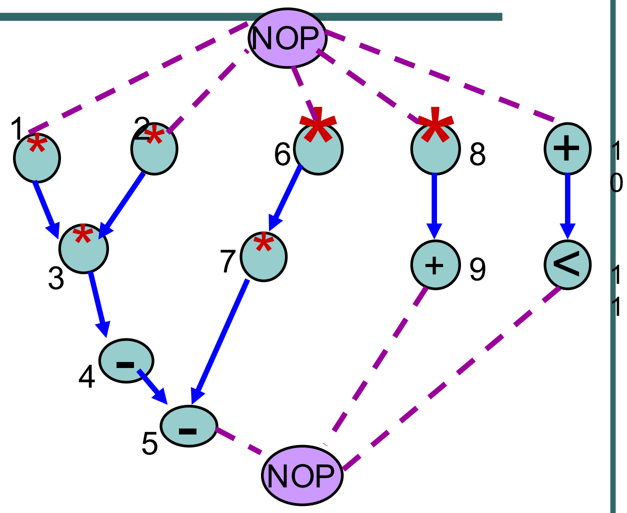
$x_{7,3} + x_{8,3} \leq 2$

$x_{10,1} \leq 2$

$x_{9,2} + x_{10,2} + x_{11,2} \leq 2$

$x_{4,3} + x_{9,3} + x_{10,3} + x_{11,3} \leq 2$

$x_{5,4} + x_{9,4} + x_{11,4} \leq 2$

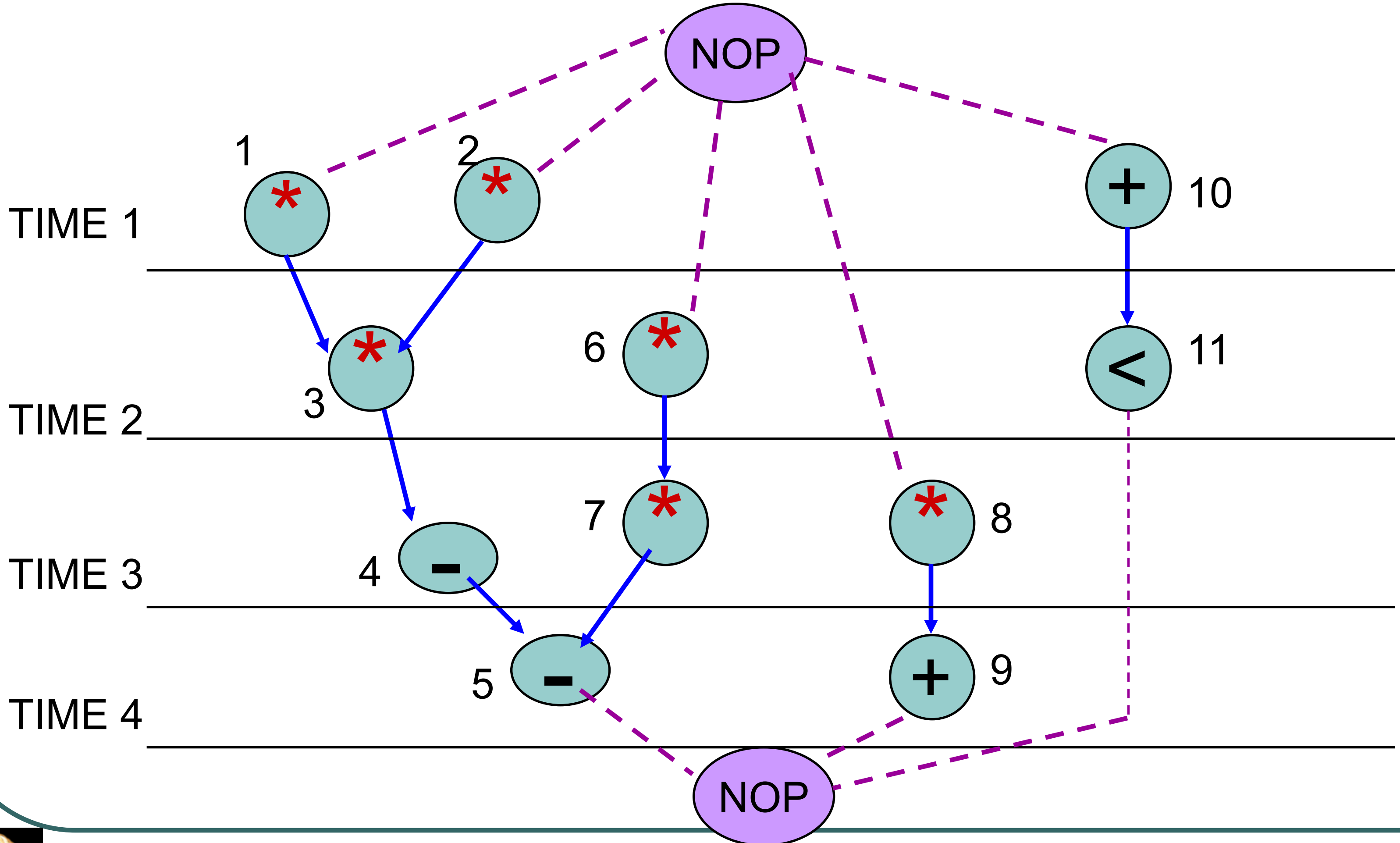


ILP Formulation

Optimize $\Sigma_i \Sigma_l \, l.x_{il}$

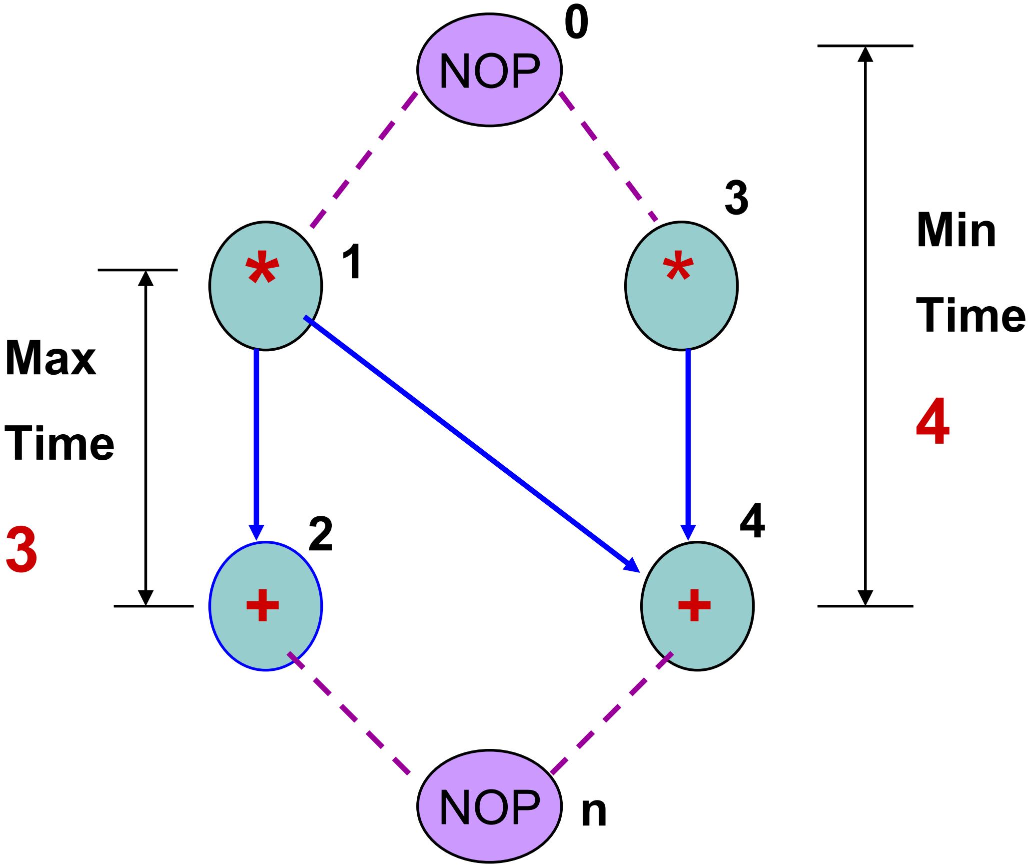
$$x_{6,1} + 2\,x_{6,2} + 3\,x_{7,2} + 3\,x_{7,3} + x_{8,1} + 2\,x_{8,2} + 3\,x_{8,3}$$
$$+ 2\,x_{9,2} + 3\,x_{9,3} + 4\,x_{9,4} + x_{10,1} + 2\,x_{10,2} + 3\,x_{10,3}$$
$$+ 2\,x_{11,2} + 3\,x_{11,3} + 4\,x_{11,4}$$

Optimum Scheduling under Resource Constraint

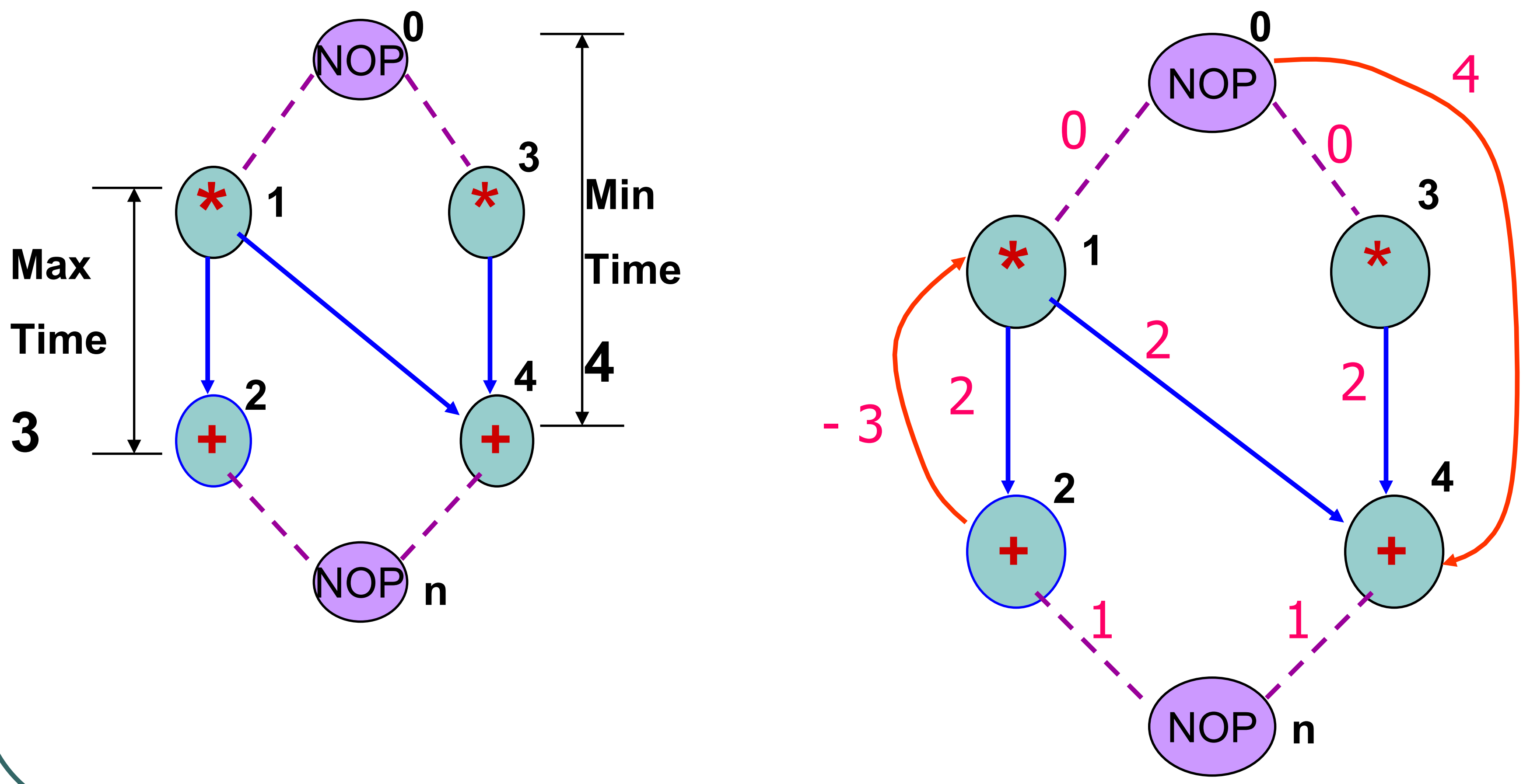


THANK YOU

Constraint Graph



Constraint Graph



Relative Scheduling

- Relative timing constraints are positive integeres specified for some operation pairs $v_i, v_j; i, j \in \{0,1,2, ..n\}$
  - A minimum timing constraints $l_{ij} \geq 0$ requires: $t_j \geq t_i + l_{ij}$
  - A maximum timing constraints $u_{ij} \geq 0$ requires: $t_j \leq t_i + u_{ij}$

Bellman Ford Algorithm

- BELLMAN_FORD (G(V,E,W)){
  - $S_0^1 = 0;$

  for (i=1 to n)
    - $S_i^1 = w_{0,i};$

  for (j=1 to n){

    for (i = 1 to n)
      - $s_i^{j+1} = \min_{k \neq i} \{s_i^j, (s_k^j + w_{k,i})\};$

      }

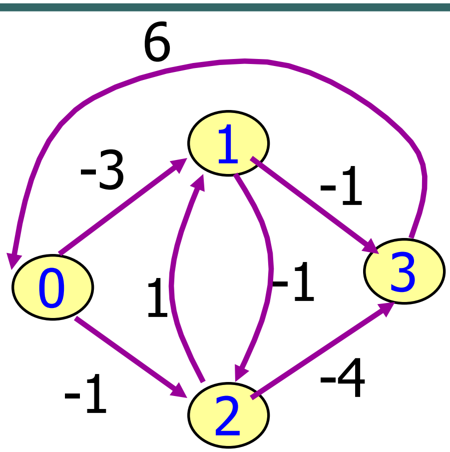
      - If ($s_i^{j+1} == s_i^j$, for all i) return (TRUE);

    }

    - Return (FALSE)

}

Bellman Ford Algorithm



First Iteration (j = 1)

S_0^2 = min $\{s_0^1, s_3^1 + w_{3,0}\}$ = $\{0, \infty+6\}$ = 0

S_1^2 = min $\{s_1^1, s_2^1 + w_{2,1}\}$ = $\{-3, -1-1\}$ = -3

S_2^2 = min $\{s_2^1, s_1^1 + w_{1,2}\}$ = $\{-1, -3+1\}$ = -2

S_3^2 = min $\{s_3^1, s_1^1 + w_{2,1}, s_2^1 + w_{2,3}\}$ = $\{\infty, -3-1, -1-4\}$ = -5

Initially

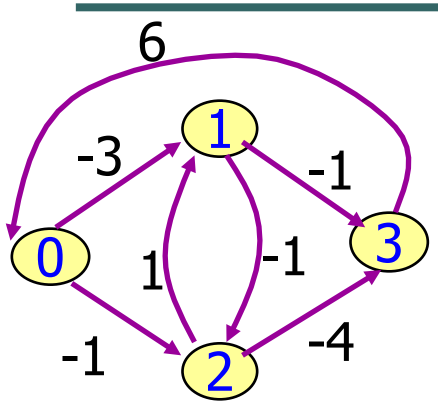
$S_0^1 = 0$

$S_1^1 = -3$

$S_2^1 = -1$

$S_3^1 = \infty$

Bellman Ford Algorithm



Second Iteration (j = 2)

$S_0^3 = \min\{s_0^2, s_3^2 + w_{3,0}\} = \{0, -5+6\} = 0$

$S_1^3 = \min\{s_1^2, s_2^2 + w_{2,1}\} = \{-3, -2-1\} = -3$

$S_2^3 = \min\{s_2^2, s_1^2 + w_{1,2}\} = \{-2, -3+1\} = -2$

$S_3^3 = \min\{s_3^2, s_1^2 + w_{2,1}, s_2^2 + w_{2,3}\} = \{-5, -3-1, -2-4\} = -6$

Third Iteration (j = 3)

$S_0^4 = \min\{s_0^3, s_3^3 + w_{3,0}\} = \{0, -6+6\} = 0$

$S_1^4 = \min\{s_1^3, s_2^3 + w_{2,1}\} = \{-3, -2-1\} = -3$

$S_2^4 = \min\{s_2^3, s_1^3 + w_{1,2}\} = \{-2, -3+1\} = -2$

$S_3^4 = \min\{s_3^3, s_1^3 + w_{2,1}, s_2^3 + w_{2,3}\} = \{-6, -3-1, -2-4\} = -6$

Relative Scheduling

Scheduling under unbounded delay

The anchors of a constraint graph G(V,E) consists of the source vertex v_0 and all vertices with unbounded delay

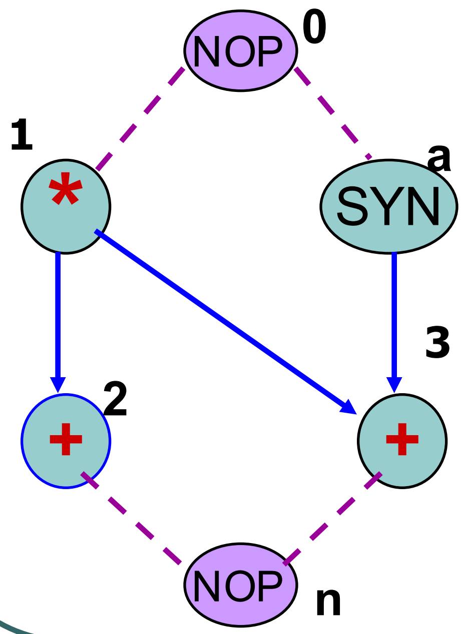
Redundant anchor

Relative Scheduling

- A defining path $\rho(a, v_i)$ from anchor a to vertex $v_i \in V$ is a path in $G_s(V,E)$ with one and only one unbounded weight
- The relevant anchor set of vertex vi Є V is the subset of anchors $R(v_i)$ s.t. $a \in R(v_i)$ if there exists a defining path $\rho(a, v_i)$
- An anchor a is redundant for vertex vi when there is another relevant anchor $b \in R(v_i)$ s.t. $|\rho(a, v_i)| = |\rho(a, b)| + |\rho(b, v_i)|$
- For a given vi V the irredundant relevant anchor set

Relative Scheduling

- For a given $v_i \in V$ the irredundant relevant anchor set $IR(v_i) \equiv R(v_i)$ represents the smallest subset of anchors that affects the start time of that vertex

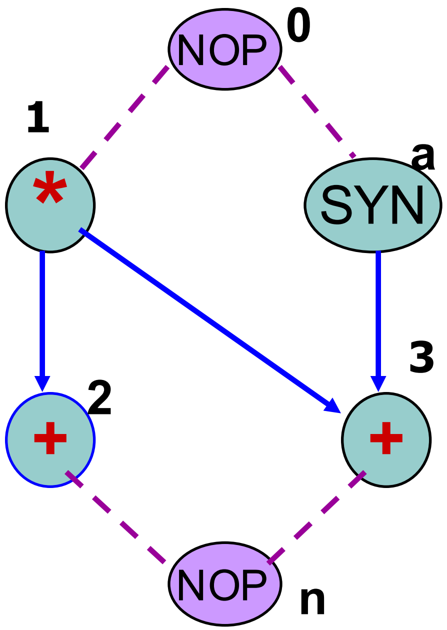


- Relevant anchor sets
  - $R(v_1) = \{v_0\}$
  - R(v2) = {v0}
  - R(v3) = {v0, va}
- These are corresponds to Irredundant anchor sets

Relative Scheduling

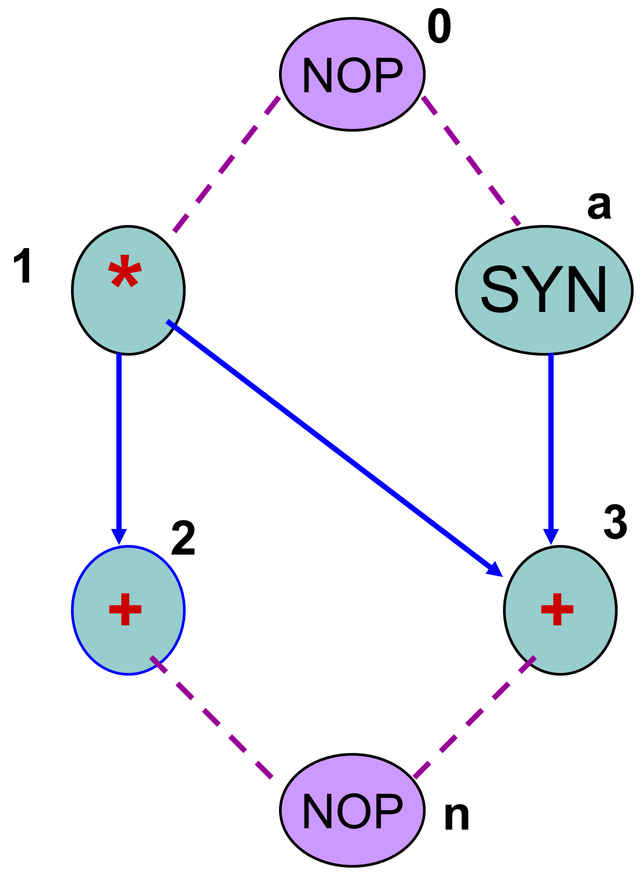
- The start time of the operation is defined on the basis of partial schedules relative to completion time of each anchors in their irredundant relevant anchor sets
- Let t_i^a be the schedule of operation v_i w.r.t anchor a
- $t_i = \max_{a \in IR(vi)} \{t_a + d_a + t^a_i\}$
- A relative schedule is collection of schedules w.r.t each anchor, or equivalently a set of offsets w.r.t the irredundant relevant anchors for each vertex

Relative Scheduling



Vertex	I.R.A.S	Offsets	
v_i	$IR(v_i)$	t_0	t_i
a	$\{v_0\}$	0	--
v1	$\{v_0\}$	0	--
v2	$\{v_0\}$	2	--
v3	$\{v_0, v_a\}$	2	0

Sequencing Graph



Relative Scheduling

- A constraint graph is feasible if all timing constraints are satisfied when the execution delay of the anchors are zero
- A constraint graph is well-posed if it can be satisfied for all values the execution delays of the anchors
- Well-posedness implies feasibility
- Relative schedule can be defined for well-posed graphs
- A feasible constraint graph $G_c(V_c, E_c)$ is well-posed or it can be made well-posed iff no cycle with unbounded weight exists in G_C

Relative Scheduling