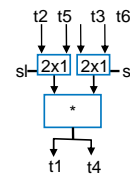
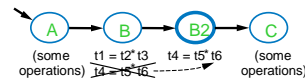
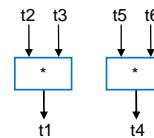
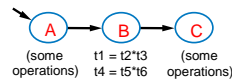


Lecture 13 Scheduling Algorithms

Scheduling

- What have we done?
 - Specified the order in which operations are performed
- What's next - Operator Scheduling
 - Assigning operations performed in each states
 - Generally speaking, determining the start time of each task/operation
- Why is scheduling important?
 - Determines the amount of concurrency of the resulting implementation – effects performance
 - Maximum amount of concurrent operations of a given type at any time step also determines the amount of hardware resources of that type required – effects area



Control/Data flow graph (CDFG)

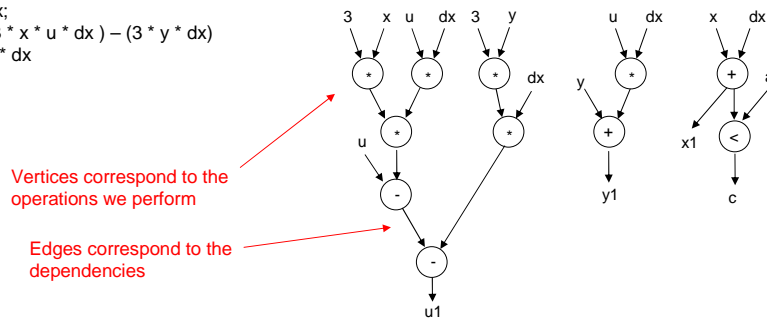
- How do we represent a scheduling?
- Dataflow graph – represents the way data flows through a computation
 - Computations limited to 2-input

Code fragment we want to implement

```

x1 = x + dx;
u1 = u - (3 * x * u * dx) - (3 * y * dx);
y1 = y + u * dx;
c = x1 < a;
    
```

Corresponding CDFG



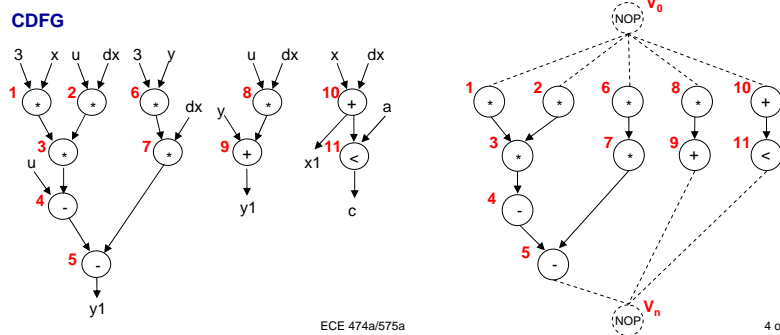
ECE 474a/575a
Susan Lysecky

3 of 72

Task Representation - Sequencing graph

- Determining a schedule
 - Don't really care about the actual input/output values
 - Just want to know the task we perform (add, subtract, compare, etc..) and the dependencies among the task
- Utilize a Sequencing graph

Sequencing Graph



ECE 474a/575a
Susan Lysecky

4 of 72

Sequencing graph

Source node

Represented by a NOP (no operation) node
Indicates start of computation

Dotted lines

Does NOT represent a dependency between tasks or operations
Represents a connection between the source and the initial task or operations

Vertices

Represents the task or operation to be performed

Edges

Represents dependencies among operations

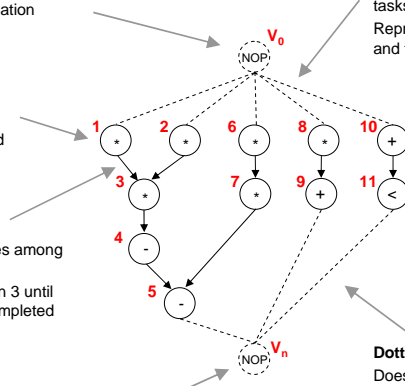
Cannot perform operation 3 until operation 1 and 2 are completed

Sink node

Represented by a NOP (no operation) node
Indicates completion of computation

Dotted lines

Does NOT represent a dependency between tasks or operations
Represents a connection between the sink and the final task or operations

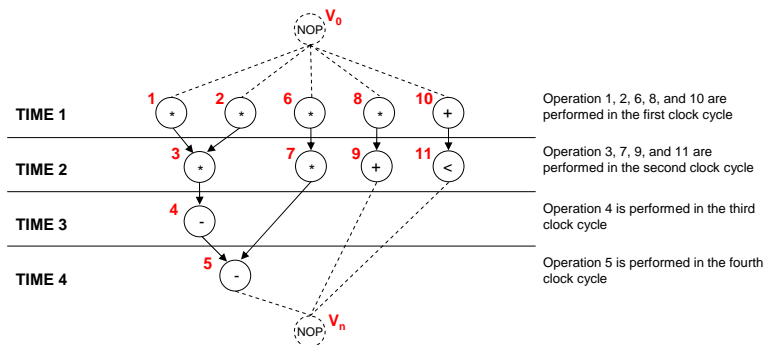


ECE 474a/575a
Susan Lysecky

5 of 72

Scheduling - Sequencing Graphs

- Sequencing graph itself only specifies the dependencies among tasks
- Scheduling requires we associate a start time for each task/operation in the sequencing graph
 - Introduce concept of time



Operation 1, 2, 6, 8, and 10 are performed in the first clock cycle

Operation 3, 7, 9, and 11 are performed in the second clock cycle

Operation 4 is performed in the third clock cycle

Operation 5 is performed in the fourth clock cycle

Notice all operations require only 1 clock cycle to execute

ECE 474a/575a
Susan Lysecky

6 of 72

ASAP Scheduling

- Unconstrained minimum-latency scheduling problem
 - We have infinite resources, all we want is the minimum time to perform the computation
 - Commonly referred to as ASAP (as soon as possible) scheduling

```

ASAP(  $G_S(V,E)$  ){
  Schedule  $v_0$  by setting  $t_0 = 1$ 
  repeat{
    Select a vertex  $v_i$  whose predecessors are all scheduled;
    Schedule  $v_i$  by setting  $t_i = \max_{j:(v_j, v_i) \in E} t_j + d_j$ 
  }
  until ( $v_n$  is scheduled);
  return  $t$ ;
}
    
```

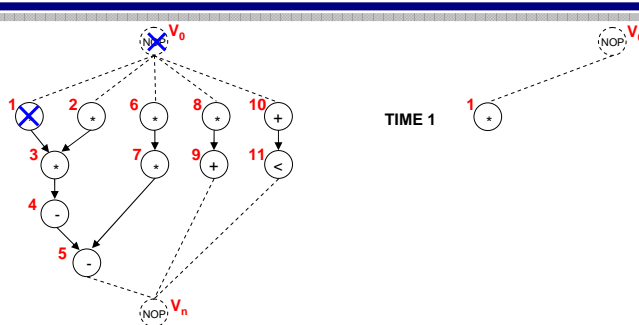
Perform ASAP scheduling on the sequencing graph
 Schedule the source node v_0 for time 1
 Look for tasks/operations that are not dependent on a task/operation that hasn't been scheduled yet
 Schedule the task/operation to time = time predecessor scheduled for + time required for predecessor to execute
 may have multiple predecessors, take maximum time
 Keep going until we have scheduled the sink node v_n

ECE 474a/575a
Susan Lysecky

7 of 72

ASAP Scheduling

Example 1



Step1
Schedule v_0 at time 1

Step2
Select a vertex v_i whose predecessors are all scheduled

All of v_i predecessors are scheduled

Step3
Schedule v_i to time = predecessor's scheduled time + time required for predecessor to execute

Time = v_0 start time + v_0 execution time
 = 1 + 0
 = 1

Step4
Has v_n been scheduled yet?

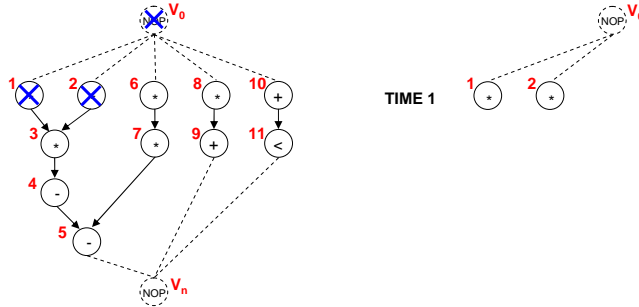
No. Repeat loop.

ECE 474a/575a
Susan Lysecky

8 of 72

ASAP Scheduling

Example 1



Step2

Select a vertex v_i whose predecessors are all scheduled

All of v_2 predecessors are scheduled

Step3

Schedule v_i to time = predecessor's scheduled time + time required for predecessor to execute

Time = v_0 start time + v_0 execution time
 = $1 + 0$
 = 1

Step4

Has v_n been scheduled yet?

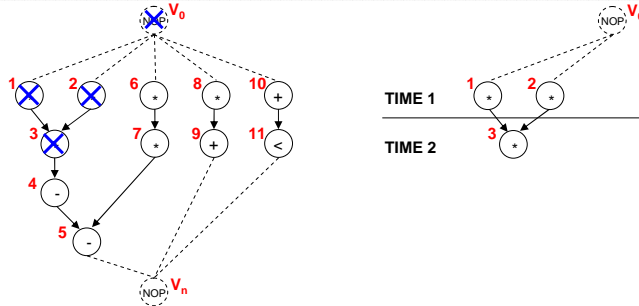
No. Repeat loop.

ECE 474a/575a
 Susan Lysecky

9 of 72

ASAP Scheduling

Example 1



Step2

Select a vertex v_i whose predecessors are all scheduled

All of v_3 predecessors are scheduled

Step3

Schedule v_i to time = predecessor's scheduled time + time required for predecessor to execute

Time = v_1 start time + v_1 execution time = $1 + 1 = 2$
 Time = v_2 start time + v_2 execution time = $1 + 1 = 2$

Step4

Has v_n been scheduled yet?

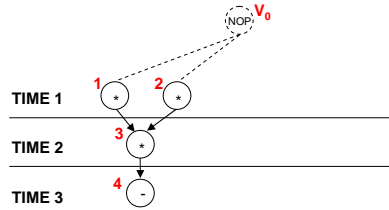
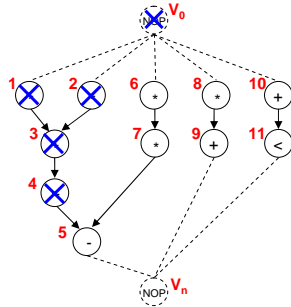
No. Repeat loop.

ECE 474a/575a
 Susan Lysecky

10 of 72

ASAP Scheduling

Example 1



Step2

Select a vertex v_i whose predecessors are all scheduled

All of v_i predecessors are scheduled

Step3

Schedule v_i to time = predecessor's scheduled time + time required for predecessor to execute

Time = v_3 start time + v_3 execution time
 = 2 + 1
 = 3

Step4

Has v_n been scheduled yet?

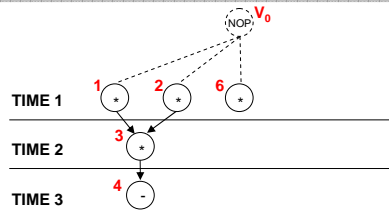
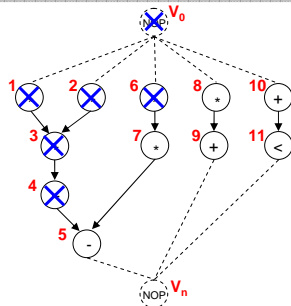
No. Repeat loop.

ECE 474a/575a
 Susan Lysecky

11 of 72

ASAP Scheduling

Example 1



Step2

Select a vertex v_i whose predecessors are all scheduled

v_3 still has predecessors not scheduled (v_2), skip for now
 All of v_6 predecessors are scheduled

Step3

Schedule v_i to time = predecessor's scheduled time + time required for predecessor to execute

Time = v_6 start time + v_6 execution time
 = 1 + 0
 = 1

Step4

Has v_n been scheduled yet?

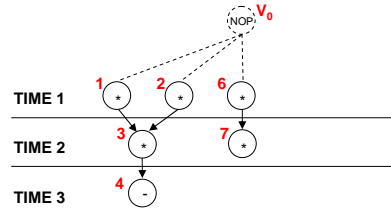
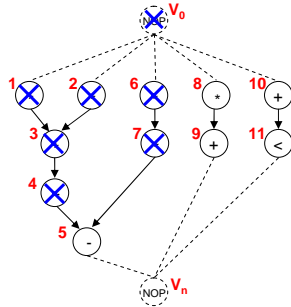
No. Repeat loop.

ECE 474a/575a
 Susan Lysecky

12 of 72

ASAP Scheduling

Example 1



Step2

Select a vertex v_i whose predecessors are all scheduled

All of v_i predecessors are scheduled

Step3

Schedule v_i to time = predecessor's scheduled time + time required for predecessor to execute

Time = v_i start time + v_i execution time
 = 1 + 1
 = 2

Step4

Has v_n been scheduled yet?

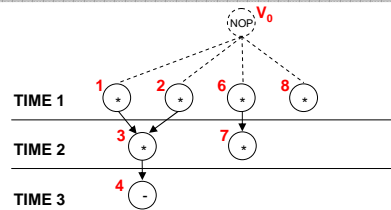
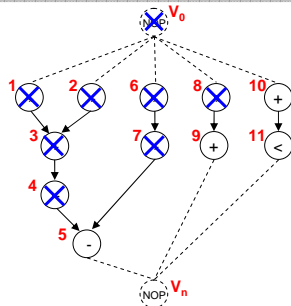
No. Repeat loop.

ECE 474a/575a
 Susan Lysecky

13 of 72

ASAP Scheduling

Example 1



Step2

Select a vertex v_i whose predecessors are all scheduled

All of v_i predecessors are scheduled

Step3

Schedule v_i to time = predecessor's scheduled time + time required for predecessor to execute

Time = v_i start time + v_i execution time
 = 1 + 0
 = 1

Step4

Has v_n been scheduled yet?

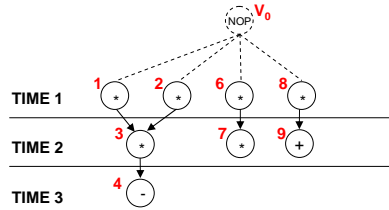
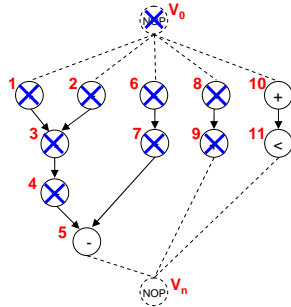
No. Repeat loop.

ECE 474a/575a
 Susan Lysecky

14 of 72

ASAP Scheduling

Example 1



Step2

Select a vertex v_i whose predecessors are all scheduled

All of v_9 predecessors are scheduled

Step3

Schedule v_i to time = predecessor's scheduled time + time required for predecessor to execute

Time = v_9 start time + v_9 execution time
 = $1 + 1$
 = 2

Step4

Has v_n been scheduled yet?

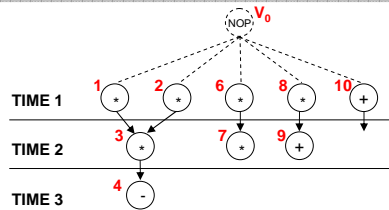
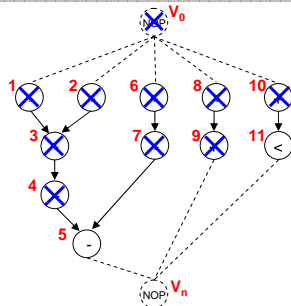
No. Repeat loop.

ECE 474a/575a
 Susan Lysecky

15 of 72

ASAP Scheduling

Example 1



Step2

Select a vertex v_i whose predecessors are all scheduled

All of v_{10} predecessors are scheduled

Step3

Schedule v_i to time = predecessor's scheduled time + time required for predecessor to execute

Time = v_{10} start time + v_{10} execution time
 = $1 + 0$
 = 1

Step4

Has v_n been scheduled yet?

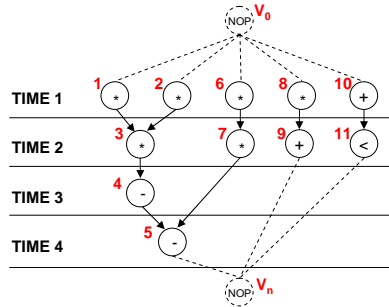
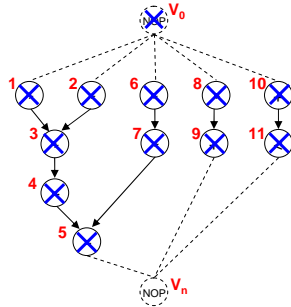
No. Repeat loop.

ECE 474a/575a
 Susan Lysecky

16 of 72

ASAP Scheduling

Example 1



Step2

Select a vertex v_i whose predecessors are all scheduled

Return to v_n , all predecessors are scheduled

Step3

Schedule v_i to time = predecessor's scheduled time + time required for predecessor to execute

$$\begin{aligned} \text{Time} &= v_5 \text{ start time} + v_5 \text{ execution time} \\ &= 4 + 1 \\ &= 5 \end{aligned}$$

$$\begin{aligned} \text{Time} &= v_9 \text{ start time} + v_9 \text{ execution time} \\ &= 2 + 1 \\ &= 3 \end{aligned}$$

Step4

Has v_n been scheduled yet?

Yes. We are done!

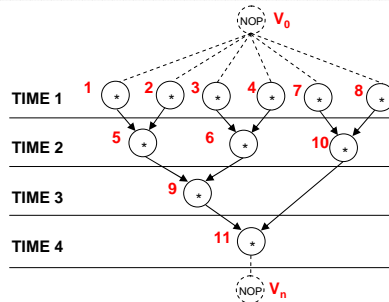
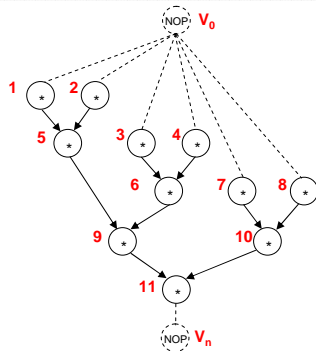
$$\begin{aligned} \text{Time} &= v_{11} \text{ start time} + v_{11} \text{ execution time} \\ &= 2 + 1 \\ &= 3 \end{aligned}$$

ECE 474a/575a
Susan Lysecky

19 of 72

ASAP Scheduling

Example 2



ASAP Scheduling goal is to schedule tasks/operations to perform as soon as possible

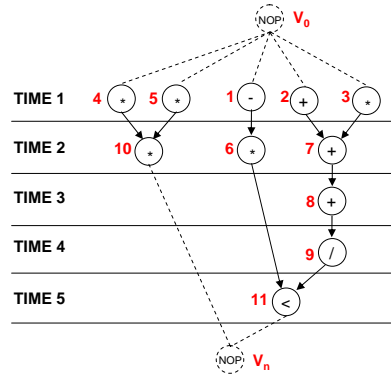
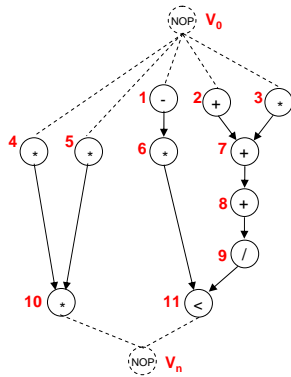
We can skip the algorithm and visually move vertices "up" as far as possible

ECE 474a/575a
Susan Lysecky

20 of 72

ASAP Scheduling

Example 3



ALAP Scheduling

- Latency constrained scheduling problem
 - Schedule must satisfy an upper bound on latency
 - Commonly referred to as ALAP (as late as possible) scheduling

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

 Schedule v_n by setting $t_n = \lambda + 1$

 repeat {

 Select a vertex v_i whose successors are all scheduled;

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

 } until (v_0 is scheduled);

 return t ;

}

Perform ALAP scheduling on the sequencing graph, λ is the upper time bound

Schedule the sink node v_n for upper latency bound + 1

Look for tasks/operations whose successors are already scheduled

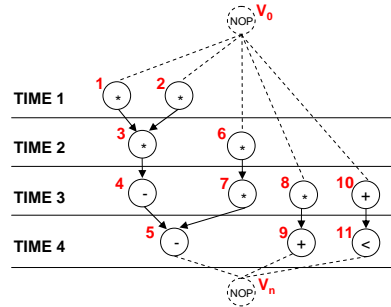
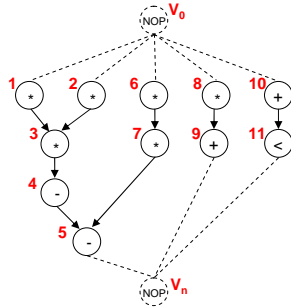
Schedule the task/operation to time = time successor scheduled for - time required for successor to execute

may have multiple successors, take minimum time

Keep going until we have scheduled the source node v_0

ALAP Scheduling

Example 1

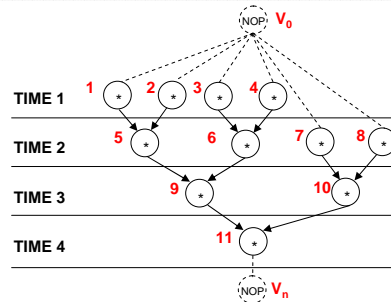
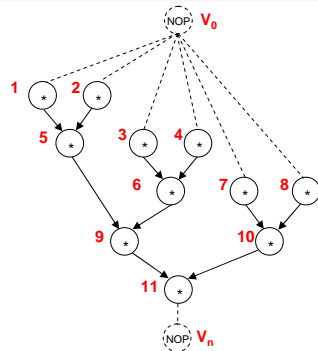


ALAP Scheduling goal is to schedule tasks/operations to perform as late as possible

We can skip the algorithm and visually move vertices "down" as far as possible

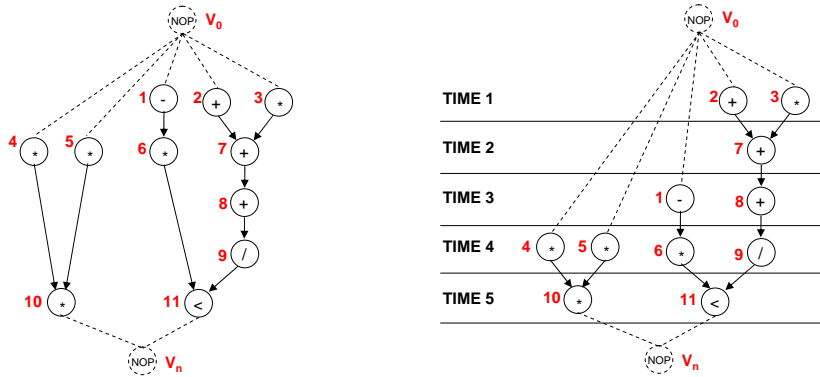
ALAP Scheduling

Example 2



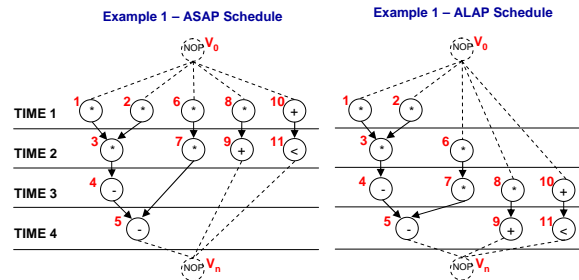
ALAP Scheduling

Example 3



Mobility

- Mobility (or slack) important quantity used by some scheduling algorithms
 - Mobility = start time_{ALAP scheduling} - start time_{ASAP scheduling}
 - Mobility = 0, task/operation can only be started at the given time in order to meet overall latency constraint
 - Mobility > 0, indicates span of possible start times
 - Helps with minimizing resources (adders, multipliers, etc.)



$$V_1 \text{ mobility} = \text{time}_{\text{ALAP}}(V_1) + \text{time}_{\text{ASAP}}(V_1) \\ = 1 - 1 \\ = 0$$

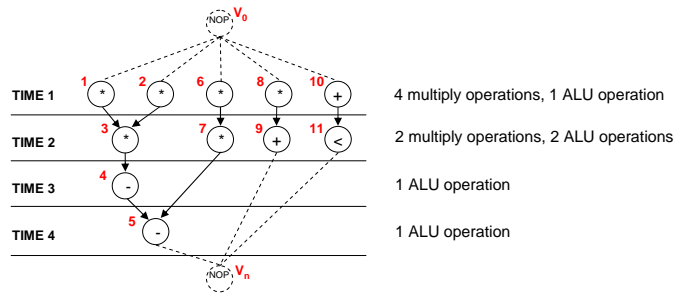
$$V_6 \text{ mobility} = \text{time}_{\text{ALAP}}(V_6) + \text{time}_{\text{ASAP}}(V_6) \\ = 2 - 1 \\ = 1$$

$$V_{11} \text{ mobility} = \text{time}_{\text{ALAP}}(V_{11}) + \text{time}_{\text{ASAP}}(V_{11}) \\ = 4 - 2 \\ = 2$$

Mobility

Example 1 – ASAP Only

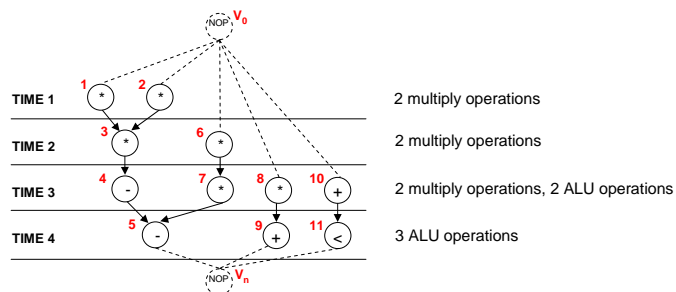
- What do we get with the ASAP Schedule?
 - Latency = 4
 - Resource requirement = 4 multipliers, 2 ALUs



Mobility

Example 1 – ALAP Only

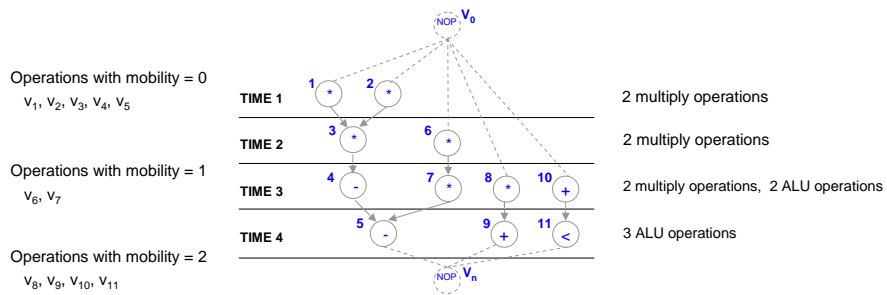
- What do we get with the ALAP Schedule?
 - Latency = 4
 - Resource requirement = 2 multipliers, 3 ALUs



Mobility

Example 1 – Modify ALAP

- Start with ALAP schedule
- Use mobility to try to improve resource requirements



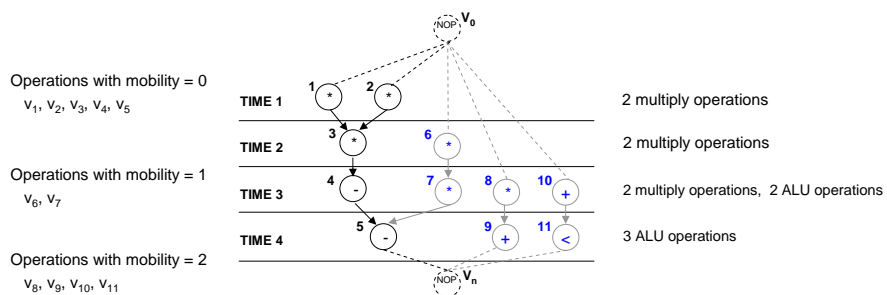
ECE 474a/575a
 Susan Lysecky

29 of 72

Mobility

Example 1 – Modify ALAP

- Start with ALAP schedule
- Use mobility to try to improve resource requirements
 - Vertices with mobility = 0 cannot be moved, they are part of the critical path



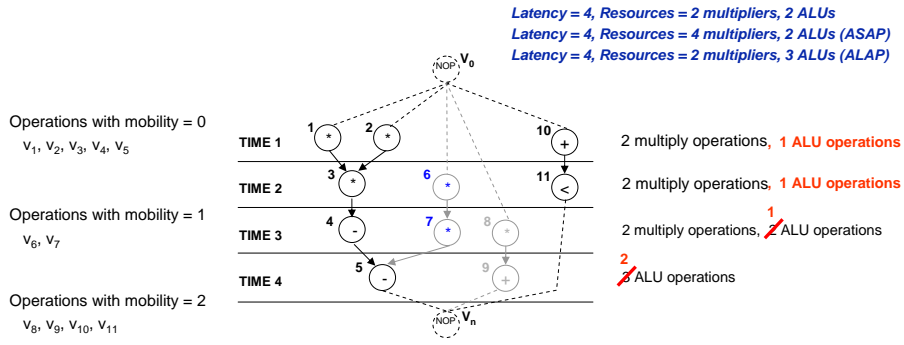
ECE 474a/575a
 Susan Lysecky

30 of 72

Mobility

Example 1 – Modify ALAP

- Start with ALAP schedule
- Use mobility to try to improve resource requirements
 - Vertices with mobility = 0 cannot be moved, they are part of the critical path
 - Vertices with mobility > 0 can be moved to minimize resource requirements

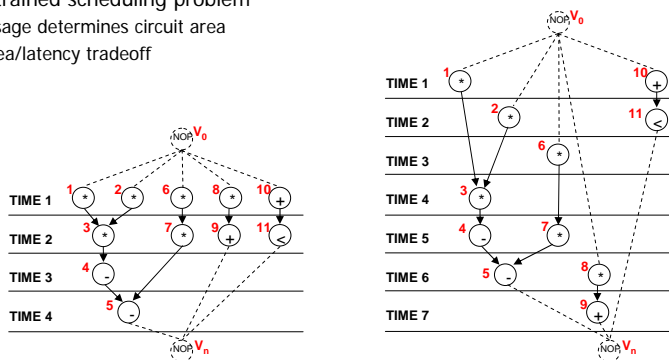


ECE 474a/575a
Susan Lysecky

31 of 72

Resource Constrained Scheduling

- Resource constrained scheduling problem
 - Resource usage determines circuit area
 - Consider area/latency tradeoff



Likely we want something in between

ECE 474a/575a
Susan Lysecky

32 of 72

Hu's Algorithm

- Exact (polynomial-time) algorithm for resource constrained scheduling
 - Assumes one resource handles all possible operations
 - Assumes all operations have 1 unit delay

```

HU(  $G_s(V,E), a$  ){
  Label the vertices;
   $l = 1$ ;
  repeat {
     $U =$  unscheduled vertices in  $V$  without predecessors
    or whose predecessors have been scheduled;
    Select  $S \subseteq U$  vertices, such that  $|S| \leq a$  and labels in  $S$  are maximal;
    Schedule the  $S$  operations at step  $l$  by setting  $t = l \forall i : v_i \in S$ ;
     $l = l + 1$ ;
  } until ( $v_n$  is scheduled);
  return  $t$ ;
}
    
```

Value of a indicates the number of resources we have available

Label with distance of vertices to sink node

Indicates the time step

Make a list of all vertices not waiting on another operation to be scheduled

Select a subset of the vertices in U , no more than a , choosing vertices with largest labels

Schedule the vertices in the subset to start at time l

update l to next time step

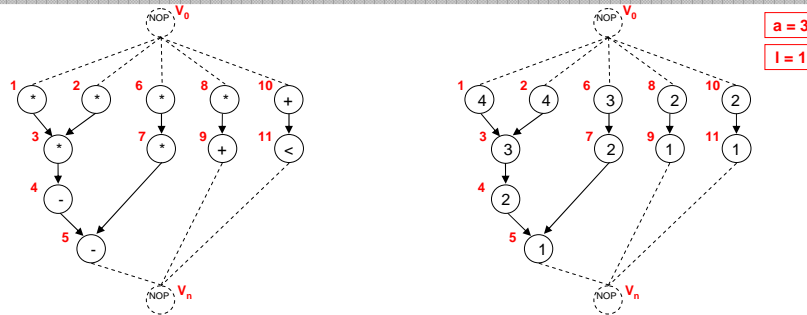
Keep going until we have scheduled the sink node v_n

ECE 474a/575a
Susan Lysecky

33 of 72

Hu's Algorithm

Example 1



Step1
Label all vertices with distance to sink

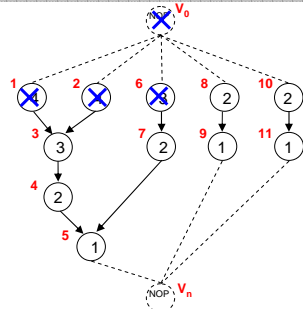
Step2
 $l = 1$

ECE 474a/575a
Susan Lysecky

34 of 72

Hu's Algorithm

Example 1



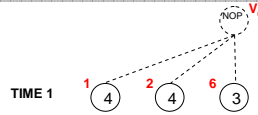
Step 3
 U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4
 S = subset set of vertices in U, no more than a, where labels are maximal

Step 5
 Schedule vertices in S to time step I

Step 6
 I = I + 1

Step 7
 Has v_n been scheduled yet?



a = 3

I = 1

U = { $v_1, v_2, v_6, v_8, v_{10}$ }

S = { v_1, v_2, v_6 }

Set vertices in S to start at 1

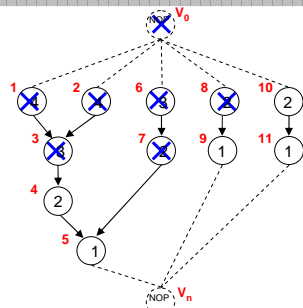
I = 1 + 1 = 2

No. Repeat loop.
 ECE 474a/575a
 Susan Lysecky

35 of 72

Hu's Algorithm

Example 1



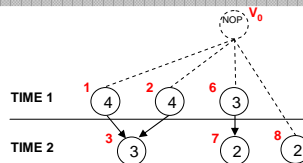
Step 3
 U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4
 S = subset set of vertices in U, no more than a, where labels are maximal

Step 5
 Schedule vertices in S to time step I

Step 6
 I = I + 1

Step 7
 Has v_n been scheduled yet?



a = 3

I = 2

U = { v_3, v_7, v_8, v_{10} }

S = { v_3, v_7, v_8 }

Set vertices in S to start at 2

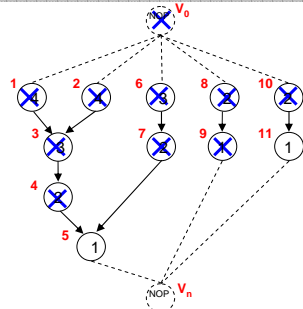
I = 2 + 1 = 3

No. Repeat loop.
 ECE 474a/575a
 Susan Lysecky

36 of 72

Hu's Algorithm

Example 1



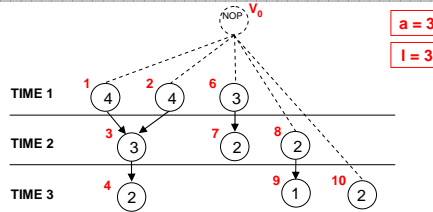
Step 3
 U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4
 S = subset set of vertices in U, no more than a, where labels are maximal

Step 5
 Schedule vertices in S to time step I

Step 6
 I = I + 1

Step 7
 Has v_n been scheduled yet?



a = 3

I = 3

U = { v_4, v_9, v_{10} }

S = { v_4, v_9, v_{10} }

Set vertices in S to start at 3

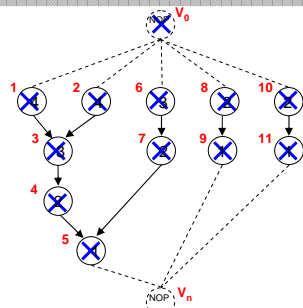
I = 3 + 1 = 4

No. Repeat loop.
 ECE 474a/575a
 Susan Lysecky

37 of 72

Hu's Algorithm

Example 1



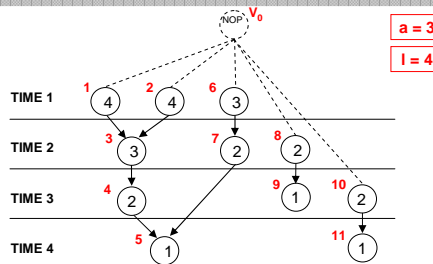
Step 3
 U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4
 S = subset set of vertices in U, no more than a, where labels are maximal

Step 5
 Schedule vertices in S to time step I

Step 6
 I = I + 1

Step 7
 Has v_n been scheduled yet?



a = 3

I = 4

U = { v_5, v_{11} }

S = { v_5, v_{11} }

Set vertices in S to start at 4

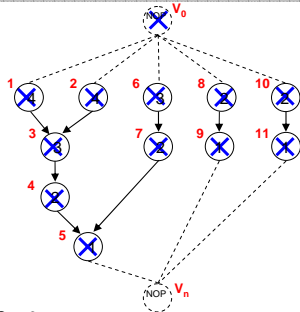
I = 4 + 1 = 5

No. Repeat loop.
 ECE 474a/575a
 Susan Lysecky

38 of 72

Hu's Algorithm

Example 1



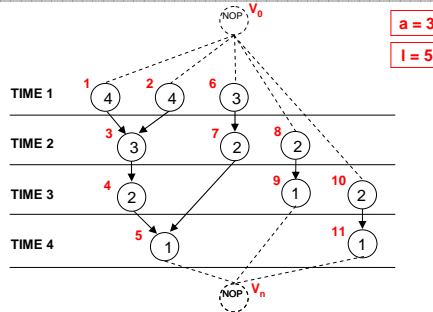
Step 3
 U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4
 S = subset set of vertices in U, no more than a, where labels are maximal

Step 5
 Schedule vertices in S to time step I

Step 6
 I = I + 1

Step 7
 Has v_n been scheduled yet?



a = 3

I = 5

U = { v_n }

S = { v_n }

Set vertices in S to start at 5

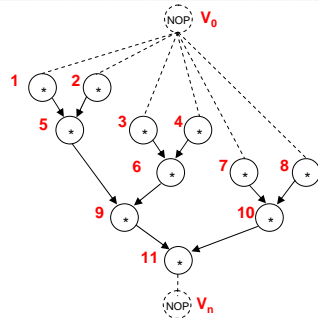
I = 5 + 1 = 6

Yes. We are done!
 ECE 474a/575a
 Susan Lysecky

39 of 72

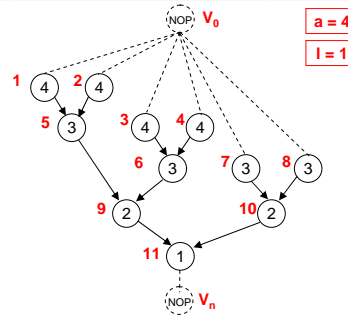
Hu's Algorithm

Example 2



Step 1
 Label all vertices with distance to sink

Step 2
 I = 1



a = 4

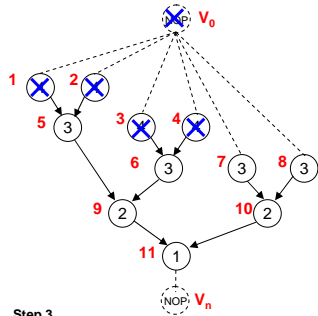
I = 1

ECE 474a/575a
 Susan Lysecky

40 of 72

Hu's Algorithm

Example 2



Step 3

U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4

S = subset set of vertices in U, no more than a, where labels are maximal

Step 5

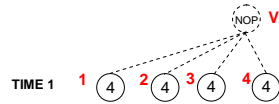
Schedule vertices in S to time step I

Step 6

I = I + 1

Step 7

Has v_n been scheduled yet?



a = 4

I = 1

U = { $v_1, v_2, v_3, v_4, v_7, v_8$ }

S = { v_1, v_2, v_3, v_4 }

Set vertices in S to start at 1

I = 1 + 1 = 2

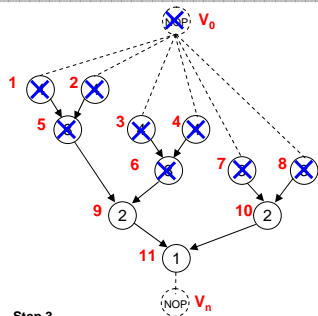
No. Repeat loop.

ECE 474a/575a
Susan Lysecky

41 of 72

Hu's Algorithm

Example 2



Step 3

U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4

S = subset set of vertices in U, no more than a, where labels are maximal

Step 5

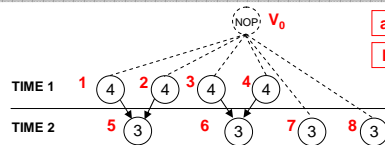
Schedule vertices in S to time step I

Step 6

I = I + 1

Step 7

Has v_n been scheduled yet?



a = 4

I = 2

U = { v_5, v_6, v_7, v_8 }

S = { v_5, v_6, v_7, v_8 }

Set vertices in S to start at 2

I = 2 + 1 = 3

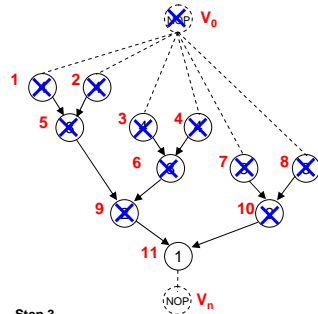
No. Repeat loop.

ECE 474a/575a
Susan Lysecky

42 of 72

Hu's Algorithm

Example 2



Step 3
 U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4
 S = subset set of vertices in U, no more than a, where labels are maximal

Step 5
 Schedule vertices in S to time step I

Step 6
 I = I + 1

Step 7
 Has v_n been scheduled yet?

U = { v_9, v_{10} }

S = { v_9, v_{10} }

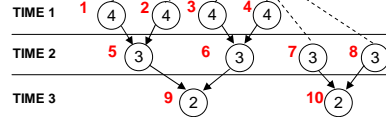
Set vertices in S to start at 3

I = 3 + 1 = 4

No. Repeat loop.
 ECE 474a/575a
 Susan Lysecky

a = 4

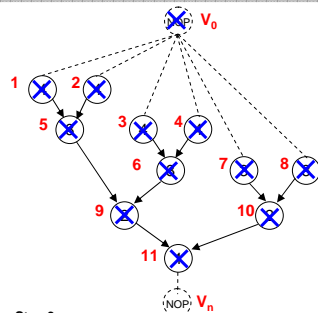
I = 3



43 of 72

Hu's Algorithm

Example 2



Step 3
 U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4
 S = subset set of vertices in U, no more than a, where labels are maximal

Step 5
 Schedule vertices in S to time step I

Step 6
 I = I + 1

Step 7
 Has v_n been scheduled yet?

U = { v_{11} }

S = { v_{11} }

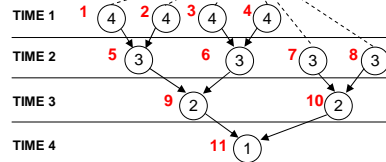
Set vertices in S to start at 4

I = 4 + 1 = 5

No. Repeat loop.
 ECE 474a/575a
 Susan Lysecky

a = 4

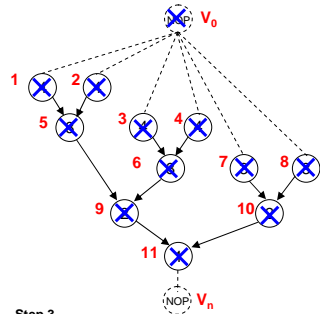
I = 4



44 of 72

Hu's Algorithm

Example 2



Step 3

U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4

S = subset set of vertices in U, no more than a, where labels are maximal

Step 5

Schedule vertices in S to time step I

Step 6

I = I + 1

Step 7

Has v_n been scheduled yet?

U = { v_n }

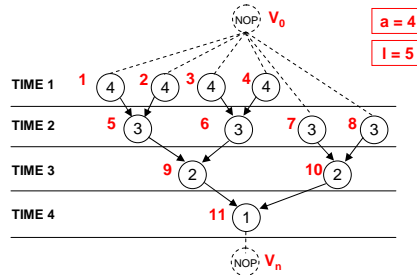
S = { v_n }

Set vertices in S to start at 5

I = 5 + 1 = 6

Yes. We are done.

ECE 474a/575a
Susan Lysecky



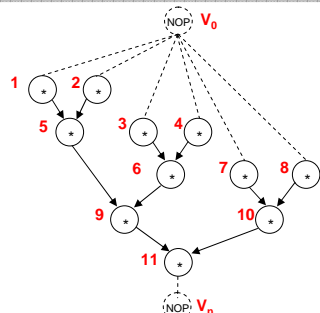
a = 4

I = 5

45 of 72

Hu's Algorithm

Example 3

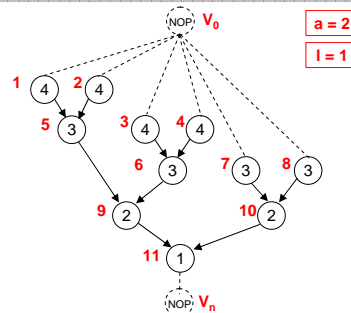


Step1

Label all vertices with distance to sink

Step2

I = 1



a = 2

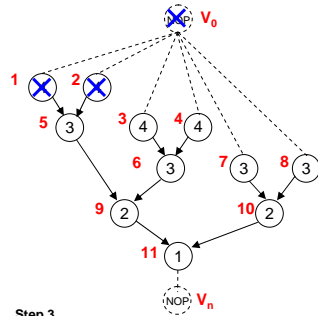
I = 1

ECE 474a/575a
Susan Lysecky

46 of 72

Hu's Algorithm

Example 3



Step 3

U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4

S = subset set of vertices in U, no more than a, where labels are maximal

Step 5

Schedule vertices in S to time step I

Step 6

I = I + 1

Step 7

Has v_n been scheduled yet?

U = { $v_1, v_2, v_3, v_4, v_7, v_8$ }

S = { v_1, v_2 }

Set vertices in S to start at 1

I = 1 + 1 = 2

No. Repeat loop.
ECE 474a/575a
Susan Lysecky

a = 2

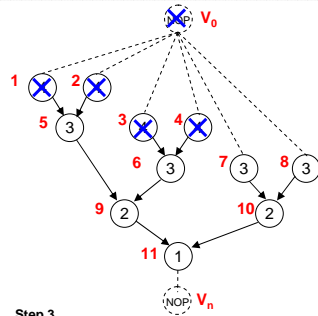
I = 1



47 of 72

Hu's Algorithm

Example 3



Step 3

U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4

S = subset set of vertices in U, no more than a, where labels are maximal

Step 5

Schedule vertices in S to time step I

Step 6

I = I + 1

Step 7

Has v_n been scheduled yet?

U = { v_3, v_4, v_5, v_7, v_8 }

S = { v_3, v_4 }

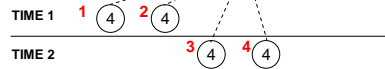
Set vertices in S to start at 2

I = 2 + 1 = 3

No. Repeat loop.
ECE 474a/575a
Susan Lysecky

a = 2

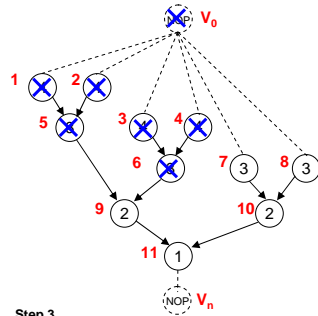
I = 2



48 of 72

Hu's Algorithm

Example 3



Step 3

U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4

S = subset set of vertices in U, no more than a, where labels are maximal

Step 5

Schedule vertices in S to time step I

Step 6

I = I + 1

Step 7

Has v_n been scheduled yet?

U = { v_5, v_6, v_7, v_8 }

S = { v_5, v_6 }

Set vertices in S to start at 3

I = 3 + 1 = 4

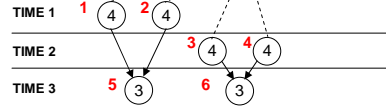
No. Repeat loop.

ECE 474a/575a

Susan Lysecky

a = 2

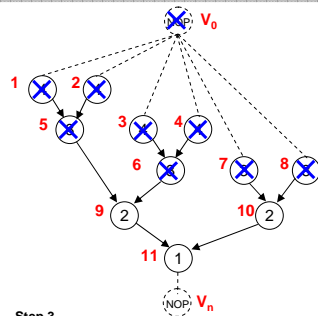
I = 3



49 of 72

Hu's Algorithm

Example 3



Step 3

U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4

S = subset set of vertices in U, no more than a, where labels are maximal

Step 5

Schedule vertices in S to time step I

Step 6

I = I + 1

Step 7

Has v_n been scheduled yet?

U = { v_7, v_8, v_9 }

S = { v_7, v_8 }

Set vertices in S to start at 4

I = 4 + 1 = 5

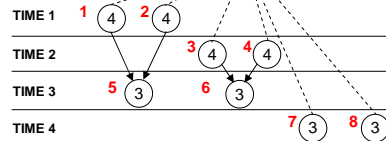
No. Repeat loop.

ECE 474a/575a

Susan Lysecky

a = 2

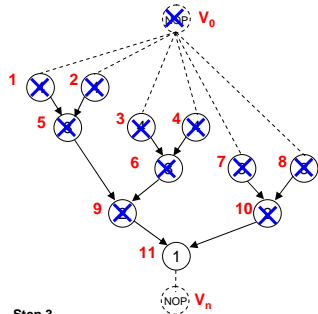
I = 4



50 of 72

Hu's Algorithm

Example 3



Step 3

U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4

S = subset set of vertices in U, no more than a, where labels are maximal

Step 5

Schedule vertices in S to time step I

Step 6

I = I + 1

Step 7

Has v_n been scheduled yet?

U = { v_9, v_{10} }

S = { v_9, v_{10} }

Set vertices in S to start at 5

I = 5 + 1 = 6

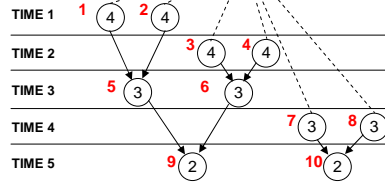
No. Repeat loop.

ECE 474a/575a

Susan Lysecky

a = 2

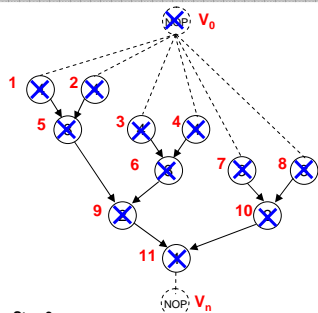
I = 5



51 of 72

Hu's Algorithm

Example 3



Step 3

U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4

S = subset set of vertices in U, no more than a, where labels are maximal

Step 5

Schedule vertices in S to time step I

Step 6

I = I + 1

Step 7

Has v_n been scheduled yet?

U = { v_{11} }

S = { v_{11} }

Set vertices in S to start at 6

I = 6 + 1 = 7

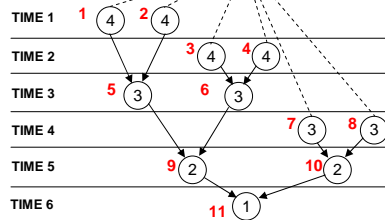
No. Repeat loop.

ECE 474a/575a

Susan Lysecky

a = 2

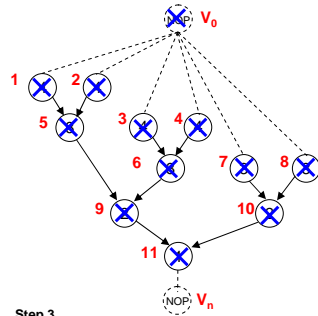
I = 6



52 of 72

Hu's Algorithm

Example 3



Step 3
 U = unscheduled vertices in V without predecessors or whose predecessors have been scheduled

Step 4
 S = subset set of vertices in U , no more than a , where labels are maximal

Step 5
 Schedule vertices in S to time step I

Step 6
 $I = I + 1$

Step 7
 Has v_n been scheduled yet?

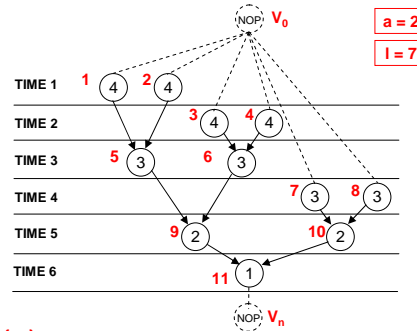
$U = \{v_n\}$

$S = \{v_n\}$

Set vertices in S to start at 7

$I = 7 + 1 = 8$

Yes. We are done.
 ECE 474a/575a
 Susan Lysecky



53 of 72

Additional Scheduling Considerations

- Hu's algorithm
 - Assumes one resource handles all possible operations
 - Assumes all operations have 1 unit delay
- Most scheduling problems have additional considerations
 - What happens when we have more than one type of task/operation?
 - What happens when a task/operation takes more than 1 unit delay?
- Increased problem space, difficult problem to solve efficiently
 - Many heuristics have been developed to address these problems
 - Minimum-latency, resource-constrained scheduling
 - Minimum-resource, latency-constrained scheduling

We consider one such heuristic from a family of heuristics called *list scheduling* that looks at the minimum-latency, resource-constrained scheduling problem

ECE 474a/575a
 Susan Lysecky

54 of 72

List Scheduling (LIST_L)

- Extension of Hu's algorithm to handle multiple operation types and multiple-cycle execution delays
- Considers minimum-latency, resource-constrained scheduling problem

```

LIST_L(  $G_S(V,E)$ ,  $a$  ){
   $l = 1$ ;
  repeat {
    for each resource type  $k = 1, 2, \dots, n_{res}$  {
      Determine candidate operations  $U_{l,k}$ ;
      Determine unfinished operations  $T_{l,k}$ ;
      Select  $S_k \subseteq U_{l,k}$  vertices, such that  $|S_k| + |T_{l,k}| \leq a_k$ ;
      Schedule the  $S_k$  operations at step  $l$  by setting  $t_i = l \forall i : v_i \in S$ ;
    }
     $l = l + 1$ ;
  } until ( $v_n$  is scheduled);
  return  $t$ ;
}

```

Vector a indicates the number of each type of resource available

indicates the time step

Operations of type k whose predecessors are completed by time l

Unfinished operations that are already scheduled but have not completed yet

Select a subset S so that the number of new operations and unfinished operations are \leq to number of resources of that type

Schedule operations in S to run at time step l

update l to next time step

Keep going until we have scheduled the sink node v_n

ECE 474a/575a
Susan Lysecky

55 of 72

List Scheduling (LIST_L)

```

LIST_L(  $G_S(V,E)$ ,  $a$  ){
   $l = 1$ ;
  repeat {
    for each resource type  $k = 1, 2, \dots, n_{res}$  {
      Determine candidate operations  $U_{l,k}$ ;
      Determine unfinished operations  $T_{l,k}$ ;
      Select  $S_k \subseteq U_{l,k}$  vertices, such that  $|S_k| + |T_{l,k}| \leq a_k$ ;
      Schedule the  $S_k$  operations at step  $l$  by setting  $t_i = l \forall i : v_i \in S$ ;
    }
     $l = l + 1$ ;
  } until ( $v_n$  is scheduled);
  return  $t$ ;
}

```

Select a subset S so that the number of new operations and unfinished operations are \leq to number of resources of that type

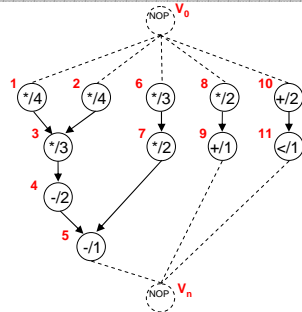
- Selection of which operations to include is based on a priority list indicating some sort of urgency measure
 - We will utilize same method of labeling vertices with weights indicating path to sink, choose operations with highest weights

ECE 474a/575a
Susan Lysecky

56 of 72

LIST_L Scheduling

Example 1



Assume all operations take 1 cycle

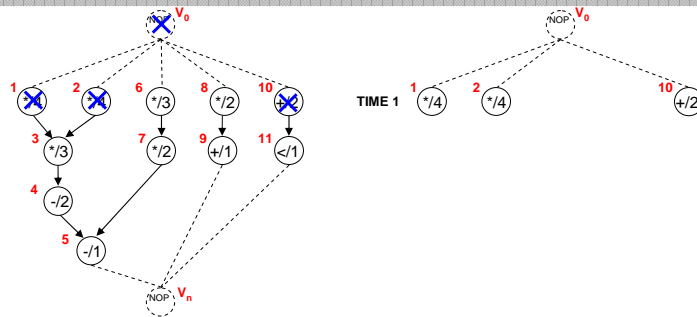
$a_1 = 2$ multipliers
 $a_2 = 2$ ALUs

$l = 1$

Step 1
 $l = 1$

LIST_L Scheduling

Example 1



Assume all operations take 1 cycle

$a_1 = 2$ multipliers
 $a_2 = 2$ ALUs

$l = 1$

Step 2/3

$U_{i,k}$ = candidate operations with predecessors finished at i
 $T_{i,k}$ = unfinished operations

Step 4

S = subset set of vertices in U and T such that $U + T \leq a$, where labels are maximal

Step 5

Schedule vertices in S to time step l

Step 6

$l = l + 1$

Step 7

Has v_n been scheduled yet?

Multipliers

$U = \{v_1, v_2, v_6, v_8\}$
 $T = \{\}$

$S = \{v_1, v_2\}$

Set vertices in S to start at 1

ALUs

$U = \{v_{10}\}$
 $T = \{\}$

$S = \{v_{10}\}$

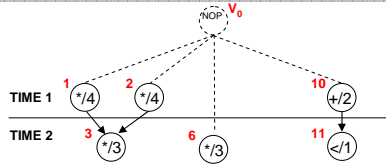
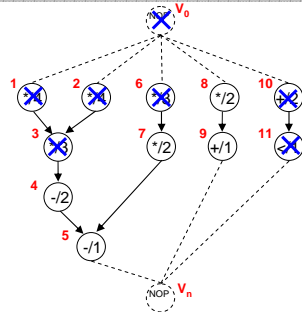
Set vertices in S to start at 1

$l = 1 + 1 = 2$

No. Repeat loop.

LIST_L Scheduling

Example 1



Assume all operations take 1 cycle

$a_1 = 2$ multipliers
 $a_2 = 2$ ALUs

I = 2

Step 2/3

$U_{i,k}$ = candidate operations with predecessors finished at i
 $T_{i,k}$ = unfinished operations

Step 4

S = subset set of vertices in U and T such that $U + T$ is $\leq a$, where labels are maximal

Step 5

Schedule vertices in S to time step i

Step 6

$i = i + 1$

Step 7

Has v_n been scheduled yet?

Multipliers

$U = \{v_3, v_6, v_8\}$
 $T = \{\}$

$S = \{v_3, v_6\}$

Set vertices in S to start at 2

ALUs

$U = \{v_{11}\}$
 $T = \{\}$

$S = \{v_{11}\}$

Set vertices in S to start at 2

$i = 2 + 1 = 3$

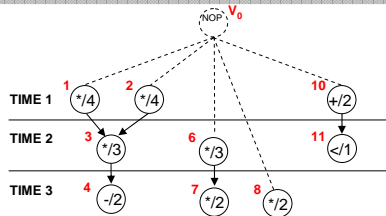
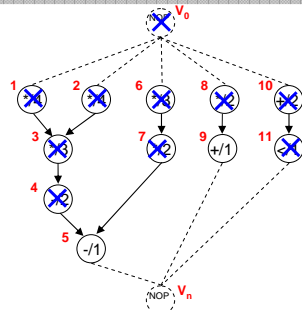
No. Repeat loop.

ECE 474a/575a
Susan Lysecky

59 of 72

LIST_L Scheduling

Example 1



Assume all operations take 1 cycle

$a_1 = 2$ multipliers
 $a_2 = 2$ ALUs

I = 3

Step 2/3

$U_{i,k}$ = candidate operations with predecessors finished at i
 $T_{i,k}$ = unfinished operations

Step 4

S = subset set of vertices in U and T such that $U + T$ is $\leq a$, where labels are maximal

Step 5

Schedule vertices in S to time step i

Step 6

$i = i + 1$

Step 7

Has v_n been scheduled yet?

Multipliers

$U = \{v_7, v_8\}$
 $T = \{\}$

$S = \{v_7, v_8\}$

Set vertices in S to start at 3

ALUs

$U = \{v_4\}$
 $T = \{\}$

$S = \{v_4\}$

Set vertices in S to start at 3

$i = 3 + 1 = 4$

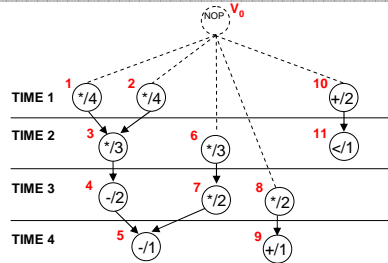
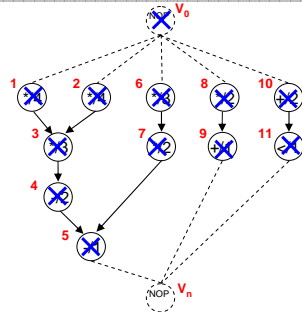
No. Repeat loop.

ECE 474a/575a
Susan Lysecky

60 of 72

LIST_L Scheduling

Example 1



Assume all operations take 1 cycle

$a_1 = 2$ multipliers
 $a_2 = 2$ ALUs

I = 4

Step 2/3

$U_{i,k}$ = candidate operations with predecessors finished at i
 $T_{i,k}$ = unfinished operations

Step 4

S = subset set of vertices in U and T such that $U + T$ is $\leq a$, where labels are maximal

Step 5

Schedule vertices in S to time step i

Step 6

$i = i + 1$

Step 7

Has v_n been scheduled yet?

Multipliers

$U = \{ \}$
 $T = \{ \}$

$S = \{ \}$

ALUs

$U = \{ v_9, v_8 \}$
 $T = \{ \}$

$S = \{ v_9, v_8 \}$

Set vertices in S to start at 4

$i = 4 + 1 = 5$

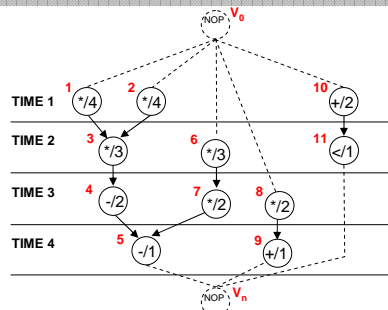
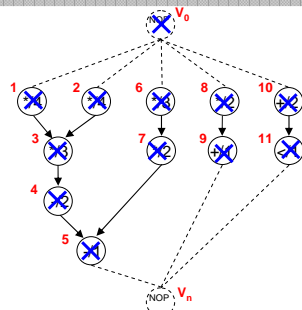
No. Repeat loop.

ECE 474a/575a
Susan Lysecky

61 of 72

LIST_L Scheduling

Example 1



Assume all operations take 1 cycle

$a_1 = 2$ multipliers
 $a_2 = 2$ ALUs

I = 5

Step 2/3

$U_{i,k}$ = candidate operations with predecessors finished at i
 $T_{i,k}$ = unfinished operations

Step 4

S = subset set of vertices in U and T such that $U + T$ is $\leq a$, where labels are maximal

Step 5

Schedule vertices in S to time step i

Step 6

$i = i + 1$

Step 7

Has v_n been scheduled yet?

Multipliers

$U = \{ \}$
 $T = \{ \}$

$S = \{ \}$

ALUs

$U = \{ \}$
 $T = \{ \}$

$S = \{ \}$

$U = \{ v_n \}$
 $T = \{ \}$

$S = \{ v_n \}$

Set vertices in S to start at 5

$i = 5 + 1 = 6$

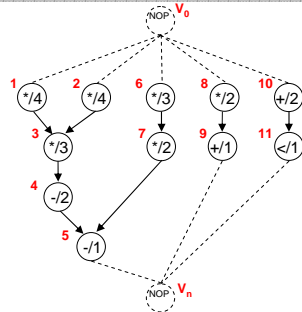
Yes. We are done.

ECE 474a/575a
Susan Lysecky

62 of 72

LIST_L Scheduling

Example 2



Mult. = 2 cycles
ALU = 1 cycle

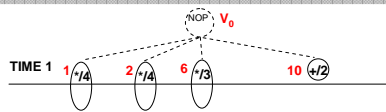
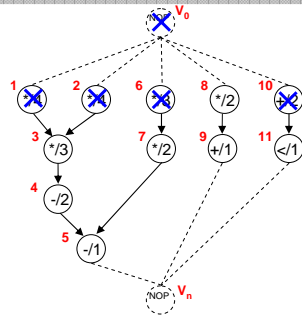
A₁ = 3 multipliers
A₂ = 1 ALU

I = 1

Step 1
I = 1

LIST_L Scheduling

Example 2



Mult. = 2 cycles
ALU = 1 cycle

A₁ = 3 multipliers
A₂ = 1 ALU

I = 1

Step 2/3

U_{i,k} = candidate operations with predecessors finished at I
T_{i,k} = unfinished operations

Step 4

S = subset set of vertices in U and T such that U + T ≤ a, where labels are maximal

Step 5

Schedule vertices in S to time step I

Step 6

I = I + 1

Step 7

Has v_n been scheduled yet?

Multipliers

U = { v₁, v₂, v₆, v₈ }
T = { }

S = { v₁, v₂, v₆ }

Set vertices in S to start at 1

ALUs

U = { v₁₀ }
T = { }

S = { v₁₀ }

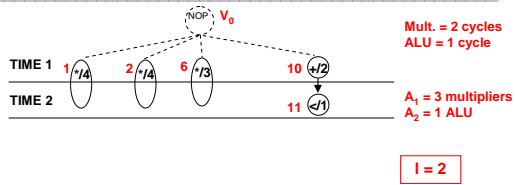
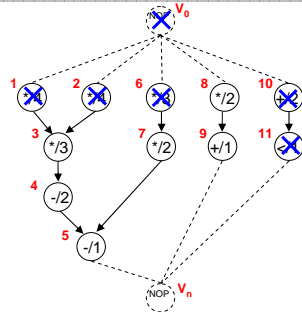
Set vertices in S to start at 1

I = 1 + 1 = 2

No. Repeat loop.

LIST_L Scheduling

Example 2



Step 2/3

$U_{i,k}$ = candidate operations with predecessors finished at i
 $T_{i,k}$ = unfinished operations

Step 4

S = subset set of vertices in U and T such that $U + T$ is $\leq a$, where labels are maximal

Step 5

Schedule vertices in S to time step i

Step 6

$i = i + 1$

Step 7

Has v_n been scheduled yet?

Multipliers

$U = \{ v_8 \}$
 $T = \{ v_{11}, v_2, v_6 \}$

$S = \{ v_{11}, v_2, v_6 \}$

Set vertices in S to start at 2

ALUs

$U = \{ v_{11} \}$
 $T = \{ \}$

$S = \{ v_{11} \}$

Set vertices in S to start at 2

$i = 2 + 1 = 3$

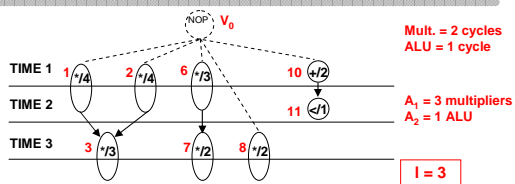
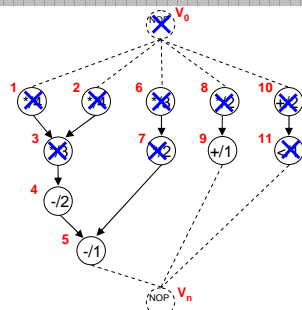
No. Repeat loop.

ECE 474a/575a
 Susan Lysecky

65 of 72

LIST_L Scheduling

Example 2



Step 2/3

$U_{i,k}$ = candidate operations with predecessors finished at i
 $T_{i,k}$ = unfinished operations

Step 4

S = subset set of vertices in U and T such that $U + T$ is $\leq a$, where labels are maximal

Step 5

Schedule vertices in S to time step i

Step 6

$i = i + 1$

Step 7

Has v_n been scheduled yet?

Multipliers

$U = \{ v_3, v_7, v_8 \}$
 $T = \{ \}$

$S = \{ v_3, v_7, v_8 \}$

Set vertices in S to start at 3

ALUs

$U = \{ \}$
 $T = \{ \}$

$S = \{ \}$

$i = 3 + 1 = 4$

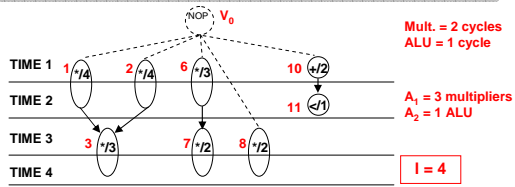
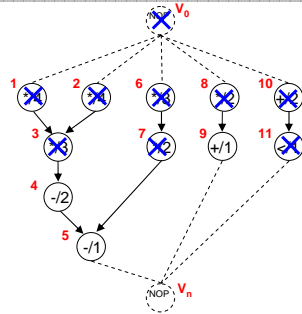
No. Repeat loop.

ECE 474a/575a
 Susan Lysecky

66 of 72

LIST_L Scheduling

Example 2



Step 2/3

$U_{i,k}$ = candidate operations with predecessors finished at i
 $T_{i,k}$ = unfinished operations

Step 4

S = subset set of vertices in U and T such that $U + T$ is $\leq a$, where labels are maximal

Step 5

Schedule vertices in S to time step i

Step 6

$i = i + 1$

Step 7

Has v_n been scheduled yet?

Multipliers

$U = \{ \}$
 $T = \{ v_3, v_7, v_8 \}$

$S = \{ v_3, v_7, v_8 \}$

Set vertices in S to start at 4

ALUs

$U = \{ \}$
 $T = \{ \}$

$S = \{ \}$

$i = 4 + 1 = 5$

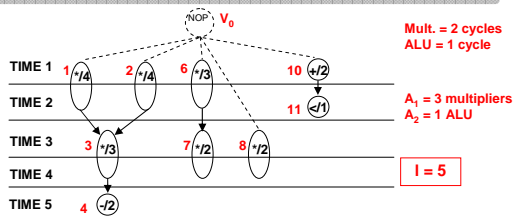
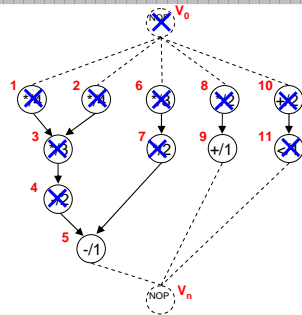
No. Repeat loop.

ECE 474a/575a
 Susan Lysecky

67 of 72

LIST_L Scheduling

Example 2



Step 2/3

$U_{i,k}$ = candidate operations with predecessors finished at i
 $T_{i,k}$ = unfinished operations

Step 4

S = subset set of vertices in U and T such that $U + T$ is $\leq a$, where labels are maximal

Step 5

Schedule vertices in S to time step i

Step 6

$i = i + 1$

Step 7

Has v_n been scheduled yet?

Multipliers

$U = \{ \}$
 $T = \{ \}$

$S = \{ \}$

ALUs

$U = \{ v_4, v_9 \}$
 $T = \{ \}$

$S = \{ v_4 \}$

Set vertices in S to start at 5

$i = 5 + 1 = 6$

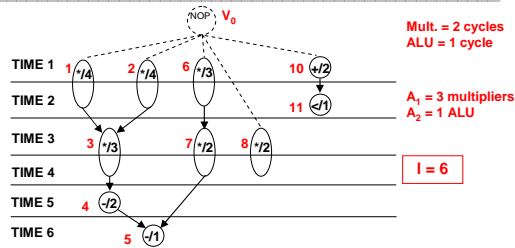
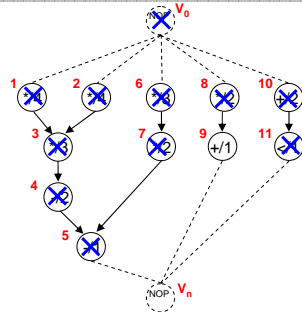
No. Repeat loop.

ECE 474a/575a
 Susan Lysecky

68 of 72

LIST_L Scheduling

Example 2



Mult. = 2 cycles
ALU = 1 cycle

A₁ = 3 multipliers
A₂ = 1 ALU

I = 6

Step 2/3

U_{i,k} = candidate operations with predecessors finished at I
T_{i,k} = unfinished operations

Step 4

S = subset set of vertices in U and T such that U + T is <= a, where labels are maximal

Step 5

Schedule vertices in S to time step I

Step 6

I = I + 1

Step 7

Has v_n been scheduled yet?

Multipliers

U = { }
T = { }

S = { }

ALUs

U = { v₉, v₉ }
T = { }

S = { v₉ }

Set vertices in S to start at 6

I = 6 + 1 = 7

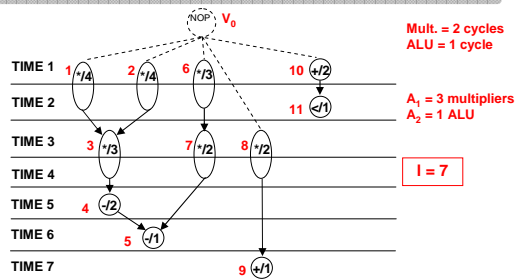
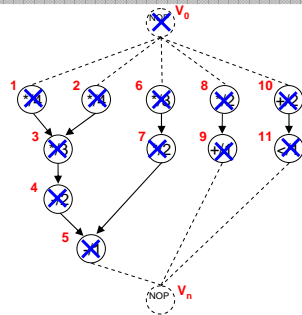
No. Repeat loop.

ECE 474a/575a
Susan Lysecky

69 of 72

LIST_L Scheduling

Example 2



Mult. = 2 cycles
ALU = 1 cycle

A₁ = 3 multipliers
A₂ = 1 ALU

I = 7

Step 2/3

U_{i,k} = candidate operations with predecessors finished at I
T_{i,k} = unfinished operations

Step 4

S = subset set of vertices in U and T such that U + T is <= a, where labels are maximal

Step 5

Schedule vertices in S to time step I

Step 6

I = I + 1

Step 7

Has v_n been scheduled yet?

Multipliers

U = { }
T = { }

S = { }

ALUs

U = { v₉ }
T = { }

S = { v₉ }

Set vertices in S to start at 7

I = 7 + 1 = 8

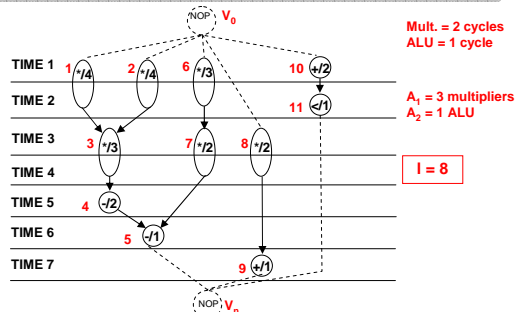
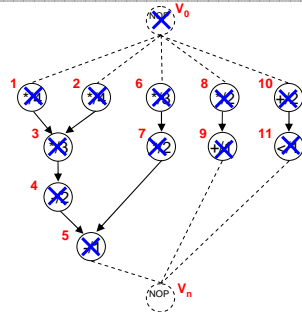
No. Repeat loop.

ECE 474a/575a
Susan Lysecky

70 of 72

LIST_L Scheduling

Example 2



Step 2/3

$U_{i,k}$ = candidate operations with predecessors finished at i
 $T_{i,k}$ = unfinished operations

Step 4

S = subset set of vertices in U and T such that $U + T \leq a$, where labels are maximal

Step 5

Schedule vertices in S to time step i

Step 6

$i = i + 1$

Step 7

Has v_n been scheduled yet?

Multipliers

$U = \{ \}$
 $T = \{ \}$

$S = \{ \}$

ALUs

$U = \{ \}$
 $T = \{ \}$

$S = \{ \}$

Mult. = 2 cycles
 ALU = 1 cycle

$A_1 = 3$ multipliers
 $A_2 = 1$ ALU

I = 8

Set vertices in S to start at 8

$i = 8 + 1 = 9$

Yes. We are done.

ECE 474a/575a
 Susan Lysecky

71 of 72

Conclusion

- Considered several types of scheduling algorithms
 - Unconstrained Scheduling - ASAP
 - Latency-Constrained Scheduling – ALAP
 - Resource-Constrained Scheduling – Hu's Algorithm
- Practical Scheduling problems possibly include multiple-cycle operations with different types
 - Minimum-Latency, Resource-Constrained and Minimum-Resource, Latency-Constrained problems become difficult to solve efficiently
 - Heuristics developed
 - List Scheduling (LIST_L)
 - List Scheduling (LIST_R)
 - Force-directed Scheduling
 - Trace Scheduling
 - Percolation Scheduling

ECE 474a/575a
 Susan Lysecky

72 of 72