Conflict Graph for each Resource:

Multiplex:

Atlas S:

N = P

Time 1

Time 2

Time 3

Time 4

Node
Interval Graphs for Scheduled Sequencing Graphs:

Intervals $I = \{ [l_i, r_i] ; i = 1, 2, \ldots, 12 \}$

$L_i$: left edge

$r_i$: right edge

Given a scheduled sequencing graph, create a set of intervals for each resource type:

For each $v_i$, create interval with $L_i$: start time of $v_i$ and $r_i$: end time of $v_i$

Multipliers:

$[1, 2], [1, 3], [2, 3], [2, 4], [3, 4], [3, 5], [4, 6]

v_1, v_2, v_3, v_4, v_5, v_6$

It captures some information as conflict graph, and supports multi-cycle intervals.

LEFT-EDGE algorithm can optimally find vertex coloring for intervals.
LEFT: EDGE(1) 

sort elements of $I$ in a list $L$ in ascending order of $l_i$

$c = 0$

while (some interval has not been scheduled) 

    $s = 0$

    $r = 0$ //initialize coordinate of rightmost edge in $s$

    while ($E$ and element in $L$ whose left edge $l_i \geq r$)

        $s = $ first element in $L$ with $l_i \geq r$

        $s = s + \{s\}$

        $r = r$

        remove $s$ from $L$

    $c = c + 1$

label elements of $S$ with color $c$
Intervals for **AUs**:

\[ v_0, [2,3], [3,4], [4,5], [4,5] \]

Solid Intervals for **AUs**

\[ v_0, v_1, v_2, v_3, v_4 \]

**LDM-Graph**:

\[ L = \{ v_0, v_1, v_2, v_3, v_4 \} \]

\[ c = 0 \]

\[ s \triangle 23, r = 0 \]

\[ s \triangle v_0, s \triangle v_0, v_3, r = 2 \]

\[ s \triangle v_1, s \triangle v_0, v_3, r = 2 \]

\[ s \triangle v_4, s \triangle v_0, v_3, r = 4 \]

\[ s \triangle v_5, s \triangle v_0, v_3, r = 5 \]

\[ c = 1, c_0 = 1, c_1 = 1, c_2 = 1, c_3 = 1 \]

\[ L = \{ v_3 \} \]
Multicycle Latency:

Multiplier: 2 cycles
ALU: 1 cycle

Scheduled sequencing graph for LPSL U
- Edges within scheduled sequence graph are variables that must be shared within registers
- Registers can be shared, just like resources
- Each variable has a lifetime that is an interval from its birth to its death, where the birth is the time a value is generated and death is the last time that variable is used

- Use vertex coloring (LEFT-EDGE) to determine register binding

**Intervals**
- \( I_1 = [1, 2] \)
- \( I_2 = [1, 2] \)
- \( I_3 = [2, 3] \)
- \( I_4 = [2, 4] \)
- \( I_5 = [3, 4] \)
- \( I_6 = [3, 4] \)

**Sorted Intervals**

**Coloring**
- \( c_1 = 1, c_3 = 1, c_5 = 1 \)
- \( c_2 = 2, c_4 = 2, c_6 = 2 \)

**Need 2 registers**