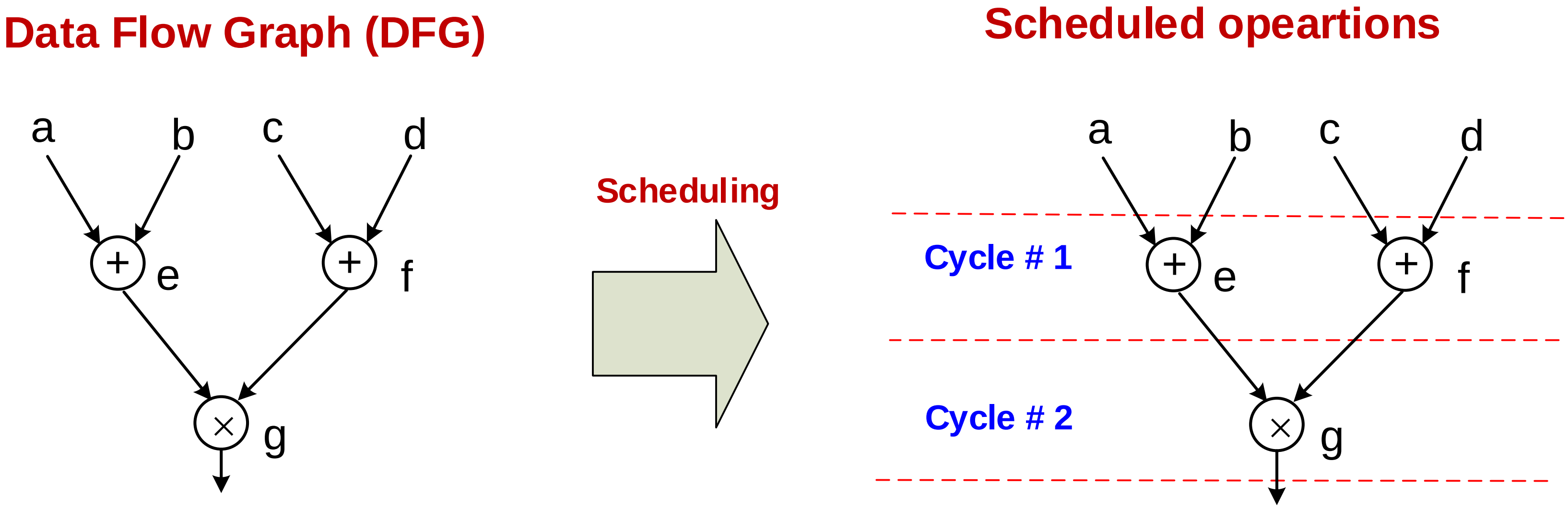
# Scheduling of DFG Operations

Digital System Design

Dr. Bassam Jamil

# Scheduling Definition

- Given number of operations (***O***) and set of constrains, scheduling assigns operations to cycles to optimize (i.e. minimize) execution cycles (***C***).
- Constraints are dependencies (***D***) and available resources (i.e. ***R*** units) .

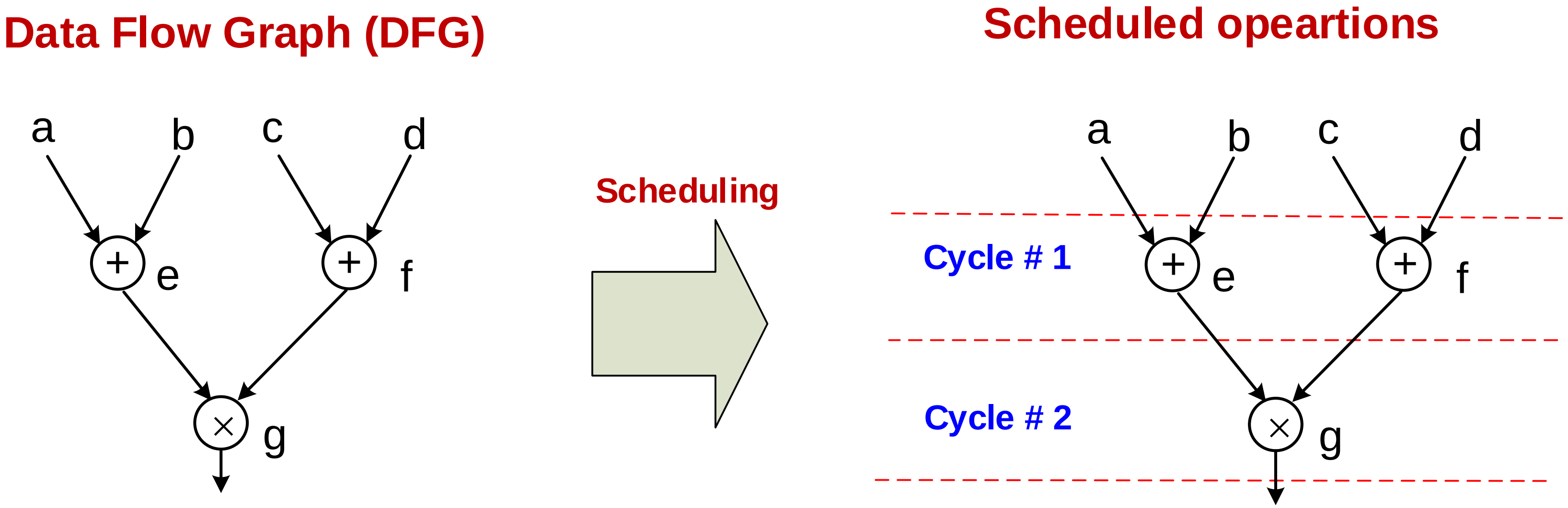

Data Flow Graph (DFG)

Scheduling

Scheduled opeartions

# Scheduling Definition

- Data flow graph (***DFG***) provides operations (***O***) and dependencies (***D***).
- Scheduling (***S***) maps operations (O) to resources in specific cycles. So, **$S : O \rightarrow C \times R$**



Data Flow Graph (DFG)

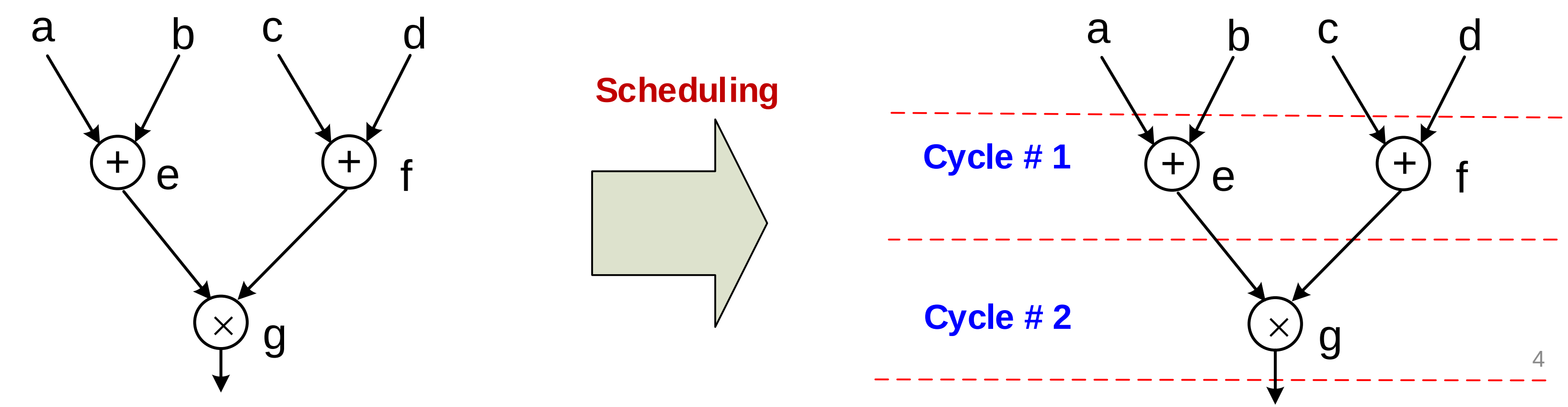
Scheduling

Scheduled opeartions

# Scheduling Example

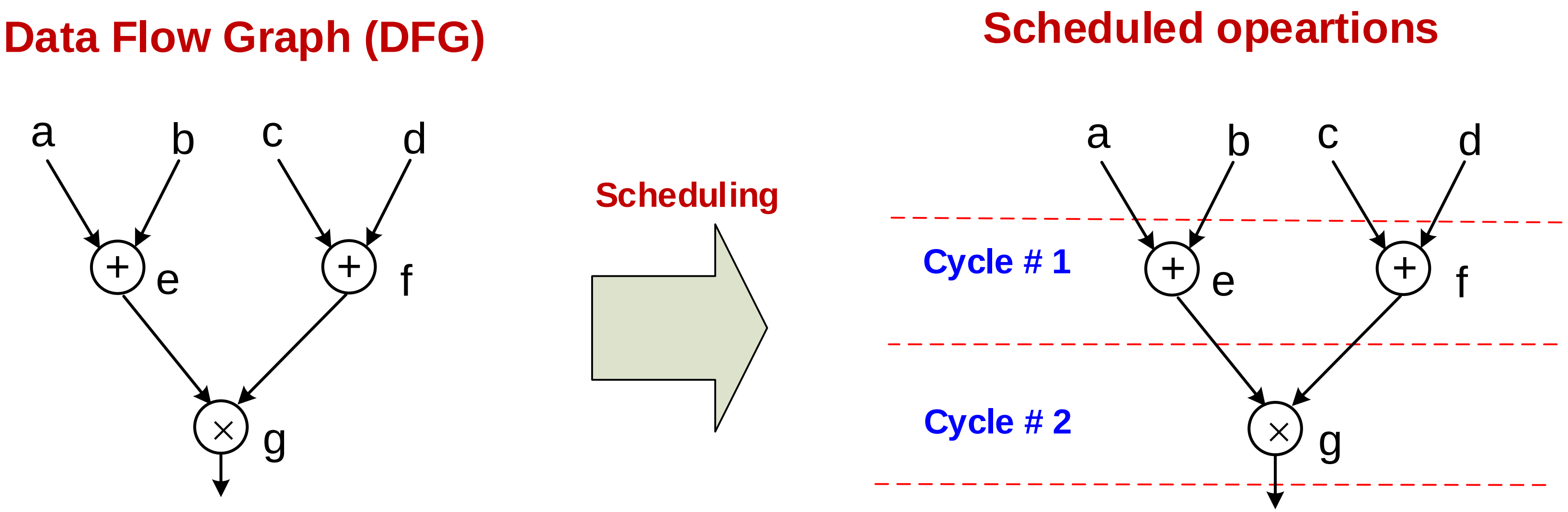
- Consider the below DFG. Assume we have 2 adders (Add1, Add2) and one multiplier (MUL).
- Scheduling produces the following mapping:
  - “a+b” is mapped to C1 (cycle 1) using Add1
  - “c+d” is mapped to C1 using Add2
  - “e×f” is mapped to C2 using MUL

**Data Flow Graph (DFG)** **Scheduled opeartions**



# Scheduling Example, Cont.

- Number of cycles is 2 cycles.
- The example was easy, since we provide optimum number of resources.



Data Flow Graph (DFG)

Scheduled opeartions

# Optimum Scheduling Definition

- Define: operations (***O***), dependencies (***D***), resources (***R***), cycles (***C***) and cost function (***f***).
- Optimum scheduling (***S***):
  1. **maps** operations to cycles and resources**: O → C × R**
  2. **minimizes** (optimizes) the cost function (***f***)
- Examples of cost function (***f***):
  - number of cycles
  - consumed power/energy
  - resource utilization
  - ……
- We assume f = number of cycles. So, our scheduling must minimize number of cycles.

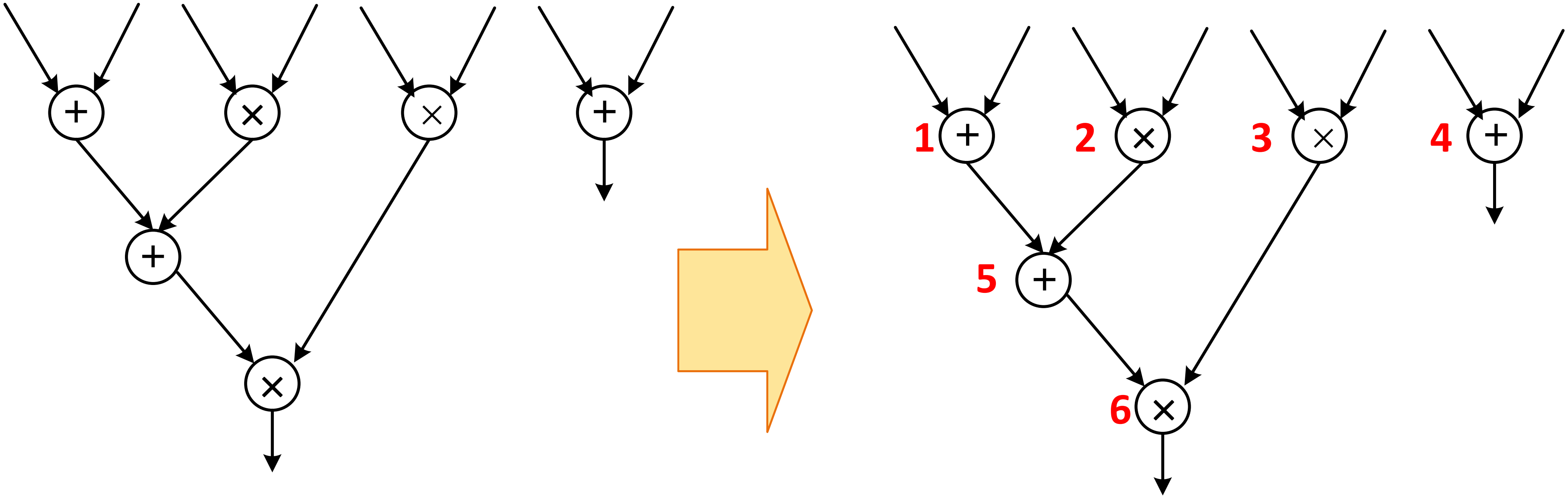
# Scheduling Methods

- There several methods for scheduling:
  - ASAP: schedule operations in the earliest possible cycle
  - ALAP: schedule operations in the latest possible cycle
  - (Speed*) Optimized scheduling:
    - scheduling prioritize operations
    - scheduling  operations with higher priority

* in this lecture: optimization goal is to reduce number of cycles. Other optimization result in different algorithms.

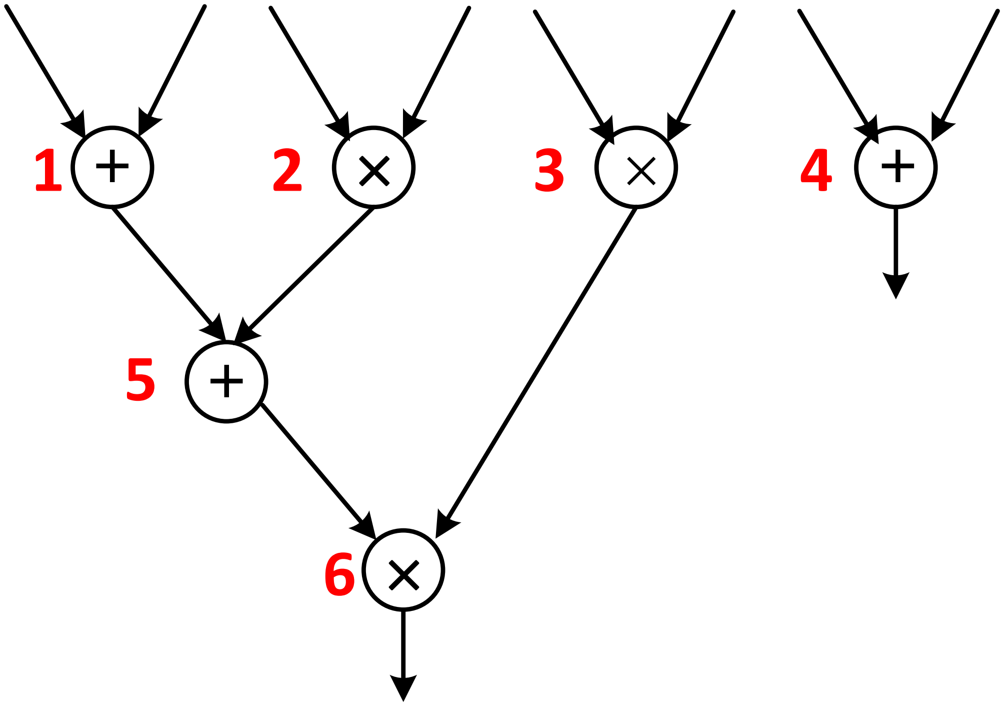
# Soring Operations

- ASAP and ALAP requires sorting operations.
- Easiest is to sort them topologically according DFG



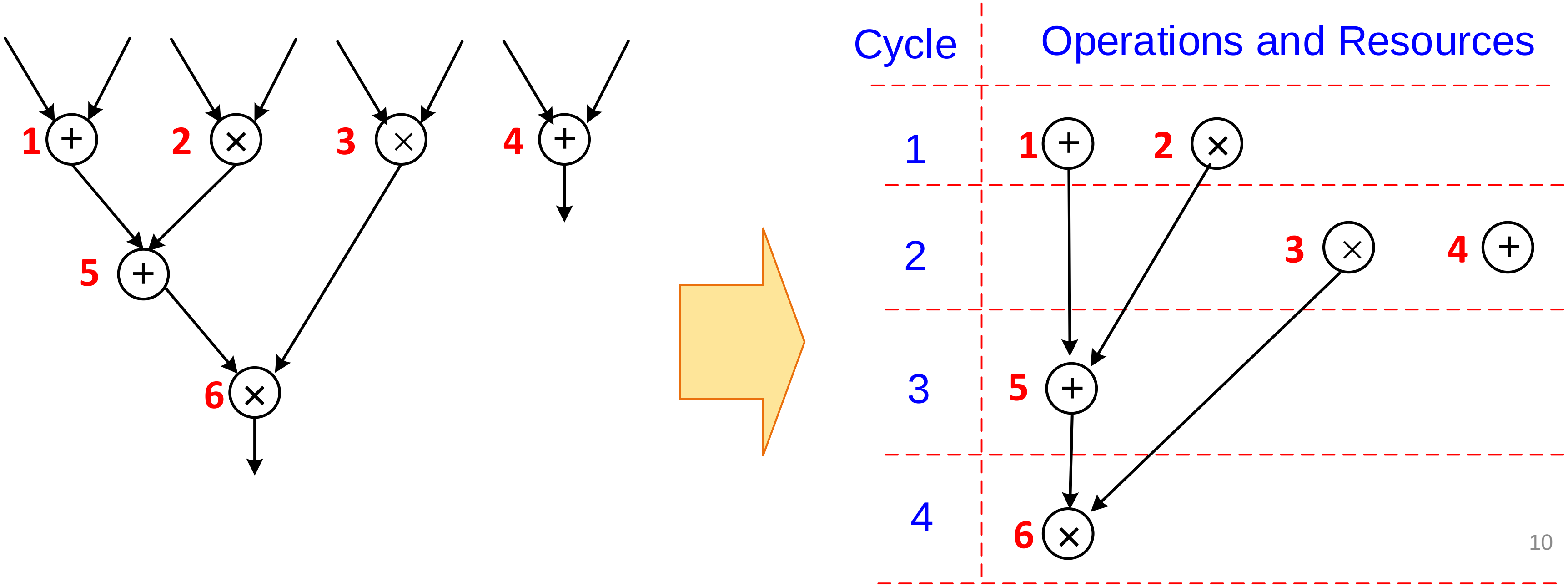
# Example (1)

- In the next slides, we will assume:
  1. Below DFG; operations already sorted.
  2. Resources : **one adder**, **one multiplier**



# ASAP with Example (1)

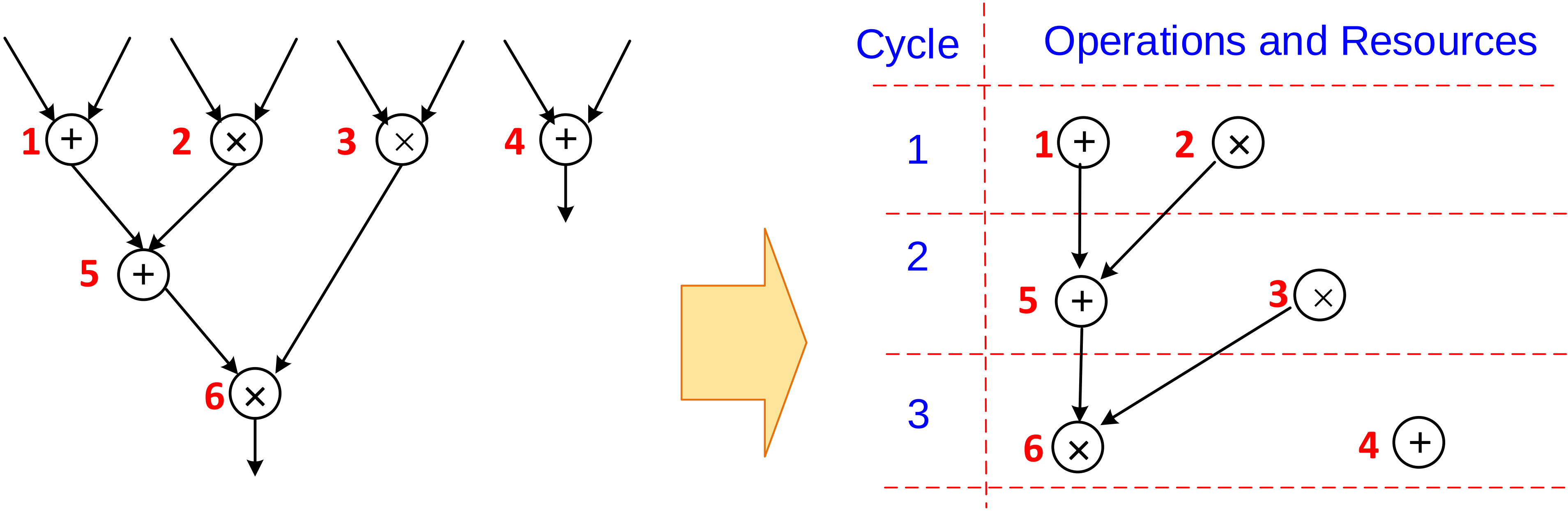
1. Sort the operations topologically according DFG
2. Schedule operations in the sorted order by placing them in the earliest possible cycle, provided that:
   a) dependencies are not violated
   b) there is a free resource



| Cycle | Operations and Resources |
|---|---|
| 1 | 1 + 2 × |
| 2 | 3 × 4 + |
| 3 | 5 + |
| 4 | 6 × |

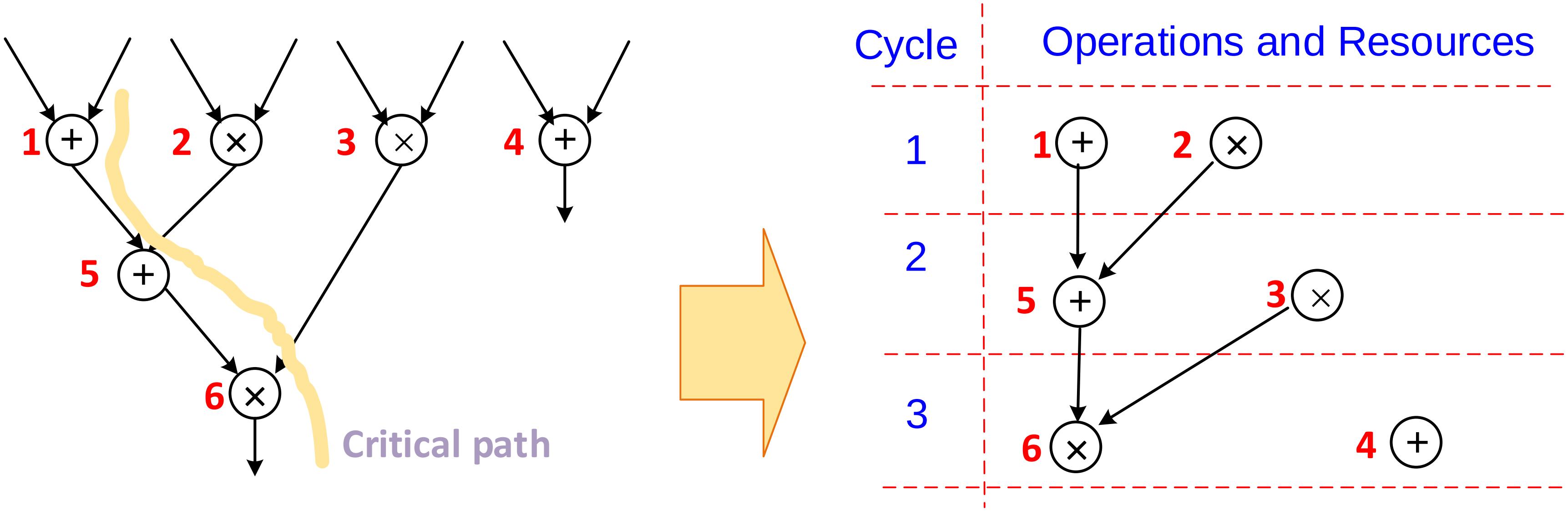
# ALAP with Example (1)

1. Sort the operations topologically according DFG
2. Schedule operations in the reversed order by placing them in the latest possible cycle, provided that:
   a) dependencies are not violated
   b) there is a free resource



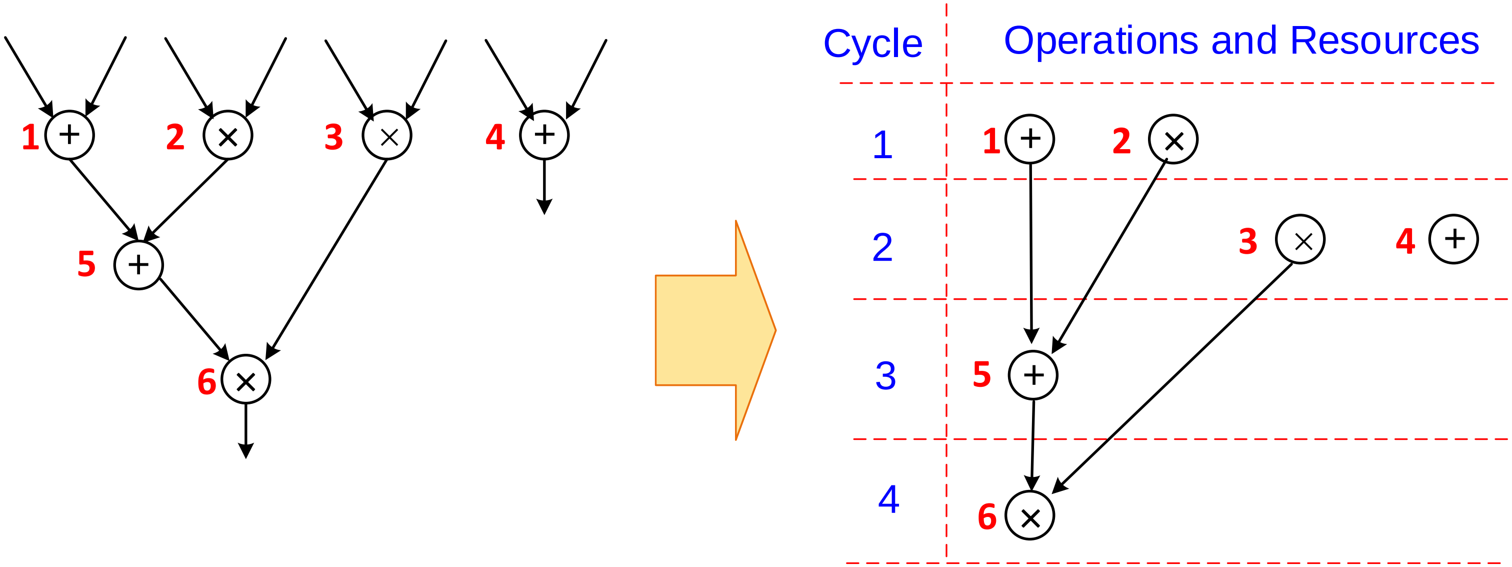
# Optimum Scheduling with Example (1)

- Optimize critical path
  - Find most critical paths.
  - Schedule operation in each cycle, giving critical paths higher priority.



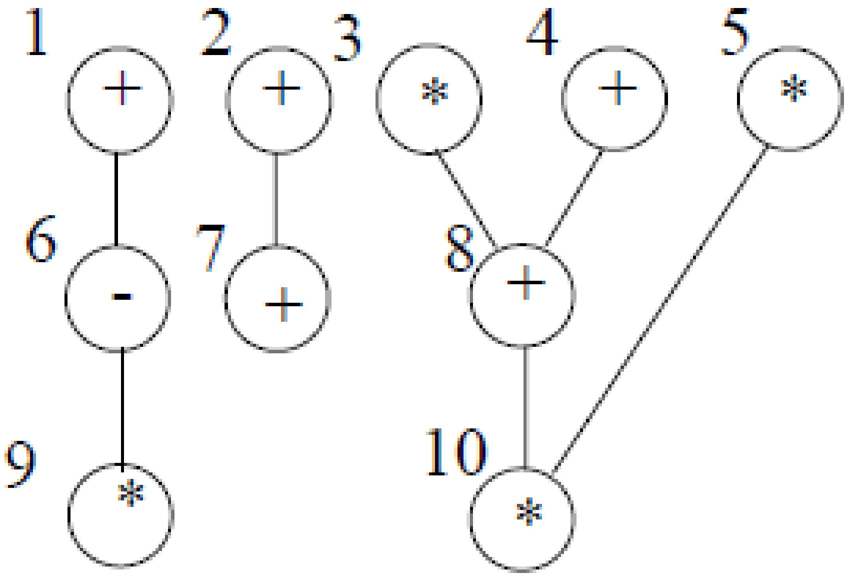
# Resource Utilization

- Resource utilization refers the percentage of time the resource was busy.
- In example (1) using ASAP:
  - Adder was used in 3 cycles (out of 4) →Adder unitization is 75%
  - Multiplier utilization is 75%

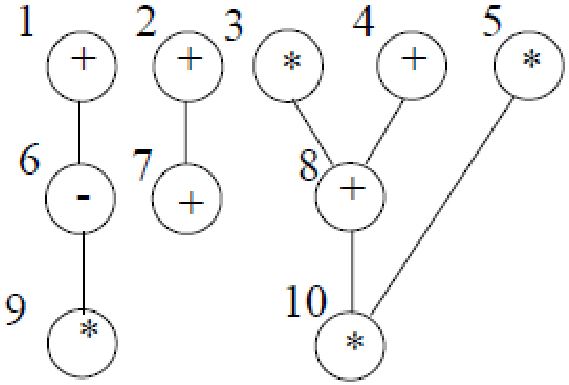


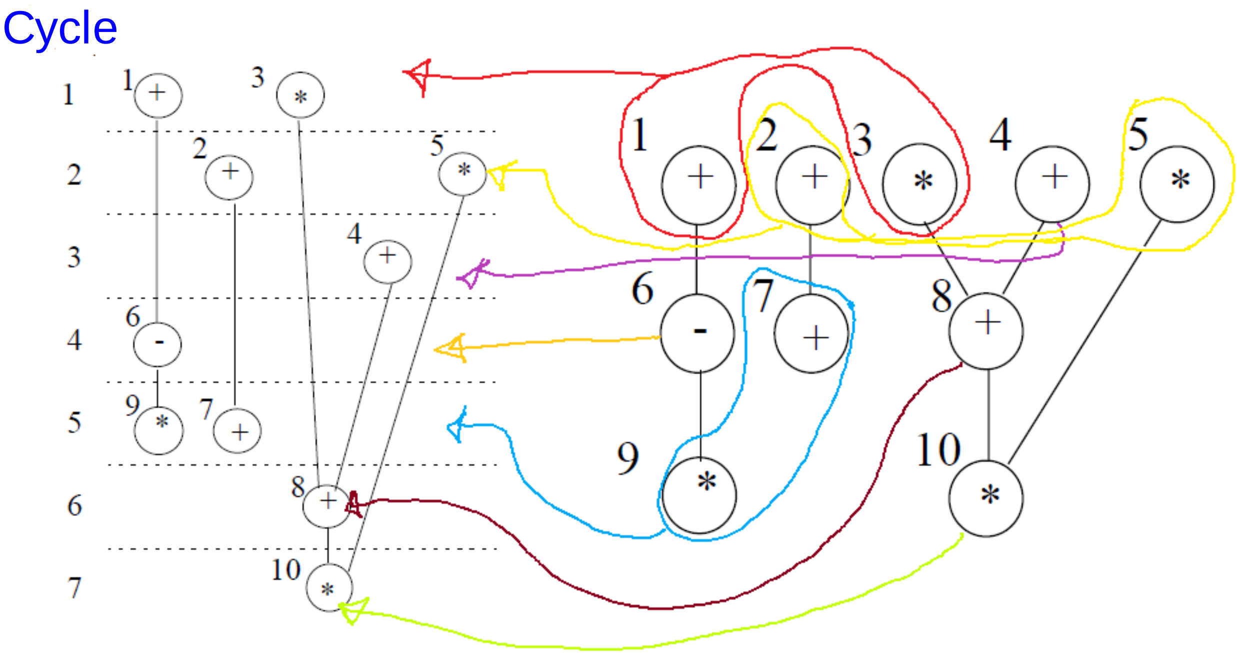
# Example (2)

- In the next slides, we will assume:
  1. Below DFG
  2. Resources : **one adder**, **one multiplier**

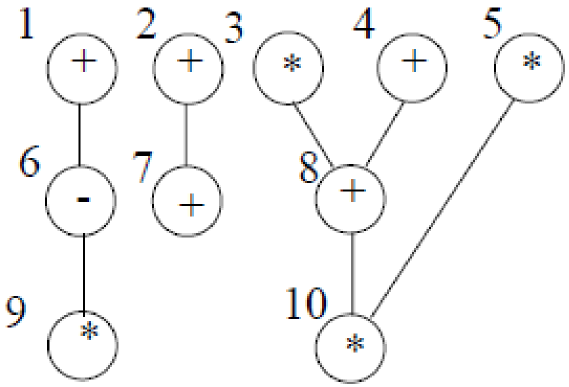


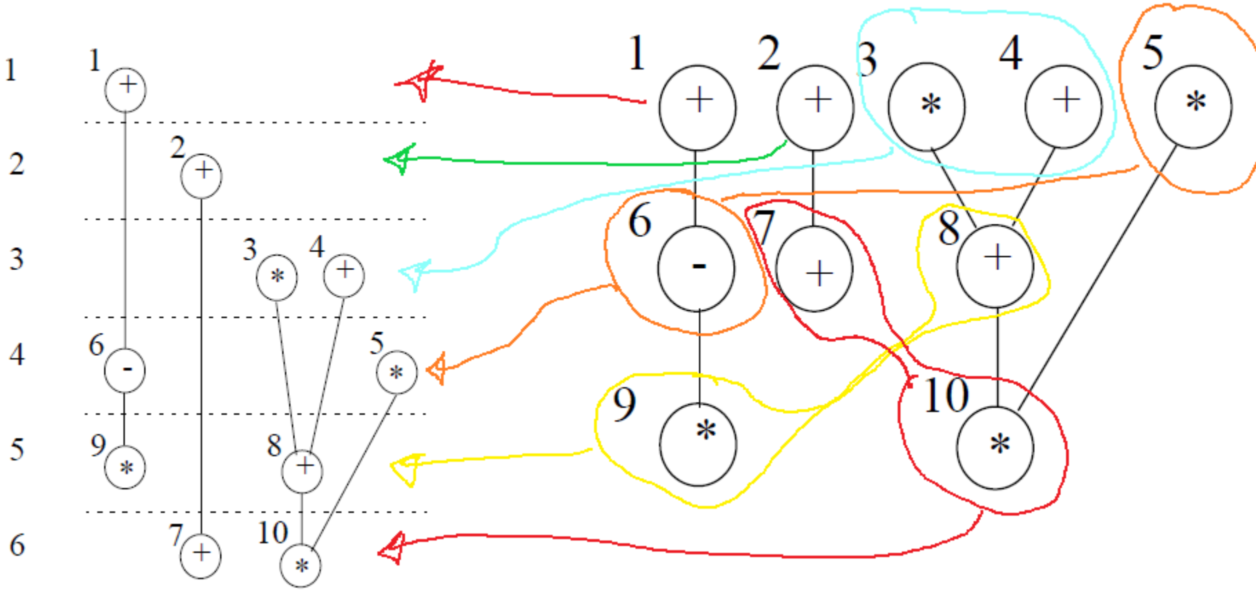
# ASAP with Example (2)



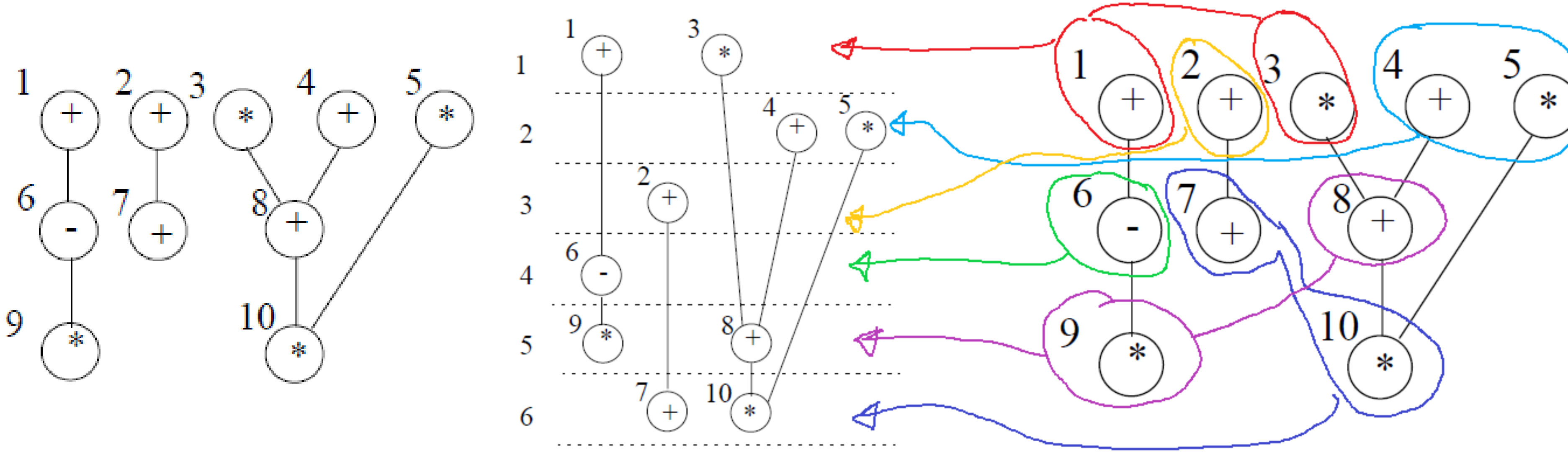

Cycle

# ALAP with Example (2)





# Optimum Scheduling with Example (2)

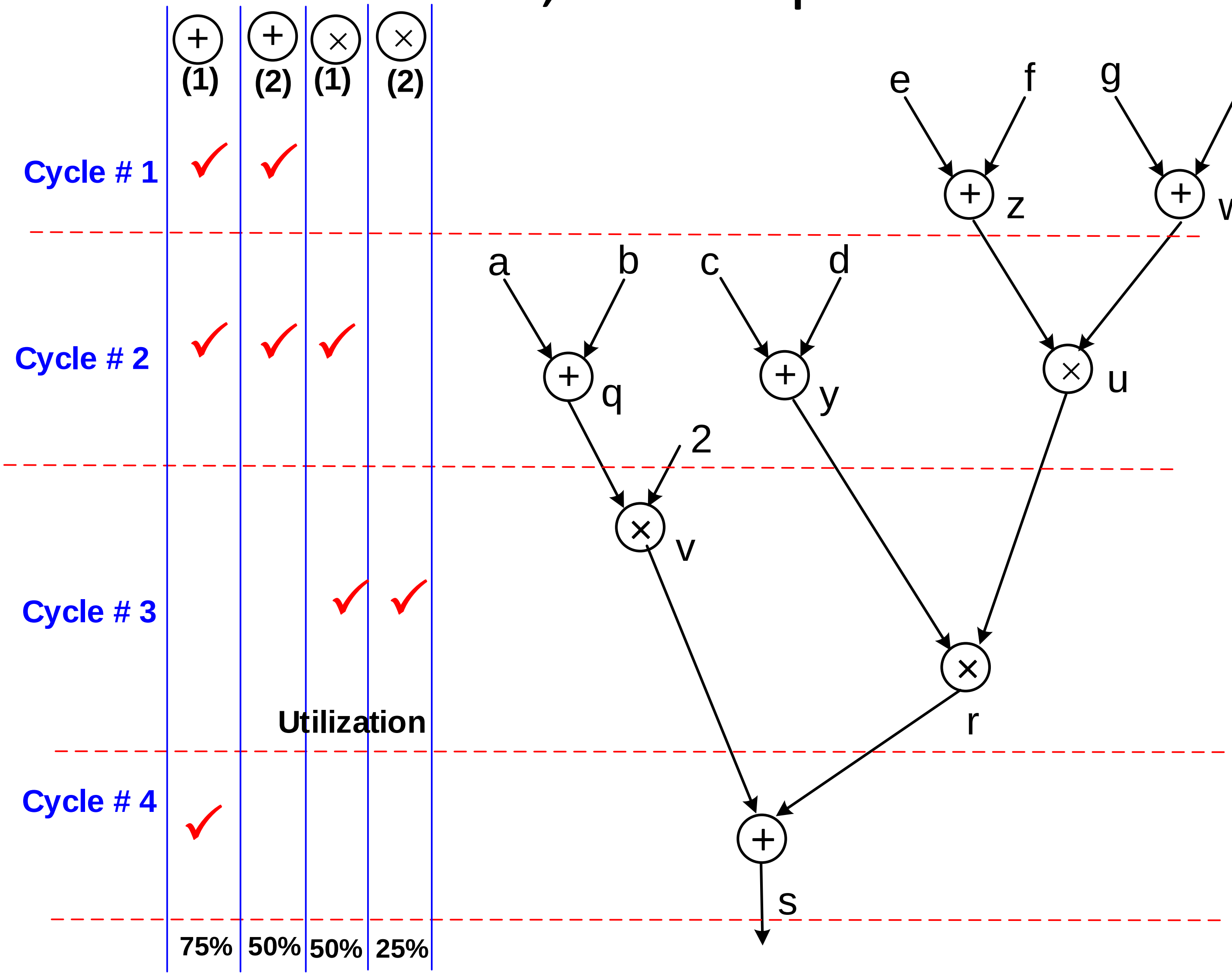


# Observation

- In Example (1) and Example (2)
  - ALAP and Optimum did better than ASAP

# Example 3: optimum

- Resources: 2 adders, 2 multipliers



| | + (1) | + (2) | × (1) | × (2) |
|---|---|---|---|---|
| Cycle # 1 | ✓ | ✓ | | |
| Cycle # 2 | ✓ | ✓ | ✓ | |
| Cycle # 3 | | | ✓ | ✓ |
| Cycle # 4 | ✓ | | | |
| Utilization | 75% | 50% | 50% | 25% |