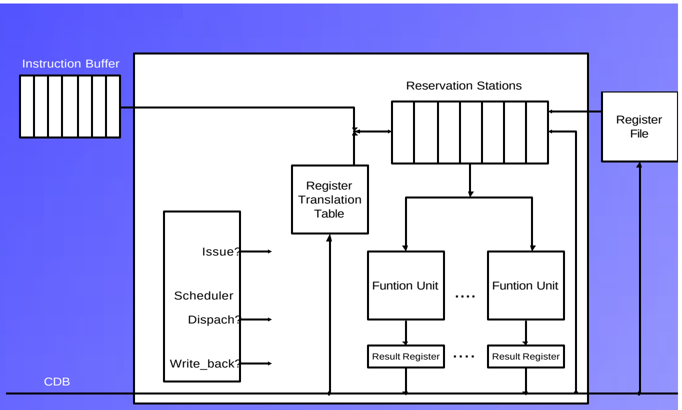


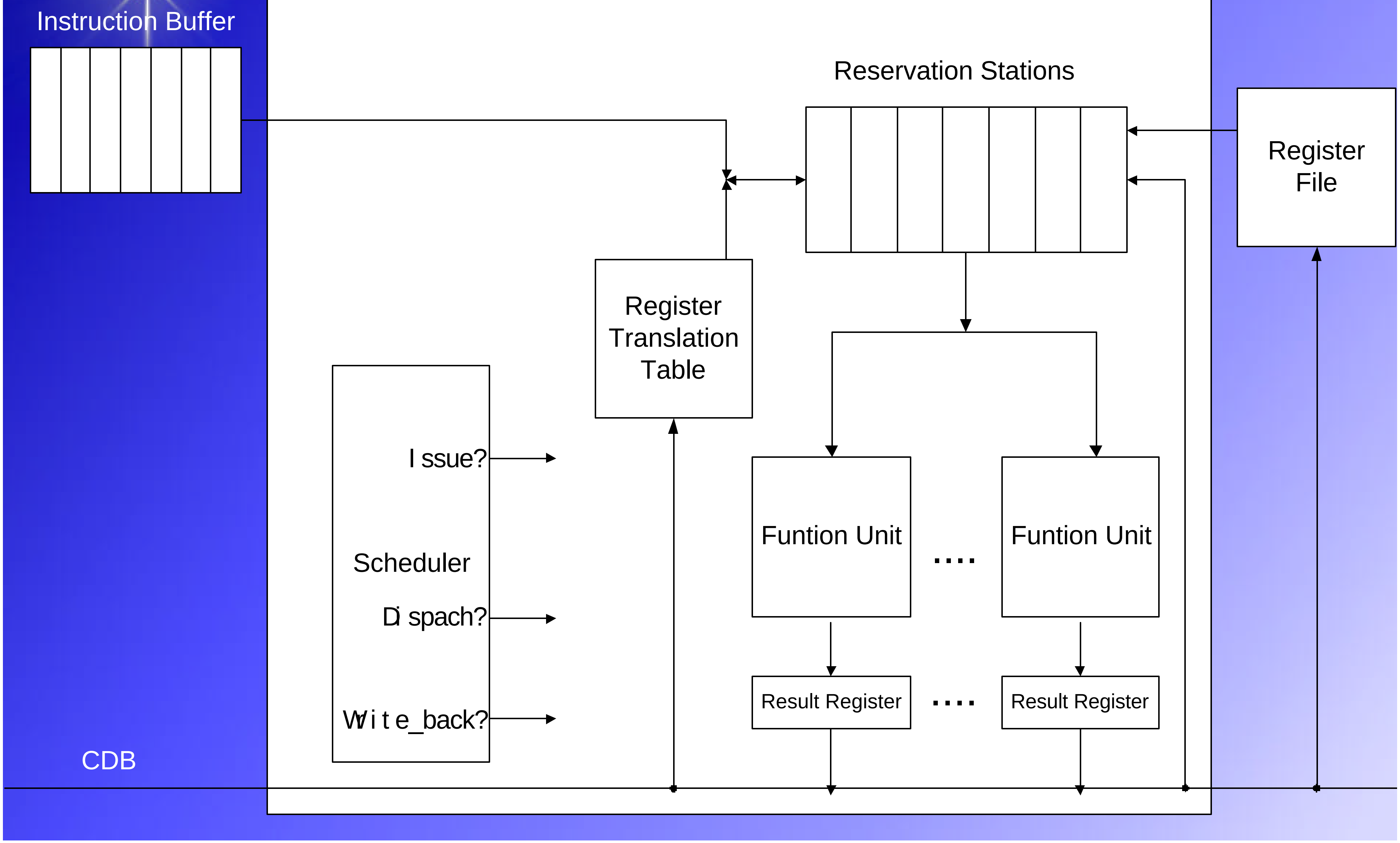
Is Tomasulo’s Algorithm Optimal?

Tian Sang and Lin Liao

Outline

- Tomasulo's Processor and Our Assumptions
- What Does Optimality Mean?
  - Reference Processor
  - Global Optimality
  - Constrained Optimality
  - Local Optimality
- Is Tomasulo's Processor Optimal?
  - Infinite Function Units and Infinite CDBs
  - Limited Function Units and Infinite CDBs
  - Infinite Function Units and Limited CDBs
  - Limited Function Units and Limited CDBs

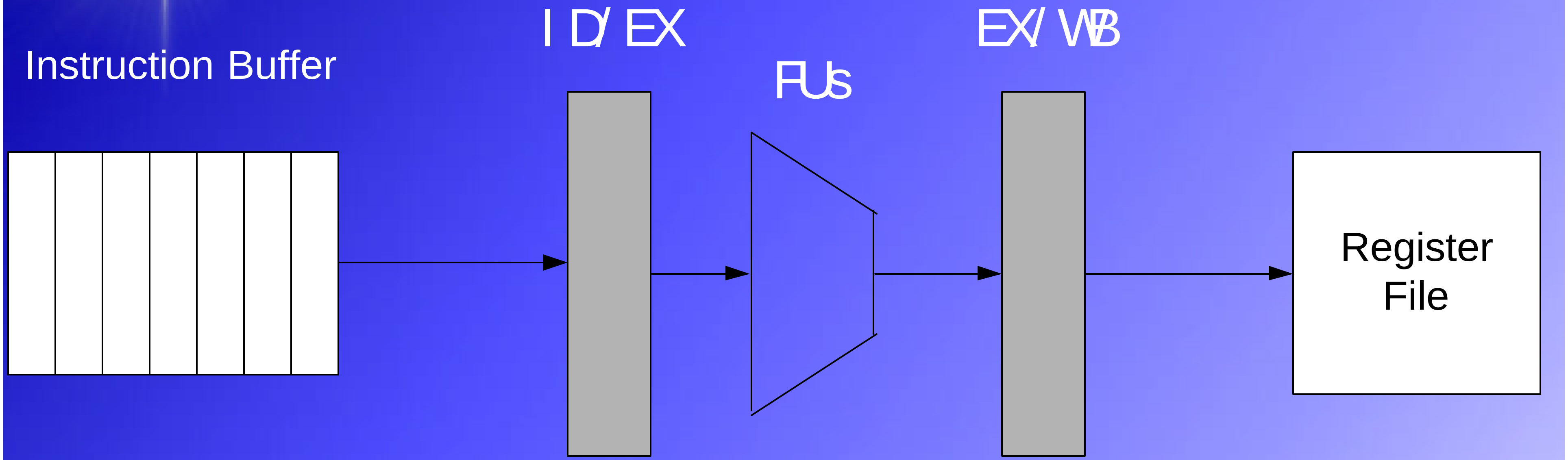
Tomasulo's Processor Model

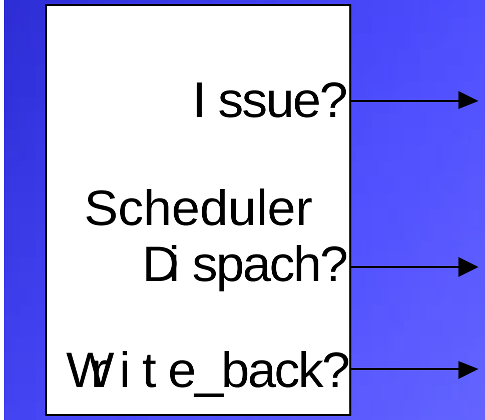


Our Assumptions for Tomasulo's Processor

- No branch instructions
- No Load/Store
- In-order and single issue
- One central reservation stations for all function units with infinite entries
- Instructions can be decoded and issued into reservation stations in one cycle

Ideal Reference Processor





- 3 pipeline stages
- Same FU with Tomasulo’s
- Always does optimal scheduling

Global Optimality

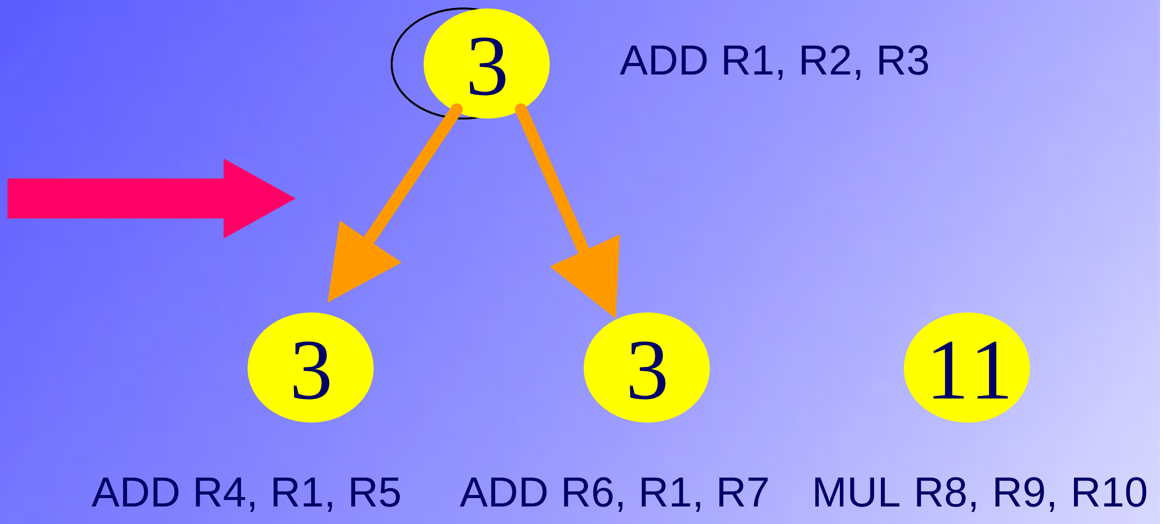
- ***Global optimal value:*** the minimum cycles needed for the *ideal reference processor* to finish all the instructions and guarantee the data dependencies

ADD R1, R2, R3

ADD R4, R1, R5

ADD R6, R1, R7

MUL R8, R9, R10



3 ADD R1, R2, R3

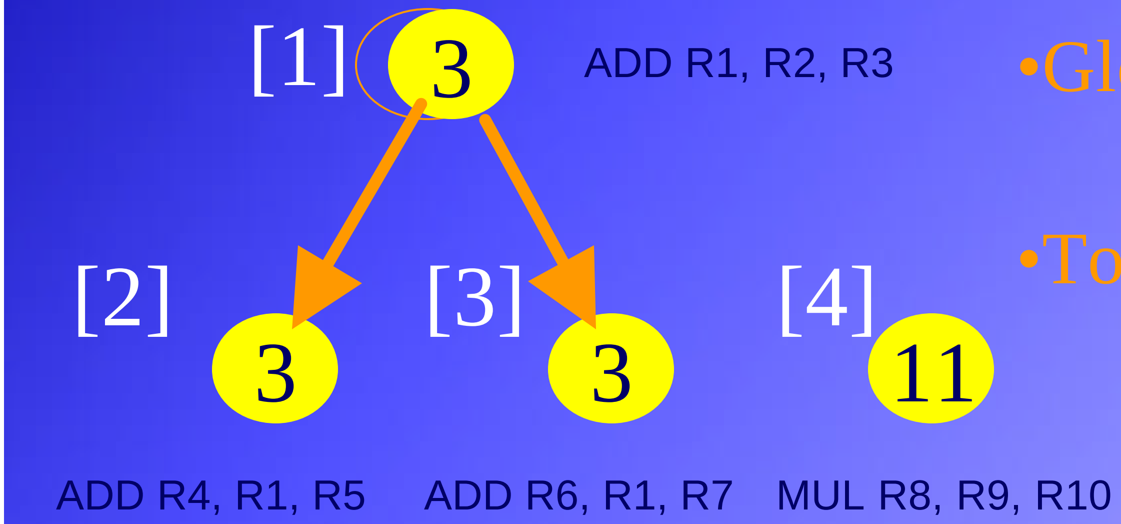
3 ADD R4, R1, R5

3 ADD R6, R1, R7

11 MUL R8, R9, R10

Global Optimality (cont.)

- Claim: The Tomasulo’s processor cannot guarantee *global optimality* even though it has infinite function units and infinite CDB



[1] 3 ADD R1, R2, R3

[2] 3 [3] 3 [4] 11

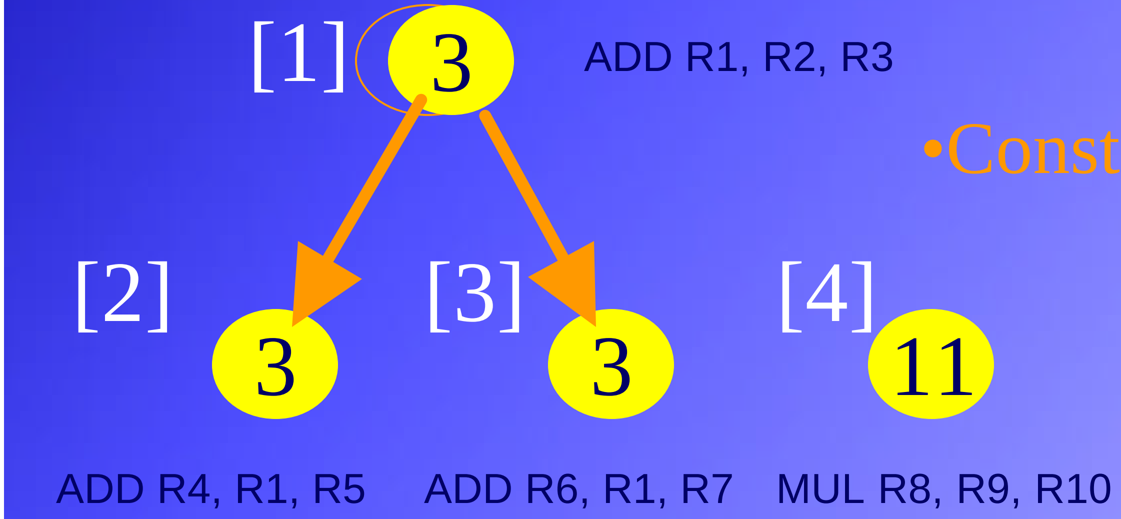
ADD R4, R1, R5 ADD R6, R1, R7 MUL R8, R9, R10

- Global Optimal Value: 11
- Tomasulo's Processor: 15

Constrained Optimality

- ***Constrained optimal value:*** minimum cycles needed for the *constrained processor* to finish all the instructions and guarantee the data dependencies.

- ***Constrained Reference Processor:*** existing ideal reference processor to issue one instruction per cycle and in order



- Constraint Optimal Value: 15

Constrained Optimality (cont.)

- Claim: If the Tomasulo’s processor has infinite functional units and infinite CDB, it can achieve *constrained optimality*.

- Proof: (Mathematical Induction)

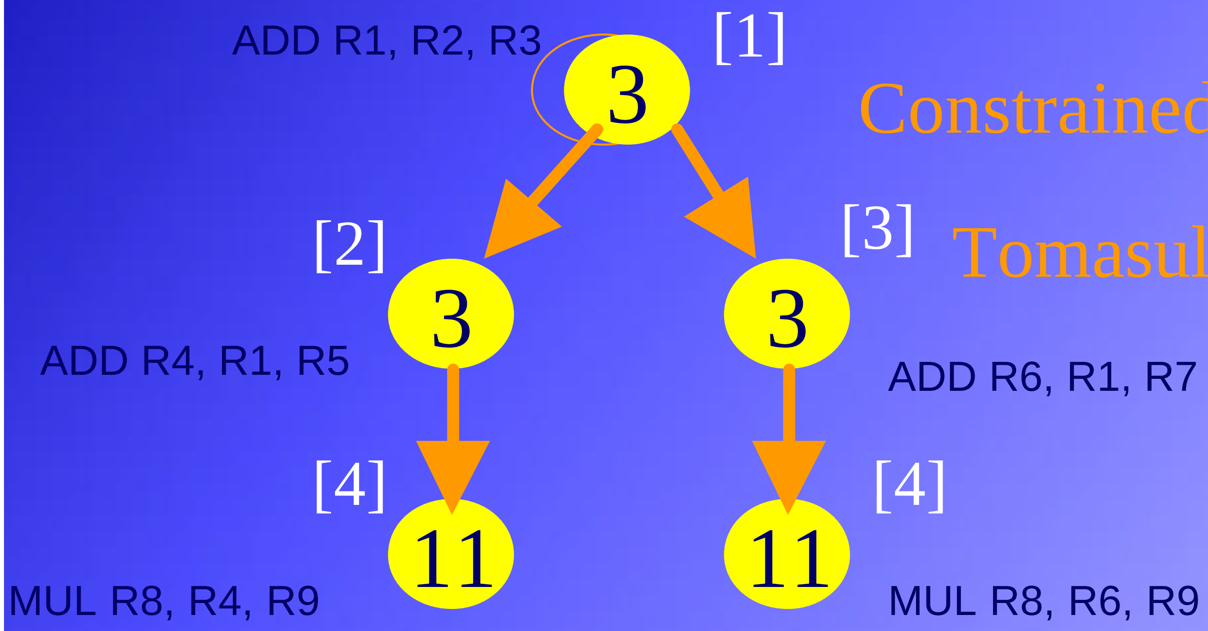
1) Basic step: n=1, only one instruction
2) Induction step: suppose in Tomasulo's processor, for the first k instructions, each instruction is able to be completed no later than reference processor, we get that the k+1 instruction is also able to.

Outline

- Tomasulo's Processor and Our Assumptions
- What Does Optimality Mean?
  - – Reference Processor
  - – Global Optimality
  - – Constrained Optimality
  - – Local Optimality
- Is Tomasulo's Processor Optimal?
  - – Infinite Function Units and Infinite CDBs
  - – Limited Function Units and Infinite CDBs
  - – Infinite Function Units and limited CDBs
  - – Limited Function Units and Limited CDBs

Limited Function Units

- Claim: The Tomasulo’s processor cannot guarantee *constrained optimality* if the number of function units is limited



ADD R1, R2, R3

[1]

3

[2]

3

ADD R4, R1, R5

[3]

3

ADD R6, R1, R7

[4]

11

MUL R8, R4, R9

[4]

11

MUL R8, R6, R9

Constrained Optimal Value: 17

Tomasulo’s processor: 20

Local Optimality

- ***Local Reference Processor:*** restrict *constrained reference processor* to schedule only based on the instructions already issued
- ***Local Optimal Value:*** minimum cycles needed for the *local reference processor* to complete all the instructions already issued and guarantee data dependencies

Local Optimality (cont.)

Given a set of issued instructions and their dependencies, find a scheduling strategy that achieves local optimal value.

- (P1): Suppose k identical, empty FU and each instruction needs 1 cycle
- Theorem 1: (P1) is NP-Complete

  [J. D. Ullman 1975]
- (P2): Same with (P1), except each instruction needs t cycles and FU are pipelined into t stages
- Theorem 2: (P2) is NP-Hard

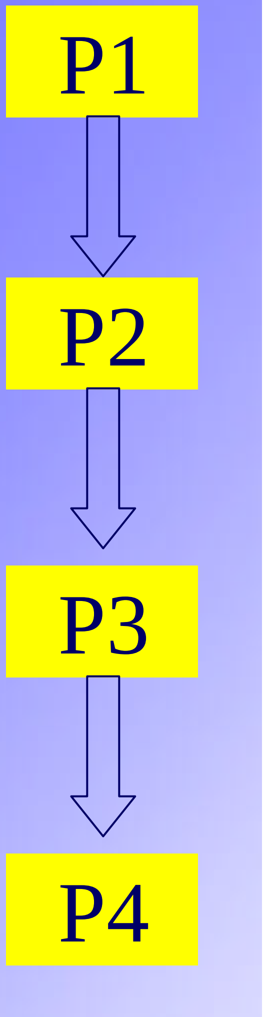
  (P1) is a special instance of (P2) when t=1

Local Optimality (cont.)

- (P3): Same with (P2), except 2 kinds of FU, addition needs t1 cycles, multiplication needs t2 cycles
- Theorem 3: (P3) is NP-Hard

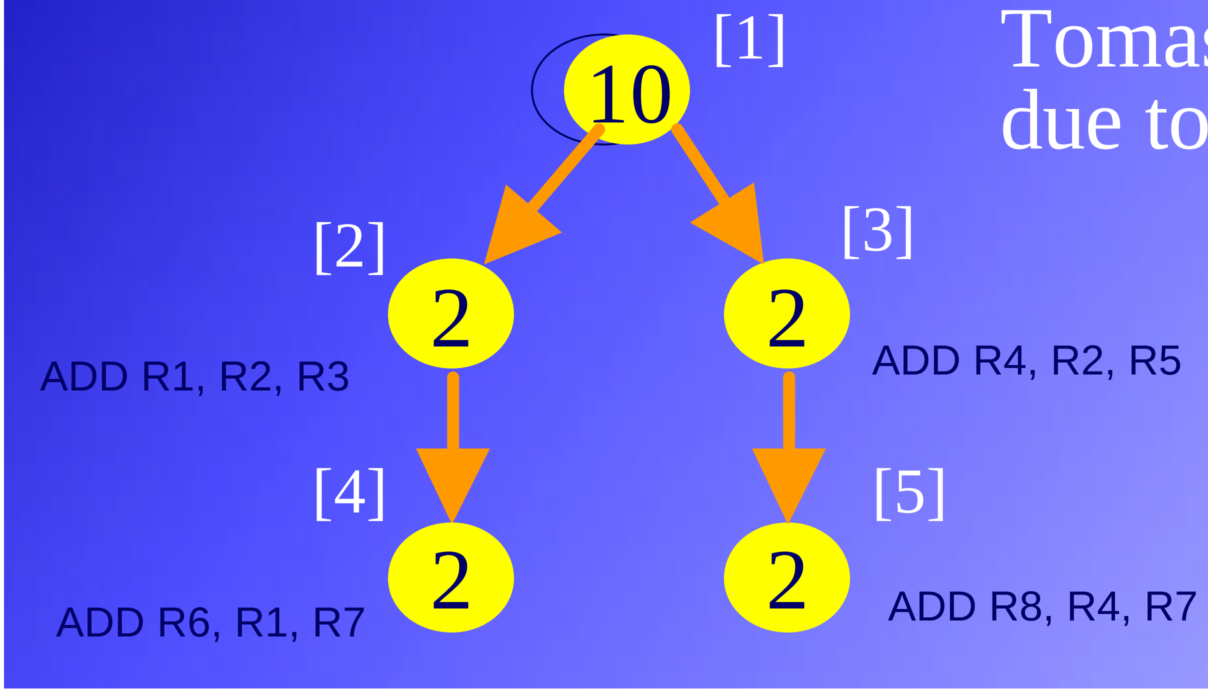
  When all the instructions are addition, (P3) becomes (P2)
- (P4): Same with (P3), except some instructions may be already in FU when scheduling
- Theorem 4: (P4) is NP-Hard

  Obviously a generalization of (P3)



Limited CDBs

- Claim: The Tomasulo's processor cannot guarantee *local optimality* if the number of CDB is limited

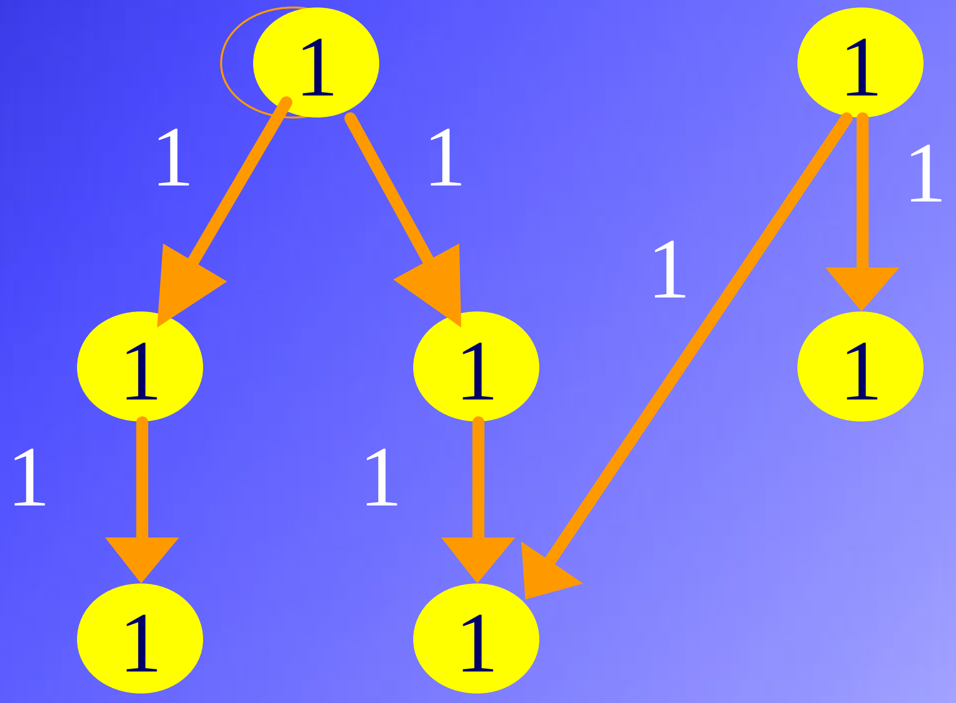


Scoreboarding: 16

Tomasulo: 17 (1 delay due to CDB conflict)

Limited FUs and limited CDBs

- The simplest case: UET-UCT scheduling
- k identical FUs and 1 CDB
- Both FUs and CDB use 1 time unit



Complexity of UET-UCT Scheduling

- Can we find a best schedule for Tomasulo's processor?
- [Lucian Finta and Zhen Liu 1994]

  The following 2 cases are NP-Hard:
- Each task may access a subset of processors (pre-allocation)
- Each task may access any processor (without pre-allocation)
- Mapping:

  pre-allocation → different FUs

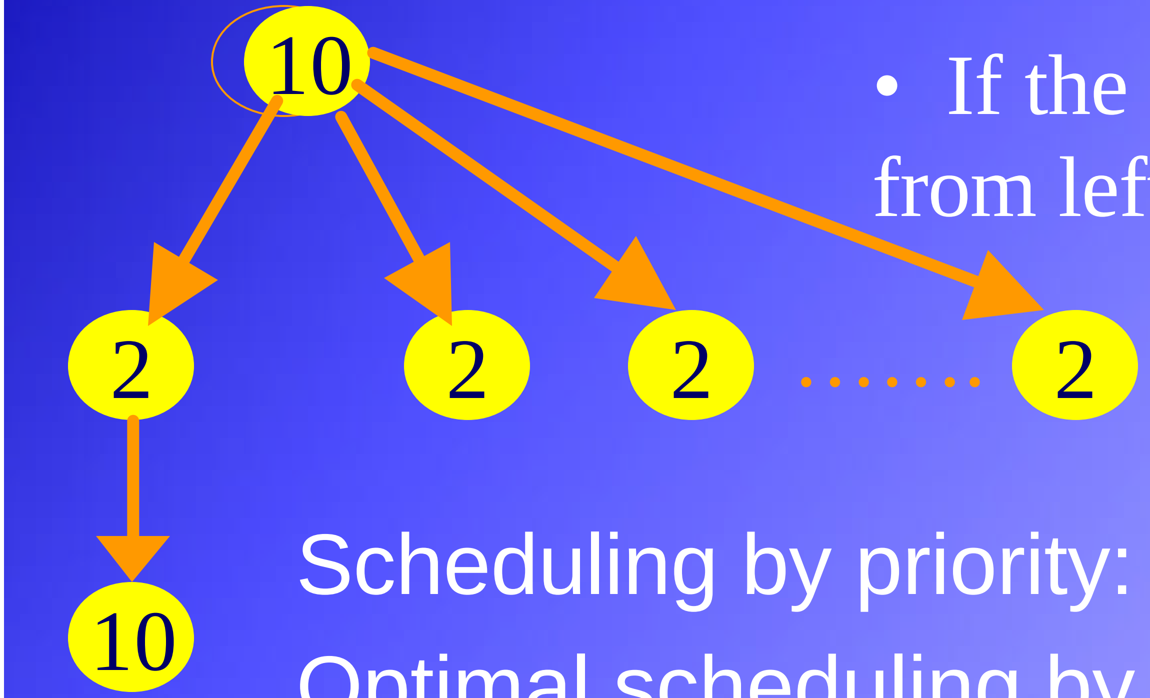
  without pre-allocation → identical FUs

Infinite FUs and limited CDBs

- Can we find a best schedule for Tomasulo’s model?
- In general this problem is NP-Hard, because the case with limited FUs and infinite CDBs can be reduced to it
- Maybe one exception: infinite FUs and 1 CDB

Infinite FUs and 1 CDB

- Resolving CDB conflicts by priority can be very bad



- If the priority increases from left to right

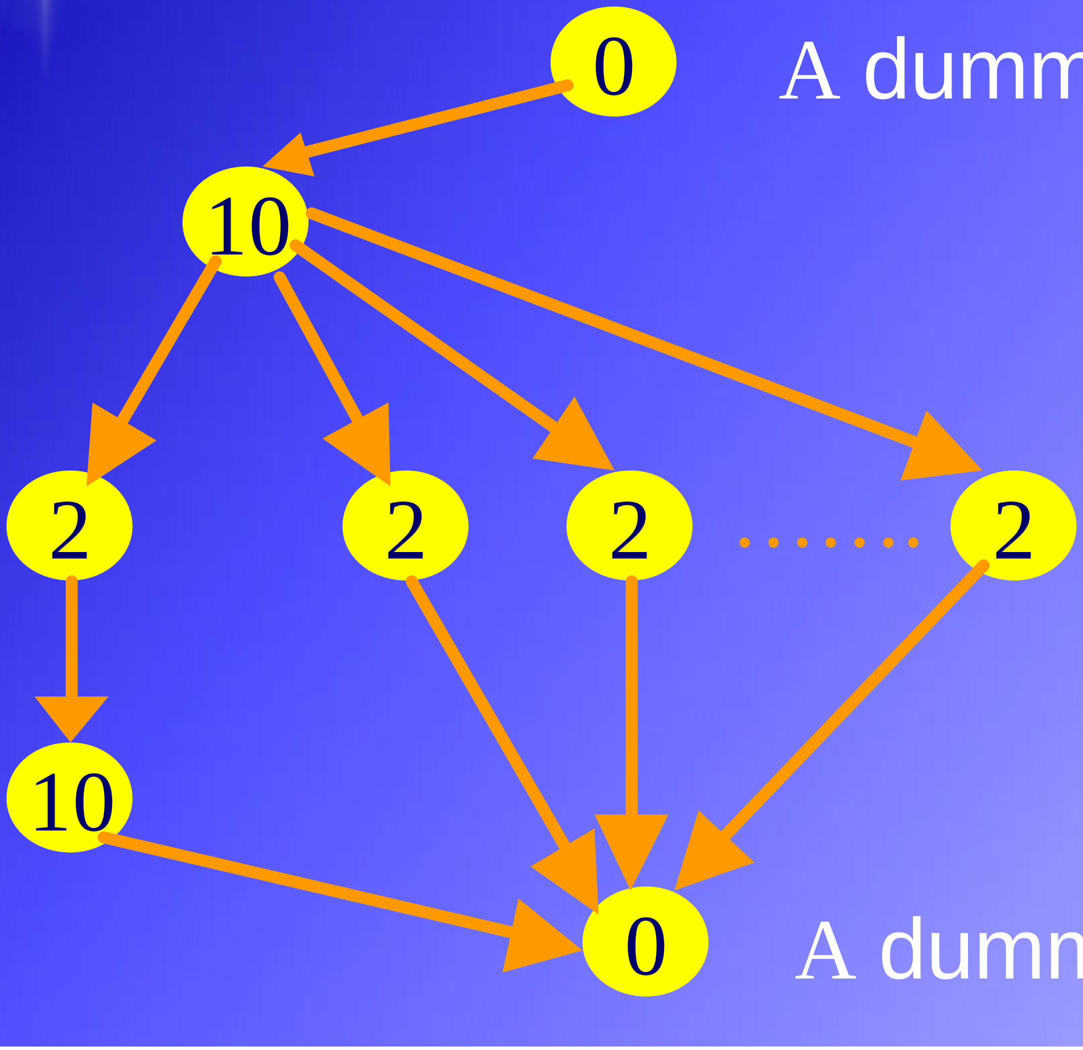
11 nodes this level

Scheduling by priority: 35

Optimal scheduling by critical path: 25

Scheduling by critical path

Complexity: O (V+E)



A dummy entry node

A dummy exit node

Conclusions

FU	CDB	Global	Constrained	Local	Best
∞	∞	NO	Yes	Yes	X
K	∞	NO	NO	NP-Hard	X
∞	N	NO	NO	NO	NP-Hard
K	N	NO	NO	NO	NP-Hard