Data Bases II

Distributed Deadlock: Obermarck Algorithm

Michele Beretta

michele.beretta@unibg.it

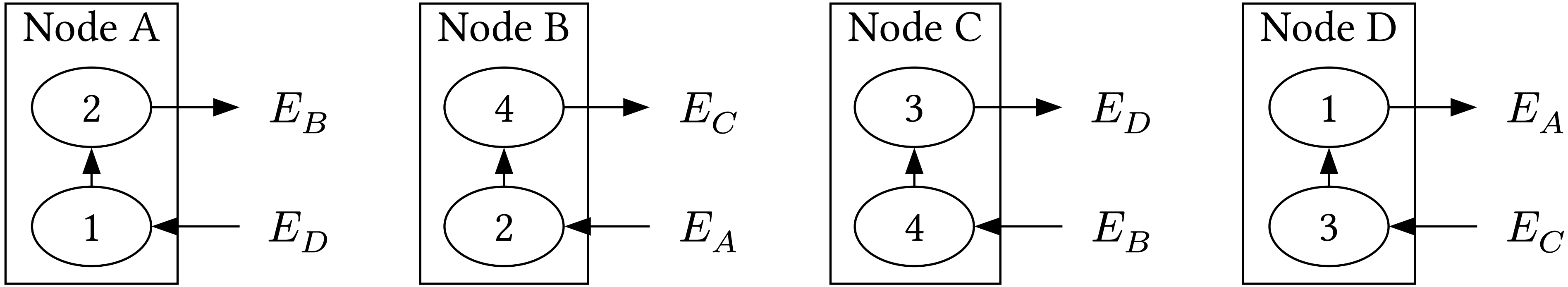
Unibg Seclab

Exercise E.1

Consider the following waiting conditions:

Node A: $E_D \rightarrow t_1, t_1 \rightarrow t_2, t_2 \rightarrow E_B$
Node B: $E_A \rightarrow t_2, t_2 \rightarrow t_4, t_4 \rightarrow E_C$
Node C: $E_B \rightarrow t_4, t_4 \rightarrow t_3, t_3 \rightarrow E_D$
Node D: $E_C \rightarrow t_3, t_3 \rightarrow t_1, t_1 \rightarrow E_A$

Indicate whether there is a deadlock.

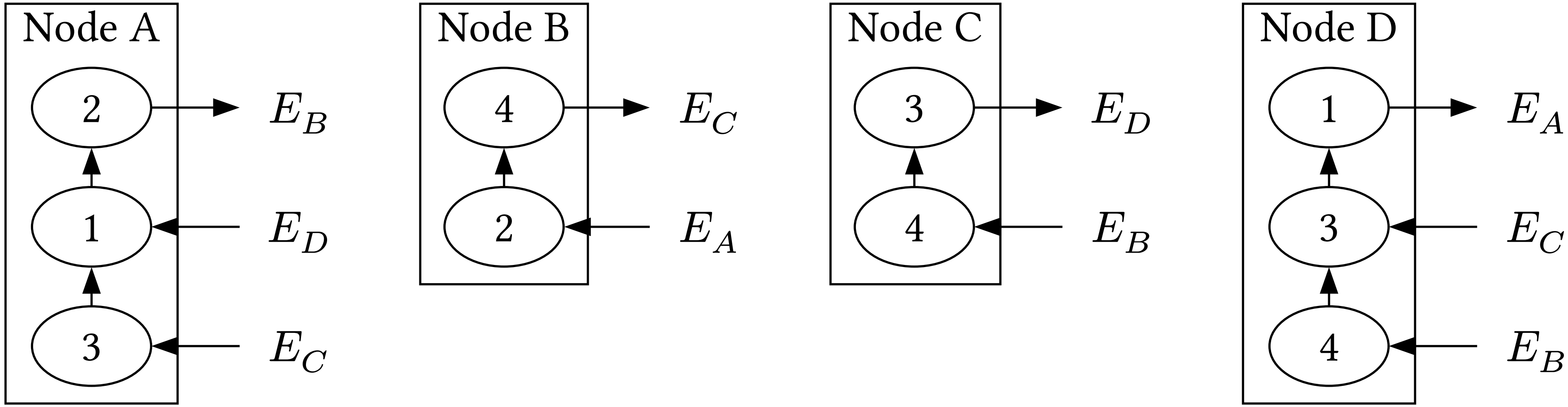


Information is only sent “ahead”. That is, given

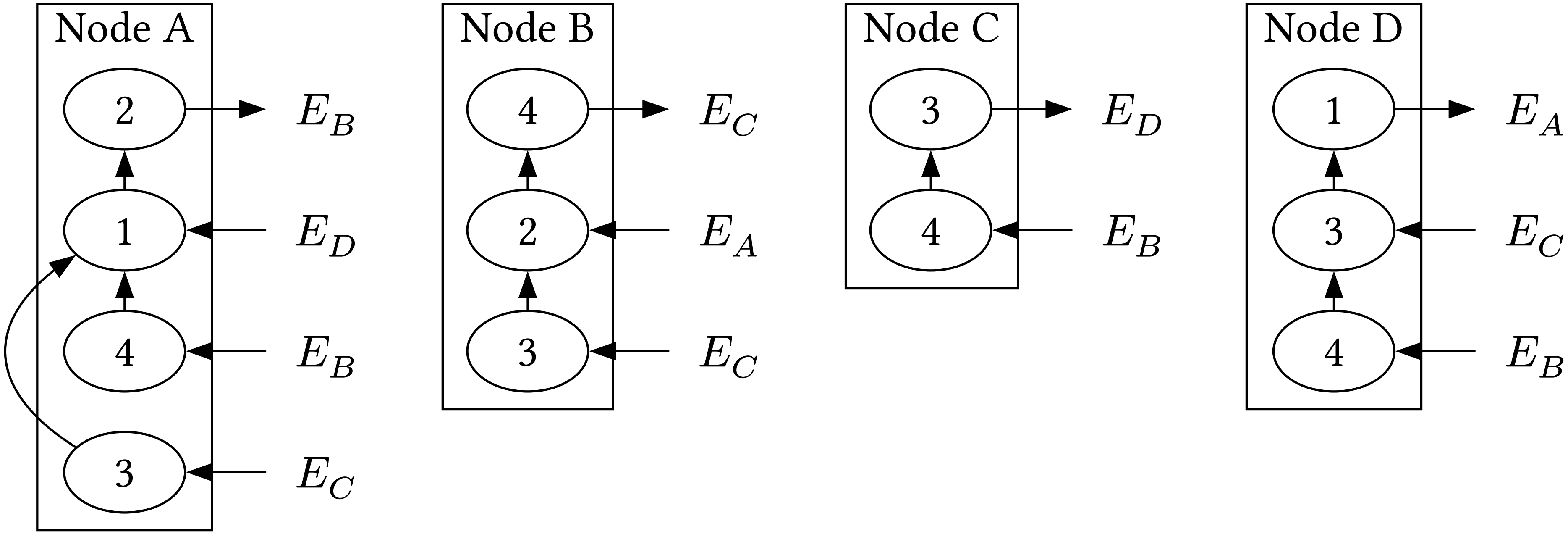
$$E_a \to t_i \to \cdots \to t_j \to E_b$$

The message is sent to E_b if $i > j$ (this is an arbitrary choice).

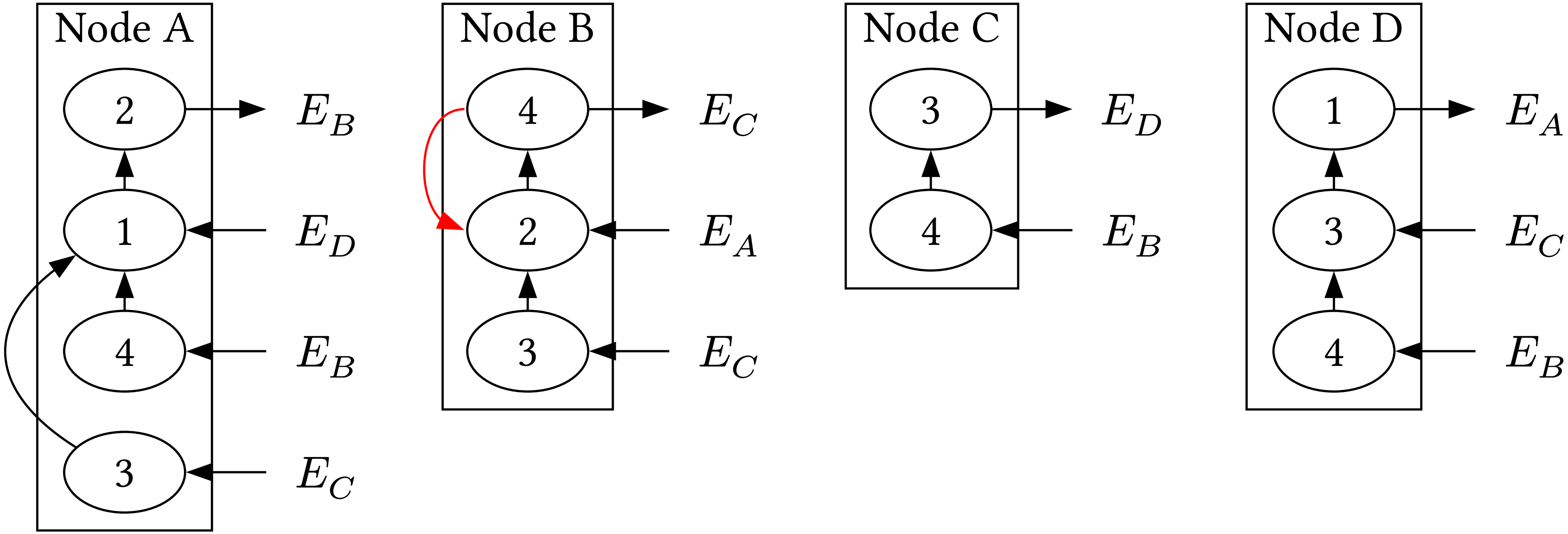
Step 1: C sends to D, and D sends to A.



Step 2: A sends to B, D sends to A.



Step 3: A sends $4 \to 2$ to B: a cycle is found, there is a deadlock.

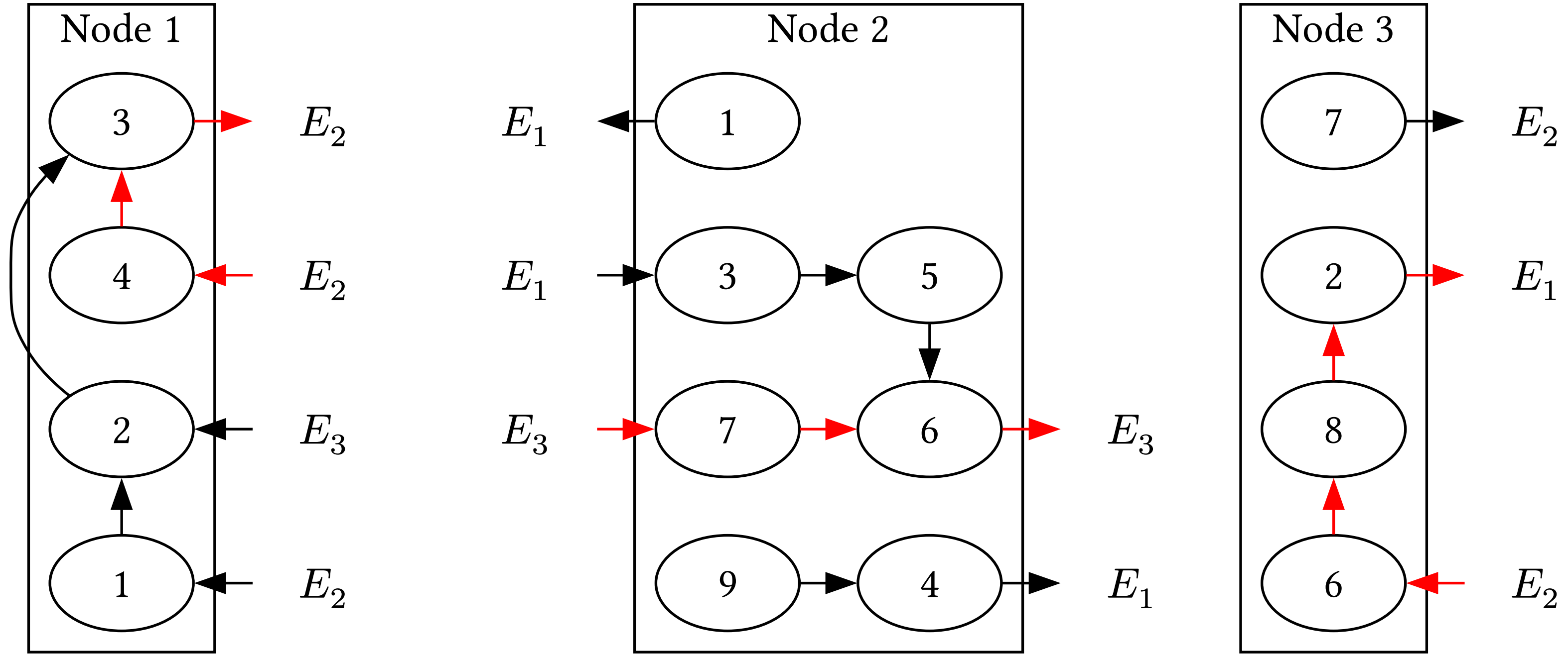


Exercise E.2

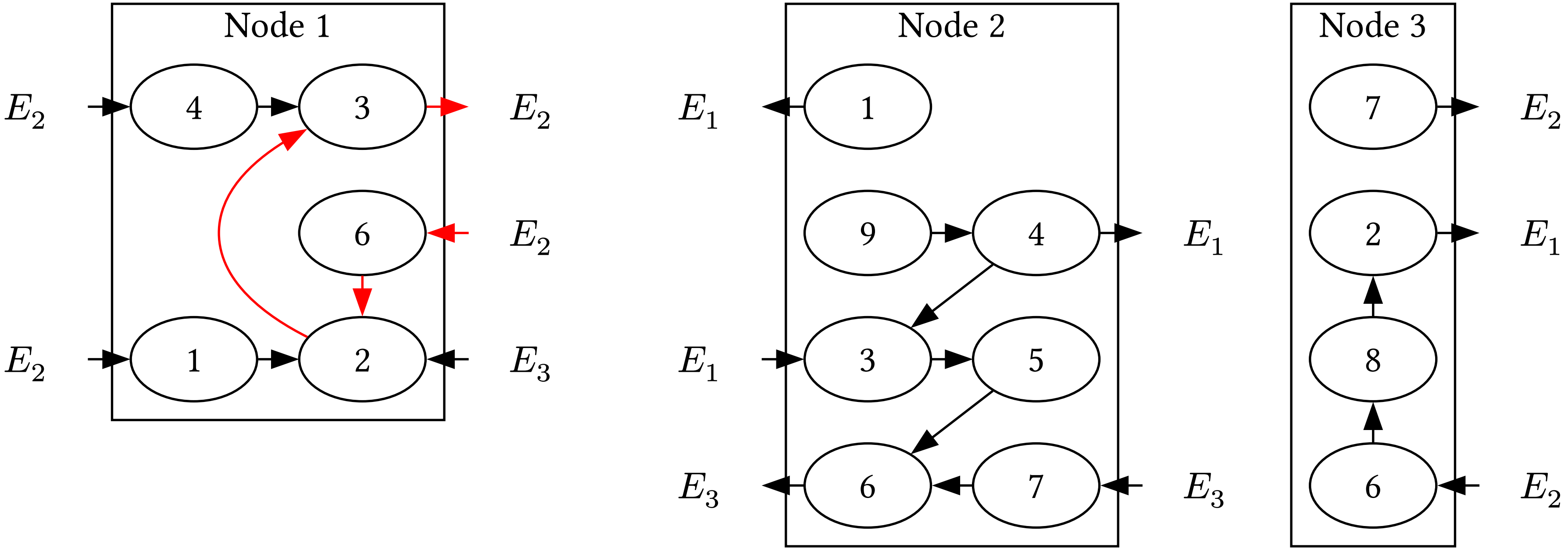
Consider the following waiting conditions:

- Node 1: $E_2 \rightarrow t_1,\ t_1 \rightarrow t_2,\ E_3 \rightarrow t_2,\ t_2 \rightarrow t_3,\ t_3 \rightarrow E_2,\ E_2 \rightarrow t_4,$ $t_4 \rightarrow t_3$
- Node 2: $E_1 \rightarrow t_3, t_3 \rightarrow t_5, t_5 \rightarrow t_6, t_6 \rightarrow E_3, E_3 \rightarrow t_7, t_7 \rightarrow t_6,$ $t_9 \rightarrow t_4, t_4 \rightarrow E_1, t_1 \rightarrow E_1$
- Node 3: $E_2 \rightarrow t_6, t_6 \rightarrow t_8, t_8 \rightarrow t_2, t_2 \rightarrow E_1, t_7 \rightarrow E_2$

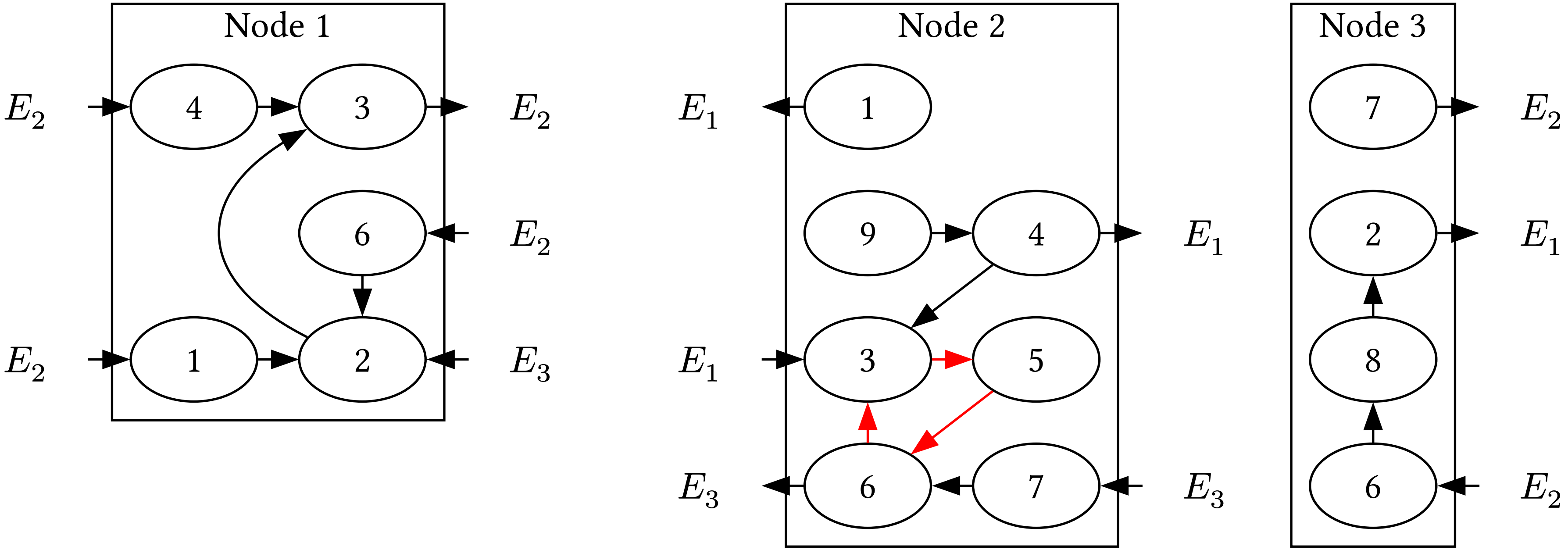
Indicate whether there is a distributed deadlock.



Step 1: node 1 sends $4 \to 3$ to node 2, node 2 sends $7 \to 6$ to node 3, node 3 sends $6 \to 2$ to node 1.



Step 2: node 1 sends $6 \rightarrow 3$ to node 2.



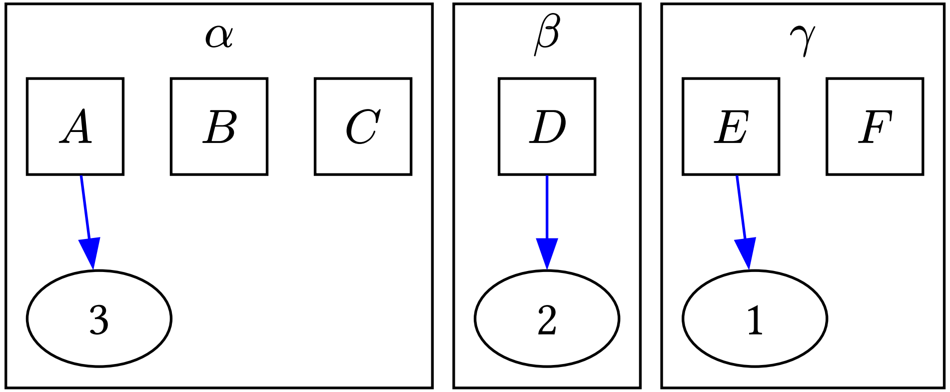
There is a cycle, which means there is a deadlock.

Exercise E.3

Suppose we have 3 nodes α, β, and γ, 6 transactions $t_1 ... t_6$, and 6 resources $A...F$. A, B, and C are on node $\alpha]$, D is on node β, and E and F are on node γ. Consider the following schedule

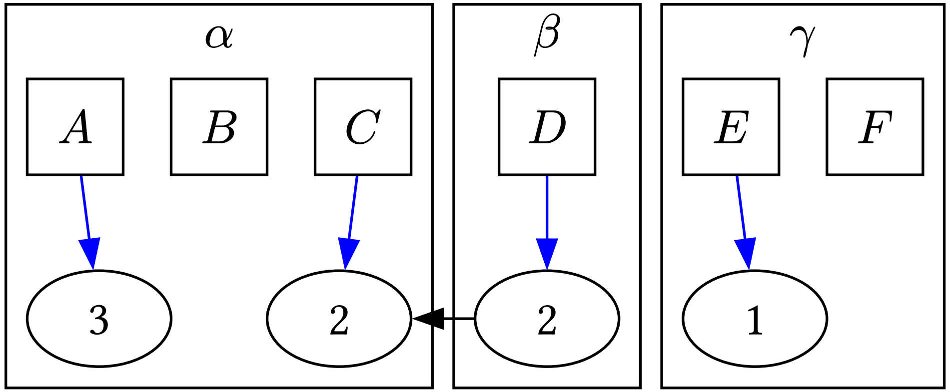
$$r_1(E)r_2(D)r_3(A)r_2(C)w_1(B)r_4(B)w_4(A)r_3(E)$$
$$r_5(D)w_1(C)w_3(F)r_6(D)w_5(E)w_6(D)$$

Assume each transactions begins on the node hosting the first used resource. Build the waiting conditions and simulate the Obermarck algorithm.

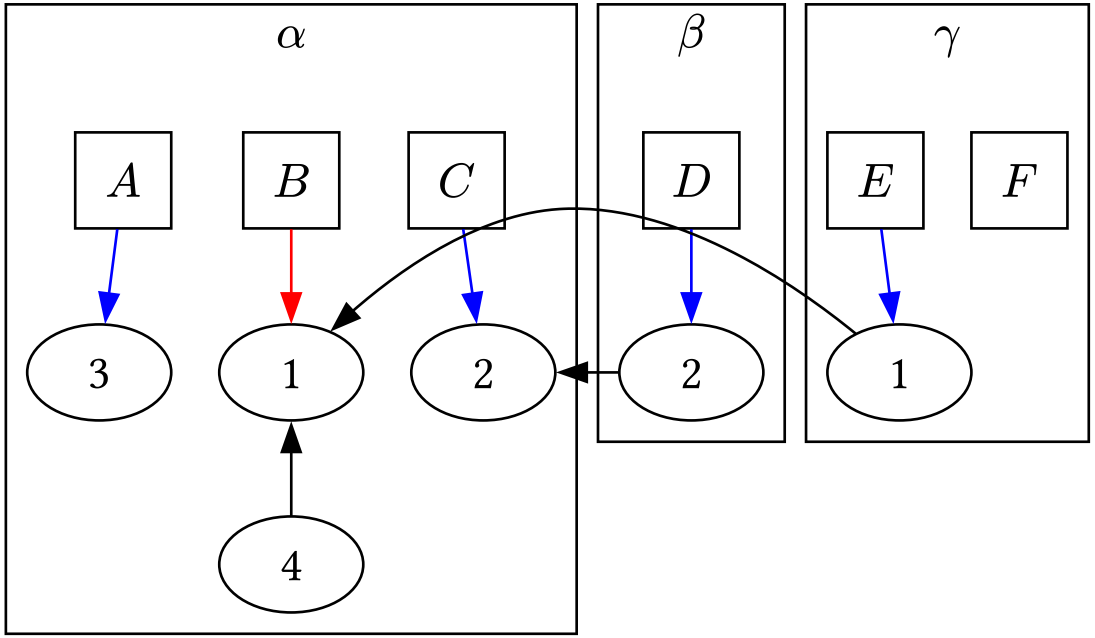


$r_1(E)r_2(D)r_3(A)r_2(C)w_1(B)r_4(B)w_4(A)r_3(E)r_5(D)w_1(C)w_3(F)r_6(D)w_5(E)w_6(D)$

In blue, shared locks. In red, exclusive locks.

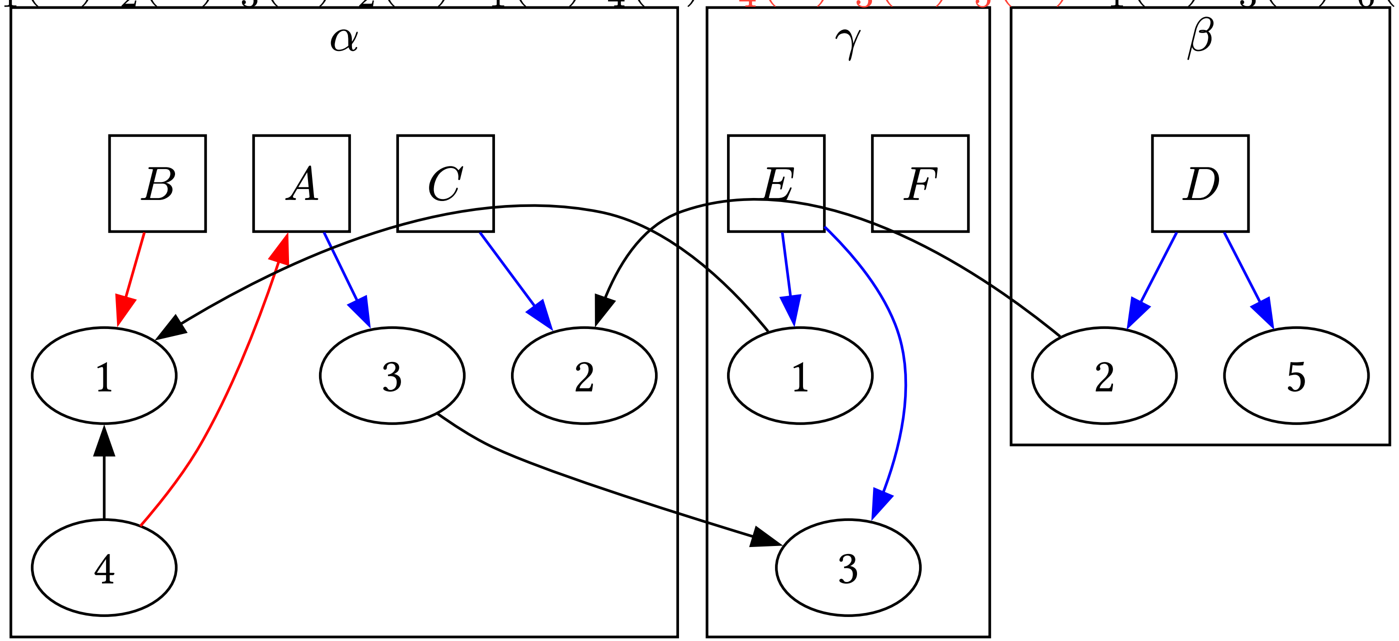


$$r_1(E)r_2(D)r_3(A)r_2(C)w_1(B)r_4(B)w_4(A)r_3(E)r_5(D)w_1(C)w_3(F)r_6(D)w_5(E)w_6(D)$$

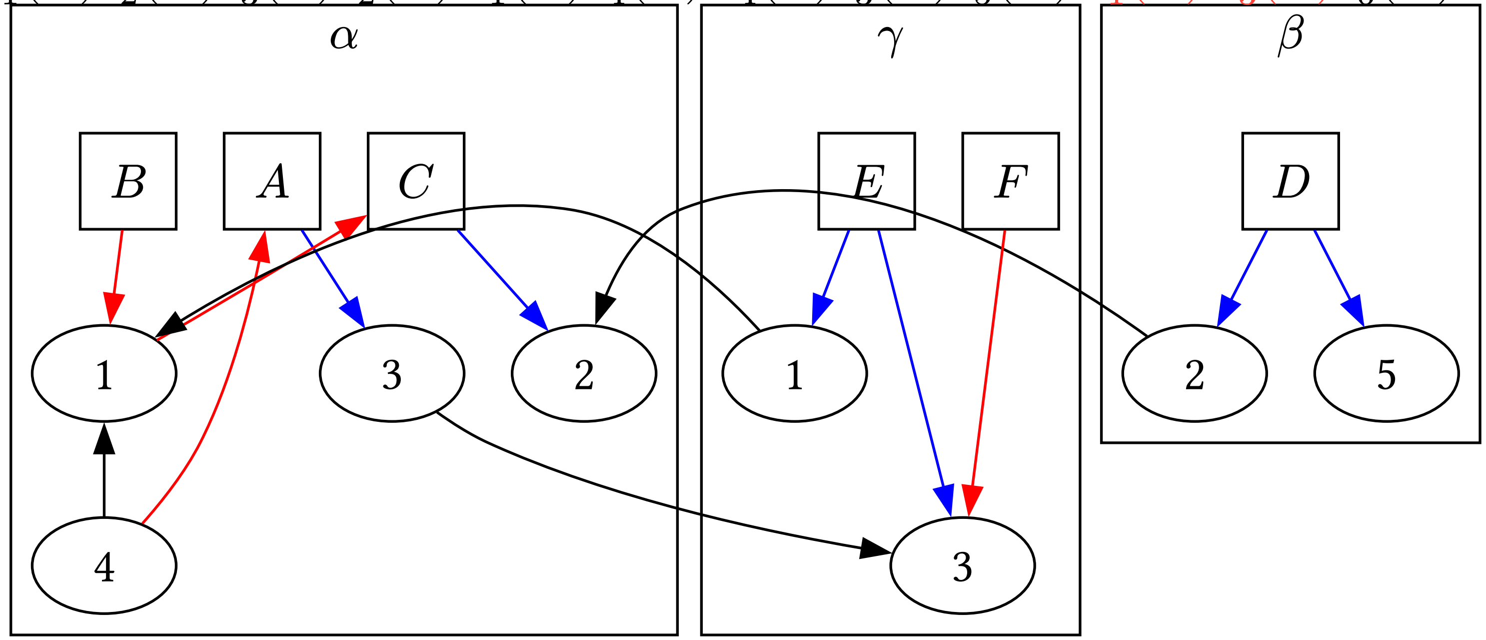


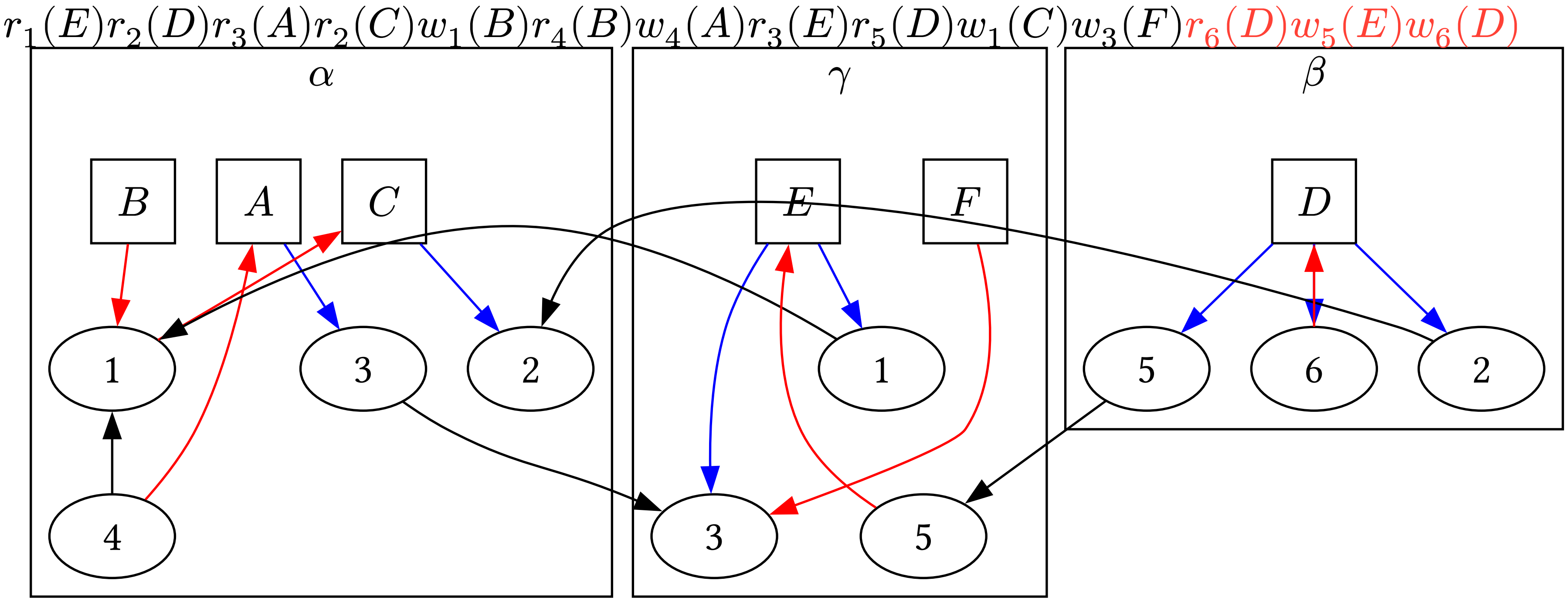
$$r_1(E)r_2(D)r_3(A)r_2(C)w_1(B)r_4(B)w_4(A)r_3(E)r_5(D)w_1(C)w_3(F)r_6(D)w_5(E)w_6(D)$$

$r_1(E)r_2(D)r_3(A)r_2(C)w_1(B)r_4(B)w_4(A)r_3(E)r_5(D)w_1(C)w_3(F)r_6(D)w_5(E)w_6(D)$



$r_1(E)r_2(D)r_3(A)r_2(C)w_1(B)r_4(B)w_4(A)r_3(E)r_5(D)w_1(C)w_3(F)r_6(D)w_5(E)w_6(D)$




$r_1(E)r_2(D)r_3(A)r_2(C)w_1(B)r_4(B)w_4(A)r_3(E)r_5(D)w_1(C)w_3(F)r_6(D)w_5(E)w_6(D)$
α
γ
β
B
A
C
E
F
D
1
3
2
4
1
3
5
5
6
2

Ignoring the situation such as t_6 on β with D (i.e., the same transaction having a shared lock and asking for an exclusive lock), there are no cycle.

Hence, no deadlock.

As a higher-level view, here is the graph without the resources, and only with transactions and dependencies.