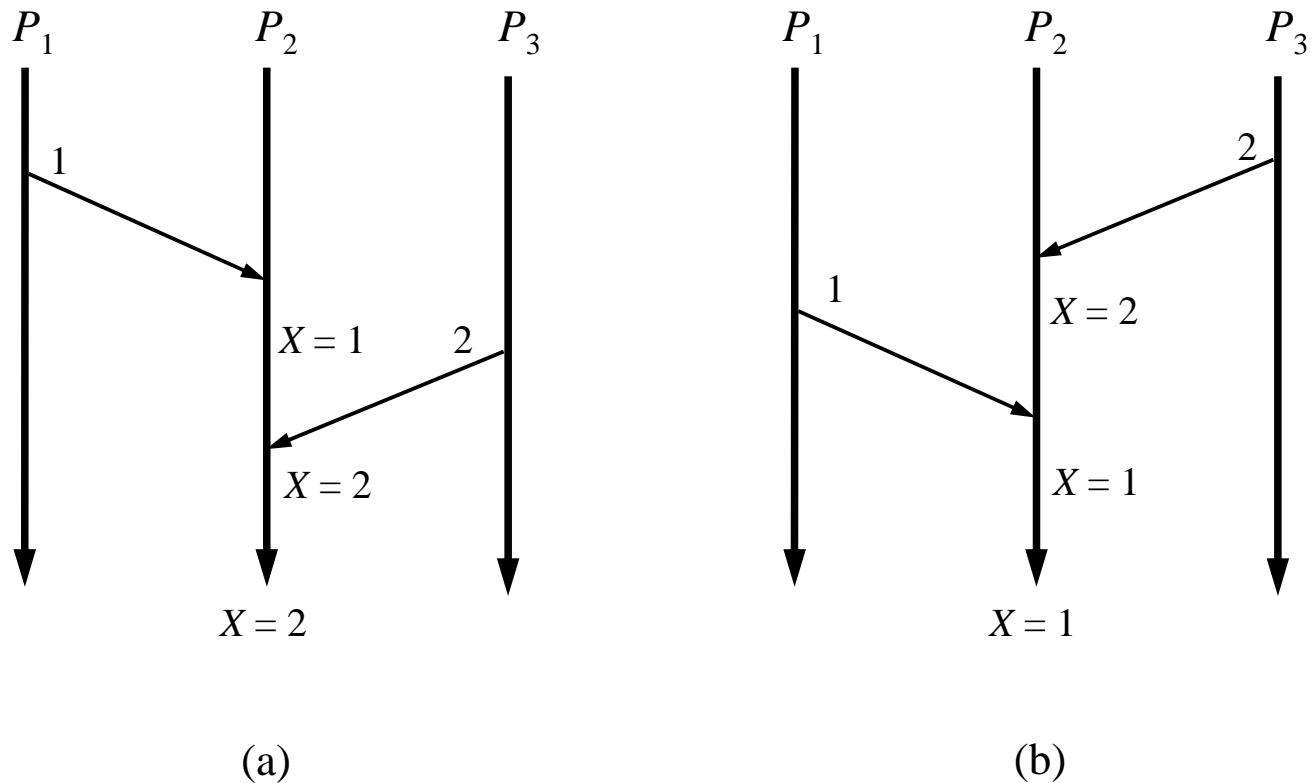


Scalable Race Visualization for Debugging Message-Passing Programs

Park, Mi-Young

Operating System Laboratory
Gyeongsang National University
South Korea

What are Message Races?



Every Races are Detected?

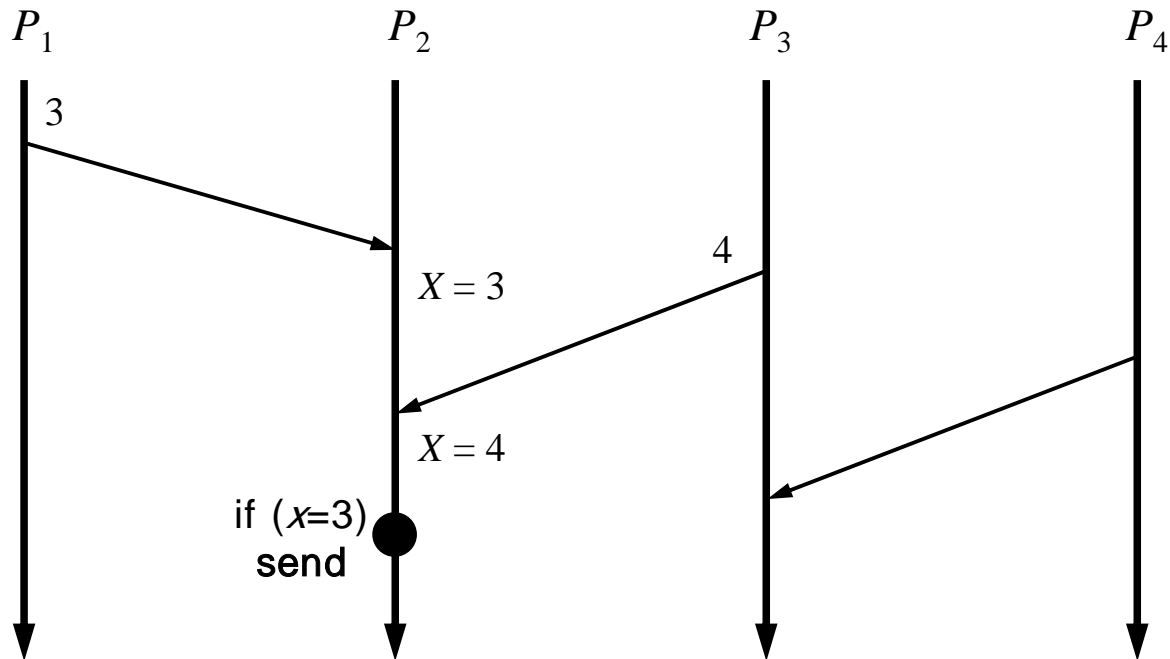
■ Unaffected Race

- Appear in every execution with same input
- Should be detected

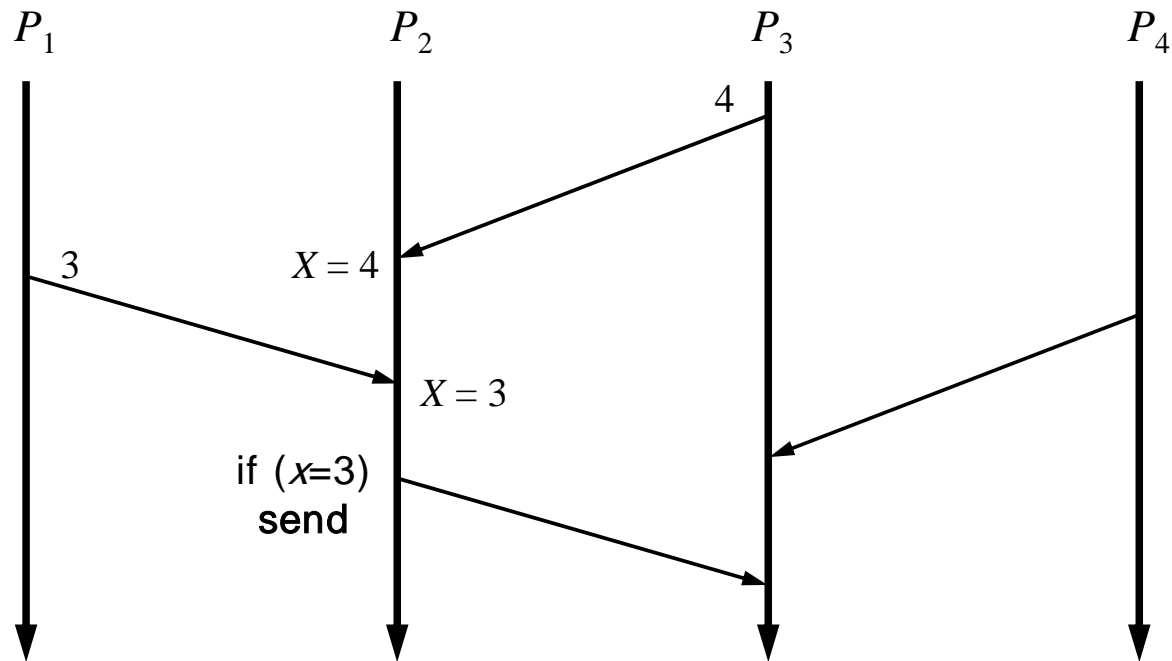
■ Affected Race

- Depend on the appearance of unaffected race
- Need not to be detected

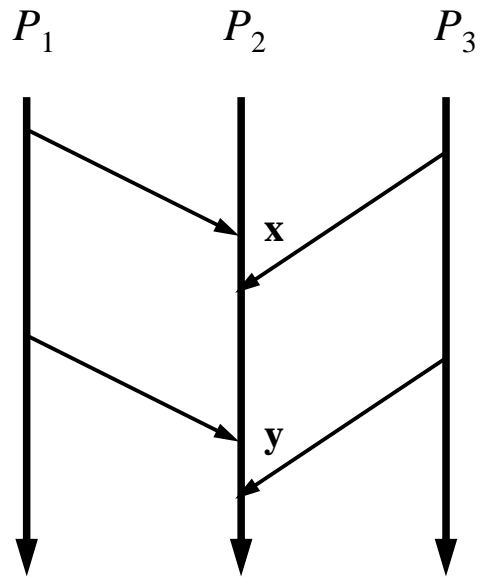
Example: Intended Order



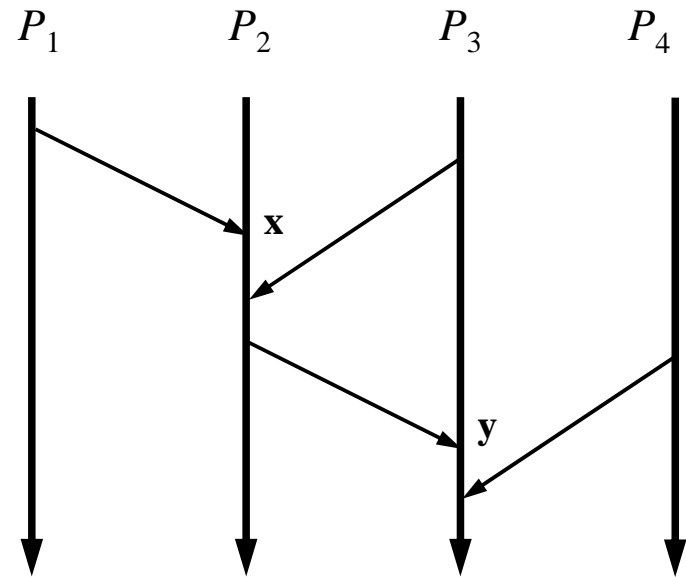
Example: Unintended Order



Affect-Relations

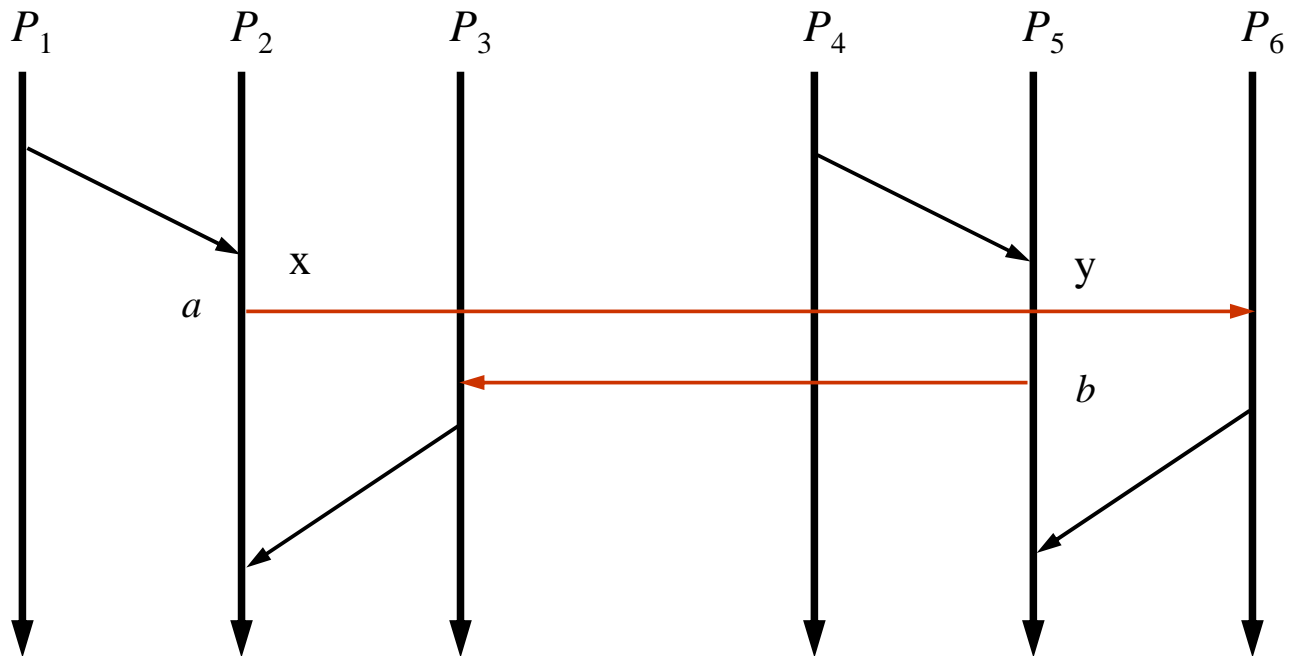


(a)

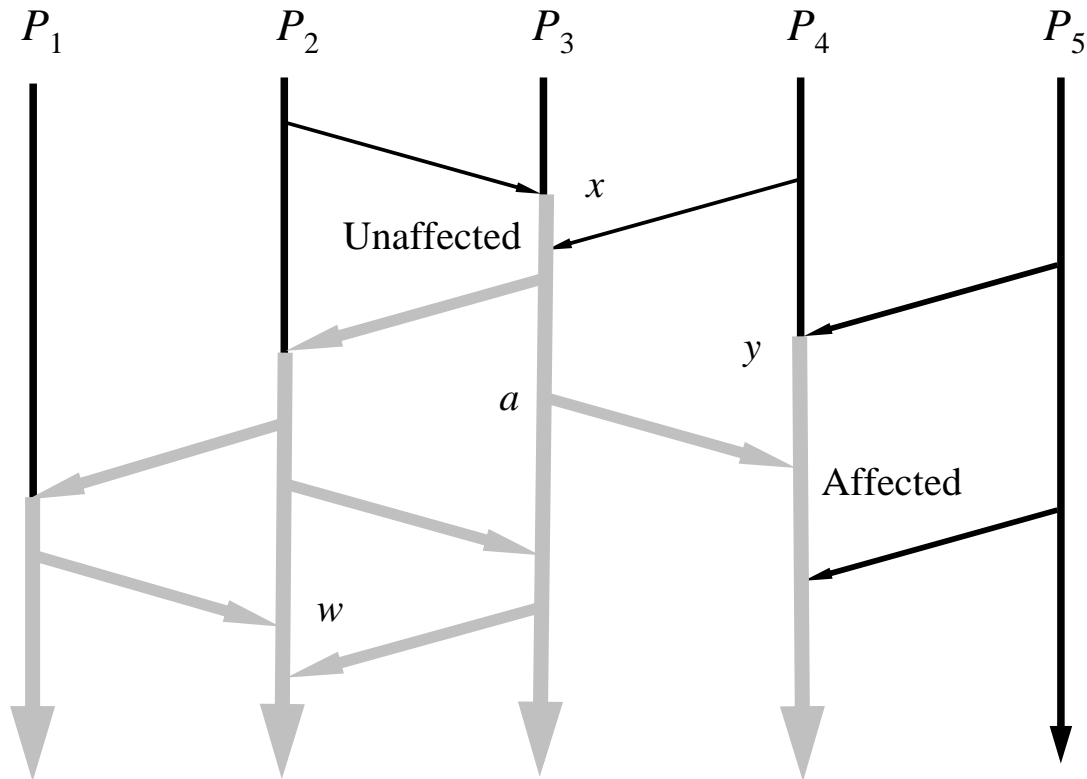


(b)

Tangled Races



Problem: Disappearance of Relation



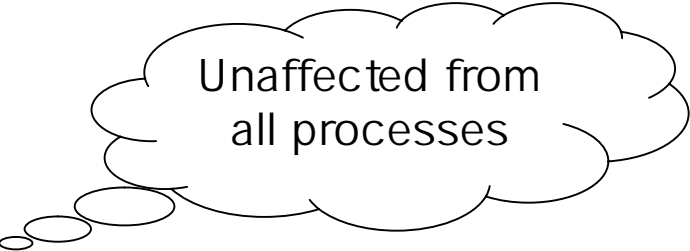
Solution: *Affector* Array

If five processes execute

(a)

P_1	P_2	P_3	P_4	P_5
0	0	0	0	0

[0,0,0,0,0]



Unaffected from
all processes



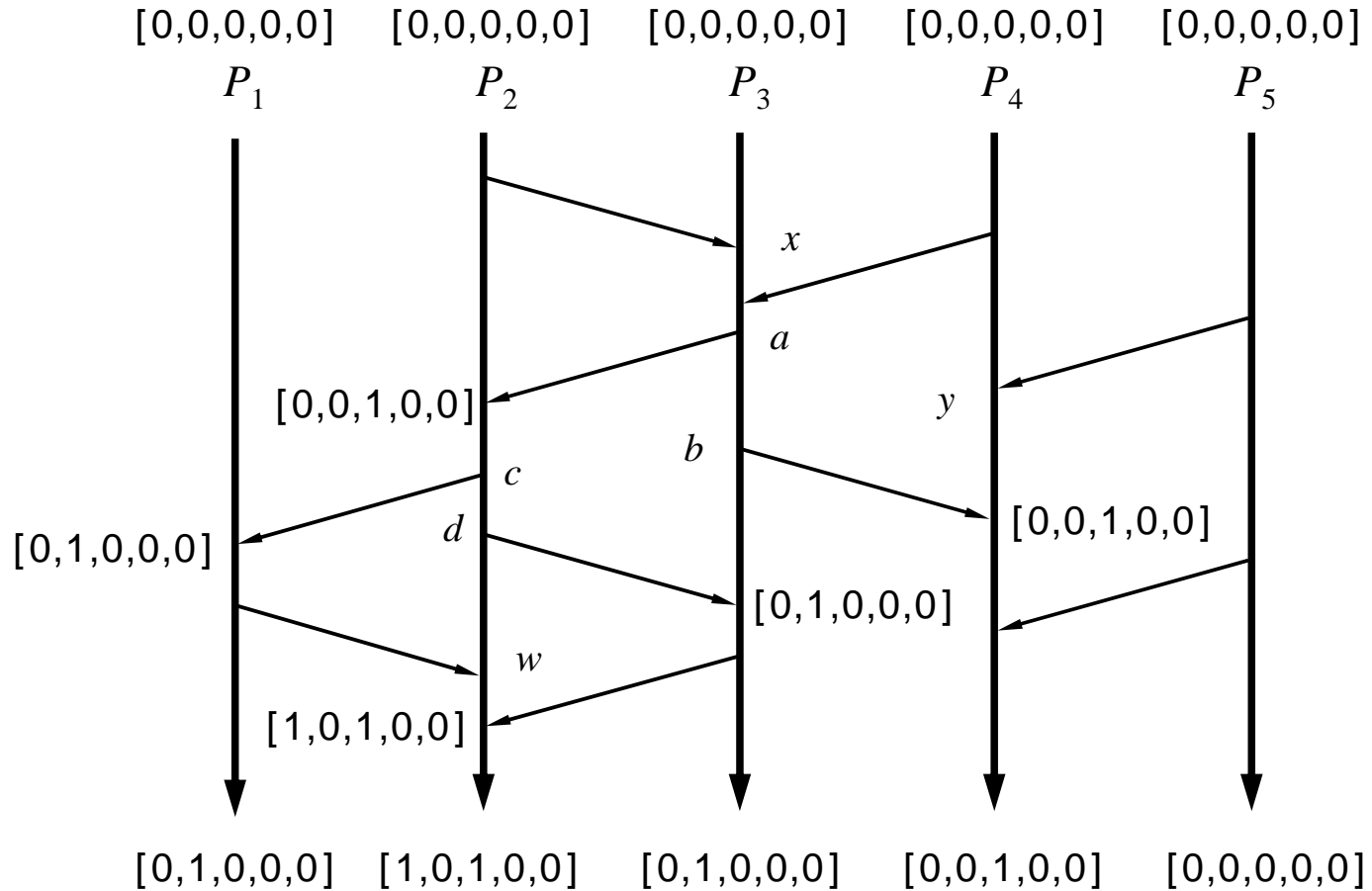
Affected from P_2
and P_4

(b)

P_1	P_2	P_3	P_4	P_5
0	1	0	1	0

[0,1,0,1,0]

Example: *Affector*



Visualizing Races with *Affector*

$P_1 \longrightarrow [0, 1, 0, 0, 0]$

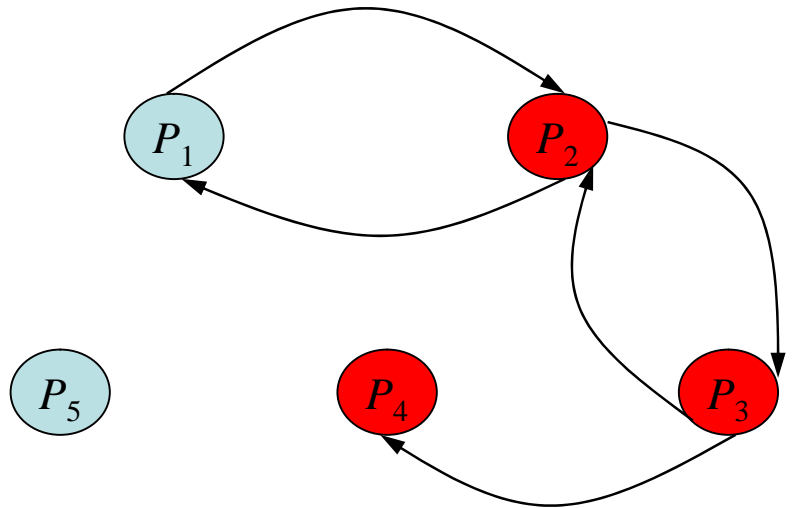
$P_2 \longrightarrow [1, 0, 1, 0, 0]$

$P_3 \longrightarrow [0, 1, 0, 0, 0]$

$P_4 \longrightarrow [0, 0, 1, 0, 0]$

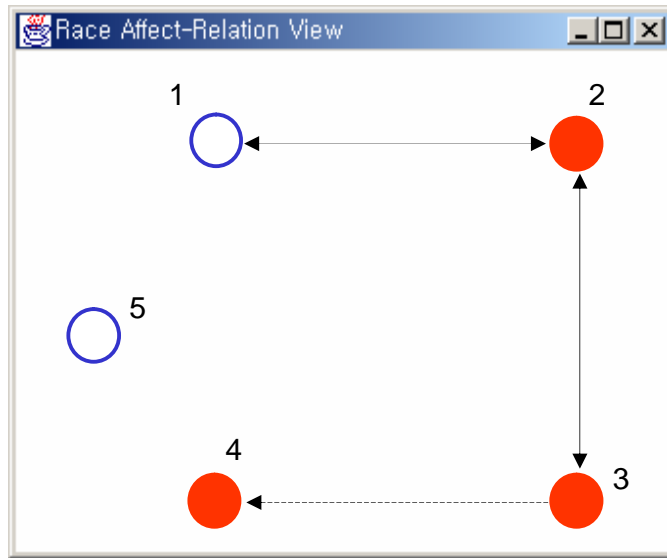
$P_5 \longrightarrow [0, 0, 0, 0, 0]$

(a)

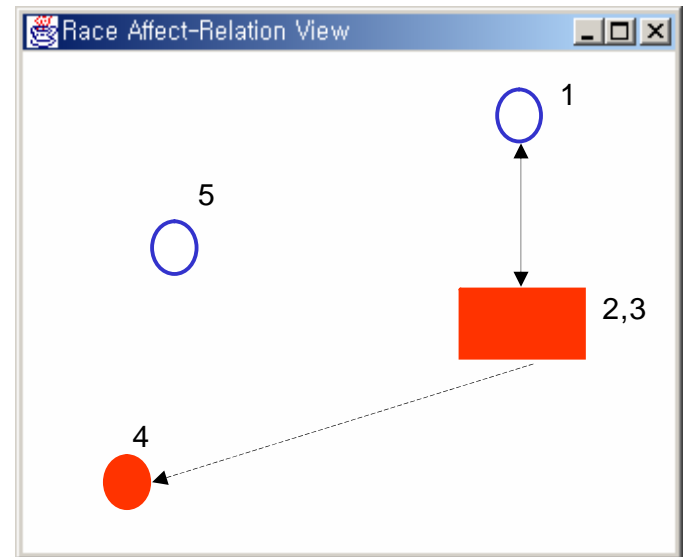


(b)

Implementation: Affect-Relations



(a)



(b)