Tips: Assignment - 1

27 Jan 2022
2-3PM

① Use Socket Programming.

- Fork / child (Parent - process table)
  - ↳ create a process (exec) (Concurrent)

② Deitel & Deitel

- "How to program JAVA"

Peer-to-peer / Multithreading architecture

③ Open MPI

(Message passing Interfaces)

→ Process / task (exec)

(n=10)

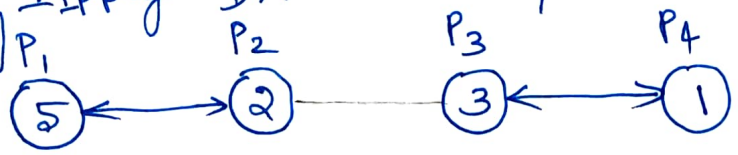
→ Socket
→ bind()
→ listen()
→ accept
→ compute
→ close

Some Constructs.

④ Apply DA in Sequential exec

odd-even

i=1



≤

for (nrounds)

g.I { I/P

P1 - P2

```
send( m , f , t , r , flag);
Receive( m , f , f , r , flag);
compute
Send( )
receive( )
```

Initial — odd-even — round

$P_1(=0)$ $P_2(=0)$ $P_3(=0)$ $P_4(=0)$ ..

5 ⟷ 2 3 ⟷ 1 ... 0

---

send (5, 1, 2, (1)) ⊗

receive (5, 2, 1, (1));

@ P_2 : (2), (5)

find min: $2 \leq 5$? ↙ update

min = 2; $Val(P_2) = \boxed{5}$ (max)

$Count_2$++;

send (min, 2, 1, (1));

receive (min, 1, 2, (1));

($P_1 - P_2$)

@ P_1 : $Val(P_1) = 5$; received = min;

$Val(P_1)$ = received;

$Count_1$++;

1

($P_2 - P_3$)

send (3, 3, 4, (1))

receive (3, 4, 3, (1))

@ P_4: $Val(P_4) = 1$, received = 3.

find min (1, 3) ⇒

min = 1, max = 3.

$Count_4 = Count_4 + 1$;

$Val(P_4)$ = max;

send (min, 4, 3, (1));

receive (min, 3, 4, (1))

@ P_3: $Val(P_3)$ = received;

$Count_3$++; if (count == N) Stop

n is known

n is odd

(1) $\lfloor n/2 \rfloor$ pairs

n is even

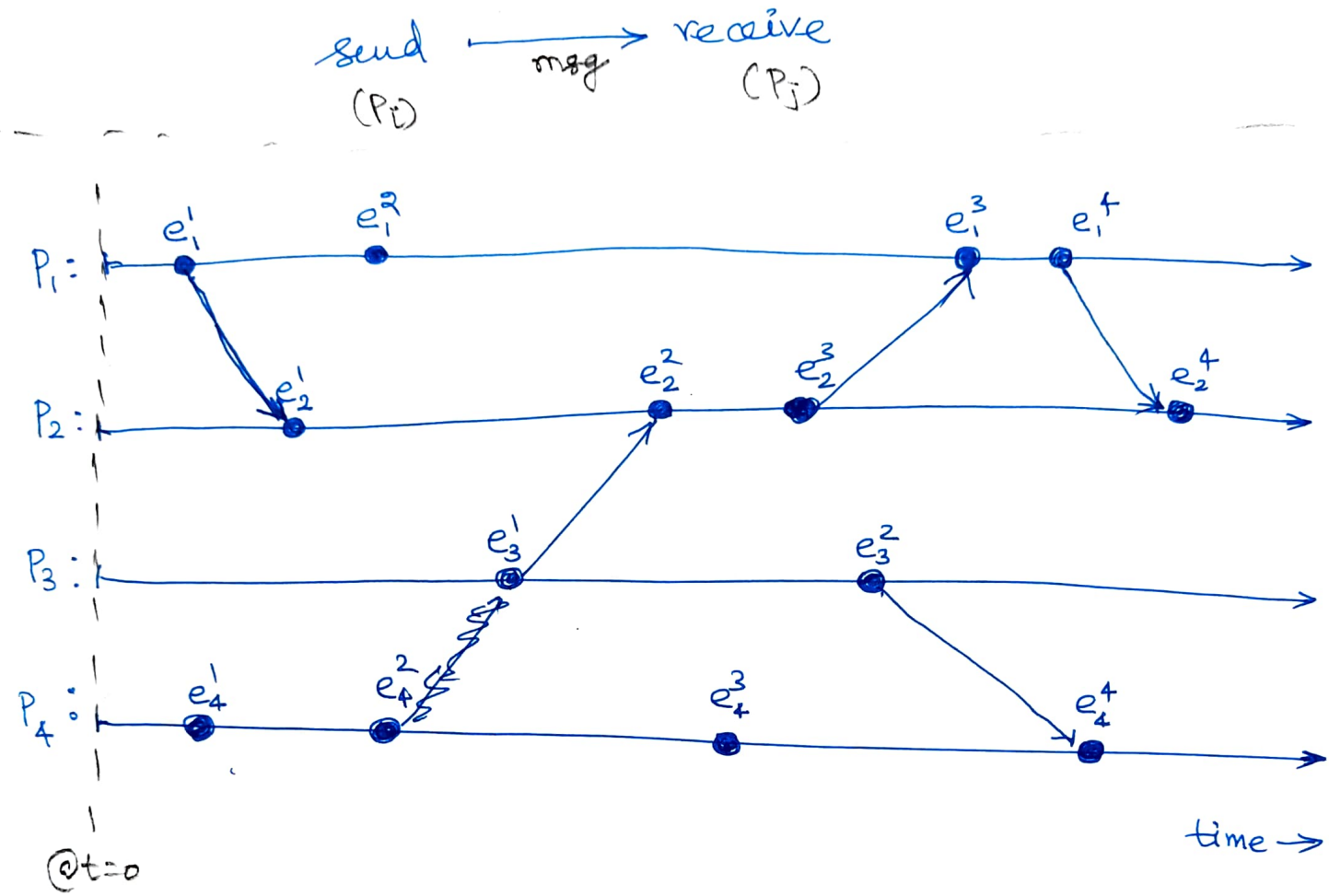
$\lfloor n/3 \rfloor$ pairs.

Print val (⟶)

2

Order of Events:

$$H = \bigcup_i h_i$$



(5) send | (5) receive | all other are local events.

$$e_i^x \longrightarrow e_j^y$$

iff

$$e_i^x \longrightarrow_i e_j^y \quad (i=j) \wedge (x<y)$$

(Same P_i)

Causal Precedence.

e_i^x "happened before" e_j^y

(or)

$$e_i^x \xrightarrow{msg} e_j^y \rightarrow (i \neq j)$$

across P_i's

(or)

$$\exists\, e_k^z \in H : e_i^x \longrightarrow e_k^z \wedge e_k^z \rightarrow e_j^y$$

either on P_i or across P_i's.