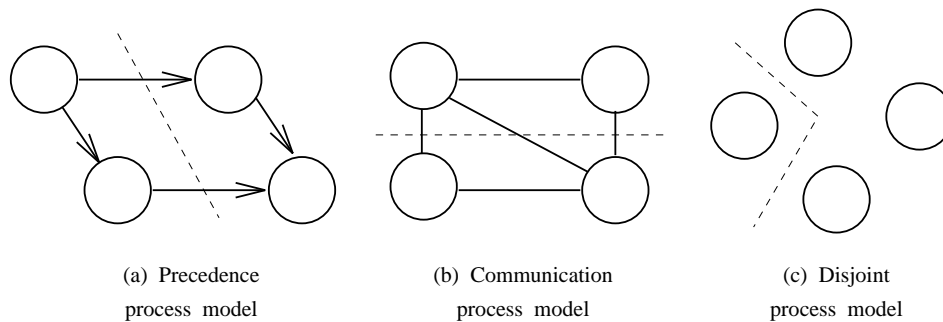


CHAPTER 5: DISTRIBUTED PROCESS SCHEDULING

Chapter outline

- Three process models: precedence, communication, and disjoint
- A system performance model that illustrates the relationship among algorithm, scheduling, and architecture
- Static scheduling: precedence and communication models
- Dynamic scheduling: load sharing and balancing for disjoint and interacting process models
- Implementation: remote service and execution, and process migration
- Real-time scheduling and synchronization

Process models



A system performance model

Speed-up factor

$$S = F(\text{Algorithm}, \text{System}, \text{Schedule})$$

$$S = \frac{OSPT}{CPT} = \frac{OSPT}{OCPT_{ideal}} \times \frac{OCPT_{ideal}}{CPT} = S_i \times S_d$$

Ideal speed-up

$$S_i = \frac{RC}{RP} \times n$$

$$RP = \frac{\sum_{i=1}^m P_i}{OSPT}$$

$$RC = \frac{\sum_{i=1}^m P_i}{OCPT_{ideal} \times n}$$

System degradation

$$S_d = \frac{1}{1 + \rho}$$

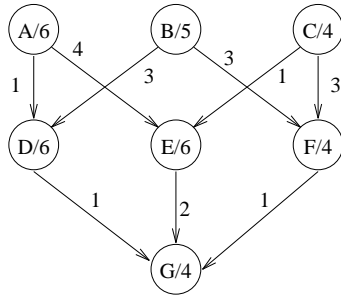
$$\rho = \frac{CPT - OCPT_{ideal}}{OCPT_{ideal}}$$

Finally

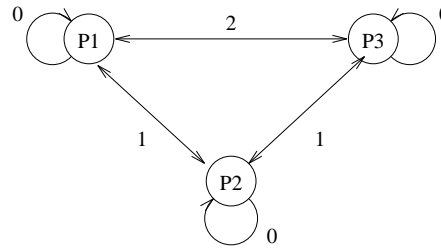
$$S = \frac{RC}{RP} \times \frac{1}{1 + \rho} \times n$$

Static scheduling

Precedence process model



(a) Precedence process model



(b) Communication system model

(a) LS

P1	A/6	D/6	G/4	
P2	B/5	F/4	7	
P3	C/4	2	E/6	4

Makespan = 16

(b) ELS

P1	A/6	2	D/6	10	G/4
P2	B/5	2	F/4	17	
P3	C/4	10	E/6	8	

Makespan = 28

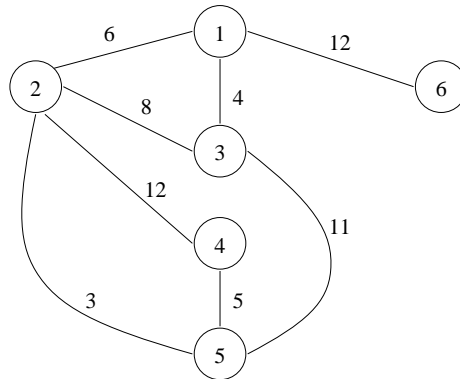
(c) ETF

P1	A/6	E/6	6		
P2	B/5	2	D/6	1	G/4
P3	C/4	4	F/4	6	

Makespan = 18

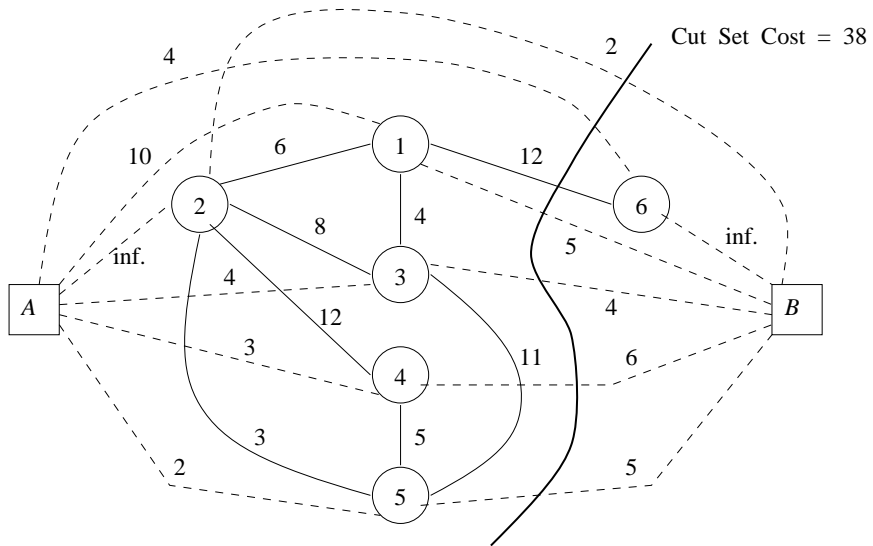
Communication process model

Process	Cost on <i>A</i>	Cost on <i>B</i>
1	5	10
2	2	infinity
3	4	4
4	6	3
5	5	2
6	infinity	4



(a) Computation cost

(b) Communication cost

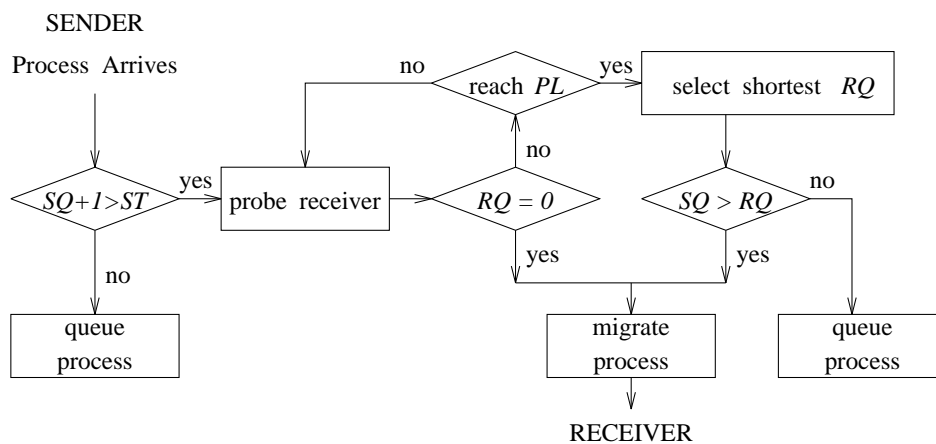


Dynamic scheduling

Load sharing and balancing

Sender initiated algorithms

- **transfer policy:** When does a node become the sender?
- **selection policy:** How does the sender choose a process for transfer?
- **location policy:** Which node should be the target receiver?



Receiver initiated algorithms

Similar policies but require preemption

Hybrid algorithms and their performance

Remote process implementation

Remote service

- As remote procedure calls at the language level
- As remote commands at the operating system level
- As interpretive messages at the application level

Remote execution

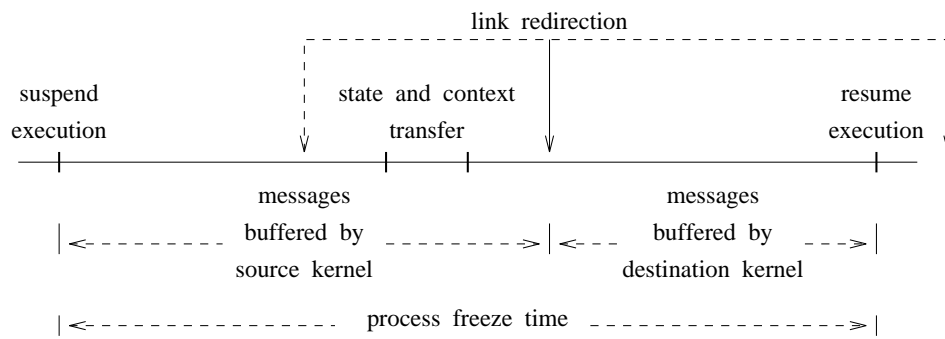
The remote operation initiated by a client is created by the client for resource or load sharing (processor-pool model).

- Load sharing algorithm
- Location independence
- System heterogeneity
- Protection and security

Process migration

Preemption and reconfiguration

link redirection and message forwarding



state and context transfer

freeze time and residual computation dependency

Real-time scheduling

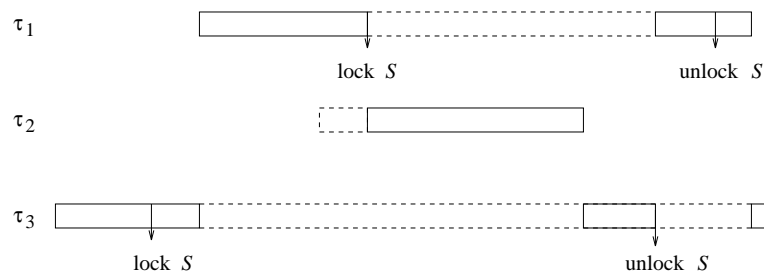
Soft/hard deadlines, periodic/apperiodic, priority scheduling

Priority scheduling

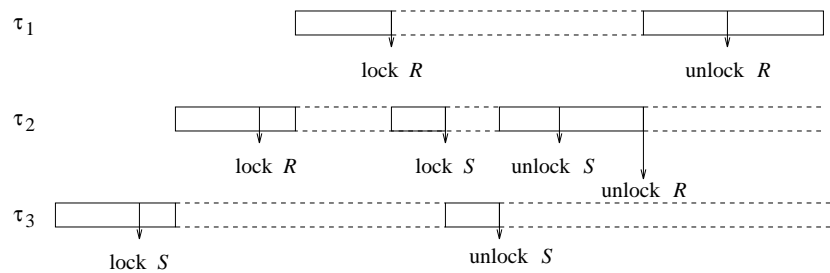
- Rate monotonic priority assignment: task period
- Deadline monotonic priority assignment: deadline
- Earliest deadline first: dynamic deadline

Real-time synchronization

Priority inversion



Chain blocking



Protocols to reduce priority inversion and blocking: PIP and PCP