

18.7

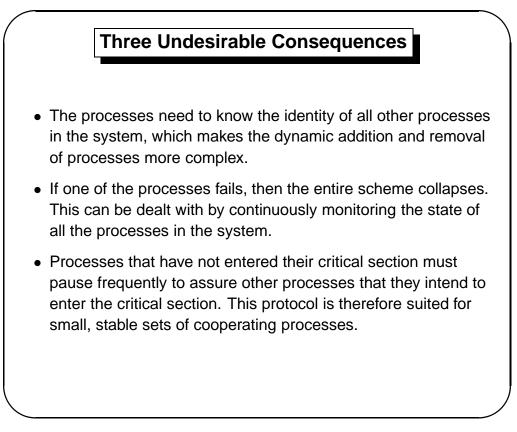
Silberschatz and Galvin © 1998

Desirable Behavior of Fully Distributed Approach

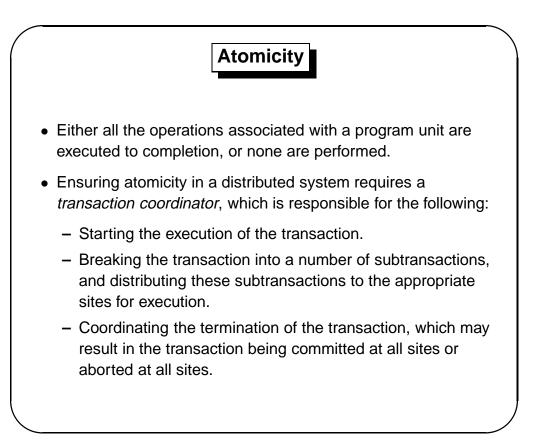
- Freedom from deadlock is ensured.
- Freedom from starvation is ensured, since entry to the critical section is scheduled according to the timestamp ordering. The timestamp ordering ensures that processes are served in a first-come, first-served order.
- The number of messages per critical-section entry is

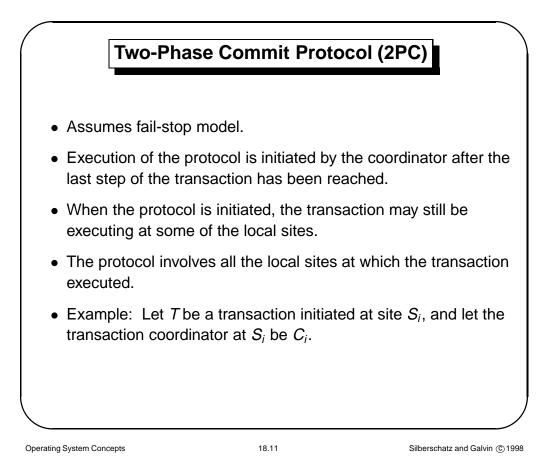
 $2 \times (n-1).$

This is the minimum number of required messages per critical-section entry when processes act independently and concurrently.



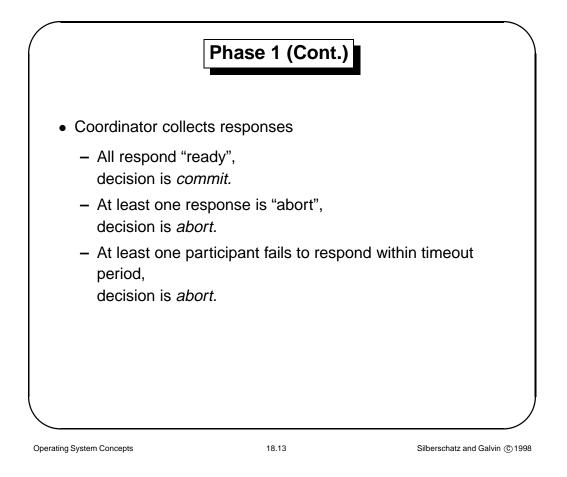
18.9

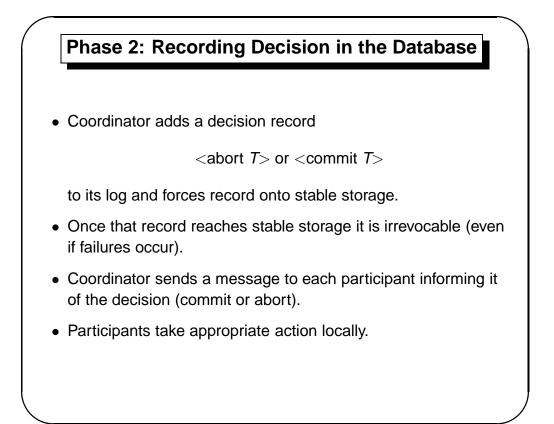


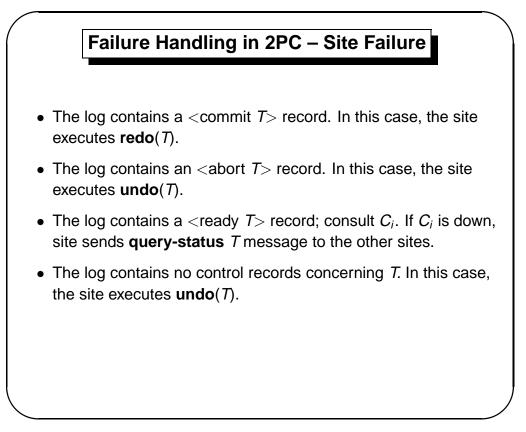




- C_i adds <prepare T> record to the log.
- C_i sends <prepare T> message to all sites.
- When a site receives a <prepare *T*> message, the transaction manager determines if it can commit the transaction.
 - If no: add \langle no $T \rangle$ record to the log and respond to C_i with \langle abort $T \rangle$.
 - If yes:
 - * add < ready *T*> record to the log.
 - * force all log records for T onto stable storage.
 - * transaction manager sends < ready T> message to C_i .





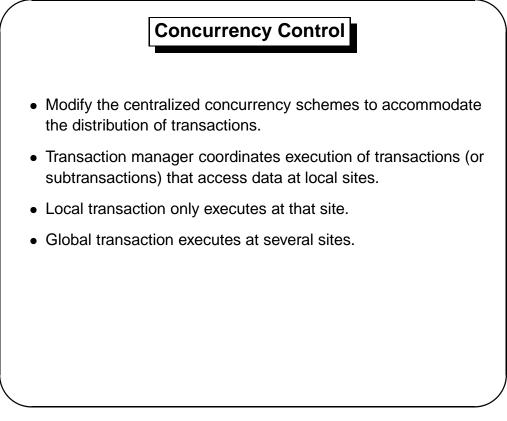


18.15

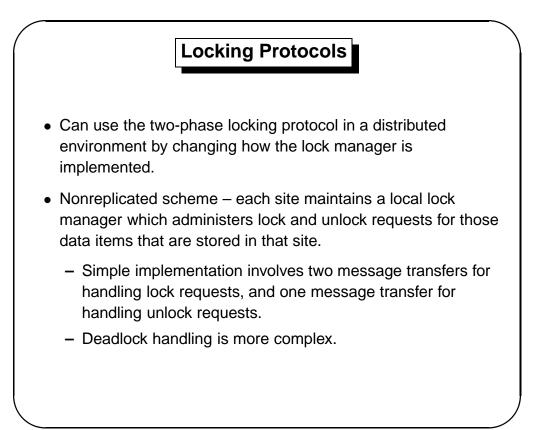
Silberschatz and Galvin © 1998

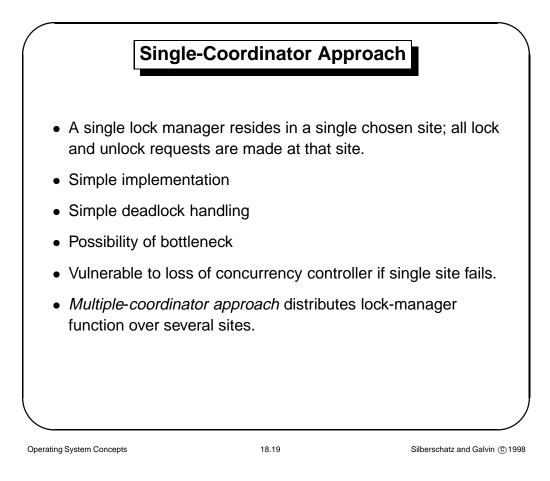
Failure Handling in 2PC – Coordinator C_i Failure

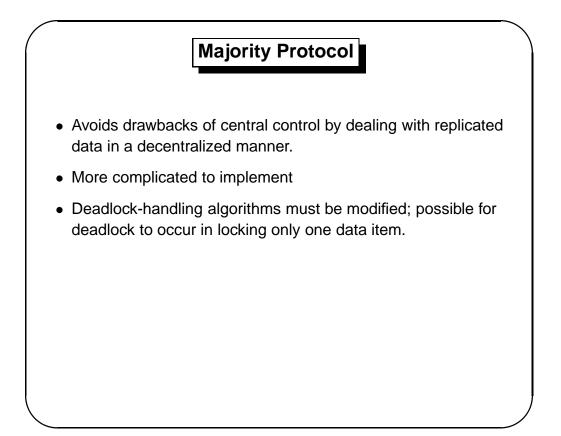
- If an active site contains a <commit *T*> record in its log, then *T* must be committed.
- If an active site contains an <abort T> record in its log, then T must be aborted.
- If some active site does *not* contain the record < ready *T*> in its log, then the failed coordinator *C_i* cannot have decided to commit *T*. Rather than wait for *C_i* to recover, it is preferable to abort *T*.
- All active sites have a <ready T> record in their logs, but no additional control records. In this case we must wait for the coordinator to recover.
 - Blocking problem T is blocked pending the recovery of site S_i .

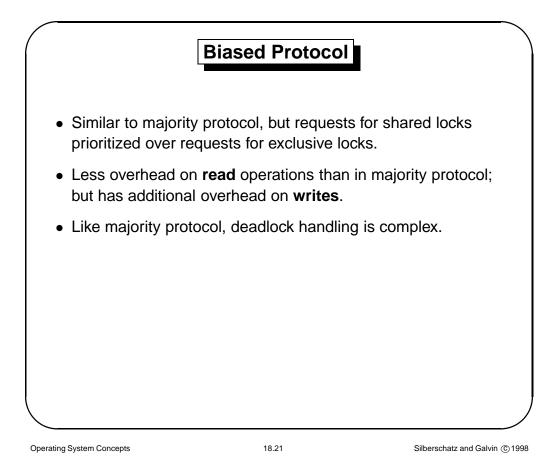


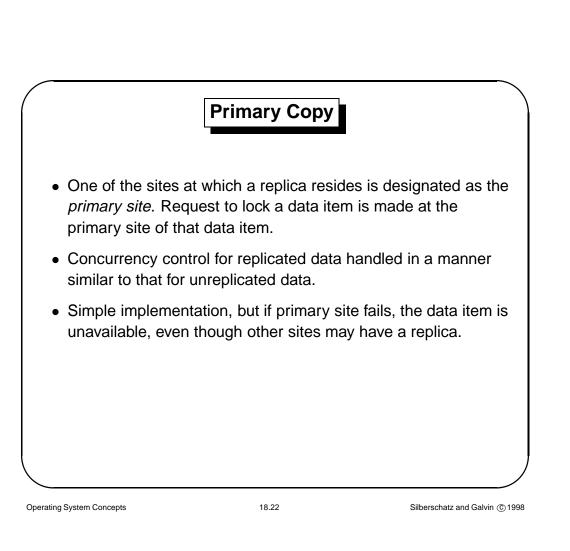
18.17

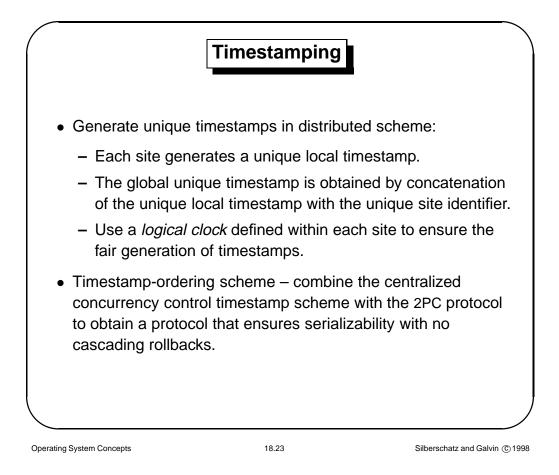


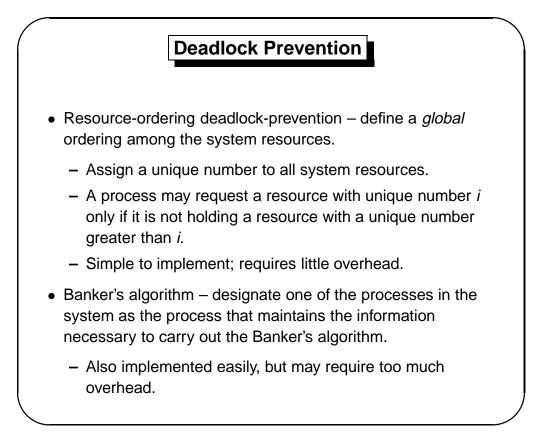


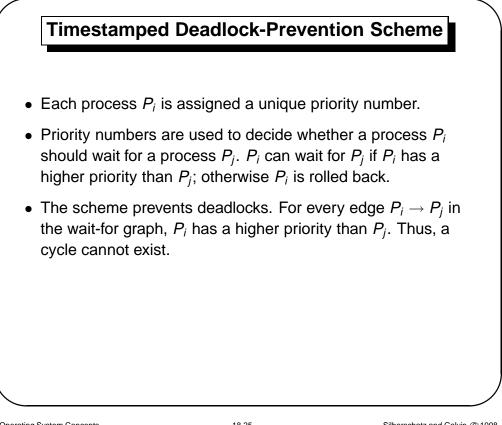




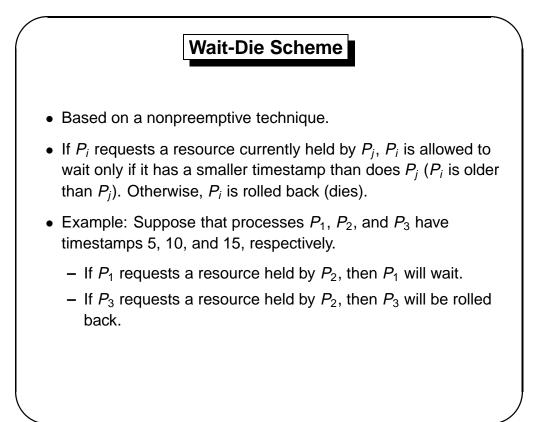


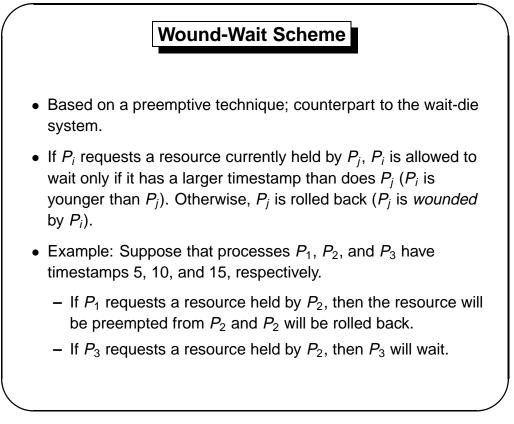






18.25



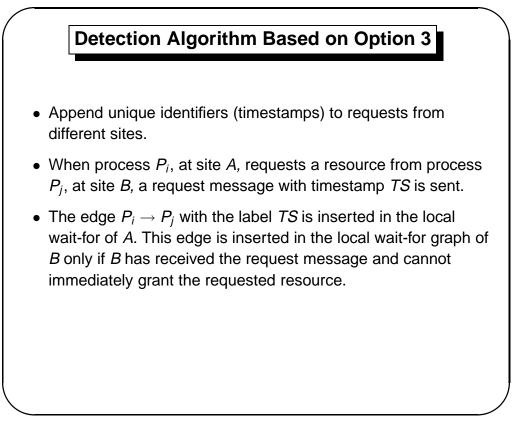


18.27

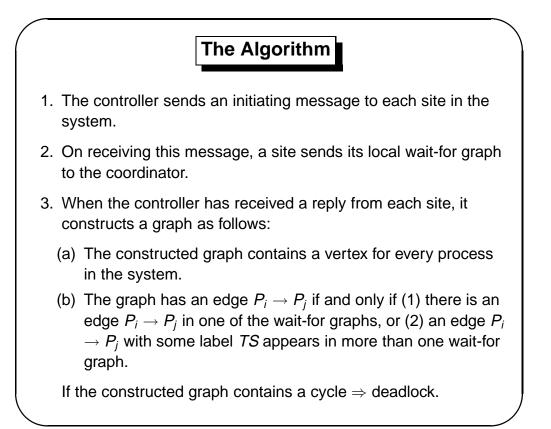
Silberschatz and Galvin © 1998

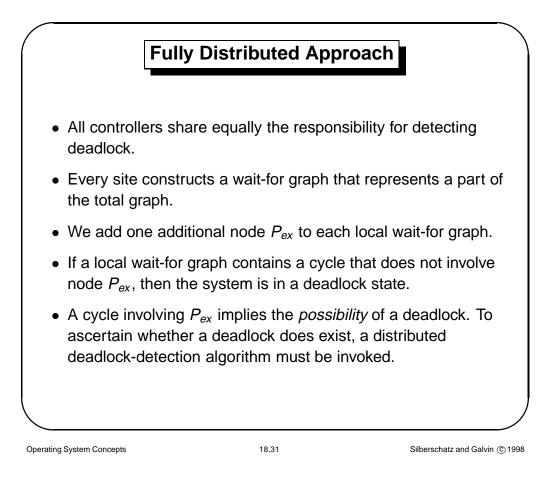
Deadlock Detection – Centralized Approach Each site keeps a *local* wait-for graph. The nodes of the graph correspond to all the processes that are currently either

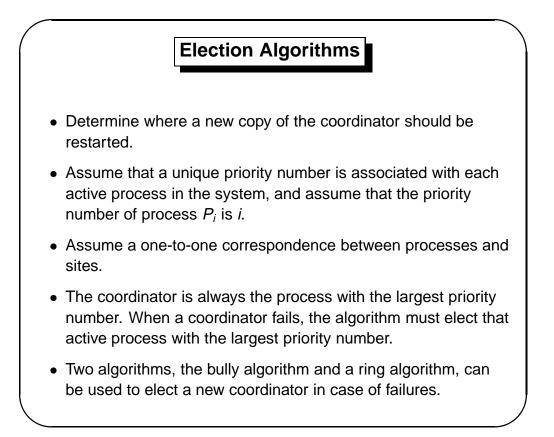
- holding or requesting any of the resources local to that site.
- A global wait-for graph is maintained in a *single* coordination process; this graph is the union of all local wait-for graphs.
- There are three different options (points in time) when the wait-for graph may be constructed:
 - 1. Whenever a new edge is inserted or removed in one of the local wait-for graphs.
 - 2. Periodically, when a number of changes have occurred in a wait-for graph.
 - 3. Whenever the coordinator needs to invoke the cycle-detection algorithm.
- Unnecessary rollbacks may occur as a result of *false cycles*.

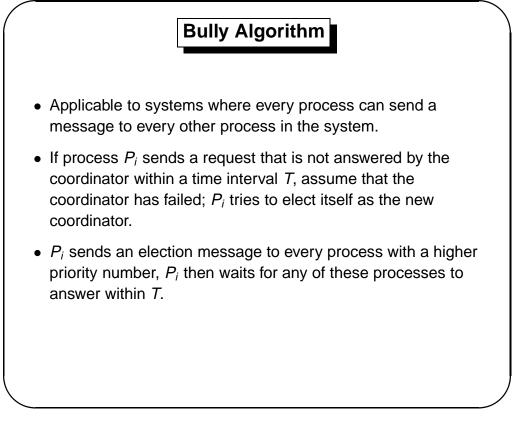


18.29

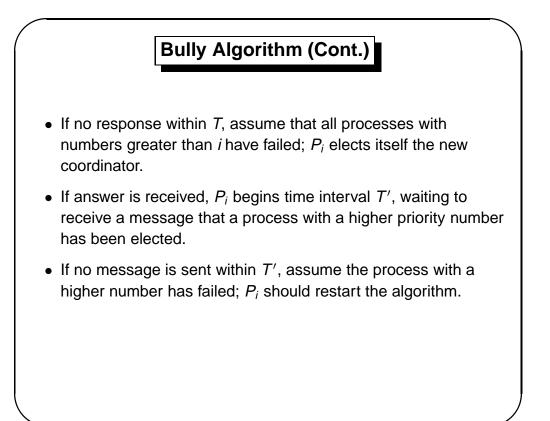


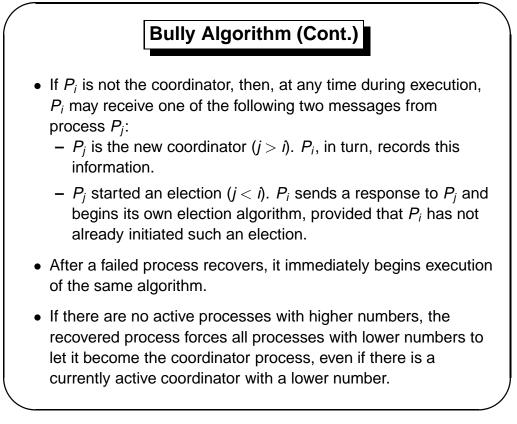






18.33





18.35

