Deadlock

Deadlock

- A system consists of finite number of resources.
- Multiple concurrent processes have to compete to use a resource.
- The sequence of events to use a resource:
  1. Request
  2. Allocate
  3. Release

Cont…

- Request
  - Number of units requested may not exceed the total no. of available units of the resource.
- Allocate
  - Allocate the resource as soon as possible.
  - Maintain a table of resources allocated or available.

Cont…

- Release
  - Release the resources after the process has finished using the allocated resource.

- Request and release are system calls, initiated by a process.

os

- System can control only allocate operation.

Cont…

- Deadlock is the state of permanent blocking of a set of processes each of which is waiting for an event that only another process in the set can cause.

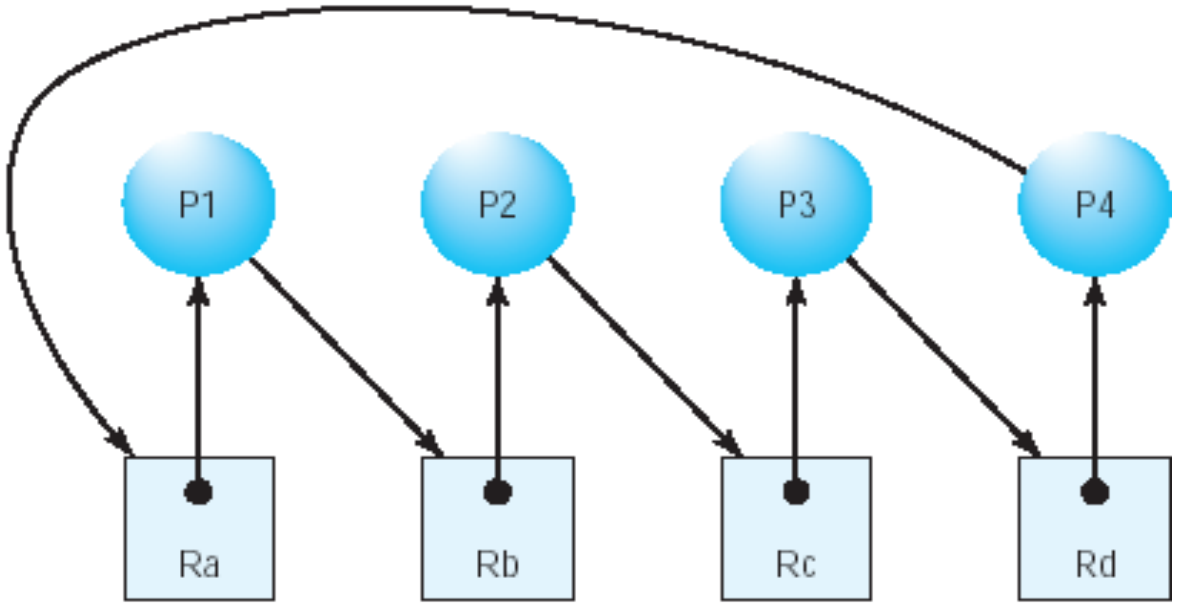
Cont… : Example

- Resources : two tape drives T1, T2.
- Allocation strategy:
  - Allocate the resource to the requester if free.
- Concurrent Processes : P1 and P2.

Cont…

- On request, P1 gets T1 and P2 gets T2.
- If either P1 or P2 requests for one more tape drives, process waits until another process releases the allocated tape.
- Processes are in the state of deadlock.

Cont…



A deadlock situation: There is a circular chain of processes and resources that results in deadlock.

Cont…

- Deadlock occurs:
  - Because the total requests of both processes exceed the total number of available resources, &
  - The resource allocation policy allocates a resource on request if the resource is free without considering the safe states.

Cont…

- This is also true for logical objects (like files, tables, records in a database, etc..).

- Non-Preemptable resource:
  - One that cannot be taken away from a process to which it was allocated until the process voluntarily releases it.

Resource Types

- Two general categories of resources:
  - Reusable &
  - Consumable.

- A reusable resource is one that can be safely used by only one process at a time and is not depleted by that use.
  - Eg.: Processors, I/O channels, main and secondary memory, devices, &
  - Data structures such as files, databases, and semaphores.

Consumable Resources

- A consumable resource is one that can be created (produced) and destroyed (consumed).
  - Example: interrupts, signals, messages, and information in I/O buffers.

Communication deadlocks

- Occurs among a set of processes, when these are blocked waiting for messages from other processes in the set, in order to start execution but there are no messages in transit between these.
  - All processes in the set are deadlocked.

Necessary conditions for deadlock

1. Mutual-exclusion condition
2. Hold-and-wait condition
3. No-preemption condition
4. Circular-wait condition

All are required for a deadlock to occur.

Cont…

- Mutual-exclusion condition
  - If a resource is held by a process, any other process requesting for that resource must wait until the resource has been released.

- Hold-and-wait condition
  - Processes are allowed to request for new resources without releasing the resources that are currently held.

Cont…

- No-preemption condition
  - A resource that has been allocated to a process becomes available for allocation to another process only after it has been voluntarily released by the process holding it.
- Circular-wait condition
  - Two or more processes must form a circular chain in which each process is waiting for a resource that is held by the next process of the chain.
  - It implies the hold-and-wait condition.

Deadlock modeling

- Deadlocks can be modeled using directed graphs.

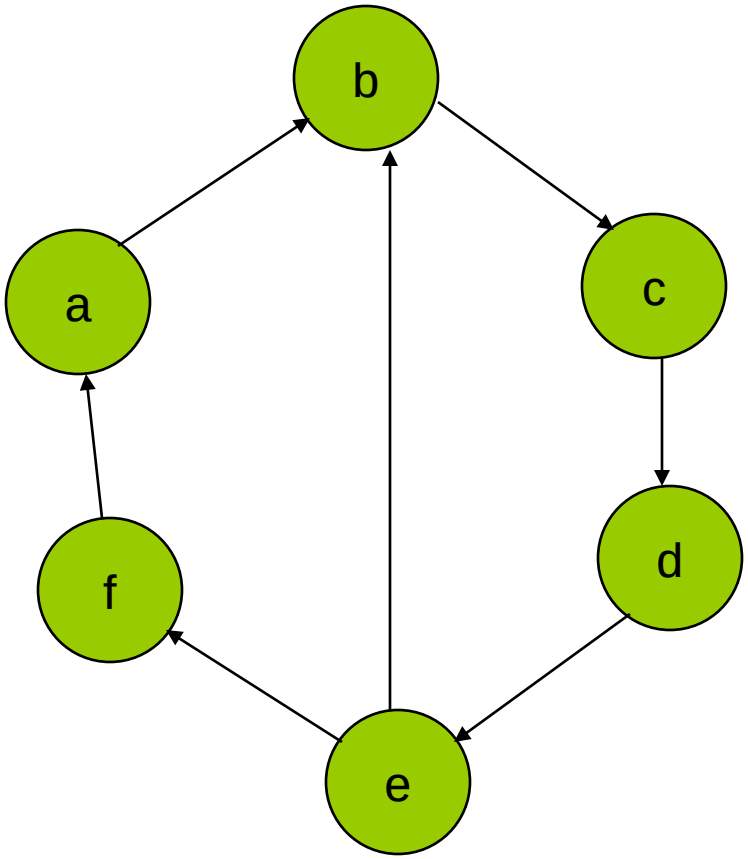
- Directed graph:
  - A pair (N,E), where N is a nonempty set of nodes and E is a set of directed edges.

- Path :
  - A sequence of nodes (a,b,c,….i,j) of a directed graph such that (a,b), (b,c),….. (i,j) are directed edges.

Cont…

- Cycle :
  - A path whose first and last nodes are the same.
- Reachable set:
  - The reachable set of a node ‘a’ is the set of all nodes ‘b’ such that a path exists from ‘a’ to ‘b’.
- Knot: A nonempty set ‘K’ of nodes such that the reachable set of each node in ‘K’ is exactly the set ‘K’.
  - It always contains one or more cycles.

A directed graph



Cycles :
1. (a,b,c,d,e,f,a)
2. (b,c,d,e,b)

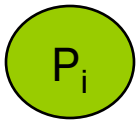
Knot:
{a,b,c,d,e,f}

Resource allocation graph

- Both the set of nodes and the set of edges are partitioned into two types, resulting in the following graph elements.

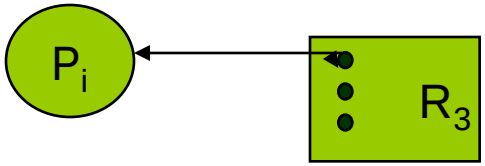
1. Process nodes
2. Resource nodes
3. Assignment edges
4. Request edges

Cont…

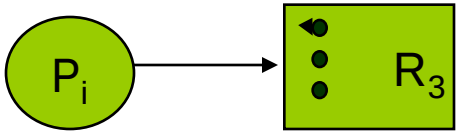


A process named P_i

A resource R_j having 3 units in the system.



Process Pi holding a unit of resource Rj.

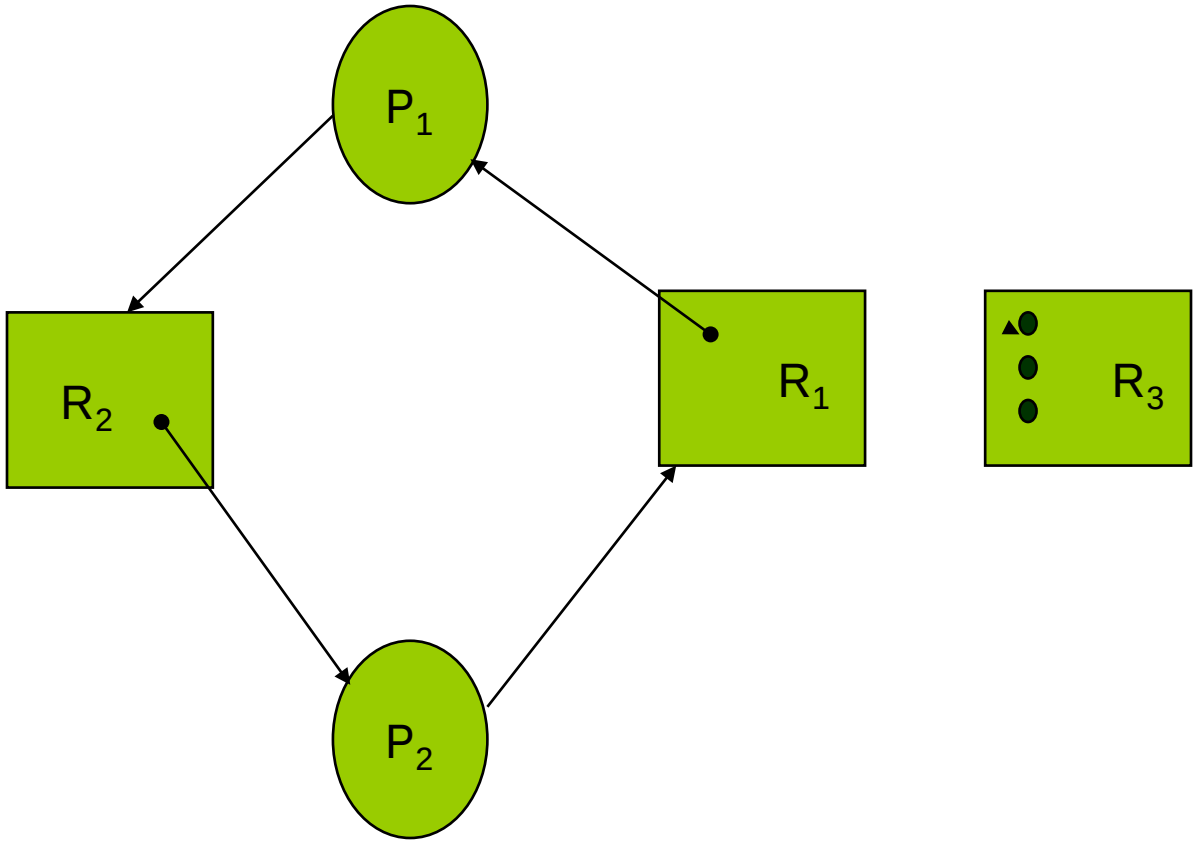


Process Pi requesting for a unit of resource Rj.

Necessary and sufficient conditions for deadlock

1. A cycle in the graph is both a necessary and a sufficient condition for deadlock
   - if all the resource types requested by the processes forming the cycle have only a single unit each.

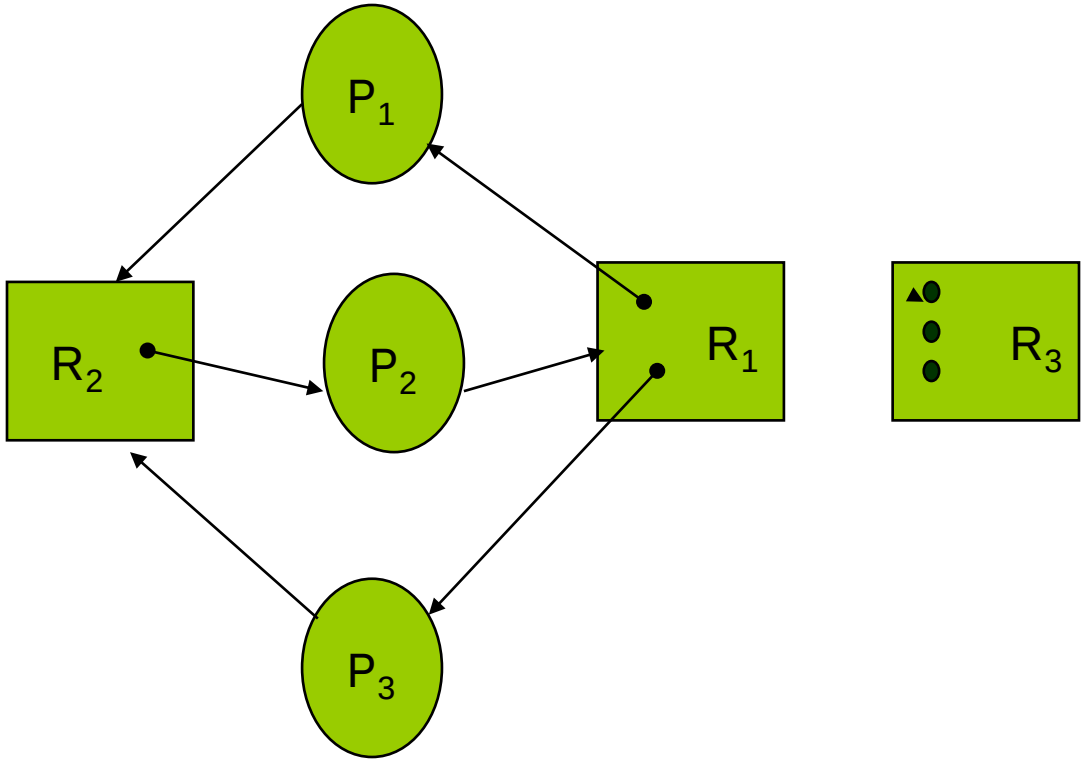
A cycle representing a deadlock



Cont…

2. A cycle in the graph is a necessary but not a sufficient condition for deadlock
   - if one or more of the resource types requested by the processes forming the cycle have more than one unit.
   - In this situation, a knot is the sufficient condition for deadlock.

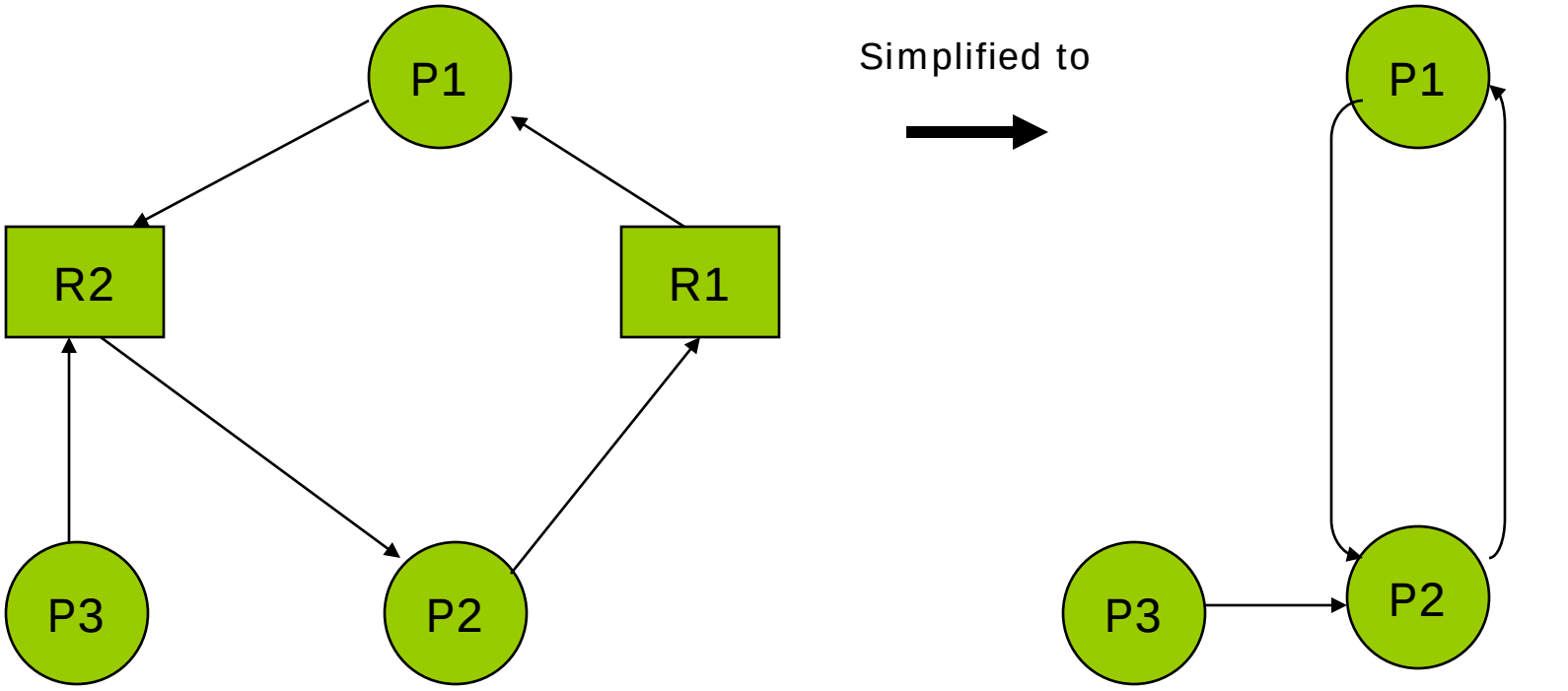
A Knot representing a deadlock



Wait-for graph

- A simplified graph, obtained from the original resource allocation graph by removing the resource nodes and collapsing the appropriate edges.
- It shows which processes are waiting for which other processes.
- Appropriate for modeling Communication Deadlocks.

Cont…



(a) Resource allocation graph

Corresponding WFG

Resource deadlocks

- Occurs when two or more processes wait permanently for resources held by each other.
- Handling deadlocks in Distributed Systems
- Strategies:
  - Avoidance
  - Prevention
  - Detection & recovery

Basic Deadlock Avoidance

- Resources are carefully allocated to avoid deadlocks.
- These methods use some advance knowledge of the resource usage of processes.
- Considers safe / unsafe states.

Cont…

- Safe state:
  - The system is not in a deadlock state.
  - There exists some ordering of the processes in which the resource requests of the processes can be granted to run all of these to completion.

Cont…

- Safe sequence:
  - Any ordering of the processes that can guarantee the completion of all the processes.
- Unsafe state:
  - A system state if no safe sequence exists.

Cont…

- Some assumptions of the algorithm:
  - The advance knowledge of the resource requirements of the various processes is available.
  - The number of processes that compete for a particular resource and the number of units of that resource are fixed and known in advance.

Deadlock Avoidance

- Approaches to deadlock avoidance:

1. Process Initiation Denial
2. Resource Allocation Denial

Process Initiation Denial

- Do not start a process if its demands might lead to deadlock.
- Define a deadlock avoidance policy that refuses to start a new process if its resource requirements might lead to deadlock.
  - A process is only started if the maximum claim of all current processes plus those of the new process can be met.
- Strategy is hardly optimal.
  - Assumes the worst: that all processes will make their maximum claims together.

Resource Allocation Denial

- Do not grant an incremental resource request to a process if this allocation might lead to deadlock.
- Also referred to as the **banker's algorithm**.

Cont…

- Consider a system with a fixed number of processes and a fixed number of resources.
- At any time, a process may have zero or more resources allocated to it.

Cont…

- State of the system
  - The current allocation of resources to processes consists of:
    - two vectors: Resource and Available &
    - two matrices: Claim and Allocation.

Deadlock Prevention

- Constraints are imposed on the ways in which processes request resources, in order to prevent deadlocks.
- Tries to ensure that at least one of the necessary conditions for a deadlock is never satisfied.

Cont…

- Deadlock prevention methods:
  - Collective requests
  - Ordered requests
  - Preemption

Collective Requests

- This method denies the hold-and-wait condition.
- Ensures that whenever a process requests a resource, it does not hold any other resources.

Cont…

- Resource allocation policies:
  1. Either allocate all requested resources or none, before a process begins execution.
     - Process would wait.

Cont…

2. Provide resources at time of execution only when process doesn't hold any other resource.

- First release existing once and then re-request all the necessary resources.
- Useful if requirements of resources is not known initially.
- Useful when some resources are required at the end of its execution.

Cont…

- Problems :
  - Low resource utilization
  - May cause starvation of a process
  - An accounting question.

Ordered Requests

- Each resource type is assigned a unique global number to impose a total ordering of all resource types.
- The process should not request a resource with a number lower than the number of any of the resources that it is already holding.
- If process requires several units of same resource type, it should make single request for all the required units of a resources.

Cont…

- A process may not acquire all its resources in strictly increasing sequence.
  - Eg. Holding resource # 7 & 10. Now release 10 & can request for 9.
- Problem :
  - Reordering may become inevitable when new resources are added to the system.

Preemption

- A preemptable resource is one whose state can be easily saved and restored later.
- Examples : CPU registers, main memory.

Resource allocation policies

1. Preempt all the resources of a process when it request for a resource that is not currently available.
   - Process is blocked.

2. If a resource is held by a waiting (waiting for another resource) process and another process requests for that,
   - Preempt the requested resource from the waiting process,
   - Allot to the requesting process.

Cont…

- Method works only for preemptable resources.

- So, further requirements are:
  - The availability of atomic transactions.
  - Global timestamps - in distributed and database transaction processing systems.

Cont…

- Transaction-based deadlock prevention method
  - Use of unique priority numbers for transactions.
  - Priority numbers are used to break the tie.
  - Global unique timestamp may serve as priority number.
  - Lower the value of timestamp means higher the priority .

Cont…

- Schemes based on the above idea:
  - Wait-die scheme
  - Wait-wound scheme

Wait-die scheme

- Three Transactions T_i (oldest), T_j and T_k(youngest)
- If a transaction T_i requests a resource that is currently held by another transaction T_j,
  - T_i is blocked because its timestamp is lower than that if Tj;
- If a transaction T_k requests a resource that is currently held by another transaction T_j,
  - T_k is aborted (dies).

Wait-wound scheme

- Three Transactions T_i (oldest), T_j and T_k(youngest)
- If a transaction T_i requests a resource that is currently held by another transaction T_j,
  - T_j is preempted because its timestamp is higher than that if T_j;
- If a transaction T_k requests a resource that is currently held by another transaction T_j,
  - T_k blocked because its timestamp is lower than that if Tj

Deadlock Detection and Recovery

- Deadlocks are allowed to occur.
- A detection algorithm is used to detect these.
- After a deadlock is detected, it is resolved by certain means.
- Detection algorithms are the same in both centralized and distributed systems.
  - Based on resource allocation graph and possibility of deadlock.

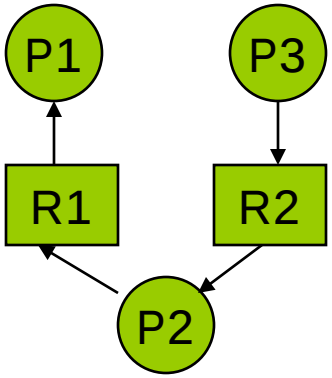
Cont…

- Properties for the correctness of deadlock detection algorithms:
  - Progress property
    - Deadlock must be detected in a finite amount of time.
  - Safety property
    - If a deadlock is detected, it must indeed exist.
    - No phantom deadlocks.

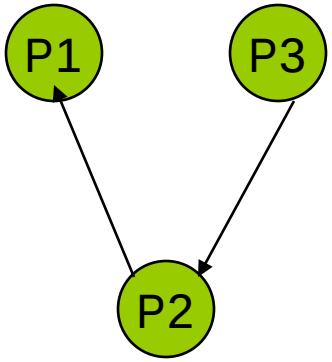
Cont…

- Steps to construct WFG for a distributed system:
  - Construct the resource allocation graph for each site of the system.
  - Convert that into a separate WFG for each site.
  - Take the union of the WFGs of all sites and construct a single global WFG.

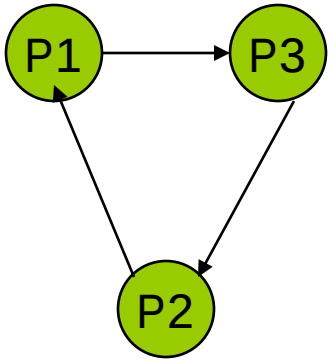
Cont… : Example



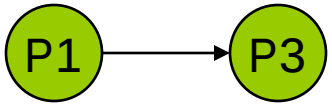
Site S1



Site S1



Site S2



Site S2

(A) Resource allocation graph for each site

(B) WFGs corresponding to graphs in (a)

(c) Global WFG by taking the union of the two local WFGs of (b).s

Cont…

- Local WFGs are not sufficient to characterize all deadlocks in a distributed system.

- So, the construction of a global WFG is required
  - Taking the union of all local WFGs.

Cont…

- Problem:
  - How to maintain WFG in a distributed system?
- Techniques are:
  - Centralized
  - Hierarchical
  - Distributed

Centralized approach for deadlock detection

- Use of local coordinator for each site
  - Maintains a WFG for its local resources
- Use of central coordinator/centralized deadlock detector
  - Receives information from the local coordinators of the sites.
  - Constructs the union of al the individual WFGs.

Cont…

- Approach:
  - Local deadlocks are detected and handled by the local coordinator.
    - Considers the cycles that exist in the local WFG of a site.
  - Deadlocks at two or more sites are detected and resolved by the central coordinator.
    - Considers the cycles in the global WFG.

Cont…

- Methods to transfer information from local coordinators to the central coordinator:
  - Continuous transfer
    - Transfer of message whenever a new edge is added to or deleted from the local WFG.
  - Periodic transfer
  - Transfer-on-request

Cont…

- Drawbacks of centralized deadlock detection approach:
  - Vulnerable to failures of the central coordinator.
  - Performance bottleneck in large systems having too many sites.
  - Detection of false deadlocks.

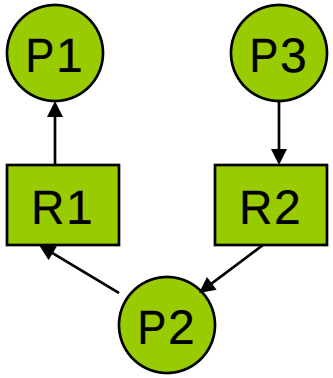
Example

- Processes : P1, P2 and P3
- Resources : R1, R2 and R3
- Steps:
  - 1: P1 requests for R1 and R1 is allocated to it.
  - 2: P2 requests for R2 and R2 is allocated to it.
  - 3: P3 requests for R3 and R3 is allocated to it.
  - 4: P2 requests for R1 and waits for it.
  - 5: P3 requests for R2 and waits for it.
  - 6: P1 releases R1 and R1 is allocated to P2.
  - 7: P1 requests for R3 and waits for it.

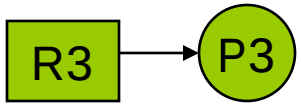
Cont…

- Sequences of messages sent to the central coordinator (using continuous transfer):
  - M1: from site S1 to add the edge (R1, P1).
  - M2: from site S1 to add the edge (R2, P2).
  - M3: from site S2 to add the edge (R3, P3).
  - M4: from site S1 to add the edge (P2, R1).
  - M5: from site S1 to add the edge (P3, R2).
  - M6: from site S1 to delete edges (R1,P1) and (P2,R1), and add edge (R1, P2).
  - M7: from site S2 to add the edge (P1, R3).

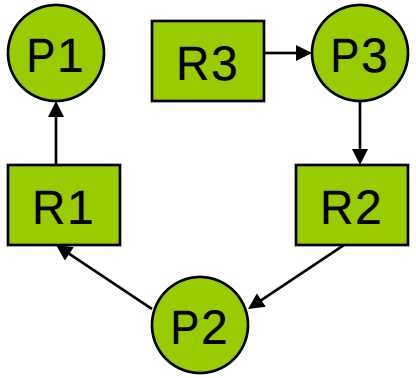
Cont…



Resource allocation graph of the local coordinator of site S1



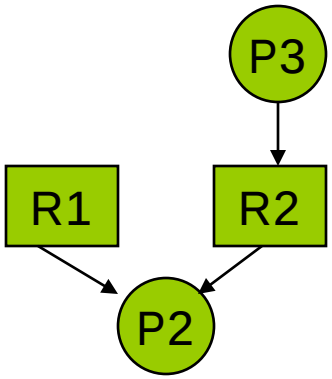
Resource allocation graph of the local coordinator of site S2



Resource allocation graph maintained by the central coordinator

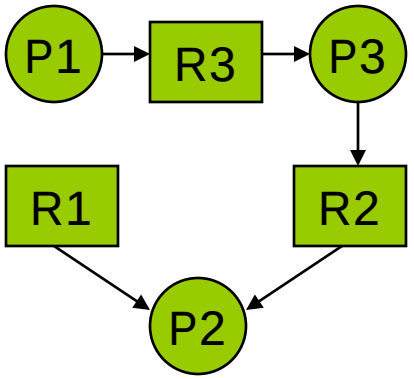
Resource allocation graphs after step 5

Cont…



Site S1

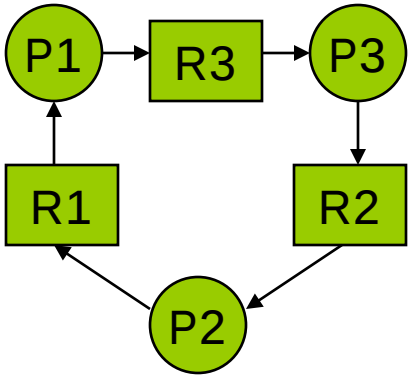
Site S2



Central coordinator

Resource allocation graphs after step 7

Cont…



Central coordinator

Resource allocation graph of the central coordinator showing false deadlock if message m7 is received before m6 by the central coordinator.

Cont…

- Result :
  - Possibility of detection of phantom deadlocks
    - when the method of continuous transfer of information is used.
    - In other methods of information transfer, due to incomplete or delayed information.

Cont…

- Solution:
  - Use unique global timestamp with each message.
- Disadvantages of centralized approach:
  - Less attractive for most real applications.
  - Time and message overhead.

Hierarchical approach for deadlock detection

- Uses a logical hierarchy (tree) of deadlock detectors, known as controllers.
- Each controller is responsible for detecting only those deadlocks that involve the sites falling within its range in the hierarchy.
- Approach is distributed over a number of different controllers.

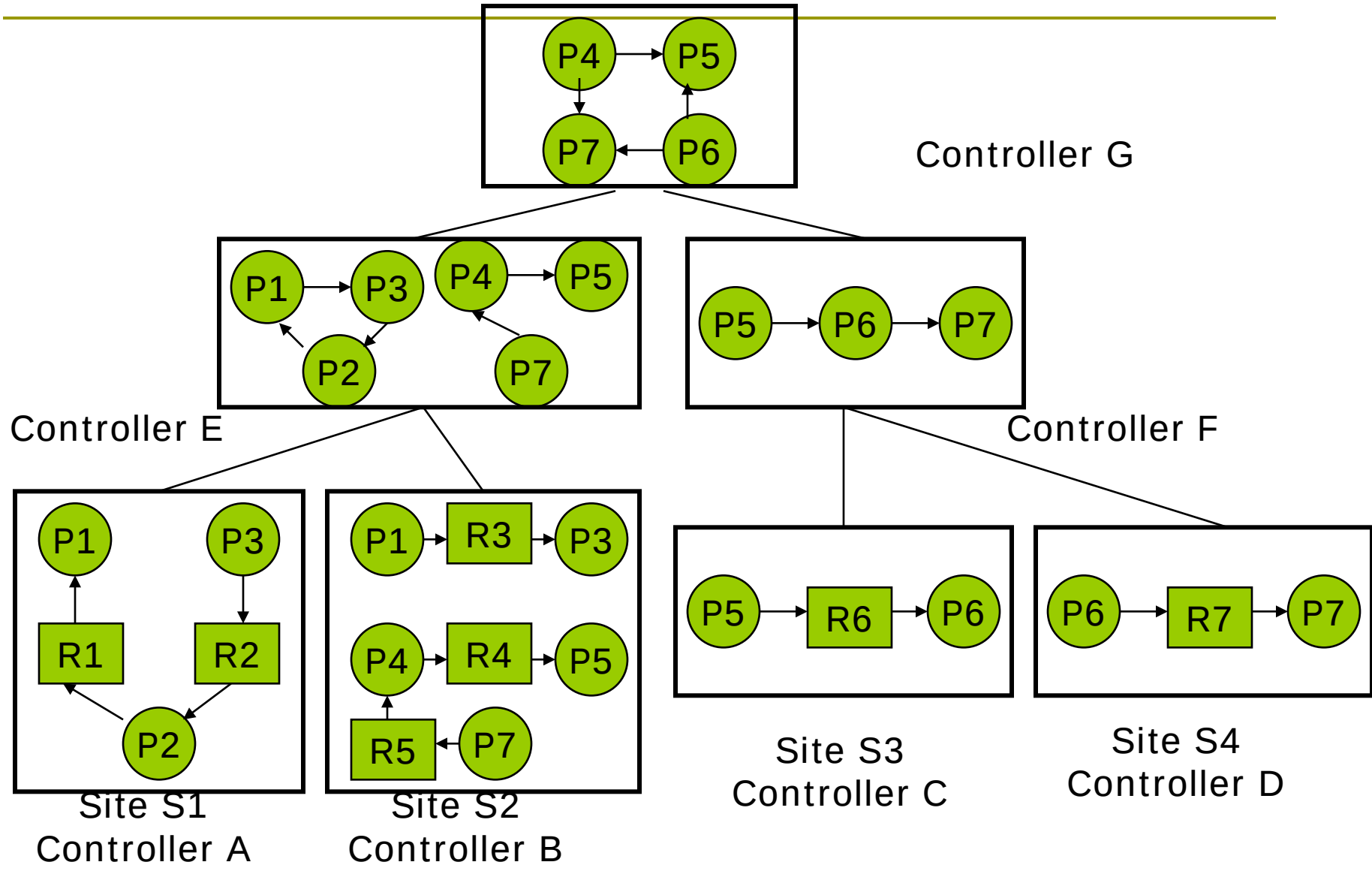
Cont…

- Rules :
  - Each controller that forms a leaf of the hierarchy tree maintains the local WFG of a single site.
  - Each nonleaf controller maintains a WFG that is the union of the WFGs of its immediate children in the hierarchy tree.

Cont…

- The lowest level controller that finds a cycle in its WFG
  - detects a deadlock and
  - Takes necessary action to resolve it.
- A WFG that contains a cycle will never be passed as it is to a higher level controller.

Example : Hierarchical deadlock detection approach



Cont…

- Deadlock cycle
  - (p1,p3,p2,p1) of site S1 and S2 gets reflected in the WFG of controller E.
  - (p4,p5,p6,p7,p4) of site S2,S3 and S4 gets reflected in the WFG of controller G.

Fully distributed approaches for deadlock detection

- Each site of the system shares equal responsibility for deadlock detection.
- Algorithms are:
  - *WFG-based distributed algorithm for deadlock detection.*
  - *Probe-based distributed algorithm for deadlock detection.*

WFG-based fully distributed deadlock detection algorithms

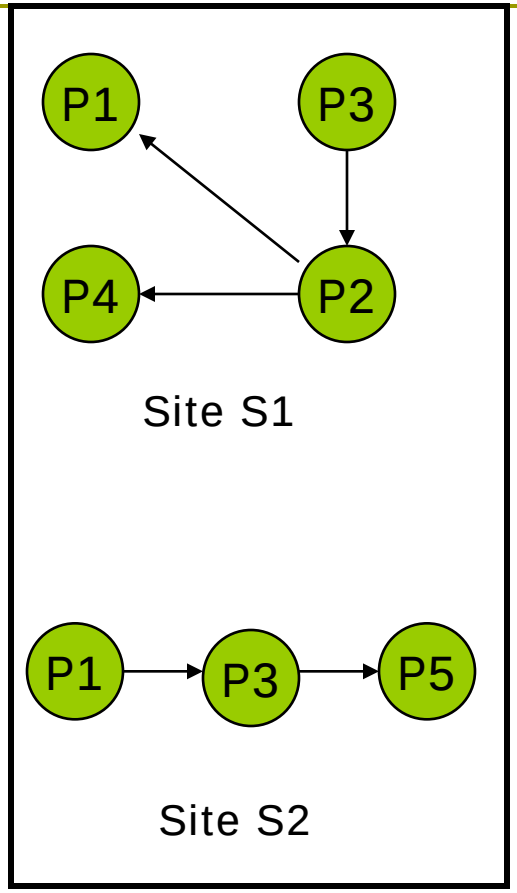
- Each site maintains its own local WFG.
- An extra node P_{ex} is added to the local WFG of each site.
- P_{ex} node is connected to the WFG of the corresponding site.

Cont…

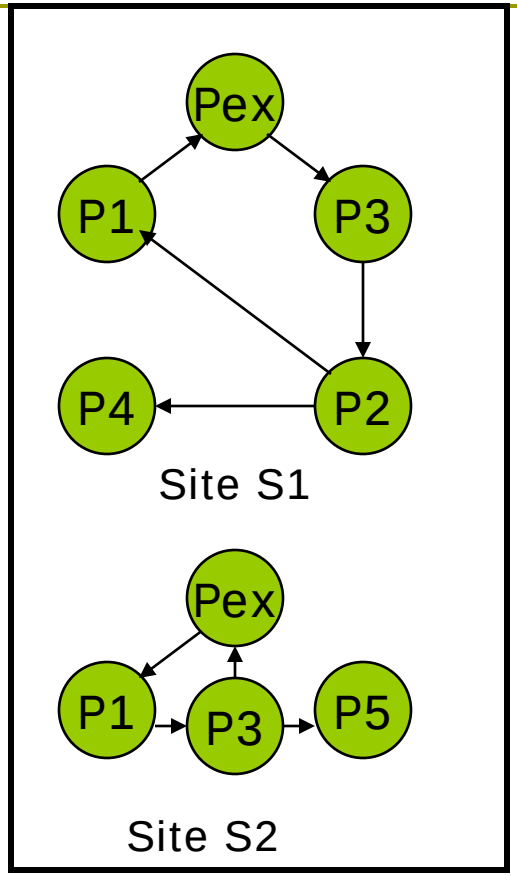
- An edge (Pi, Pex) is added if process Pi is waiting for a resource in another site being held by any process.
- An edge (Pex, Pj) is added if Pj is a process of another site that is waiting for a resource currently being held by a process of this site. (Pj is indicated as external process pointed by Pex).

- In the WFG of site S1, edge (P1, Pex) is added because P1 is waiting for a resource in the site S2 that is held by P3 &
- Edge (Pex, P3) is added because P3 is a process of site S2 that is waiting to acquire a resource held by P2 of S1.
- WFG of S2, edge (P3, Pex) added because P3 is waiting for a resource in S1 held by P2 &
- Edge (P3, Pex) added because P1 of S1 is waiting to acquire resource held by P3 of S2.

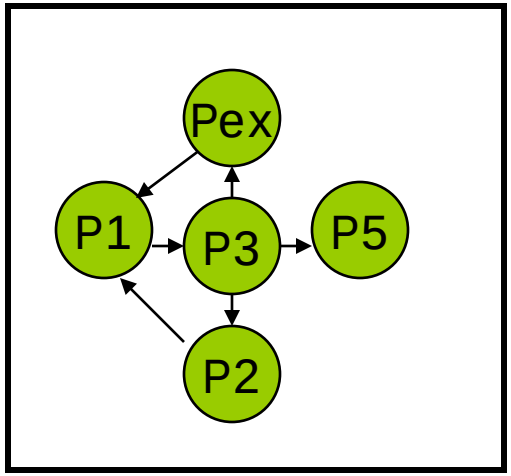
Example



Local WFGs



Local WFGs after addition of node Pex



Updating local WFGs of site S2 after receiving the deadlock detection message from site S1.

Cont…

- If a local WFG contains a cycle that:
  - does not involve node Pex, a deadlock that involves only local processes of that site has occurred – Resolved Locally
  - does involve Pex, there is a possibility of a distributed deadlock that involves processes of multiple sites – By Distributed Deadlock Detection Algorithm.

Cont…

- Problem:
  - Two sites may initiate the deadlock detection algorithm independently for a deadlock that involves the same processes.
  - More than the necessary process may be killed.

- Solution:
  - Assign a unique identifier to each process.

Probe-based distributed algorithm for deadlock detection

- Proposed by Chandy et al. in 1983.
- Also known as Chandy-Mishra-Hass or CMH algorithm.
- Use of probe message by the requesting process for a resource to the process holding the requested resource.

Cont…

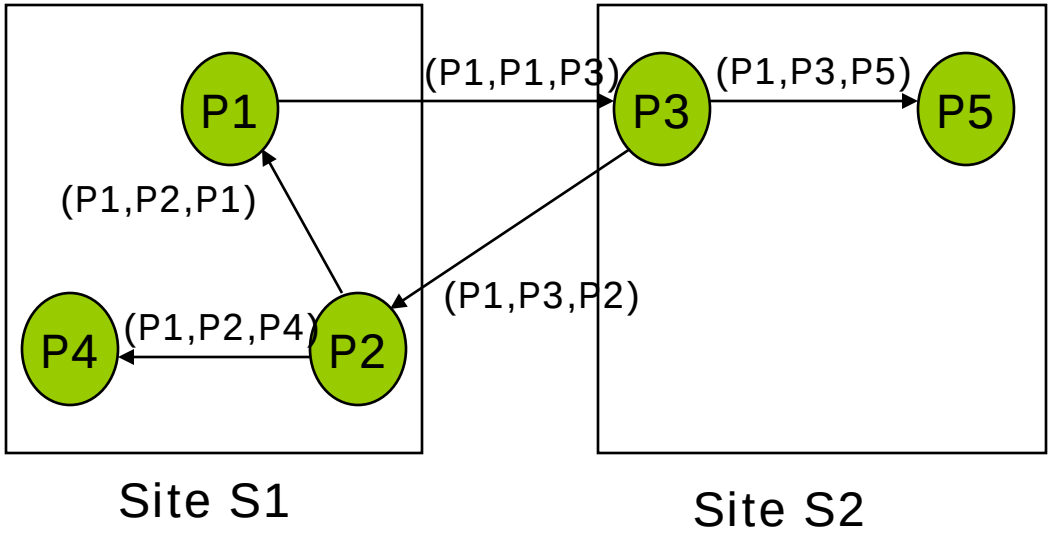
- Fields of probe message:
  - The identifier of the process just blocked.
  - The identifier of the sender process.
  - The identifier of the receiver process.

Cont…

- If recipient is using the resource, it ignores the probe message.
- If recipient itself is waiting for any resource, it forwards the probe message to the process holding the resource for which it is waiting.
- Cycle exists if the probe message returns back to the original sender.
  - System is deadlocked.
- A probe packet : (p1,p1,p3)

  p1 generates a probe message and sends it to p3.

Cont…: CMH distributed deadlock detection algorithm



Cont…

- Features of the CMH algorithm:
  - Easy to implement.
    - Each message is of fixed length.
    - Few computational steps.
  - Low overhead of the algorithm
  - No graph construction and information collection
  - Doesn’t detect false deadlocks
  - Does not require any particular structure among the processes.

Methods for recovery from deadlock

- Asking for operator intervention.
- Termination of process (es).
- Rollback of process (es).

Asking for operator intervention

- Inform the operator about the deadlock.
- Let the operator deal with it manually.
- Works for a centralized system.
- Not suitable for dealing with the deadlocks involving processes of multiple sites.

Termination of process

- Terminate one or more processes and reclaim the resources held by them for reallocation.

- Requirements:
  - Analyze the resource requirements and interdependencies of the processes involved in a deadlock cycle.

Rollback of process

- Reclaim the needed resources from the processes that were selected for being killed.
- Rollback the process to a point where the resource was not allocated to the process.
- Processes are checkpointed periodically.
- Approach is less expensive than the process termination approach.
  - Extra overhead involved in periodic checkpointing of all the processes.

Issues in recovery from deadlock

- Selection of victim (s)
- Use of transaction mechanism

- Victim: the process which is killed or rolled back to break the deadlock.

- Factors for victim selection:
  - Minimization of recovery cost
  - Prevention of starvation

Election algorithms

- Most algorithms requires a coordinator process to perform some type of coordination activity.
- Election algorithms are used for electing a coordinator process from among the currently running processes.
- At any instance of time there is a single coordinator for all processes in the system.

Cont…

- Assumptions:
  - Each process in the system has a unique priority number.
  - Process with highest priority will be elected as the Coordinator.
  - On recovery, a failed process can take appropriate actions to rejoin the set of active processes.

Bully algorithm

- Assumption :
  - Every process knows the priority number of every other process in the system.

- When a process (pi) sends a request to the coordinator and does not receive a reply within a fixed timeout period, it starts election process assuming that the coordinator has failed.

Cont…

- Pi sends election message to all with a higher priority number than itself.
  - Informing others that now it is the coordinator.
- Process (Pj) with the higher priority number than the mentioned one, sends alive message.
  - Now Pj is the coordinator.
- On recovery of previous coordinator
  - It simply sends a coordinator message to all other processes and bullies the current coordinator into submission.

Cont…

- Algorithm requires
  - $O(n^2)$ messages in worst case.
  - (n-2) messages in the best case.

Ring Algorithm

- All the processes in the system are organized in a logical unidirectional ring.
- Every process in the system knows the structure of the ring.
- If the successor of the sender process (Pi) is down, may be the coordinator,
  - It may start election process.
  - the sender can skip over the successor, or
  - The one after that, until an active member is located.

Cont…

- The election message contains the priority number of sender process (Pi).
- Processes enter their priority number to the message and pass it to the successor.
- Pi recognizes its own election message.
  - It elects the process having the highest priority as the new coordinator.
  - It sends a coordination message over the ring about the new coordinator.

Cont…

- If previous coordinator (Pj) recovers after failure:
  - Sends an inquiry message to its successor.
  - The current coordinator replies to Pj that it is the current coordinator.

Cont…

- Drawback :
  - Two or more processes may circulate an election message over the ring on discovery that the coordinator has crashed.
  - An election always requires 2(n-1) messages.
  - n/2 messages are required on an average for recovery action.
  - More efficient and easier to implement than bully algorithm.