Evaluating synchronization mechanisms

Categorizing resource synchronization methods

Two contradictory criteria:
- Expressiveness
- Power

or: area of requests or synchronization is to be addressed
or: flexibility of requirements to be expressed

Eased use
- Access to parameters for control
- How to organize methods for complex resource control systems?

Handling requests by type

Max_Resources : Range
resource_control

Handling requests by parameters

Resource_Range

Resource Manager

Evaluating synchronization mechanisms

Accepting or Avoiding?

Conditional wait
- All threads are waiting in the synchronization format
- Client might not be able to handle resource requests

Avoidance synchronization
- All requests at the bottom of the synchronization server
- Client can easily handle requests

Handling requests:
- Entry
- Accept
- Update
- Free

Potential implications:
- Requests are no longer atomic
- Requests might be handled in a different order
- Deadlocked client might be treated differently than others
- Various priorities or some clients might be served before others

Resource Control

Topics in real-time resource control

- Synchronization and schedulers to resource management
- Resource availability and interaction
- Resource reclaiming
- Resource reservation schemes

References for this chapter


Evaluating synchronization mechanisms

Handling requests by type and parameters

Type: Resource_Range_Groups

Type: Resource_Request

Timing constraints occur in all forms of resource requests.

Eliminate the double interaction by means of a unification, single request type and requeuing.

Motivation for resource reclaiming

1. Worst case assumptions give schedulable systems, but might leave only a few spare resources.
2. Some scheduled resources might not be actually used at runtime.
3. Some aspects of schedulability in real-time systems rely directly on the amount of spare resources.

Resource reclaiming may enhance the system's reliability.

Resource reclaiming algorithms

Two categories:

- Greedy method
- Postrun schedule

Postrun schedule: S

Prerun schedule: P

Resource reclaiming for independent tasks

Greedy method

Postrun schedule: S

Resource reclaiming for independent tasks

Greedy method

Postrun schedule: S
Resource reclaiming for interdependent tasks

Restriction vectors

1. Computed RV(p) by checking the latest most tasks c_i(p)
2. For any task t the release schedule is zero or
   a. If the resource is free, then
   b. If not, the release schedule is the demand of the task

Restriction vectors have the same meaning that if task checking the 6 most recent tasks c_i(7..p)

Resource reclaiming for interdependent tasks

Resource reclaiming for interdependent tasks

Resource reclaiming for interdependent tasks

Resource reclaiming for interdependent tasks

Resource reclaiming for interdependent tasks

Resource reclaiming for interdependent tasks

Proof of correctness

Lemma Given a non-release schedule is \( S(\cdot, \cdot) \) then its corresponding resource release schedule is feasible

Proof Assume the release schedule is not feasible

Case 1: \( S(\cdot, \cdot) \) has at least one resource conflict

Case 2: \( S(\cdot, \cdot) \) has no resource conflict

Resource reclaiming for interdependent tasks

Resource reclaiming for interdependent tasks

Resource reclaiming for interdependent tasks

Resource reclaiming for interdependent tasks

Resource reclaiming for interdependent tasks

Resource reclaiming for interdependent tasks

Resource reclaiming for interdependent tasks

Resource reclaiming for interdependent tasks
Resource Control

Real-time Resource Control [Mercer97]

Resource Control Issues

Policies:

- Priority assignment problem: Mapping and prioritization of the known and arising events.
- Overload problem: Predicting and protecting the system from overload conditions.
- Flexibility problem: Locally adjusting the system behavior to the current timing constraints.

Run-time environment:

- Enforcement problem: Handling tasks and resources which exceed their anticipated worst case limits.
- Measurement problem: Recording all relevant information in sufficient resolution and frequency.
- Coordination problem: Synchronizing system components according to different policies.

Resource Control

Summary

- Resource synchronization primitives:
  - Evaluation of resource synchronization methods.
  - Priority, deadline, and reliability consideration.
- Resource reclaiming schemes:
  - Basic reclaiming
  - Early start algorithm
  - Restriction strategy
  - Resource reallocation with task migration.
- Real-time resource control:
  - Task preemption to be considered.