CS681 :
Simulation Program Design

Anuj Mittal • 140050024
Sumith Kulal • 140050081

Overview

Table of Contents

Classes

Classes

- Simulation - Top level class
- Queuing Network
- Server
- Core
- User
- Thread
- Request
- EventHandler
- Metrics

Simulation

- QueuingNetwork
  - A class to handle the queuing network
- EventHandler
  - A class to handle events
- Metrics
  - A class to handle the metrics
- lastEventTime
- simulationTime

Queuing Network

- vector<User> users
  - User - a class to handle an user
- Server
  - A class to handle the server
- Buffer
  - Queue of request pointers

Server

- vector<Core> cores
    - Core - a class to handle a core
- timeQuantum
- numActiveThreads
- maxNumThreads
- numCoresInUse

Core

- ID
- status
  - busy or idle
- LinkedList<Thread> threads
  - Thread - a class to handle a thread
- Iterator to the current Executing Thread
- Server* server

Thread

- Request* currRequest
- Core* affinedCore
  - core to which this thread is affined

Request

- ID
- status
  - good or bad
- Arrival Time
- Remaining Service Time
  - Initialized with the service time of the request
- Start Time Of Current Quantum
  - -1 if request not being executed
- User* issuer
- Thread* assignedThread

User

- ID
- state
    - thinking or waiting
- Request* issuedReq

EventHandler

- EventList
  - priorityQueue of Event
  - 4 types of Event
    - NEW_REQ : New request Issue
      - stores Pointer to User Issuing Request
    - REQ_COMP : Request gets Completed
      - stores pointer to Request Completing
    - REQ_OUT : Request Time Out
      - Stores pointer to Request getting timed out
    - CTX_SWH : Core Context Switch
      - stores pointer to Core Context Switching

Metrics

- Simulation Start/End Time
- Number of Good/Bad Requests Completed
- Number of Requests Dropped
- Total Response Time of Good/Bad Requests
- Total Area of Core Utilization
  - Sum of (Number of core Utilized) * (time period)
- Total Area of Number of Requests in System
  - Sum of (Number of Requests in System) * (time period)

Program Flow

System Inputs

- Number of Users (M)
- Number of Cores
- Max Number of Threads
- Buffer Size
- Time Quantum for Round Robin Scheduling
- Max Simulation Time
- Distribution for Think Time, Service Time and Timeout

Initialization & Termination

- Initialize all the classes with the given input parameters
- For each user, insert a NEW_REQ event after a time sampled from think time distribution
- Terminate if the simulation time becomes greater than max Simulation Time

Loop

- Pick the event in order of time from the EventHandler
- Update simulation time
- Update metrics
  - Only to be done if simulation not in transient phase
- Take steps to handle the events
  - Depends on the event type

Logic : Event Handling

New Request - (1/2)

- Update Metrics
    - Core Utilization / Number of Requests
- Create a new Request Class
- Create a REQ_OUT Event for this request
- If Buffer is Full
    - Drop the request
    - Update metric for number of Requests Dropped

(contd...)

New Request - (2/2)

- If MaxThreads are Active
  - Insert In Queue if Max Threads Active
- Else
  - create a Thread and assign it to Core with least number of Threads
  - If assigned Core has no currently executing thread
    - Schedule this thread
    - Add CTX_SWH/ REQ_COMP Event depending on if request will be completed in time Quantum or not
- Update status variables like core status, user status, num of cores in use, num of active threads etc

Request Complete - (1/2)

- Update Metric
  - Core Utilization/Number of Requests
  - Number of Good/Bad Req Completed and Response Time Depending on request status
- Delete Request and Thread
  - Updates status variables like numActiveThreads, core status etc
- Insert NEW_REQ event for the issuing user at (current time + think time)

(contd...)

Request Complete - (2/2)

- Schedule Next Process In the List (if any)
    - Add CTX_SWH/ REQ_COMP Event depending on if request will be completed in time Quantum or not
- Update status variables like core status, user status, num cores in use, num of active threads etc

Request Time Out

- Ignore if Request Already Served
- Update Metrics
  - Core Utilization/Number of Requests
- Change the status of the request to BAD
- Insert NEW_REQ event for the issuing user at (current time + think time)
- Update status variables like user status etc

Context Switch

- Update Metrics
  - Core Utilization/Number of Requests
- If the core has more than one thread
  - Update the remaining time of current executing request,
  - Schedule next request in the list
  - Add REQ_COMP Event if the request will complete in time Quantum
  - Else add CTX_SWH Event

Thank You