Multimedia Operating System (2)

Hai Tao
Continuous Media Resource Model

- A model frequently adopted to define QoS parameters
- Based Linear Bounded Arrival Process (LBAP)
- A distributed system is decomposed into chain of resources traversed by the messages on their end-to-end path. Resource can be CPU, networks, etc
- The data stream consists of LDUs, or messages. Various data streams are independent to each other
- LBAP is a message arrival process
  - \( M \) = maximum message size (byte/message)
  - \( R \) = maximum message rate (message/sec)
  - \( B \) = maximum burstiness (message)
  - Burst consists of messages arriving ahead of schedule
LPAB Example

- Two computers connect by LAN
- One computer reads data from CD player and transmit the data over LAN
- Remote computer delivers the data to speakers
- Audio data are sampled at 44.1 kHz, 16 bit coding
- The data rate is \( R_{\text{byte}} = 44100 \times 2 = 88200 \, \text{bytes/s} \)
- Sample are assembled into “frames”, which are messages sent to the remote computer
- \( R = 75 \, \text{messages/s} \)
- \( M = 88200 \, \text{bytes} / 75 \, \text{messages} = 1176 \, \text{bytes/message} \)
- Up to 12000 bytes are packed into one packet and sent over LAN
LPAB Example

- Variance in the data rate results in an accumulation of messages (burst). The maximum range is defined by the maximum allowed number of messages. Since the adjacent data rate is low, we assume the burst is never larger than 1 packet, or $12000/1176=10$ messages.

- Therefore
  - $M=1176$ bytes/message
  - $R=75$ messages/s
  - $B=10$ messages

- Maximum average data rate
  - $M \times R=88200$ bytes/s

- Maximum buffer size
  - $M \times (B+1)$ bytes $=1176 \times (10+1)=12936$ bytes
  - Burst must be stored in the buffer
LPAB example

- **Burst**
  - Since the data rate is relatively low, the burst is less than or equal to one packet.
  - During a time interval $t$, the maximum number of messages received is $B + Rxt$ (messages).
  - In our example, for a period of 1s, this is $10 + 75 \times 1 = 85$ messages.
  - The introduction of $B$ allows us to model short time violations of rate constraint.

- **Maximum average data rate**
  - The maximum average data rate is $M \times R = 75 \times 1176$ bytes/s = 88200 bytes/s.
LPAB model

- **Maximum buffer size**
  - Message arrive ahead of time must be queued. If the burst 10 message, then we need buffer of size
    \[(B+1)xM=11x1176 \text{ bytes} = 12936 \text{ bytes}\]

- **Logical backlog**
  - For each message m, we can define a function \( b(m) \), which means the number of message which have already arrived ahead of schedule at the arrival of m. Let \( a_i \) be the actual arrival time of \( m_i \), the function \( b(m_i) \) is defined as
    \[
b(m_0) = 0
    \]
    \[
b(m_i) = \max(0 \text{ messages}, b(m_{i-1})-(a_i - a_{i-1})R+1 \text{ messages})
    \]

For example

- \( a_{i-1} = 1.00s, a_i = 1.013s, b(m_{i-1}) = 4 \text{ messages} \)
- \( b(m_i) = \max(0, 4 \text{ messages}- (1.013-1.0)s \times 75 \text{ messages/s}+1 \text{ message}) \)
  - \( = \max(0, 4 \text{ messages} - 0.975+1 \text{ messages}) = 4 \text{ messages} \)

The first term is the previous backlog; the second term is how many messages can be consumed; the last term 1 is current message
LPAB model

- **Logical arrival time**
  - Defines the earliest time a message $m_i$ can arrive at a resource when all messages arrive according to their rate (including backlog).
  - This is can be defined as
    \[ l(m_i) = a_i + \frac{b(m_i)}{R} \]
  - For example
    \[ l(m_i) = 1.013s + \frac{4}{75s} = 1.06s \]

  Another way of computing this is
  \[ l(m_0) = a_0 \]
  \[ l(m_i) = \max(a_i, l(m_{i-1}) + \frac{1}{R}) \]
  - For example
    \[ l(m_{i-1}) = 1.053s \]
    \[ l(m_i) = \max(1.013s, 1.053s + \frac{1}{75s}) = 1.06s \]
LPAB model

- Workahead messages
  - If a message arrives ahead of schedule and the resource is in an idle state, the message can be processed immediately. A maximum workhead time $A$ can be specified for each process. This results in a maximum workahead limit $W$ defined as

$$W = AxR$$

For example: $A = 0.04s$

$$W = 0.04s \times 75 \text{ message/s} = 4 \text{ messages}$$
Process Management

- Allocate the resource of the main processor
- Map single processes onto resource according to a specified scheduling policy such that all processes meet their requirement
- In a system, a process can be in one of the following states
  - Idle state: no process is assigned to the program
  - Blocked state: waiting for necessary resource to process
  - Ready-to-run: All other necessary resources are assigned to the process, ready to run
  - Running state: the system processor is assigned to the process
Process Management

- Process manager assign ready-to-run processes positions in the respective queue of the dispatcher.
- The dispatcher manages determine which process to be assigned to the processor:
  - Priority policy can be used.
  - Longest ready time can be used as another criterion.
Process management example

- UNIX, Windows-NT, IBM OS/2, and some other operating system are designed with multi-task scheduling capability.
- An example of operating system without timesharing is the early MS DOS system.
- The timesharing scheduler in these conventional operating system is not designed particularly for real-time applications.
- Understanding of the basic concepts in these system will shed light on what is needed for true real-time multimedia system.
Threads

- Thread can be considered as a light weight process
  - Dispatchable unit of execution
  - Each thread belongs to one process
  - All threads share the resources of the process they belong to
  - Each thread has its own execution stacks, register values, and dispatch states
  - Each thread belongs to one of the following priority classes
    - Time-critical class – immediate attention
    - Fixed-high classes – require good response but not time critical
    - Regular – normal tasks
    - Idle-time class – lowest priority, dispatched if there is no other processes
Priority

- Time slicing: process time is divided into fixed length units, lasting from 32 msec to 65536 msec
  - Larger time slice: efficient in executing large jobs, slow response time
  - Small time slice: quick response, less efficient because of context switch
  - The optimal slice time is determined by an adaptive algorithm
- For OS/2, in each class, 32 priorities exist
- Threads of the same priority have equal chances to execute
- Priority may be changed for regular class based on waiting time
- Threads are preemptive: higher priority means execution (non-preemptive means only voluntarily give up)