Real Time Systems 2

Focused Addressing and Bidding (FAB) algorithm

Lecture 10

Focused Addressing and Bidding (FAB)

- FAB algorithm can be used as an online procedure for both critical and non-critical real-time tasks.

- Critical tasks must have sufficient time to execute successfully, even if they need their worst-case execution time.

- The non-critical tasks are either processed or not, depending on the system's ability to do so.

Focused Addressing and Bidding (FAB)

- The underlying system is:
- When a noncritical task arrives at processor ***pi***, the processor checks to see if it has the resources and time to execute it, without missing any deadlines of the critical tasks or the previously guaranteed noncritical tasks.

- If yes, ***pi*** accepts this new noncritical task and adds it to its list of tasks to be executed and reserves time for it.

Focused Addressing and Bidding (FAB)

- Since the task is noncritical, the guarantee can be based on the expected-case run time of the task rather than the worst-case run time.
- If not; the FAB algorithm tries to find out some other processor in the system to ship task, so it works as follows:

- Every processor maintains two tables called: *status table* and *system load table*.

Focused Addressing and Bidding (FAB)

- **Status table**: includes the execution time and periods of critical tasks have been already committed to run (which were preassigned statically), and any additional noncritical tasks that have been accepted..

- **System load table**: contains the surplus computational capacity of the latest load information at every other processors in the system.

- Every processor on receiving a broadcast from a node updates the system load table.

Focused Addressing and Bidding (FAB)

- Since the system is distributed, this information may never be completely up to date.
- As a result, when a task arrives at a node, the node first checks whether the task can be processed locally.
- If yes, it updates its status table.
- If not, it examines the system load table to look for a processor to offload the task.
- An overloaded processor *pi* checks the surplus information and:

Focused Addressing and Bidding (FAB)

- 1) Selects a processor (called the focused processor) ***ps*** that is most likely to successfully execute that task by its deadline.

- 2) The system load table information might be out of date and its possible that the selected processor will not have time to execute the task, as insurance against this, the originating processor ***pi*** decide to send *requests for bids* (RFB) to other lightly loaded processor in parallel with sending the task to the focused processor ***ps*** .

Focused Addressing and Bidding (FAB)

- This is to gain time in case ***ps*** refuses the task.
- The RFB contains the vital statistics of the task, its expected execution time, any other resource requirements, its deadline, etc.

- 3) The RFB asks any processor that can successfully execute the task, to send a bid to the focused processor ***ps*** stating how quickly it can execute the task.
- 4) RFB is only sent out if the sending processor ***pt*** estimates that there will be enough time for timely response to it.
- 5) Specifically, two times t_{bid} and $t_{offload}$ are calculated.

Time Calculations By Fab Algorithm

- t_{bid} = (estimated time taken by RFB to reach its destination) + (the estimated time taken by the destination to respond with a bid) + (the estimated time taken to transmit the bid to the focused processor);

- $t_{offload}$ = (task deadline) – [(current time) + (time to move the task) + (task-execution time)];
- If $t_{bid} <= t_{offload}$, then RFB is sent out.

Time Calculations By Fab Algorithm

- When a processor ***pt*** receives an RFB, it checks to see if it can meet the task requirements and still execute its already-scheduled tasks successfully.
- First will estimate when the new task will arrive and how long will take, to be either guaranteed or rejected:
- t_{arr} = (current time) + (time for bid to be received by ***ps***) + (time taken by ***ps*** to make a decision) + (time taken to transfer the task) + (time taken by ***pt*** to either guarantee or reject the task);

Time Calculations By Fab Algorithm

- Next calculates the surplus time between the absolute deadline D, current time, and the computational time spoken for in the interval [*tarr*, D]:
- *tcomp* = (time allotted to critical tasks in [*tarr*, D]) + (time needed in [*tarr*, D] to run already-accepted noncritical tasks] + (fraction of recently accepted bids) * (time needed in [*tarr*, D] to honor pending bids);
- *tsurplus* = D−(current time)−*tcomp*;
- If the task worst-case run times are used the bids will be very conservative; if the average-case values are used, the bids will be less conservative

Time Calculations By Fab Algorithm

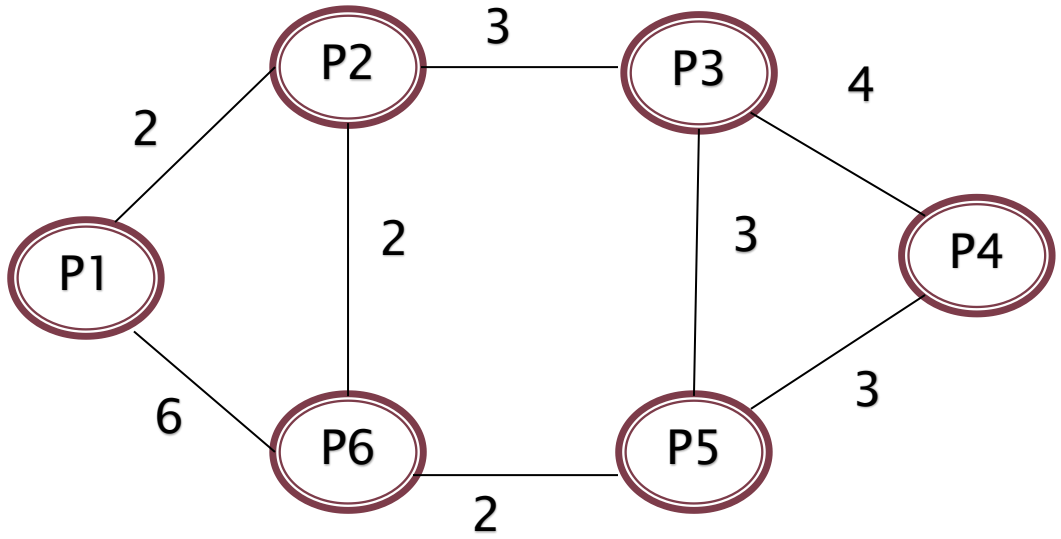
- If $t_{surplus}$ < (task execution time), then no bid is sent out.
- If $t_{surplus}$ >= (task execution time), ***pt*** sends out a bid to ***ps*** (the focus processor);
- The bid contains t_{arr}, $t_{surplus}$, and an estimate of how long a task transferred to ***pt*** will have to wait before it is either guaranteed or rejected;
- All bids are sent to the focused processor ***ps***, if ***ps*** is unable to process the task successfully, it can review the bids it gets and see which other processor is most likely to be able to run the task and transfer the task to that processor.

FAB Examlpe:

- Consider a system have 6 CPUs with current surplus = 0,15,7,13,8 and 9 respectively, the originating is CPU1and communication cost (ci,j) is, c12=2, c16=6, c23=3, c26=2, c34=4, c35=3, c45=3, c56=2, every CPU takes 1 time unit to make decision if it can accept the bid or not. A non-critical task T(0,2,20) its arrival, execution time and deadline arrives at CPU1.
- a): which CPU is the focused CPU?
- b): which CPU will receive the bid and continues its calculation?

FAB Examlpe:

- $t_{bid}\,3 = (2+3)+1+3 = 9$
- $t_{bid}\,4 = (2+3+4)+1+7 = 17$
- $t_{bid}\,5 = (2+2+2)+1+4 = 11$
- $t_{bid}\,6 = 6+1+2 = 9$
- $t_{offload} = 20 - (0+2+2) = 16$



- If $t_{bid} <= t_{offload}$, then RFB is sent out.
- **So, CPU3 ,CPU5 and CPU6 will receive the RFB signal.**