EECS 571 “Principles of Real-Time Embedded Systems”

Lecture Note #9: More on Uniprocessor Scheduling

Kang G. Shin
EECS Department
University of Michigan

Online Load Sharing of Aperiodics

K. G. Shin and C.-J. Hou, “Analytic models of adaptive load sharing schemes in distributed real-time systems,” *IEEE Trans. on Parallel and Distributed Systems* vol. 4, no. 7, pp. 740–761, July 1993

- *Meet* deadlines of aperiodics by transferring them whenever some of their deadlines can’t be met locally.

- Three basic issues:

  - o When to xfer tasks?
    Transfer policy: static or dynamic thresholds

  - o Where to xfer tasks?
    Location policy: sender-/receiver-initiated, or a hybrid.

  - o What information to exchange?
    Information policy: type, frequency, and format

Transfer Policy

- Need to measure the workload of each node by
  - Queue length (QL)
  - Cumulative execution time (CET)
- Using thresholds of QL, a node is classified as underloaded, fully-loaded, or overloaded.
- Using CET, a node can tell if a task can be guaranteed or not.
- *State* of a node: QL, CET, # and type of resources available, or combinations thereof

Bidding Algorithm

When a task arrives at a PN,

- the PN checks if it has sufficient resources to execute the task in time.
- If not, it sends a request for "bids" (RFB) from other PNs and allocates the task accordingly.

Questions:

- What to include in RFB and bid?
- How to calculate/estimate elements of RFB and bids?
- When to send or not send them?
- How to deal with outdated information in bids?
- How aggressive should a bid be?

Focused Addressing & Bidding

Online assignment & scheduling of non-critical aperiodics.

- Each PN maintains
  - o a table of both critical and noncritical tasks it accepted to run.
  - o a table of other PNs' surplus computational capacity.
- Each PN regularly sends fraction of next window that is currently free
- An overloaded PN checks its surplus info and selects a PN, N_s, that is most likely to meet task deadline.
- Surplus information could be obsolete
  $\Rightarrow$ While sending the task to N_s, the overloaded PN sends RFB to lightly-loaded PNs, asking them to send bids to N_s.

Focused Addressing & Bidding — Continued

Sending PN: if $t_{bid} \leq t_{offload}$ then send RFB

$t_{bid} =$ time taken by RFB to reach its destinations + time taken by dests to respond with bids + time to xmit bid to N_s;

$t_{offload} =$ latest time N_s can offload a task onto a bidder w/o missing its deadline

= $D_i -$ current time $-$ task xfer time $- e_i$.

N_t**, PN receiving an RFB:** check if it can meet task deadline w/o compromising existing guarantees, i.e., if $t_{surplus} < e_i$ then no bid sent out else a bid ($t_{arr}, t_{surplus}$, and sojourn time at N_t) sent to N_s.

$t_{arr} =$ current time + time for bid to be received by N_s + time for N_s to make a decision + time to xfer the task + time taken by N_t to guarantee/reject

$t_{surplus} = D -$ current time $- t_{comp}$

$t_{comp} =$ time allotted to critical tasks in $[t_{arr}, D]$ + time needed in $[t_{arr}, D]$ to execute non-criticals already accepted + fraction of recently-accepted bids $\times$ time in $[t_{arr}, D]$ to honor pending bids.

Focused Addressing & Bidding — Continued

- *If* N_s cannot meet task deadline, it waits for a certain # of bids to arrive, or until a specified time has expired since receiving the task *then* computes

$$t_{est}(i) = t_{surplus}(i) \times \frac{D - \eta(i)}{D - t_{arr}(i)}$$

for each bidding PN N_i and chooses N_k with max t_{est}, where $\eta(i)$ is estimated task arrival time at N_i.

- Responding to RFB requires schedulability check, disrupting the original schedule:
  - a periodic task for sched check
  - set/reset a flag to indicate if PN has time for both sched check and meeting deadlines of accepted tasks

Problems with Adaptive LS

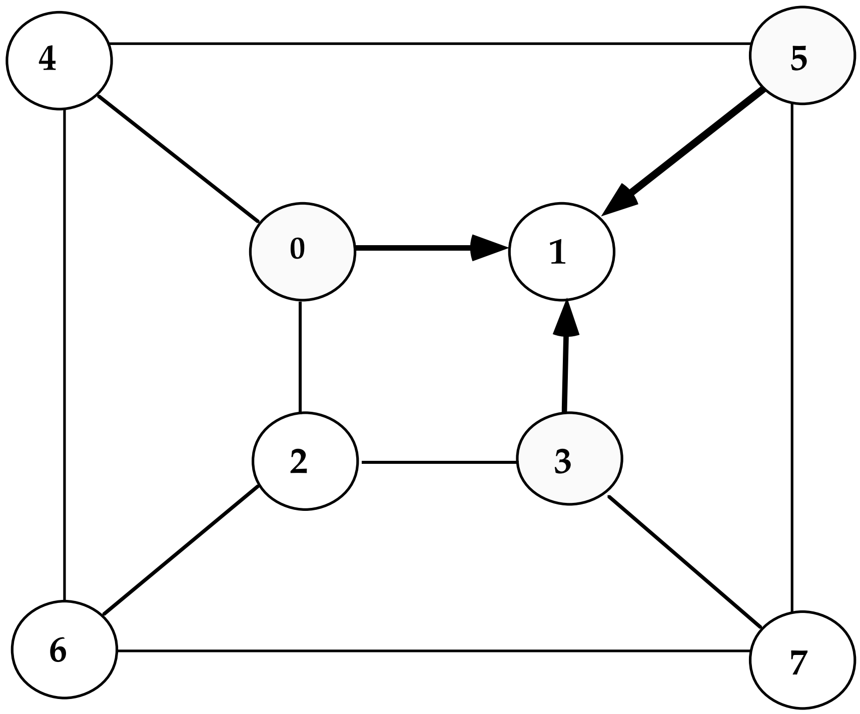
- Each PN needs up-to-date state information on other PNs
  $\Rightarrow$ Time and storage overheads associated with collection and update of state information

- Coordination problem

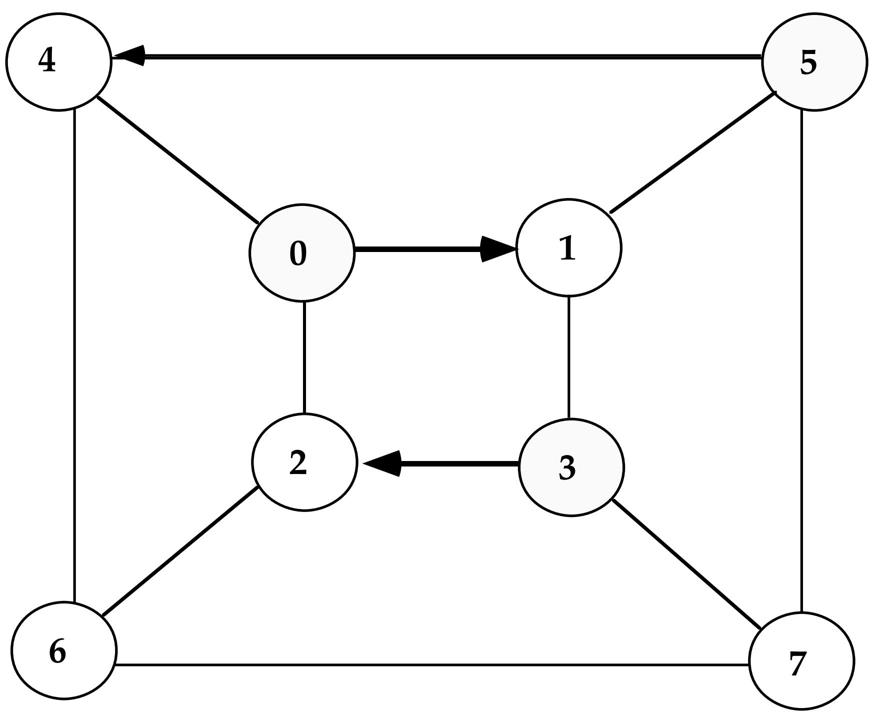
- Congestion problem

- Overhead of task transfer.

Coordination Problem

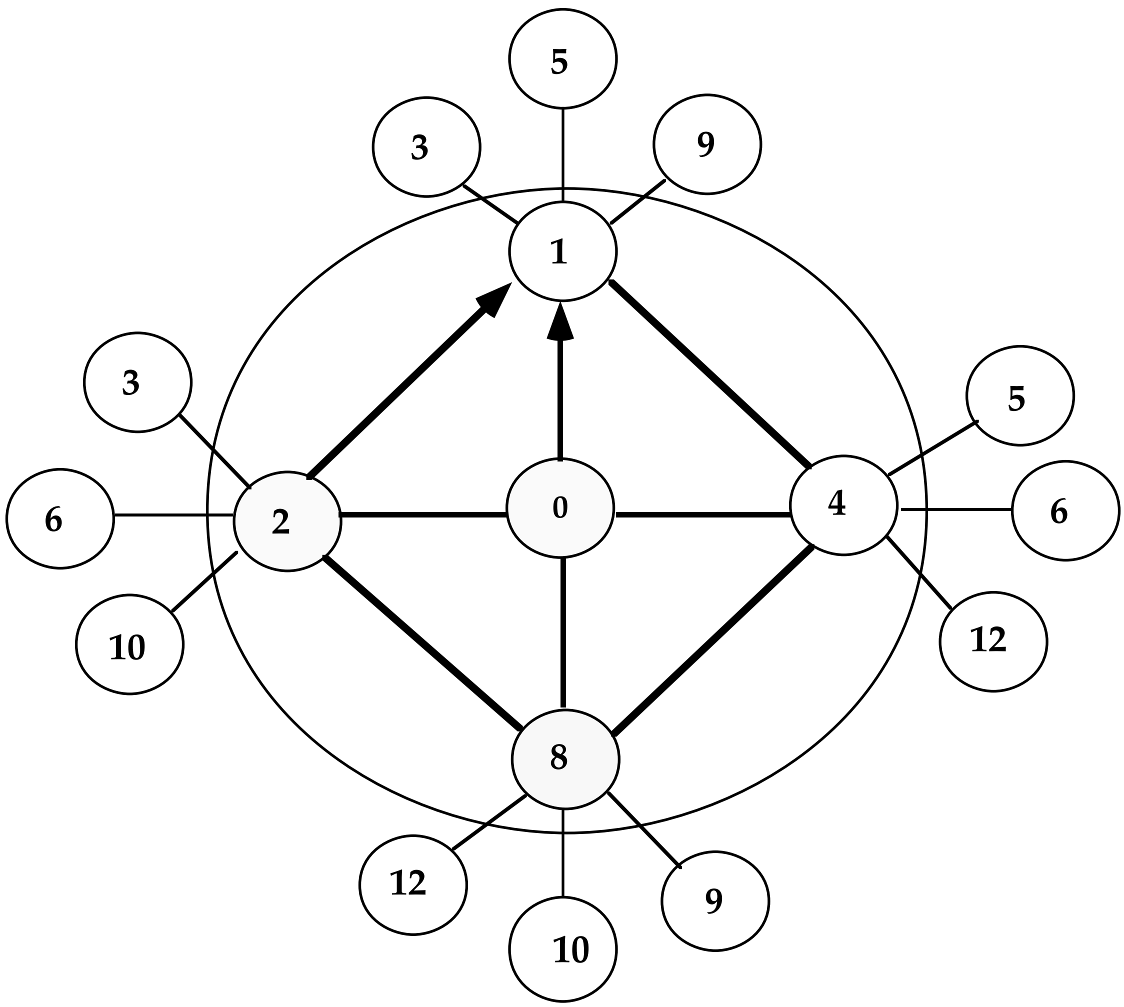


Bad coordination

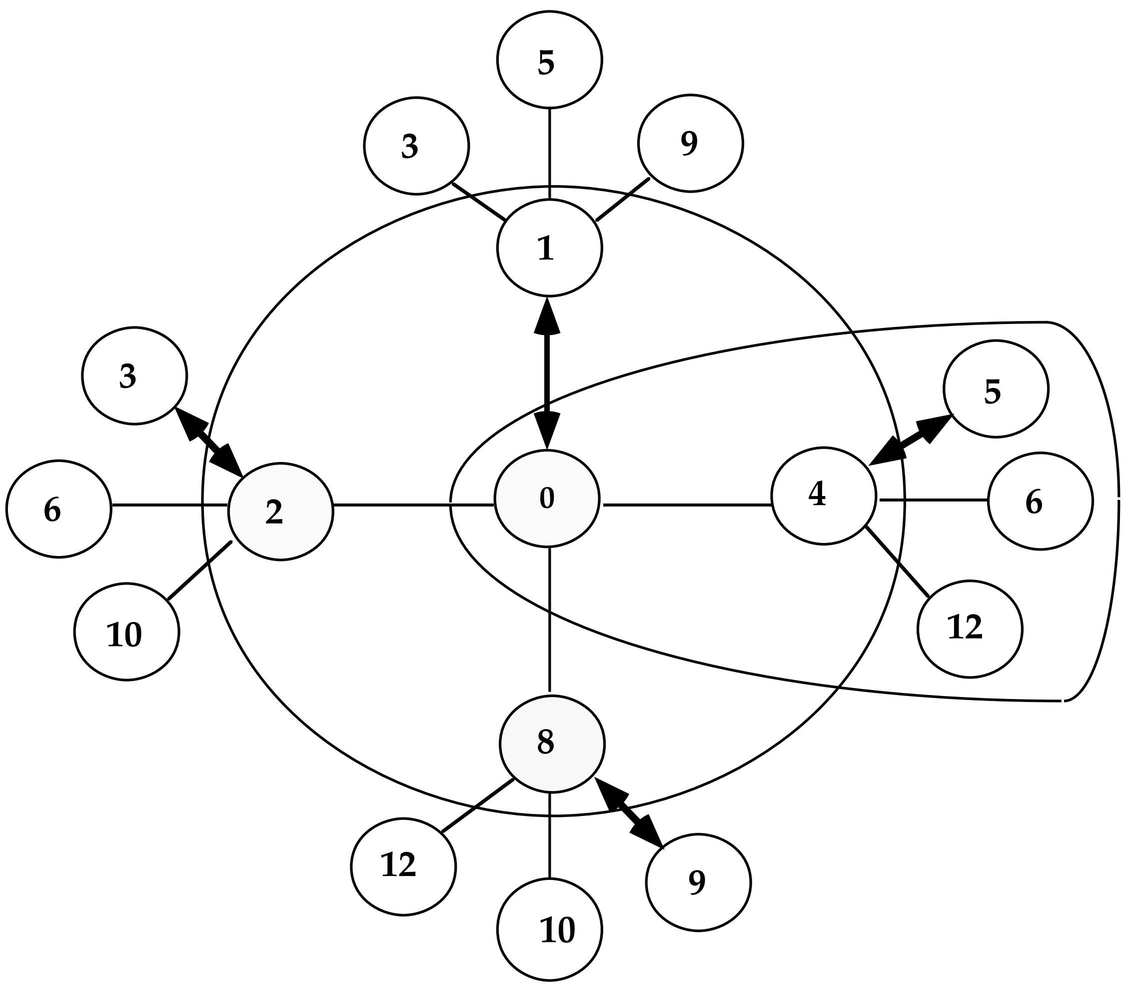


Good coordination

Congestion Problem



Solution to Congestion Problem



Collection of State Information

- Periodic exchange of state information
- State probing/bidding/drafting
- State-change multicast
  - How to set thresholds?
  - Scalability

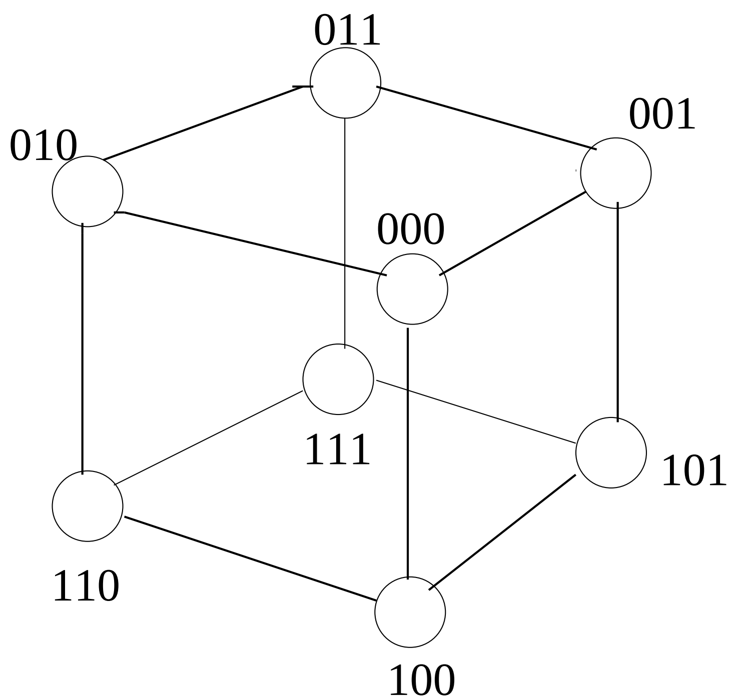
Goal and Approach

Goal: minimize P_{dyn} and alleviate the communication problem

Approach:

- Group PNs into overlapping *buddy sets*
- Associate each PN with a preferred list of potential receivers
- State collection with time-stamped region-change broadcasts
- Characterize the inconsistency between observed and true states
- Update loss-minimizing decisions with Bayesian analysis

Buddy Sets and Preferred Lists



PN	Preferred List						
0	1	2	4	5	6	3	7
1	0	3	5	4	7	2	6
2	3	0	6	7	4	1	5
3							
4							

Local Scheduler's Operations

1. When a new task arrives with exec time e_i and laxity D_i
   -
   -
2. When a state-region change broadcast is received
   -
   -
3. At every clock tick
   -
   -
4. At every T_p clock ticks,
   update probability distributions and table of loss-minimizing decisions

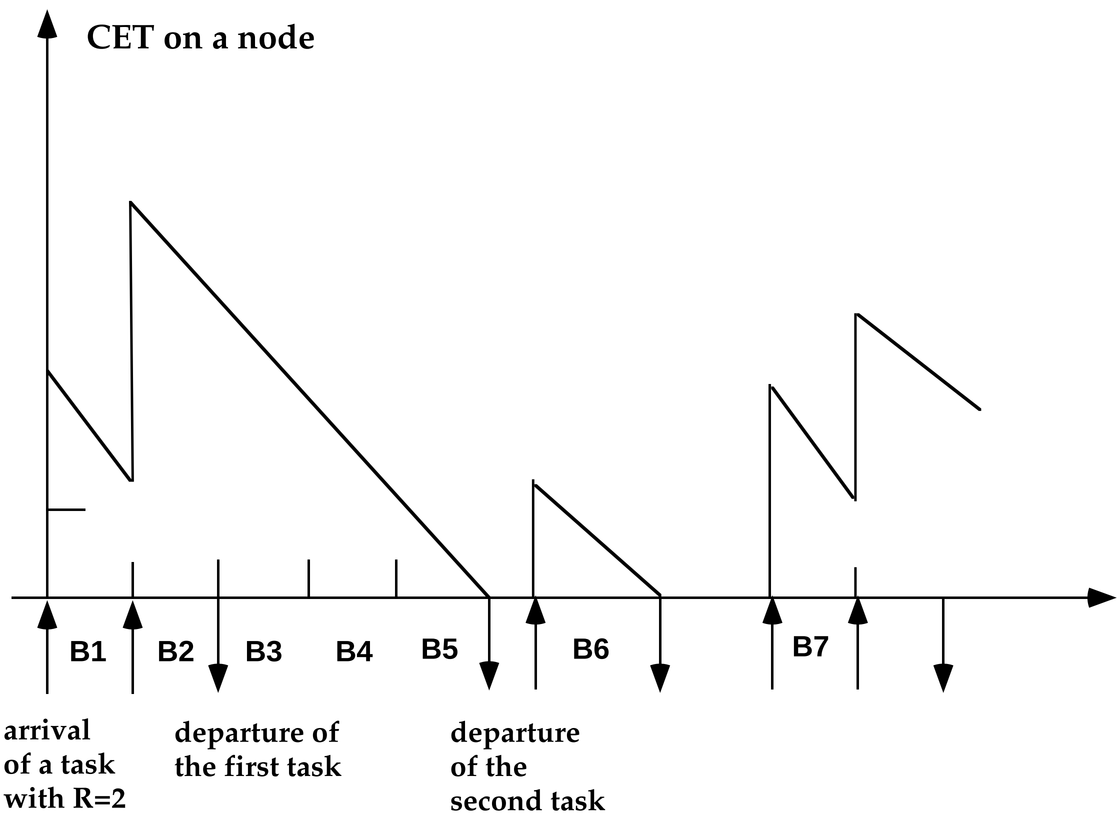
Merits of the Approach

- Significantly lower P_{dyn}
- Robust to the choice of tunable parameters in adaptive LS
- Insensitive to communication delays
- Modest computation and small task transfer delays

Analytic Modeling

- Used continuous time semi-Markov chains
- Approach:
  - Model CET evolution of a single node in isolation
  - Combine node-level models into a *system-level* model by characterizing task arrival and xfer processes.
- Two-step iterative algorithm for numerical solutions.

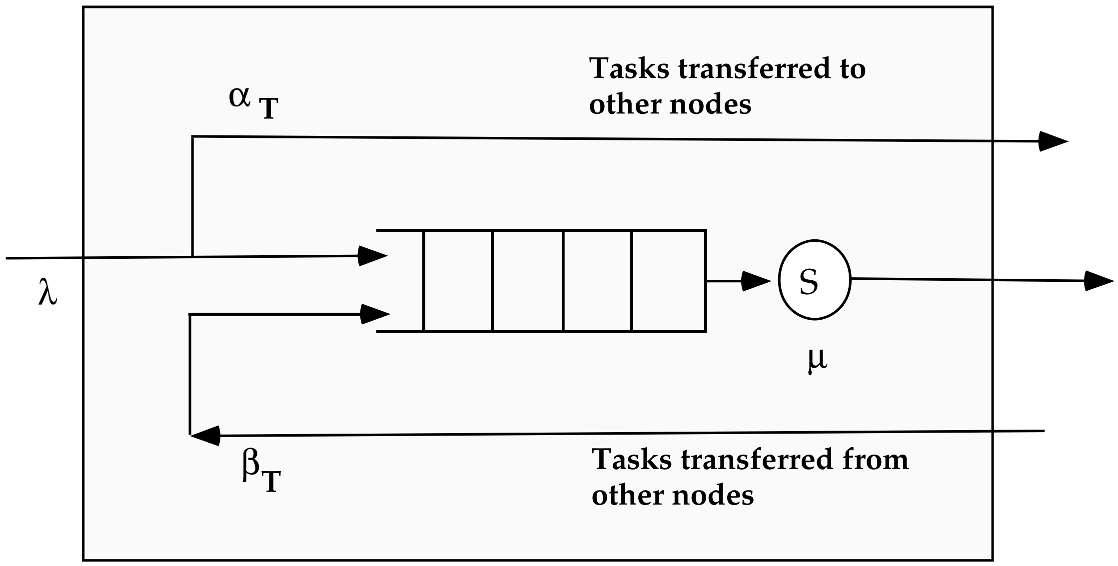
Evolution of Remaining CET



Where Bi is the node's i-th busy cycle

Kang Shin (kgshin@eecs.umich.edu)

Queueing Model on Each PN



where α_T is the rate of transferring tasks out of node given the remaining CET=T,

β_T is the rate of transferring tasks into a node given the remaining CET=T,

and $\lambda(T) = \lambda - \alpha_T + \beta_T$.

Advantages of Analytic Models

- Based on CET, not QL
- Allows both task laxity and execution time to be drawn from different distributions
- Accounts for all computation and communication overheads
- Can derive performance measures relevant to real-time applications.

Performance Evaluation

- CET distribution
- P_{dyn}
- Maximum system utilization
- Task xfer-out ratio
- Mean response time
- Sensitivity to communication delays
- Comparison between using CET and QL.