A Theory of Rate-Based Execution

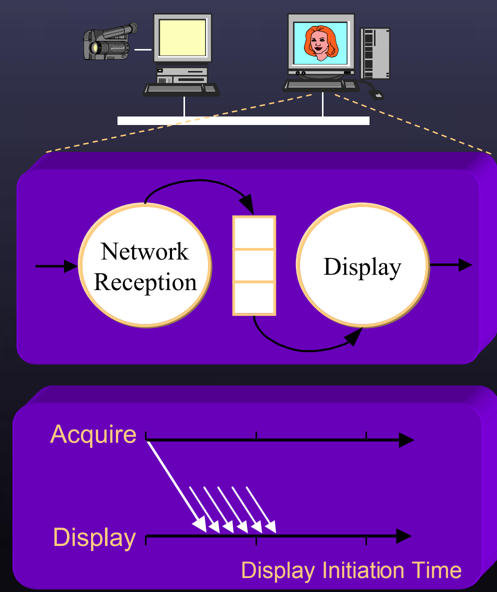
Kevin Jeffay
Department of Computer Science
University of North Carolina
at Chapel Hill
jeffay@cs.unc.edu

Steve Goddard
Computer Science & Engineering
University of Nebraska – Lincoln
goddard@cse.unl.edu

http://www.cs.unc.edu/Research/Dirt/

A Theory of Rate-Based Execution
What's wrong with the Liu & Layland model?

- Loosely speaking, nothing is periodic or sporadic in a distributed system
- The essential problem seems to be the requirement that the arrival process be somehow constrained



A Theory of Rate-Based Execution
Goals

- Extend the Liu and Layland theory of real-time processor scheduling to:
  - –Support notions of execution rate that are more general than periodic or sporadic execution
  - –Support integrated real-time device and application processing
  - –Support responsive non-real-time computing

Rate-Based Execution
Concept

- Schedule tasks at the *average rate* at which they are expected to be invoked
  - –Make buffering a first-class concept in the model
  - –Understand the fundamental relationships between feasibility, response time, and processing rate
- Develop a model of tasks wherein:
  - –Tasks complete execution before a well-defined deadline
  - –Tasks make progress at application-specified rates
  - –No constraints are placed on the external environment

Rate-Based Execution

Formal model

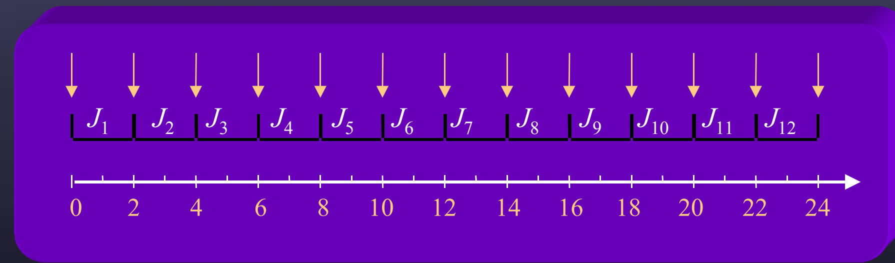
- Process make progress at the rate of processing x events every y time units, each event is processed within d time units
- For task i with rate specification (x_i, y_i, d_i), the j^{th} event for task i, arriving at time $t_{i,j}$, will be processed by time

$$D(i,j) = \begin{cases} t_{i,j} + d_i & \text{if } 1 \le j \le x_i \\ \text{MAX}(t_{i,j} + d_i,\ D(i, j-x_i)+y_i) & \text{if } j > x_i \end{cases}$$

  - Deadlines occur at least d time units after a job is released
  - Deadlines separated by at least y time units

Rate-Based Execution

Example: Periodic arrivals, periodic service

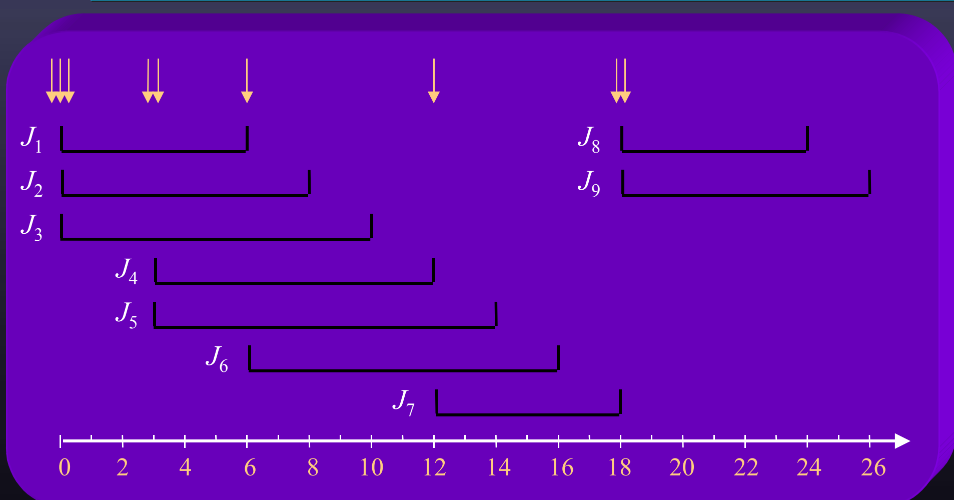


- Task with rate specification ($x = 1$, $y = 2$, $d = 2$)

$$D(i,j) = \begin{cases} t_{i,j} + d_i & \text{if } 1 \le j \le x_i \\ \text{MAX}(t_{i,j} + d_i,\ D(i, j-x_i)+y_i) & \text{if } j > x_i \end{cases}$$

Rate-Based Execution

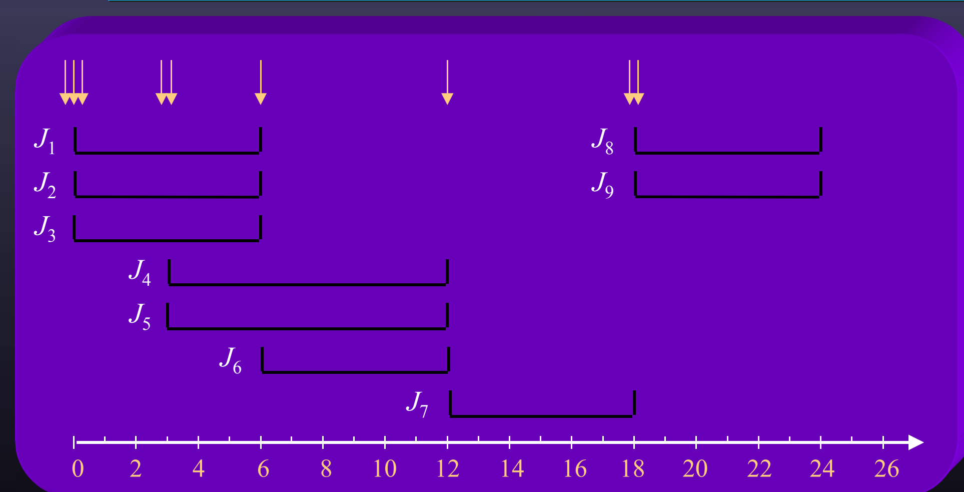
Bursty arrivals



- Task with rate specification ($x = 1$, $y = 2$, $d = 6$)

Rate-Based Execution

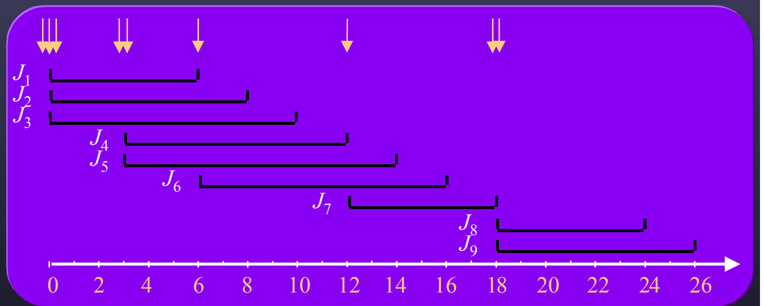
Bursty arrivals



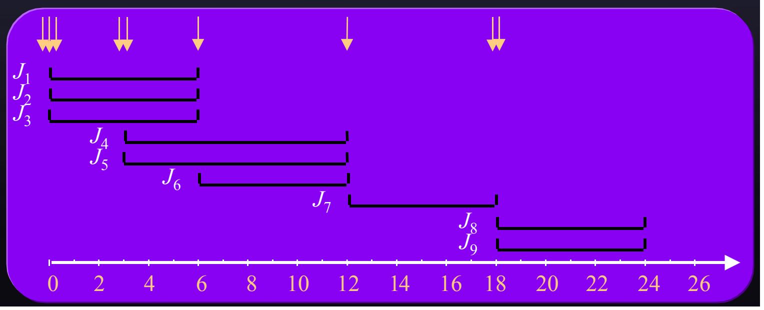
- Task with rate specification ($x = 3$, $y = 6$, $d = 6$)

Rate-Based Execution
Comparison of different rate specifications

Rate specification
($x = 1$, $y = 2$, $d = 6$)

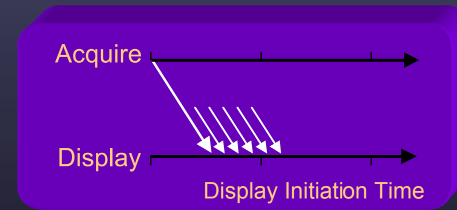


Rate specification
($x = 3$, $y = 6$, $d = 6$)

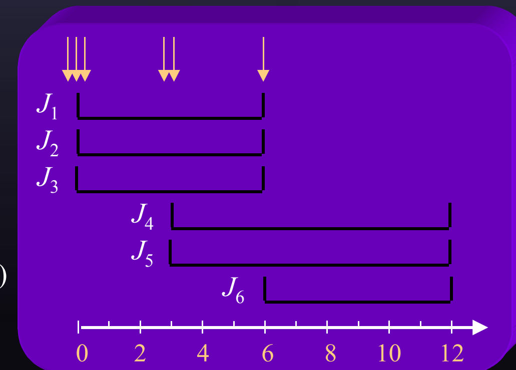


Using RBE Tasks
What problems do they solve?

- RBE tasks provide a more natural way of modeling inbound packet processing of fragmented messages



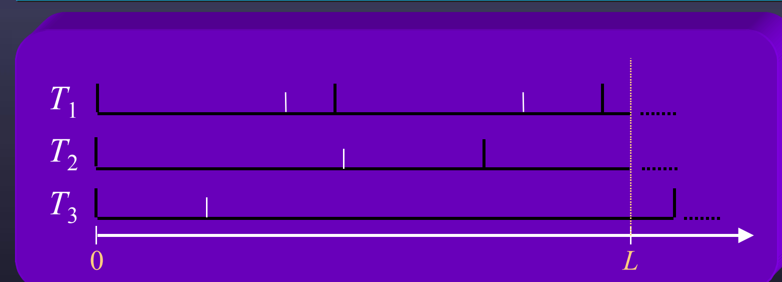
Rate specification
($x = 3$, $y = 6$, $d = 6$)



A Theory of Rate-Based Execution
Feasibility under preemption constraints

- Feasibility conditions for periodic and sporadic tasks, for all other known execution environments, also hold for *RBE* tasks
  - Feasibility under non-preemptive scheduling
  - Feasibility under scheduling with critical sections
  - Feasibility under scheduling with interrupt handlers
- Thus feasibility is not inherently a function of release times
  - Under deadline-driven scheduling, feasibility is a function of the implementation of a task set
  - Under static-priority scheduling, feasibility is a function of the behavior of the external environment

Feasibility of RBE Tasks
Feasibility under preemptive scheduling

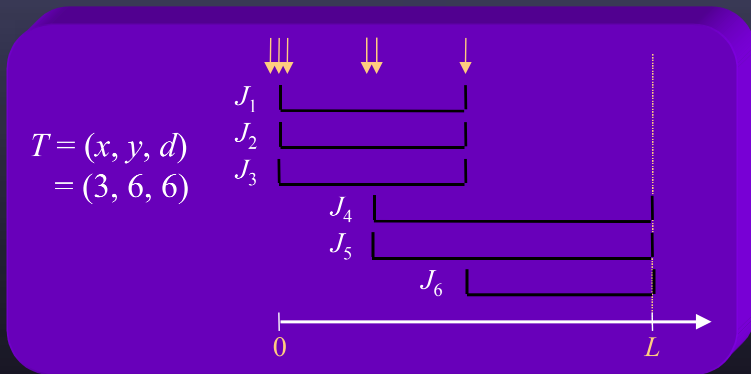


- Feasibility conditions of *RBE* tasks with rate specifications (x, y, c, d) are precisely the same as for periodic tasks

$$\forall L, L > 0: \; L \geq \sum_{i=1}^{n} \left\lfloor \frac{L - d_i + y_i}{y_i} \right\rfloor x_i c_i$$

A Theory of Rate-Based Execution

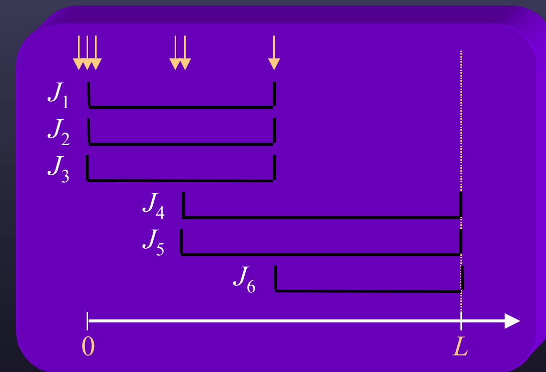
On the relationship to periodic tasks



- But can't an RBE task be modeled as x instances of a periodic task (with some appropriate precedence relationship between instances)?

A Theory of Rate-Based Execution

A corollary on static priority scheduling



$$L \geq \sum_{j=1}^{i} \left\lceil \frac{L}{p_j} \right\rceil c_j$$

- Under a static priority scheduling scheme, the processor demand in any interval can be unbounded
  - Thus event driven, rate-based execution is not possible under static priority scheduling schemes

A Theory of Rate-Based Execution

Summary

- Traditional Liu & Layland theory is not directly applicable to distributed real-time systems
- The theory of scheduling periodic & sporadic tasks applies verbatim to RBE tasks
  - Polynomial & pseudo-polynomial time schedulability conditions exist for
    - » Preemptive scheduling
    - » Non-preemptive scheduling
    - » Scheduling with interrupt handlers
    - » Scheduling with critical sections
  - The *earliest-deadline-first* scheduling algorithm is optimal

A Theory of Rate-Based Execution

Summary

- The feasibility of a set of "periodic tasks" was never inherently a function of the periodic arrival requirement
  - The only requirement is that exist a minimal separation between deadlines
- But if static priority scheduling methods are employed then (in the worst case) periodic arrivals are required
  - Static priority methods require a well-behaved external environment
  - Deadline methods require a well-behaved operating system