Multiprocessor Scheduling II: Global Scheduling

Prof. Dr. Jian-Jia Chen

LS 12, TU Dortmund

28, June, 2016

Global Scheduling

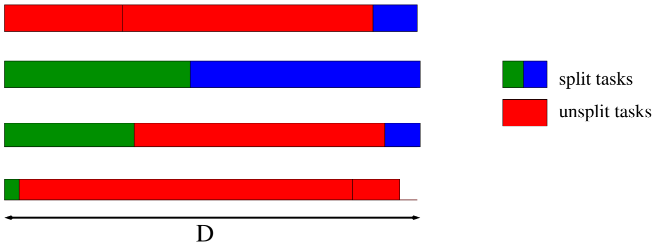
- We will only focus on identical multiprocessors in this module.
- The system has a global queue.
- A job can be migrated to any processor.
- Priority-based global scheduling:
  - Among the jobs in the global queue, the M highest priority jobs are chosen to be executed on M processors.
  - Task migration here is assumed no overhead.
  - Global-EDF: When a job finishes or arrives to the global queue, the M jobs in the queue with the shortest absolute deadlines are chosen to be executed on M processors.
  - Global-FP, Global-DM, Global-RM: When a job finishes or arrives to the global queue, the M jobs in the queue with the highest priorities (defined by fixed-priority ordering, deadline-monotonic strategy, or rate-monotonic strategy) are chosen to be executed on M processors.
- Pfair scheduling, and the variances (not discussed in this lecture).

Good News for Global Scheduling

- McNaughton's wrap-around rule for $P|pmtn|C_{\max}$ on M processors (historically, task migration is also called task preemption in the literature)
  - Compute $C_{\max}$ as $\max\{\max_{\tau_i \in \mathcal{T}} C_i, \frac{\sum_{\tau_i \in \mathcal{T}} C_i}{M}\}$
    - Assign the tasks according to any order from time 0 to $C_{\max}$
    - If a task's processing exceeds $C_{\max}$, the task is migrated to a new processor from time 0
    - Repeat the assignment of tasks until all the tasks are assigned
  - The resulting schedule minimizes $C_{\max}$

R. McNaughton. Scheduling with deadlines and loss functions. Management Science, 6:1-12, 1959.

McNaughton's Algorithm: Example



Weakness of Partitioned Scheduling

- Restricting a task on a processor reduces the schedulability
- Restricting a task on a processor makes the problem $\mathcal{N}P$-hard
- The $\mathcal{N}P$-completeness for EDF does no hold any more if the migration has *no overhead*.
  - Proportionate Fair (pfair) algorithm introduced by Baruah et al. provides an optimal utilization bound for schedulibility
  - A task set with implicit deadlines is schedulable on M identical processors if the total utilization of the task set is no more than M.
  - The idea is to divide the time line into quanta, and execute tasks proportionally in each quanta.
  - It has very high overhead.
  - There are several variances to reduce the overhead.

Sanjoy K. Baruah, N. K. Cohen, C. Greg Plaxton, Donald A. Varvel: Proportionate Progress: A Notion of Fairness in Resource Allocation. Algorithmica 15(6): 600-625 (1996)

Bad News for Global Scheduling

For Global-EDF or Global-RM, the least upper bound for schedulability analysis is at most 1.

Input:

$M + 1$ tasks:

- One heavy task τ_k: $D_k = T_k = C_k$
- M light tasks τ_is: $C_i = \epsilon$ and $D_i = T_i = C_k - \epsilon$, in which ϵ is a positive number, very close to 0.

Sudarshan K. Dhall, C. L. Liu, On a Real-Time Scheduling Problem, OPERATIONS RESEARCH Vol. 26, No. 1, January-February 1978, pp. 127-140.

Bad News for Global Scheduling

For Global-EDF or Global-RM, the least upper bound for schedulability analysis is at most 1.

Input:

$M+1$ tasks:

- One heavy task τ_k: $D_k = T_k = C_k$
- M light tasks τ_is: $C_i = \epsilon$ and $D_i = T_i = C_k - \epsilon$, in which ϵ is a positive number, very close to 0.

Result:

The M light tasks (with higher priority than the heavy task) will be scheduled on M processors. The heavy task misses the deadline even when the utilization is $1 + M\epsilon$.

Sudarshan K. Dhall, C. L. Liu, On a Real-Time Scheduling Problem, OPERATIONS RESEARCH Vol. 26, No. 1, January-February 1978, pp. 127-140.

Gold Approach: Resource Augmentation

- The bad news on the least upper bound was very important in 80's, since the research in this direction suffered from the so called "Dhall effect".
- With resource augmentation, by Phillips et al., the "Dhall effect" disappears
  - For Global-EDF, the resource augmentation factor by "speeding up" is $2 - \frac{1}{M}$.
  - That is, if a feasible schedule exists on M processors, applying Global-EDF is also feasible on M processors by speeding up the execution speed with $2 - \frac{1}{M}$.
  - We will focus on schedulability test here first (for the first two parts) and the resource augmentation at the end.

Cynthia A. Phillips, Clifford Stein, Eric Torng, Joel Wein: Optimal Time-Critical Scheduling via Resource Augmentation. STOC 1997: 140-149

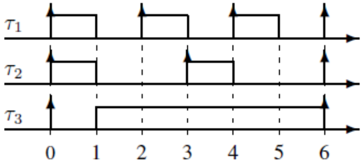
Articles for This Module

- Sanjoy K. Baruah: Techniques for Multiprocessor Global Schedulability Analysis. RTSS 2007: 119-128 (*First part*)
- Nan Guan, Martin Stigge, Wang Yi, Ge Yu: New Response Time Bounds for Fixed Priority Multiprocessor Scheduling. IEEE Real-Time Systems Symposium 2009: 387-397 (*Second part*)
- Vincenzo Bonifaci, Alberto Marchetti-Spaccamela, Sebastian Stiller, Andreas Wiese: Feasibility Analysis in the Sporadic DAG Task Model. ECRTS 2013: 225-233 (*Appendix*)
  - Vincenzo Bonifaci, Alberto Marchetti-Spaccamela, Sebastian Stiller: A Constant-Approximate Feasibility Test for Multiprocessor Real-Time Scheduling. ESA 2008: 210-221

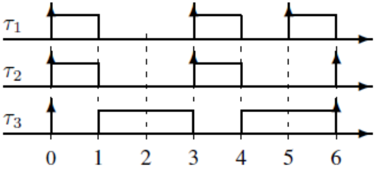
We will mainly focus on task sets with constrained deadlines.

Critical Instants?

- The analysis for uniprocessor scheduling is based on the gold critical instant theorem.
- Synchronous release of higher-priority tasks and as early as possible for the following jobs do not lead to the critical instant for global multiprocessor scheduling
  - Suppose that there two identical processors and 3 tasks: (C_i, D_i, T_i) are $\tau_1 = (1, 2, 2), \tau_2 = (1, 3, 3), \tau_3 = (5, 6, 6)$

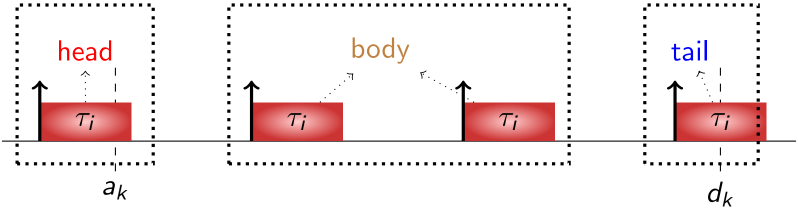


Feasible for τ_3.



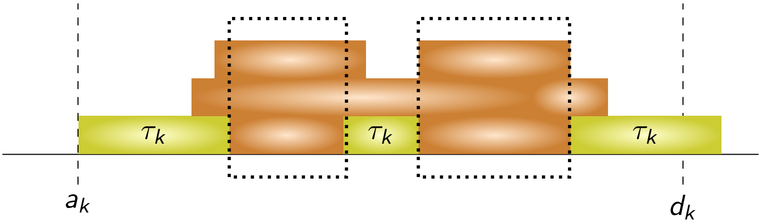
Infeasible for τ_3.

Identifying Interference



- Problem window (interval) is defined in $[a_k, d_k)$.
- The jobs of task τ_i in the problem window can be categorized into three types:
  - Head job (at most one): some computation demand is *carried in* to the problem window for a job arrival before a_k.
  - Body jobs: the computation demand has to be done in the problem window.
  - Tail job (at most one): some computation demand can be *carried out* from the problem window.

Necessary Condition for Deadline Misses



- If τ_k misses the deadline at d_k, there must be at least $D_k - C_k$ units of time in which all M processors are executing other higher-priority jobs.
- Definition: *demand* $W(\Delta)$ in a time interval with length Δ is the total amount of computation that needs to be completed within the interval.
- If τ_k misses its deadline at time d_k, then

$$W(D_k) > M(D_k - C_k) + C_k$$

Outline

Introduction

Schedulability Analysis: Global EDF

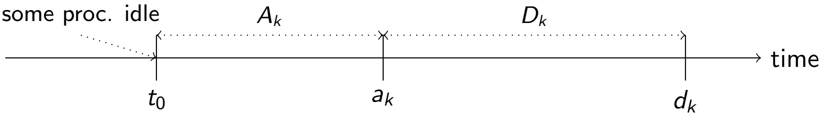
Schedulability Analysis: Global RM

Appendix: Augmentation Factor

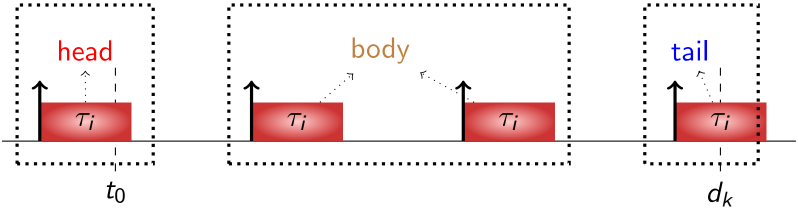
Baruah's Approach

For contrapositive, assume that a job of task τ_k misses its absolute deadline at time d_k with release time a_k.

- Bound the carry-in computation demand more precisely.
- Let t_0 be the earliest time instant such that the system executes jobs on M processors from t_0 to a_k.
    - The ready queue before t_0 is with less than M jobs.
    - The ready queue has at least M jobs in time interval $[t_0, a_k)$.
- Let $\mathcal{I}$ be the set of intervals in $[t_0, d_k)$, in which all the M processors are executing. By considering the worst cases, the job of task τ_k arriving at time a_k is not executed at all in $\mathcal{I}$.
- Let A_k be $a_k - t_0$.



Identifying Interference



- Problem window (interval) is defined in $[t_0, d_k)$.
- The jobs of task τ_i in the problem window can be categorized into three types:
  - Head job (at most one): some computation demand is *carried in* to the problem window for a job arrival before a_k.
  - Body jobs: the computation demand has to be done in the problem window.
  - Tail job (at most one): some computation demand can be *carried out* from the problem window.

Necessary Condition for Deadline Misses

- Let $w_i(\mathcal{I})$ be the demand executed in the set $\mathcal{I}$ of time intervals for task τ_i. The necessary condition for τ_k to miss its deadline is
$$\sum_{i=1}^{N} w_i(\mathcal{I}) > M(A_k + D_k - C_k).$$
- Let's consider two types of interferences in $w_i(\mathcal{I})$.
  - Type 1: tasks that are not executing at time t_0. There will be no carry-in demand at time t_0.
  - Type 2: tasks that are executing at time t_0. There might be carry-in demand at time t_0.

Interference Type 1: No Carry-In at Time t_0

- Case 1: $i \neq k$
  - The demand of τ_i to be done in the time intervals in $\mathcal{I}$ is at most
$$\min\left\{dbf(\tau_i, A_k + D_k), A_k + D_k - C_k\right\}.$$
- Case 2: i is k
  - The demand of τ_k to be done in the time intervals in $\mathcal{I}$ is at most
$$\min\left\{dbf(\tau_i, A_k + D_k) - C_k, A_k\right\}.$$
  - Specifically, we need to remove the job that arrives at a_k since its execution is not counted as part of $\mathcal{I}$.

Therefore,

$$w_i^1(\mathcal{I}) =^{\text{def}} \begin{cases} \min\left\{dbf(\tau_i, A_k + D_k), A_k + D_k - C_k\right\} & \text{if } i \neq k \\ \min\left\{dbf(\tau_i, A_k + D_k) - C_k, A_k\right\} & \text{if } i = k. \end{cases}$$

Interference Type 2: With Carry-In at Time t_0

- Case 1: $i \neq k$
  - The demand of τ_i to be done in the time intervals in $\mathcal{I}$ is at most
$$\min\left\{dbf^{\dagger}(\tau_i, A_k + D_k), A_k + D_k - C_k\right\},$$
where $dbf^{\dagger}(\tau_i, \delta)$ is $\left\lfloor \frac{\delta}{T_i} \right\rfloor C_i + \min\{C_i, \delta \mod T_i\}$.
- Case 2: i is k
  - The demand of τ_k to be done in the time intervals in $\mathcal{I}$ is at most
$$\min\left\{dbf^{\dagger}(\tau_i, A_k + D_k) - C_k, A_k\right\}.$$

Therefore,

$$w_i^2(\mathcal{I}) =^{\text{def}} \begin{cases} \min\left\{dbf^{\dagger}(\tau_i, A_k + D_k), A_k + D_k - C_k\right\} & \text{if } i \neq k \\ \min\left\{dbf^{\dagger}(\tau_i, A_k + D_k) - C_k, A_k\right\} & \text{if } i = k. \end{cases}$$

Putting Together

- Let $w_i^{diff}(\mathcal{I})$ be $w_i^2(\mathcal{I}) - w_i^1(\mathcal{I})$.
- The necessary condition for τ_k to miss its deadline becomes

$$\sum_{i=1}^{N} w_i^1(\mathcal{I}) + \sum_{M-1 \text{ largest}} w_i^{diff}(\mathcal{I}) > M(A_k + D_k - C_k).$$

Putting Together

- Let $w_i^{diff}(\mathcal{I})$ be $w_i^2(\mathcal{I}) - w_i^1(\mathcal{I})$.
- The necessary condition for τ_k to miss its deadline becomes

$$\sum_{i=1}^{N} w_i^1(\mathcal{I}) + \sum_{M-1 \text{ largest}} w_i^{diff}(\mathcal{I}) > M(A_k + D_k - C_k).$$

Theorem

A task set is schedulable under Global-EDF if for every task τ_k and for all $A_k \geq 0$

$$\sum_{i=1}^{N} w_i^1(\mathcal{I}) + \sum_{M-1 \text{ largest}} w_i^{diff}(\mathcal{I}) \leq M(A_k + D_k - C_k).$$

Outline

Introduction

Schedulability Analysis: Global EDF

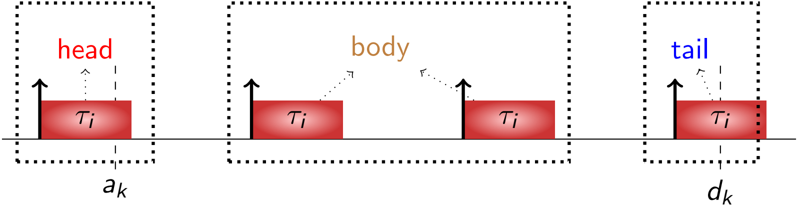
Schedulability Analysis: Global RM

Appendix: Augmentation Factor

Strategy

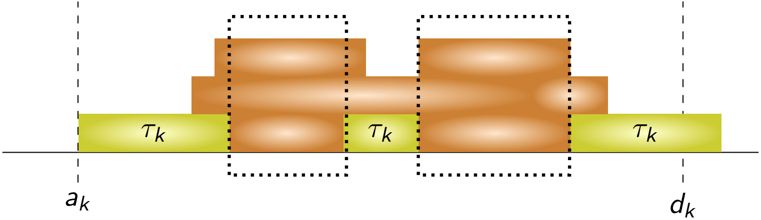
- We focus on Global RM in this part. This basically implies that $D_i = T_i$ for every task τ_i. Some of the strategies can be applied to Global DM.
- We are looking for the necessary condition such that a deadline miss happens.
- Suppose that Global scheduling fails by missing the deadline d_k of task τ_k, which is the first instant with deadline missing.
- The job with the earliest deadline miss arrives at time a_k, in which $T_k = D_k = d_k - a_k$.

Identifying Interference



- Problem window (interval) is defined in $[a_k, d_k)$.
- The jobs of task τ_i in the problem window can be categorized into three types:
  - Head job (at most one): some computation demand is *carried in* to the problem window for a job arrival before a_k.
  - Body jobs: the computation demand has to be done in the problem window.
  - Tail job (at most one): some computation demand can be *carried out* from the problem window.

Necessary Condition for Deadline Misses



- If τ_k misses the deadline at d_k, there must be at least $D_k - C_k$ units of time in which all M processors are executing other higher-priority jobs.
- Definition: *demand* $W(\Delta)$ in a time interval with length Δ is the total amount of computation that needs to be completed within the interval.
- If τ_k misses its deadline at time d_k, then

$$W(D_k) > M(D_k - C_k) + C_k$$

Bound Carry-In Interference

Theodore P. Baker: Multiprocessor EDF and Deadline Monotonic Schedulability Analysis. RTSS 2003: 120-129 (earlier results)

- Baker's approach tries to bound the carry-in interference by extending the busy-interval to the left hand side while satisfying some load condition.
- This step is called *downward extension of an interval* for global RM. For your reference, the procedures are included in the appendix.
- Here, I am presenting a very simple strategy to analyze the schedulability for global RM.
- This is based on the schedulability analysis we did earlier in the utilization bound analysis for global RM.

A Pessimistic Sufficient Test for Global RM

For all $0 < t \leq T_k$

$$W_k(t) = \sum_{i=1}^{k-1} (\left\lceil \frac{t}{T_i} \right\rceil - 1) C_i + 2C_i.$$

This implies that we just greedily take a head job immediately. Clearly, lower-priority jobs have no effect for the unschedulability or schedulability.

Theorem

A system $\mathcal{T}$ of periodic, independent, preemptable tasks is schedulable by Global-RM on M processors if

$$\forall \tau_k \in \mathcal{T}\ \exists t \text{ with } 0 < t \leq T_k \ \ and \ \ C_k + \frac{W_k(t)}{M} \leq t$$

holds. This condition is NOT a necessary condition.

Recall k-Point Effective Schedulability Test: k^2U

Suppose that $\{t_1, t_2, \ldots t_k\}$ are given.

Definition

A k-point effective schedulability test is a sufficient test by verifying the existence of $t_j \in \{t_1, t_2, \ldots t_k\}$ with $t_1 \leq t_2 \leq \cdots \leq t_k$ such that

$$C_k + \sum_{i=1}^{k-1} \alpha_i t_i U_i + \sum_{i=1}^{j-1} \beta_i t_i U_i \leq t_j, \quad (1)$$

where $C_k > 0$, $\alpha_i > 0$, $U_i > 0$, and $\beta_i > 0$ are dependent upon the setting of the task models and task τ_i.

Lemma

[Lemma 1] For a given k-point effective schedulability test of a scheduling algorithm, in which $0 < \alpha_i \leq \alpha$, and $0 < \beta_i \leq \beta$ for any $i = 1, 2, \ldots, k-1$, $0 < t_k$, task τ_k is schedulable by the scheduling algorithm if the following condition holds

$$\frac{C_k}{t_k} \leq \frac{\frac{\alpha}{\beta} + 1}{\prod_{j=1}^{k-1}(\beta U_j + 1)} - \frac{\alpha}{\beta}. \quad (2)$$

Constrained-Deadline: Schedulability Test for TDA

This is basically very similar to TDA with a minor difference by dividing the higher-priority workload by M. Testing the schedulability condition of task τ_k can be done by using the same strategy used in the k^2U framework.

A simple exercise will lead you to

- $0 < \alpha_i \leq \frac{2}{M}$ and $0 < \beta_i \leq \frac{1}{M}$ for $i = 1, 2, \ldots, k-1$ when testing task τ_k.

Hyperbolic Bound

The task set is schedulable under Global RM if

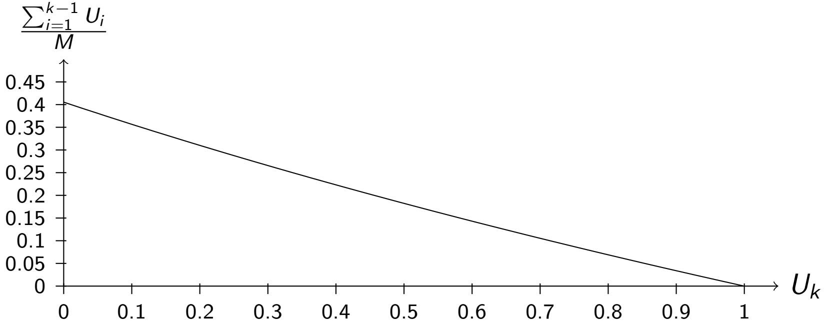
$$\forall k, \qquad (2 + U_k)\,\Pi_{i=1}^{k-1}(U_i/M + 1) \leq 3.$$

Hyperbolic Bound

The task set is schedulable under Global RM if

$$\forall k, \qquad (2 + U_k)\,\Pi_{i=1}^{k-1}(U_i/M + 1) \leq 3.$$

The following figure is the hyperbolic bound for the extreme case when k goes to ∞, in which $(2 + U_k)e^{\frac{\sum_{i=1}^{k-1} U_i}{M}} \leq 3$



Capacity Augmentation Bound

Given a task set $\mathcal{T}$ with total utilization of $U_{\sum}$, a scheduling algorithm $\mathcal{A}$ with capacity augmentation bound b can always schedule this task set on M processors of speed b as long as $\mathcal{T}$ satisfies the following conditions:

$$\text{Utilization does not exceed total cores, } \sum_{\tau_i \in \mathcal{T}} U_i \leq M \tag{3}$$

$$\text{For each task } \tau_i \in \mathcal{T}, \text{the utilization } U_i \leq 1 \tag{4}$$

Capacity Augmentation Bound

Given a task set $\mathcal{T}$ with total utilization of $U_{\sum}$, a scheduling algorithm $\mathcal{A}$ with capacity augmentation bound b can always schedule this task set on M processors of speed b as long as $\mathcal{T}$ satisfies the following conditions:

$$\text{Utilization does not exceed total cores, } \sum_{\tau_i \in \mathcal{T}} U_i \leq M \tag{3}$$

$$\text{For each task } \tau_i \in \mathcal{T}, \text{the utilization } U_i \leq 1 \tag{4}$$

This means that the algorithm guarantees the schedulability if the following conditions are satisfied:

$$\text{Utilization does not exceed total cores, } \sum_{\tau_i \in \mathcal{T}} U_i \leq \frac{M}{b} \tag{5}$$

$$\text{For each task } \tau_i \in \mathcal{T}, \text{the utilization } U_i \leq \frac{1}{b} \tag{6}$$

Capacity Augmentation Bound of Global RM

The task set is schedulable under Global RM if

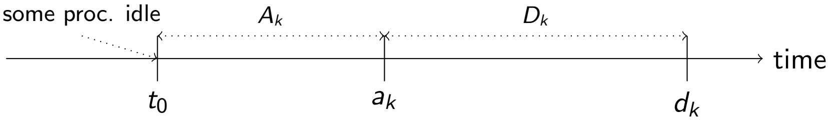
$$\forall k, (2 + U_k)\,\Pi_{i=1}^{k-1}(U_i/M + 1) \leq 3. \tag{7}$$

$$\Rightarrow \left(2 + \frac{1}{b}\right)\left(\frac{1}{(k-1)b} + 1\right)^{k-1} \leq 3. \tag{8}$$

$$\Rightarrow \left(2 + \frac{1}{b}\right) e^{1/b} \leq 3. \tag{9}$$

Again, we use the worst cases by setting all the tasks with the same utilization as we did in the analysis for uniprocessor systems. This concludes that $b \geq 3.6215$ enforces the above inequality.

Guan's Extension



- Baruah's analysis for Global EDF in fact also works with Global RM for constrained-deadline task systems.
- In the time interval from t_0 to d_k, we only have to consider $M - 1$ tasks with carry-in jobs.

Bounded Carry-In

We can define two different time-demand functions, depending on whether task τ_i is with a carry-in job or not:

$$w_i^2(t) = \begin{cases} C_i & 0 < t < C_i \\ C_i + \left\lceil \frac{t-C_i}{T_i} \right\rceil C_i & otherwise, \end{cases} \tag{10}$$

and

$$w_i^1(t) = \left\lceil \frac{t}{T_i} \right\rceil C_i. \tag{11}$$

We can further over-approximate $w_i^2(t)$, since $w_i^2(t) \leq w_i^1(t) + C_i$. Therefore, a sufficient schedulability test for testing task τ_k with $k > M$ for global RM is to verify whether

$$\exists 0 < t \leq T_k, C_k + \frac{(\sum_{\tau_i \in \mathbf{T}'} C_i) + (\sum_{i=1}^{k-1} w_i^1(t))}{M} \leq t, \tag{12}$$

for all $\mathbf{T}' \subseteq hp(\tau_k)$ with $|\mathbf{T}'| = M - 1$.

Adopting k^2U

There are two ways to use k^2U.

- Case 1: we consider that C_i for task τ_i is known.
  - We simply have to put the $M-1$ higher-priority tasks with the largest execution times into $\mathbf{T}'$.
  - This can be imagined as if we increase the execution time of task τ_k from C_k to $C'_k = C_k + \frac{\sum_{\tau_i \in \mathbf{T}'} C_i}{M}$.
  - Therefore, we still have $0 < \alpha_i \leq \frac{1}{M}$ and $0 < \beta_i \leq \frac{1}{M}$ for $i = 1, 2, \ldots, k-1$
- Case 2: only the task utilizations are given.
  - We need to figure out $\mathbf{T}'$
  - For a higher-priority task τ_i in $\mathbf{T}'$, its α_i is upper-bounded by $\frac{2}{M}$
  - For a higer-priority task τ_i not in $\mathbf{T}'$, its α_i is upper-bounded by $\frac{1}{M}$
  - This is a more complicated case. I am not going to discuss this.

Adopting k^2U: Case 1

Theorem

Task τ_k in a sporadic implicit-deadline task system is schedulable by global RM on M processors if

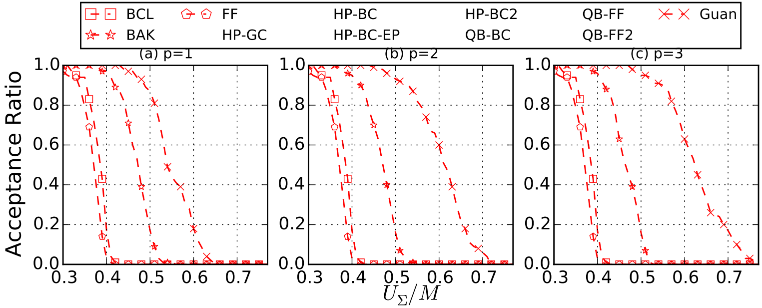
$$\left(\frac{C'_k}{T_k}+1\right)\prod_{i=1}^{k-1}\left(\frac{U_i}{M}+1\right)\leq 2, \tag{13}$$

or

$$\sum_{i=1}^{k-1}\frac{U_i}{M}\leq \ln\left(\frac{2}{\frac{C'_k}{T_k}+2}\right), \tag{14}$$

where $C'_k = C_k + \frac{\sum_{\tau_i \in \mathbf{T}'} C_i}{M}$.

A Brief Look of the Evaluation Results



Chen, Huang, Liu, 2015

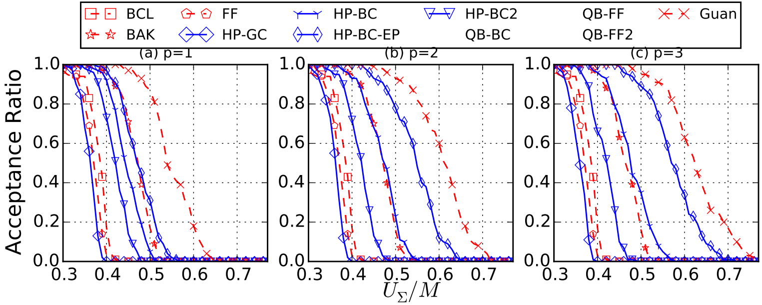
Red curves: existing results

- BCL: Bertonga et al. 2005; BAK: Baker 2003; FF: Baruah et al. 2010; Guan: Guan et al. 2009

Blue curves: results by adopting k^2U

Black curves: results by adopting k^2Q

A Brief Look of the Evaluation Results



Chen, Huang, Liu, 2015

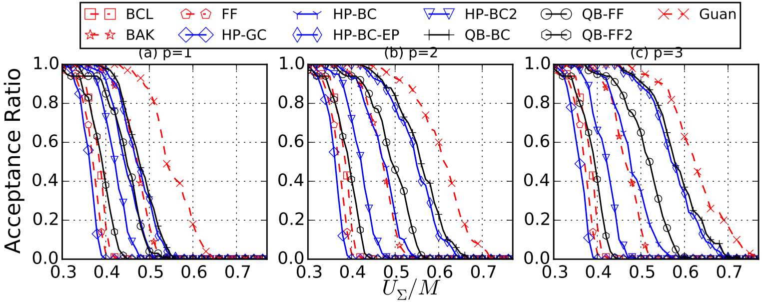
Red curves: existing results

- BCL: Bertonga et al. 2005; BAK: Baker 2003; FF: Baruah et al. 2010; Guan: Guan et al. 2009

Blue curves: results by adopting k^2U

Black curves: results by adopting k^2Q

A Brief Look of the Evaluation Results



Chen, Huang, Liu, 2015

Red curves: existing results

- BCL: Bertonga et al. 2005; BAK: Baker 2003; FF: Baruah et al. 2010; Guan: Guan et al. 2009

Blue curves: results by adopting k^2U

Black curves: results by adopting k^2Q

Summary of Existing Results

Regarding to speedup factors

	implicit deadlines	constrained deadlines	arbitrary deadlines
Global EDF	$2 - \frac{1}{M}$ (Bonifaci et al. 2008)		
Global DM	$3-\frac{1}{M}$ (Bertogna et al. 2005) $\frac{3+\sqrt{7}}{2} \approx 2.823$ (Chen et al. 2015, k^2Q)	$3 - \frac{1}{M}$ (Baruah et al. 2010) 3 (Chen et al. 2015, k^2Q)	$4 - \frac{1}{M}$ (Baruah/Fisher 2008)

Remarks on Global Scheduling

- pfair: Optimal for implicit-deadline task systems, but with very high overhead. Not introduced in the lecture.
- Global EDF/RM: lower online scheduling overhead, compared to pfair, but not optimal.
- A tradeoff: less management overhead (less task migrations) without losing the optimality.
  - Paul Regnier, George Lima, Ernesto Massa, Greg Levin, Scott A. Brandt: RUN: Optimal Multiprocessor Real-Time Scheduling via Reduction to Uniprocessor. RTSS 2011: 104-115

Outline

Introduction

Schedulability Analysis: Global EDF

Schedulability Analysis: Global RM

Appendix: Augmentation Factor

Normal Collection of Jobs

A job collection $\mathcal{J}$ is a set of jobs that are revealed online over time:

- a job $j \in \mathcal{J}$ becomes known upon the release date of j
- Each job $j \in \mathcal{J}$ is characterized by its arrival time r_j, absolute deadline d_j, and an unknown execution time c_j.

Note that the actual execution time c_j of a job is discovered by the scheduler only after the job signals completion.

Optimal Schedule for $\mathcal{J}$

Given $\mathcal{J}$, suppose that infinitely many (or, say, $|\mathcal{J}|$) processors of unit speed were available.

Optimal Schedule for $\mathcal{J}$

Given $\mathcal{J}$, suppose that infinitely many (or, say, $|\mathcal{J}|$) processors of unit speed were available.

Then, the following scheduling algorithm S_∞ is optimal:

- just allocate one processor to each job and schedule each job as early as possible.

Schedulability for EDF

Theorem

Consider a normal collection $\mathcal{J}$ of jobs and let $\alpha \geq 1$. Then at least one of the following conditions holds:

1. all jobs in $\mathcal{J}$ are completed within their deadline under EDF on M processors of speed α, or
2. $\mathcal{J}$ is infeasible under S_∞, or
3. there is an interval I such that any feasible schedule for $\mathcal{J}$ must finish more than $(\alpha M - M + 1) \cdot |I|$ units of work within I.

Proof

- The details are omitted, please refer to Bonifaci et al. in ECRTS 2013 (Lemma 3 in Page 228).

Speedup for Normal Collection of Jobs

Theorem

Any normal collection of jobs that is feasible on M processors of unit speed is EDF-schedulable on M processors of speed $2 - 1/M$.

Proof

The feasibility on M processors of unit speed implies that the demand at any interval I is at most $M \cdot |I|$. By setting α to $2 - \frac{1}{M}$, for any interval I, we have

$$(\alpha M - M + 1) \cdot |I| = M \cdot |I|.$$

Hence, this implies that EDF finishes all jobs by their respective deadline at speed $2 - \frac{1}{M}$.

Putting Together

Theorem

If $\forall t > 0$, we have $dbf(\tau_i, t) \leq t$ for every task τ_i and $\sum_{i=1}^{N} dbf(\tau_i, t) \leq M \cdot t$, then this task set with N tasks is EDF-schedulable on M processors of speed $2 - 1/M$.

Theorem

If $\forall t > 0$, we have $dbf(\tau_i, t) \leq \frac{t}{2-1/M}$ for every task τ_i and $\sum_{i=1}^{N} dbf(\tau_i, t) \leq M \cdot \frac{t}{2-1/M}$, then this task set with N tasks is EDF-schedulable on M processors.

This analysis also works for arbitrary deadlines.