Component-Based Software Design

Hierarchical Real-Time Scheduling
lecture 1/4

Enrico Bini

March 10, 2015

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Outline

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Info

- Please interrupt me anytime if something is unclear;
- Topics: hierarchical real-time scheduling over single/multi-cores.
- My classes:

Tue	2015-03-10	11:00–13:00
Wed	2015-03-11	10:00–13:00
Tue	2015-03-24	11:00–13:00
Wed	2015-03-25	10:00–13:00

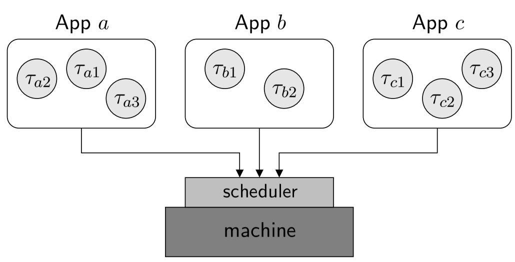
  - start at $+10$ min,
  - depending on the topic break for b minutes, during classes;
  - end around $-15 \times h + 10 + b$ minutes, with h number of hours.
- Exam: one/two questions in the written examination.
- Material: slides, your notes.

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Outline

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Motivations

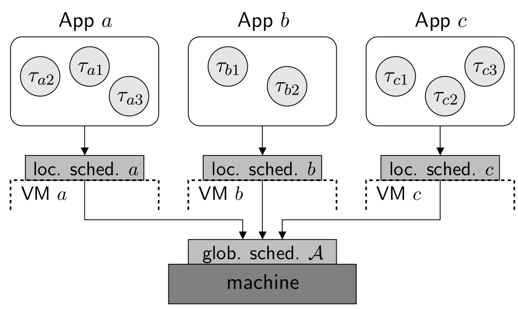


Applications sharing the same machine. Issues:

- a misbehavior in App a (running longer than expected, etc.) may cause misbehaviors on Apps b, c
- priority of application tasks needs to be comparable
  - may violate intellectual properties
  - when a new application joins, priorities of new tasks need to be assigned w.r.t. other tasks

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Solution



- *global scheduler* $\mathcal{A}$ assigns physical machine to *virtual machines*
- tasks scheduled over VMs by *local scheduler*
- misbehavior confined at app boundary
- task prio meaningful only within app

Hierarchical real-time scheduling

Focus of my lectures:

how can real-time constraints of components be guaranteed over VMs?

Starting point:

1. Model of a real-time component (periodic tasks, deadlines, etc, we assume you are familiar with it)
2. Model of a scheduler (rules that determine what applications, tasks to execute over the platform)
3. Model of the VM: how is a virtual machine for real-time application characterized? (**main focus of my classes**)

Outline

Real-time model of VMs

- A *virtual machine* is characterized by the amount of resource assigned to it by the global scheduler.
- Review of scheduling problems (to be applied at global scheduler level).
- Basic elements of a scheduling problem:
  - a set $\mathcal{N}$ of VMs requiring work to be made. Since they are finite, we represent them by $\mathcal{N} = \{1, 2, \ldots, n\}$;
  - a set $\mathcal{M}$ of (physical) *machines* capable to perform some work. Since they are finite we represent them by $\mathcal{M} = \{1, 2, \ldots, m\}$;
  - a *time set* $\mathcal{T}$, over which the scheduling is performed (in my classes always $[0, \infty)$, in general it may be $\mathbb{N}$);

Schedule

A schedule S of $\mathcal{N}$ over $\mathcal{M}$ is a function

$$S : \mathcal{M} \times \mathcal{T} \to \mathcal{N} \cup \{0\}$$

with "0" denoting *idle*.

Graphical representation of a schedule

- If $S(k, t) = i$ then the machine k is assigned to the i-th VM at time t.
- If $S(k, t) = 0$ then the machine k is not assigned at time t (we say that the k-th machine is *idle* at t).

- This definition of schedule implies that at every instant t each machine is assigned to at most one VM.
- Conversely, at every instant each VM may run over any number of machines in $\mathcal{M}$.

VM schedule

An equivalent representation of a schedule S is given by a set of $n + 1$ *VM schedule functions*

$$i \in \mathcal{N} \cup \{0\} \qquad s_i : \mathcal{T} \to 2^{\mathcal{M}}$$

such that

$$\forall t \in \mathcal{T},\ \forall i \neq j, \qquad s_i(t) \cap s_j(t) = \emptyset$$

$$\bigcup_{i=0}^{n} s_i(t) = \mathcal{M}$$

that is no machine can be used simultaneously by two VMs.

- A schedule S and a set of VMs schedules $\{s_1, \ldots, s_n\}$ are equivalent through

$$\forall t \in \mathcal{T}, k \in \mathcal{M}, \qquad S(k, t) = i \quad \Leftrightarrow \quad k \in s_i(t)$$

Outline

Single proc assumption

From now on, we assume a single processor.

Also, we drop the index i of the VM, since we consider only one in isolation.

- $m = |\mathcal{M}| = 1$,
- the schedule is just

$$S : \mathcal{T} \to \mathcal{N} \cup \{0\}$$

- the VM schedule functions $s(t)$ made by the global scheduler are

$$s : \mathcal{T} \to \{0,1\}$$

  - if $s(t) = 1$ the processor is allocated to the VM,
  - if $s(t) = 0$ the processor is not allocated to the VM.

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Legal schedules

Given a global scheduling algorithm $\mathcal{A}$, we denote by $\mathcal{L}_*$ the set of all possible legal VM scheduling functions $s_i(t)$, defined as

$$\mathcal{L}_* = \{(s_0, s_1, \dots, s_n) : \text{scheduling functions } s_1(t), \dots, s_n(t) \text{ may be generated by } \mathcal{A}\},$$

while the set $\mathcal{L}_i$ of legal scheduling functions for i-th VM is defined as a projection of $\mathcal{L}_*(\mathcal{P})$, that is

$$\mathcal{L}_i(\mathcal{P}) = \{s_i : 1 \le i \le n, (s_1, \dots, s_n) \in \mathcal{L}_*\}.$$

Example:

- $\mathcal{L}_i$ legal schedules of a VM implemented by a periodic servers with period P_i and budget Q_i;
- then $s_i \in \mathcal{L}_i$ is such that...

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Supply functions

Definition (*supply lower bound function* of a VM)

We define the *supply lower bound function* $\mathsf{slbf}(t)$ of a VM as

$$\mathsf{slbf}(t) = \min_{t_0 \in \mathcal{T}, s \in \mathcal{L}} \int_{t_0}^{t_0+t} s(x)\,dx.$$

Informally, "minimum amount of resource made available by the VM in every interval of length t".

Definition (*supply upper bound function* of a VM)

We define the *supply upper bound function* $\mathsf{subf}(t)$ of a VM as

$$\mathsf{subf}(t) = \max_{t_0 \in \mathcal{T}, s \in \mathcal{L}} \int_{t_0}^{t_0+t} s(x)\,dx.$$

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Outline

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Usage of supply bounds

Theorem (Resource bounds)

For any legal VM schedule $s \in \mathcal{L}$, it is always

$$\mathsf{slbf}(b-a) \le \int_a^b s(t)\,dt \le \mathsf{subf}(b-a).$$

How long does it take to compute a work W over a VM?

- *worst-case response time* $R_{\mathsf{w}}(W)$ of work W over a VM

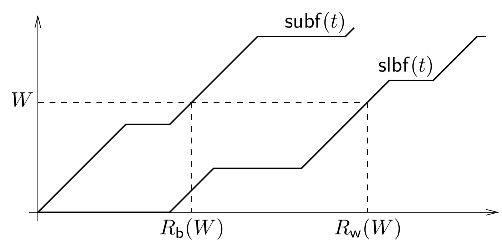
$$R_{\mathsf{w}}(W) = \sup\{t : \mathsf{slbf}(t) < W\},$$

- *best-case response time* $R_{\mathsf{b}}(W)$ of work W over a VM

$$R_{\mathsf{b}}(W) = \inf\{t : \mathsf{subf}(t) \ge W\}.$$

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Illustration of supply bounds and response-time bounds



Illustrate where the following sets are

- $\{t : \mathsf{slbf}(t) < W\}$
- $\{t : \mathsf{subf}(t) \ge W\}$

Example slbf(t), subf(t): static partition 1

- Let us assume the static VM schedule

$$s(t) = \begin{cases} 1 & 0 \le (t \mod 4) < 1 \\ 0 & \text{otherwise} \end{cases}$$

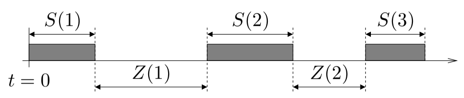
- What are its slbf(t) and subf(t)?
  - Remember: "any interval of length t" not necessarily $[0, t]$
- What are $R_\text{b}(2)$ and $R_\text{w}(2)$?

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Resource assigned with a given pattern

If resource/idle alternate according to a known sequence:

- $S(k)$, with $k = 1, 2, \ldots$ is the sequence of budgets
- $Z(k)$, with $k = 1, 2, \ldots$ is the sequence of idle intervals



Let $t = 0$ coincide with the start of the first budget $S(1)$

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

slbf(t) with given pattern

Theorem (slbf of sequence of budgets/idle)

$$\text{slbf}(t) \ge \min\{t - \sigma_Z(n), \sigma_S(n)\}, \quad t \in I_n, n \in \mathbb{N}$$

with the sequence of intervals $\{I_n\}_{n \in \mathbb{N}}$ *defined as*

$$I_n = \begin{cases} [0, \sigma_Z(1)) & n = 0 \\ [\sigma_Z(n) + \sigma_S(n-1), \sigma_Z(n+1) + \sigma_S(n)) & n \ge 1 \end{cases}$$

and with

$$\sigma_S(n) = \inf_{n_0} \sum_{k=n_0}^{n_0+n-1} S(k),$$

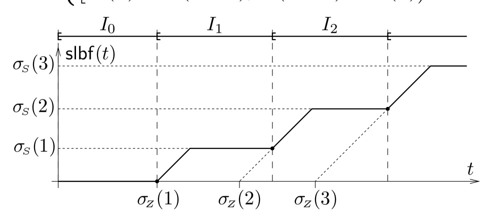
$$\sigma_Z(n) = \sup_{n_0} \sum_{k=n_0}^{n_0+n-1} Z(k).$$

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Illustration of slbf

$$\text{slbf}(t) \ge \min\{t - \sigma_Z(n), \sigma_S(n)\}, \quad t \in I_n, n \in \mathbb{N}$$

$$I_n = \begin{cases} [0, \sigma_Z(1)) & n = 0 \\ [\sigma_Z(n) + \sigma_S(n-1), \sigma_Z(n+1) + \sigma_S(n)) & n \ge 1 \end{cases}$$



Remark: *the worst cases for $S(k)$ and $Z(k)$ are considered independently, leading to potential extra pessimism (see next example)*

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Example slbf(t), subf(t): static partition 2

- Let us assume the static VM schedule

$$s(t) = \begin{cases} 1 & (t \mod 12) \in [2, 3) \cup [5, 7) \cup [10, 12) \\ 0 & \text{otherwise} \end{cases}$$

- What are its slbf(t) and subf(t)?
  - Remember: the interval with the minimum supply can start at different points for different t
- What is W such that $R_\text{w}(W) - R_\text{b}(W)$ over the VM, is maximized?

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Example slbf(t), subf(t): periodic server

- Let us assume that
  - the VM is implemented by a *periodic server*
  - the global scheduler guarantees a budget Q every period P
- What is its slbf(t)? Remember: scenario of minimum possible supply must be assumed, among all legal schedules in $\mathcal{L}$

Example slbf(t), subf(t): periodic server

- Let us assume that
  - the VM is implemented by a *periodic server*
  - the global scheduler guarantees a budget Q every period P
- What is its slbf(t)? Remember: scenario of minimum possible supply must be assumed, among all legal schedules in $\mathcal{L}$

$$\text{slbf}(t) = \begin{cases} 0 & t \in [0, P-Q] \\ (k-1)Q & t \in (kP-Q, (k+1)P-2Q] \\ t-(k+1)(P-Q) & \text{otherwise} \end{cases}$$

with $k = \left\lceil \frac{t-(P-Q)}{P} \right\rceil$.

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Properties of slbf(t)

1. slbf$(0) = 0$ (no resource in empty interval),
2. $\forall s \geq t \geq 0,$ slbf$(s) \geq$ slbf(t) (non-decreasing)
3. $\forall s, t \geq 0,$ slbf$(s+t) \geq$ slbf(s) + slbf(t) (super-additivity)
4. $\forall s \geq t \geq 0,$ slbf$(s) -$ slbf$(t) \leq s - t$ (Lipschitz continuous with factor 1):

Component-Based Software Design
Enrico Bini
Introduction to the course
Composition of real-time systems
Scheduling problems, background
Supply function, definition
Supply function, properties

Exploiting properties

Let us assume that at some instant t^* we know that

$$\text{slbf}(t^*) \geq c^*, \tag{1}$$

From Lipschitz-continuity and (1), we find

$$\forall t \in [0, t^*] \quad \text{slbf}(t) \geq \begin{cases} 0 & 0 \leq t \leq t^* - c^* \\ c^* + (t - t^*) & t^* - c^* < t \leq t^* \end{cases}$$

Let q and r be quotient and rest of the Euclidean division of any $t \geq 0$ by t^*, that is

$$t = qt^* + r, \qquad q \in \mathbb{N},\ r \in [0, t^*).$$

Then, for any $t \geq 0$, the following lower bound holds

$$\text{slbf}(t) = \text{slbf}(\underbrace{t^* + \cdots + t^*}_{q \text{ times}} + r) \geq$$
$$\geq \underbrace{\text{slbf}(t^*) + \cdots + \text{slbf}(t^*)}_{q \text{ times}} + \text{slbf}(r) = qc^* + \text{slbf}(r).$$