Compositional Schedulability Analysis

Insup Lee
PRECISE (Penn Research in Embedded Computing and Integrated Systems Engineering) Center
Department of Computer and Information Science
University of Pennsylvania

October 19, 2008

Workshop on Foundations and Applications of Component-based Design (WFCD’2008)
Motivating Example
Component Abstraction

Another VM
Real-Time Tasks
Real-Time Demand

Java Virtual Machine
Real-Time Tasks
Real-Time Demand

OS Scheduler

CPU

Oct 19, 2008
WFCD'2008
PRECISE Center
Component Abstraction

Another VM

J₁(25,2)  J₂(30,4)

VM Scheduler

CPU Share

Java Virtual Machine

J₁(50,3)  J₂(75,5)

VM Scheduler

CPU Share

OS Scheduler

CPU

Oct 19, 2008

WFCD'2008

PRECISE Center
The CSA Problem Statement
Two Problems: Abstraction & Composition

- **Abstraction Problem:** abstracts the real-time requirements of component (application) with *interface*

  ![Diagram](Diagram.png)

- **Compute the minimum real-time requirements necessary for guaranteeing the schedulability of a component**
Two Problems: Abstraction & Composition

- **Composition Problem**: composes component-level properties into system-level (or next-level component) properties.

![Diagram showing composition process with component interfaces and a scheduling algorithm.]

Oct 19, 2008

WFCD'2008

PRECISE Center
Compositionality

- Compositionality:
  - system-level properties can be established by composing independently analyzed component-level properties
- Compositional reasoning based on assume/guarantee paradigm
  - components are combined to form a system such that properties established at the component-level still hold at the system level.
- Compositional schedulability analysis using the demand/supply bounds
  - Establish the system-level timing properties by combining component-level timing properties through interfaces
Resource Satisfiability Analysis

- Given a task set and a resource model, resource satisfiability analysis is to determine if, for every time,

\[ \text{resource demand, which a task set needs under a scheduling algorithm} \leq \text{(minimum possible) resource supply} \]
Resource Demand Models
Real-time demand composition

- Combine real-time requirements of multiple tasks into real-time requirement of a single task

```
[Real-Time Constraint] || [Real-Time Constraint] = [Real-Time Constraint]
atches

[Periodic Constraint] || [Periodic Constraint] = [Periodic Constraint]
```

EDF / RM
Non-composable periodic models?

- What are right abstraction levels for real-time components?
- (execution time, period)
- P1 = (p1,e1); e.g., (3,1)
- P2 = (p2,e2); e.g., (7,1)
- What is P1 || P2?
  - \((\text{LCM}(p1,p2), e1*n1 + e2*n2); \text{e.g.}, (21,10)\)
    where \(n1*p1 = n2*p2 = \text{LCM}(p1,p2)\)
- What is the problem?
  - beh(P1) || beh(P2) = beh(P1||P2)?
- Can we do
  - \((P1 || P2) || P3 = P || P3, \text{where } P = P1 || P2?\)
Simple Observation (1)

- Given a task group $G$ such that
  - Scheduling algorithm: EDF
  - A set of periodic tasks: $\{ T_1(3,1), T_2(7,1) \}$,
  model the timing requirements of the task group with a periodic task model
- $G(3, 1.43)$ based on utilization does not work!!

Deadline miss for $T_2$
Simple Observation (2)

• Given a task group $G$ such that
  - Scheduling algorithm : EDF
  - A set of periodic tasks : $\{ T_1(3,1), T_2(7,1) \}$, model the timing requirements of the task group with a periodic task model

• $G \ (3, 2.01)$ works !!
Resource Demand Bound

- Resource demand bound during an interval of length $t$
  - $\text{dbf}(W,A,t)$ computes the maximum possible resource demand that $W$ requires under algorithm $A$ during a time interval of length $t$

- Periodic task model $T(p,e)$ [Liu & Layland, ’73]
  - characterizes the periodic behavior of resource demand with period $p$ and execution time $e$
  - Ex: $T(3,2)$
Demand Bound - EDF

- For a periodic workload set $W = \{T_i(p_i,e_i)\}$,
  - $dbf(W, A, t)$ for EDF algorithm [Baruah et al., '90]

$$dbf (W, EDF, t) = \sum_{T_i \in W} \left[ \frac{t}{p_i} \right] \cdot e_i$$
\[ T_1 = \{(cpu, 1)\} : \emptyset : T_1 \]
\[ T_2 = \{(cpu, 1)\} : \emptyset : T_2 \]
\[ +\emptyset : \{(cpu, 1)\} : \emptyset : T_2 \]
\[ T_3 = \{(cpu, 1)\} : \emptyset : \emptyset : T_3 \]
\[ +\emptyset : \{(cpu, 1)\} : \emptyset : T_3 \]
\[ +\emptyset : \emptyset : \{(cpu, 1)\} : T_3 \]
Task (resource demand) representations
Resource Supply Models
Resource Modeling

- **Dedicated resource**: always available at full capacity

- **Shared resource**: not a dedicated resource
  - **Time-sharing**: available at some times
  - **Non-time-sharing**: available at fractional capacity
Resource Modeling

- Time-sharing resources
  - Bounded-delay resource model [Mok et al., ’01] characterizes a time-sharing resource w.r.t. a non-time-sharing resource
  - Periodic resource model $\Gamma (\Pi, \Theta)$ [Shin & Lee, RTSS ’03] characterizes periodic resource allocations
  - EDP model [Easwaran et al., RTSS 07]
Resource Supply Bound

- Resource supply during an interval of length $t$
  - $sbf_R(t)$: the minimum possible resource supply by resource $R$ over all intervals of length $t$

- For a single periodic resource model, i.e., $\Gamma(3,2)$
  - we can identify the worst-case resource allocation
Schedulability Condition - EDF

- A periodic workload set \( W \) is schedulable under a scheduling algorithm \( A \) over a periodic resource model \( \Gamma \left( \Pi, \Theta \right) \) if and only if

\[
\forall t > 0 \quad \text{dbf}(W, \text{EDF}, t) \leq \text{sbfr}(t)
\]

- \( A = \text{EDF} \)
EDP resource model based Interfaces

- Explicit Deadline Periodic resource

- Specification: $\Omega = (\Pi, \Theta, \Delta)$
  - Explicit deadline $\Delta$
  - $\Theta$ resource units in $\Delta$ time units
  - Repeat supply every $\Pi$ time units
EDP supply bound function ($sbf_\Omega$)

1. $sbf_\Omega(t)$

$$sbf_\Omega(t) = y\Theta + \max \{0, t - (\Pi + \Delta - 2\Theta) - y\Pi\}$$

where

$$y = \left\lfloor \frac{t - (\Delta - \Theta)}{\Pi} \right\rfloor, \quad t \geq \Delta - \Theta$$

2. $lsbf_\Omega(t)$

$$lsbf_\Omega(t) = \frac{\Theta}{\Pi} (t - (\Pi + \Delta - 2\Theta))$$

Bandwidth

Starvation length
Supply bound function \( (sbf_\Omega) \)

\[ \Gamma(5,3) \]

Starvation length = 4

\[ \Omega(5,3,4) \]

Starvation length = 3

Oct 19, 2008
ACSR+ for supply partition specification

\[ S_1 = \{(cpu, -)\} : \emptyset : S_1 \]

\[ S_2 = \{(cpu, -)\} : \emptyset : S_2 \oplus \emptyset : \{(cpu, -)\} : S_2 \]

\[ T_1 = \{(cpu, 1)\} : \emptyset : T_1 \]

\[ T_2 = \{(cpu, 1)\} : \emptyset : T_2 \]

Notion of “schedulable under”

1. \( T_1 \) is schedulable under \( S_1 \)
2. \( T_2 \) is schedulable under \( S_2 \)
3. \( T_1 \) is not schedulable under \( S_2 \)
Resource Supply Models

ACSR+

Recurring branching resource supply model

Bounded-delay Resource model

Tree schedule

EDP model

Periodic model
Compositional Schedulability Analysis
Component Abstraction
Compositional Real-Time Guarantees
Hierarchical Scheduling Framework

• Resource allocation from parent to child

• Notations
  - Leaf $\rightarrow C_1, C_2, C_3$
  - Non-leaf $\rightarrow C_4, C_5$
  - Root $\rightarrow C_5$

ARINC 653 $\rightarrow$ Two-level hierarchical framework
Compositional Schedulability Analysis (CSA)

- Assume/Guarantee reasoning
  - Let $C_R$ be a system configuration: Component $C$ is running on resource $R$.
  - Let $\text{supply}(C_R)$ be the residual supply of $R$ after $C$; I.e., supply to the rest of the system.
  - If
    - $C_1$ guarantees schedulability assuming $\text{demand}(C_1) \leq \text{supply}(C_2_R)$
    - $C_2$ guarantees schedulability assuming $\text{demand}(C_2) \leq \text{supply}(C_1_R)$
  - Then,
    - $C_1 || C_2$ guarantees schedulability in $(C_1 || C_2)_R$
Questions on CSA

- Dbf/sbf bounds
  - Associativity
  - Minimum bounds on hierarchical scheduling

- ACSR/ACSR+
  - Non-deterministic supply alternatives
  - Definition and characterization of “schedulable under”
    - Given demand process $T$ and supply partition $S$, when $T$ schedulable with respect to $S$.
    - Relation to Linear Logic?
Hierarchical Scheduling Framework for Virtual Clustering of Multiprocessors
Multicore Processor Virtualization

1. Compositional analysis of hierarchical multiprocessor real-time systems, through component interfaces
2. Using virtualization to develop new component interface for multiprocessor platforms
Multiprocessor Embedded Systems

- **Why consider multiprocessors**
  - Better tradeoff between computational power and costs (energy, fabrication)
  - Ability to exploit inherent concurrency in embedded software

- **Problem Statement**
  - Constrained deadline sporadic task model
    \[ \tau = \{\tau_1, \ldots, \tau_n\}, \text{where each } \tau_i=(T_i,C_i,D_i) \text{ with } C_i \leq D_i \leq T_i \]
    \(C_i\) units must be supplied non-concurrently
  - Identical, unit-capacity multiprocessor platform
    \(m\) processors
  - How can they be scheduled?
Partitioned Scheduling

\[
\tau_{x_1} \cup \tau_{x_2} \ldots \cup \tau_{x_m} = \tau
\]

\[
\tau_{x_i} \cap \tau_{x_j} = \phi \text{ for all } i \text{ and } j
\]
Global Scheduling

Physical processors

Single task cluster
Motivation for Virtual Clustering

- Task set and number of processors
  \( \tau_1 = \tau_2 = \tau_3 = \tau_4 = (3,2,3) \), \( \tau_5 = (6,4,6) \), and \( \tau_6 = (6,3,6) \), \( m=4 \)
- Schedule under clustered scheduling
  \( \tau_1, \tau_2, \tau_3 \) scheduled on processors 1 and 2
  \( \tau_4, \tau_5, \tau_6 \) scheduled on processors 3 and 4
Virtual Clustering Interface

- Physical processors
- Virtual processors
- Task clusters

\[ \tau_{x_1} \cup \tau_{x_2} \ldots \cup \tau_{x_k} = \tau \]
\[ \tau_{x_i} \cap \tau_{x_j} = \phi \text{ for all } i \text{ and } j \]
For each $\Gamma_i$, $m_i (\leq m)$ is maximum number of physical processors that can be assigned to $\Gamma_i$ at any instant.
Multiprocessor Periodic Resource (MPR) model

Does not capture concurrency bound of cluster

\[ \Gamma_1(\Pi_1, \Theta_1) \quad \Gamma_2(\Pi_2, \Theta_2) \quad \Gamma_k(\Pi_k, \Theta_k) \]

\[ C_1 \quad C_2 \quad \ldots \quad C_k \]

\[ \tau_{x_1} \quad \tau_{x_2} \quad \tau_{x_k} \]
Multiprocessor Periodic Resource (MPR) model

- $\Gamma = (\Pi, \Theta, m')$
  - $\Theta$ units of resource guaranteed in every $\Pi$ units of time, with amount of concurrency at most $m'$ in any time slot

- Why MPR model?
  - Periodicity enables transformation of resource model to tasks that can be used by various inter-cluster schedulers (Schedulers at higher level)
Virtual Cluster based Interface

1. Split task set $\tau$ into clusters $\tau_{x_1}, ..., \tau_{x_k}$
   - We assume that clusters are given

2. Abstract cluster $\tau_{x_i}$ into MPR model $\Gamma_i$
   - Solution for global EDF intra-cluster interface
   - Present sufficient schedulability condition and minimize overhead of $\Gamma_i$

3. Transform each $\Gamma_i$ into periodic tasks
   - Enables inter-cluster scheduler to schedule $\Gamma_i$
   - Preserves concurrency bound of $\Gamma_i$
Conclusions

• Interface framework for real-time system based on hierarchical schedulability analysis
  - Independent implementation of components
  - Interface-based component composition
  - Virtual clustering for multiprocessors

• Other issues
  - Task blocking due to synchronization
  - Context switch overheads

• Applications
  - ARINC 653
  - Automotive SAE J2056/Class C Vehicle Communication Requirements
  - Real-Time Virtual Machines (esp. multicore processors)
References

- Hierarchical Scheduling Framework for Virtual Clustering of Multiprocessors, Insik Shin, Arvind Easwaran, Insup Lee, ECRTS, Prague, Czech Republic, July 2-4, 2008 (Runner-up in the best paper award)
- Robust and Sustainable Schedulability Analysis of Embedded Software, Madhukar Anand and Insup Lee, LCTES, Tucson, AZ, Jun 12-13, 2008
- Compositional Feasibility Analysis for Conditional Task Models, Anand et al, ISORC 2008
- Compositional Real-Time Scheduling Framework, Shin & Lee, RTSS 2004
- Periodic Resource Model for Compositional Real-Time Guarantees, Shin & Lee, RTSS 2003
Related work

- Much work on hierarchical scheduling
  - Provide schedulability conditions that are needed for instantiation
  - Serves as the basis for abstraction
    - Shin and Lee, '03 '04, Easwaran et al., '06
- Real-time interface frameworks
  - Henzinger and Matic, '06
  - Wandeler and Thiele, '06
Compositional Analysis (Dependency using Task Parameters)

• [DaBu05] R. I. Davis and A. Burns, “Hierarchical fixed priority preemptive scheduling”. In RTSS 2005.
• [DaBu06] R. I. Davis and A. Burns, “Resource sharing in hierarchical fixed priority pre-emptive systems”. In RTSS 2006.
Compositional Analysis  (Dependency using Protocols)

- [DaBu06] R. I. Davis and A. Burns, “Resource sharing in hierarchical fixed priority pre-emptive systems”. In RTSS 2006.
Compositional Analysis (Conditional Task Models)

Incremental and Compositional Analysis


Two-level hierarchical scheduling

Bounded-delay Resource Models

- **FeMo02** X. Feng and A. Mok, “A model of hierarchical real-time virtual resources.” In RTSS 2002.
Periodic Resource Models

Virtual Clustering


Partitioned Scheduling


Global Scheduling

Global Scheduling

Global Scheduling

Portioning and Task Splitting


• [ABB08] Bjorn Andersson, Konstantinos Bletsas, and Sanjoy K. Baruah. “Scheduling arbitrary-deadline sporadic tasks on multiprocessors”. In RTSS, 2008.


Other Multiprocessor Algorithms

- [BaCa03] Sanjoy K. Baruah and John Carpenter. “Multiprocessor fixed-priority scheduling with restricted interprocessor migrations”. In ECRTS, 2003. (restricted task migration)

Questions?

Thank You!