The Computation Time Process Model

Martin Korsgaard and Sverre Hendseth

Norwegian University of Science and Technology
Department of Engineering Cybernetics

CPA 2011

NTNU

Introduction to Introduction

Definition: Computation Time Process (CTP)

A CTP is an abstract representation of execution time with a SEQ/PAR structure.

Motivation

To explore general, temporal properties of executing processes with a SEQ/PAR structure in multiprocessor real-time environments.

Outline

Background

Definition: Real-time System

A system is real-time, if its correctness depends not only on computational results, but also on the time when those results are produced.

Definition: Schedulability Analysis

Real-time schedulability analysis is to take a real-time system and prove in advance that all deadlines will be met.

Definition: Worst-case Execution Time (WCET)

The WCET is an upper bound to the execution time of some computation.

Actual execution time is variable and undecidable; only WCET can be found in advance.

Therefore, analyses must take into account that execution times may be less than expected.

Real-time Schedulability Analysis

Typical System Model for RTSA

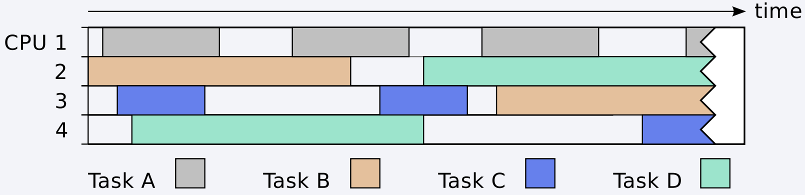
- Tasks may be sporadic (triggered) or periodic.
- Tasks are defined by their computation, deadline and (minimum) period
- Each job (=task instance) executes on one processor.
- System defined by job scheduler (EDF, RMS, DMPO..) and number of processors.

This paper

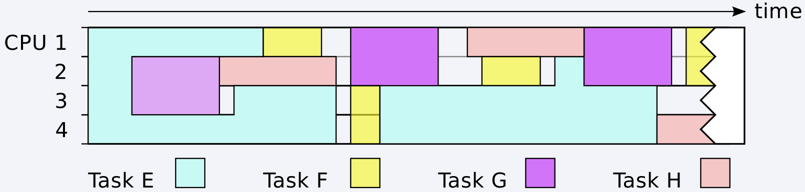
- We look at the timing behaviour of one job only, written with a SEQ/PAR structure
- The number of processors available to the job is considered time-varying and non-deterministic, due to the possible existence of higher priority jobs
- The intra-job scheduler is assumed to be work-conserving but otherwise undefined.

Structure of tasks

Serial jobs (traditional model)



Jobs with parallel structure (this paper)



Basic Definitions

A CTP can be **0** (do nothing), **1** (do one thing), or a sequence, or parallel composition of two other CTPs:

$$
\begin{aligned}
P \in \mathbb{P} \iff & P = \mathbf{0} & \\
\vee\ & P = \mathbf{1} & \\
\vee\ & P = Q \,;R & Q, R \in \mathbb{P} \\
\vee\ & P = Q \,||\, R & Q, R \in \mathbb{P}
\end{aligned}
$$

The value of **1** with respect to real time represents the minimum quantification of time for the system.

These processes satisfy the following intuitive laws

$$
\begin{aligned}
\mathbf{0} \,; P &= P & \mathbf{0} \,||\, P &= P \\
P \,; \mathbf{0} &= P & P \,||\, Q &= Q \,||\, P \\
(P \,; Q) \,; R &= P \,;(Q \,; R) & (P \,||\, Q) \,||\, R &= P \,||(Q \,||\, R)
\end{aligned}
$$

Basic Measures

Three measures ($\mathbb{P} \to \mathbb{N}$) are used in the paper;

Total computation ($\mathcal{C}$)

which is the count of **1**s in the process

The length ($\mathcal{L}$)

which is the length of the longest sequence

The immediate height ($\mathcal{H}$)

which is the number of **1**s that may be executed at the first step.

$$\mathcal{H}(\mathbf{1}) = 1$$

$$\mathcal{H}(\mathbf{0}) = 0$$

$$\mathcal{H}(P\,;Q) = \begin{cases} \mathcal{H}(P) & \text{if } P \neq \mathbf{0} \\ \mathcal{H}(Q) & \text{if } P = \mathbf{0} \end{cases}$$

$$\mathcal{H}(P \,||\, Q) = \mathcal{H}(P) + \mathcal{H}(Q)$$

Stepping

Definition: Step

$\text{step}(P, m)$ yields all possible outcomes of executing P with given m processors for a single unit of time.

$\text{step}\colon \mathbb{P} \times \mathbb{N} \to \{\mathbb{P}\}$

A step must satisfy the work-conservation requirement of the intra-job scheduler.

$$\text{step}(\mathbf{1}, m) = \begin{cases} \{\mathbf{1}\} & \text{if } m = 0 \\ \{\mathbf{0}\} & \text{if } m \geq 1 \end{cases}$$

$$\text{step}(\mathbf{0}, m) = \{\mathbf{0}\}$$

Stepping ($P\,;Q$ and $P\,||\,Q$)

A single step for a sequence $P\,;Q$ depends on whether there is anything to execute in P.

$$\mathrm{step}((P\,;Q),m) = \begin{cases} \{(P'\,;Q) \colon P' \in \mathrm{step}(P,m)\} & \text{if } P \neq \mathbf{0} \\ \mathrm{step}(Q,m) & \text{if } P = \mathbf{0} \end{cases}$$

A single step of a parallel $P\,||\,Q$ is any steps of P and Q in parallel that satisfies the work-conservation requirement.

$$\begin{aligned}\mathrm{step}((P\,||\,Q),m) = \Big\{(P'\,||\,Q') \colon\ & P' \in \mathrm{step}(P,m_P), Q' \in \mathrm{step}(Q,m_Q), \\ & m_P \in [0,\mathcal{H}(P)], \\ & m_Q \in [0,\mathcal{H}(Q)], \\ & m_P + m_Q = \min\{\mathcal{H}(P)+\mathcal{H}(Q),m\}\Big\}\end{aligned}$$

Examples of steps

$$\text{step}\left((\mathbf{1} \,||\, \mathbf{1}), 1\right) = \{\mathbf{1}\} \tag{1}$$

$$\text{step}\left(((\mathbf{1} \,||\, \mathbf{1})\,;\,\mathbf{1}), 1\right) = \{\mathbf{1}\,;\,\mathbf{1}\} \tag{2}$$

$$\text{step}\left((\mathbf{1}\,;(\mathbf{1} \,||\, \mathbf{1})), 1\right) = \{\mathbf{1} \,||\, \mathbf{1}\} \tag{3}$$

$$\text{step}\left((\mathbf{1} \,||\, \mathbf{1}), 2\right) = \{\mathbf{0}\} \tag{4}$$

$$\text{step}\left((\mathbf{1}\,;(\mathbf{1} \,||\, \mathbf{1})), 2\right) = \{\mathbf{1} \,||\, \mathbf{1}\} \tag{6}$$

$$\text{step}\left(((\mathbf{1} \,||(\mathbf{1}\,;\,\mathbf{1})), 1\right) = \{(\mathbf{1}\,;\,\mathbf{1}),\ (\mathbf{1} \,||\, \mathbf{1})\} \tag{7}$$

Schedules

Notation

$\langle 3, 4\rangle$ means 3 processors for the first step and 4 for the second step. The set of all schedules is $\mathbb{S}$.

Concatenation

$\langle 3, 4\rangle \frown \langle 10, 11\rangle = \langle 3, 4, 10, 11\rangle$

Execution on a schedule

The possible processes remaining after executing P on schedule s is denoted $P \otimes s$.

$\otimes : \mathbb{P} \times \mathbb{S} \to \{\mathbb{P}\}$

$$P \otimes \langle\rangle = \{P\}$$

$$P \otimes (\langle m\rangle \frown s) = \bigcup_{P' \in \mathrm{step}(P,m)} P' \otimes s$$

Complete on Schedule

- A process P will complete on a schedule s if $P \otimes s = \{\mathbf{0}\}$
- P may complete on a schedule s if $\mathbf{0} \in P \otimes s$

Scheduling Example

$$P = (\mathbf{1} \,;\, \mathbf{1}) \,||\, \mathbf{1} \,||\, \mathbf{1}$$
$$s = \langle 2, 3 \rangle$$

$$P \otimes \langle 2 \rangle = \{(\mathbf{1} \,;\, \mathbf{1}), (\mathbf{1} \,||\, \mathbf{1})\} \tag{1}$$
$$(\mathbf{1} \,;\, \mathbf{1}) \otimes \langle 3 \rangle = \{\mathbf{1}\} \tag{2.1}$$
$$(\mathbf{1} \,||\, \mathbf{1}) \otimes \langle 3 \rangle = \{\mathbf{0}\} \tag{2.2}$$

$$P \otimes s = \{\mathbf{1},\ \mathbf{0}\} \tag{3}$$

so P may or may not complete on the schedule.

Partial Order: Upper Bound Order ($\sqsupseteq$)

Definition

Q is an upper bound of P, written $P \sqsupseteq Q$ if P can be derived from Q by replacing $\mathbf{1}$s with $\mathbf{0}$s

Motivation

If WCET analysis yields process Q, then an execution will behave as some process $P \sqsupseteq Q$.

Examples

$$\mathbf{1}\,;\mathbf{1} \sqsupseteq \mathbf{1}\,;\mathbf{1}\,;\mathbf{1} \tag{1}$$

$$\mathbf{1}\,;\mathbf{1} \sqsupseteq (\mathbf{1} \,||\, \mathbf{1})\,;(\mathbf{1} \,||\, \mathbf{1}) \tag{2}$$

$$P \otimes s \sqsupseteq P \tag{3}$$

$$\mathbf{0} \sqsupseteq Q \tag{4}$$

Example of incomparable processes are $\mathbf{1}\,;\mathbf{1}$ and $\mathbf{1} \,||\, \mathbf{1}$.

Partial Order: Schedulability Order ($\leq$)

Definition

A process P is easier to schedule than a process Q, written $P \leq Q$, iff for all schedules s,

$$Q \otimes s = \{\mathbf{0}\} \implies P \otimes s = \{\mathbf{0}\}$$

(Read: Q will complete on s implies that P will also complete on s.)

Examples

$$\mathbf{1} ; \mathbf{1} \leq \mathbf{1} ; \mathbf{1} ; \mathbf{1} \tag{1}$$

$$\mathbf{1} \mid\mid \mathbf{1} \leq \mathbf{1} ; \mathbf{1} \tag{2}$$

$$\mathbf{0} \leq Q \tag{3}$$

Example of incomparable processes are $\mathbf{1} ; \mathbf{1}$ and $\mathbf{1} \mid\mid \mathbf{1} \mid\mid \mathbf{1}$

Timing Anomalies

Big question:

$$P \sqsupseteq Q \overset{?}{\Longrightarrow} P \leq Q$$

In words:

Removing **1**s from a process will not make it harder to schedule

Consequence

WCETs of sub-process would yield worst-case composite process.

Unfortunately, the relation does not hold

Example of Timing Anomaly

$$Q = (\mathbf{1}\,;(\mathbf{1}\,||\,\mathbf{1}))\;\Big|\Big|\;(\mathbf{1}\,;(\mathbf{1}\,||\,\mathbf{1}))$$
$$s = \langle 2, 4\rangle$$
$$u = \langle 1, 2, 4\rangle$$

$$Q \otimes \langle 2\rangle = \{\mathbf{1}\,||\,\mathbf{1}\,||\,\mathbf{1}\,||\,\mathbf{1}\}$$
$$(\mathbf{1}\,||\,\mathbf{1}\,||\,\mathbf{1}\,||\,\mathbf{1}) \otimes \langle 4\rangle = \{\mathbf{0}\}$$
$$Q \otimes s = \{\mathbf{0}\}$$

$$Q \otimes \langle 1\rangle = ((\mathbf{1}\,;(\mathbf{1}\,||\,\mathbf{1}))\;\Big|\Big|\;\mathbf{1}\;\Big|\Big|\;\mathbf{1})$$
$$((\mathbf{1}\,;(\mathbf{1}\,||\,\mathbf{1}))\;\Big|\Big|\;\mathbf{1}\;\Big|\Big|\;\mathbf{1}) \otimes \langle 2\rangle = \{(\mathbf{1}\,||\,\mathbf{1}\,||\,\mathbf{1}),\; \color{red}{(\mathbf{1}\,;(\mathbf{1}\,||\,\mathbf{1}))}\}$$
$$(\mathbf{1}\,||\,\mathbf{1}\,||\,\mathbf{1}) \otimes \langle 4\rangle = \{\mathbf{0}\}$$
$$\color{red}{(\mathbf{1}\,;(\mathbf{1}\,||\,\mathbf{1}))} \otimes \langle 4\rangle = \color{red}{\{\mathbf{1}\,||\,\mathbf{1}\}}$$
$$Q \otimes u = \{\mathbf{0},\; \color{red}{(\mathbf{1}\,||\,\mathbf{1})}\}$$

Consequences of Timing Anomalies

The Example

$$Q \otimes \langle 2, 4 \rangle = \{\mathbf{0}\} \qquad Q \otimes \langle 1, 2, 4 \rangle = \{\mathbf{0},\ (\mathbf{1} \,||\, \mathbf{1})\}$$

Observations that follow:

- A process may be harder to schedule after it has performed some execution
- A process known to complete may no longer complete if given more processors.
- WCETs of sub-processes do not constitute worst-case when combined.

Possible counter-argument

The scheduler makes a “wrong decision”.

$\Longrightarrow$ Make a better the scheduler?

No Perfect Scheduler

Observation 2

There exists some process Q and schedule s so that the set $Q \otimes s$ has no least element in the schedulability order.

(The set $Q \otimes s$ may have incomparable elements.)

Example:

$$Q = \big(\mathbf{1}\,;(\mathbf{1}\,||\,\mathbf{1})\big)\,\big|\big|\,\big(\mathbf{1}\,;\mathbf{1}\,;\mathbf{1}\big)$$
$$s = \langle 1, 3\rangle$$

$$Q \otimes \langle 1\rangle = \Big\{\big(\mathbf{1}\,||\,\mathbf{1}\,||\,(\mathbf{1}\,;\mathbf{1}\,;\mathbf{1})\big),\ \big(\mathbf{1}\,;(\mathbf{1}\,||\,\mathbf{1})\big)\,\big|\big|\,\big(\mathbf{1}\,;\mathbf{1}\big)\Big\}$$
$$\big(\mathbf{1}\,||\,\mathbf{1}\,||(\mathbf{1}\,;\mathbf{1}\,;\mathbf{1})\big) \otimes \langle 3\rangle = \{\mathbf{1}\,;\mathbf{1}\}$$
$$\Big(\big(\mathbf{1}\,;(\mathbf{1}\,||\,\mathbf{1})\big)\,\big|\big|\,\big(\mathbf{1}\,;\mathbf{1}\big)\Big) \otimes \langle 3\rangle = \{\mathbf{1}\,||\,\mathbf{1}\,||\,\mathbf{1}\}$$

$$Q \otimes s = \Big\{(\mathbf{1}\,;\mathbf{1}),\ (\mathbf{1}\,||\,\mathbf{1}\,||\,\mathbf{1})\Big\}$$

No Perfect Scheduler (contd.)

There exists situations where the scheduling decision may lead to either

$$(\mathbf{1} \,;\, \mathbf{1})$$

$$\text{or} \qquad (\mathbf{1} \,||\, \mathbf{1} \,||\, \mathbf{1})$$

- If the schedule is $\langle 1, 1 \rangle$, only the first process completes.
- If the schedule is $\langle 3 \rangle$, only the second process completes.

Therefore

- in general, no "correct choice" for a scheduler
- "correct choice" may depend on the future schedule.

Well-behaved Processes

Definition: Well-behaved Process

A process P is well-behaved iff

$$Q \sqsupseteq P \implies Q \leq P$$

- Analysis of real-time systems with an ill-behaved process is not safe
- Instead, when analysing schedulability, replace it with some well-behaved process that is harder to schedule than any process for which the ill-behaved process is an upper bound.

Well-behaved Process Structures

Which processes are well-behaved?

- $\mathbf{0}$ and $\mathbf{1}$ are well-behaved
- PAR/SEQ/1 is well-behaved, e.g.

$$(\mathbf{1} \,;\, \mathbf{1} \,;\, \mathbf{1} \,;\, \mathbf{1}\ldots) \,||\, (\mathbf{1} \,;\, \mathbf{1} \,;\, \mathbf{1}\ldots) \,||\, (\mathbf{1} \,;\, \mathbf{1}\ldots)\ldots$$

- A sequence of well-behaved processes is well-behaved:

$$P_1 \,;\, P_2 \,;\, P_3 \,;\, \ldots \,;\, P_n$$

  is well-behaved if all the P_is are well-behaved.

Example of well-behaved process not with this structure:

$$\big(\mathbf{1} \,; (\mathbf{1} \,||\, \mathbf{1})\big) \,\big|\big|\, \mathbf{1}$$

Safe Upper Bounds

Definition: Safe upper bound

A process Q is a safe upper bound for a process P if

$$\forall P' \sqsupseteq P \colon P' \leq Q$$

Definition: Best safe upper bound

$$Q^{\star} = \min_{\leq} \{Q \in \mathbb{P} \colon \forall P' \sqsupseteq P \colon P' \leq Q\}$$

- A safe upper bounds always exist, e.g. $\mathbf{1} \mathbin{;} \mathbf{1} \mathbin{;} \mathbf{1} \ldots$ with length $\mathcal{C}(P)$ is a safe upper bound of P
- Do not yet know how to find the best safe upper bound, or if it is unique.

Example of Safe Upper Bound

Example: $Q^\star$ is a better SUB than Q_1:

$$P = \big(\mathbf{1}\,;(\mathbf{1}\,||\,\mathbf{1})\big)\,\big|\big|\,\big(\mathbf{1}\,;(\mathbf{1}\,||\,\mathbf{1})\big)$$

$$Q_1 = \mathbf{1}\,;\mathbf{1}\,;\mathbf{1}\,;\mathbf{1}\,;\mathbf{1}\,;\mathbf{1}$$
$$Q^\star = (\mathbf{1}\,||\,\mathbf{1})\,;(\mathbf{1}\,||\,\mathbf{1})\,;(\mathbf{1}\,||\,\mathbf{1})$$

Why?

- Both $Q^\star$ and Q_1 are SUBs of P.
- $P \leq Q^\star \leq Q_1$
- $Q^\star$ completes on all schedules where Q_1 completes.
- Q_1 has suppressed all parallelization.
- $Q_\star$ has only suppressed some structure.

Summary

- CTPs:

$$
\begin{aligned}
P \in \mathbb{P} \iff & P = \mathbf{0} \\
\vee\ & P = \mathbf{1} \\
\vee\ & P = Q\,;R && Q, R \in \mathbb{P} \\
\vee\ & P = Q \,||\, R && Q, R \in \mathbb{P}
\end{aligned}
$$

- Executing a process may make it harder to schedule.
- No perfect intra-job scheduling strategy exists.

- A process that is easier to schedule when computation is removed is well-behaved
- A safe upper bound is a well-behaved upper bound.
- Replacing an ill-behaved process with a safe upper bound enables safe schedulability analysis.

Future Work

- Introduce explicit non-determinism (choice)
  - \+ Allows conditional parallels
  - \+ Allows alternation
  - \+ More elegant definition of $\sqsupseteq$.
  - − Much more complicated definition of step.
- Demand bounds, which are needed for real-time schedulability analysis of systems of CTPs.
- Communication between CTPs (blocking terms).
- Algorithm for finding good, safe upper bounds

Questions