On the temporal robustness of uniprocessor real-time systems scheduled with FP and EDF

Laurent George

Contact: lgeorge@ieee.org

Univ. of Paris 12 / ECE

Outline

1. Introduction
2. Classical FC
3. Sensitivity of WCETs
4. Sensitivity of periods
5. Sensitivity of deadlines
6. Temporal robustness at run time
7. Conclusion

1. Introduction

- Classical feasibility conditions do not consider possible deviations from a problem specification.
  - Study the robustness of the Feasibility Conditions (FC) in the case of such deviations.
- This can be done :
  - In the dimensioning phase
  - At run time
- Sensitivity analysis aims a studying the ability to introduce more flexibility in the specifications, in the dimensioning phase.

1. Introduction (cont'd)

- We consider a problem specified by a task set τ of *n* periodic tasks

- A periodic tasks τ_i (i=1…n) is defined by:
  - C_i : its Worst Case Execution Time (WCET)
  - T_i : its minimum inter-arrival time or period
  - D_i : its relative deadline

- We suppose one-dimension robustness (one task parameter can evolve, the other task parameters are supposed to be constant).

1. Introduction (cont'd)

In classical FCs, the task parameters are supposed to be constant.

- The WCET can be hard to obtain. Obtained by analyzing the code on a given architecture or by measurement. In both cases:

  - The correctness of the WCET is hard to guarantee or at a cost of over-estimating the CPU resources.

  - The duration of a task at run time depends on the condition of execution (type of architecture, type of memory or cache).

- Constant task parameters might no be suitable for any application and should be adapted to the situations at runtime (e.g. a process should be run more frequently in a given situation to obtain more precision **=> impact on the periods**)

1. Introduction (cont'd)

❑ New architecture propose variable speed processor to scale the performances to the required performances and reduce energy consumption when possible **=> impact on the WCETs or on the Periods**

❑ It would be interesting to determine the effect of a change in the task parameters.

- Should the FCs be recomputed ?
- Can we extend the FCs obtained on a given architecture to another one ?

❑ FCs enables us to meet timeliness constraints but other parameters can be considered.

- Output jitter that result in the execution of a task, for control tasks and multimedia applications
- Reducing a deadline can reduce the output jitter if the scheduling is related to deadlines **=> impact on the deadlines**

1. Introduction (cont'd)

The temporal robustness of a real-time system in the case of a change in the tasks parameters: WCET, deadline and period can be studied:

❑ In the dimensioning phase:

The sensitivity analysis aims at defining the acceptable variations in the task configurations (WCETs, periods or deadlines) such that the system is feasible.

- Scaling factor to expand or reduce the tasks parameters. The correction can be applied to one or many tasks.

- Feasibility region (Bini & al in 06)
  => C-space, T-space and D-Space

1. Introduction (cont'd)

- At run-time, where an evolution in the task parameters happens or is provoked, possibly leading to:
  - Execution overruns faults (the WCET is exceeded),
  - Overload situations (the period is smaller or the WCET is exceeded)
  - Deadline miss.

$\Rightarrow$ Algorithms to deal with such situations should be implemented to stabilize the system and ensure its robustness.

2. Classical FC

For FP:

- Based on worst case response time computation
  - *NSC*: Joseph & Pandya 86, Tindell & al 95
    Recursive equations (pseudo polynomial-time complexity)
  - *SC*: Fisher & Baruah 05, George & al 96
    (polynomial-time approximation)
- Based on processor utilization functions
  - *SC*: Liu and Layland 73, Bini & al 03: hyperbolic bound (polynomial-time complexity)
  - *NSC*: Applied to a set of times in the interval [0, Di] for a task τ_i
    * Case Ti=Di: Lehoczky & al 90
    (pseudo-polynomial time complexity)
    * Case Di≤Ti, Bini & Buttazzo 04
    (pseudo-polynomial time complexity)
    => significant reduction of the number of times to consider.
    Used for Sensitivity analysis of FP

2. Classical FC (Cont'd)

For EDF (Baruah & al 90)

- NSC: Processor demand function h(t)≤t for a set of times in [0,L[(pseudo polynomial-time complexity)
  - L is the length of the synchronous busy period
  - $h(t) = \sum_{i=1}^{n} Max\left\{0, 1 + \left\lfloor \frac{t-D_i}{T_i} \right\rfloor\right\} C_i$
- Special case: based on processor utilization when Di≥Ti => U≤1 (polynomial-time complexity) (Lui & Layland 73, Baruah & al 90)
- Worst case response time computation (Spuri 96) (pseudo polynomial-time complexity)

2. Classical FC (Cont'd)

For independent deadlines and periods FP and EDF the FCs have both pseudo polynomial-time complexity => deterministic multidimentional sensitivity analysis might be very costly.

⇒ One-dimension Sensitivity for FP focuses on the case Di≤Ti

3. Sensitivity of WCETs

C-space characterisation (Bini & al 05 for FP)

If ∀ i, Di≤Ti, the C-space is defined as the region such that for any vector C={C1,…Cn} in R+n:

$$\forall i = 1 \ldots n, \exists t \in \mathcal{P}_{i-1}(D_i), W_i(t) \leq t$$

Where:

$$W_i(t) = C_i + \sum_{\tau_j \in hp(i)} \left\lceil \frac{t}{T_j} \right\rceil C_j$$

and

$$\begin{cases} \mathcal{P}_0(t) = t \\ \mathcal{P}_i(t) = \mathcal{P}_{i-1}(\left\lfloor \frac{t}{T_i} \right\rfloor T_i) \cup \mathcal{P}i - 1(t) \end{cases}$$

3. Sensitivity of WCETs (Cont'd)

Maximum scaling factor α such that $\forall i$, $C_i \rightarrow \alpha C_i$ and $D_i \le T_i$

Bini & al 05 Show how to compute λ such that $\alpha=\lambda+1$ for FP

$$\lambda = \min_{i=1\ldots n} \max_{t \in \mathcal{P}_{i-1}(D_i)} \left(\frac{t}{W_i(t)} - 1 \right)$$

- If $\lambda<0$ then the initial task set is not feasible and the WCET must be reduced
- In the general case of Independent Di and Ti, computing α can be costly as the busy periods tends to the lcm(T_1,…T_n) when α is increasing

3. Sensitivity of WCETs (Cont'd)

Maximum scaling factor α for EDF:

L tends to P=lcm(T1,…Tn) when WCETs increase.

- If for only one task τ_i, Ci -> αCi (Balbastre & Rippoll al 02) with α=λ+1

$$\lambda = \frac{1}{C_i} \max_{0<t<P} \left(\frac{t - h(t)}{\left\lfloor \frac{t+T_i-D_i}{T_i} \right\rfloor} \right)$$

- If for all tasks τ_i, Ci -> αCi (Hermant & George 07)

$$\alpha = \frac{1}{Max\left\{U, Sup_{t\in[Min\{D_1,\ldots,D_n\},P)}\left\{\frac{h(t)}{t}\right\}\right\}}$$

3. Sensitivity of WCETs (Cont'd)

Maximum scaling factor α for EDF (Cont'd):

Example with a configuration A:

τ_1={C_1^A=40, T_1^A=70, D_1^A=50}

τ_1={C_1^A=60, T_1^A=110, D_1^A=70}

τ_1={C_1^A=100, T_1^A=130, D_1^A=100}

$$U^A = C_1^A/T_1 + C_2^A/T_2 + C_3^A/T_3 = 1.89$$

NOT FEASIBLE !

But $\alpha = \dfrac{1}{Max\left\{U, Sup_{t\in[Min\{D_1,\ldots,D_n\},P)}\left\{\frac{h(t)}{t}\right\}\right\}}$ =1/2

⇒Feasible with a configuration B where C_1^B=20, C_2^B=30 and C_3^B=50 (periods and deadlines are unchanged))

4. Sensitivity of the periods

- For EDF: still an open problem

T-space

- For FP, Bini & al in 05 when forall i, Di≤Ti show that:
  - no linear inequalities in a closed form can be obtained to express the T-space
  - the T-space is composed of an infinite number of hyperplanes

Relation between the T-space and the C-space (Bini & al 05)

- Let τ^C be the task set where all the WCETs are multiplied by a scaling factor α. Let τ^T be the task set where all the periods is divided by α.

The task set τ^C is feasible if and only if the task set τ^T is feasible.

4. Sensitivity of the periods (Cont'd)

Computation of the minimum feasible period T_i^{min} of a task τ_i

With T_i^{min}, we have either (Bini & al 05, 06):

- $T_i^{min} = \frac{r_i}{\delta_i}$ where $0 < \delta_i = \frac{D_i}{T_i}$

or

- There exists a lower priority in lp(i) such that the worst case response time rj of τ_j is an integer multiple of T_i^{min} with n_j^i interferences of τ_i

$$r_j(n_j^i) = C_j + n_j^i C_i + \sum_{\tau_k \in hp(j), k \neq i} \left\lceil \frac{r_j(n_j^i)}{T_k} \right\rceil C_k$$

- Finally, we have:

$$T_i^{min} = max \left\{ \frac{r_i}{\delta_i}, \max_{j \in lp(i)} \min_{n_j^i = 1 \dots N_j^i} \frac{r_j(n_j^i)}{n_j^i} \right\}$$

5. Sensitivity of the deadlines

Computation of the minimum deadline D_i^{min} of a task τ_i

❑ Balvastre & al [28]: Let t' be the modified task set where task τ_i has for deadline:

$$D_i^{min} = max_{j=1}^{k_i}(d'_{i,j} - s_{i,j})$$

Where $d'_{i,j} = h(t_1) + C_i$ *and* t_1 *is the smallest value if any such that* $s_{i,j} + C_i \leq t_1 \leq d_{i,j}$ *and* $h(t_1) > t_1 - C_i$, $d'_{i,j} = C_i$ *else. Then* τ' *is feasible.*

❑They show that D_i^{min} is also the worst case response time of τ_i

5. Sensitivity of the deadlines (cont'd)

Computation of the maximum deadline reduction factor α (applied to all the deadlines):

$\tau = \{\tau_1, \ldots, \tau_n\}$: **task set**;

L ← compute-L(τ) : **integer**; $\alpha \leftarrow 1$: **real**

$slack = min_{t \in [0,L)}(t - h(t))$: **real**;

While $(slack \neq 0)$ **do**

  $\alpha = min_{i=1 \ldots n}(1 - \frac{slack}{D_i})$;

  For $(i = 1; i < n; i++)$ **do**

    $D_i = \alpha D_i$;

  end For

  $slack = min_{t \in [0,L)}(t - h(t))$;

done

Return α;

6. Temporal robustness at run time

What happen if a deviation from a problem specification occurs ?

Can we allow some flexibility in the system to deal with WCETs or periods deviations at run time ?

Deviations of WCETs:

- Possible execution overruns at run time, i.e task durations exceeding their estimated WCET but more CPU resources left to deal with such a situation.
- If not correctly handled, a WCET deviation might lead to a deadlinemiss.
- The value of the WCETs can also be influenced by the occurrence of faults in the system.
- The use of scalable architectures: Some systems propose variable speed processor able to run applications at different frequencies. Reducing the frequency of a processor will increase the WCETs but might result in system overloads and possibly deadline miss.

6. Temporal robustness at run time (Cont’d)

- **Variable speed processor:**
  - Can result in overloads => Adapt the periods in the case of overloads to come back to a normal load condition.
  - Can be used for power constraints but discrete frequency

Dealing with WCET overruns:

An execution overrun fault does not necessarily means a deadline miss.

With enough free CPU resources, a system can self stabilize and still meet the deadlines of all the tasks.

The problem is to determine how long the execution overrun can be allowed => Sensitivity analysis

6. Temporal robustness at run time (Cont'd)

Problem: how to detect when execution overruns occur ?

- ❑ Few systems propose a CostOverrun handler:
This requires being able to determine for every task, at any time, how much CPU has been used, to detect that the task duration exceeds its prescribed WCET

- ❑ Many systems do not provide CostOverrun handlers but have only classical timers. A CostOverrun is then detected at a deadlinemiss (very pessimistic as a deadline miss of a task might cause cascading effects on lower priority tasks).

- ❑ Another approach introduced by Bougueroua & al. 07 is to compute the worst case Latest Execution Time (LET) of a task used to set the timer of a task.
  - The extra duration= allowance
  - Based on a sensitivity analysis

6. Temporal robustness at run time (Cont'd)

FP case

The LET is computed at each task activation of a task τ_i released at time t_i

The LET of τ_i and the LET of lower priority than τ_i are updated as follows (FP case):

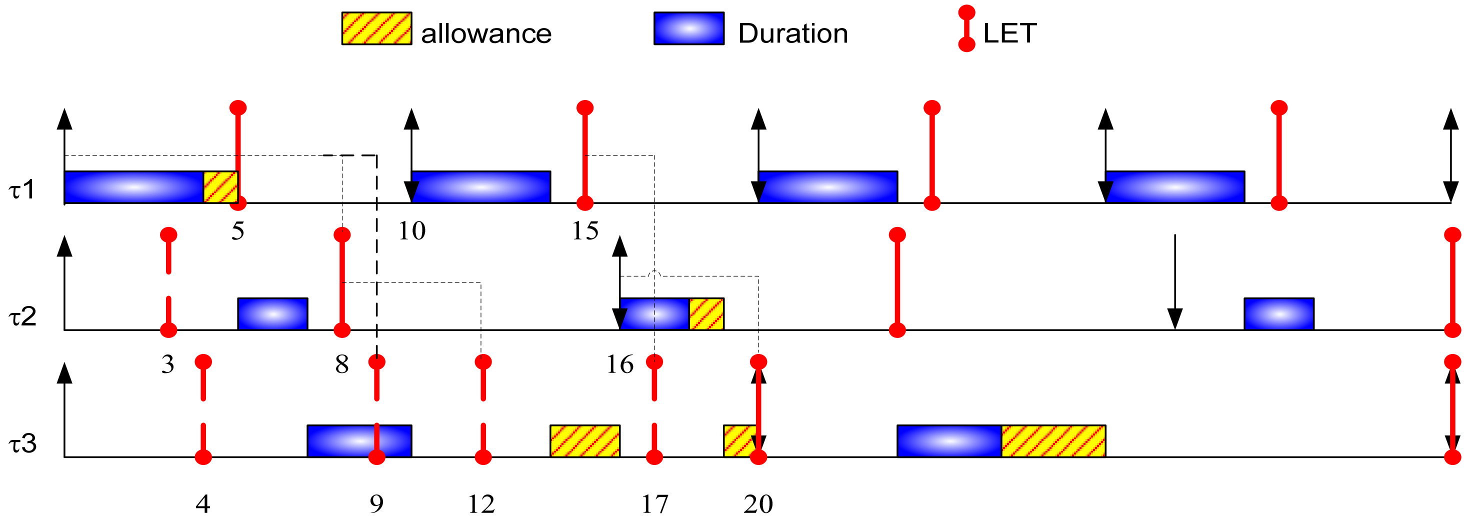
$$\begin{cases} LET_i(t_i) = \max(t_i, \max_{\tau_j \in hp^R(i)} (LET_j(t_j)) + C_i + A_i \ldots\ldots\ldots\ldots(1) \\ \forall \tau_j \in lp^R(i): \quad LET_j(t_j) = LET_j(t_j) + C_i + A_i \ldots\ldots\ldots(2) \end{cases}$$

With the dynamic LET a task can collect the unused Allowance of higher priority tasks

The LET grants the isolation of faults

6. Temporal robustness at run time (Cont'd)

Task	Ci	Di	Ti	Pi	Ai	LETi(0)
$\tau 1$	4	10	10	10	1	5
$\tau 2$	2	16	16	5	1	3
$\tau 3$	3	20	20	1	1	4



Example of dynamic LET

$t=10$: LET of τ_3 => 17

$t=16$: LET of τ_3 => 20

6. Temporal robustness at run time (Cont'd)

Dealing with deadline miss:

A deadline miss can be due to the following cases:

❑ The deviation oft the WCET of a task is too important => overload situation

Solutions:

❑ Stop the faulty task or put it in background

❑ Select tasks to remove : Locke 86, Robust-EDF (Baruah & al 93),

D-over (Koren & al 02)

❑ In the case of a variable speed processor, reducing the frequency can create overloads that can result in deadline miss.

=> Adapt the periods to come back to a normal load

6. Temporal robustness at run time (Cont’d)

Dealing with deadline miss (Cont’d):

Buttazzo & al 02 introduce the spring model and the elastic model to adjust the task periods:

- ❑ The flexibility of a task is modeled as a pring able to increase or decrease its lenght according to workload conditions.
- ❑ The length of a spring is associated to the current processor utilization of its tasks.
- ❑ The spring has an elastic coefficient E_i. The greater E_i,the more elastic the task.
- ❑ Decreasing processor utilization result in applying a compression force on the spring that results in a period decrease.
    - Requires polynomial time FCs based on processor utilisation to adapt the spring in real-time

7. Conclusion

- Temporal robustness can be investigated:
  - In the dimensioning phase => Sensitivity analysis
  - At run time to stabilize a real system, for fault prevention
- Introduces more flexibility for the conception of real-time systems
- Better use of system resources
- Temporal robustness mechanism are considered for automotive applications (Autosar specification)
- New mechanism providing temporal robustness must be implemented in real system

Open issues: sensitivity analysis in multiprocessor plaforms, deterministic multidimension sensitivity