The Cost of Harmonicity

Anton Cervin

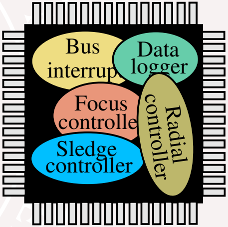
Background

How to optimally share a computing platform between n periodic tasks?

Each task i is described by:

- Period, T_i
- Execution time, C_i
- (Implicit) deadline, $D_i = T_i$

Total utilization: $U = \sum_{i=1}^{n} \frac{C_i}{T_i}$



Two optimal scheduling algorithms

[Liu & Layland, 1973]

Rate-monotonic scheduling:

- Fixed task priorities
- Schedulability bound (sufficient): $U_b = n(2^{1/n} - 1)$

Earliest-deadline-first (EDF) scheduling:

- Dynamic task priorities
- Schedulability bound (exact): $U_b = 1$

Optimal task period assignment

[Seto *et al.*, 1996]

- The performance of each task is characterized by cost function, $J_i(T_i)$
- Period assignment: Solve the optimization problem

$$\min_{T_1,\ldots,T_n} \sum_{i=1}^{n} J_i(T_i)$$
$$\text{s.t. } U \leq U_b$$

Example: Affine cost functions

[Eker *et al.*, 2000], [Cervin *et al.*, 2002]

Assume that the cost of each task is described by

$$J_i(T_i) = v_i + w_i T_i$$

The optimal periods are then given by

$$T_i^* = \sqrt{\frac{C_i}{w_i}}\ \frac{\sum_j \sqrt{w_j C_j}}{U_b}$$

Harmonic task periods

[Real-time systems folklore]

Harmonic periods: $\forall i,j$: $\dfrac{T_i}{T_j} \in \mathbb{N}$ or $\dfrac{T_j}{T_i} \in \mathbb{N}$

Advantages:

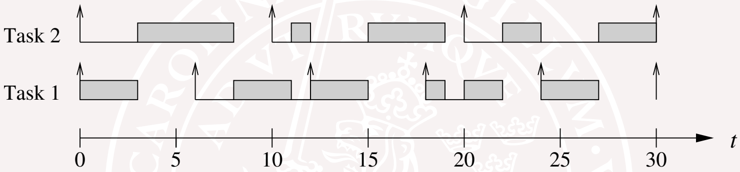
- $U_b = 1$ also under rate-monotonic scheduling
- Constant execution times $\Rightarrow$ no jitter
- Short hyperperiod

Disadvantage:

- Must deviate from the “optimal” periods

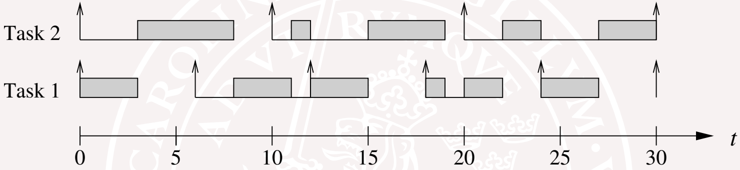
Example

EDF scheduling with two tasks, $T_1 = 6$, $T_2 = 10$, $C_1 = 3$, $C_2 = 5$:

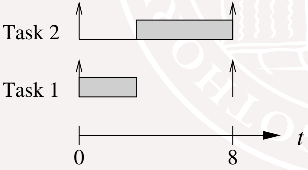


Example

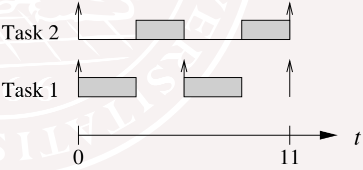
EDF scheduling with two tasks, $T_1 = 6$, $T_2 = 10$, $C_1 = 3$, $C_2 = 5$:



Two different ways two harmonize them:



$T_1 = T_2$



$T_2 = 2T_1$

Questions

1. How to perform the harmonization?
2. What is the cost of deviating from the optimal task periods?

Algorithm 1 – Simple harmonization

[Morteza *et al.*, 2016]

Assume set of increasing optimal periods $T^* = [T_1^* \ \ldots \ T_n^*]$

$T_1 \leftarrow T_1^*$
for $i \leftarrow 2 \ldots n$ **do**
    $T_i \leftarrow \left\lceil \frac{T_i^*}{T_{i-1}} \right\rceil T_{i-1}$
end for
/* Rescale to obtain $U = 1$ */
$U \leftarrow \sum_{i=1}^{n} C_i / T_i$
$T \leftarrow UT$

Theorem

[Morteza *et al.*, 2016]

Assume $U_b = 1$ and linear cost functions, $J_i = w_i T_i$.

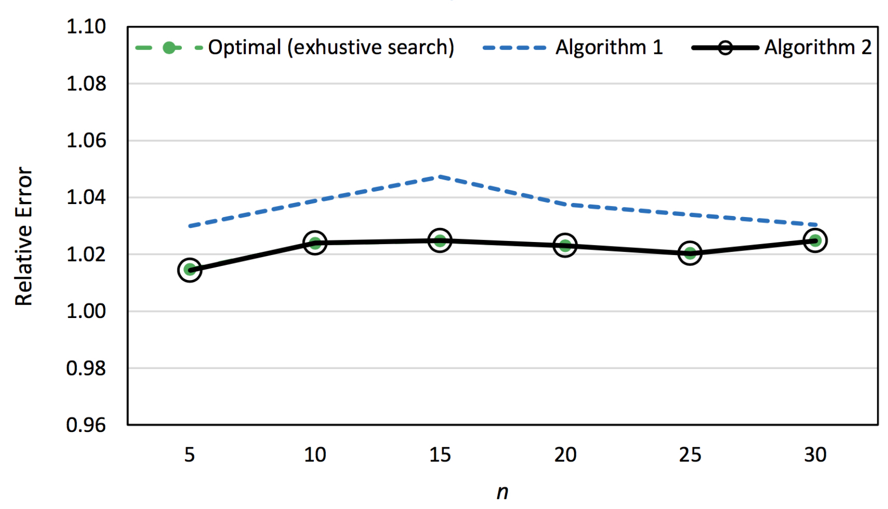
The relative cost of applying Algorithm 1 is smaller than 2:

$$\frac{J}{J^*} = \frac{\sum_{i=1}^{n} w_i T_i}{\sum_{i=1}^{n} w_i T_i^*} < 2$$

(Proof: In the worst case, each period (except the first one) doubles, which doubles the cost. Rescaling does not make things worse.)

Average results on synthetic task sets

[Morteza *et al.*, 2016]



A stronger theorem

The worst-case relative cost of Algorithm 1 is $\frac{9}{8} = 1.125$

Proof

Scale the weights w_i so that

$$J^* = \sum_{i=1}^{n} w_i T_i^* = 1$$

implying

$$U_i^* = \frac{C_i}{T_i^*} = \sqrt{w_i C_i} = w_i T_i^* = J_i^*$$

Harmonizing, the period of each task $2, \ldots, n$ is extended as

$$\hat{T}_i = (1 + \beta_i) T_i^*, \quad 0 \leq \beta_i \leq 1$$

After rescaling, the final cost becomes

$$J = \underbrace{\left(1 + \sum_{i=2}^{n} \beta_i U_i^*\right)}_{\text{extension}} \underbrace{\left(1 - \sum_{i=2}^{n} \frac{\beta_i}{1+\beta_i} U_i^*\right)}_{\text{rescaling to } U = 1}$$

Proof, cont'd

$$J = \left(1 + \sum_{i=2}^{n} \beta_i U_i^*\right)\left(1 - \sum_{i=2}^{n} \frac{\beta_i}{1+\beta_i} U_i^*\right)$$

This function is maximized when $\beta_i = 1$ and $\sum_{i=2}^{n} U_i^* = \frac{1}{2}$, yielding the worst-case cost

$$J_{wc} = (1 + \tfrac{1}{2})(1 - \tfrac{1}{4}) = \tfrac{9}{8}$$

Conjecture – optimal harmonization

The worst-case relative cost of an *optimally* harmonized task set with linear cost functions is

$$J^*_{wc} = \frac{1}{2}(\ln 2)^{-2} \approx 1.041$$

(Optimal: Exhaustive search among all possible harmonizations to find the one with the smallest cost.)

Algorithm for optimal harmonization

for $\forall \alpha \in [\frac{1}{2}, 1]$ **do**
    $T'_0 \leftarrow \alpha T^*_1$
    for $i \leftarrow 1 \ldots n$ **do**
        $T'_i \leftarrow \left\lceil \frac{T^*_i}{T'_{i-1}} \right\rceil T'_{i-1}$
    end for
    // Rescale to full utilization
    $U \leftarrow \sum_{i=1}^{n} C_i / T'_i$
    $T' \leftarrow U T'$
    $J \leftarrow \sum_{i=1}^{n} w_i T'_i$
    if $J < J_{min}$ **then**
        $T \leftarrow T'$
    end if
end for

(Conjecture: A factorial number of α values need to be tested)

Conjecture – optimal harmonization

The worst case occurs when

$$C_i = 2^{\frac{i-1}{n}}, \quad w_i \propto \frac{1}{C_i}$$

Example ($n = 2$): $\quad C_1 = 1,\ C_2 = \sqrt{2},\ T_1^* = 2,\ T_2^* = 2\sqrt{2}$

$$\Rightarrow\ \frac{J}{J^*} = \frac{4 + 3\sqrt{2}}{8} \approx 1.030$$

Conjecture – optimal harmonization

For a given n, the worst-case relative cost is

$$J_{wc}^*(n) = \frac{1}{2n^2\left(2^{\frac{1}{n}} + 2^{-\frac{1}{n}} - 2\right)}$$

n	J_{wc}^*
2	1.030
3	1.036
4	1.038
∞	1.041

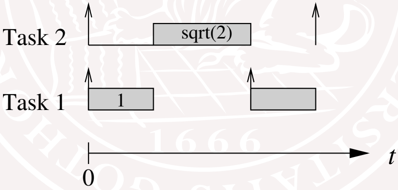
Relation to rate-monotonic schedulability

Worst-case scenario for harmonization

$\Leftrightarrow$

Worst-case scenario for rate-monotonic schedulability

Example ($n = 2$):



Corollary

Harmonizing a rate-monotonic-schedulable set of periodic tasks with linear cost functions always yields a **lower** cost (even with Algorithm 1)

Conclusion

General recommendation: If possible, choose harmonic periods for your tasks.

- The cost of harmonization is very small
- The gain in terms of rate-monotonic schedulability is large
- Everything, from scheduling analysis to control design, becomes easier with harmonic periods