Scuola Superiore
Sant'Anna

Periodic Charging Scheme for Fixed-Priority Real-Time Systems with Renewable Energy

Mario Bambagini and Hakan Aydin

IEEE International Symposium on Industrial Embedded Systems (SIES 2014)

Retis
Real-Time Systems Laboratory

Introduction

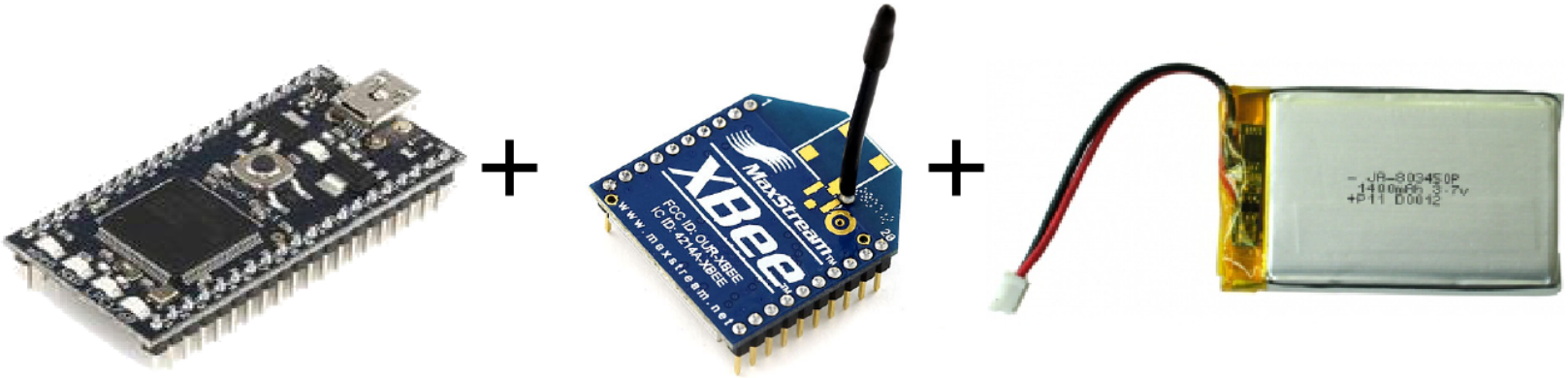
This paper presents an energy harvesting scheduling algorithm (PCS) for periodic real-time fixed-priority tasks on single core

Main contributions:

- more deterministic and predictable
- realistic power model
- low online complexity: linear
- non-real-time tasks are involved into the analysis

The number of feasible task sets is significantly increased

Energy harvesting problem



Board: ~1W + Radio: ~1W + Battery: 1.4Ah

NO energy-aware algorithms $\Rightarrow$ lifetime $\approx$ 4 hours

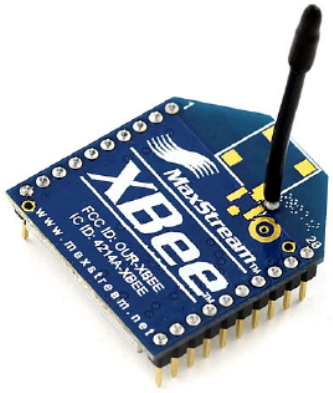
Energy-saving algorithm $\Rightarrow$ lifetime $>>$ 4 hours

©2014 Scuola Superiore Sant'Anna

Energy harvesting problem

Board: ~1W

+

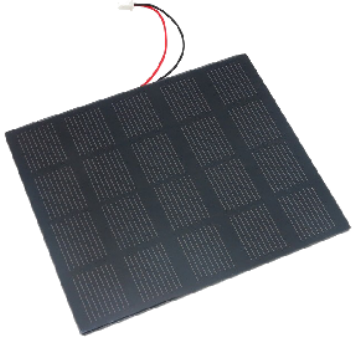


Radio: ~1W

+

Battery: 1.4Ah

+

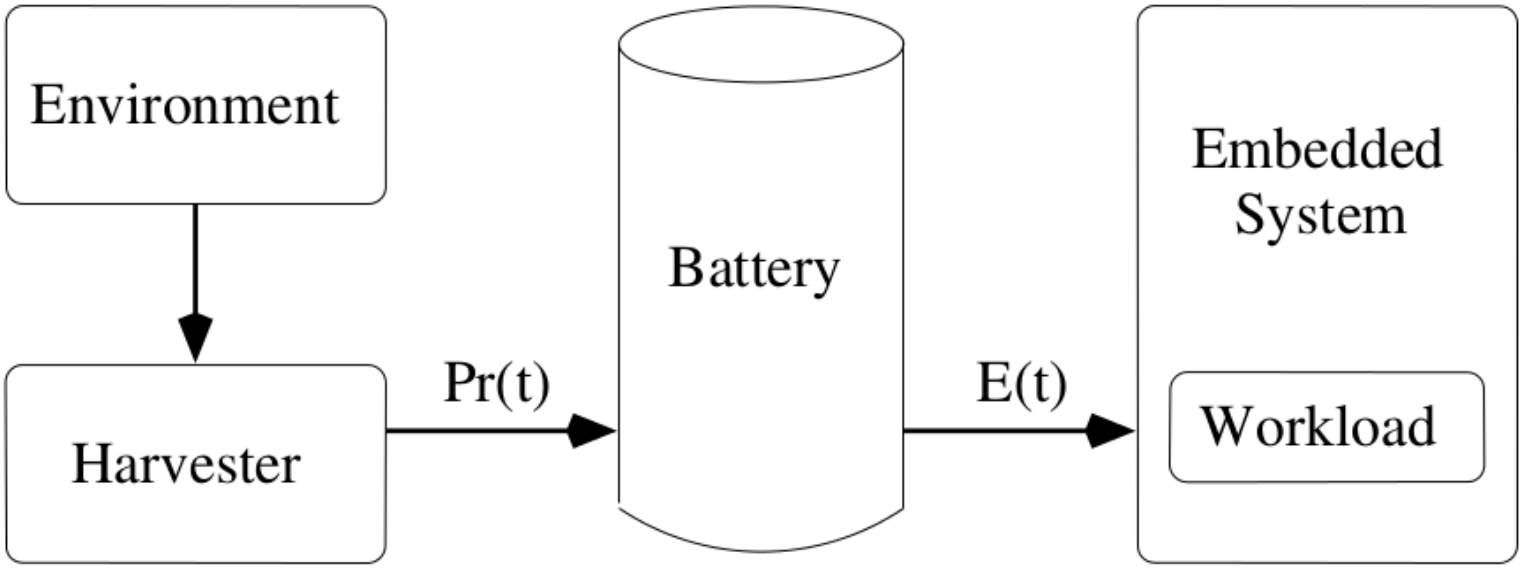


Solar panel: [0, 0.5]W

Energy neutrality $\Rightarrow$ lifetime $\approx \infty$

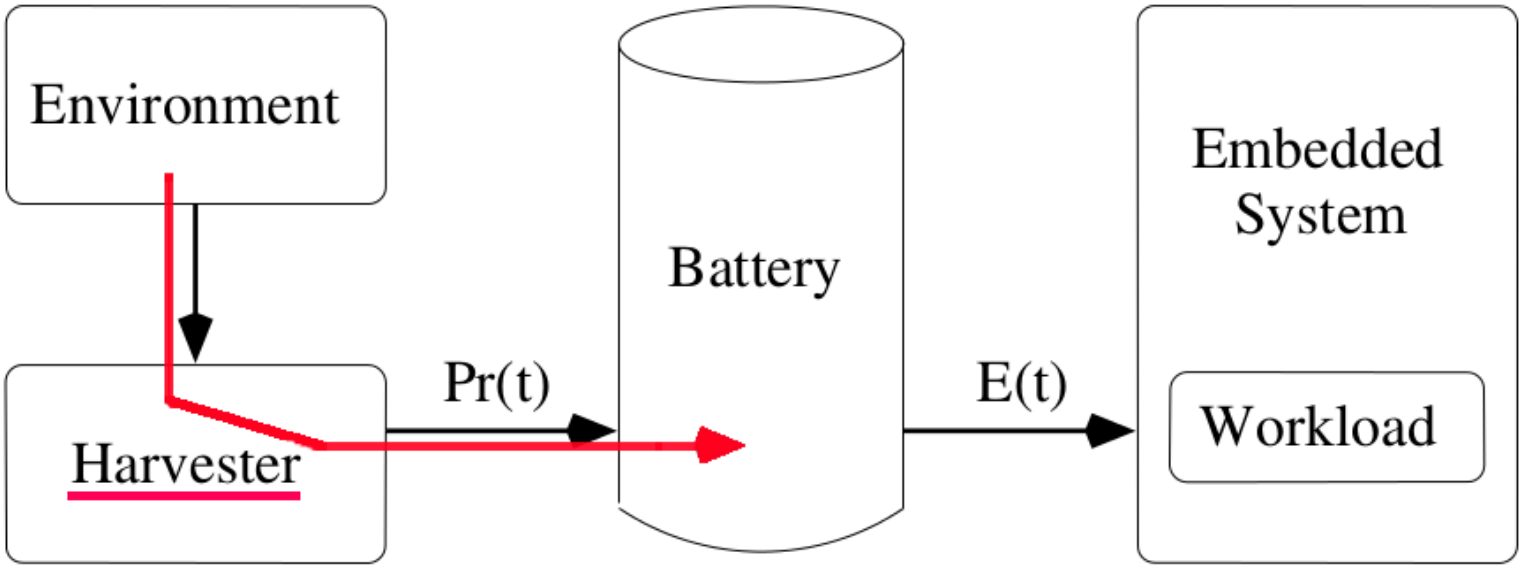
©2014 Scuola Superiore Sant'Anna

System model

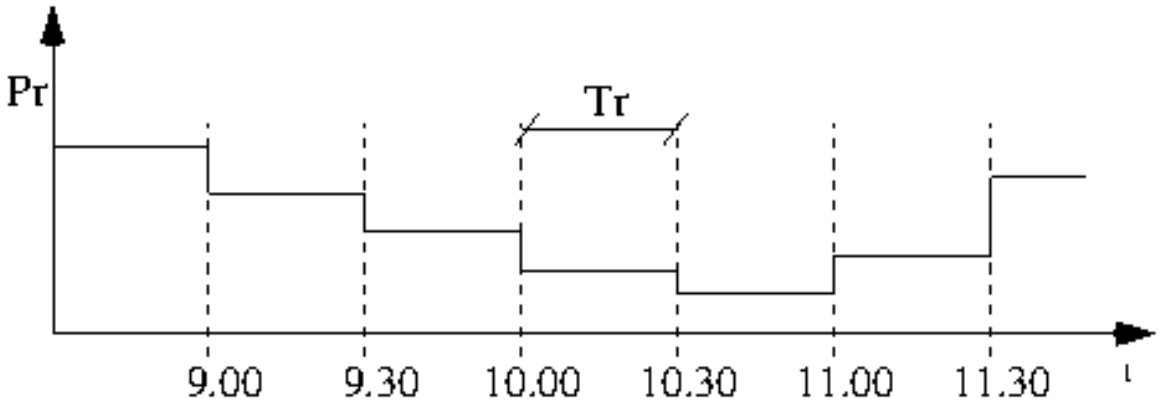


©2014 Scuola Superiore Sant'Anna

System model

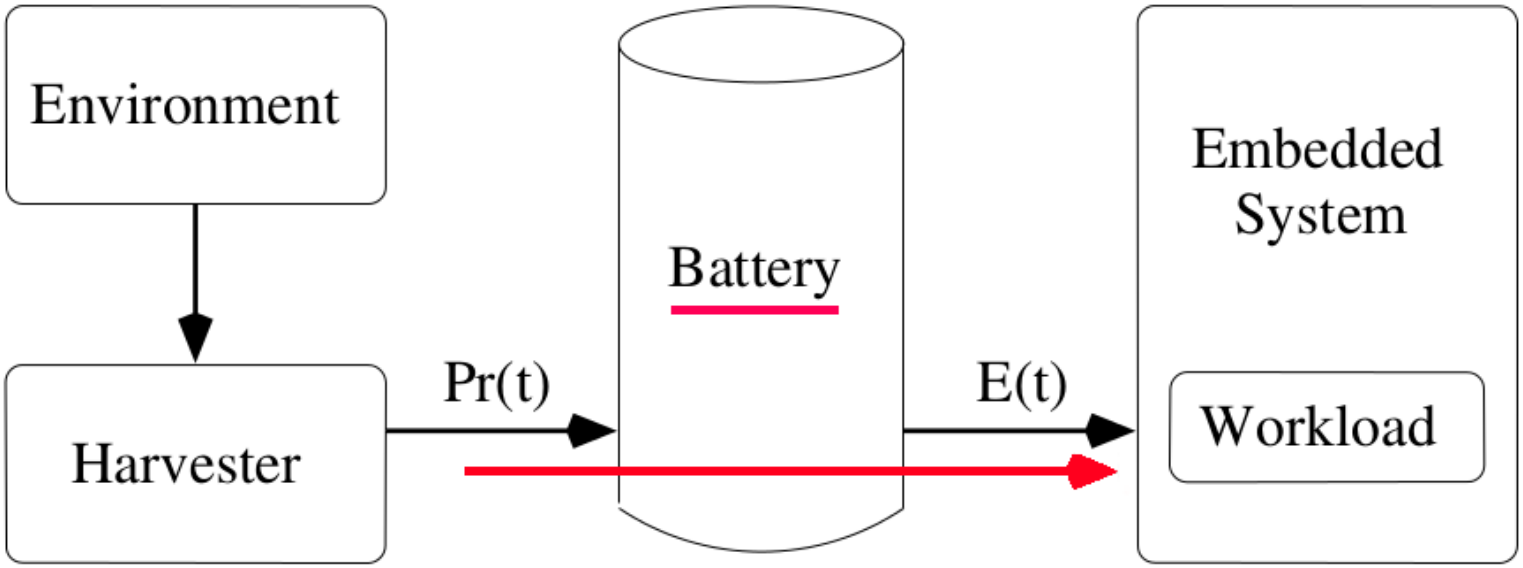


$P_r(t)$: replenishment function



©2014 Scuola Superiore Sant'Anna

System model

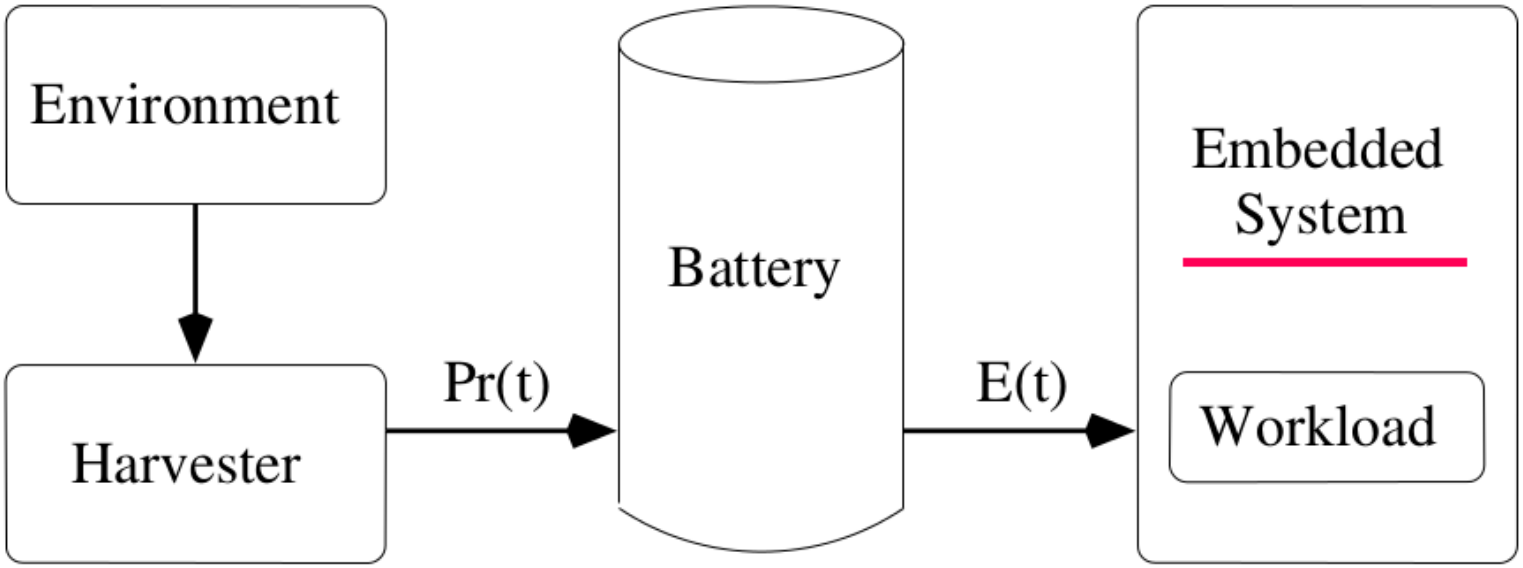


Energy level of the battery at time t: $E(t) \in [0, C]$

- it collects energy from the Harvester
- it provides energy to the Embedded System

©2014 Scuola Superiore Sant'Anna

System model

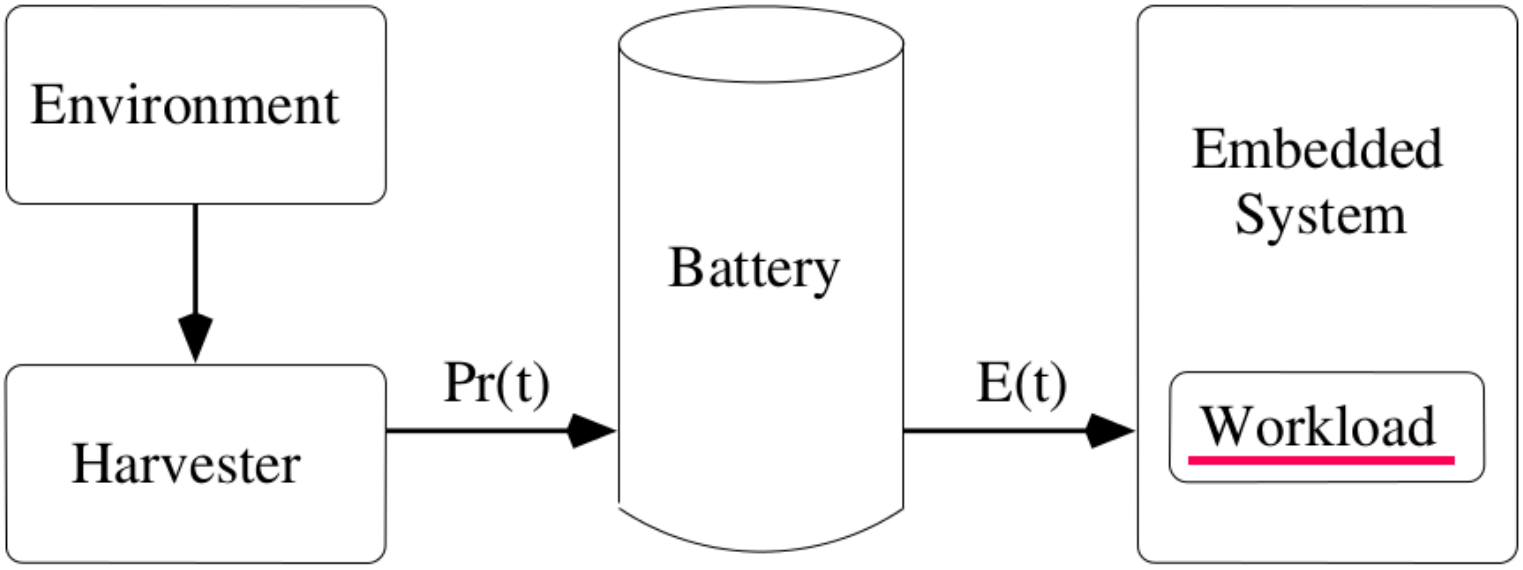


Embedded systems: single-core processor + peripherals

- active state σ_0: $P_{CPU} \leq P_{\sigma_0} \leq P_{CPU} + P_{dev}$
- m low-power states σ_j: P_{σ_j}, ξ_{σ_j} (transition overhead)

©2014 Scuola Superiore Sant'Anna

System model



Workload: n sporadic independent tasks $\{\tau_1, \tau_2, ..., \tau_n\}$

- $\tau_i \rightarrow C_i$ (WCET), T_i (Period), D_i (Deadline), P_i (Power)
- implicit deadlines ($\forall \tau_i : \ C_i = T_i$)
- fixed priorities assigned according to RM

©2014 Scuola Superiore Sant'Anna

Algorithm

State of the art techniques:

- energy-greedy algorithms
- computation-greedy algorithms

Our proposal $\Rightarrow$ add periodic charging phases:

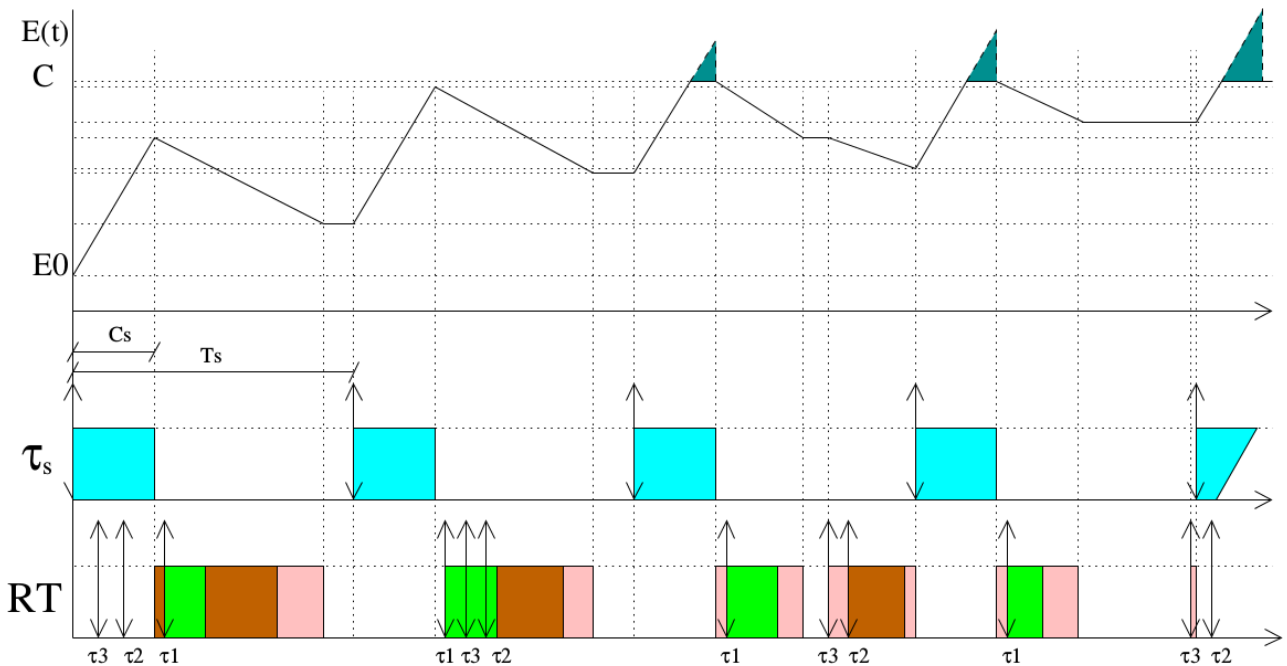
- *PCS*: introduces a new fictitious periodic task τ_s
  - at DESIGN time: computing the parameters of τ_s
  - at RUN time: compacting idle intervals
- PCS^*: offline sufficient (very pessimistic) test
- PCS^{NRT}: non-real-time enhancement

©2014 Scuola Superiore Sant'Anna

Algorithm - *PCS* - design time

PCS at design time: computing the parameters of τ_s

- $T_s = minT_i$ $\Rightarrow$ τ_s has the highest priority
- C_s: sensitivity analysis $\Rightarrow$ longest continuous budget
- low-power state: the deepest one whose $\xi_{\sigma_k} \leq C_s$

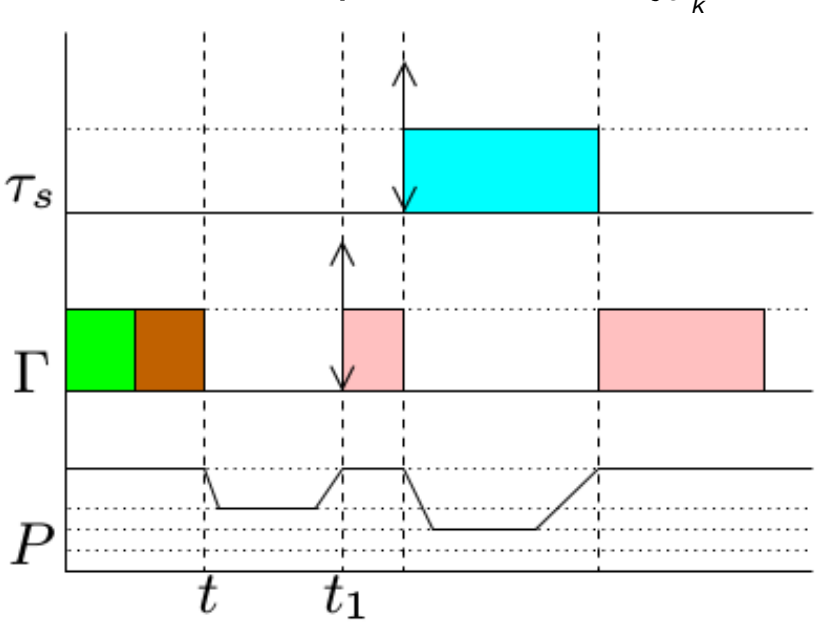


Offline complexity: pseudo-polynomial

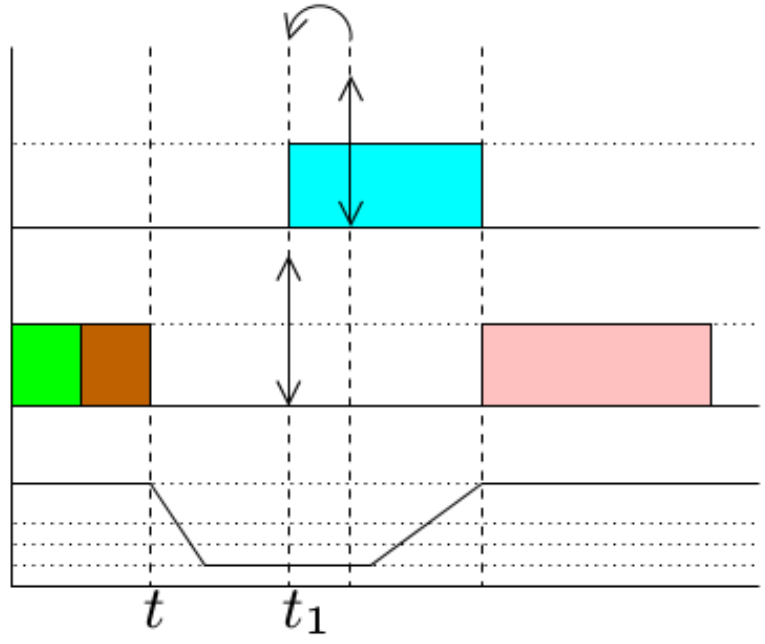
Algorithm - *PCS* - run time

PCS at run time: compacting idle intervals — anticipate τ_s

- when an idle happens at t
  - t_1 = next task arrival
  - make τ_s arrive at t_1
  - $T_{charge} = C_s + (t_1 - t)$
  - low-power state: $\xi_{\sigma'_k} \leq T_{charge}$



without online step



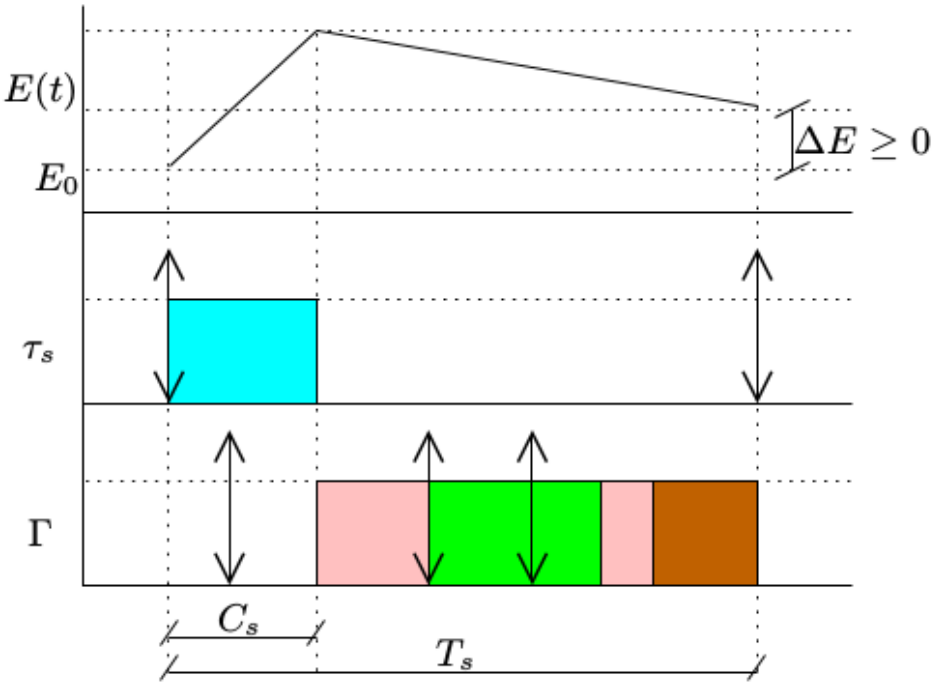
with online step

Online complexity: $O(m)$

©2014 Scuola Superiore Sant'Anna

Algorithm - *PCS**

Check the worst interval T_s: the harvested energy during C_s must be not lower than the consumed energy in $T_s - C_s$



$$C_s \geq \frac{\max_{\tau_i} P_i - P_r}{\max_{\tau_i} P_i - P_{\sigma_k}} Ts$$

Note: $T_r >> T_s$

Offline complexity: constant

Very pessimistic: the highest consume is assumed for $T_s - C_s$

©2014 Scuola Superiore Sant'Anna

Algorithm - PCS^{NRT}

Target: improve the responsiveness of non-real-time tasks while fully refilling the battery by the end of actual T_r interval

Note: NRT tasks are scheduled in a best-effort manner

Solution: skipping 1 out of j consecutive instances of τ_s, yielding the CPU to the workload

$$j = \left\lfloor \frac{(P_{act} - P_r) \cdot T_s + \frac{C - E_0}{T_r} T_s}{(P_{act} - P_{\sigma_k}) \cdot C_s - (P_{act} - P_r) \cdot T_s - \frac{C - E_0}{T_r} T_s} \right\rfloor$$

Online complexity: constant

©2014 Scuola Superiore Sant'Anna

Results

Experimental results were obtained through simulations:

- UUniFast algorithm: $T_i \in [40, 500]$ms
- $P_i = P_{CPU} + x_i \cdot P_{dev}$, random $x_i \in [0.0, 1.0]$
- 4000 task sets per test

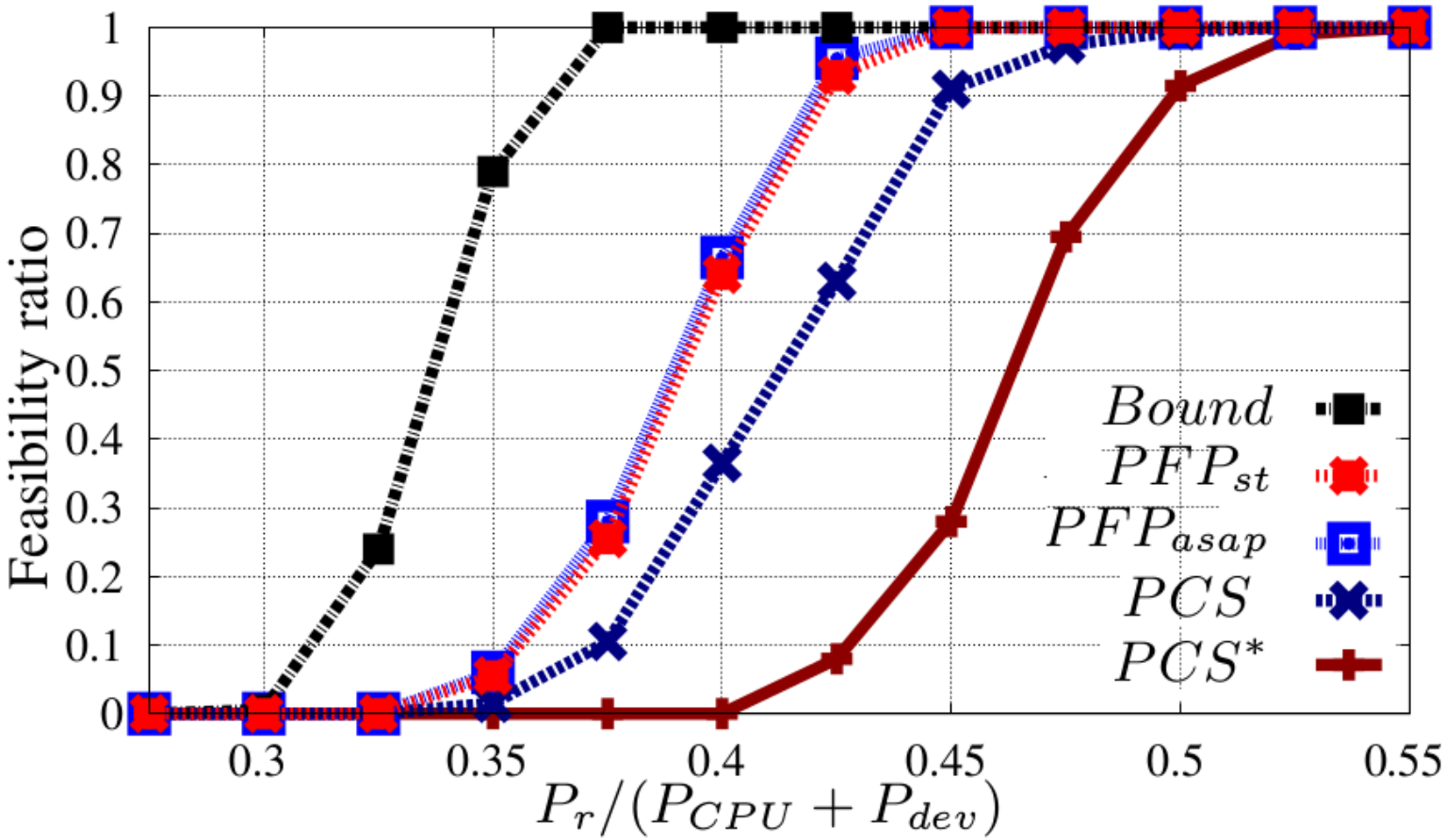
Power model (from NXP LPC1768 — ARM Cortex M3)

- active state: $P_{CPU} = 690$mW, $P_{dev} = 310$mW
- idle state: $P_{\sigma_1} = 490$mW, $\xi_{\sigma_1} \approx 0$ms
- sleep state: $P_{\sigma_2} = 290$mW, $\xi_{\sigma_2} \approx 15$ms
- Battery capacity: $C = 250$mAh

©2014 Scuola Superiore Sant'Anna

Results - Feasibility ratio vs. P_r

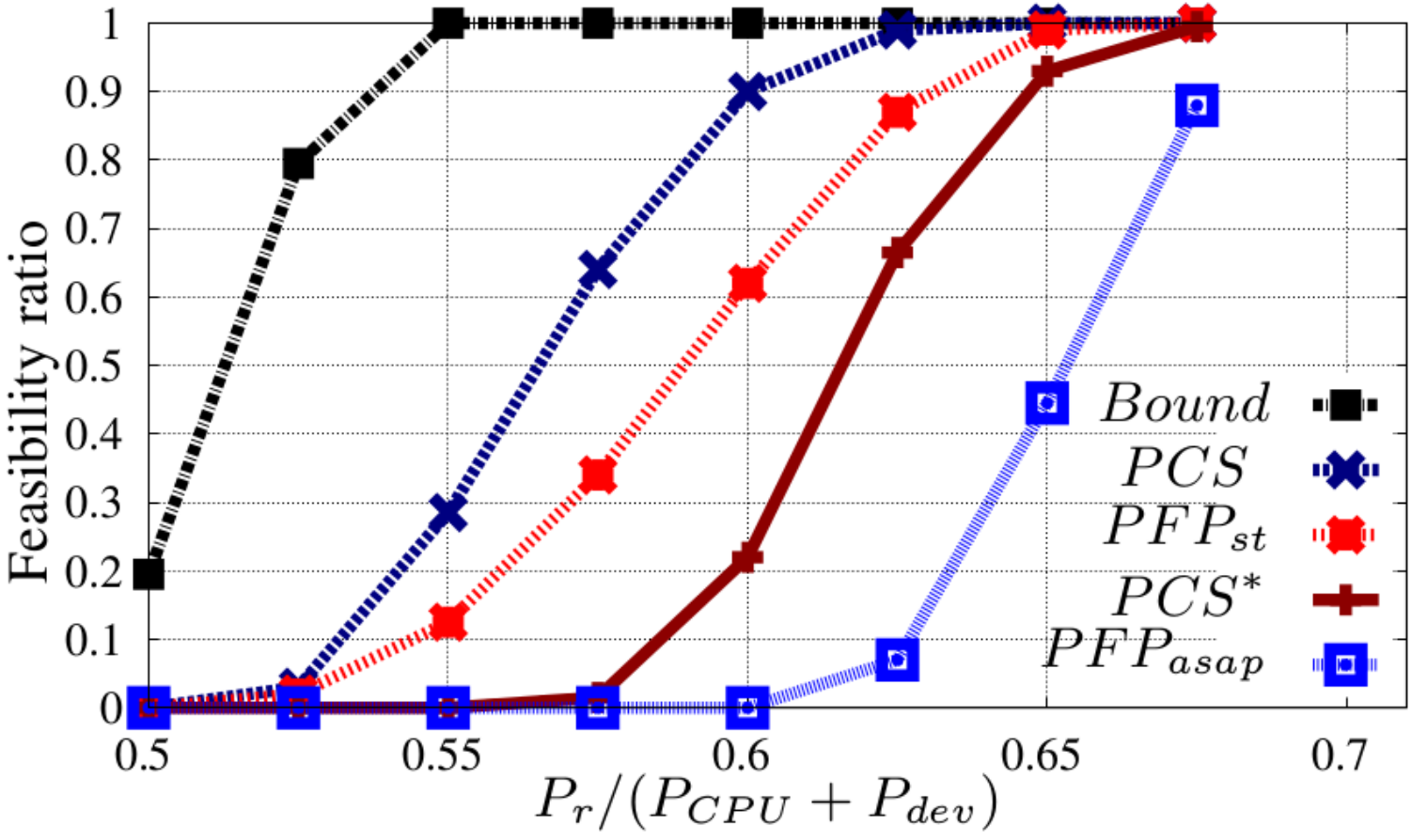
Ideal power model ($\xi_{\sigma_2} = 0$ms and $P_{\sigma_2} = 0$W) and $U = 0.4$



©2014 Scuola Superiore Sant'Anna

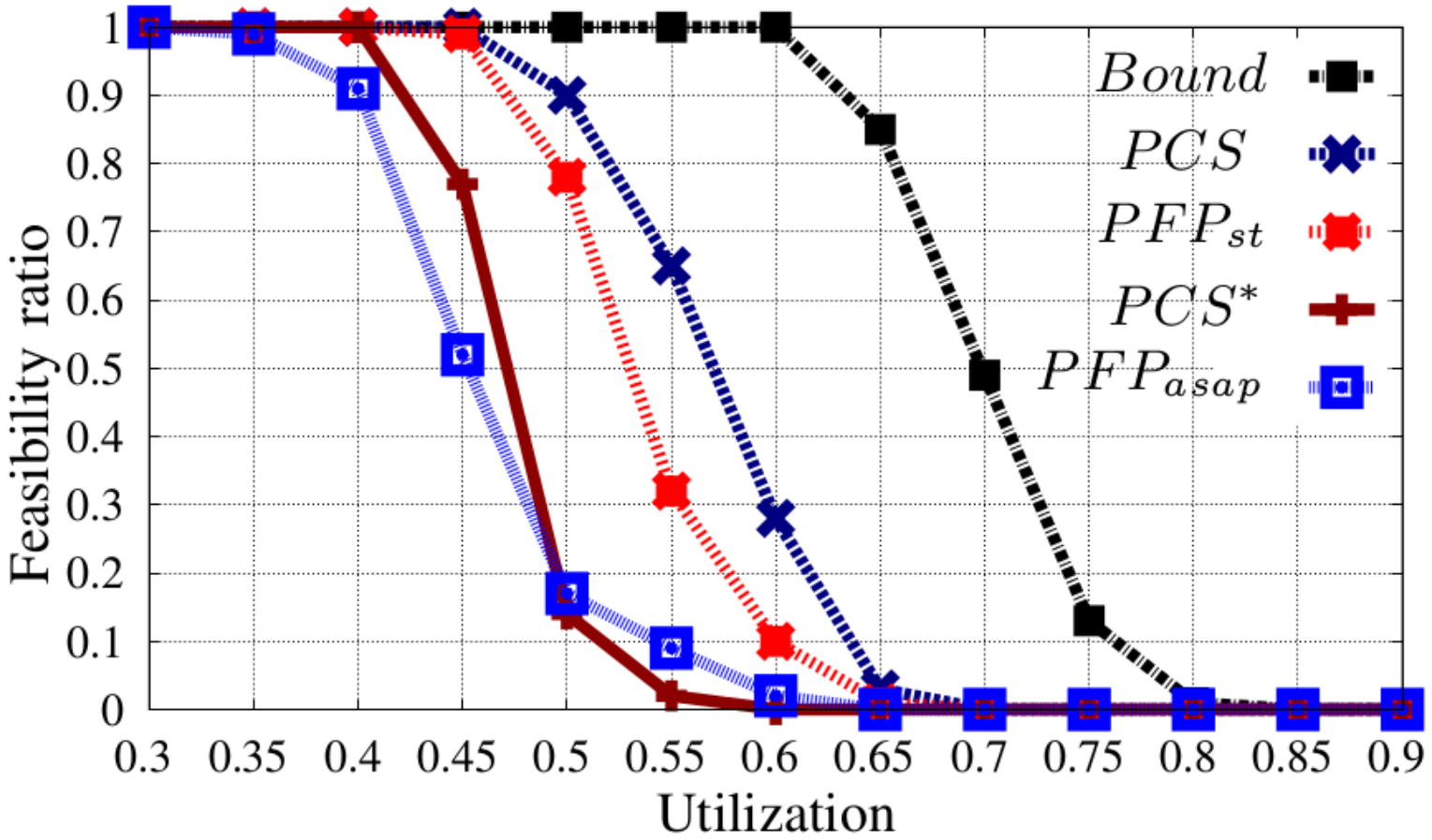
Results - Feasibility ratio vs. P_r

Realistic power model and $U = 0.4$



Results - Feasibility ratio vs. U

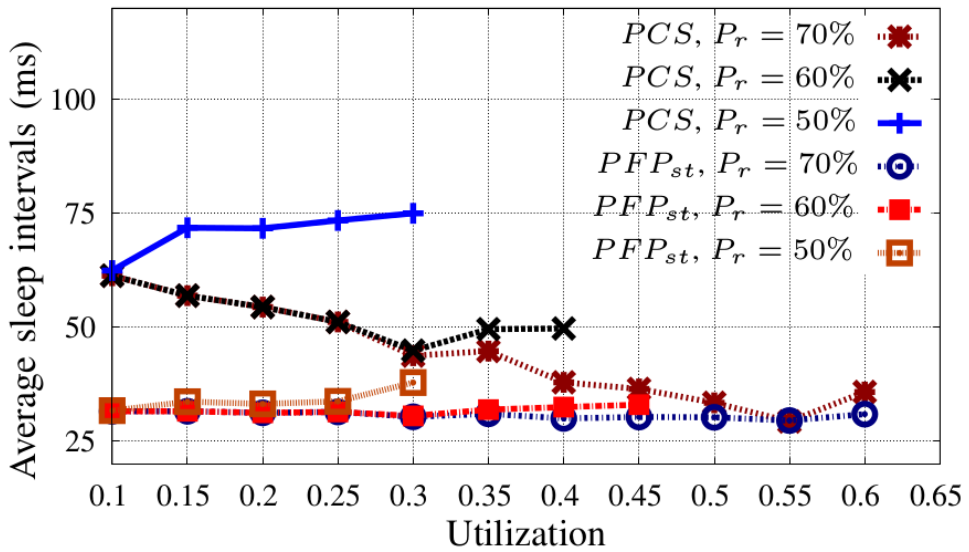
Realistic power model and $P_r = 70\%$ wrt $P_{CPU} + P_{dev}$



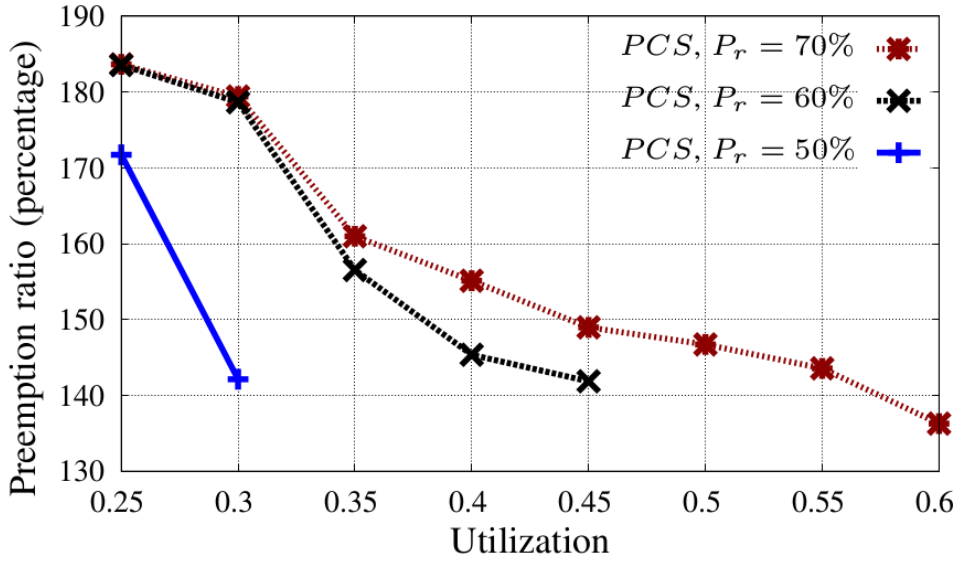
©2014 Scuola Superiore Sant'Anna

Results - Online metrics

Longer sleep intervals



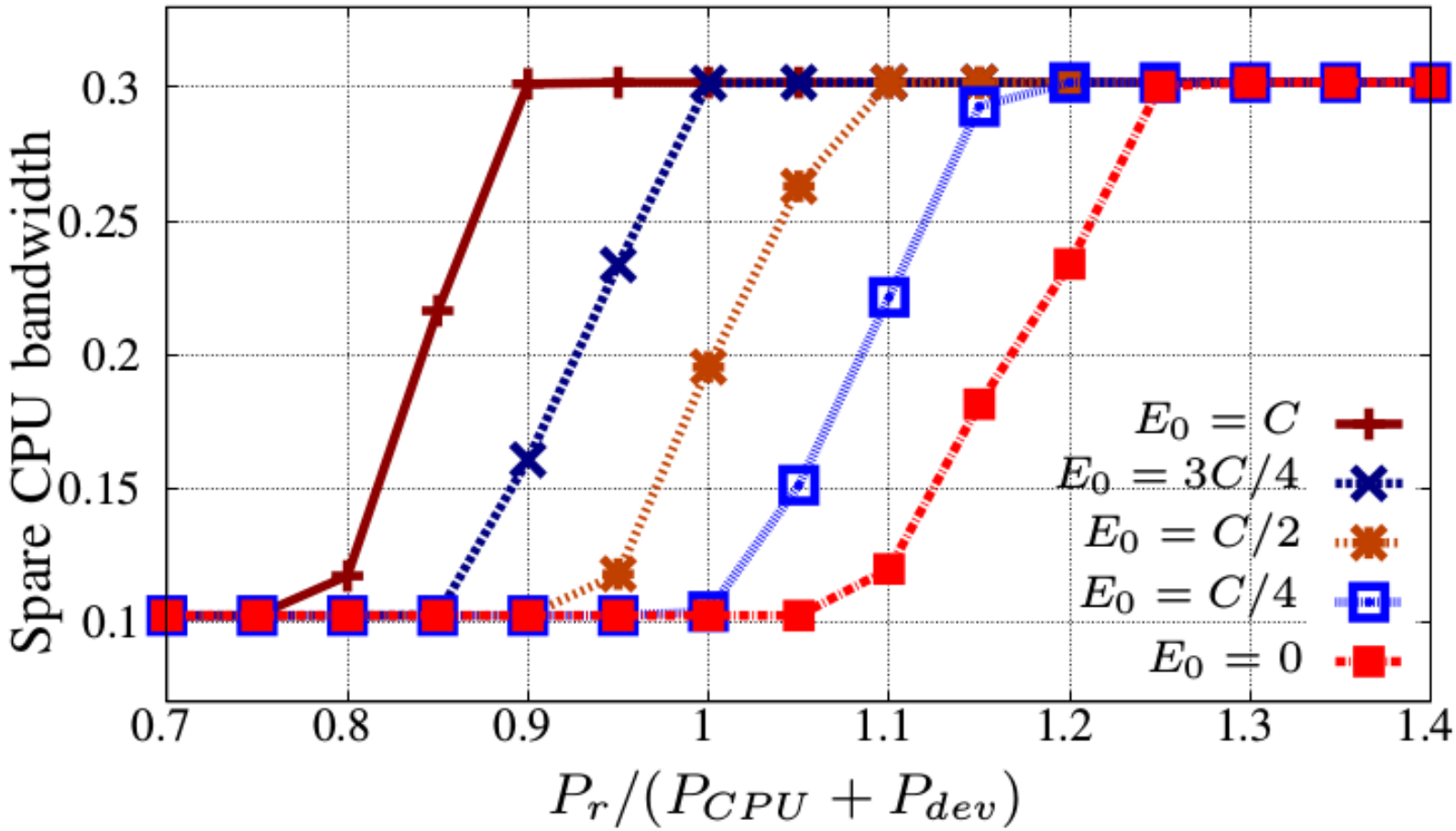
≈ 50% more preemptions



©2014 Scuola Superiore Sant'Anna

Results - NRT bandwidth

Realistic power model, $C = 250$mAh, $U = 0.5$ and $C_s/T_s = 0.4$



©2014 Scuola Superiore Sant'Anna

Conclusions

The proposed algorithm addressed the energy harvesting scheduling issue in real-time systems with fixed priorities

Its main contributions:

- better performance: lower P_r to guarantee schedulability
- high predictability and low online complexity
- performance does not drop when realistic sleep states are used

Drawback: a higher preemption number is introduced

©2014 Scuola Superiore Sant'Anna

Retis
Real-Time Systems Laboratory

thank you

m.bambagini@sssup.it

©2014 Scuola Superiore Sant'Anna