Power-Aware Temporal Isolation with VBS

Silviu S. Craciunas, Christoph M. Kirsch, Ana Sokolova
Department of Computer Sciences
University of Salzburg, Austria
firstname.lastname@cs.uni-salzburg.at

Contribution

We present methods that may reduce CPU power consumption with variable-bandwidth servers (VBS) while maintaining temporal isolation of concurrently running processes. We propose a frequency-scaling VBS algorithm that exploits CPU slack to minimize operating frequencies with maximal CPU utilization while maintaining temporal isolation. Furthermore, we show that, given knowledge of future events, further reductions in CPU power consumption may be possible by allowing the scheduler to redistribute computation time of process actions among the server periods during which the actions execute without affecting the actions' original response time bounds.

VBS

A VBS[1, 2] is configured by a single number u that determines a utilization bound (bandwidth cap). To configure their actual execution speed, each action of a process chooses a pair (λ, π) (virtual periodic resource) such that λ over π is less than or equal to the bandwidth cap u of the used VBS. Switching to different periods allows to trade off scheduling overhead and temporal isolation at runtime.

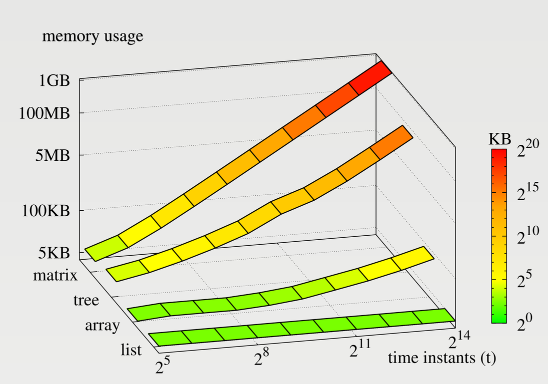
Let $\{P_i \mid i \in I\}$ be a set of processes each running on a VBS with utilization u_i. If

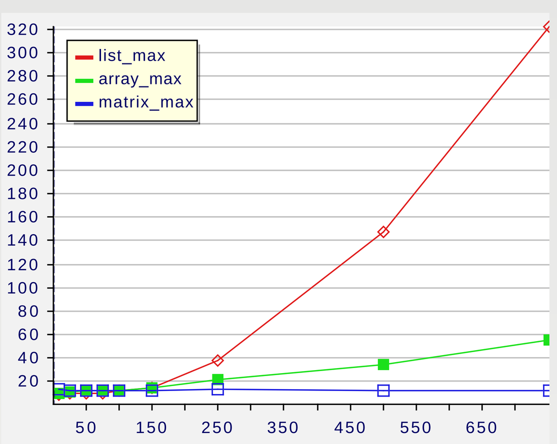
$$\sum_{i \in I} u_i \leq 1, \qquad (1)$$

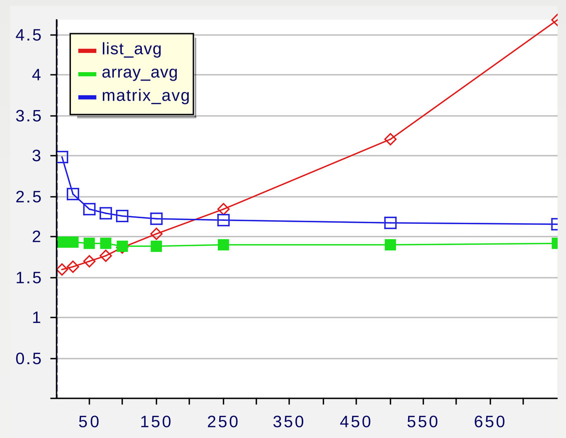
then this set of processes is schedulable using the EDF strategy so that each action meets its response bounds.

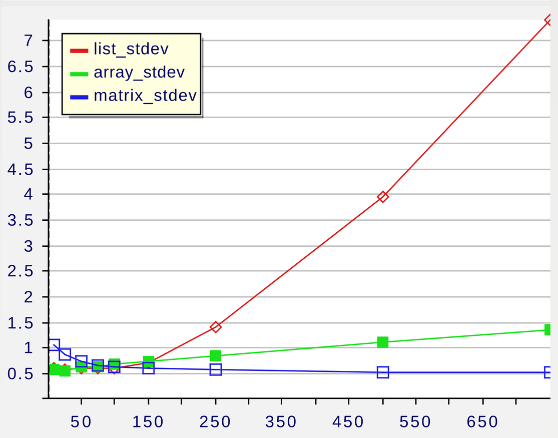
Three different implementations:

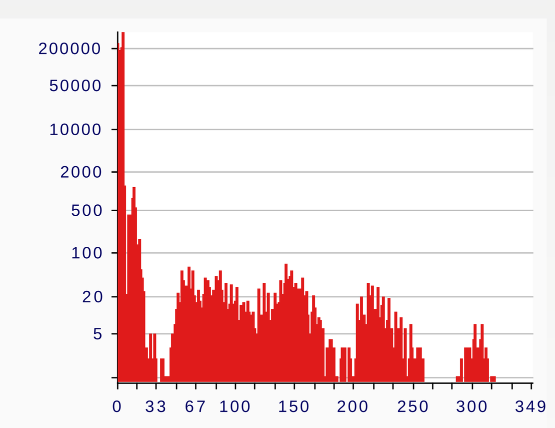
	list	array	matrix/tree
time	$O(n^2)$	$O(\log(t) + n \cdot \log(t))$	$\Theta(t)$
space	$\Theta(n)$	$\Theta(t+n)$	$O(t^2+n)$

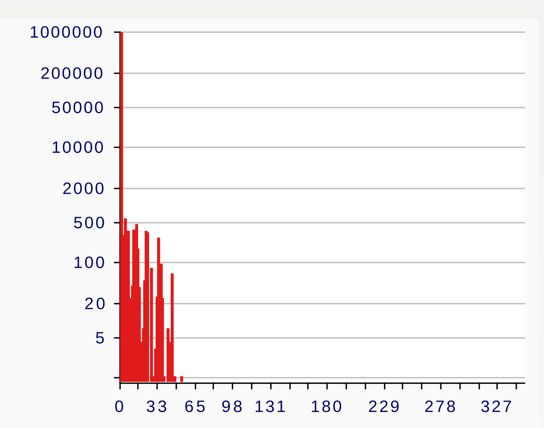


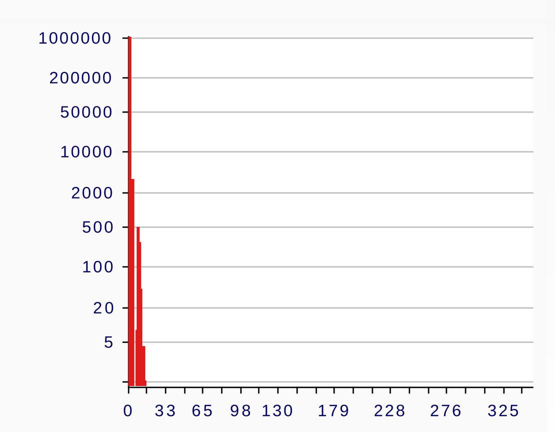


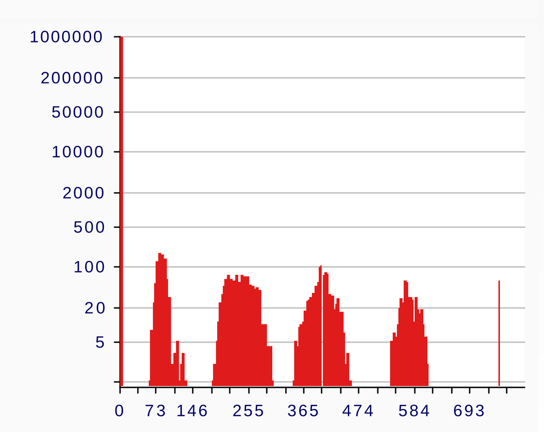












Frequency-scaling VBS

Modern processors often support dynamic scaling of CPU voltage and operating frequency, which opens up the general possibility to reduce CPU power consumption.

$$P \propto V^2$$

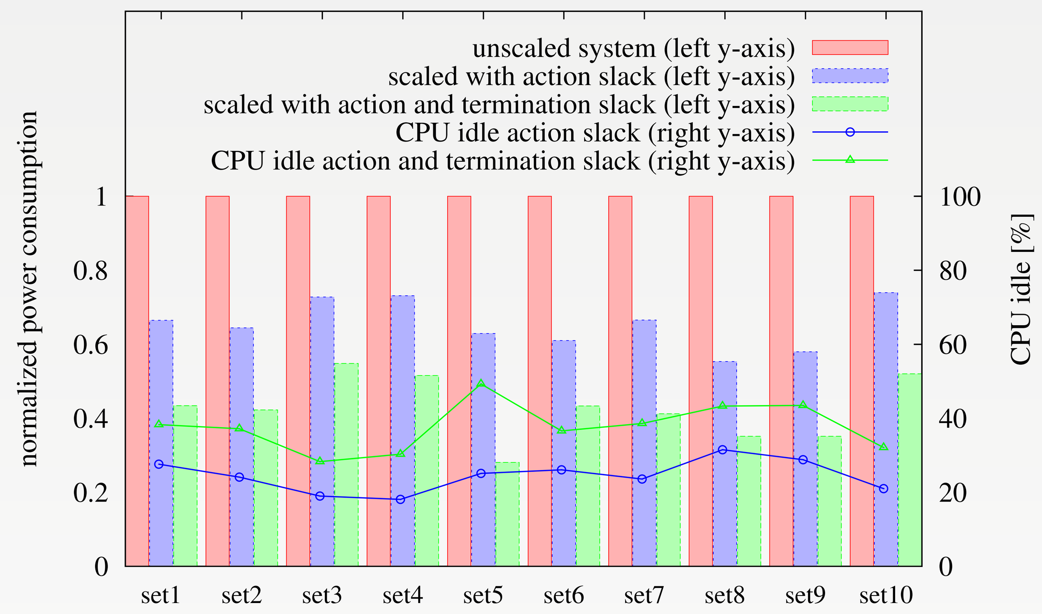
Lemma 1. *An EDF-schedulable set of tasks with release times, computation times, and deadlines, is still schedulable if the processor frequency in between any two release times is set to at least $U_c \cdot f_{max}$, with U_c being the current total utilization of all released tasks in the considered interval of time between two releases.*

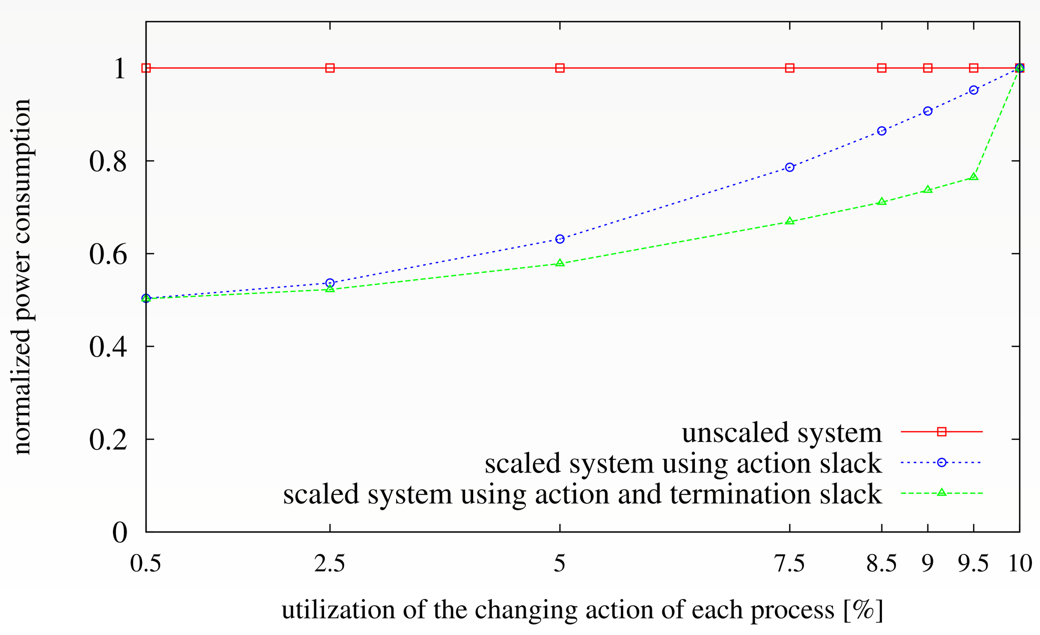
Possible whenever there is slack in the system:

- Static slack
  $f = f_{max} \cdot \sum_{i=1}^{n} u_i$
- Dynamic slack
  - Action slack
    $f = f_{max} \cdot \sum_{i=1}^{n} \frac{\lambda_{i,j}}{\pi_{i,j}}$
  - Termination slack
    $f = f_{max} \cdot \sum_{i=1}^{n} \frac{\lambda^*_{i,j}}{\pi_{i,j}}$,
    $\lambda^*_{i,j} = \left\lceil \frac{l_{i,j}}{n_{i,j}} \right\rceil$, $n_{i,j} = \left\lceil \frac{l_{i,j}}{\lambda_{i,j}} \right\rceil$

Action and termination slack can be used separately or together.

Algorithm FS-VBS(t) {
    $\mathcal{AA} =$ ARRIVAL[t]
    forall $(\alpha_{i,j} \in \mathcal{AA})$
        $n_{i,j} = \left\lceil \frac{l_{i,j}}{\lambda_{i,j}} \right\rceil$
        $\lambda_{i,j} = \left\lceil \frac{l_{i,j}}{n_{i,j}} \right\rceil$
    endfor
    $\mathcal{RA} =$ RELEASED[t]
    forall $(\alpha_{i,j} \in \mathcal{RA})$
        $U_c = U_c + \frac{\lambda_{i,j}}{\pi_{i,j}}$
    endfor
    return $f_{new} = U_c \cdot f_{max}$
}





Funding

Supported by the EU ArtistDesign Network of Excellence on Embedded Systems Design and the Austrian Science Funds P18913-N15 and V00125.

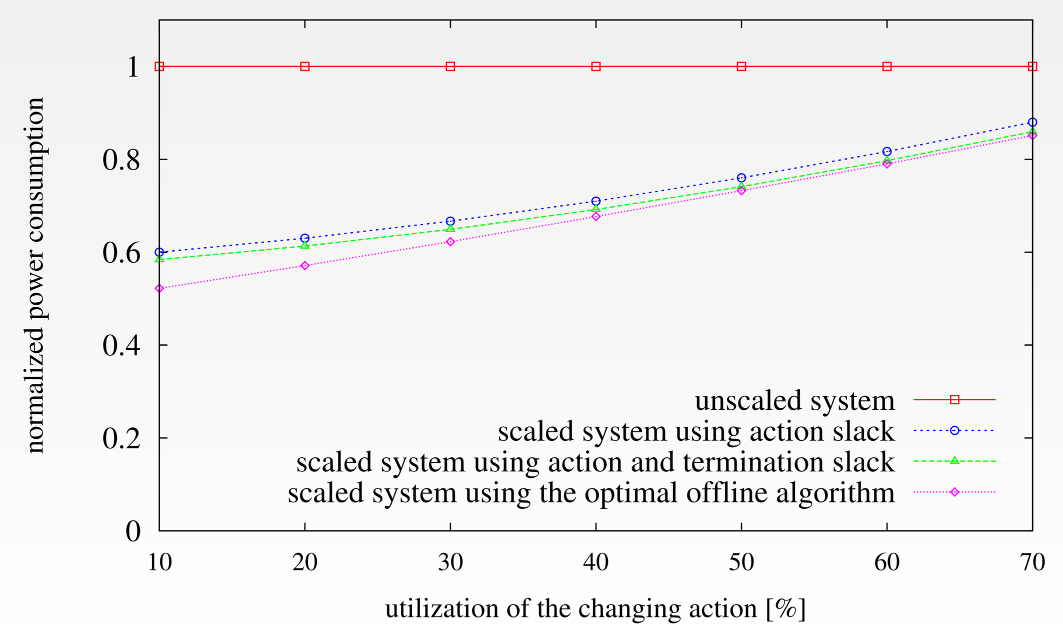
Optimal offline FS-VBS

Depending on the system model, more energy can be saved by allowing an action more freedom on how the load is executed within the periods, i.e., an action may assign a different limit for every period of its execution as long as the original response time bounds are met. In order to use this freedom, an action must have knowledge of the future.

$F(\lambda_{i,j,k} \mid i \in I, j \geq 0, 1 \leq k \leq n_{i,j}) =$
$gcd(\Pi) \sum_{n \in \mathbb{N}} F_n(\lambda_{i,j,k_n} \mid i \in I, j \geq 0)$.

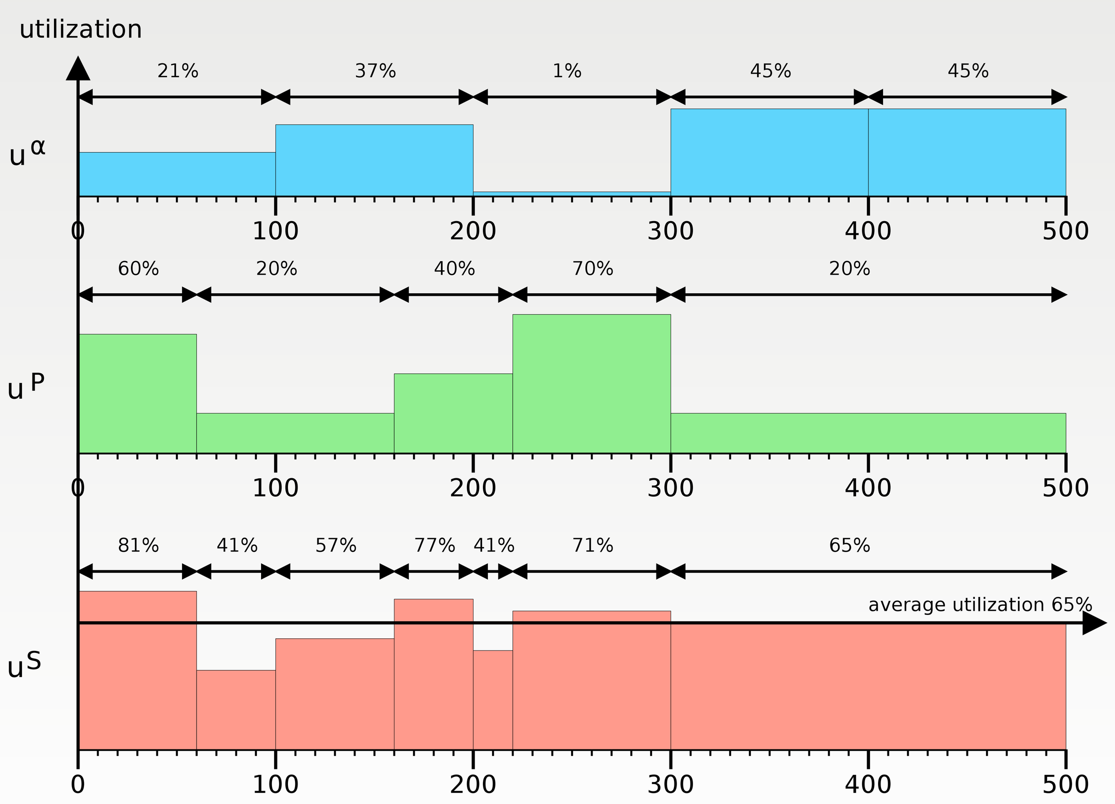
$F_n(\lambda_{i,j,k_n}) = E_{a,n}(\lambda_{i,j,k_n}) + E_{s,n}(\lambda_{i,j,k_n})$.

By plugging particular power-consumption functions $E_{a,n}$ and $E_{s,n}$, one gets the global power consumption function. We minimize $F(\lambda_{i,j,k})$ subject to constraints imposed by the semantics of VBS.



Online FS-VBS

Given a simplified power consumption model, the online algorithm decreases CPU utilization jitter to approximate the optimal algorithm by steering actual CPU utilization towards a computed average.



References

[1] S. S. Craciunas, C. M. Kirsch, H. Payer, H. Röck and A. Sokolova - Programmable Temporal Isolation through Variable-Bandwidth Servers In *Proc. SIES 2009, IEEE*

[2] S.S. Craciunas, C.M. Kirsch and A. Sokolova - Response Time versus Utilization in Scheduler Overhead Accounting In *Proc. RTAS 2010, IEEE*