

The Case for Altruistic Resource Management

Christian Poellabauer (cpoellab@cse.nd.edu, student: no) and Timothy Durnan (tdurnan@cse.nd.edu, student: yes)
Department of Computer Science and Engineering, University of Notre Dame

Problem: Selfish Resource Management. Consider the following scenario taken from the mobile and wireless computing domain. Energy has been receiving increasing attention, resulting in a number of different energy management techniques, including *Dynamic Voltage Scaling (DVS)* [1]. DVS is based on the concept of reducing the speed/voltage of a CPU when it is under-utilized, thereby reducing its power consumption while increasing the task execution times. In real-time systems, DVS algorithms have to compute energy-saving speed/voltage levels while ensuring that task deadlines are met. Consider Figure 1a, where the shaded area shows the power consumption of a CPU executing a task for a certain amount of time. The arrow denotes communication between two devices and the vertical line indicates an *end-to-end deadline* T_d , i.e., the processing and communication steps of both devices *A* and *B* in Figure 1a have to be concluded before T_d . A typical example for such a scenario is a video streaming application: *A* captures images, compresses them, and sends them to *B*, which decompresses and displays them. Figure 1b repeats the same experiment, however both *A* and *B* utilize their DVS capabilities, leading to reduced power needs of the CPU, but also to increased execution times. This may lead (as shown in the graph) to a missed deadline. To allow *B* to meet its deadline, it would have to execute faster, i.e., consume more energy.

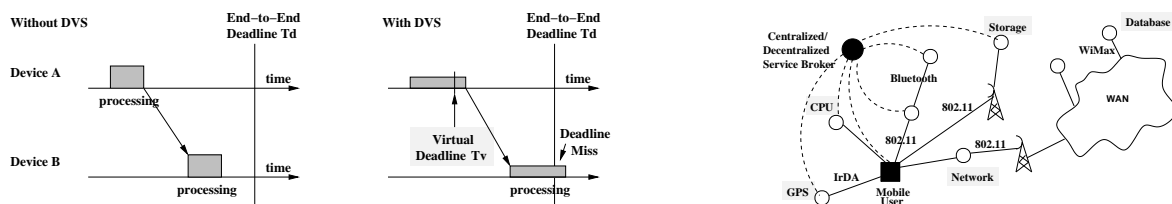


Figure 1. (a) Video streaming without DVS, (b) with DVS, and (c) the SPIRIT sharing infrastructure.

Now assume that *B* is essential to the operation of the distributed system, but at the same time also the more energy-constrained (i.e., the remaining battery charge is lower than *A*'s). In this case, it is desirable that *A* reduces its use of DVS, such that *B* can continue to fully exploit its DVS capability to prolong its battery life. To achieve that, it is necessary for *A* and *B* to negotiate limits to the use of DVS, e.g., by introducing a deadline on *A*, called *virtual deadline* T_v (see Figure 1b). This deadline forces *A* to run faster (limiting the extent to which *A* can exploit DVS), but allowing *B* to fully utilize DVS.

The Case for Altruistic Distributed Resource Management. To summarize, we argue that resource management in general (and energy management in particular) must be *altruistic* in that devices have to negotiate the limits of the use of their resource management techniques, considering the overall goal of the system and the system state (e.g., the current battery charge levels). Devices therefore become selfless in that they have to voluntarily limit their exploitation of resource management techniques in order to support a common goal in a distributed system. For example, in a video conference, it is often preferable to be able to continue for another 30 minutes with *all* participants than to continue for another 60 minutes, but only half the audience (assuming the other half has depleted batteries).

Example 1: Distributed Dynamic Voltage Scaling. The first example is called *DDVS – Distributed Dynamic Voltage Scaling* – an end-to-end or distributed DVS algorithm that implements such a negotiation in form of a feedback control loop. Here, the controller (e.g., PI or PID) computes a *virtual deadline* T_v , which is imposed on task execution at device *A* depending on input variables (e.g., task parameters), battery charge levels of *A* and *B*, and the *slack* d_s , which is defined as $d_s = T_d - T_f$, where T_f is the *finish time* of the image processing task at device *B*. Based on the monitored slack and charge levels, T_v is dynamically adjusted to ensure the desired balance between maximum energy savings for both devices and meeting the end-to-end deadlines. The feedback controller increases T_v when the slack is positive (indicating that there is room for more savings) and decreases T_v when the slack is negative (indicating a missed deadline).

Example 2: Personal Wireless Grids. Grid computing addresses the problem of finding and accessing resources for scientific purposes. In wireless and mobile settings, the same principle can be applied to allow nomadic users spontaneous access to other mobile users' resources (CPU for processing, disks for temporary storage, etc.) or information (e.g., sensor information such as GPS). The SPIRIT resource/information sharing infrastructure (Figure 1c) not only provides the basis for above described selfless energy management, but implements a general sharing tool, allowing mobile users to build 'personal wireless Grids'. It is based on a publish/subscribe system, whose decoupled nature provides the optimal basis for communication in dynamic and unreliable wireless environments. A *broker* (either a central/dedicated device or implemented in a distributed fashion) manages multiple communication channels, each channel providing access to a specific resource or information.

[1] P. Pillai and K. Shin. Real-Time Dynamic Voltage Scaling for Low-Power Embedded Operating Systems. In *Proc. of the 18th ACM Symposium on Operating Systems Principles*, 2001.

[2] C. Poellabauer, L. Singleton, and K. Schwan. Feedback-Based Dynamic Frequency Scaling for Memory-Bound Real-Time Applications. In *Proc. of the 11th Real-Time and Embedded Technology and Applications Symposium*, March 2005.