Combined Dynamic Voltage Scaling and Adaptive Body Biasing for Heterogeneous Distributed Real-time Embedded Systems

Le Yan, Jiong Luo and Niraj K. Jha
Department of Electrical Engineering
Princeton University, Princeton, NJ 08544
{lyan, jiongluo, jha}@ee.princeton.edu

Abstract— **Dynamic voltage scaling (DVS) is a powerful technique for reducing dynamic power consumption in a computing system. However, as technology feature size continues to scale, leakage power is increasing and will limit power savings obtained by DVS alone. Previous system-level real-time scheduling approaches use DVS alone to optimize power consumption without considering leakage power. To overcome this limitation, we propose a new scheduling algorithm that combines DVS and adaptive body biasing (ABB) to simultaneously optimize both dynamic power consumption and leakage power consumption for real-time distributed embedded systems. First, we derive an analytical expression to determine the optimal supply voltage and body bias voltage under a given clock frequency. Based on this expression, we compute the optimal energy consumption at a given clock frequency and analyze the tradeoff between energy consumption and execution time for a set of tasks with precedence relationships and real-time constraints. We then propose a scheduling algorithm to reduce total power consumption under given real-time constraints. This algorithm also considers variations in power consumption of different tasks and characteristics of different voltage-scalable processing elements (PEs) to maximize power reduction. Experimental results show that the average power reduction of our technique with respect to DVS alone is 34.7%, while the average saving compared to no voltage scaling is 68.3% for the 0.07μm technology.**

I. Introduction

Power consumption has become the limiting factor for both battery-operated electronics and high-performance computer systems. While dynamic power consumption has traditionally been the primary source of power consumption, leakage power consumption is becoming an increasingly important concern as the technology feature size shrinks. Supply voltage scaling, which is supported by various embedded processors such as Intel XScale processor [1], Transmeta Crusoe processor with LongRun power management technology [2], and AMD's mobile processor with AMD PowerNow! technology [3], is effective in reducing dynamic power consumption quadratically. However, supply voltage scaling often requires a reduction in the threshold voltage that increases the subthreshold leakage current exponentially, and hence the leakage power consumption. Leakage power is expected to become comparable to dynamic power in the forthcoming generations of technology [4]. Hence, it is increasingly important for run-time power optimization techniques to trade off supply voltage and threshold voltage to optimize dynamic power consumption and leakage power consumption jointly. Dynamic voltage scaling (DVS) [5–11], an effective run-time technique to manage the dynamic power at the system level, is no longer sufficient to manage the total power consumption for the new technology generations.

Many techniques have been proposed to reduce leakage power consumption. The work in [12, 13] summarizes the most commonly used leakage reduction techniques. The leakage power dissipated by a circuit depends on the input vector to the circuit [14, 15]. Hence, input vector control has been proposed to find the input pattern that maximizes the number of disabled transistors in all transistor stacks across the design [16]. The work in [17] develops an efficient algorithm that determines a low-leakage input vector using a sampling of random vectors. Supply voltage gating [4, 18] is another approach to obtain leakage power savings, in which the power supply is disconnected from the circuit so that idle units do not consume leakage power. Decreasing leakage current via increasing threshold voltage [19–22] exploits the exponential dependence of subthreshold leakage power on threshold voltage to reduce leakage power. The work in [19] demonstrates the effectiveness of reverse body bias to minimize leakage power consumption in scaled dual-V_t CMOS circuits. In [20], a variable threshold voltage scheme, based on controlling body bias, has been successfully applied to a commercial digital signal processor. Adaptive body biasing (ABB) is a technique that adjusts the body bias at run-time to reduce leakage power consumption. It has been proposed to dynamically control the threshold voltage over a continuous range [22], as well as in a discrete fashion through threshold voltage hopping [21]. The work in [22] presents a dynamic threshold voltage scaling hardware. The hardware, which has a feedback loop consisting of a voltage controlled oscillator, charge pumps a feedback controller to dynamically control the body bias voltage. In [21], a back-gate bias controller is implemented in a small scale RISC processor to realize threshold voltage hopping.

Optimizing dynamic power and leakage power simultaneously has been demonstrated to be important for energy reduction [23–26]. The work in [23] automatically adjusts the supply and threshold voltages to minimize power consumption at the circuit level. In [26], a combined DVS and ABB method is proposed for a single voltage-scalable processor to reduce both dynamic power and leakage power. This work derives an expression to obtain an optimal tradeoff between supply voltage and body bias voltage for a given clock frequency. However, it only considers this tradeoff for a single task, and is not applicable to a scenario containing interactions among a set of tasks with real-time constraints. The circuit delay model it uses is different from the standard alpha-power model [27].

In this paper, we propose a new scheduling algorithm that addresses both dynamic power and leakage power effectively. The algorithm can be applied to heterogeneous distributed embedded systems for real-time applications in the form of a set of periodic task graphs with precedence relationships, hard deadlines

Acknowledgments: This work was supported by DARPA under contract no. DAAB07-02-C-P302.

ICCAD'03, November 11-13, 2003, San Jose, California, USA.

and varying task power consumptions. The real-time scheduling algorithm needs to efficiently trade off energy consumption and task execution time, both of them dependent on the supply voltage and threshold voltage, in order to maximize energy savings without violating real-time constraints. To tackle this issue, we propose a novel two-phase approach. First, we derive an analytical energy consumption model using the standard alpha-power model and determine the optimal supply voltage and body bias voltage for a given clock frequency. Next, we compute the energy consumption under the optimal points for supply voltage and threshold voltage, and evaluate the curves of optimal energy consumption vs. clock period. The scheduling algorithm thus performs DVS and ABB simultaneously to trade off energy consumption and task execution time, by utilizing the convex characteristic of these curves.

The rest of this paper is organized as follows. In Section II, we present some preliminary concepts. Section III uses a motivational example to illustrate the importance of considering leakage power. In Section IV, we derive the optimal tradeoff point between supply voltage and body bias voltage for a given clock frequency, as well as the curves for optimal energy consumption vs. clock period. Section V presents a new scheduling algorithm combining both DVS and ABB methods for distributed real-time embedded systems. Section VI gives the experimental results, which demonstrate the effectiveness of our proposed algorithm in optimizing both dynamic power and leakage power. Section VII draws the conclusions.

II. Preliminaries

We use the following expressions to evaluate the two main sources of power consumption, dynamic power and leakage power [28], where dynamic power can be represented as

$$P_{dynamic} = \frac{1}{2} NCfV_{dd}^2 \tag{1}$$

and subthreshold leakage power can be represented as

$$P_{leakage} = V_{dd} I_{sub} = I_s (\frac{W}{L}) V_{dd} e^{\frac{-V_{th}}{nV_T}} \tag{2}$$

In the above equations, N is the switching activity, C is the capacitance, I_{sub} is the subthreshold leakage current, I_s and n are technology parameters, W and L are device geometries, V_T is the thermal voltage, and V_{dd} and V_{th} are the supply voltage and threshold voltage, respectively. The operational frequency f can be expressed as

$$f = \frac{(V_{dd} - V_{th})^{\alpha}}{kV_{dd}} \tag{3}$$

where k is a constant for a given technology process and $1 < \alpha \leq 2$.

Another source of leakage power is the source-body and drain-body junction leakage, which is becoming increasingly important. We also consider this component of leakage power later in Section IV.

There is a third source of power consumption, short-circuit power, which results from the short-circuit current path between the power supply and ground during switching. Short-circuit power is projected to be constant around 10% for succeeding technologies [29]. We thus ignore it throughout this paper.

Embedded systems are frequently specified in the form of a set of task graphs. A simple specification consisting of a single task graph is shown in Fig. 1. It consists of three tasks $T_1 - T_3$ (such tasks are of coarse granularity, e.g., discrete cosine transform may be a task). Each task is annotated by its worst-case execution time (WCET) on each type of processor it can run on. To simplify matters, for this example, only one type of processor is assumed. Edges between tasks denote communication, and are also labeled by their WCET on a given communication link. There is no WCET assigned to the $T_1 - T_2$ edge as both T_1 and T_2 are assigned to the same PE and intra-PE communication time is assumed to be zero based on the traditional assumption in distributed computing (this is because inter-PE communication takes much more time than intra-PE communication in a distributed system). All the source (sink) nodes have an arrival time (deadline) by when computation can begin (must finish). In general, arrival times (deadlines) may also exist for some intermediate nodes. The time interval at which the task graph is repeatedly invoked is called its period. A period may be less than, equal to or greater than the deadlines. Different task graphs in a specification may have different periods, giving rise to a multi-rate specification. The least common multiple (LCM) of all periods is called the hyperperiod. It is known that scheduling the task graphs in their hyperperiod leads to a valid schedule [30].

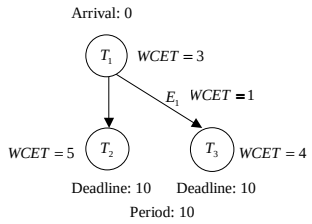


Fig. 1. Task graph

III. Motivational example

This section presents a motivational example to illustrate the significance of considering leakage power as technology scales down.

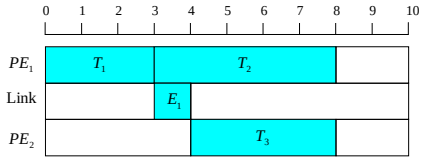
Example 1: Consider the task graph shown in Fig. 1 once again. Suppose the power consumption of tasks T_1, T_2, and T_3 are c_1W, c_2W, and c_3W, respectively, where c_1, c_2, and c_3 are some constants. We consider three base technologies, $0.07\mu m$, $0.05\mu m$, and $0.035\mu m$, as shown in Table I, which provide three different power consumption scenarios [24].

TABLE I

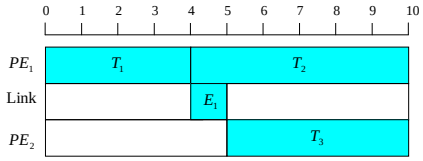
TECHNOLOGIES PROVIDING DIFFERENT POWER CONSUMPTION SCENARIOS

Technology	$0.07\mu m$	$0.05\mu m$	$0.035\mu m$
Dynamic power	78%	56%	33%
Leakage power	22%	44%	67%

Fig. 2(a) gives a feasible schedule for the task graph on a distributed system consisting of two PEs connected by a link. The following calculations take into account the different supply and threshold voltages of PE_1 ($V_{dd} = 2.0V$, $V_{th} = 0.6V$) and PE_2 ($V_{dd} = 1.5V$, $V_{th} = 0.4V$). Fig. 2(b) shows the new schedule if we extend the execution times of task T_1, T_2 and T_3 to $4s$, $6s$, and $5s$, respectively. Since the execution time of T_1 is extended from $3s$ to $4s$, the speed of processor PE_1 can be scaled down by a ratio of $4/3$. Given the scaled clock frequency, we consider two voltage scaling approaches to reduce power consumption. One is supply voltage scaling alone, and the other is combined supply



(a) At nominal supply and threshold voltages



(b) At scaled supply and threshold voltages

Fig. 2. Different schedules for a distributed system

and threshold voltage scaling.

Let us first consider the $0.07\mu m$ technology, in which dynamic power consists of 78% of total power consumption, while leakage power consists of 22%. One approach to reduce power consumption is supply voltage scaling. Based on Equation (3), given the frequency scaling ratio of $4/3$ for T_1, the supply voltage V_{dd} of PE_1 can be scaled down from $2.0V$ to $1.55V$ correspondingly (assuming $\alpha = 1.4$). The dynamic power of T_1 is thus reduced from $0.78c_1W$ to $0.35c_1W$, and the leakage power is reduced from $0.22c_1W$ to $0.17c_1W$, based on Equations (1) and (2), respectively. The power consumption of T_1 including both dynamic power and leakage power is thus reduced to $0.52c_1W$. Similarly, the power consumption of T_2 and T_3 is reduced to $0.65c_2W$ and $0.58c_3W$, respectively. This produces an average power reduction of 41.7% over no voltage scaling, assuming $c_1 = c_2 = c_3$.

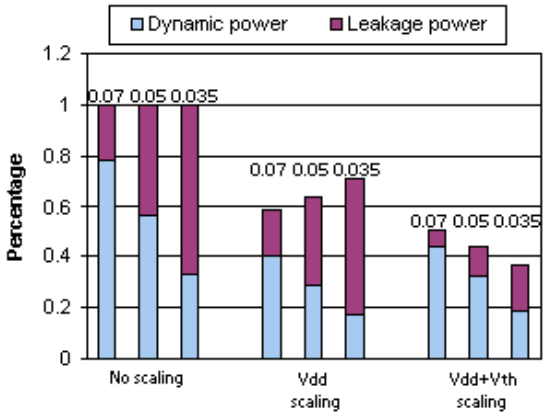


Fig. 3. Power reduction based on the three different technologies

Another approach for reducing power consumption is to combine supply and threshold voltage scaling. For T_1, the frequency of PE_1 can be scaled down by a ratio of $4/3$. If the threshold voltage V_{th} of PE_1 is increased from $0.6V$ to $0.64V$, then the supply voltage V_{dd} can be reduced from $2.0V$ to $1.62V$ based on Equation (3). The dynamic power of T_1 is thus reduced to $0.38c_1W$, and the leakage power is reduced to $0.06c_1W$. Correspondingly, the power consumption of T_1 is reduced to $0.44c_1W$. Similarly, for T_2 and T_3, the power consumption is reduced to $0.56c_2W$ and $0.51c_3W$, respectively. The average power reduction of supply and threshold voltage scaling compared to supply voltage scaling alone is 13.7%, while the reduction with respect to no voltage scaling is 49.7%, again assuming $c_1 = c_2 = c_3$.

Fig. 3 gives power consumption for the three different approaches under the three technologies. It can be observed that both dynamic power and leakage power are lowered by either using supply voltage scaling or combined supply and threshold voltage scaling. However, as technology scales from $0.07\mu m$ to $0.035\mu m$, and correspondingly leakage power increases from 22% to 67%, supply voltage scaling becomes less effective in reducing power consumption. On the other hand, combined supply and threshold voltage scaling provides more power savings as leakage power increases. This is reasonable given the fact that subthreshold leakage power is exponentially dependent on threshold voltage, while it is only linearly dependent on supply voltage. Table II compares power reduction of supply voltage scaling and combined supply and threshold voltage scaling under the three base technologies. Combined supply and threshold voltage scaling provides 64.0% power reduction over no voltage scaling in the $0.035\mu m$ process, while supply voltage scaling only provides 28.3% power reduction. Combined supply and threshold voltage achieves 49.3% power reduction over supply voltage scaling in the $0.035\mu m$ process, in contrast with only 13.7% in the $0.07\mu m$ process. This example demonstrates that combined supply and threshold voltage scaling becomes increasingly more powerful than supply voltage scaling alone, as technology scales and the leakage power becomes comparable to or larger than dynamic power. □

TABLE II

POWER REDUCTION COMPARISON BETWEEN $V_{dd}+V_{th}$ SCALING AND V_{dd} SCALING

Technology (μm)	0.07	0.05	0.035
V_{dd} scaling vs. no scaling	41.7%	35.7%	28.3%
$V_{dd}+V_{th}$ scaling vs. no scaling	49.7%	56.0%	64.0%
$V_{dd}+V_{th}$ scaling vs. V_{dd} scaling	13.7%	31.7%	49.3%

IV. ENERGY CONSUMPTION MODEL FOR COMBINED DYNAMIC VOLTAGE SCALING AND ADAPTIVE BODY BIASING

This section derives the energy consumption model considering both dynamic power and leakage power for a set of tasks implemented in a distributed embedded system, under the assumption that supply voltage and body bias voltage can be scaled simultaneously in a dynamic fashion.

A. Threshold voltage model

The threshold voltage model of MOSFET transistors is given by:

$$V_{th} = V_{th0} + \gamma(\sqrt{\Phi_s - V_{bs}} - \sqrt{\Phi_s}) + \Delta V_{th}(SCE) + \Delta V_{th}(DIBL) + \Delta V_{th}(NW) \quad (4)$$

where V_{th0} is the threshold voltage at zero substrate bias, γ is the body bias coefficient, Φ_s is the surface potential, V_{bs} is the body bias voltage, and $\Delta V_{th}(SCE)$, $\Delta V_{th}(DIBL)$ and $\Delta V_{th}(NW)$ are the modifications caused by substrate doping concentration, short channel, and narrow channel, respectively. In [26], it is

shown that V_{th} has a linear dependence on supply voltage V_{dd} and body bias voltage V_{bs} based on the SPICE simulation of the Berkeley predictive models for a $0.07\mu m$ process:

$$V_{th} = V_{th1} - k_1 V_{dd} - k_2 V_{bs} \tag{5}$$

where V_{th1}, k_1, and k_2 are constants.

B. Delay model

The operational frequency, f, is given by Equation (3). Substituting (5) into (3) yields the expression of f in terms of V_{dd} and V_{bs},

$$f = \frac{((1+k_1)V_{dd} + k_2 V_{bs} - V_{th1})^{\alpha}}{kV_{dd}} \tag{6}$$

C. Power consumption model

The total power consumption includes both dynamic power and leakage power. The dynamic power, $P_{dynamic}$, is given by Equation (1). The leakage power, $P_{leakage}$, is due to subthreshold leakage current, I_{sub}, as shown in Equation (2), as well as the contributions of drain-body junction leakage current, I_j, and source-body junction leakage current, I_b [19]. Thus, a more exact equation for leakage power is given by:

$$P_{leakage} = I_s(\frac{W}{L})V_{dd}e^{\frac{-V_{th}}{nV_T}} + |V_{bs}|(I_j + I_b) \tag{7}$$

Substituting (5) into (7), we get:

$$P_{leakage} = k_3 V_{dd} e^{k_4 V_{dd} + k_5 V_{bs}} + |V_{bs}|(I_j + I_b) \tag{8}$$

where k_3, k_4, and k_5 are new constants. Therefore, the power consumption of task i (with switching activity N_i) running on a PE can be modeled as:

$$P_i = \frac{1}{2}N_i C f V_{dd}^2 + k_3 V_{dd} e^{k_4 V_{dd} + k_5 V_{bs}} + |V_{bs}|(I_j + I_b) \tag{9}$$

D. Tradeoff between supply voltage and body bias voltage

For each task i with switching activity N_i, its energy consumption, E_i, can be expressed in terms of clock frequency f_i, supply voltage V_{di} and body bias voltage V_{bi} as:

$$E_i = \frac{1}{2}t_i N_i C f_i V_{di}^2 + t_i k_3 V_{di} e^{k_4 V_{di} + k_5 V_{bi}} + t_i |V_{bi}|(I_j + I_b) \tag{10}$$

where t_i is the task's corresponding execution time under clock frequency f_i, i.e.,

$$t_i = \frac{\eta_i}{f_i} \tag{11}$$

where η_i is the number of clock cycles for executing task i.

There are three variables in Equation (10), f_i, V_{di}, and V_{bi}. To reduce the number of variables, from Equation (6), V_{bi} be can be represented as a function of f_i and V_{di} as:

$$V_{bi} = k_6 V_{di} + k_7 (f_i V_{di})^{\frac{1}{\alpha}} + k_8 \tag{12}$$

where k_6, k_7 and k_8 are new constants. Substituting (12) into (10) gives:

$$E_i = \frac{1}{2}t_i N_i C f_i V_{di}^2 + t_i k_9 V_{di} e^{k_{10} V_{di} + k_{11}(f_i V_{di})^{\frac{1}{\alpha}}} + t_i k_{12} V_{di} + t_i k_{13}(f_i V_{di})^{\frac{1}{\alpha}} + t_i k_{14} \tag{13}$$

where $k_9 - k_{14}$ are new constants. E_i can thus be determined by f_i and V_{di}. Given the clock frequency f_i, to get the optimal supply voltage V_{di}^{opt}, we require

$$\frac{\partial E_i}{\partial V_{di}}|_{V_{di}=V_{di}^{opt}} = 0 \tag{14}$$

Substituting (13) into (14) yields the relationship between V_{di}^{opt} and f_i:

$$k_9 e^{k_{10}V_{di}^{opt} + k_{11}(f_i V_{di}^{opt})^{\frac{1}{\alpha}}}[1 + k_{10}V_{di}^{opt} + \frac{1}{\alpha}k_{11}(f_i V_{di}^{opt})^{\frac{1}{\alpha}}] + N_i C f_i V_{di}^{opt} + k_{12} + \frac{1}{\alpha}k_{13} f_i^{\frac{1}{\alpha}}(V_{di}^{opt})^{\frac{1}{\alpha}-1} = 0 \tag{15}$$

V_{di}^{opt} corresponds to the optimal supply voltage leading to minimum energy consumption if the second derivative of E_i with respect to V_{di} at V_{di}^{opt} satisfies

$$\frac{\partial^2 E_i}{\partial V_{di}^2}|_{V_{di}=V_{di}^{opt}} > 0 \tag{16}$$

Thus, given the clock frequency f_i, the optimal supply voltage V_{di}^{opt} can be calculated based on Equation (15), and then the optimal body bias voltage can be determined by Equation (12). Fig. 4 shows optimal tradeoff between supply voltage and body bias voltage with respect to clock frequency for the $0.07\mu m$ technology. The parameters for the energy model are adapted from values provided in [26], where $V_{dd} \geq 0.5V$ and $-1V \leq V_{bs} \leq 0V$.

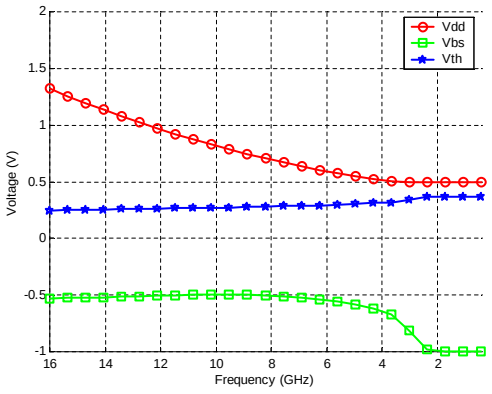


Fig. 4. Optimal tradeoff between supply voltage and body bias voltage at different clock frequencies for the $0.07\mu m$ technology

E. Tradeoff between energy consumption and clock period

Substituting the optimal points of supply voltage and body bias voltage for a given clock frequency into Equation (10), we can derive the optimal energy consumption $E_i^{opt}(f_i, t_i)$ at a given clock frequency, which is a function of clock frequency f_i and task execution time t_i. The curves of optimal per cycle energy consumption vs. clock period for three tasks with different values of N_i are shown in Fig. 5.

To analyze the tradeoff between energy consumption and clock period, we compute the negative of the first derivative of optimal energy consumption for task i with respect to its execution time, $-\frac{\partial E_i^{opt}(f_i,t_i)}{\partial t_i}$, based on Equations (13) and (15). Let us denote this as $energy\_derivative(f_i)$, which is a function of f_i [11]. It indicates the energy reduction rate if the execution time of task i at

clock frequency f_i is extended by dt time unit. When the execution time of task i is extended by dt, the clock frequency f_i can be scaled down based on Equation (11). Given the scaled clock frequency, the supply voltage and body bias voltage can be calculated based on Equations (15) and (12), respectively. The total energy reduction achieved by extensions of a set of tasks is given by:

$$\Delta E = \int_0^{total\_slack} energy\_derivative(f)dt \qquad (17)$$

where *total_slack* is the total slack in the system. Allocating dt time unit from slack $[t, t+dt]$ to some task can result in the energy reduction equal to $energy\_derivative(f)dt$, when the task with switching activity N is executed under clock frequency f. Maximizing this integral can provide maximum energy savings ΔE.

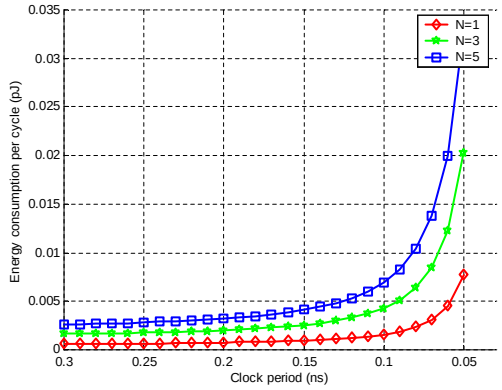


Fig. 5. Energy consumption per cycle vs. clock period for tasks with different switching activities

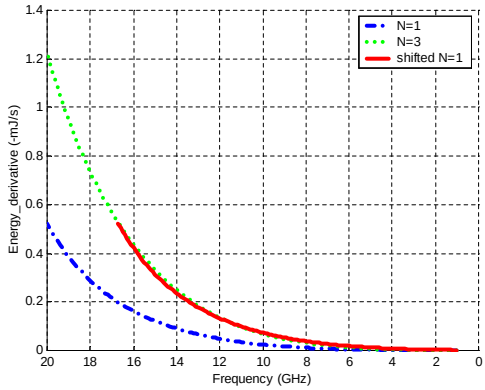


Fig. 6. *Energy_derivative* with respect to clock frequency at different switching activities

As seen from Fig. 5, the curve of optimal per cycle energy consumption vs. clock period at a given switching activity is convex. Hence, $energy\_derivative(f_i)$ is a monotonically decreasing function with respect to clock period, and correspondingly a monotonically increasing function with respect to clock frequency for a given switching activity, as shown in Fig. 6. Under the same clock frequency, *energy_derivative* for a higher switching activity ($N = 3$) is higher, which means the energy reduction rate is relatively higher. It is therefore important to allocate slack to the task with a higher *energy_derivative* (task with $N = 3$) first. After its *energy_derivative* drops to the point equal to the maximum *energy_derivative* of the task with a lower switching activity ($N = 1$), the energy reduction rates of both tasks should be kept at almost the same level, as seen by comparing curves $N = 3$ and $shifted\ N = 1$. Then the slack can be allocated to them in a balanced way.

For a single processor, optimal slack allocation can be achieved as follows. A task is defined as non-extensible if extending its execution time will lead to a violation of real-time constraints. For a set of extensible tasks *ext_set*, given the frequency scaling step df, the slack allocation should guarantee that for any task i in *ext_set*,

$$energy\_derivative_i(f_i) = max_{j \in ext\_set} energy\_derivative_j(f_j) \qquad (18)$$

where $f_{min} \le f_i \le f_{max}$. f_{max} and f_{min} are maximum and minimum frequency levels of the voltage-scalable PE, respectively. When a task is not extensible anymore, or it has reached the minimum frequency level of its assigned PE, it is deleted from *ext_set*. The remaining slack can be allocated to the tasks in *ext_set* until all the available slack has been allocated or *ext_set* is empty.

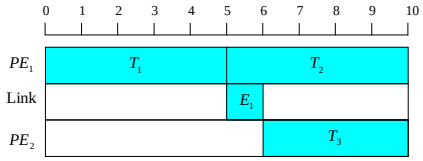


Fig. 7. A schedule achieving different *total_slack*

For distributed systems, *total_slack* is not fixed when the initial schedule is given. For example, for the initial schedule shown in Fig. 2(a), Fig. 2(b) gives a schedule if the execution time of tasks $T_1 - T_3$ are extended by 1s simultaneously. The achieved *total_slack* is 3s. On the other hand, if the execution time of T_1 is fully extended as shown in Fig. 7, the achieved *total_slack* is 2s. Hence, slack allocation based on Equation (18) is no longer optimal. However, it is still a good heuristic as shown later through experimental results for distributed systems.

V. The Scheduling Algorithm

This section proposes a new scheduling algorithm addressing both dynamic power and leakage power effectively for heterogeneous distributed embedded real-time systems. This algorithm is an extension of the power-profile driven DVS scheduling algorithm [11]. In [11], the scheduling algorithm optimizes power consumption without considering leakage power for a set of tasks under real-time constraints. In our algorithm, there are two major concerns: maximizing energy savings and meeting real-time constraints. To achieve maximum energy reduction, it identifies the optimal tradeoff point between supply voltage and body bias voltage based on Equations (12) and (15) when multiple tasks update their operating clock frequencies. It also considers the tradeoff between energy consumption and clock period using the heuristic based on Equation (18). It thus reduces dynamic power and leakage power simultaneously. To guarantee real-time constraints, it evaluates the validity of the generated schedule by checking the *earliest_start_time* (*EST*) and *latest_finish_time* (*LFT*) of scheduled events. The *EST* of a scheduled event is the earliest time at which it can start its execution without violating its arrival time and precedence relationships. The *LFT* is the latest time at which the event must finish its execution without violating any deadlines and precedence relationships. For each scheduled event i (i is either a task or an inter-PE communication edge), its EST_i and LFT_i

can be expressed as:

$$EST_i = max(a_i, max_{j \in p}(EST_j + WCET_j)) \quad (19)$$

$$LFT_i = min(d_i, min_{j \in c}(LFT_j - WCET_j)) \quad (20)$$

where a_i (d_i) is the arrival time (deadline) of event i, p (c) are direct parents (children) of event i, and $WCET_j$ is the worst execution time of event j.

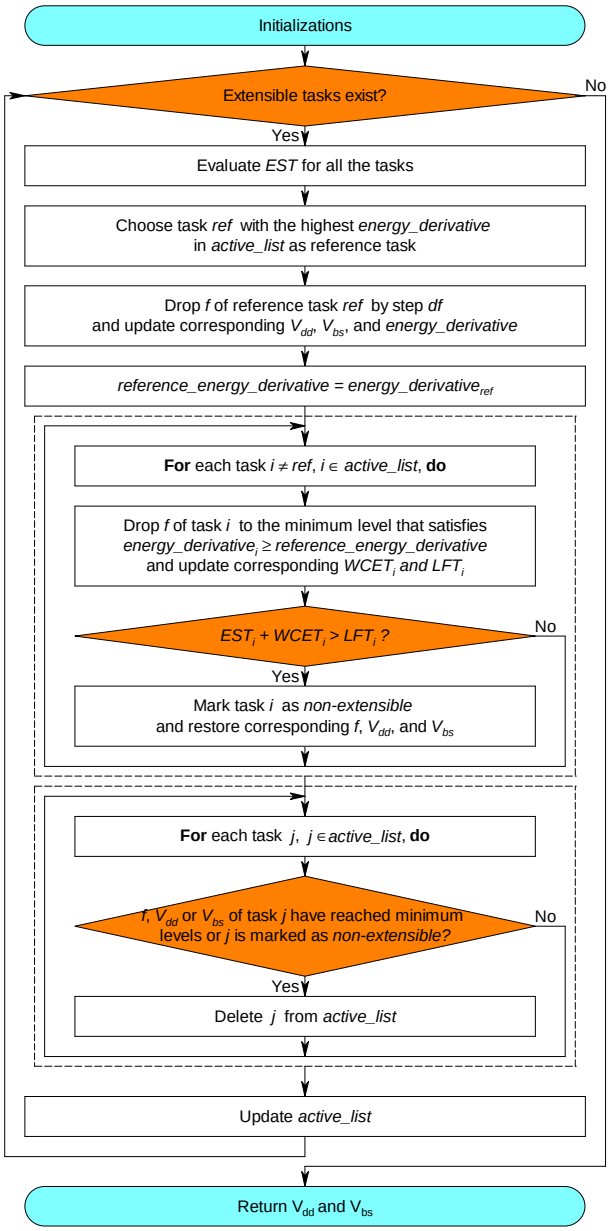


Fig. 8. Combined DVS and ABB based scheduling algorithm for distributed systems

Fig. 8 shows the flowchart of the proposed scheduling algorithm. The initial schedule is generated by list scheduling. The input of the algorithm is a directed graph $G(V,E)$, which is created based on the task graphs as well as the constraints imposed by resource sharing and link contention. V is the set of vertices including all the scheduled events in the initial schedule, and E is

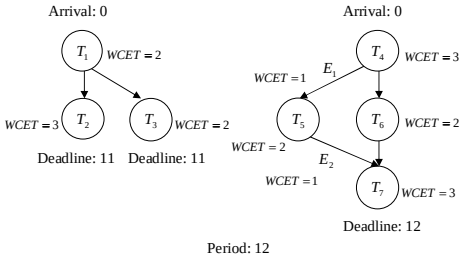


Fig. 9. Task graphs

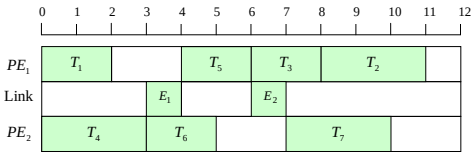


Fig. 10. The initial schedule

the set of directed edges between vertices, which represents the precedence relationships due to data dependencies and execution order constraints on any PE or communication link. We illustrate the creation of $G(V,E)$ through Example 2.

Example 2: Fig. 9 gives the specification of an embedded system in the form of two task graphs. Fig. 10 gives a feasible schedule for the task graphs on a distributed system consisting of two PEs connected by a link. We assume both PE_1 and PE_2 have communication buffers. The derived directed graph $G(V,E)$ for this schedule is shown in Fig. 11. A directed edge is inserted from one event to another if one is a direct predecessor of another (solid edges), or if one is scheduled just ahead of another on the same PE or communication link (dotted edges). A dummy event (small empty box) is inserted between two events to represent the transition time overhead for clock frequency, supply voltage, and body bias voltage, wherever a possible transition may occur. □

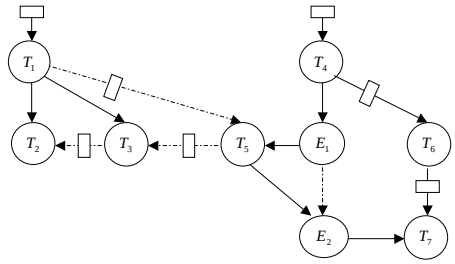


Fig. 11. The directed graph $G(V,E)$

In the algorithm, initially, the frequency f, supply voltage V_{dd}, and body bias voltage V_{bs} of all the tasks are initialized to f_{max}, V_{ddmax} and V_{bsmax} of the PEs they are assigned to, respectively. The tasks assigned to the voltage-scalable PEs are marked as *extensible* and their *energy_derivative*s are calculated. We maintain two task lists, the extensible task list, *extensible_list*, and the active extensible task list, *active_list*. *extensible_list* is initialized with all the extensible tasks in decreasing order of *energy_derivative*. The extensible tasks with the highest *energy_derivative* are inserted into *active_list* from *extensible_list*. Whenever a task is added to *active_list*, it is removed from *extensible_list* at the same time. All the extensible tasks are thus either in *extensible_list* or in *active_list*. The following loop is repeated until there is no extensible task remaining in either *extensible_list*

or *active_list*.

In each iteration, for each event in the order of topological sort of $G(V,E)$, its *EST* is evaluated based on its current clock frequency f. Next, the task with the highest *energy_derivative* from *active_list* is chosen as the reference task, whose frequency is dropped by frequency scaling step df. Its corresponding V_{dd}, V_{bs}, and *energy_derivative* are evaluated based on the new scaled frequency. Its new *energy_derivative* is the reference energy derivative, *reference_energy_derivative*. Next, in the order of reverse topological sort of $G(V,E)$, the frequencies of all the other tasks in *active_list* are updated. The frequency scalings ensure that the corresponding *energy_derivative*s do not drop lower than the reference level, *reference_energy_derivative*. The frequency scalings of some tasks might influence the validity of the new schedule. For each scheduled event, its *LFT* is updated based on its extended execution time *WCET*. If $EST + WCET > LFT$, it means the frequency scaling of this task yields an invalid schedule which violates real-time constraints. Such a task is marked as *non-extensible* and its f_i, V_{di} and V_{bi} are restored to their old valid values. For each task in *active_list*, if any value of its frequency, supply voltage, or body bias voltage has reached its minimum level or it is marked as *non-extensible*, it is removed from *active_list*. Then *active_list* is updated in the following way. If *active_list* becomes empty, the extensible tasks with the highest *energy_derivative* are inserted into *active_list* from *extensible_list*. Otherwise, *active_list* is appended with tasks whose *energy_derivative*s are higher than *reference_energy_derivative* from *extensible_list*. If no extensible tasks are left, the algorithm terminates by returning the scaled supply voltage and body bias voltage levels.

The proposed algorithm has a complexity of $O(K(n+e)+(n+e)log(n+e))$, where n is the number of tasks, e is the number of inter-PE communication edges, and K is the number of iteration steps. During the initializations, reordering the extensible task list, *extensible_list*, in decreasing order of *energy_derivative* requires a time of $(n+e)log(n+e)$. In each iteration, the *EST* and *LFT* of scheduled events are evaluated to verify the validity of the schedule after frequency and voltage scaling. The schedule is invalid if it does not guarantee real-time constraints. Each iteration takes a linear time of $O(n+e)$. Suppose M is the maximum number of frequency scaling steps of voltage-scalable PEs given by $M = max_{i \in voltage-scalable\ PEs}(f_i^{max} - f_i^{min})/df$. K depends on the value of M and the power variations of executed tasks.

VI. Experimental results

In our experiments, we obtain system architectures through the system synthesis tool, called CORDS, described in [31]. CORDS synthesizes multi-rate, real-time, periodic distributed embedded systems. It automatically selects an allocation from a set of field programmable gate arrays (FPGAs), general-purpose processors, and communication resources. It assigns tasks to FPGAs and general-purpose processors and determines the connectivity of communication events. Finally, it derives schedules for tasks and communication events. Voltage-scalable PEs are allowed to have different maximum and minimum f, V_{dd}, and V_{bs}. We assume the transition time of V_{dd} and V_{bs} is $20\mu s$, which is a conservative estimate based on the current circuit technology [4, 20, 32]. When supply voltage changes from V_{dd1} to V_{dd2} and body bias voltage changes from V_{bs1} to V_{bs2}, transition energy is derived based on Stratakos's analysis [33]:

$$E_{tran} = (1-\tau)(|V_{dd2}^2 - V_{dd1}^2|C_r + |V_{bs2}^2 - V_{bs1}^2|C_s) \qquad (21)$$

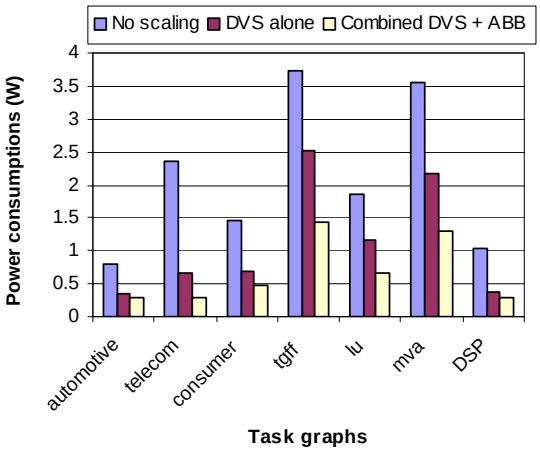


Fig. 12. Power consumption for three different schemes

where τ is the power efficiency, C_r is the capacitance of power rail, and C_s is the total capacitance of the substrate and wells of the device. The constant parameters for the $0.07\mu m$ process are provided in [26]. We assume 90% power efficiency.

To demonstrate the effectiveness of our combined DVS and ABB method for distributed real-time embedded systems, we applied our proposed real-time scheduling algorithm to embedded systems in the form of the task graphs described in Table III. The task graphs include: *automotive*, *telecom*, and *consumer*, as the name implies, based on automotive, telecom and consumer applications [34], *tgff* generated by a randomized task graph generator [35], *lu* and *mva* based on LU decomposition of a large matrix and mean value analysis [36], and *DSP* based on a DSP example [37].

TABLE III
Characteristics of embedded systems in the form of task graphs

Task graphs	#Tasks/#edges	#PEs/#links
automotive	69/3	2/1
telecom	88/15	2/1
consumer	12/5	3/1
tgff	53/22	3/1
lu	189/221	4/3
mva	361/52	2/1
DSP	120/27	4/2

We compare our combined DVS and ABB method with two other schemes, no voltage scaling and DVS alone. All the comparisons are based on the same initial schedule generated by as late as possible (ALAP) list scheduling. We first set the frequency scaling step df to 20% of the maximum frequency. Fig. 12 shows the power consumption for the benchmarks for the different schemes for the $0.07\mu m$ technology. It can be observed that the combined DVS and ABB method provides more power reduction than the other two schemes. Table IV gives power reduction of *DVS vs. no scaling*, *DVS+ABB vs. no scaling*, and *DVS+ABB vs. DVS alone*. The combined DVS and ABB method is more effective than DVS alone from the power reduction point of view. It yields an average power reduction of 34.7% with respect to using DVS alone, and 68.3% compared to no voltage scaling. This justifies the advantages of considering dynamic power consumption and leakage power consumption simultaneously during voltage selection to reduce power consumption for distributed real-time embedded systems.

TABLE IV

POWER REDUCTION COMPARISON BETWEEN DVS ALONE AND COMBINED DVS+ABB

Task graphs	*automotive*	*telecom*	*consumer*	*tgff*	*lu*	*mva*	*DSP*	Average
DVS vs. no scaling	56.8%	72.0%	53.9%	64.5%	32.9%	36.8%	38.4%	50.8%
DVS+ABB vs. no scaling	63.9%	87.2%	66.7%	70.9%	62.1%	63.6%	63.5%	68.3%
DVS+ABB vs. DVS	16.5%	54.4%	27.7%	18.0%	43.5%	42.4%	40.7%	34.7%

As discussed earlier, the complexity of our proposed real-time scheduling algorithm depends on the frequency scaling step df. If we change the frequency scaling step to 10% of the maximum frequency, the improvement in the power reduction achieved is observed to be negligible. We can thus adjust the frequency scaling step to provide a good tradeoff between algorithm performance and complexity.

VII. CONCLUSIONS

This paper proposed a new scheduling algorithm combining DVS and ABB to optimize both dynamic power consumption and leakage power consumption for distributed real-time embedded systems. The algorithm is based on a novel two-phase approach, which can trade off energy consumption and task execution time by performing DVS and ABB simultaneously. Experimental results show that the proposed algorithm can achieve substantial power reduction for the $0.07\mu m$ technology.

REFERENCES

[1] http://www.intel.com/design/intelxscale/.

[2] http://www.transmeta.com/technology/architecture/longrun.html.

[3] http://www.amd.com/.

[4] A. Chandrakasan, W. J. Bowhill, and F. Fox, *Design of High-Performance Microprocessor Circuits*. Wiley-IEEE Press, Sept. 2000.

[5] F. Yao, A. Demers, and S. Shenker, "A scheduling model for reduced CPU energy," in *Proc. Symp. Foundations Computer Science*, Oct. 1995, pp. 374–382.

[6] T. Pering, T. Burd, and R. Brodersen, "The simulation and evaluation of dynamic voltage scaling algorithms," in *Proc. Int. Symp. Low Power Electronics & Design*, Aug. 1998, pp. 76–81.

[7] F. Gruian and K. Kuchcinski, "LEneS: Task scheduling for low-energy systems using variable supply voltage processors," in *Proc. Conf. Asian South Pacific Design Automation*, Jan. 2001, pp. 449–455.

[8] J. Luo and N. K. Jha, "Power-conscious joint scheduling of periodic task graphs and aperiodic tasks in distributed real-time embedded systems," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2000, pp. 357–364.

[9] N. K. Bambha, S. S. Bhattacharyya, J. Teich, and E. Zitzler, "Hybrid search strategies for dynamic voltage scaling in embedded multiprocessors," in *Proc. Int. Wkshp. Hardware/Software Co-Design*, Apr. 2001, pp. 243–248.

[10] N. K. Jha, "Low power system scheduling and synthesis," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2001, pp. 259–263.

[11] J. Luo and N. K. Jha, "Power-profile driven variable voltage scaling for heterogeneous distributed real-time embedded systems," in *Proc. Int. Conf. VLSI Design*, Jan. 2003, pp. 369–375.

[12] D. Duarte, Y. Tsai, N. Vijaykrishnan, and M. J. Irwin, "Evaluating run-time techniques for leakage power reduction," in *Proc. Int. Conf. VLSI Design*, Jan. 2002, pp. 31–38.

[13] A. Abdollahi, F. Fallah, and M. Pedram, "Runtime mechanisms for leakage current reduction in CMOS VLSI circuits," in *Proc. Int. Symp. Low Power Electronics & Design*, Aug. 2002, pp. 475–478.

[14] R. X. Gu and M. I. Elmasry, "Power dissipation analysis and optimization of deep submicron CMOS digital circuits," *IEEE J. Solid-state Circuits*, vol. 31, no. 5, pp. 707–713, May 1996.

[15] A. Ferre and J. Figueras, "On estimating leakage power consumption for submicron CMOS digital circuits," in *Proc. Int. Wkshp. Power & Timing Modeling, Optimization & Simulation*, Oct. 1997, pp. 269–279.

[16] S. Bobba and I. N. Hajj, "Maximum leakage power estimation for CMOS circuits," in *Proc. IEEE Alessandro Volta Memorial Wkshp. Low-Power Design*, Mar. 1999, pp. 116–124.

[17] J. P. Halter and F. N. Najm, "A gate-level leakage power reduction method for ultra-low-power CMOS circuits," in *Proc. Conf. Custom Integrated Circuits*, May 1997, pp. 475–478.

[18] S. Mutoh, T. Douseki, Y. Matsuya, T. Aoki, S. Shigematsu, and J. Yamada, "1-V power supply high-speed digital circuit technology with multithreshold-voltage CMOS," *IEEE J. Solid-State Circuits*, vol. 30, no. 8, pp. 847–854, Aug. 1995.

[19] A. Keshavarzi, S. Ma, S. Narendra, B. Bloechel, K. Mistry, T. Ghani, S. Borkari, and V. De, "Effectiveness of reverse body bias for leakage control in scaled dual-Vt CMOS ICs," in *Proc. Int. Symp. Low Power Electronics & Design*, Aug. 2001, pp. 207–212.

[20] T. Kuroda, T. Fujita, S. Mita, T. Nagamatsu, S. Yoshioka, K. Suzuki, F. Sano, M. Norishima, M. Murota, M. Kako, M. Kinugawa, M. Kakumu, and T. Sakurai, "A 0.9-V 150-MHz 10-mW, 4mm^2, 2-D discrete cosine transform core processor with variable-threshold-voltage (VT) scheme," *IEEE J. Solid-State Circuits*, vol. 31, no. 11, pp. 1770–1779, Nov. 1996.

[21] K. Nose, M. Hirabayashi, H. Kawaguchi, S. Lee, and T. Sakurai, "V_{TH}-hopping scheme for 82% power saving in low-voltage processors," in *Proc. Custom Integrated Circuits Conf.*, May 2001, pp. 93–96.

[22] C. H. Kim and K. Roy, "Dynamic V_{TH} scaling scheme for active leakage power reduction," in *Proc. Design, Automation & Test in Europe Conf.*, Mar. 2002, pp. 163–167.

[23] V. Kaenel, M. Pardoen, E. Dijkstra, and E. Vittoz, "Automatic adjustment of threshold and supply voltages for minimum power consumption in CMOS digital circuits," in *Proc. Int. Symp. Low Power Electronics*, Aug. 1994, pp. 78–79.

[24] D. Duarte, N. Vijaykrishnan, M. J. Irwin, H.-S. Kim, and G. McFarland, "Impact of scaling on the effectiveness of dynamic power reduction schemes," in *Proc. Int. Conf. Computer Design*, Sept. 2002, pp. 382–387.

[25] M. Miyazaki, J. Kao, and A. Chandrakasan, "A 175mV multiply-accumulate unit using an adaptive supply voltage and body bias architecture," in *Proc. Int. Conf. Solid-State Circuits*, Feb. 2002, pp. 58–59.

[26] S. M. Martin, K. Flautner, T. Mudge, and D. Blaauw, "Combined dynamic voltage scaling and adaptive body biasing for lower power microprocessors under dynamic workloads," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2002, pp. 721–725.

[27] T. Sakurai and A. R. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," *IEEE J. Solid-State Circuits*, vol. 25, no. 2, pp. 584–594, Apr. 1990.

[28] N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design*. Addison-Wesley, 1993.

[29] A. Chatterjee, M. Nandakumar, and I. Chen, "An investigation of the impact of technology scaling on power wasted as short-circuit current in low voltage static CMOS circuits," in *Proc. Int. Symp. Low Power Electronics & Design*, Aug. 1996, pp. 145–150.

[30] E. L. Lawler and C. U. Martel, "Scheduling periodically occuring tasks on multiple processors," *Information Processing Letters*, vol. 7, pp. 9–12, Feb. 1981.

[31] R. P. Dick and N. K. Jha, "CORDS: Hardware-software co-synthesis of reconfigurable real-time distributed embedded systems," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 1998, pp. 62–68.

[32] T. Burd and R. Brodersen, "Design issues for dynamic voltage scaling," in *Proc. Int. Symp. Low Power Electronics & Design*, July 2000, pp. 9–14.

[33] A. Stratakos, "High-efficiency low-voltage DC-DC conversion for portable applications," Ph.D. dissertation, Univ. of California, Berkeley, 1998.

[34] "E3S: The embedded system synthesis benchmark suite," at http://www.ee.princeton.edu/~cad/projects.html.

[35] R. P. Dick, D. L. Rhodes, and W. Wolf, "TGFF: Task graphs for free," in *Proc. Int. Wkshp. Hardware/Software Codesign*, Mar. 1998, pp. 97–101.

[36] Y. Kwok and I. Ahmad, "Dynamic critical-path scheduling: An effective technique for allocating task graphs to multiprocessors," *IEEE Trans. Parallel & Distributed Systems*, vol. 7, no. 5, pp. 506–521, May 1996.

[37] C. M. Woodside and G. G. Monforton, "Fast allocation of processes in distributed and parallel systems," *IEEE Trans. Parallel & Distributed Systems*, vol. 4, no. 2, pp. 164–174, Feb. 1993.