GDP: A Greedy Based Dynamic Power Budgeting Method for Multi-Core Systems in Dark Silicon

***Abstract*—**

I. Introduction

Cite sample [?].

A. Background

II. New Method

A. Problem summary

For multi-core chip in dark silicon era, cores on the chip can not be all turned on at the same time due to the thermal constraint. In order to make full use of the chip resources, we need to decide the number of active cores as well as the corresponding specific distribution that lead to maximum chip performance. Usually, we can use the frequency for performance measurement. Which should be noticed is that, active cores on the chip may not run at its maximum supply voltage and frequency, instead, it can be applied with Dynamic Voltage and Frequency Scaling(DVFS).

B. Relation between voltage and frequency

Actually, the operating frequency of core is decided by its supply voltage, following equation can be used to shows the dependency of frequency on voltage:

$$f \propto (V - V_{th})^{\gamma}/V. \tag{1}$$

Where V is the transistor's supply, V_{th} is the threshold voltage, f is the operating frequency and the exponent γ is an experimentally derived constant decided by the process of chip.

With the above equation, we can further develop a linear equation relating frequency and voltage. First, we normalize the voltage and frequency to their maximum values as $V_{norm} = V/V_{max}$ and $f_{norm} = f/f_{max}$, then we can approximate a linear relationship with following equation:

$$V_{norm} = \beta_1 + \beta_2 \cdot f_{norm}, \tag{2}$$

where $\beta_1 = V_{th}/V_{max}$ and $\beta_2 = 1 - \beta_1$, value of β_1 can be considered to be around 0.4 for typical process[?]. From this equation, we can recognize that the frequency will have to be reduced to zero when supply voltage is reduced to V_{th}, which surely becomes the lower bound of voltage in DVFS operation.

This work is supported in part by National Natural Science Foundation of China under grant No. 61404024, in part by the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry.

C. Problem formulation

Power of cores on the chip can be divided to be dynamic power p_d and static power p_s, where

$$\begin{aligned} p_d &= ACV^2 f, \\ p_s &= VI_{leak}, \end{aligned} \tag{3}$$

where A is the fraction of gates actively switching and C is the total capacitance load of all gates, I_{leak} is the leakage current of core. Assume the dynamic power and static power with no DVFS of cores are $P_{d_0} = [p_{d_{0_1}}, p_{d_{0_2}}, \ldots, p_{d_{0_l}}]^{\mathrm{T}}$ $P_{s_0} = [p_{s_{0_1}}, p_{s_{0_2}}, \ldots, p_{s_{0_l}}]^{\mathrm{T}}$, where $p_{d_{0_i}}$ and $p_{s_{0_i}}$ are dynamic power and static power of i-th core respectively. When DVFS is performed, assume the voltage scaling ratio of the i-th core is α_i, then according to (**??**), the corresponding scaling ratio of frequency is $\theta_i = (\alpha_i - \beta_1)/\beta_2$, so the scaling ratio of dynamic power is $\lambda_i = \alpha_i^2 \theta_i$, according to (**??**). For static power, the effect of voltage and frequency scaling is much less than dynamic power, and its scaling ratio can be considered as α_i^{κ}, where κ is a process-related constant, for simplicity sake, we can assume $\kappa = 1$, in other words, we can ignore the effect of leakage's dependency on voltage. Also, we use Boolean variable $\xi_i \in \{0, 1\}$ to represent the i-th core is active or not: $\xi_i = 0$ represents inactive and $\xi_i = 1$ represents active situation.

As mentioned before, we could use operating frequency to measure the performance of cores, and DVFS operation determine the real frequency, our optimization goal can be expressed as $\sum_{i=1}^{l} \xi_i \theta_i f_{max_i}$. Because the main limit of chip performance is the cooling capacity, so we can express the steady temperature use thermal model and let it under the temperature threshold: $G^{-1}B\xi\left(\Lambda P_{d_0} + \alpha P_{s_0}\right) \prec T_{th}$. Where $\Lambda \in \mathbb{R}^{l\times l}$, $\alpha \in \mathbb{R}^{l\times l}$ and $\xi \in \mathbb{R}^{l\times l}$ are all diagonal matrices, their i-th diagonal element are λ_i, α_i and ξ_i respectively. $\prec$ is vector inequality, for m-dimensional vector, u and v, $u \prec v$ represents that $u_i < v_i$ for i = 1,2,. . .,m.

Finally, we can express the whole optimization problem using following equation:

$$\begin{aligned} &\text{maximize} \sum_{i=1}^{l} \xi_i \theta_i f_{max_i} \\ &\text{subject to} \begin{cases} G^{-1}B\xi\left(\Lambda P_{d_0} + \alpha P_{s_0}\right) \prec T_{th} \\ \alpha_i \in (0, 1]\,, i = 1, 2, \ldots, l \\ \xi_i \in \{0, 1\}\,, i = 1, 2, \ldots, l \end{cases} \end{aligned} \tag{4}$$

Solving this optimization problem, we can get the best α_i and ξ_i, then get the best number and distribution of active cores using the number of non-zero elements in ξ, and also get the power budget of cores with the value of α_i. A basic solution for the above equation is to traverse all possible values of λ_i, α_i and ξ_i of all cores under the thermal limit, and choose the values that result in maximum total performance. However, although this method can allow us to find the optimal solution, the time consumption is unacceptable, with number of cores increasing, this method will soon be useless.

D. Problem approximation

First, for a certain core on the chip, we consider the relation between the total power p_{tot_i} and the scaling ratio of operating frequency θ_i:

$$\begin{aligned} p_{tot_i} &= p_d + p_s \\ &= (\beta_2\theta_i + \beta_1)^2\theta_i p_{d_{0_i}} + (\beta_2\theta_i + \beta_1)p_{s_{0_i}}. \end{aligned} \tag{5}$$

It is easy to see that θ_i and p_{tot_i} are third-order related, and θ_i will monotonically increase as P_{tot_i} increase. We also test this relationship for a certain chip and get the result as shown in Fig. **??**, and we can see that these two variables are approximately of linear relation. What is more, we can easily prove that for a certain dark-silicon chip under a thermal constraint T_{th} and certain number of active cores, the summary of total power of all active cores $\sum_{i=1}^{l} p_{tot_i}$ will achieve the maximum value just when temperatures of all active cores achieve T_{th}.

Inspired by these two facts, if we assume that p_{tot_i} and θ_i are just linearly related, then we can change our optimal goal to be $\sum_{i=1}^{l} p_{tot_i}$ while keeping the constraint conditions. Thus, we can take some advantages from this approximation and simplify the solving process. When the number of active cores is fixed as m, as well as the detail distribution of these m active cores, which means the matrix ξ is fixed ,too. Let $A = G^{-1}B\xi$ and we can simplify the first constraint condition in (**??**) as $Ap_{tot} \prec T_{th}$, where $p_{tot} = [p_{tot_1}, p_{tot_2}, \ldots, p_{tot_l}]^{\mathrm{T}}$. Assume the m columns of A corresponding to these fixed m active cores are collected into matrix $A_m \in \mathbb{R}^{m\times l}$, and the total power budget of them are expressed as vector $Ptot_m \in \mathbb{R}^{m\times 1}$. Then, we can form the following optimization problem to describe the approximated problem:

$$\begin{aligned} &\text{minimize } \|T_{th} - A_m P_{tot_m}\|_2 \\ &\text{subject to } A_m P_{tot_m} \prec T_{th} \end{aligned} \tag{6}$$

This optimization problem can be equivalently transformed into a quadratic programming (QP) problem as

$$\begin{aligned} &\text{minimize } p_{tot_m}^{\mathrm{T}} A_m A_m p_{tot_m}^{\mathrm{T}} - 2T_{th}^{\mathrm{T}} A_m p_{tot_m}^{\mathrm{T}} + T_{th}^{\mathrm{T}} T_{th} \\ &\text{subject to } A_m P_{tot_m} \prec T_{th} \end{aligned} \tag{7}$$

which can be solved efficiently by calling the standard QP solving routines for the power budget P_{tot_m}. With p_{tot_m}, we can easily calculate the maximum chip performance for these fixed active-core number m and fixed active-core distribution.

Based on these derivation, we can generate a better method to solve the original problem: we can go through all possible active-core numbers and all possible distributions for a certain chip and calculate the corresponding performance, and choose the one result in global maximum total performance as the best solution. This improved method can save some time of testing all possible value of α_i and λ_i when m and ξ are fixed by directly calculate the maximum chip performance using (**??**), but the cost of going through all possible active-core numbers and active-core distributions are still huge and make it hard to be realized.

E. use greedy algorithm to solve the optimization problem

To reduce the time cost and find a acceptable approximate solution, we propose a greedy algorithm to replace the ergodic method. For a chip with m total cores, the basic idea of our proposed greedy algorithm can be described as follow: first we find a sub-optimal distribution for each active core number from 1 to m, together with the total chip performance, then compare the chip performance with different number of active cores and get the best active-core number which result in best chip performance, as well as the corresponding distribution.

The details of finding sub-optional for certain active-core number m is as follow: we first find the optimal solution for only one active core. Then, we keep the first active core position determined by first step, and find the optimal solution of two cores, with the second active core position determined. Such strategy will be iteratively processed for m steps, and finally we can arrive at a sub-optimal solution for m active cores.

Naturally, the first active position affect the subsequent active positions a lot, especially when the total active-core number m is small. That is because the first active core will determine the second position and thus the 3rd one, ... untill the last one. A natural idea to relieve this is to add more choices for the starting position instead of only the optimal one, then start from different core position to find the sub-optimal distribution for the certain active-core number m, at last, compare the total performance resulted by different beginning position and finally determine the best distribution for active cores. Take the actual situation as a reference, we can make a reasonable assumption that the cores on the chip are distributed as square form. Cores distributed in this particular distribution form has strong symmetry, which enable us to decrease the choices of the first active core position without losing completeness, and thus save a lot time to find the sub-optional distrubution for certain active-core number.

As shown in Fig. **??**, we can take a 16-core chip as example, which has only three different positions when determine the first active core when the symmetry of cores are considered. And it is easy to realize that for a chip with $n \times n$ cores, the different number of positions for first active core is just $(\bar{n}^2 + \bar{n})/2$, where $\bar{n}$=$\lceil n/2 \rceil$.

F. Temporary experimental result

We test our greedy algorithm on a chip with 4×4 cores and get some acceptable experimental results as we excepted. Fig. **??** shows the comparison of our new method and the basic ergodic method in finding the maximum total performance when a certain number of active cores is given. It is clear that the result of our new method is nearly the same as that of the time consuming ergodic method. We also show the distribution given by these two methods when the active-core number is 10 , which result in the maximum total performance for this

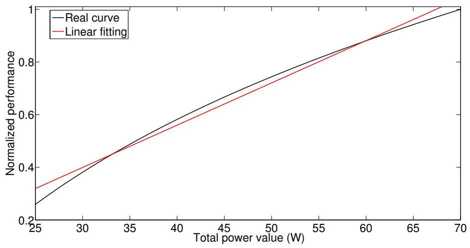


Fig. 1. The relation between total power and performance for a single core.

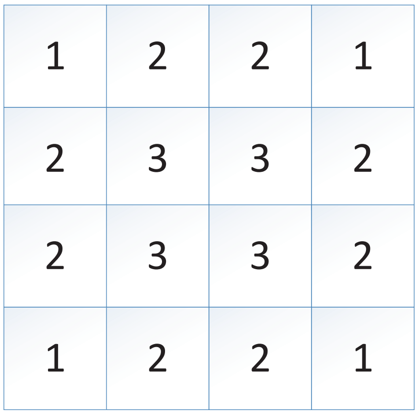


Fig. 2. An example of symmetry of a chip of 4×4 cores, cores marked with same number can be considered to be equivalent when determine the first active core.

16-core chip just as shown in Fig. **??**, as we can see, the two distribution are just the same because of the symmetry of cores. As for time cost, the ergodic method takes $13.507\,\mathrm{s}$ to go through all possible active-core numbers and distributions while our greedy method only cost $0.056\,\mathrm{s}$.

III. Experimental Results

IV. Conclusion

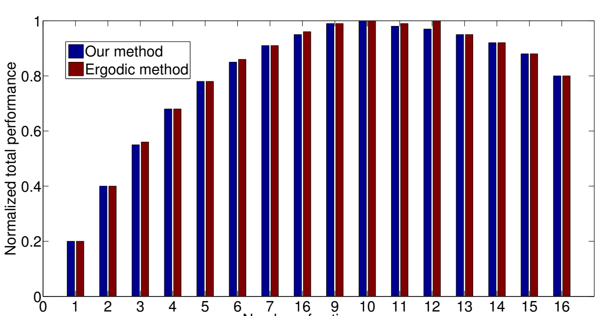
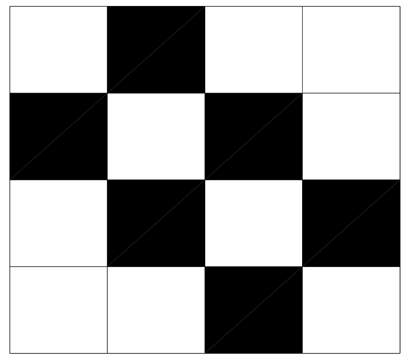
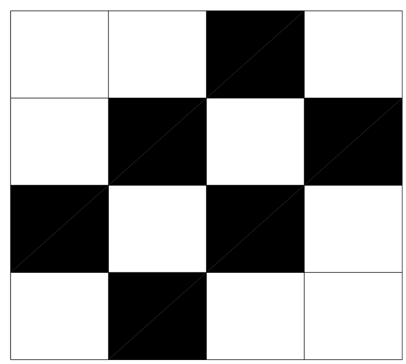


Fig. 3. Comparison of Normalized total performance given by two methods for different number of active cores.



(a) Best distribution given by ergodic method.



(b) Best distribution given by our new method.

Fig. 4. Comparison of distributions given by two methods when the active-core number is set to be 10.