Adaptive Power Allocation for Many-core Systems Inspired from Multiagent Auction Model

Xiaohang Wang[1], Baoxin Zhao[1], Terrence Mak[2,1], Mei Yang[3], Yingtao Jiang[3], Masoud Daneshtalab[4], Maurizio Palesi[5]
[1] Guangzhou Institute of Advanced Technology, CAS, China, Email: {xh.wang,bx.zhao}@giat.ac.cn
[2]The Chinese University of Hong Kong, China, Email: stmak@cse.cuhk.edu.hk
[3]University of Nevada, Las Vegas, USA, Email: mei.yang@unlv.edu, yingtao@egr.unlv.edu
[4]University of Turku, Finland, Email: masdan@utu.fi
[5]University of Enna, Kore, Italy, Email: maurizio.palesi@unikore.it

***Abstract*—Scaling of future many-core chips is hindered by the challenge imposed by ever-escalating power consumption. At its worst, an increasing fraction of the chips will have to be shut down, as power supply is inadequate to simultaneously switch all the transistors. This so-called dark silicon problem brings up a critical issue regarding how to achieve the maximum performance within a given limited power budget. This issue is further complicated by two facts. First, high variation in power budget calls for wide range power control capability, whereas most current frequency/voltage scaling techniques cannot effectively adjust power over such a wide range. Second, as the applications' behavior becomes more complicated, there is a pressing need for scalability and global coordination, rendering heuristic-based centralized or fully distributed control schemes inefficient. To address the aforementioned problems, in this paper, a power allocation method employing multiagent auction models is proposed, referred as Hierarchal MultiAgent based Power allocation (HiMAP). Tiles act the role of consumers to bid for power budget and the whole process is modeled by a combinatorial auction, whereas HiMAP finds the Walrasian equilibria. Experimental results have confirmed that HiMAP can reduce the execution time by as much as 45% compared to three competing methods. The runtime overhead and cost of HiMAP are also small, which makes it suitable for adaptive power allocation in many-core systems.**

Keywords—many-core; power allocation; multiagent

I. INTRODUCTION

Large, complex many-core chips, with dozens to hundreds or even thousands of integrated cores, are about to emerge by the year 2015 [1]. A critical issue in such many-core systems is their energy efficiency, which implies the pressing need to maximizing the performance over a power or energy budget. Traditionally, various power management methods in single-core or multi-core systems, based upon frequency/voltage scaling [2] [3, 4], use of sleep modes [5], etc., have been employed to help improve energy efficiency. However, further improvement of energy efficiency in future many-core systems faces the following challenges.

2) Need of globally coordinated control. On-chip resources have different implications on overall system performance and power consumption. For example, slowing down or shutting down a processor core running critical threads shall have a high penalty on the overall performance. Thus, heuristic-based methods [2] [3, 4] without considering the global application behavior often result in poor performance and/or low energy efficiency.

3) Need for hierarchical control. As the number of on-chip cores increases, centralized controllers [2] [3, 4] themselves will become the bottleneck, penalizing the performance or power consumption. A fully distributed approach, on the other hand, might miss the opportunity to find optimal solutions due to lack of global information. As such, a hierarchical approach is preferred due to its scalability and yet its ability to find optimal solutions by exploiting global information.

To meet all the aforementioned challenges imposed upon energy efficient many-core systems, in this paper, we propose an adaptive power allocation scheme named Hierarchical MultiAgent based Power allocation (HiMAP). HiMAP mimics a *multiagent auction process*, which can be modeled as a combinatorial auction game [6].

For the on-chip many-core systems, if we link the customers to cores and resources to power, we can readily borrow the above idea to form a multiagent auction process that can effectively allocate the scare resource (power budget) to the cores (customers). More precisely, in HiMAP, each core bids for the opportunity to become active at a given time interval based on the power budget. Agents are associated with the tiles and they are organized hierarchically so that de-centralized auctions are performed in accordance with an applicable power allocation policy. Our contributions can be summarized as follows.

1. HiMAP uses clock gating and frequency scaling to achieve a wide power control range in a globally coordinated fashion.
2. HiMAP follows a multiple agent auction process in a de-centralized way, which avoids the centralized control problem in current methods.
3. The experimental results confirm the superiority of HiMAP over other competing power allocation methods.

The paper is organized as follows. Section II reviews the existing power allocation methods. Section III presents the HiMAP in detail, followed by the experimental evaluation in Section IV. Finally, Section V concludes the paper.

978-3-9815370-2-4/DATE14/©2014 EDAA
This research program is supported by the Natural Science Foundation of China No. 61376024 and 61306024, Natural Science Foundation of Guangdong Province No. S2013040014366.

II. Related Work

Power consumption has been one of the major challenges to many-core chips. To curb the chip power consumption, power allocation methods [2] [3, 4] [7] have been adopted to control the power consumption of each on-chip resource. Power allocation methods can achieve energy efficiency by either 1) reducing power consumption [7] or 2) maximizing performance within a given power budget [2] [3, 4].

Almost all the aforementioned approaches are applied in a centralized manner. When the number of cores grows, however, the decision maker itself tends to become a bottleneck which causes severe performance and power consumption penalties.

To overcome the inefficiency in those centralized approaches, fully distributed control such as [8] has emerged whereas information is exchanged between neighboring cores. However, such heuristic-based approach lacks the global information of the application and thus, the result is likely to converge to a locally sub-optimal solution. Some of the de-centralized resource allocation methods for many-cores use multiagent models [9] or game theory [6] in decision making.

III. The HiMAP algorithm

A. Step 1: Determining the number of active tiles

Suppose there are N tiles in total, and let the system peak dynamic power be denoted as P. Each tile can run at a frequency $f_i \in \{F_1, \ldots, F_M\}$, with $F_1 \geq \ldots \geq F_M$. To keep N_j tiles active, the power consumption cannot be lower than $\frac{F_M}{F_1}\frac{N_j}{N}P$ (all the N_j tiles running at the lowest frequency F_M); and shall not exceed $\frac{N_j}{N}P$ (all the N_j tiles running at the highest frequency F_1). Thus, the actual power consumption falls into the range of $P_j = [\frac{F_M}{F_1}\frac{N_j}{N}P, \frac{N_j}{N}P]$. If voltage scaling is also supported, i.e., each tile can take a voltage from a finite set, $v_i \in \{V_1, \ldots, V_M\}$ with $V_1 \geq \ldots \geq V_M$, the total power needed to keep all the N_j tiles active becomes $P_j = [\frac{V_M^2}{V_1^2}\frac{F_M}{F_1}\frac{N_j}{N}P, \frac{N_j}{N}P]$. The number of active tiles thus can be determined by finding the active tile number N_j such that the input power budget,

$$P_{input} \in [\frac{F_M}{F_1}\frac{N_j}{N}P, \frac{N_j}{N}P].$$

B. Step 2: Determining which tiles to be active: combinatorial auction model

After the number of active tiles (N_j) is determined, all the tiles or the agents now can compete to become active at the next time interval. For example, if the active tile number is 16, only 16, out of the total 64 tiles competing for the right to become active, can be active at the next time interval. The problem now becomes,

Given the number of active tiles, decide which tiles can be active in the next interval, so as to minimize the execution time.

This competition process to become active can be modeled using the combinational auction model [6]. Each tile is associated with a tile agent and tiles are virtually grouped into clusters. Each cluster is also associated with a cluster agent which manages the requests from the tile agents within the cluster, and a global auctioneer manages the requests from the cluster agents.

Two auction processes, performed at various hierarchical levels, occur periodically as in Fig. 1.

- Global auction, which involves the cluster agents and the global auctioneer. Each cluster agent accumulates the bids of the corresponding tile agents and sends the accumulated bid to the global auctioneer. Upon receiving the bids from all the cluster agents, the global auctioneer then decides the number of active tiles $N_{j,m}$ and power budgets P_m' for each cluster m.
- Cluster auction within each cluster. Each cluster agent is an auctioneer and the corresponding tile agents bid to be active. Each tile agent sends its bid to the corresponding cluster agent, and the cluster agent decides who can be active.

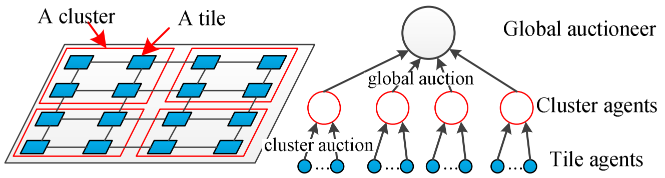

Fig. 1. The virtual partition of the many-core system and the corresponding auctions.

1) Valuation functions and prices

Valuation of a tile agent: Each tile agent t_i has a valuation defined in Eqn. (1).

$$v_i = corr_i / b_i - p_{i,t} \quad (1)$$

where b_i is the power parameter of each tile, $corr_i$ is the correlation of f_i w.r.t. the performance, and $corr_i / b_i$ is the performance-power ratio. Larger $corr_i/b_i$ indicates higher performance benefit to keep tile t_i active at the next interval, corresponding to the “bid” offered by an agent. $p_{i,t}$ is the price a winner agent has to pay.

To obtain the correlation value $corr_i$, a training process can be performed either on-line or off-line. At the k-th training interval, the frequencies of the tiles are set randomly with a vector $\overrightarrow{f_k} = < f_1, \ldots, f_N >$ and we measure the cycles $Cycle_k$. With K training intervals, the training data $\{< \overrightarrow{f_k}, Cycle_k >, k = 1, \ldots, K\}$ are collected. Eqn. (2) can be used to determine the correlation of frequency f_i and performance.

$$corr_i = \frac{\text{cov}(f_i, cycle)}{\sigma_{f_i}, \sigma_{cycle}} \quad (2)$$

where cov is covariance, and σ means standard deviation.

Valuation of a cluster agent: For a cluster agent m, the aggregated valuation is approximated as,

$$\bar{v}_m = (\sum_i v_i) \cdot N_{j,m} \quad (3)$$

if this cluster agent is requesting $N_{j,m}$ active cores. Each v_i is the valuation of a tile agent in the same cluster.

Price: The price $p_{i,t}$ of an agent *i* (a cluster agent or a tile one) is a term each winner has to pay to ensure fairness. $p_{i,t}$ can be defined as in Eqn (4).

$$p_{i,t} = \begin{cases} p_{i,t-1} + \gamma, \text{if } t_i \text{ is ON at } t-1 \\ 0, \text{ if } t_i \text{ is OFF at } t-1 \end{cases} \quad (4)$$

where γ is a user-defined parameter. If a tile becomes active (being a winner), a price counter starts to count. Otherwise, the price counter of a loser is reset to be 0. $p_{i,t}$ ensures fairness, i.e., the longer time a tile is active, the higher probability that it will be switched OFF in the future.

2) Determining the active tiles: mapping to the combinatorial auction model

Agents and auctioneers: In the global auction, there are *M* cluster agents involved in bidding for N_j items. The bundle *S* is the $N_{j,m}$ active cores, and the valuation *v*(*S*) of each cluster is the aggregated valuations of the tile agents within the same cluster as in Eqn (3).

In the cluster auction, there are *N/M* tile agents bidding for $N_{j,m}$ items in each cluster, where *N* is the network size and *M* is the cluster number. Whether the bundle *S* of each tile agent is to be active or not is dependent on whether a tile agent can be ON, and the valuation $v(S) = v_i$.

Allocation policies: In the global auction, the objective is to maximize the social welfare, i.e., $\max \sum_m \bar{v}_m, m = 1,...,M$. The allocation policy in the global auction is: allocating the active cores to cluster *m* which has the highest aggregated valuation first, then to the cluster with the second highest valuation if $N_j > N_{j,m}$, and so on. The allocation policy in the cluster auction is: setting the top $N_{j,m}$ tiles with the highest valuations to be ON. Both of those two allocations satisfy the demand request and can achieve a Walrasian equilibrium [6].

C. Step 3:Frequency setting over a given power budget

In the last step, the frequencies of the ON tiles are set under the power budget inside each cluster, which can be achieved by the optimal dynamic programming network based method [10].

IV. Experimental Results

A. Experimental setup

Experiments are performed using an in-house developed event-driven many-core simulator used in [11]. The simulation configuration and power models can also be found in [11]. The allowable frequencies are {1GHz, 800MHz, 500MHz, 330MHz}. Table I lists the mixes of the benchmarks selected from PARSEC and SPLASH-2.

In all the experiments, we select an 8×8 2D mesh as the underline NoC topology with 4 clusters each including a 4×4 mesh block. We compare the performance of the proposed HiMAP against three other best known schemes: (1) PGCapping [4], where both the number and the frequency of tiles can be adjusted, (2) PEPON [3], where the frequency of each processor and last-level cache bank can be adjusted, and (3) DPPC [2], a linear programming based approach.

In what follows, the impact of the price is first studied, followed by the performance comparison of the proposed HiMAP against that of related approaches. In the end, the cost and overhead of HiMAP are analyzed.

TABLE I. Workload mixes used as multiple application running in a single NoC

Mix-1	blackscholes, ferret, freqmine, swaptions
Mix-2	streamcluster, dedup, canneal, vips
Mix-3	barnes, raytrace, swaptions, vips
Mix-3	dedup, freqmine, fft, canneal

B. Impact of the auction price

A larger price ensures more fairness among the agents, while a smaller price favors agents who have more contribution to the overall performance. We use the standard derivation of the tiles' correlations (Eqn . (2)) to indicate the tiles' performance contribution, i.e.,

$$\sigma_{corr} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(corr_i - \overline{corr})} \quad (5)$$

where $\overline{corr}$ is the expectation of the tiles' correlations.

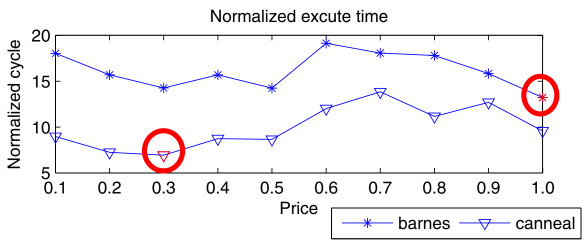

Fig. 2. The impact of price on the execution time, for benchmarks *barnes* and *canneal*. The x-axis is the price-to-valuation ratio. The optimal prices can be found at 1 and 0.3 which results in minimum execution time for *barnes* and *canneal*, respectively. The cycles are the optimal prices of the two benchmarks.

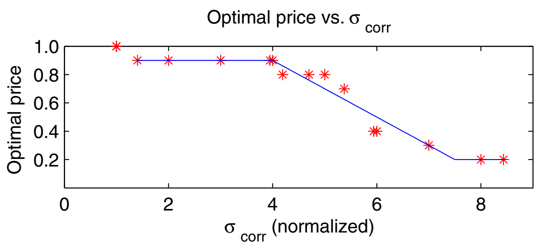

Fig. 3. The relationship of the standard derivation of tiles correlations vs. optimal price. The optimal price is obtained for each benchmark as in Fig. 2.

For *each* benchmark, we find the optimal price or γ (Eqn. (4)). For example, the benchmarks *barnes* and *canneal* have price-to-valuation ratios equal to 1 and 0.3 that produce the minimum execution time respectively, as shown in Fig. 2. Following the same manner, the two parameters, (σ_{corr}, optimal price), of each benchmark are plotted in Fig. 3. From Fig. 3 the relationship between the optimal price and the standard derivation of the tiles' correlations (tiles' performance contribution) can be found by regression,

$$\gamma = \begin{cases} 0.9, \sigma_{corr} < 4, \\ -0.35 \times \sigma_{corr} + 2.3, 4 \le \sigma_{corr} \le 6, \\ 0.2, \sigma_{corr} > 6, \end{cases} \quad (6)$$

An explanation to Eqn (6) is straightforward. Larger standard derivation of the tiles' correlations indicates wider gap of performance contributions among the tiles. Thus, a lower price would favor the tiles with higher performance contribution, which resembles that some of the customers bid with much higher valuations and rent the cars for a longer time. In contrary, smaller standard derivation indicates the gap in performance contributions of the tiles is not very obvious. Thus, a higher price to pay would ensure fairness among the tiles, which resembles that each customer bids with similar valuations and each holds the car with equal time period.

C. Performance evaluation of HiMAP

Fig. 4 shows the performance of the four power allocation methods when the power budget is high, 150W as in (a), and low, 50W as in (b). The execution time is measured as the total execution cycles of all the applications in one mix. From Fig. 4, one can see that HiMAP has the lowest execution time under both high and low power budgets. When the power budget is high, HiMAP records an average of 36 %, 35 % and 40% less execution time than that of PGCapping, PEPON, and DPPC, respectively. When the power budget becomes low, on average, HiMAP needs 38 %, 42 % and 45 % less execution time than that of PGCapping, PEPON, and DPPC, respectively.

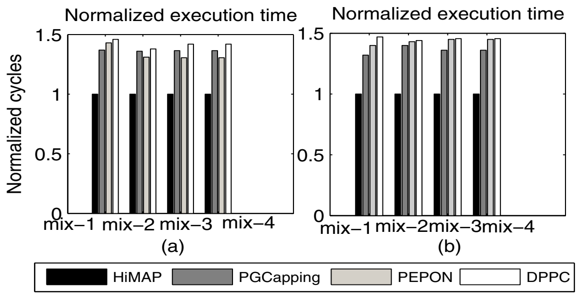

Fig. 4. Execution time of the application mixes under power budget of (a) 150W and (b) 50W.

D. Overhead of HiMAP

The overhead of the proposed HiMAP is attributed to two sources: the communication cost and calculation time of agents.

The agents and the auctioneers need to communicate with each other as they participate in both the global and the cluster auctions. In the global auction, each cluster sends the aggregated valuations to the global auctioneer followed by the active core numbers sent back to each cluster agent. This procedure involves 8 messages, each with a payload of 32 bits. For each cluster auction, each tile sends the valuations to the cluster agent, and waiting to be turned ON or OFF. This procedure involves 32 messages in total, each with a payload of 32 bits.

The execution time of the agents is within a few hundred cycles, four orders of magnitude less than that of the interval measured as 1M cycles.

Above analysis has shown that, the multiagent auctions have low communication and computing overhead. The last step of frequency scaling using a hardware dynamic programming network also has low area and power overhead, e.g., 14 % and 42% of a single router [10].

V. CONCLUSION

In this paper, a hierarchical power allocation method, HiMAP, has been proposed for improving energy efficiency in many-core systems. HiMAP is based on the multiagent auction models where the cores are viewed as customers competing for the power budget through two types of auction processes, i.e., a global auction and cluster auctions. Tiles are grouped into clusters and each tile tries to be switched ON in the cluster auctions. The clusters, in turn, bid for power budget under a global auctioneer. The power allocation policies described in this paper can achieve the Walrasian equilibra which maximizes the overall performance (mimicking the social welfare system). The experimental results have confirmed that HiMAP can reduce the application execution time as much as 45%, given different power budgets.

REFERENCES

[1] S. Borkar, "Thousand core chips: a technology perspective," in Proc. Design Automation Conf, pp. 746-749, 2007.

[2] K. Ma, X. Wang, and Y. Wang, "DPPC: dynamic power partitioning and control for improved chip multiprocessor performance," IEEE Trans. Computers, in press, 2013.

[3] A. Sharifi, A. K. Mishra, S. Srikantaiah, M. Kandemir, and C. R. Das, "PEPON: performance-aware hierarchical power budgeting for NoC based multicores," in Proc. Int'l Conf. Parallel Architectures and Compilation Techniques, pp. 65-74, 2012.

[4] K. Ma and X. Wang, "PGCapping: exploiting power gating for power capping and core lifetime balancing in CMPs," in Proc. Int'l Conf. Parallel Architectures and Compilation Techniques, pp. 13-22, 2012.

[5] S. Reda, R. Cochran, and A. Coskun, "Adaptive power capping for servers with multi-threaded workloads," IEEE Micro, vol. 32, no. 5, pp. 64-75, 2012.

[6] N. Nisan, Algorithmic game theory: Cambridge University Press, 2007.

[7] C. Li, A. Qouneh, and T. Li, "iSwitch: coordinating and optimizing renewable energy powered server clusters," in Proc. Int'l Symp. Computer Architecture, pp. 512-523, 2012.

[8] Y. Ge, Q. Qiu, and Q. Wu, "A multi-agent framework for thermal aware task migration in many-core systems," IEEE Trans. Very Large Scale Integration Systems, vol. 20, no. 10, pp. 1758-1771, 2012.

[9] A. Faruque, M. Abdullah, R. Krist, and J. Henkel, "ADAM: run-time agent-based distributed application mapping for on-chip communication," in Proc. Design Automation Conf., pp. 760-765, 2008.

[10] X. Wang, Z. Li, M. Yang, Y. Jiang, M. Daneshtalab, and T. Mak, "A low cost, high performance dynamic-programming-based adaptive power allocation scheme for many-core architectures in the dark silicon era," Accepted by ESTIMedia, 2014.

[11] X. Wang, T. Mak, Y. Jiang, M. Yang, M. Daneshtalab, and M. Palesi, "On self-tuning networks-on-chip for dynamic network-flow dominance adaptation," in Proc. IEEE/ACM Int'l Symp. Networks on Chip, pp. 1-8, 2013.