Redeeming Chip-level Power Efficiency by Collaborative Management of the Computation and Communication

Ning Lin[1,2] Hang Lu[1,2], Xin Wei[1,2] and Xiaowei Li[1,2]
State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences[1]
University of Chinese Academy of Sciences[2]
{ linning, luhang, weixin, lxw}@ict.ac.cn

Abstract—Power consumption is the first order design constraint in future many-core processors. Conventional power management approaches usually focus on certain functional components, either computation or communication hardware resources, trying to optimize its power consumption as much as possible, while leave the other part untouched. However, such unilateral power control concept, though has some potentials to contribute overall power reduction, cannot guarantee the optimal *power efficiency* of the chip. In this paper, we propose a novel **Co**llaborative management approach, coordinating both **Com**putation and **Com**munication infrastructure in tandem, termed as *CoCom*. Apart from prior work that deals with power control separately, it leverages the correlations between the two parts, as the "key chain" to guide their respective power state coordination to the appropriate direction. Besides, it uses dedicated hybrid on-chip/off-chip mechanisms to minimize the control cost and simultaneously guarantee the effectiveness. Experimental results show that, compared with the conventional unilateral baselines, CoCom is able to achieve abundant power reduction with minimal performance degradation at the same time.

I. Introduction

The increasing power density has hindered the integration of transistors in modern many-core processors, from one generation to the next, along with the process technology scaling. Modern many-core processors seek to increase the active number of cores for as many as possible while at the same time, lower the operating voltage and frequency to bridge the gap between the computational strength and the target performance for parallel applications and scale-out workloads. To sustain such continuous power and performance demand, many power management schemes nails certain on-chip components, optimizing their corresponding power consumption, as a way to contribute the power minimization of the entire chip.

A majority of researches have typically focused on either computation (i.e. *cores*) or communication (Networks-on-Chip, *NoC*) components (or both, but separately) as the target to achieve chip-level power reduction. Some approaches [1, 2, 3] resort to exploring the best voltage and frequency operating point for each core, striving to obtain an acceptable power and performance portfolio, while communication fabric is simply ignored, or unrealistically assumed it to be capable of providing constant packet latency and fixed power consumption. Apart from different implementations targeting the computational resources, there are plenty of works targeting communicational infrastructure as well, ranging from power-aware NoC architectures [4, 5, 6, 7], bandwidth scaling [8], hop-distance (latency) optimizations [9, 10] and so forth, which all aim to passively accommodate core-to-memory and cache coherence streams in the network but oblivious to the associate core execution state on the spot. The design goal of these schemes is uniformly to procure maximum power reduction, on the NoC side, by solely leveraging the information of traffic conditions imposed by the cores.

Intuitively, the aforementioned unilateral approaches have substantial potentials to reduce power consumption, but with the fast integration of the many-core chip, we observe that the overall system power efficiency is no longer dominated by any certain functional component alone, like either cores or NoCs. Specifically speaking, from the computation side, in order to tackle the *utilization wall* problem, most of the cores are usually forced to work at the near threshold frequency and voltage, or remain dim/dark for most of the time. Under such circumstances, the fraction of system level power occupied by the communication fabric is hence on par with that of its computation counterpart [7, 9, 11]. The importance of power and performance efficiency of the NoC, though essentially non-trivial, is further magnified and simply assuming a computation-monopolized system is unpractical. On the other end of the spectrum, degrading V/F settings of the cores in batches could to some extent compensate for the performance loss in executing parallel applications, but a direct consequence is that the volume of the traffic imposed to the communication fabric is also reduced. Conventional power management institutes power control based on the current network traffic by recklessly diminishing the power state of the communication fabric accordingly, regardless of the severe blow inflicted to the instruction throughput on the computation side.

Unilateral control scheme suffices to provide the headroom for power reduction, but it brings no direct benefit for the *efficiency* of the chip. An oracle solution should account for the effective coordination of both *computation* and *communication* resources. However, this heightened demand faces two grand challenges that impede its deployment in many-core power management. First, it is difficult to comb the implications of power control operations issued to the different hardware resources. Sometimes, instrumenting power control to one side may harm the execution of the other side. Especially when

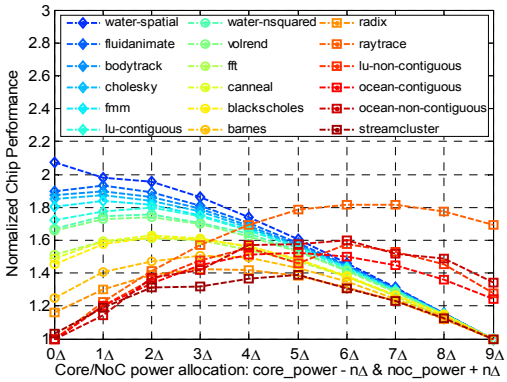


Figure 1 Chip power efficiency under different power allocations. The scaling of △ denotes the power quota shifted from cores to NoC. The whole chip power budget is fixed.

ASPDAC '19, January 21–24, 2019, Tokyo, Japan

ACM ISBN 978-1-4503-6007-4/19/01…$15.00
https://doi.org/10.1145/3287624.3287647

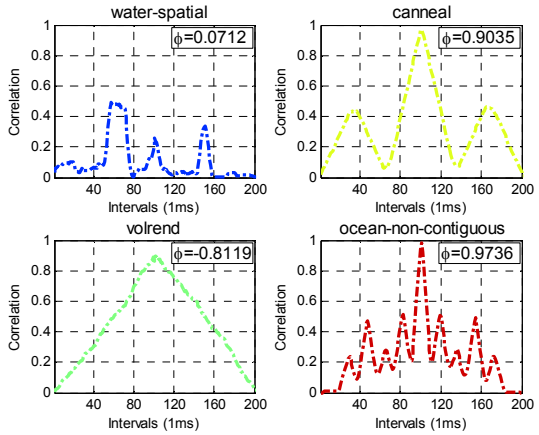


Figure 2 Different correlation scenarios of core and NoC performance counters. The sign of the coefficient (ϕ) denotes the correlation is positive or inverse.

cores spontaneously change its power state, the respond in NoC, i.e. upgrading or degrading bandwidth, would have unpredictable consequences apiece on the overall power efficiency, as reinforced in many research studies [8, 12]. Second, existing communication fabric design cannot effectively support the *collaborative* management, because of the connectivity maintenance and handovers between neighboring power control domains. That is also why many commercial many-core chips regulate the whole NoC at unitary voltage and frequency domain during the workload execution [13, 14], limiting the flexibility of the *chip-level* power control combined with the computation resources.

While in this paper, on top of the above analysis and challenges needed to be addressed, we propose a **Co**llaborative power management technique for the **Com**putation and **Com**munication resources in the many-core chip, namely **CoCom**, in an attempt to find a harmonious way to coordinate the two power consumers and obtain optimal system-level power efficiency compared with the classic unilateral schemes. CoCom intends to find the "key chain" between the performance counters of the two parts, taking it as the reference to guide the power state coordination to the right direction. CoCom uses a hybrid on-chip and off-chip mechanism (together with the proposed hardware) to, on one side, alleviate the computational cost by distributing power control knob in parallel, and on the other side, timely monitor and detect the opportunities of global performance acceleration or power reduction. The novelty of CoCom lies in exploring the *correlations* between the computation and communication resources and conjugate their respective power control effectively, instead of solely delving into one part while leaving the other part untouched as in previous schemes.

In the next section, we will elaborate the necessities of chip-level power control in manycores through exploring power (re)allocations between the computation and communication infrastructures, and specify the opportunities of the collaborative management by referring to the correlation of core/NoC performance counters.

II. Background and Motivation

A. Rethinking Unilateral Power Control

Despite the power reduction brought by any singular component, we found that the two instances have deep implications in terms of contributing chip-level power efficiency, as illustrated in Figure 1. We construct a 64-core many-core platform in our simulation framework, fixing the overall chip power budget but allocating different power quota to the computation and communication infrastructures. In the figure, 'Δ' is used to denote the power token shifted from cores to the NoC. We manipulate the core voltage and frequency to get close to the power quota allocated to it. The manipulation to NoC is analogous to cores by modulating on-chip router and packet traversal frequency, but

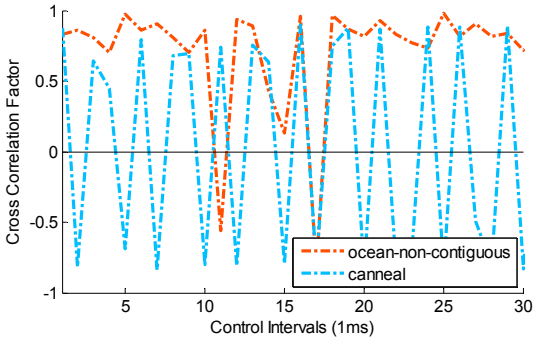


Figure 3 Cross correlation coefficient alters with workload phase changes.

strictly guarantee that the overall chip power budget is not violated. We scale the power quota shifted from Δ to 9Δ for each evaluated parallel workload and at each time, record the instruction throughput. The results in Figure 1 demonstrates an interesting phenomenon; that is, along with the scaling of power quota, almost all the workloads manifest a salient point at certain $n\Delta$ (i.e. for `radix` n=6; for `water-nsquared`, n=4). The evaluation motivates that application performance peaks at the proper power allocation between cores and NoC under fixed power budget, which hence denotes an optimal power efficiency.

Through the above observation, unilateral control concept should be reconsidered in large scale manycores. Figure 1 proves that chip power efficiency is not dictated by either computation or communication infrastructure. Instead, they should be carefully and comprehensively managed to contribute chip power efficiency as a whole.

B. Opportunities of Collaborative Management

Although in practical implementation, we cannot only focus on globally power allocation to each 'COM' once and for all, but on dynamic collaborations at each control interval for the running application. However, the complexity of phase changing and network traffic conditions make it difficult to find a uniform way to coordinate the power consumption of each part, let alone make them collaborate with each other. To address this challenge, we propose a novel method by analyzing the *cross correlation* of the core and NoC performance counters, as the key-chain to connect the two parts and guide their respective power control.

Cross correlation is widely used in system signal processing, as an effective method to identify the similarities of two time consecutive series, as a function of the lag of one relative to the other. We enroll this analytical method in many-core power management, exploiting it to determine whether the performance counters of different components are related with each other, and the point in time that the two performance counters are best aligned. The level of correlation is quantified by the *correlation coefficient*. In specific, we use core instruction throughput as well as the traffic towards memory hierarchy, both in consecutive fixed intervals, to construct the two sequential time series and calculate their cross correlation[1], as a way to locate the impact of core-memory accesses to instruction executions within the observing sets of intervals. As shown in Figure 2, we evaluate `4` parallel workloads on the same platform, each of which is extracted `100` discrete values of executed instructions on one random core and the packets injected into its router at an interval of `1ms`.

We found that the correlations do exist, with different behaviors that can be categorized into `4` patterns, and based on this information, power management could then be effectively guided for the core and the associate communication infrastructure. For example, workload `water-spatial` has a near-zero correlation coefficient (ϕ=0.0712) which indicates that the two time series are almost not correlated, in

[1] Cross correlation and coefficient is calculated by the following formulas:

$$R_{IM}(i) = \sum_{n=0}^{N-1} Inst(n) * Mem(i-n) \qquad \phi_{IM} = \frac{R_{IM}(0)}{\sqrt{R_{II}(0) \times R_{MM}(0)}}$$

Inst and *Mem* are the input instruction and memory traffic time sequence, respectively. $R_{IM}(i)$ is the output correlation series. $R_{II}(i)$ *and* $R_{MM}(i)$ is the auto-correlation of each sequence, and is calculated by the same way as $R_{IM}(i)$. φ is the correlation coefficient obtained with $R_{IM}(0)$, $R_{II}(0)$ and $R_{MM}(0)$.

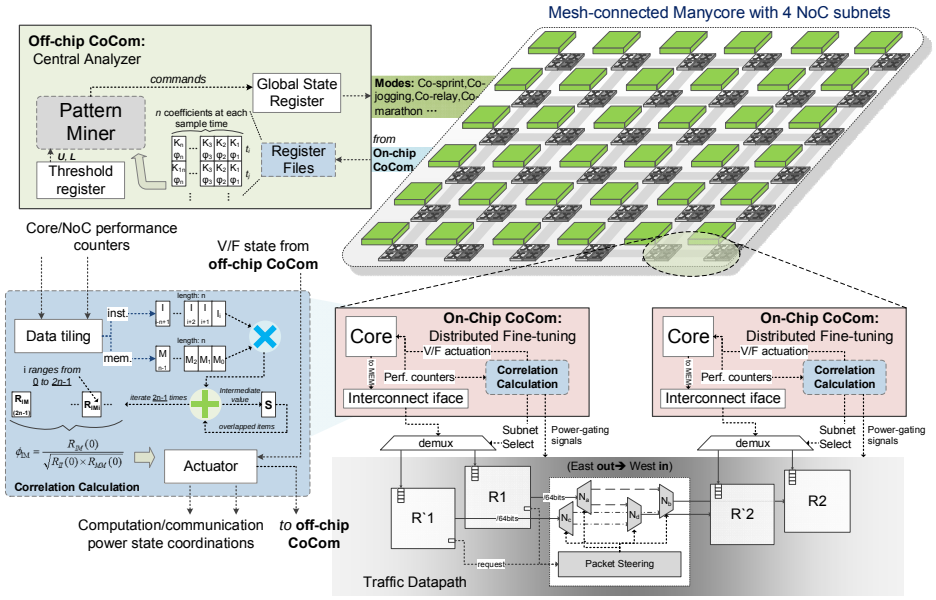


Figure 4 CoCom system architecture, primarily including two integrals: off-chip and on-chip CoCom. The distributed on-chip CoCom is responsible for monitoring core/NoC performance counters and calculate the correlations between them. Off-chip CoCom only collects the stats reported from each on-chip CoCom and analyzes global patterns exhibited by the workload.

other words, the core instructions, though imposing network traffic, is not directly influenced by the memory accesses or cache coherence and power management could be issued separately based on their respective performance states. By sharp contrast, `canneal` peaks its correlation at the `100ms` timestamp with a positive coefficient of `0.9035`. The value much closer to `1.0` means instruction execution behaves more similarly to memory access, which further means the two strongly correlated series vary in the same pace: increasing instruction throughput leads to more network injections, while high memory intensity in turn has severe impact on the instruction executions as well. Under this scenario, core and NoC power state could be collaboratively boosted or lowered to acquire optimal application performance or power efficiency. The coefficient of `volrend`, on the contrary, shows an inverse correlation (`-0.8119`), indicating that the two series behave towards opposite directions. Computation and communication power state should thereby be coordinated to the opposite levels, i.e. degrade core V/F setting but broaden communication bandwidth, or vice versa.

From the phase changing perspective, most workloads, though exhibiting different behaviors, demonstrate relatively stable patterns, as shown in Figure 3. The correlation coefficient remains relatively streamlined but fluctuates sporadically at some point (`ocean-non-contiguous`), while some other workload (`canneal`) shows a 'ping-pong' effect that swings frequently between positive and inverse correlations. This observed behavior could be leveraged as runtime guidance to determine application phase changes and in the next section, we will elaborate how the proposed CoCom implements chip-level power control based on these already explored correlations between computation and communication in manycores.

III. Collaborative Power Management in Manycores

A. System Design

CoCom architecture incorporates two major integrals: *on-chip* and *off-chip* CoCom, as shown in Figure 4. Specifically, on-chip CoCom is a set of distributed hardware agents associated with each core/router in a many-core processor that monitors core level performance counters, calculates correlation coefficients and actuates the final power control. Apart from conventional power management schemes for multi-core or small-scale manycores that employ a centralized manager to govern massive computations for each component, CoCom architects its hardware agents in a distributed way, to alleviate the computational overhead in power management. Each on-chip CoCom agent continuously probes core instruction throughput and memory accesses imposed to the communication infrastructure, and all agents work in parallel so local runtime information and power demands is able to be detected in an efficient manner, unnecessary to report to the central power manager. However, on-chip CoCom could only fine tune the computation/communication power state locally, lacking the global information that could be utilized to accelerate execution for higher performance, or gear down the execution for acceptable power reduction. Hence, off-chip CoCom is introduced in the framework, responsible for analyzing the coefficients collected from each on-chip CoCom and mining the trends within a historical observing interval, to determine the most appropriate operation modes for the computation and communication as a whole. For example, if the positive coefficients dominate most of the time in the window and emerges in most of the agents, off-chip CoCom hence uniformly operates the power state of the whole chip, because under such scenario, solely focusing on power fine-tuning on several locations will have very limited contribution to the global power efficiency.

B. Hardware Implementations

The hardware implementation of on-chip CoCom primarily targets the calculation of runtime correlations, as shown in Figure 4. It deals with two sorts of performance counters as input, which is: network injection that denotes memory access, prefetches or cache coherence traffic, and instruction throughput within the same observing interval that denotes the computation intensity. Cross correlation of the two time series are all about doing *convolutions*, so first of all, on-chip CoCom implements data tiling to either one of the series. The length of the series that involves convolution is configurable, and we construct the instruction throughput and the memory access series both with the same length, indicated by `n`. Two series carry out `multiply-and-accumulate (MAC)` one after another in the overlapped regions of the series to get an output correlation $R_{IM}(i)$. The MAC operation will iterate `2n-1` times to get all items of the output cross correlation series. Final correlation coefficient φ_{IM} is obtained with $R_{IM}(0)$, $R_{II}(0)$ and $R_{MM}(0)$, in which $R_{II}(0)$ and $R_{MM}(0)$ is the auto-correlation of each series calculated by the same way (`MACs`) as $R_{IM}(0)$.

Another obstacles to deploy collaborative power management in manycores stems from the communication fabric. Conventional fixed-bandwidth architecture nails one-case-for-all design concept that brings up considerable difficulties in fine-grained node-level power

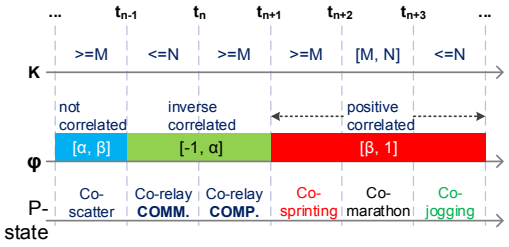


Figure 5 On-chip CoCom control mechanism. By referring to core instruction throughput and correlation coefficient, it fine tunes the power state of local core and router.

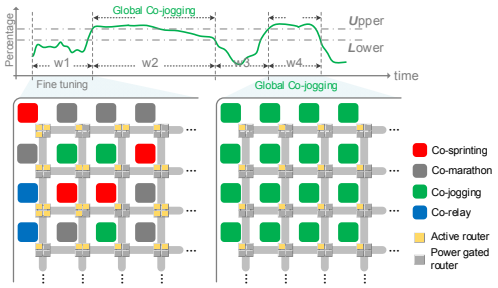


Figure 6 Off-chip CoCom control mechanism. By analyzing historical data in the observing window (*w1*, *w2*..), it determines the global mode of the chip (i.e. Co-jogging in the figure).

control [4, 8]. If we want the power state of communication fabric is tunable at each node, substantial synchronization buffers must be absorbed in each router to tackle the crossover between neighboring V/F domains, introducing severe power consumption and performance degradation [15]. While within CoCom context, we use the techniques proposed in [5] to circumvent this weakness. As shown in Figure 4, instead of the fixed-bandwidth design, communication fabric is comprised of several "subnetworks", and the traffic is allowed to be steered between different subnetworks for more precise local power adaptation, and the most important is that it enables fine-grained collaborative power control at each node. Detailed implementation could be referred to in [5].

C. *Mechanisms*

CoCom employs distributed on-chip agents to monitor and compute the runtime correlations, and at the same time uses an off-chip CoCom to analyze historical statistics to manipulate the power efficiency globally. The concrete procedure of this hybrid method fully relies on the values and variations of the correlation coefficients that can appropriately convey workload runtime demands. In specific, we define 5 modes in CoCom framework, both in on-chip fine-tuning aiming at local core/router combination, and the off-chip uniform control for optimal chip power efficiency, as shown in Figure 5. $\boldsymbol{K}$ is the derivatives of the instructions per cycle, which could be either positive or negative indicating the variations of instruction throughput. In conjunction with the runtime coefficient $\boldsymbol{\phi}$, we can identify the power modes suitable for local core and its associate router. We use four parameters to denote the boundaries of $\boldsymbol{K}$ and $\boldsymbol{\phi}$: $\boldsymbol{M}$ and $\boldsymbol{N}$ is the thresholds to identify the increment or decrement of instruction throughput, while $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are the boundaries identifying positive or inverse correlations for computation and communication intensity. If $\boldsymbol{\phi}$ falls between $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, it denotes computation and communication is not correlated with each other. Similarly, if $\boldsymbol{K}$ is larger than the preconfigured threshold $\boldsymbol{M}$ and simultaneously $\boldsymbol{\phi}$ exhibits a strong positive correlation, we define this phase as ***Co-sprinting***, in which case both computation (core) and communication (associate router) could boost their respective power state, to a higher level, to accelerate instruction execution and memory access in synergy. From the value of $\boldsymbol{\phi}$, we can infer that the computation intensity is strongly coupled

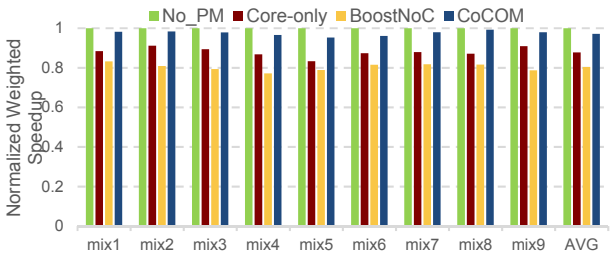


Figure 7 Performance evaluation (weighted speedup) of CoCom versus the baselines.

with communication intensity, and coordinating their power states to the same direction is supposed to improve the power efficiency in this location.

Similarly, we define other power modes depending on other $\boldsymbol{K}$- $\boldsymbol{\phi}$ combinations. ***Co-jogging*** is the case when computation exhibits a decrement beyond the threshold $\boldsymbol{N}$, and $\boldsymbol{\phi}$ also exhibits a positive correlation. The implication is that the decline of instruction execution also causes a synchronous decline in memory access. Lowering the computation and communication power state in synergy in this mode would be beneficial to attain abundant power reduction and graceful performance degradation. That is why we term it as 'jogging' --- both core and router work at a low speed. Besides, when $\boldsymbol{K}$ does not exhibit obvious variation (b/w [$\boldsymbol{M}$, $\boldsymbol{N}$] in Figure 5), on-chip CoCom regards that there is no need to issue power control and this mode is termed as ***Co-marathon***. For the inverse-correlated scenario (termed as ***Co-relay***), it means the increment of the computation intensity has a negative impact on the memory access. Power state coordination should be issued conversely for the two counterparts in this mode, i.e. boosting local network bandwidth and lowering core V/F setting (`COMM`. in the figure), or vice versa (`COMP`. in the figure). If they are not correlated ($\boldsymbol{\phi}$ b/w [$\boldsymbol{\alpha}$, $\boldsymbol{\beta}$]), we call this scenario as ***Co-scatter*** that core and network could issue power control freely in this mode, based on their own performance state, without concerning the impact to the other side.

Compared to on-chip CoCom, the mechanism of off-chip CoCom only accounts for monitoring and performing light-weight computations to analyze the patterns from historical observing windows. It receives the $\boldsymbol{K}$- $\boldsymbol{\phi}$ pairs from each on-chip CoCom and mines if a particular power mode dominates in the window. Taking "Co-jogging" as an example in Figure 6, off-chip CoCom contiguously monitors its percentage in window *w1*, and once it exceeds the threshold $\boldsymbol{U}$, off-chip CoCom mandates all the cores and the whole NoC working at the lowest power state to obtain a chunk of power reduction in *w2*. When time arrives at *w3*, a decrease in percentage beyond $\boldsymbol{L}$ is detected for this mode, so off-chip CoCom releases the control ownership back to each on-chip CoCom, restoring fine tuning at each location of the chip. The procedure for other modes is the same. Note that observing windows may not have the same length in off-chip CoCom, as opposed to conventional schemes with a fixed control interval. Off-chip CoCom only interferes fine tuning when $\boldsymbol{U}$ threshold is violated, or in other words, the running workload has exhibited obvious power demand.

IV. Evaluation

We use Graphite [16], a full system simulator for manycores as the basic simulation framework to evaluate our scheme. McPAT and DSENT power estimation tools are also integrated into the simulation platform for modeling precise power consumptions of all cores, NoC fabric and the dedicated on/off-chip CoCom hardware. We use a Tiled-like many−core architecture with 256 single issue, in-order cores. L1 I/D cache is privately occupied and last level cache is shared by all the cores. We run multi−programmed workloads selected from Parsec-3.0 and Splash-2 benchmark suite. We categorize 17 benchmarks selected from 'PARSEC-3.0' and 'Splash-2' benchmark suites and classify them into computation or memory intensive based on the prior profiling

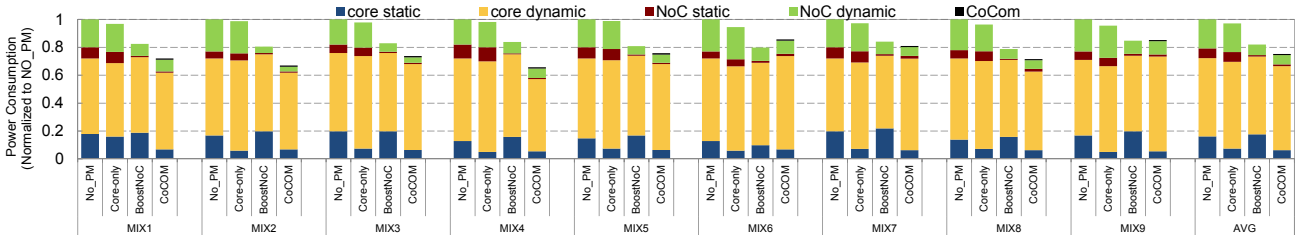
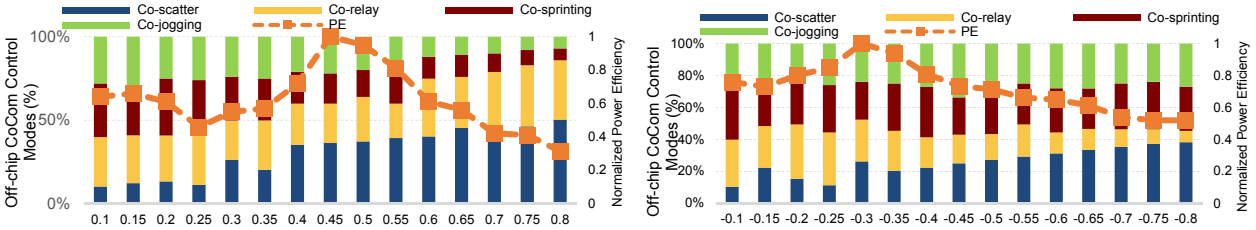


Figure 8 Power Consumption of CoCom versus various baselines under workload mixes (normalized to No_PM).



(a) α = -0.1 fixed, β from 0.1~0.8

(b) β = 0.1 fixed, α from -0.1~-0.8

Figure 9 CoCom dynamics analysis. Power efficiency result is normalized to the peak.

knowledge, and then create "bundles" that include a variety of mixed workloads. Since we use 4 benchmarks as a bundle, we allocate 64 threads for each of them and one thread is bounded to one core. We use 4 frequency levels for each core at the 32nm technology node: 0.6 GHz, 0.79 GHz, 1 GHz, 1.35 GHz and this configuration is also aligned with many commercial many-core processors like Intel SCC [13] (125 MHz→1.3 GHz) and Tile64 [14] (500 MHz→866 MHz). We also use 4 subnetworks in NoC [5] with each node independently tuned locally. On-chip CoCom may boost or degrade the power state of local core and router by one level at each time upon different modes. When off-chip CoCom interferes, all the resources may work at the highest level (i.e. Co-sprinting) or lowest level (i.e. Co-jogging) or two extremes (i.e. Co-relay), until off-chip CoCom releases the ownership back to the on-chip CoCom. We made a simple assumption that the communication between on/off chip CoCom hardware is negligible.

Within CoCom framework, it involves 6 parameters in total: `M`, `N`, α, β and `U`, `L`. Based on the profiling knowledge, we set `M=20%`, `N=-20%`, α`=-0.3`, β`=0.45`, `U=80%`, `L=60%` in the whole evaluation, but note that these parameters are all empirical values that could be configured based on different many-core platform and workload conditions. In terms of the baselines, we instantiate `3` cases: (1) `No_PM`: indicating no power management is ever implemented, and (2) `Core-only`: indicating that power management is only issued to computation infrastructure but leaves the communication untouched, and (3) `BoostNoC`: this is a state-of-the-art scheme [7] using two subnetworks to dynamically accommodate different communication intensities, but leaves core power state untouched. We use these baselines to prove the efficacy of CoCom in chip-level power efficiency control. For `No_PM`, the frequencies are tuned based on the variation of $\boldsymbol{K}$, and for `BoostNoC`, we fixed the core frequency to 1GHz, only coordinating the network bandwidth at a holistic manner, as reported in its paper.

A. *Power & Performance Tradeoff*

In this set of experiment, we evaluate the performance of all workload mixes under CoCom and the baselines. We use the metric "Weighted Speedup (WS)[2]" [17] as the representative. The result is shown in Figure 7. CoCom incurs only `2.9%` performance degradation compared with `No-PM`, but it outperforms `Core-only` and `BoostNoC` by `10.8%` and `20.9%` in power reduction respectively. The benefits come from the collaborative management and again it proves that unilateral approach limits the headroom that power efficiency could attain in practical: `Core-only` changes power state based on the variations of instructions executed ($\boldsymbol{K}$), agnostic of the criticality of the communication infrastructure to the chip performance, so it exhibits severe performance degradation compared with `No_PM`. While `BoostNoC` passively focuses on the latency optimizations in the network, it ignores the appeals of the instructions executed in cores to the communication infrastructure. CoCom's on/off-chip control framework, considering the key chain between the computation and communication, is adaptable to fine-grained tuning locally, as well as coarse-grained control globally, contingent to the workload runtime power demand, so it yields graceful performance results.

In terms of power consumption, we evaluate dynamic and static power of the computation and communication hardware resources. The result is shown in Figure 8. We use stacked bars to demonstrate the overall power consumed, constituted by the item power, of a particular scheme. CoCom has `25.7%` average lower power reduction compared to `No_PM`, and `22.7%` and `7.6%` compared to the `Core-only` and `BoostNoC`. The power state of cores could be possibly be upgraded in modes like "Co-sprinting" or "Co-relay" etc. for performance maintenance as well as be degraded in modes like "Co-jogging" for an abundant power reduction. Off-chip CoCom probes the overall state and drives the chip into low-power mode as a whole, which all guarantee the effective power reductions. `BoostNoC`, though has potentials in optimizing communication power, is a unilateral approach. By anchoring the core power states (potentially limiting the chip power reduction headroom), power efficiency is only decided by the network conditions, despite that this singular power control may be either beneficial or malignant to the instruction execution in cores.

B. *CoCom Dynamics*

As can be expected, different configurations account for various concrete power control operations in CoCom, so in this set of experiment, we aim to explore the design space that how the correlation boundary settings could impact chip power efficiency, by considering the two configurable parameters, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. In specific, we fixed one parameter and scale the other one within its range, as shown in Figure 9. At each <$\boldsymbol{\alpha}$ $\boldsymbol{\beta}$> combination evaluated, we record the percentage of various power modes that off-chip CoCom issued to the chip, as well as the power efficiency under this combination when the workload mix has finished execution. The power efficiency results are normalized to the peak value. $\boldsymbol{\alpha}$ is the left boundary to distinguish "not correlated" and "inverse correlated". In Figure 9(a), when $\boldsymbol{\alpha}$ is fixed and $\boldsymbol{\beta}$ varies from `0.1~0.8`, power efficiency does not always increase but reaches its peak at `0.45` on X-axis. As for the ratio of power modes, `Co-sprinting` and `Co-jogging` diminishes continually, while `Co-scatter` increases sharply to the maximum when $\boldsymbol{\beta}$ equals to `0.8`. This is because when $\boldsymbol{\beta}$ increases, the range of "not correlated" is enlarged but "positive correlated" is shrunken, so in most cases, on-chip

[2] Weighted speedup is calculated by the following formula:

$$WS_{scheme} = \sum_{i=0}^{N} \frac{IPS^{i}_{scheme}}{IPS^{i}_{alone}}$$

We measure "Instruction per Second (*IPS*)" of the employed scheme (i.e. `CoCom`, `BoostNoC`) for each application when executed alone as well as in workload mixes, based on which weighted speedup is then calculated.

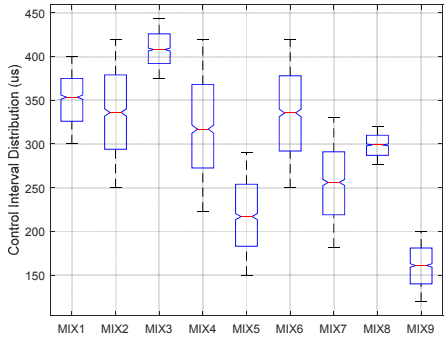


Figure 10 On-chip CoCom control interval distribution, demonstrated using boxplot. This plot is used to quantify the overhead of V/F state transition caused by each on-chip agent.

CoCom determines that the computation and communication is relatively independent with each other and power management gradually degenerates to unilateral control paradigm. Power efficiency hence suffers from the large scaling of $\boldsymbol{\beta}$. Similar phenomenon also happens when $\boldsymbol{\beta}$ is fixed with $\boldsymbol{\alpha}$ scaling in Figure 9(b), but this time, power efficiency peaks at $\boldsymbol{\alpha}$ equals to -0.3. This experiment concludes that $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ settings have significant impact on chip power efficiency, and that also explains why we chose 0.45 and -0.3 in the previous evaluations of this section.

C. *Overhead Analysis*

Intuitively, power state transition could make the processor core halt execution until a new V/F state is finally attained. Frequent power state transition is harmful to the overall performance and power efficiency. Within CoCom, this overhead mainly comes from the on-chip CoCom, and to explore its impact, we evaluate the distribution of the power *control intervals* of each workload mix. A control interval is defined as the time slot between two consecutive V/F state transitions triggered by the correlation changes beyond the preset thresholds. As shown in Figure 10, we use 'boxplot' to statistically represent the control interval distribution. For MIX9, the one that has the smallest median value, the data varies between 140us ~ 170us, while one MAC operation can be accomplished within 2~3 cycles in the distributive hardware agent. If 1 cycle is regarded as 1ns (1GHz), 100 MACs would cost 300ns at most if we trace 100 core/router performance counter values. So the overhead of state transition would be around 0.2% of the total control interval, tiny enough for on-chip power control. Off-chip CoCom only does analysis at the chip level and commands each on-chip CoCom for the final power control actuation, so its control interval is even larger. The whole hardware architecture mainly includes minimal register files and integer ALUs, so the overall chip power reduction will not be overshadowed by the power consumption itself as shown in Figure 8.

V. Conclusion

In this paper, we propose a collaborative power management approach, aiming at optimizing the chip-level power efficiency by exploring the correlations between two major hardware resources in the many-core chip: computation cores and the NoC fabric. The proposed CoCom blends the benefits of on-chip distributed control (on-chip CoCom) and off-chip global coordination (off-chip CoCom), while at the same time guarantee the power control responsiveness and low cost. We also evaluate the many-core power and performance compared with the baselines, as well as the design space exploration of the CoCom specific parameters. We hope our work could provide a promising perspective in optimizing chip-level power efficiency in future many-core processors.

VI. Acknowledgement

This work is supported in part by the National Natural Science Foundation of China (NSFC) under grant No. (61532017, 61432017, 61522406, 61572470, 61521092, 61602442), and in part by the Youth Innovation Promotion Association, CAS under grant No.Y404441000. Corresponding authors are Hang Lu and Xiaowei Li.

References

[1] M. Shafique, A. Ivanov, B. Vogel, and J. Henkel, "Scalable Power Management for On-Chip Systems with Malleable Applications," *IEEE Transactions on Computers*, vol. 65, pp. 3398-3412, 2016.

[2] X. Wang, B. Zhao, T. Mak, M. Yang, Y. Jiang, M. Daneshtalab, *et al.*, "Adaptive power allocation for many-core systems inspired from multiagent auction model," in *2014 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2014, pp. 1-4.

[3] S. Iqtedar, O. Hasan, M. Shafique, and J. Henkel, "Formal probabilistic analysis of distributed dynamic thermal management," in *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2015, pp. 1221-1224.

[4] R. Das, S. Narayanasamy, S. K. Satpathy, and R. G. Dreslinski, "Catnap: energy proportional multiple network-on-chip," presented at the Proceedings of the 40th Annual International Symposium on Computer Architecture, Tel-Aviv, Israel, 2013.

[5] H. Lu, G. Yan, Y. Han, Y. Wang, and X. Li, "ShuttleNoC: Boosting on-chip communication efficiency by enabling localized power adaptation," in *Design Automation Conference (ASP-DAC), 2015 20th Asia and South Pacific*, 2015, pp. 142-147.

[6] A. Rezaei, D. Zhao, M. Daneshtalab, and H. Wu, "Shift sprinting: Fine-grained temperature-aware NoC-based MCSoC architecture in dark silicon age," in *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2016, pp. 1-6.

[7] C. Rajamanikkam, J. Rajesh, K. Chakraborty, and S. Roy, "BoostNoC: Power efficient network-on-chip architecture for near threshold computing," in *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2016, pp. 1-8.

[8] A. K. Mishra, O. Mutlu, and C. R. Das, "A heterogeneous multiple network-on-chip design: An application-aware approach," in *Design Automation Conference (DAC), 2013 50th ACM / EDAC / IEEE*, 2013, pp. 1-10.

[9] J. Zhan, Y. Xie, and G. Sun, "NoC-sprinting: Interconnect for fine-grained sprinting in the dark silicon era," in *Design Automation Conference (DAC), 2014 51st ACM/EDAC/IEEE*, 2014, pp. 1-6.

[10] B. Fu and J. Kim, "Footprint: Regulating Routing Adaptiveness in Networks-on-Chip," presented at the Proceedings of the 44th Annual International Symposium on Computer Architecture, Toronto, ON, Canada, 2017.

[11] A. Sharifi, A. K. Mishra, S. Srikantaiah, M. Kandemir, and C. R. Das, "PEPON: performance-aware hierarchical power budgeting for NoC based multicores," presented at the Proceedings of the 21st international conference on Parallel architectures and compilation techniques, Minneapolis, Minnesota, USA, 2012.

[12] A. K. Mishra, N. Vijaykrishnan, and C. R. Das, "A case for heterogeneous on-chip interconnects for CMPs," in *Computer Architecture (ISCA), 2011 38th Annual International Symposium on*, 2011, pp. 389-399.

[13] J. Howard, "A 48-core IA-32 processor with on-die message-passing and DVFS in 45nm CMOS," in *Solid State Circuits Conference (A-SSCC), 2010 IEEE Asian*, 2010, pp. 1-4.

[14] S. Bell, B. Edwards, J. Amann, R. Conlin, K. Joyce, V. Leung, *et al.*, "TILE64 - Processor: A 64-Core SoC with Mesh Interconnect," in *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International*, 2008, pp. 88-598.

[15] A. K. Mishra, R. Das, S. Eachempati, R. Iyer, N. Vijaykrishnan, and C. R. Das, "A case for dynamic frequency tuning in on-chip networks," in *2009 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2009, pp. 292-303.

[16] J. E. Miller, H. Kasture, G. Kurian, C. Gruenwald, N. Beckmann, C. Celio, *et al.*, "Graphite: A distributed parallel simulator for multicores," in *High Performance Computer Architecture (HPCA), 2010 IEEE 16th International Symposium on*, 2010, pp. 1-12.

[17] S. Eyerman and L. Eeckhout, "System-Level Performance Metrics for Multiprogram Workloads," *IEEE Micro*, vol. 28, pp. 42-53, 2008.