DISSERTATION

DESIGN-TIME AND RUN-TIME FRAMEWORKS FOR MULTI-OBJECTIVE OPTIMIZATION OF 2D AND 3D NOC-BASED MULTICORE COMPUTING SYSTEMS

Submitted by

Nishit Kapadia

Department of Electrical and Computer Engineering

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Spring 2016

Doctoral Committee:

Advisor: Sudeep Pasricha

Anura Jayasumana
Michelle Strout
H. J. Siegel

ABSTRACT

DESIGN-TIME AND RUN-TIME FRAMEWORKS FOR MULTI-OBJECTIVE OPTIMIZATION OF 2D AND 3D NOC-BASED MULTICORE COMPUTING SYSTEMS

As a result of semiconductor technology scaling persisting over the last five decades, chip designers are today faced with the task of managing over a billion on-chip transistors. With feature sizes of no more than a few tens of nanometers in contemporary technologies, several undesirable phenomena at such nanoscale geometries have significantly complicated System-on-Chip (SoC) design. These phenomena include: (i) an increased influence of process variations that has introduced considerable unpredictability in circuit-behavior; (ii) a lowering of the critical charge of logic- and memory-cells that has given rise to elevated levels of soft-errors; (iii) a steep rise in power-densities due to higher transistor-densities, that has introduced the problem of dark-silicon, where a significant portion of the chip is required to be shut down at any given time; (iv) circuit aging that has increased significantly because of higher severity of aging factors such as electromigration and bias temperature instability (BTI) in circuits fabricated in advanced technology nodes; and (v) high voltage drops in the power delivery network (PDN) that have worsened due to the shrinking widths of on-chip interconnects. Additionally, even though the design complexity has risen exponentially, the time-to-market window for design companies has not changed markedly. Despite the numerous daunting challenges faced by the semiconductor design community, each new generation of SoCs are expected to meet higher and higher performance demands. Therefore, there is an urgent need for holistic automated system-level design tools that produce feasible and optimized design solutions efficiently while satisfying application and platform constraints.

As a lot more transistors become available to designers with every new technology node, we are witnessing a trend of increasing number of processing cores on the semiconductor die. With tens to hundreds of cores being integrated on emerging multicore SoCs, network-on-chip (NoC) based communication architectures have been found to be more suitable compared to the traditional bus-based communication architectures. Also, the recently evolved paradigm of 3D stacking of ICs has opened up new avenues for extracting higher performance from future systems by stacking multiple layers of cores and memory. In this thesis, we propose design-time optimization frameworks for synthesis of 2D and 3D NoC-based multicore SoCs. We present novel algorithms and heuristics for application-mapping, voltage-island partitioning, and NoC routing path allocation to optimize metrics such as communication and computation power and energy, chip-cooling power, voltage-drops in the PDN, design-yield, and energy-delay-squared product (ED^{2}P), while satisfying temperature, PDN, and performance constraints. In addition, to address the critical need for system-level solutions that can simultaneously and adaptively manage the constraints imposed by dark silicon, process variations, soft-error reliability, and lifetime reliability, we propose run-time frameworks for OS-level adaptations based on the circuit-level characteristics of multicore SoCs. Experimental results show that the techniques proposed in this thesis produce design solutions that provide much better overall optimality while considering multiple optimization metrics pertinent to modern semiconductor design.

ACKNOWLEDGEMENTS

I would first like to express my deep gratitude for the unyielding support I have received from my PhD advisor, Dr. Sudeep Pasricha. As a doctoral student at Colorado State University, I have always enjoyed complete academic freedom to pursue my research interests at the same time I have been fortunate to receive excellent research guidance from my advisor. His can-do attitude and enthusiasm for research has been a true inspiration for me. Also, various classes taught by Dr. Pasricha on embedded system-design and computer architecture have contributed immensely to the overall quality of my research. I cannot stress enough that this dissertation would not be possible without Dr. Pasricha's support.

I would like to thank my PhD committee members: Dr. Anura Jayasumana, Dr. H. J. Siegel, and Dr. Michelle Strout for the direction and encouragement I have received from them. I would also like to thank my colleagues at MECS Lab, especially Yong Zou, Dalton Young, and Xiang Yi, for their help and support during my PhD. I would like to acknowledge the numerous intellectually-stimulating and fun conversations I have had with my other colleagues: Sai Vineel, Ishan, Srinivas, Tejasi, Saideep, Yaswanth, Vipin, Sai Kiran, Daniel, Bhavesh, Ryan, Pramit, Viney, Brad, Soohyun, Shirish, Shoumik, and Prof. Kangchul Kim at MECS-lab research group-meetings and elsewhere. In my experience, our weekly group-meetings in addition to the ISTEC guest-lecture series serve as fertile grounds for great research ideas.

In addition, I would like to acknowledge the contribution of Dr. Ramin Roosta and my Master's adviser Dr. Nagi El Naga at California State University, Northridge for arousing in me the academic passion for electronic design, which became the root cause for me to pursue doctoral studies. I am thankful to Dr. Ali Ebnenasir at Michigan Technological University for his inspirational class on Advanced Algorithms that has also helped me reach my research goals. Besides, I am grateful for the broad academic experience I received while working at SVNIT Surat as a VLSI faculty.

On a personal note, I would like thank my Kapadia family and my in-laws for their constant love and support. I am also grateful to Mani Jayaraman and Dr. Gopal Jayaraman for guiding and inspiring me to persist with my graduate studies. Last but by no means the least, I would like to thank my wife, Radhika, for being a constant companion and support through my PhD journey. Her pride for my research has helped me keep my spirits high.

DEDICATION

To my mother Devyani Kapadia and my late father Ashok Hasmukhlal Kapadia

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF RESEARCH PUBLICATIONS

Book Chapters:

1. **N. Kapadia, S. Pasricha, "Robust Application Scheduling with Adaptive Parallelism in Dark-Silicon Constrained Multicore Systems", book chapter in the Springer book entitled 'The Dark Side of Silicon (Computing in the Dark Silicon Era)'.**

Journal Publications:

1. **N. Kapadia, S. Pasricha, "A Framework for Low Power Synthesis of Interconnection Networks-on-Chip with Multiple Voltage Islands",** ***Integration, the VLSI Journal,*** **vol. 45, no. 3, pp. 271-281, June 2012.**

2. **N. Kapadia, S. Pasricha, "A System-Level Co-Synthesis Framework for Power Delivery and On-chip Data Networks in Application-Specific 3D ICs",** ***IEEE Transactions on Very Large Scale Integration Systems (TVLSI),*** **vol. PP, no. 99, Feb. 2015.**

3. **N. Kapadia, S. Pasricha, "A Runtime Framework for Robust Application Scheduling with Adaptive Parallelism in the Dark-Silicon Era",** ***under review*** **at** ***IEEE Transactions on Very Large Scale Integration Systems (TVLSI).***

Conference Publications:

1. **N. Kapadia, S. Pasricha, "VISION: A Framework for Voltage Island Aware Synthesis of Interconnection Networks-on-Chip",** ***ACM Great Lakes Symposium on VLSI (GLSVLSI),*** **pp. 31-36, May 2011.**

2. **N. Kapadia, S. Pasricha, "A Power Delivery Network Aware Framework for Synthesis of 3D Networks-on-Chip with Multiple Voltage Islands",** ***IEEE International Conference on VLSI Design (VLSID),*** **pp. 262-267, Jan. 2012.**

3. **N. Kapadia, S. Pasricha, "VERVE: A Framework for Variation-Aware Energy Efficient Synthesis of NoC-based MPSoCs with Voltage islands",** ***IEEE International Symposium on Quality Electronic Design (ISQED),*** **pp. 603-610, Mar. 2013.**

4. **N. Kapadia, S. Pasricha, "A Co-Synthesis Methodology for Power Delivery and Data Interconnection Networks in 3D ICs",** ***IEEE International Symposium on Quality Electronic Design (ISQED),*** **pp. 73-79, Mar. 2013.**

5. **N. Kapadia, S. Pasricha, "Process Variation Aware Synthesis of Application-Specific MPSoCs to Maximize Yield",** ***IEEE International Conference on VLSI Design (VLSID),*** **pp. 270-275, Jan. 2014.**

6. **N. Kapadia, S. Pasricha, "PRATHAM: A Power Delivery-Aware and Thermal-Aware Mapping Framework for Parallel Embedded Applications on 3D MPSoCs",** ***IEEE International Conference on Computer Design (ICCD),*** **pp. 525-528, Oct. 2014.**

7. **N. Kapadia, S. Pasricha, "VARSHA: Variation and Reliability-Aware Application Scheduling with Adaptive Parallelism in the Dark-Silicon Era",** ***Design Automation and Test in Europe (DATE) conference,*** **pp. 1060-1065, Mar. 2015.**

8. **N. Kapadia, S. Pasricha, "Process-Variation and Soft-Error Reliability-Aware Workload Mapping with Adaptive Parallelism in SoCs",** ***Semiconductor Research Corporation (SRC)-TECHCON Conference,*** **Sept. 2015.**

9. **N. Kapadia, Y. Raparti, S. Pasricha, "ARTEMIS: An Aging-Aware Run-Time Resource Management Framework for 3D CMPs in the Dark-Silicon Era",** ***IEEE/ACM NOCS,*** **2015.**

10. **S. Maiti, N. Kapadia, S. Pasricha, "Process Variation Aware Dynamic Power Management in Multicore Systems with Extended Range Voltage/Frequency Scaling",** ***IEEE International Midwest Symposium on Circuits & Systems (MWSCAS),*** **Aug. 2015.**

1 Introduction

This section outlines recent trends and challenges of semiconductor chip design, and explains the need for consideration of a holistic view of the various aspects of the modern design flow. It also gives a general overview of the contributions made in this thesis.

1.1 Design Automation in Modern Semiconductor Chip-Design

Over the past few decades, improvements in semiconductor fabrication technology have enabled integrating billions of transistors on a chip. Management of on-chip resources has thus become enormously complex. Additionally, with shrinking transistor feature sizes numerous second-order effects have been introduced in circuit operation that make the design effort even more complex. Such a herculean design task cannot be undertaken without the use of intelligent software tools.

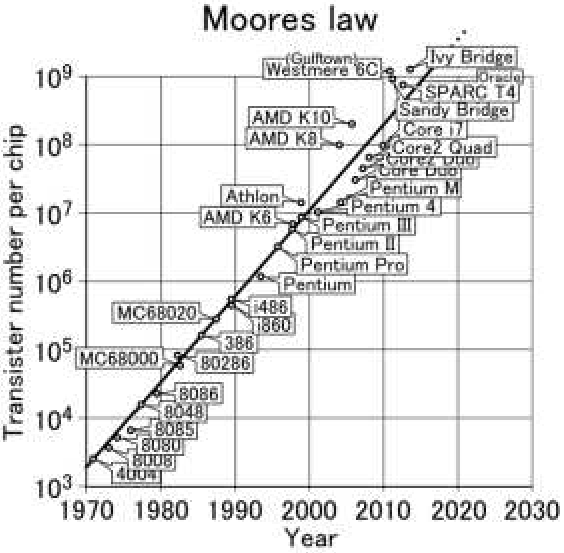


Figure 1: Number of transistors on a chip has been steadily increasing according to Moore's law [1]

As the semiconductor technology generation advances (e.g. from 28 nm to 22 nm) roughly every two years (Moore's law), the chip-design companies are required to meet strict time-to-market demands to remain commercially viable. Therefore, simulation software tools with efficient algorithms that produce optimized design solutions quickly remain critically important. Moreover, there remains an ever increasing demand for low-power and high-performance designs. Portable devices such as smart phones are expected to execute more performance-intensive applications with even higher energy-efficiency; desktop and server-systems are expected to run more and more complex workloads. Thus, contemporary designs require efficient utilization of the system-power-budgets while optimizing performance.

Due to the above mentioned reasons, efficient automation tools that produce optimized design solutions at various levels of abstraction (such as system-level, logic-level, and circuit-level) are indispensable in contemporary semiconductor design-flows.

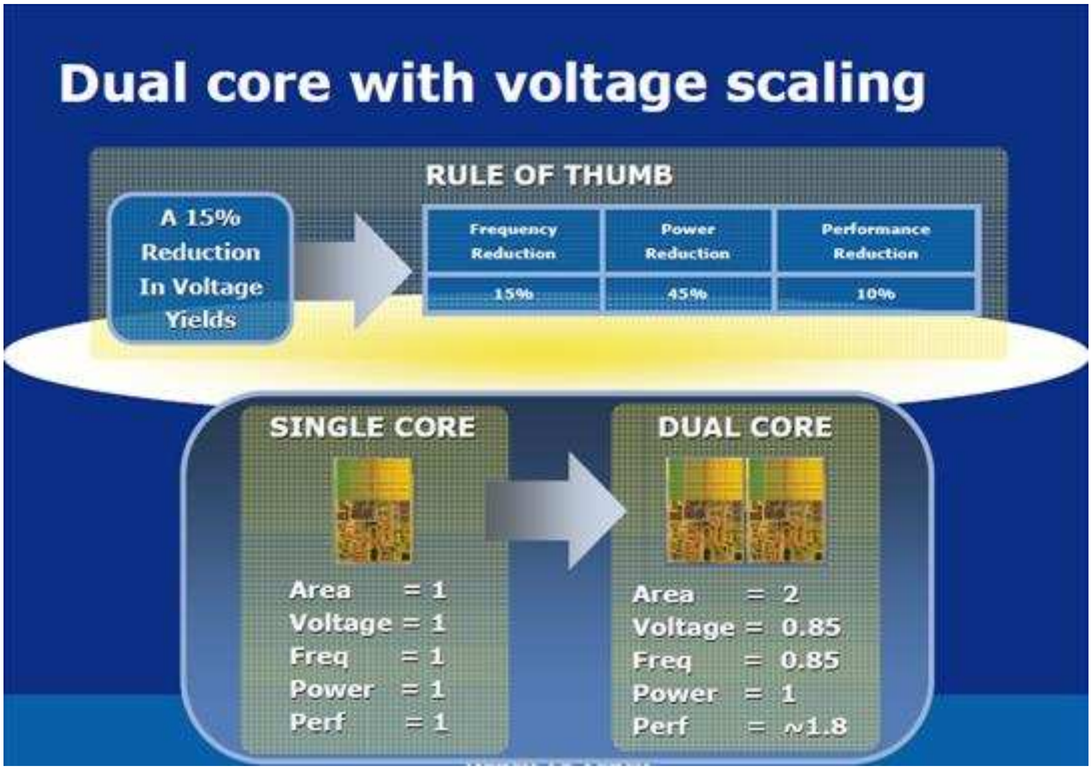
1.2 Multi-core Systems with Networks on Chip

As many more transistors are available to computer architects for every new technology node, we are witnessing a trend of increasing number of cores on the semiconductor die. The semiconductor design community seems to have reached a consensus that the additional real-estate on the chip may be better utilized by employing numerous simpler cores instead of building more complex uniprocessors. Some of the main reasons for such a paradigm shift are as follows:

1) Clock speeds have essentially stopped increasing due to much higher power densities.
2) Attempting to extract higher instruction-level parallelism (ILP) from uniprocessors has been found to result in lesser performance per watt.
3) Due to scaling in wire widths along with transistor gate-lengths in scaled technologies, interconnect delays have a significantly higher impact on system-performance. Therefore, delays introduced by long metal interconnects running across the chip can no longer be tolerated.

4) Multi-core architectures are better suited to exploit the inherent parallelism in today's multi-threaded workloads.

5) Multiple applications are required to be executed simultaneously on modern day microprocessors.



Dual core with voltage scaling

RULE OF THUMB

A 15% Reduction In Voltage Yields

Frequency Reduction	Power Reduction	Performance Reduction
15%	45%	10%

SINGLE CORE

Area = 1
Voltage = 1
Freq = 1
Power = 1
Perf = 1

DUAL CORE

Area = 2
Voltage = 0.85
Freq = 0.85
Power = 1
Perf = ~1.8

Figure 2: For power-limited design regimes, increasing processor-cores on a chip could potentially boost performance significantly with similar power dissipation [2]

In such a scenario, the traditional bus-based fabric for inter-core communication could easily be rendered untenable (due to long communication latencies) with tens to hundreds of on-chip cores. The networks-on-chip (NoC) based communication architecture has been extensively researched, both in the academia and the industry, and has been found to be more suitable for multi-core environments. Mesh-based NoC fabrics have even been employed in commercial processor chips ([3], [4], [5]).

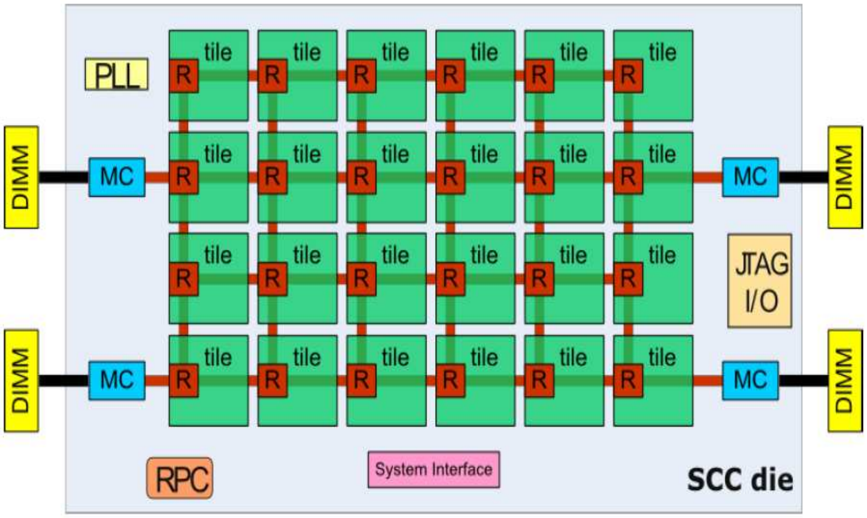


Figure 3: Intel's SCC block diagram [3] with mesh topology of the NoC fabric

1.3 Advantages of 3D Integrated Circuits

3D integration of ICs is enabled by stacking multiple silicon dies and interconnecting them vertically using through-silicon-vias (TSVs) forming a single package with a much smaller foot-print compared to the conventional 2D ICs. By enabling interconnections in the third dimension, 3D ICs can boost performance and reduce power dissipation with much shorter interconnect lengths on average. As, keeping designs at newer technology nodes commercially viable is becoming more and more difficult due to the exploding design-complexity and reducing yield, 3D ICs hold great promise of extending (or even excelling) the Moore's law for a much longer time in the future. Although designing 3D ICs could be quite complex because of challenges such as: *(i)* excessive runtime chip-temperatures due to smaller heat dissipation area and *(ii)* higher voltage drops (IR drops) in the power delivery network (PDN) of the chip due to higher current densities.

1.4 Dominant Physical Phenomena in Scaled Technologies

1.4.1 Process Variations

With feature sizes far below the wavelength of light, variations in fabrication processes are becoming more common and can lead to unpredictable behavior in modern semiconductor designs. One of the primary effects of such process variations is the deviation of the circuit threshold voltage (V_T) from its nominal value. A rise in the value of V_T increases circuit delay (which decreases maximum operating frequency) and decreases the leakage power; on the other hand, a reduction in V_T decreases circuit delay and increases leakage power. Thus, process variations introduce unpredictability in power and performance profiles of manufactured chips. The design costs associated with margining required to overcome this unpredictability can be prohibitively high.

1.4.2 Operating Temperature

With the scaling down of the dimensions of the on-chip transistors, power densities have risen quite significantly. Higher power densities have in turn given rise to excessive operating temperatures. Higher temperatures have several negative effects on circuit operation. First, circuit-delay increases (performance worsens) with temperature. Second, it is also commonly known that leakage power has a super-linear relationship with temperature and that temperature in turn depends on the power profile of the chip. In contemporary electronic systems, this cyclic inter-dependence could severely affect performance. Third, if operating temperatures are left unchecked the power-temperature dependence could lead to thermal-runaway potentially causing burn-out and rendering the chip unusable. Fourth, higher temperatures expedite aging in transistors due to circuit-aging phenomena such as bias-temperature instability (BTI), hot-carrier injection (HCI), and time-dependent dielectric breakdown (TDDB). In addition, higher operating temperatures induce higher rate of electro-migration in on-chip metal wires (logic-interconnects and power-grid wires). Such aging phenomena progressively slow down circuit-operation over the lifetime of the chip. High rates of degradation could significantly limit the service life of the semiconductor chip.

The already severe thermal problems in 2D ICs are exacerbated in 3D ICs due to even higher power-densities and higher thermal-resistivity to the heat-sink. In fact, traditional air-cooled heat-sinks have been shown to be inadequate in removing the heat produced from 3D ICs.

1.4.3 Voltage Drops in the Power Delivery Network

Designing a robust power delivery network (PDN) is critical to the overall performance of today's computer systems. The PDN is required to deliver a stable power supply across the chip that is within a desired voltage range and tolerate large variations in load currents [100]. Unfortunately, with increasing on-chip device density and decreasing voltage levels, the supply currents have risen; however, the scaling of PDN impedance has not kept up with this trend. The resulting worsening of IR drops in the PDN has led to a reduction in the quality of voltage supply and negatively impacted performance, because circuit delay in modern technologies has been shown to have a strong nonlinear relationship to the supply voltage drop [6]. This problem is even more severe in 3D ICs as the current in the PDN can be as many times more as the number of device layers compared with a 2D IC. Moreover, the number of I/O pins on an *n*-layered 3D design is about *n* times smaller than its 2D counterpart, thus exacerbating the problem of a degraded voltage supply in 3D designs [7].

1.4.4 Soft Errors

With increasing transistor miniaturization, circuit densities have drastically increased, and the critical charge, which is the minimum charge capable of a bit-flip in a memory- or a logic-cell, has significantly decreased [8]-[10]. This phenomenon has caused newer process technologies to be more susceptible to transient-faults due to the effects of radiation, e.g., alpha-particle and neutron strikes. The rate of such transient-faults at run-time has thus been increasing with technology scaling [11]. Thus, design challenges due to such soft-errors in integrated circuits are projected to become even more severe. As the soft-error rate (SER) increases with decreasing voltage and frequency values [11], the dynamic voltage and frequency scaling (DVFS) scheme in modern systems needs to be cognizant of the reliability constraints as well.

1.4.5 Aging

Aging in circuits due to phenomena such as BTI, HCI, and TDBB causes gate-delay degradation that could potentially lead to untimely failure of the chip. The principal effect of such a circuit-aging mechanism is to increase circuit-threshold voltage (V_T), which results in higher circuit-delay. Additionally, electromigration (EM) in metal wires on the chip leads to increase in interconnect resistance over time. This phenomenon is most dominant in power delivery network (PDN) wires that carry larger unidirectional currents compared to signal wires [12], [13]. The increased resistance of the power-grid results in higher IR-drops in the PDN, which causes further circuit slowdown due to degradation of supply voltage [6]. These adverse effects of EM are expected to be particularly severe in 3D ICs that possess limited number of power-pins and higher current densities. Also, with process technology scaling, this problem is exacerbated due to the reduction in cross-sections of metal wires, which causes further increase in PDN current-densities [12].

1.4.6 Dark-silicon

The slowdown of power scaling with technology scaling, due to leakage and reliability concerns [14], [15], has led to a rise in chip power-densities, giving rise to the dark-silicon phenomenon – a significant fraction of the chip needs to be shut-down at any given time to satisfy the chip power-budget. With the extent of dark-silicon increasing every technology-generation (30%-50% for 22nm) [16], [17], designs are becoming increasingly power-limited rather than area-limited.

1.5 Challenges for Sytem-level Design-time Frameworks

Multiprocessor-system-on-chips (MPSoCs) are a class of multi-processors that are designed to run embedded applications. The computation and communication profiles of such applications are generally known at design-time. Therefore, MPSoCs are best suited for design-time synthesis. In this section, we would discuss various modern-day design-challenges that need to be addressed during system-level MPSoC synthesis.

As the number of cores integrated on multi-core systems continues to increase into the hundreds [4], [5], the complexity of network-on-chip (NoC) fabrics required to interconnect communicating cores on a chip has also increased. NoCs have been shown to dissipate significant power (e.g., ~30% of chip power in Intel's 80-core teraflop [4] and ~40% of chip power in MIT RAW [5] chips). As a result, reducing both computation and communication power has become a high priority for chip designers. To mitigate problems of high power dissipation, voltage islands (*VIs*) are commonly used. In the presence of *VIs*, not only can cores run on optimal voltages with minimal overhead to reduce chip power dissipation, but the on-chip interconnection network can also exploit *VIs* to reduce communication power. However, *VIs* complicate the problem of NoC synthesis, requiring designers to revisit the problems of *VI*-partitioning, core-to-die mapping, and routing path allocation so that both computation and communication power can be minimized.

Moreover, for each new configuration of computation core and communication mapping on an MPSoC, the corresponding inter-core communication patterns, 3D on-chip thermal profile, as well as IR-drop distribution in the PDN can vary significantly. Therefore, due to the interdependence between these design objectives, performing core-to-tile mapping for optimization of communication-profiles exclusively could possibly make it extremely difficult to meet PDN constraints (such as max-IR-drop) as well as thermal constraints. Consequently, it is our contention that a holistic design approach that simultaneously considers optimization of NoC, PDN, and thermal objectives is essential for effective MPSoC-synthesis.

Additionally, due to significant within die (WID) variations and die-to-die (D2D) variations in semiconductor processes of advanced technology nodes, the design costs associated with margining required to overcome the unpredictability can be prohibitively high. Therefore, system-level design approaches that are aware of process variations can be crucial for designing energy-efficient systems.

1.6 Challenges for System-level Run-time Frameworks

Chip Multiprocessors (CMPs) are a class of multi-processors built for general purpose computing. As the applications to be executed on CMP-platforms are not known a priori, run-time optimization frameworks that adapt to the time-varying characteristics of workloads are better suited for CMPs. In this section, we would discuss the key considerations in the design of run-time optimization frameworks for CMPs fabricated at advanced technology nodes.

With significant impact of process variations (WID + D2D) in modern technologies, fabricated chips have widely varying power-performance-profiles. Therefore, awareness of the variation-profile of each chip in a run-time optimization framework could potentially yield immense benefits in terms of power-performance trade-offs in comparison to a traditional variation-unaware approach. As a result, run-time optimization frameworks are essentially required to perform dynamic voltage scaling (DVS) and application mapping with the knowledge of the variation-profile of the individual CMP chip to optimally save power and maximize performance.

The traditional objective of DVS schemes has been to save power while meeting performance and/or thermal constraints. With soft-error reliability a real concern in current and future systems, DVS would need to be cognizant of the soft error reliability requirements as well.

Besides, for run-time mapping of a queue of incoming applications on a CMP, applications traditionally have a fixed degree of parallelism (DoP). Given that higher DoP values could possibly yield better performance, by using fixed application-DoPs the CMP cannot exploit the changing application-arrival rates at runtime, and thus suffer from poor system utilization. Therefore, intelligent runtime adaptations of application-DoPs are essential in contemporary CMPs to maximize utilization and performance of the system.

Finally, the rate of aging in transistors and metal-wires increases with the active times of individual circuit components and the supply voltages used. As chip-lifetime is now one of the key considerations at advanced technology nodes, run-time application-mapping and DVS scheme are also required to balance the lifetime requirements with performance and power requirements of the system. Moreover, under certain scenarios when the CMP is executing predominantly communication-intensive (data-intensive) applications aging in the NoC could potentially be lifetime-limiting. Here, the application mapping and routing framework would be expected to extend the service life of even the NoC fabric on the CMP.

1.7 Our Contributions in Design-time Synthesis of MPSoCs

In this thesis, we propose several design-time optimization frameworks for synthesis of MPSoCs. We present novel algorithms and heuristics for application-mapping, voltage-island partitioning, and routing path allocation to optimize metrics such as communication and computation power and energy, chip-cooling power, IR-drops in the PDN, design-yield, and energy-delay-squared product (ED^{2}P), while satisfying temperature, PDN, and performance constraints.

As our first contribution, we propose a framework for *VI*-aware synthesis of interconnection networks on chip (VISION) that combines novel algorithms and heuristics to perform *VI* partitioning, core to die mapping, and routing path allocation to minimize power dissipation while satisfying application bandwidth requirements. We propose three variants of our framework based on different core-to-tile mapping heuristics: (i) incremental swapping (VISION), (ii) Probabilistic Incremental Swapping (VISION-P), and (iii) Incremental Swapping using Branch & Bound (VISION-B).Our partitioning techniques consider not only the optimization of computation power, but communication power as well. Our mapping approach uses a distributed decision making over the whole mesh instead of a centralized approach used in previous works. Our routing path allocation algorithm integrates link-insertion and routing in contrast to previous work. Our best performing framework VISION-B produces up to 32% savings in communication power and up to 13% savings in total power compared to the best known prior work on *VI*-aware NoC synthesis.

In our second contribution on design-time synthesis, we bring to light the interdependence of optimizing communication and thermal profiles, as well as PDN IR-drop distribution on the chip. We propose a novel design-time system-level application-specific co-synthesis framework that intelligently maps computation and communication resources on a die, for a given workload. The goal is to minimize the NoC power as well as chip-cooling power and optimize the 3D PDN architecture; while meeting performance goals and satisfying thermal constraints, for a microfluidic cooling-based application-specific 3D MPSoC. Our experimental results indicate that the proposed 3D NoC-PDN cosynthesis framework provides better overall optimality with the solution quality improvement of up to 35.4% over a probabilistic metaheuristic-based co-optimization approach proposed in prior work.

Our third contribution, PRATHAM, extends our co-synthesis (second) contribution with a holistic solution evaluation methodology that accurately captures the different metric interdependences. We integrate the effects of temperature and supply voltage drops in the performance and energy modeling of the compute cores as well as the communication resources; and show that considering such awareness in the solution evaluation framework results in significantly improved design solutions. Our experimental results indicate that PRATHAM improves system-ED^{2}P by up to 43.6% over prior work.

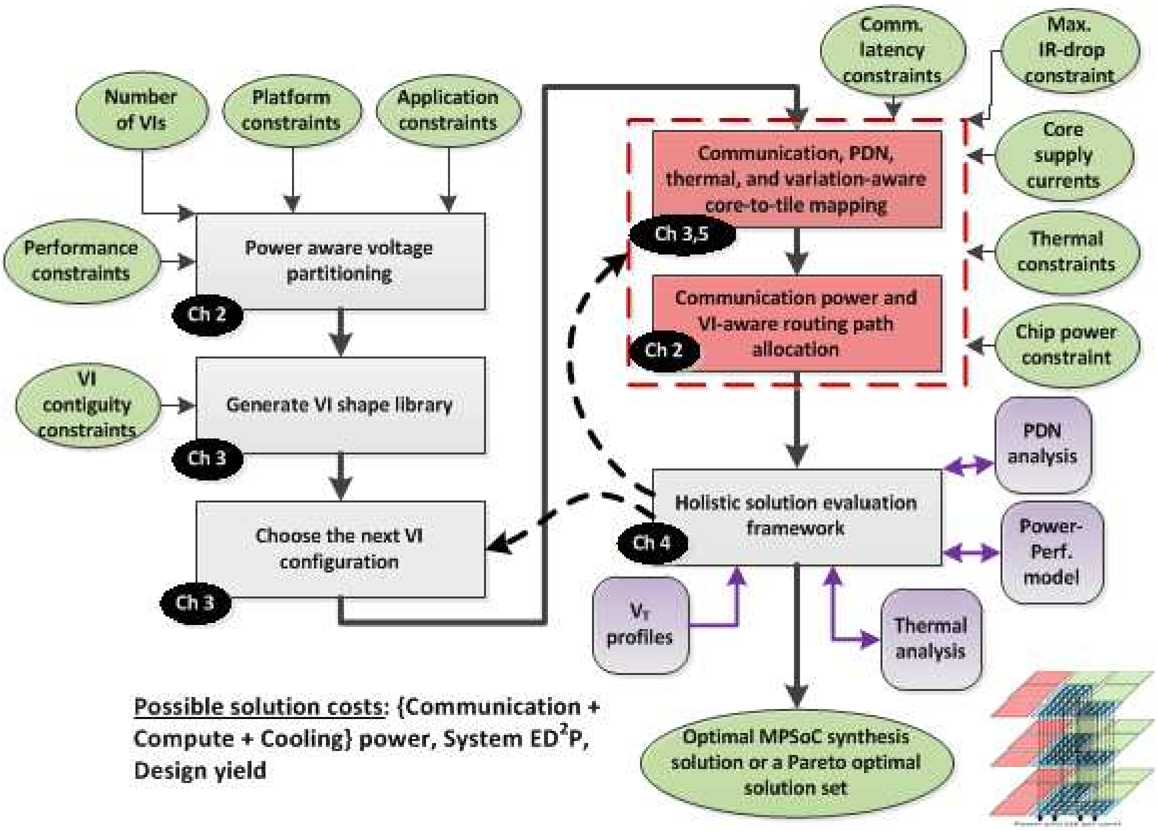

Number of VIs
Platform constraints
Application constraints
Comm. latency constraints
Max. IR-drop constraint
Performance constraints
Power aware voltage partitioning
Ch 2
VI contiguity constraints
Generate VI shape library
Ch 3
Choose the next VI configuration
Ch 3
Communication, PDN, thermal, and variation-aware core-to-tile mapping
Ch 3,5
Communication power and VI-aware routing path allocation
Ch 2
Core supply currents
Thermal constraints
Chip power constraint
PDN analysis
Holistic solution evaluation framework
Ch 4
Power-Perf. model
V_T profiles
Thermal analysis
Possible solution costs: {Communication + Compute + Cooling} power, System ED^{2}P, Design yield
Optimal MPSoC synthesis solution or a Pareto optimal solution set

Figure 4: MPSoC-synthesis framework envisioned as a combination of our design-time synthesis contributions. Small circles (in black) contain chapter-numbers that corresponding blocks are discussed in.

In traditional design flows that exhibit variation-awareness, voltage assignments are done without the core-mapping information and require resorting to pessimistic voltage assignments based on worst case V_T values. This leads to higher voltage assignments and correspondingly higher overall power dissipation. As our fourth contribution, we propose a novel process variation-aware MPSoC synthesis framework that performs simultaneous mapping and voltage assignment of cores to mitigate the adverse effects of process variations while maximizing yield. Note that as the variation-profiles of individual chips are not accurately known at design-time, we consider *N* test chips (*N* variation-profiles) that capture the effects of process variations on performance and power, to represent the variation-profiles produced at the target process technology node. Our framework incorporates novel *VI*-aware and variation-aware optimization techniques, and produces on average 2× to 3.8× improvements in power-performance yield (PPY) over other variation-aware MPSoC synthesis frameworks.

1.8 Our Contributions in Design of Run-time Adaptations for CMPs

To address the critical need for system-level solutions that can simultaneously and adaptively manage the constraints imposed by dark silicon, process variations, soft-error reliability, and lifetime reliability, in this thesis we propose two runtime frameworks for OS-level adaptations based on the circuit-level characteristics of the CMP-chip.

Our first such contribution, VARSHA, is a novel run-time application scheduling framework that employs dynamically adaptable application degrees of parallelism (DoPs) to minimize average application service times and energy, while meeting a chip-wide dark-silicon power constraint and application performance and soft-error reliability constraints, in the presence of process variations. VARSHA utilizes a novel heuristic to integrate within the application-mapping process a DVS mechanism that is constrained not just by performance but also by application-reliability requirements. Our VARSHA framework is well-suited to the emerging dark-silicon-constrained-design-regime, improving over traditional mapping approaches optimized for the area-constrained-design-regime, where it simultaneously manages all dynamically arriving applications while adapting application-DoPs to optimally utilize the system-power-slack. Our experimental results show that VARSHA produces savings of 35%-80% in application service-times, 13%-15% in energy, and avoids reliability violations unlike state of the art prior works that suffer from reliability violations in up to 11% of application instances arriving at run-time.

In our final contribution we recognize the key insight that the mechanism to enhance lifetime reliability in 3D CMPs must consider circuit-aging together with PDN-aging, as the impacts of PDN- and circuit-aging on system-performance are correlated. We propose a novel runtime framework (*ARTEMIS*) for intelligent dynamic application-mapping and voltage-scaling to simultaneously manage aging in circuits and the PDN, and enhance the performance and lifetime of 3D NoC-based CMPs. *ARTEMIS* adapts to different aging scenarios to extend the lifetime of a 3D NoC-based CMP. It encapsulates a symmetric aging enabled routing algorithm that balances the degree of aging between NoC routers and cores, thereby increasing the combined lifetime of both. Compared to a framework based on the best known prior works on aging-aware mapping techniques, *ARTEMIS* is able to service 25% more applications (on average) over the chip lifetime, which highlights its promise for emerging 3D CMPs.

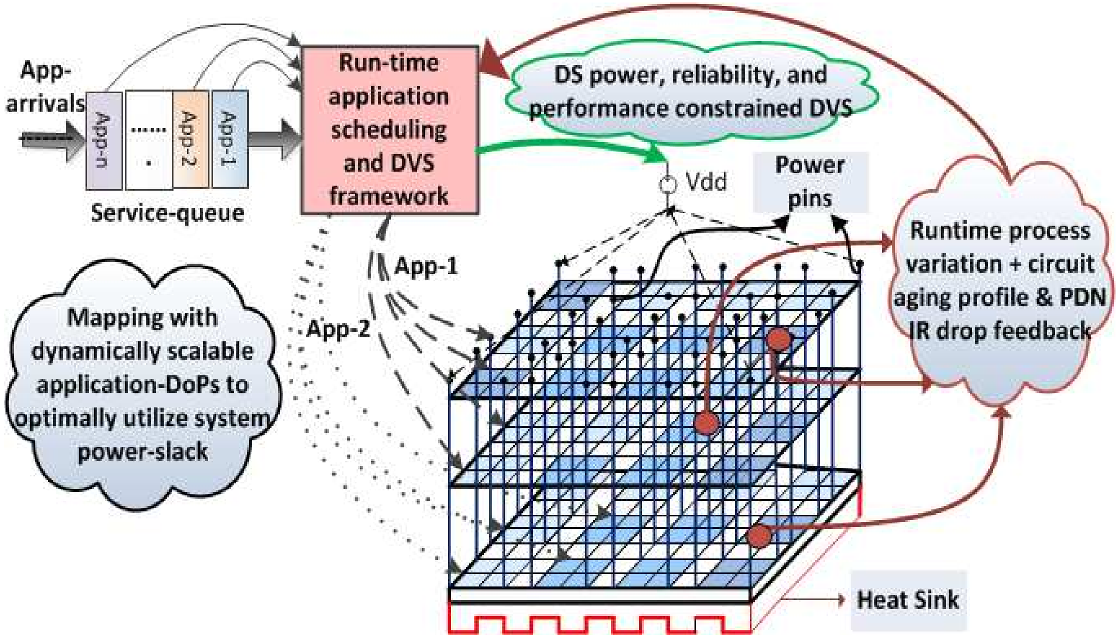

App-arrivals
App-n
......
App-2
App-1
Service-queue
Run-time application scheduling and DVS framework
DS power, reliability, and performance constrained DVS
Vdd
Power pins
Runtime process variation + circuit aging profile & PDN IR drop feedback
App-1
App-2
Mapping with dynamically scalable application-DoPs to optimally utilize system power-slack
Heat Sink

Figure 5: Overview of the application-mapping and DVS framework envisioned as a combination of our contributions in runtime CMP management

1.9 Outline

This thesis contributes in the advancement of state of the art in system-level design automation for multi-core SoCs in three main ways: *(i)* it presents several new optimization techniques for previously known system-level design problems; *(ii)* it brings to light some previously unexplored optimization search-spaces thereby discovering new opportunities for further optimization; *(iii)* it proposes efficient algorithmic techniques to produce design-solutions with better overall optimality for these expanded multi-dimensional search-spaces. The research presented in this thesis is organized as follows. Chapter 2 discusses our communication and *VI*-aware MPSoC synthesis frameworks (VISION and its extensions) that optimize communication and computation power while meeting application performance constraints. Chapter 3 presents our design-time co-synthesis framework that minimizes the 3D NoC power as well as chip-cooling power and optimizes the 3D PDN architecture, while meeting performance goals and satisfying thermal constraints, for a microfluidic cooling-based application-specific 3D MPSoC. Our enhanced co-synthesis framework, PRATHAM, which utilizes a holistic solution evaluation methodology, is discussed in chapter 4. Chapter 5 presents our variation-aware MPSoC synthesis framework, which mitigates the adverse effects of process variations thereby improving the power-performance yield. Chapter 6 discusses our variation and reliability-aware runtime application scheduling framework (VARSHA) that employs adaptive parallelism and suited to the emerging dark-silicon regime. Chapter 7 improves over the VARSHA framework by additionally considering memory traffic and detailed NoC simulation, and proposes novel mapping techniques suited for the new set assumptions. Chapter 8 discusses *ARTEMIS*, our aging-aware runtime application mapping framework for 3D NoC-based CMPs that extends the service life of the 3D chip by balancing the PDN-aging and circuit-aging as well as the aging in compute cores and associated NoC routers. Finally we present the thesis-summary and future work in chapter 9.

2 A Framework for Low Power Synthesis of Interconnection Networks-on-chip with Multiple Voltage Islands

The problem of VI-aware Network-on-Chip (NoC) design is extremely challenging, especially with the increasing core counts in today's power-hungry Chip Multiprocessors (MPSoCs). In this chapter, we propose a novel framework for automating the synthesis of regular NoCs with VIs, to satisfy application performance constraints while minimizing chip power dissipation.

2.1 Introduction

Emerging multiprocessor systems on chips (MPSoCs) today are severely constrained because of their high power dissipation in compute cores due to high levels of core integration and nanoscale CMOS process technology characteristics. High power dissipation on a chip not only reduces overall performance, but also negatively impacts reliability and cooling costs. As the number of cores integrated in MPSoCs continues to increase into the hundreds [4], [5], the complexity of network-on-chip (NoC) fabrics [18] required to interconnect communicating cores on a chip has also increased. NoCs have been shown to dissipate significant power (e.g., ~30% of chip power in Intel's 80-core teraflop [4] and ~40% of chip power in MIT RAW [5] chips). As a result, reducing both computation and communication power has become a high priority for chip designers.

Among the several circuit and (micro)-architectural techniques proposed by designers in recent years to reduce power dissipation, dynamic voltage and frequency scaling (DVFS) is widely used to reduce dynamic power [19], while clock/power gating techniques [20] are commonly used to reduce leakage power in cores. However, implementing these techniques at a per-core granularity requires separate V_{DD} and ground lines, as well as a prohibitively large number of VLCs (voltage level converters) and MCFIFO (multiple clock first-in-first-out) queue based frequency level converters [21], [22], especially as the number of cores on a die increase. The use of voltage islands (*VIs*) limits the overhead of implementing these power management schemes by combining cores into islands that use the same V_{DD} and ground lines, and thus also minimizing the number of VLCs and MCFIFOs required [23]. Different islands can then operate at different voltage levels and perform runtime optimizations independent of each other to more aggressively reduce chip power. In the presence of VIs, not only can cores run on optimal voltages with minimal overhead to reduce chip power dissipation, but the on-chip interconnection network can also exploit *VIs* to reduce communication power. However, VIs complicate the problem of NoC synthesis, requiring designers to revisit the problems of *VI* partitioning, core to die mapping, and routing path allocation so that both computation and communication power can be minimized.

In this work we address the problem of synthesizing NoCs for regular MPSoCs with *VIs* to reduce overall power dissipation. We focus on MPSoCs with homogenous and regular sized cores because of the prevalence of regular topologies in almost all recently released commercial MPSoCs [4], [5]. Our proposed framework for VI-aware Synthesis of IntercOnnection Networks on chip (VISION) combines novel algorithms and heuristics to perform *VI* partitioning, core to die mapping, and routing path allocation to minimize power dissipation while satisfying application bandwidth requirements. We propose three variants of our framework based on different core-to-tile mapping heuristics: *(i)* incremental swapping (*VISION*), *(ii)* Probabilistic Incremental Swapping (*VISION-P*), and *(iii)* Incremental Swapping using Branch & Bound (*VISION-B*). Experimental studies presented in Section 2.5 indicate that the proposed VISION framework outperforms the best known prior work [24] that focuses on the same problem of *VI*-aware regular NoC synthesis. In particular, our framework generates solutions that have lower network traffic by up to 62%, lower communication power by up to 32%, and lower total power dissipation by up to 13% compared to solutions generated by the approach in [24].

2.2 Related Work

Many researchers [25]-[29] have proposed custom topologies for NoC architectures that improve overall performance at the cost of sacrificing the regularity of mesh-based topological structures. Although these custom architectures are expected to achieve better latency and area utilization, their design process is significantly more complex and faces several challenges, such as greater crosstalk and uncertainty in link delays due to irregular interconnect structures. Thus, a conservative enough custom design may actually offset the advantages of better performance [30]; especially for medium to large sized NoC architectures (in terms of total number of cores).

Lackey et al. [23] give an overview of the preliminary considerations for power and performance for NoC designs with *VIs*. The problem of NoC synthesis on regular structures with multiple supply VIs has since been addressed in several works [24], [31]-[36]. Ghosh et al. [32] solve a linear programming formulation of the same problem, but without considering the effects of multiple frequencies and the required MCFIFO-based converters. Leung et al. [31] propose a genetic algorithm to combine the different steps in a *VI*-aware NoC synthesis flow. Ogras et al. [33] perform the *VI* partitioning and static voltage-frequency assignment on a pre-mapped NoC to optimize communication and energy consumption while meeting task deadline constraints. By performing voltage-partitioning before core mapping, [24] demonstrated improvements over prior work (e.g., [33]) to achieve less power overhead by reducing total number of MCFIFOs and VLCs needed for inter-island communication.

The problem of mapping cores on to regular NoCs has been handled differently by the works above. Heuristics implementing incremental mapping on NoCs with *VIs* that have been proposed to date [24], [35] map the cores in order of their communication bandwidths. As the order of mapping is fixed, every new mapping decision is based on optimizing some communication metric with just the previously placed cores, thereby restricting the search space to a possible local minimum and failing to capture a holistic view for optimizing the communication over the entire network. In this work, in addition to proposing novel techniques for voltage partitioning and routing path allocation, we present three new mapping techniques that use the foundational concept of incremental swaps with a distributed decision making process, as opposed to a central one used in prior work.

2.3 Problem Formulation

We are given the following inputs to our problem:

A core graph $G\ (V, E)$; with the set of N vertices $\{V_1, V_2, V_3, \ldots, V_N\}$ representing the homogeneous cores on which tasks have already been mapped and the set of M edges $\{e_1, e_2, e_3, \ldots, e_M\}$ that represent communication dependencies between cores;

(i) A regular mesh-based NoC with T tiles such that $T \geq N$, and $T = (d^2)$, where d is the dimension of the mesh, and each tile consists of a compute core, a NoC router, and a network interface (*NI*) between them;

(ii) A set of minimum operating voltage levels required for meeting task performance requirements on each of the N cores $\{min\_v(V_1), min\_v(V_2), \ldots., min\_v(V_N)\}$;

(iii) A set of n candidate voltage levels $\{v_1, v_2, \ldots, v_n\}$ and the corresponding candidate frequency values $\{f_1, f_2, \ldots, f_n\}$ that the cores can take; and

(iv) The maximum allowable number of voltage islands on the die, Ω; where $\Omega \leq n$.

Objective: Given the above inputs, the goal of our synthesis framework is to obtain a core to die mapping and synthesize a regular 2D mesh NoC architecture for a specific application, such that all application performance requirements are satisfied, while minimizing the total power dissipation in the compute cores and the communication resources (routers, links, VLCs, MCFIFOs).

Definition 1: Voltage Island Integrity of any island is said to be respected when every core within it has at least one neighbor (out of the maximum of four neighbors possible in a mesh) of the same voltage level as itself. Mathematically:

$$\forall\ v(t_{xy}) = v_i\!: \exists\{v(t_{x-1,y}) = v_i \,|\, v(t_{x,y-1}) = v_i \,|\, v(t_{x+1,y}) = v_i \,|\, v(t_{x,y+1}) = v_i\}$$

where t_{xy} represents the tile at the respective co-ordinates of the mesh, with integral values in the range *1* to *d*.

Definition 2: Pre-Routing Traffic is an early estimation of the total traffic assuming that all routing paths are minimal. It is equal to the sum of products of Manhattan distances and bandwidths of all individual communication flows. Mathematically, this can be expressed as: $\sum_j MD_j * BW_j$; where *j* is a uni-directional communication flow between any two cores on the die.

Definition 3: Link Tension is defined as the product of the communication bandwidth and post-mapping Manhattan distance for any edge on the core graph *G (V, E)*. Thus, for two cores adjacent to each other (i.e., Manhattan distance = 1), the tension on the link connecting them is equal to the bandwidth of communication between them.

Definition 4: Total Traffic is the overall bandwidth a routed-network is required to suport. It is the sum total of bandwidths of all communication flows over their respective actual (minimal or non-minimal) path-lengths, expressed as: $\sum_j Path\text{-}length_j * BW_j$; where *j* is a uni-directional data communication flow between any two cores.

Not unlike previous works [24], [33], [34], our proposed NoC synthesis framework is subdivided into three major steps: voltage partitioning, core-to-tile mapping, and routing path allocation. Where our framework differs from previous works is in the algorithms used and more rigorous implementation assumptions that we consider. As the computation power (dissipated in compute cores) still dominates the communication power (dissipated in routers, links, MCFIFOs, VLCs) for MPSoCs with up to several tens of cores, optimizing the former should take precedence over the latter for minimal total power. Therefore, in this work (unlike in [33]), we perform the voltage partitioning step before core-to-tile mapping. The problems addressed in the three major steps of our framework are summarized below:

A. Communication-power aware voltage partitioning

The objective in this first step is to assign voltages to cores in the core graph to minimize the computation power and the communication power at the same time, such that the constraints on the minimum operating voltage levels of all *N* cores *{min_v(V$_1$), min_v(V$_2$), …., min_v(V$_N$)}* and the maximum number of *VI*s allowed (*Ω*), are satisfied.

B. Core to tile mapping

The objective in this second step is to perform a one to one mapping of voltage assigned cores onto NoC tiles such that the integrity of each *VI* is respected (*Definition 1*), hikes in compute power of cores are avoided, and total estimated traffic (pre-routing traffic in *Definition 2*) is minimized, to optimize overall communication power dissipation.

C. Routing Path Allocation:

The objective in this third and final step is to allocate routing paths for all application-specific communication flows in the NoC, such that the communication power overhead associated with MCFIFOs and VLCs needed for inter-island communication is minimized.

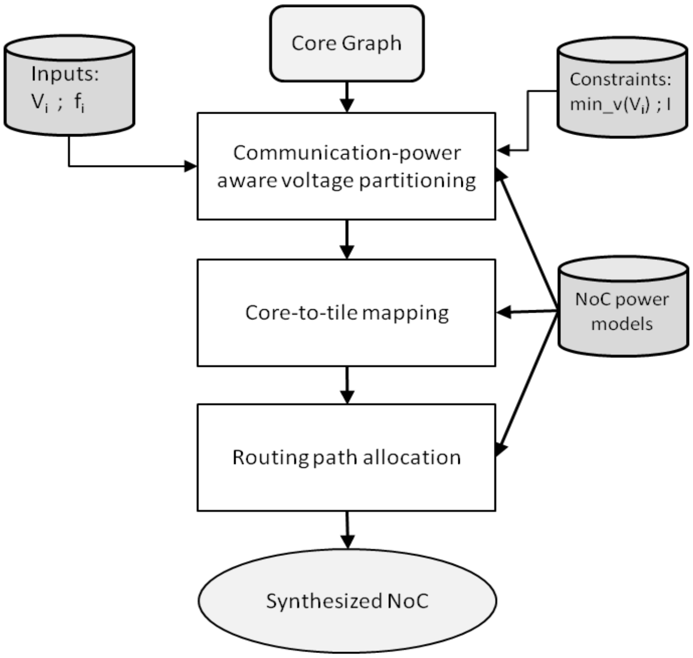


Figure 6: NoC synthesis design flow

2.4 NOC Synthesis Flow

In this section, we present details of our framework for the synthesis of NoCs with *VI*s. Figure 6 shows the high level flow of our synthesis framework that improves upon the state of the art *VI*-aware NoC synthesis frameworks, such as that proposed in [24]. The *partitioning* stage supports trade-offs between computation power and the estimated communication power, while assigning voltages to cores and partitioning them into islands. The *mapping* stage attempts to meet physical proximity constraints for cores in islands and performs swapping heuristics to reduce pre-routing traffic. Finally, the routing path allocation stage integrates link insertion and routing path selection for communication flows, to minimize traffic and reduce communication latency.

In the following subsections (Sections 2.4.1 – 2.4.3), we present a detailed explanation of the algorithms used in the three major stages of our synthesis framework.

2.4.1 Communication Power-aware Voltage Partitioning

Voltage partitioning attempts to assign core voltages, in order to meet constraints imposed by application performance and the Ω levels of allowable voltages. The voltage partitioning approach in [24] performs an exhaustive search on all n voltage level candidates using a maximum of Ω levels of allowable voltages (for Ω *VIs*). Thus, ${}^{n}C_{\Omega}$ combinations of different voltages are generated. Then, each core is assigned the least voltage level out of the available voltages for each of the ${}^{n}C_{\Omega}$ combinations, to ensure that application performance constraints are satisfied. Out of the ${}^{n}C_{\Omega}$ combinations of voltage assignments, the least expensive assignment in terms of power is then finally chosen. The shortcoming of this approach is that the voltage partitioning only optimizes the computation power, ignoring the effects of the communication power on the resultant total power. We propose a new *repartitioning* stage that can be appended to the previous approach to overcome the above mentioned drawback.

After the voltage partitioning of [24] is completed based on optimal computation power, in our repartitioning stage we allow certain cores to be moved to a neighboring voltage island with the next higher allowable voltage level, in order to reduce the communication power associated with the core.

Such a move is meant to reduce communication traffic overhead by merging the core with an island it communicates heavily with; and is referred to as an 'allowable move' as long as the increase in resultant power is within a certain threshold ψ. Note that the allowable move is restricted to only the voltage island with an immediately higher voltage level because in low to medium sized NoCs (where the maximum distance between any two cores is restricted by the dimension of the mesh), the computation power dominates the communication power. Therefore, the reduction in communication power by performing a move that would increase the voltage of any core to a much higher level would seldom justify the penalty incurred on the computation power.

With repartitioning, cores that heavily communicate with each other end up residing in the same island and thus are likelier to be mapped in each other's vicinity. The probable reduction in the communication power as well as the total traffic is primarily due to lesser inter-island communication that leads to: *(i)* shorter average communication path lengths, and *(ii)* a reduction in the number of MCFIFOs and VLCs, with lesser inter-island communication. The cost function for any candidate core move from island *i* to island *j* can be expressed as:

$$C(i,j) = (P_{Rj} - P_{Ri}) + (O_j - O_i) + \mathscr{H}$$

where P_{Rj} and P_{Ri} are the router powers if the core were in island *j* or island *i* respectively, O_j and O_i are the power overhead of MCFIFOs and VLCs associated with the respective islands, and $\mathscr{H}$is the compute power hike for the core because of the move. The mapping threshold ψ can be expressed as:

$$\psi = (comm_j/(comm_i))\times(1/(v_j - v_i))*\omega$$

where $comm_j$ and $comm_i$ represent the total communication bandwidths of the core when it is mapped to island *j* and island *i*, respectively, v_j and v_i are the corresponding island voltages, and ω is designer specified parameter that regulates the aggressiveness of the moves. We use a conservative value of ω = 0.0001 to balance computation and communication power, and avoid dramatically increasing total power during a move. Ultimately, the mapping threshold ψ quantifies the projected reduction in post-mapping traffic (and thus the communication power) resulting from the move under consideration. Algorithm 2.1 below summarizes the repartitioning approach.

```
Algorithm 2.1. Repartitioning
input: voltage-partitioned core graph G(V, E)

1: Compute the cost C and threshold ψ for all allowable moves
2: Make the move with the lowest cost (below the corresponding move
   threshold ψ) and lock this core in the new island.
3: Re-compute a new set of allowable moves for all the unlocked cores
   and the associated costs and thresholds.
4: Repeat steps (2) and (3) until all costs of the remaining  unlocked cores
   are above their respective thresholds

output : voltage-repartitioned core graph G'(V, E)
```

2.4.2 Core to Tile Mapping

In the next stage, we map cores to tiles on the die. Our proposed approach consists of two major steps: initial mapping generation, and incremental swapping. The following subsections describe the initial mapping generation, and three variants of incremental swapping.

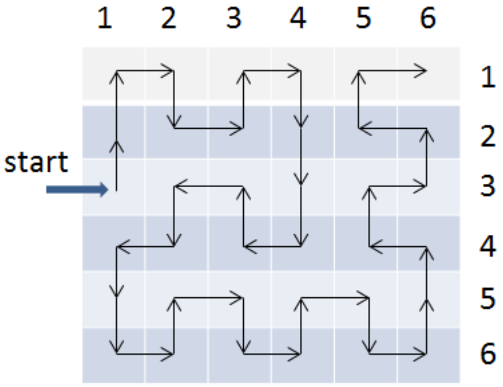


Figure 7: Sequence of tile co-ordinates for the initial mapping on a 36 core (6x6) regular mesh NoC

2.4.2.1 Initial Mapping Generation

In this step, we generate an initial core to tile mapping by traversing an Inter-Island Communication Graph *IICG*(V_{isl}, E_{isl}), where the vertices constitute the entire islands and edges constitute aggregate communication bandwidth between the respective islands. A breadth-first search (BFS) starting with each of the Ω islands as the root node, would produce Ω distinct sequences of islands, each of length Ω. The order of islands in each sequence is based on decreasing communication bandwidths with the island selected as the root node. Subsequently, the cores are mapped onto the tiles of the NoC in order of the island $sequence_j$ to generate a $mapping_j$.

We follow a pre-defined sequence of tile co-ordinates for the mapping process as shown in Figure 7. Such an ordering of the core to tile mapping grows in both the *x* and *y* directions of the mesh in a symmetrical way, thereby keeping the Manhattan distances between the currently placed core and recently placed cores shorter; as compared to, say, growing the mapping in just the x or y directions (i.e., row-wise or column-wise mapping). Furthermore, our ordering ensures that the placement of the current core is adjacent to the previously placed core in order to guarantee *VI* integrity. For all Ω mapping configurations generated, we perform Incremental Swapping (sections 2.4.2.2 - 2.4.2.4 describe three variants of the swapping step), Routing Path Allocation (section 2.4.3) and then compute the resultant total power of the network. Algorithm 2.2 below summarizes the key steps in the initial mapping generation procedure. It generates Ω initial mappings to constitute a *mapping-set*.

Algorithm 2.2. Initial Mapping Generation

```
input: core graph G'(V, E)

Create an inter-island communication graph IICG($V_{isl}$, $E_{isl}$)
for j = 1 to $\Omega$ do
  With $V_j$ as the root node; perform BFS on IICG to get a $sequence_j$ of islands
  Map the cores within each island in the order of the $sequence_j$ to obtain $mapping_j$
end for

output : mapping-set
```

2.4.2.2 Incremental Swapping

The incremental swapping step is intended to reduce link tension (*Definition 3*) in the NoC. Cores connected by communication links with greater tension have a higher force of attraction between them. The tensions in the mesh network force cores with high communication bandwidths to come closer during the incremental swapping step. After the initial mapping, all communicating cores have some tensions associated with them, in one or more directions. The incremental swap algorithm attempts to decrease the Manhattan distance of the core with the highest tension in the entire mesh by swapping it with another core to reduce the tension. The swaps are allowed to occur between neighbors either in the x-direction, y-direction or diagonally on the mesh structure. A swap is considered valid if it can pass three checks:

1. *Total tension decrease check:* the swap is valid only if it will result in a decreased total tension (total pre-routing traffic).
2. *VI integrity check:* the swap is valid only if it will not disintegrate any islands.
3. *Tabu-direction check:* the swap is valid only if the direction of the move is not in the Tabu list.

If a swap does not pass the checks, the swap is aborted, and the core is marked-off to restrict it from swapping for the next *d* (dimension of the mesh) number of swap attempts (iterations). If a swap is valid, then the x, y tensions and x, y co-ordinates of the cores are updated after the swap, and the complementary move direction for the core is added to a Tabu list (e.g., -y if the core moved in the +y direction) so that the core will never move back in the direction it came from. If a diagonal swap takes place, both the x and y directions are added to the Tabu list. Also, the total pre-routing traffic is updated after every swap. The incremental swapping continues until *d* consecutive aborts have been encountered. This state of the NoC, where valid swaps which reduce total NoC tension are no longer readily available, is defined as equilibrium.

Algorithm 2.3 below describes the key steps in the incremental swap algorithm, which eventually decreases the Manhattan distances of high bandwidth communication flows in the mesh NoC. The input is the mapping set for which each of the *Ω* constituent mappings is processed, to create a final mapping-set. Note that cores are never swapped back in the direction of their previous locations; this gives us a theoretical upper bound on the total number of swaps that the *N* cores in the mesh can undergo: $O(N*2*(N^{1/2}-1))$.

Algorithm 2.3. Incremental Swapping

```
input: core graph G'(V, E) and mapping-set

for each mapping-solution $\in$ mapping-set do
  do until $d$ consecutive aborts are encountered
    Choose the core with the maximum total tension
    Choose the direction with the most tension
    Perform the three pre-swap checks
    If the swap is invalid, consider other directions
    If no directions valid, abort;  mark-off the core for next $d$ iterations
    If some direction is found to be valid, perform the swap
end for

output : final-mapping-set
```

2.4.2.3 Probabilistic Incremental Swapping

On an initial mapping solution, instead of just considering the core under most tension for the next swap (as discussed in section 2.4.2.2); probabilistic incremental swapping selects a core for the next swap out of up to *p* cores which are under most tensions. The *p* cores under highest cumulative tensions in the NoC having at least one 'valid swap' (i.e., satisfying the 3 pre-swap checks) in the direction of their net x-y tension are considered for the next swap. Here, *p* is a parameterizable value defined as an upper bound on the number of cores in the mesh that become candidates for the next swap. The pseudo code (Algorithm 2.4) is given below, which is run on each Initial Mapping (IM).

Algorithm 2.4. Probabilistic Incremental Swapping

```
input: core graph G'(V, E) and mapping-set
for each mapping-solution (IM) $\in$ mapping-set do
  do $K$ times
    do until $d$ consecutive aborts are encountered
      Find $p$ cores under most cumulative tensions
      For the $p$ cores, find $q$ cores which have at least one valid swap
      If no valid swaps found ($q$=0), abort; mark-off all the $p$ cores for
      next $d$ iterations and goto next iteration
      Probabilistically, choose one of the $q$ cores for the next swap
      Perform a valid swap in the direction of most tension
end for

output : K final mapping candidate solutions for each IM
```

Each of the candidate cores are assigned swap probabilities for the next swap, proportional to the value of cumulative tensions they are under. The swap probabilities are computed as follows. Let q be the number of candidate cores for the next swap, where $q \leq p$. Let $T_1, T_2, \ldots, T_q$ be the cumulative or net tensions associated with the q swapping candidates and $T = T_1 + T_2 + \ldots + T_q$ be the sum of q tensions. Then, the swap probabilities for the q candidate cores $P_1, P_2, \ldots, P_q$ become:

$$P=T_1/T,$$

$$P=T_2/T,$$

$$:$$

$$P=T_q/T.$$

For every swap, the q probabilities are computed and one of the candidates is selected based on the respective probabilities for the next swap; a valid swap is performed in the direction (x, y or diagonal) of most tension. The procedure of probabilistic incremental swapping continues until equilibrium is reached. This procedure is performed K times (as shown in Figure 8; K is a designer specified parameter) for each initial mapping solution to produce $K*\Omega$ final mapping solutions, out of which the one corresponding to least communication power (obtained after performing routing path allocation on each of the final mapping solutions) is selected as the final NoC synthesis solution for the particular value of p.

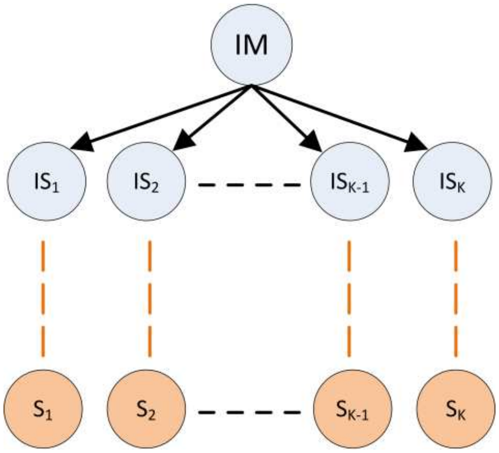


Figure 8: Probabilistic incremental swapping technique for a single Initial mapping (IM). IS_x are the intermediate mapping solutions and S_1-S_K are the K final mapping solutions

2.4.2.4 Incremental Swapping with Branch & Bound (BB)

The probabilistic swapping approach (section 2.4.2.3) selects a single core (out of *p* probable candidates) for every swap and performs this procedure iteratively until equilibrium is reached. Therefore, it is a sequential process where just one mapping solution (intermediate or final solution) exists at any given time during execution. We also propose a branch and bound based incremental swapping approach where multiple mapping solutions co-exist during execution. In the BB approach, up to *n* mapping solutions are branched out from an initial or an intermediate mapping solution in the search tree (as shown in Figure 9).

The BB technique combines random search (constituting of random swaps) with directed search (constituting of directed swaps) to generate multiple final mapping candidate solutions. In the 'directed search', the set of next swaps are determined by the current link-tension map of the NoC, while a set of random swaps is selected in the 'random search'. The directed swaps are geared to reduce the highest tensions in the NoC in order to reduce communication power; where a 'best swap' swaps the core under most tension (on the NoC link-tension map) in a direction of most tension. We combine the directed swaps with random swaps for effective exploration of the solution search space. Note that random swaps are performed within the same island (to satisfy *VI* integrity requirements). Also, total tension/Tabu-direction checks are not considered and Tabus for the cores participating in the swap are reset during random swaps.

Let *n* be the maximum branching degree and *K* an upper bound on the total number of final candidate solutions which is a multiple of *n*-1; $\boldsymbol{\alpha}$ be a positive fraction which governs the weight of the random component in BB, and *C* be the number of current mapping solutions. The pseudo code for the BB procedure is given in Algorithm 2.5 below, which is run on each Initial Mapping (IM).

Algorithm 2.5. Incremental Swapping using BB

input: core graph G'(V, E) and mapping-set

```
1: for each mapping-solution (IM) ∈ mapping-set do
2:   while (($C<K$) && (at least one non-leaf node exists in C)); do ∀ non-leaf nodes on the current
      BB level {
3:      Compute B, R and D
4:      Find out the D best swaps (directed search) and check their validity
5:      If one or more valid swaps found, proceed to step 8
6:      Find the best valid swap while considering all cores
7:      If no swap is valid, mark this candidate (node) as a leaf; else execute swap (branch out child)
        and delete current node; then goto next iteration
8:      Compute the R random valid swaps
9:      Execute computed random and directed swaps, branching out a new
        child for each swap and delete current node; then, goto next iteration }
10:  ∀ non-leaf nodes, run Algorithm 3
11: end for
```

output : up to K final mapping candidate solutions for each IM

At any level of a *B*-way search tree (*B* is variable representing current degree of branching) of intermediate mapping solutions, *D* best swaps and *R* random swaps are considered for each node. The branching degree for any swap, *B* is computed according to the following equation:

$$B = (n + 1) - \lceil (n - 1) * (C + 1)/K \rceil$$

With *K* as an upper bound on the total number of final candidate solutions, the branching degree proportionally decreases with increasing number of intermediate solutions. The number of directed swaps and random swaps that could be branched out are computed from the value of *B* as follows:

$$R = \lfloor \alpha * B/2 \rfloor$$

$$D = B - R$$

At each BB node, only *D* cores (with *D* highest tensions) are considered for swapping. If no valid directed swaps are available for the *D* cores under consideration, no random swaps are branched out either and a 'Best Valid Swap' (BVS) is attempted on the current solution (IS_{11} & IS_{13} in Figure 9). The BVS swaps the core under most tension (on the NoC link-tension map) for which a valid swap exists, in a valid direction of most tension. The intermediate mapping solution (IS_{21} in Figure 9) obtained from the BVS could potentially participate in the BB search once again, as one of the *D* best swaps might now be valid on the updated solution. On the other hand, when no more directed swaps are possible, i.e., no BVS is available; the intermediate solution node (*IS_{13}*) becomes a leaf node (*L-1*), signifying a final candidate solution, and is never again considered for further swaps.

When the existing number of solutions in the BB search reach the upper bound of *K*, only the best swaps are made on all the non-leaf solutions (*B* is reduced to 1) until they converge to equilibrium, i.e. Algorithm 3 (Incremental Swapping) is run on each of the non-leaf solutions (*IS_{24}*, *IS_{31-36}*). Note that, running Algorithm 3 on a solution which has no BVS available is redundant and therefore such a node is marked as a leaf (*L-1*). Alternatively, if no non-leaf solutions remain, BB terminates as no random swaps are allowed on leaf-nodes. Finally, a set of up to *K* final mapping candidates are obtained from a single initial mapping. Out of the *K*Ω* mapping candidate solutions, the one corresponding to least communication power (obtained after performing the routing path allocation on each of these solutions) is selected as the final NoC synthesis solution for the particular value of *n*.

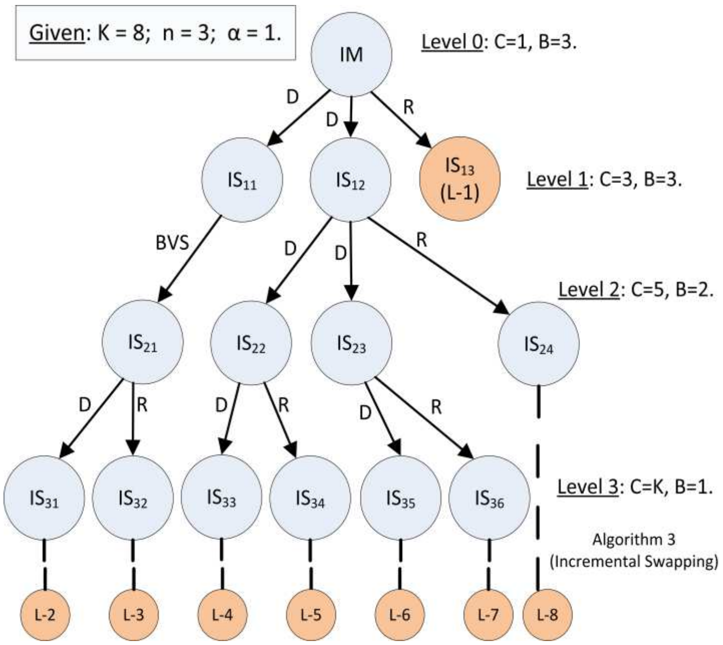


Figure 9: BB search tree for incremental swapping from a single IM. *IS_{ij}* are intermediate mapping solutions at level *i*, *L-x* are leaf nodes; R (or D) represents a random (or directed) swap branching-out a new mapping solution

2.4.3 Routing Path Allocation

The mapped NoC obtained after the previous stage consists of multiple *VIs* that not only run on different voltage levels, but also different frequencies. Therefore, whenever an inter-island link is inserted, voltage level converters (VLCs) and frequency level converter resources (MCFIFOs) are required in the corresponding routers. Whenever a low voltage core transmits to a higher voltage core, a VLC is needed on the outgoing port of the source router. Also, for any inter-island link, an MCFIFO is needed for the higher frequency/voltage core as the connecting link works at the lower frequency. These frequency and voltage conversion components incur an overhead in terms of power dissipation and delay. Thus, the main objective of routing is to find a path for each communication flow such that communication path lengths and the number of inter-island links are reduced.

The routing algorithm in [24] minimizes the number of inter-island links by calculating the inter-island bandwidths between different neighbor-islands and inserting a proportional number of links between them. Then, the inter-island routing is carried out by using just the inserted links. One of the drawbacks of this approach is increased traffic because of less flexible link insertion that is performed once and never changed again during the disjoint routing step, leading to long communication path-lengths. Because the link-insertion and routing steps are disjoint, communication between non-neighbor islands is also not considered during the link-insertion step, giving rise to a possibility of creating isolated islands that do not communicate with any of their neighbor-islands and therefore are unable to communicate with non-neighboring islands.

Our proposed routing algorithm integrates the link-insertion and the routing steps thereby alleviating the above mentioned drawbacks from [24]. We employ minimal routing in order to minimize total traffic. The inter-island links are inserted only when minimal paths cannot be found within the residual bandwidth capacities of the existing inter-island links. Figure 10 shows how the disjoint link-insertion and routing approach from [24] has longer path lengths because the routing is constrained to the available inter-island links (shown with the thick black arrows); whereas, with an integrated approach, path lengths would be optimal. As the inter-island link does not lie in the direct path between the source (2, 2) and the destination (4, 2) in (a); path lengths are longer when compared to (b).

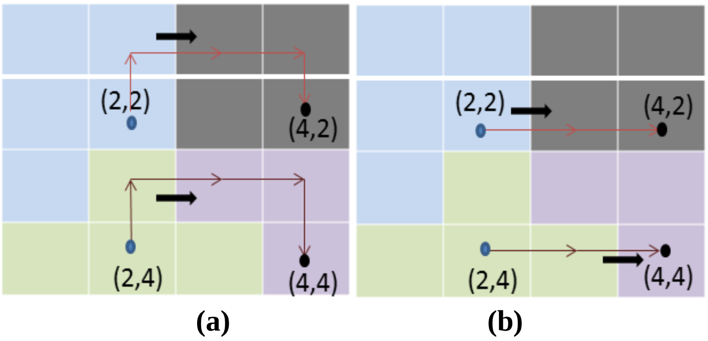


Figure 10: Cores of the same color belong to the same *VI*. Shown are two communication flows (a) with the algorithm in [24], (b) by integrating link insertion and minimal-path routing

For each communication flow, we consider all candidate minimal paths. Out of these candidate minimal paths, we choose a routing path based on the following optimization objectives (in that order):

1) Minimize total number of inter-island link insertions needed on the path
2) Minimize total number of intra-island link insertions needed on the path

The communication flows with longer minimal paths have more choices for routing and thus have a larger scope for optimization. Also, flows with smaller bandwidths, require lesser residual capacities to be accommodated within existing links. Therefore, communication flows are sorted in the increasing order of their path lengths, in decreasing order of their communication bandwidths for the same path length; and routed in that order. While routing any communication flow over a given path, links insertions are performed whenever the existing link(s) cannot support the bandwidth of the current flow or if no links are available. In summary, this routing scheme optimizes both the number of inter-island links and total traffic in the NoC, thereby resulting in significant savings in communication power. Finally, a voltage-assigned, mapped and routed NoC is obtained. Algorithm 2.6 below summarizes the routing path allocation approach.

Algorithm 2.6. Routing Path Allocation

```
input: core graph G'(V, E), final-mapping-set

Sort all communication flows  in the increasing order of path lengths
  and in decreasing order of bandwidths for the same path length
for each communication flow in sorted list do
    Out of all the candidate minimal paths, choose the set of paths that
    would require the least number of inter-island link insertions
    Out of the chosen paths, choose the one path that would result in the
    minimum number of intra-island link insertions
    Insert the necessary links and allocate the current flow bandwidth
    over the chosen routing path.
end for

output : synthesized NoC with all communications routed
```

2.5 Experiments

2.5.1 Experimental Setup

We performed several experimental studies and generated NoCs for different applications to evaluate the quality of solutions generated by our VI-aware NoC synthesis frameworks: VISION (uses the basic incremental swapping approach for tile-to-core mapping) and its variants VISION-P (uses probabilistic incremental swapping) and VISION-B (uses branch and bound based incremental swapping). All three of the proposed frameworks use the same VI-partitioning and routing path allocation schemes, which are discussed in this work.

We used the ARM11 MPCore multi-core processors [37] as the base compute cores in our experiments, which support six operating voltage levels as shown in Table 1. Correspondingly, we constrained the maximum number of *VIs* to be synthesized by our framework to six. Our experiments were conducted on three applications based on pseudo-random core graphs derived using TGFF [38], and consisting of 16 (4x4), 36 (6x6), 64 (8x8) and 100 (10x10) cores, with edge weights annotated with bandwidths representing the inter-core communication requirements. The diverse core counts allow us to ascertain the scalability and applicability of our approach to low and high complexity systems. We conservatively assume that the square of the voltage scales linearly with the frequency, as in previous works [39]. The compute core power values account for both dynamic and leakage power, and vary on average as shown in Table 1. The router and link powers for different voltages, frequencies and router complexities; and with varying communication loads are obtained from a modified version of ORION 2.0 [40]. The voltage/frequency assignment for a router is the same as the voltage/frequency of the core it is connected to in its corresponding tile. The voltage/frequency pair value for a link connecting two routers is the lower voltage/frequency pair value of the two routers connected by the link. As the VLCs and MCFIFOs required to interact between *VI*s incur a power overhead that is proportional to their voltage supply, we assume that every MCFIFO or VLC introduced in the router consumes 10% of the base router power, based on reported overheads from existing literature [41]. For deadlock avoidance, we use two virtual escape channels per router [42], with appropriate power overhead in our experiments.

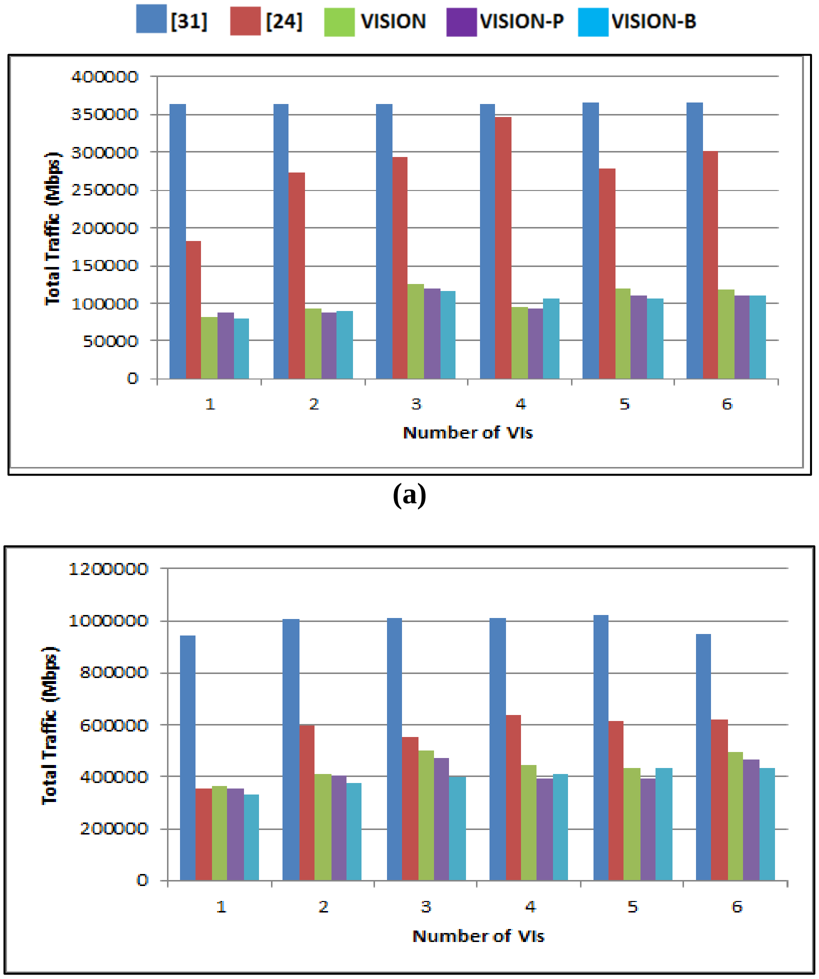
Table 1: Core voltages and the corresponding frequencies and average power values used in the experiments

Voltage (volts)	1.26	1.2	1.15	1.1	1.0	0.9
Freq. (MHz)	483	437	401	368	304	246
Power (mW)	126	101	85	72	49	32

For probabilistic incremental swapping in VISION-P and the branch and bound in VISION-B, a value of *K*=400 is used. Therefore, for same number of islands, the upper bounds on the total number of candidate mapping solutions are the same for both the procedures.

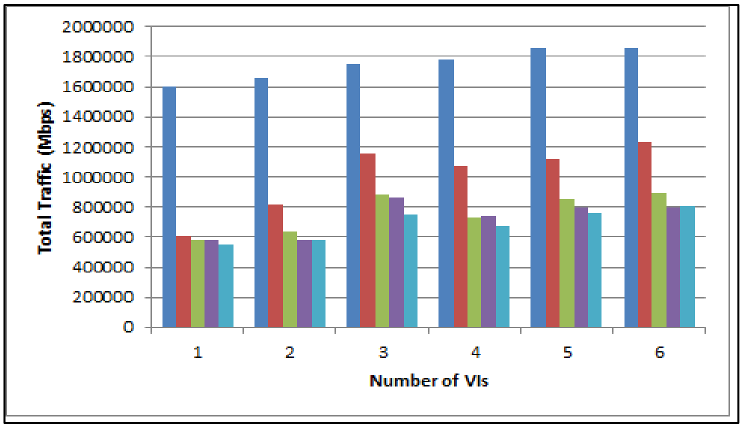
In VISION-B, our goal with the BB search tree is to decrease branching degree *B* in proportion to the increasing number of existing mapping solutions *C*. Therefore, in order for *B* (range: *n* down to 1) to span approximately uniformly over the range of *C* (1 to *K*), in our experiments *n* takes values where (*n-1*) is a factor of *K*. (*n-1*) thus takes values of 2, 4, 8, 10, 16, 20, 25, 40 and 50 (all of which are factors of 400). Also, a value of α=1 is used in the BB procedure.

In VISION-P, with increasing values of *p* during probabilistic incremental swapping, more cores are considered as potential candidates for the next swap; therefore, cores under relatively low tensions can be selected for the next swap with higher likelihood (even though with proportionately low probabilities). At the same time, with higher values of *p*, the search space is not restricted to the swaps constituting of just the cores with highest tensions. In our experiments, we have found that relatively low values of *p* (less than 20% of *N*) yield the best results in terms of communication power in the NoC. The results shown for VISION-P later in this section utilize the following empirically derived optimal values of *p*: 3, 7, 12 and 20 for NoC sizes of 16, 36, 64 and 100 respectively.

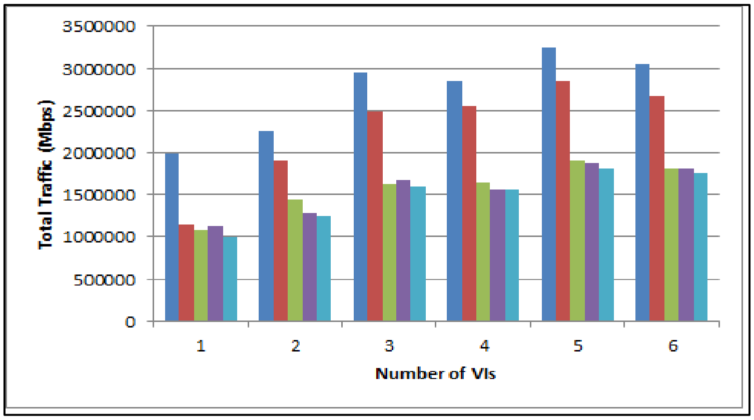


(a)

(b)



(c)



(d)

Figure 11: Total traffic in (a) 16 core mesh, (b) 36 core mesh, (c) 64 core mesh, (d) 100 core mesh

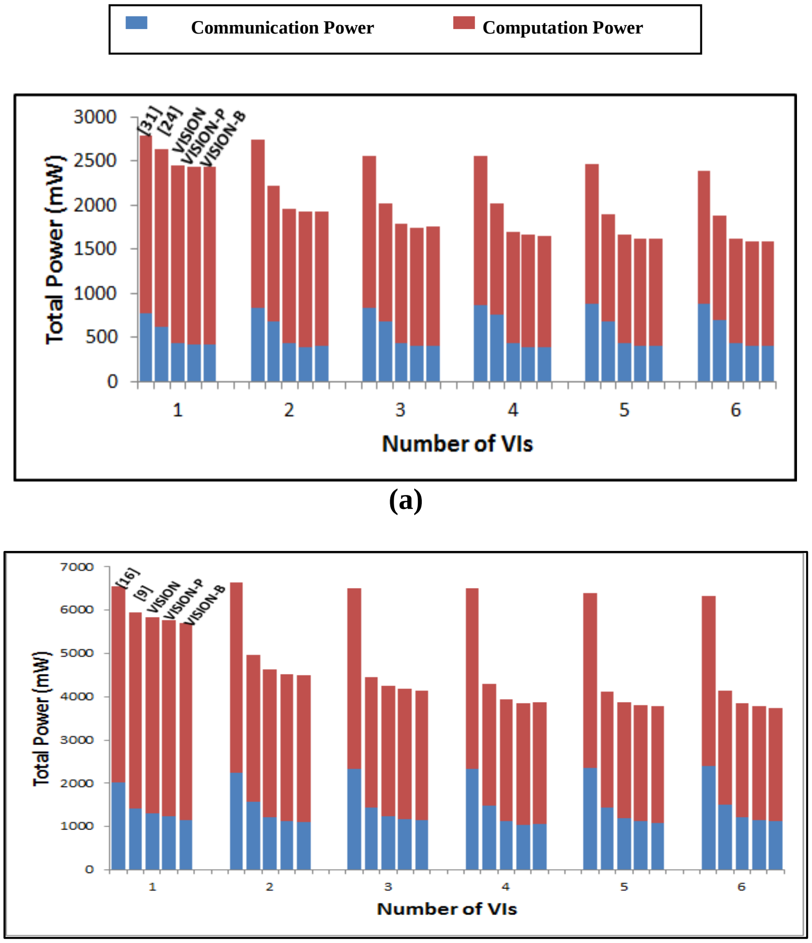
2.5.2 Results

We compare the results of our synthesis frameworks with the results obtained by using *VI*-aware NoC synthesis approaches presented in two prior works: a heuristic based recent work on *VI*-aware regular NoC synthesis [24] that claims to improve upon other previous works such as [33]; and a synthesis approach based on a genetic algorithm (GA) in [31]. We implemented the frameworks presented in [24] and [31] to the best of our understanding and used the same performance and power models for all implemented approaches to ensure a fair comparison of the algorithms.

Table 2: Total number of inter-island links for the configurations with least communication power

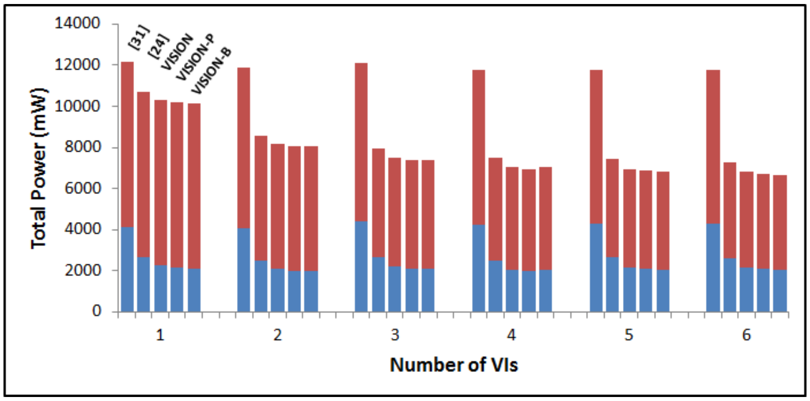
n/w sizes	Core_16					Core_36				
# of VIs	*[31]*	*[24]*	*VISION*	*VISION-P*	*VISION-B*	*[31]*	*[24]*	*VISION*	*VISION-P*	*VISION-B*
1	0	0	0	0	0	0	0	0	0	0
2	16	12	9	6	6	22	18	19	10	11
3	20	20	12	12	12	37	33	25	22	30
4	23	35	16	14	12	40	33	24	28	21
5	32	33	18	17	16	42	50	41	45	30
6	25	38	22	22	22	76	64	33	34	34
n/w sizes	Core_64					Core_100				
# of VIs	*[31]*	*[24]*	*VISION*	*VISION-P*	*VISION-B*	*[31]*	*[24]*	*VISION*	*VISION-P*	*VISION-B*
1	0	0	0	0	0	0	0	0	0	0
2	25	43	36	28	34	35	46	34	52	37
3	34	48	45	32	40	68	69	93	82	80
4	33	50	44	48	40	74	90	94	77	85
5	56	67	54	52	51	90	99	118	111	103
6	58	69	61	63	54	92	112	120	110	105

Figure 11 (a)-(d) shows the total traffic that exists in the results generated (for the four applications considered) by the five approaches: GA-based [31], heuristic based [24], and 3 of our proposed frameworks: VISION, VISION-P and VISION-B. Results are generated for a range of input *Ω* values that vary the number of VIs to be generated in the solution. The x-axis shows the solutions for the different input values of *Ω*. It can be seen that our frameworks have significantly lower traffic compared to approaches proposed in prior work. In particular, VISION-B generates solutions with the lowest traffic compared to [24] (by up to 62%), as well as [31] (by up to 78%). This is because the core-to-tile mapping approach in VISION-B not only utilizes the link-tension methodology for directed search, but unlike VISION-P it also permits random swaps, which are not necessarily between neighbors. Therefore, VISION-B provides for a more systematic and efficient exploration of solution search space resulting in lower traffic. The repartitioning approach used in all three of our approaches clusters the heavily communicating cores into the same *VI*, with a constraint on the hike in the communication power. This not only results in a reduction in the number of inter-island links, but also reduces the average Manhattan distance over all communication paths; as any two cores in separate islands are more likely to be farther apart than if they are within the same island. Our incremental swap mapping always attempts to reduce the communication Manhattan distances for the heaviest communication flows; leading to much lower traffic. Lastly, our routing path allocation algorithm also produces significant reduction in total traffic by prioritizing reduced path lengths. In comparison, the mapping approaches in [24] and [31] do not consider reducing the Manhattan distances over the entire network in a holistic manner. Also, unlike our approach, [31] does not employ minimal routing to reduce the total number of inter-island links.

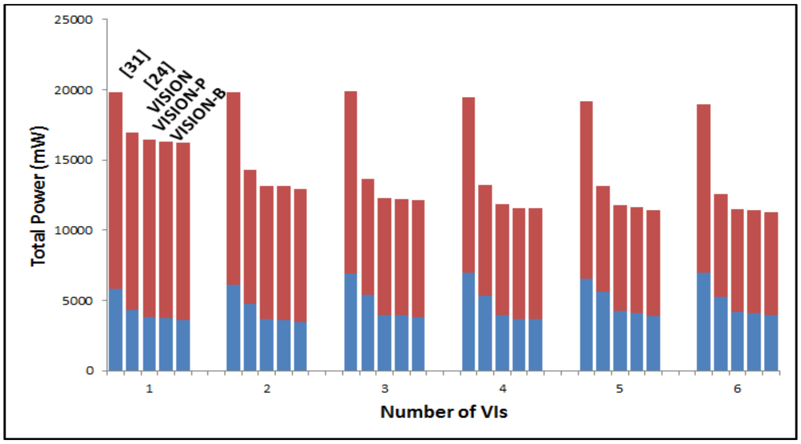


(a)

(b)



(c)



(d)

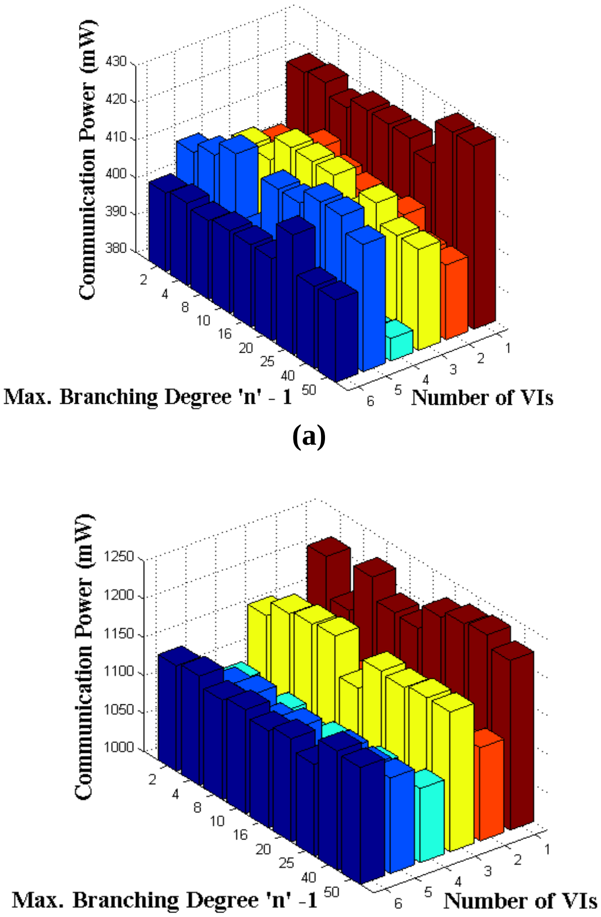
Figure 12: Total power {compute+communication} (a) 16 core mesh, (b) 36 core mesh, (c) 64 core mesh, (d) 100 core mesh

Table 2 shows the total number of inter-island links (accompanied by corresponding VLCs and MCFIFOs) generated by the five approaches. For most cases, the GA based approach [31] has the lowest number of inter-island links compared to [24] and our approaches. However, [31] accomplishes this by having many more cores being assigned higher voltages thereby reducing the variability of the voltage distribution of the cores. Thus the number of inter-island links is much lower but with highest penalties on computation power. On average, our synthesis frameworks have fewer inter-island links compared to [24], particularly because our routing path allocation considers all candidate minimal paths and prioritizes the paths with the least number of inter-island links. Also, as the core-to-tile mapping in VISION-P and

VISION-B further optimize for communication power (over VISION), they produce even fewer inter-island links.

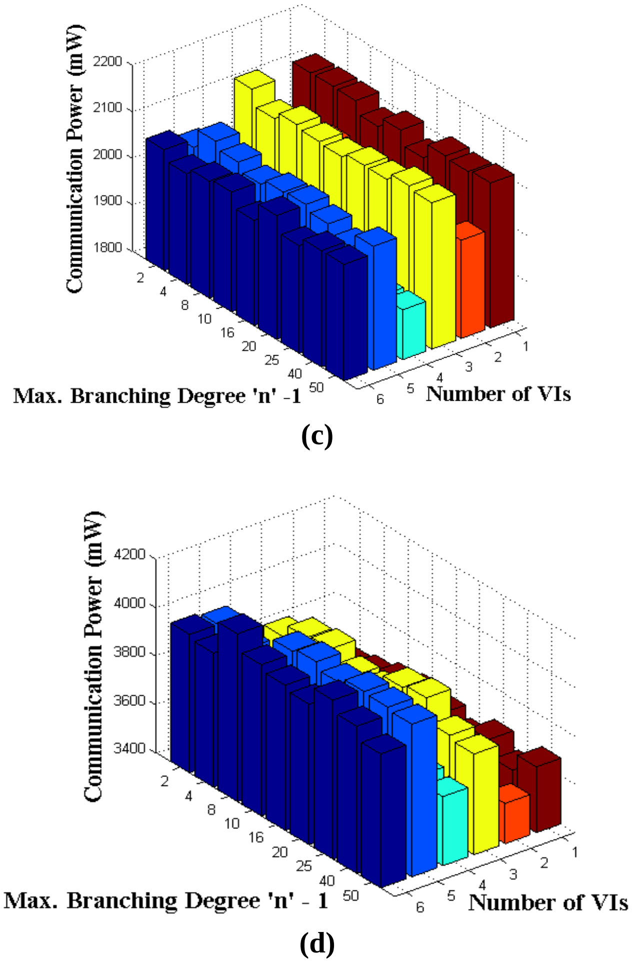
Figure 12 summarizes the total power dissipation of the solutions generated by the five approaches, for the four applications considered. A decomposition of the total power into computation and communication power is also presented. As the number of islands in the NoC increases, it can be seen that the total power decreases. This decrease in total power is due to the decreasing computation power, because with more voltage levels available, the cores in general can run at lower voltages. With reductions in total traffic as well as inter-island communication, the communication power for our proposed frameworks show quite significant improvements; for instance, our best performing VISION-B framework improves by up to 32% over [24] and by up to 54% over [31]. Even though most of these gains are obtained due to our more communication-centric partitioning and mapping techniques, for larger applications (where the total communication power is a more significant portion of the total power, due to longer communication paths), the routing technique used provides for significant improvements as well. In terms of total power dissipation, the VISION-B framework outperforms both [24] (by up to 13%) and [31] (by up to 41%). These results clearly indicate that our proposed approach can provide superior solutions when compared to the best known prior works, and is thus a promising set of tools for automated exploration and synthesis of emerging *VI* enabled homogenous multi-core NoC systems.

Finally, we present a sensitivity analysis for the value of maximum branching degree *n* in VISION-B in Figure 13 (a)-(d). The impact on total communication power dissipation of different values of *n* across different number of *VI*s is presented. Note that compute power does not change when the value of *n* is varied; therefore, we only show changes in communication power. For the branch and bound approach, the total number of final candidate solutions is (K*Ω); therefore the solution search space increases linearly with the maximum allowable number of *VI*s. With increasing number of *VI*s, routers and links can in general run at lower voltage/frequency levels, thus lowering the communication power; at the same time, additional MCFIFOs and VLCs needed for inter-island communication will add-up to the total communication power in the NoC. The additional components needed for inter-island communication is the reason why communication power does not monotonically decrease with increasing number of VIs, but rather shows an uneven trend. As the maximum degree of branching *n* increases, the BB search spawns more children out early in the search, but ends sooner with relatively less number of total swaps. Note that irrespective of the value of *n*, the incremental swapping is performed at each non-leaf node at the end of the BB procedure.



(a)

(b)



(c)

(d)

Figure 13: Sensitivity analysis of total communication power with VISION-B over a range of values of maximum branching degree *n*, across different number of VIs for (a) 16 core mesh, (b) 36 core mesh, (c) 64 core mesh, (d) 100 core mesh

Even though from Figure 13 (a)-(d), no obvious pattern of communication power in relation to varying *n* may seem to emerge; it can be observed that one of the best (if not the best) communication power values are almost certainly obtained by using values of 2, 4 or 8 for (*n-1*), even for the 100-core scenario. Therefore, if simulation time is constrained (which is the case for most real world design flows rushing to meet time-to-market deadlines), the authors recommend simulating for a few relatively small values of *n* to achieve an optimized solution.

2.6 Conclusion

In this chapter, we have proposed a set of novel synthesis frameworks (VISION, VISION-P & VISION-B) for *VI*-aware synthesis of mesh-based NoCs. Our partitioning techniques consider not only the optimization of computation power, but communication power as well. Our mapping approach uses a distributed decision making over the whole mesh; instead of a centralized approach used in previous works. Our routing path allocation algorithm integrates link-insertion and routing in contrast to previous work. Our best performing framework VISION-B produces up to 32% savings in communication power and up to 13% savings in total power; compared to the best known prior work on VI-aware NoC synthesis.

3 A System-Level Co-Synthesis Framework for Power Delivery and On-chip Data Networks in Application-specific 3D ICs

With increasing core counts ushering in power-constrained 3D multiprocessor system-on-chips (MPSoCs), optimizing communication power dissipated by the 3D network-on-chip (NoC) fabric is critical. At the same time, with increased power densities in 3D ICs, problems of IR-drops in the Power Delivery Network (PDN) as well as thermal hot spots on the 3D die are becoming very severe. Even though the PDN and NoC design goals are non-overlapping, both the optimizations are interdependent. Unfortunately, designers today seldom consider design of the PDN while designing NoCs. Moreover, for each new configuration of computation core and communication mapping on an MPSoC, the corresponding inter-core communication patterns, 3D on-chip thermal profile, as well as IR-drop distribution in the PDN can vary significantly. Based on this observation, we propose a novel design-time system-level application-specific co-synthesis framework that intelligently maps computation and communication resources on a die, for a given workload. The goal is to minimize NoC power as well as chip-cooling power and optimize the 3D PDN architecture; while meeting performance goals and satisfying thermal constraints, for a micro-fluidic cooling based application-specific 3D MPSoC.

3.1 Introduction

Designing a robust Power Delivery Network (PDN) is critical to the overall performance of today's multi-processor system-on-chips (MPSoCs). The PDN is required to deliver a stable power supply across the chip that is within a desired voltage range and tolerate large variations in load currents [100]. For the case of multiple voltage islands (*VIs*) that are used in modern MPSoC designs to minimize power dissipation, the PDN is required to supply power at different voltage levels corresponding to the *VIs* while keeping power loss to a minimum. Unfortunately, with increasing on-chip device density and decreasing voltage levels, the supply currents have risen, however the scaling of PDN impedance has not kept up with this trend [135]. The resulting worsening of IR-drops in the PDN has led to a reduction in the quality of voltage supply and negatively impacted MPSoC performance, because circuit delay in modern technologies has been shown to have a strong non-linear relationship to the supply voltage drop [6]. This problem is even more severe in 3D MPSoCs as the current in the PDN can be as many times more as the number of device layers compared to a 2D MPSoC. Moreover, the number of I/O pins on an *n*-layered 3D design is about *n* times smaller than its 2D counter-part, thus exacerbating the problem of a degraded voltage supply in 3D designs [7].

The design of 3D MPSoCs faces another well documented challenge: that of achieving an acceptable thermal profile across the 3D die. Due to increased power densities and thermal resistivity in 3D chips, conventional air-cooled heat-sinks may be insufficient to remove heat dissipated in high-performance 3D chips [72]. Micro-fluidic cooling has recently been proposed as an attractive alternative due to the superior heat removal capability of liquids in comparison to air [43], [44]. *But co-optimizing cooling and PDN costs remains a challenging and unaddressed problem for 3D MPSoCs.*

Another critical component at the heart of emerging 3D MPSoCs is the network-on-chip (NoC) architecture that enables intra- and inter-layer communication between multiple cores. As the power dissipated in the NoC has become a significant portion of the total on-chip power, optimizing communication power has become a critical step in today’s chip design methodologies. *Prior works on NoC* synthesis *do not consider the design of the PDN while mapping cores and designing the NoC fabric, and typically generate a single power and performance optimized NoC configuration that may or may not meet thermal constraints.* Performing synthesis of the PDN for the generated NoC configuration in these cases can put stringent demands on the already strained PDN, making it extremely difficult to meet PDN constraints such as maximum IR-drop or leading to over-margining for the PDN (e.g., requiring large grid-wire width) that can be wasteful and prohibitively increase overall system cost.

We recognize the key insight that different instances of voltage partitioning and core-to-tile mapping can *significantly* alter the IR-drop distribution map seen by the PDN as well as the thermal profile of the

3D chip. Therefore our framework integrates efficient mechanisms for *VI*-aware core mapping on the 3D die and *exploits the interdependence between the synthesized 3D NoC configuration, the resulting thermal profile of the 3D chip, and its corresponding IR-drop distribution across the 3D PDN*. The novel contributions of our co-synthesis framework are summarized as follows:

- We integrate models of the PDN, micro-fluidic cooling, *VIs*, and 3D NoCs into a system-level optimization framework;
- We propose a design time force-directed algorithm to map cores onto the tiles of a 3D die while co-optimizing communication, thermal, and PDN design objectives to generate an optimized application-specific 3D MPSoC;
- We also propose a simulated-annealing based co-synthesis approach to optimize communication, thermal, and PDN goals;
- We design an algorithm for 3D routing path allocation which considerably minimizes power in the 3D NoC;
- Our framework ultimately generates a set of design points (Pareto mappings) that allow a designer to trade-off the quality of the PDN against NoC power costs and cooling power costs, and select a suitable solution that meets power, performance, and PDN cost based design goals.

The rest of the article is organized as follows. Section 3.2 presents an overview of related work in the area of PDN design, and NoC synthesis for 2D and 3D ICs. Section 3.3 discusses key issues related to PDN design with multiple voltages. Section 3.4 gives a brief background on the inter-tier liquid cooling setup in 3D ICs. Section 3.5 presents the problem formulation. Section 3.6 describes our co-synthesis framework (FDS-CoSyn). Section 3.7 elaborates on another framework that is based on a probabilistic metaheuristic and solves the same problem. Section 3.8 presents experimental results and Section 3.9 summarizes our conclusions.

3.2 Related Work

The problem of 2D NoC synthesis on regular structures with multiple supply *VIs* has been addressed in several prior works, such as [55], [24]. Authors in [24] limit power overhead by reducing the total number of voltage level converters (VLC) and multiple-clock first-in first-out (MCFIFO) frequency level converters needed for inter-island transfers. Our prior work [55] improves upon [24] with heuristics that generate a better core-to-tile mapping, and a routing scheme that more aggressively optimizes inter-island communication.

Given the promise of 3D technologies, 3D NoC synthesis in recent years has also attracted significant research efforts [45], [62], [66]. The authors in [62] perform min-cut partitioning of cores to establish core-to-router connectivity in 3D ICs, find paths for the communication flows and place network components on the 3D layers; while reporting savings in NoC power consumption and delay. In [66], improvements are shown over [62] by additionally considering the impact of TSV serialization and network interface (NI) placement in irregular 3D NoCs. Recently, [45] has proposed a novel 3D NoC topology employing randomized long-links at one or more dies of the 3D stack, thereby significantly reducing the average hop-count, communication-latency, and communication-energy compared to a regular 3D mesh topology. *However, none of these existing approaches have considered the impact of 3D NoC synthesis on the efficiency and overheads associated with 3D PDN design; in other words, these approaches are not PDN-aware. These prior efforts also do not consider optimization for micro-fluidic cooling based 3D ICs.*

On the other hand, techniques for optimizing PDNs in 3D ICs have been studied in several recent works [7], [49], [50]. In [7] the impact of TSV size/spacing on IR drops and voltage droops in several 3D PDN configurations is analyzed; for SPEC benchmarks running on a quad-core SoC. In [49] simulated annealing is used to co-synthesize the floorplan and P/G network, optimizing wire length, area, P/G routing area, and IR-drops. In [50] an integrated 3D TSV, thermal, and power distributed network (STDN) is proposed; and a simulated annealing floorplanner is used to minimize voltage drop and temperature in STDN. *None of these PDN optimization techniques considers the impact of the 3D NoC fabric and core mapping across layers, or considers 3D chips with micro-fluidic cooling as an optimization platform.*

Our recent work [67] motivates the need for a PDN-NoC co-synthesis methodology, and proposes a simulated-annealing based co-synthesis framework for 3D MPSoCs, which is shown to generate more efficient overall solutions compared to a PDN-unaware synthesis approach. But the proposed approach fails to consider the thermal costs and cooling costs associated with the generated solutions.

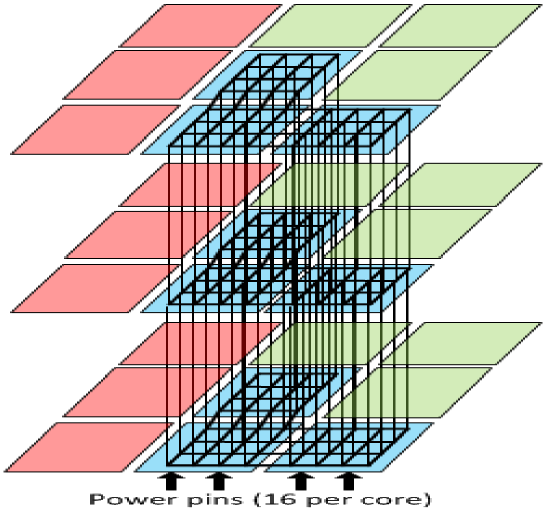


Figure 14: Example of contiguous 3D voltage islands (*VI*s) in a 27 core (3x3x3) 3D MPSoC. Cores within a single 3D-*VI* are vertically aligned (*VI*s are color-coded). Only the 3D power grid for the blue *VI* is shown for clarity. Each *VI* has a separate power grid, with 16 grid-nodes per core.

Unlike any prior work, in this article we present a novel thermal-aware co-synthesis framework for core mapping, PDN design, and mesh-based NoC design in 3D MPSoCs. To the authors' knowledge, this is the first work that proposes a system-level co-synthesis framework to co-optimize a 3D NoC fabric, 3D chip thermal profile, and 3D PDN fabric to produce a more efficient overall MPSoC design. This work significantly extends our recent conference publication [67] with new heuristics and the integration of thermal and cooling awareness during the optimization flow. Our experimental studies present detailed comparisons between our new framework in this article and the approach from [67].

3.3 PDN Design for 3D MPSOC

High circuit density in small footprint 3D ICs presents a unique challenge for designers of PDNs, as it requires the network to deliver significantly more current than in 2D ICs with fewer power-ground (P/G) bumps, while overcoming increasingly daunting IR-drop issues. As circuit delay is strongly co-related to the supply voltage drop in modern technologies, the PDN should at least be able to restrict IR drops at each core-input within the set tolerance limit, usually 5-10% of the rated core voltage [6]. In this work, we consider an MPSoC platform with a 3D mesh of tiles, with multiple voltage domains (i.e., voltage islands or *VI*s), where there is a one-to-one mapping of processing cores onto these tiles. The voltage-assignments of cores running a multi-programmed workload represent the operating frequency requirements of the tasks mapped on them, and the supply-current assignments of cores represent the power requirements for running the mapped tasks. In order for the PDN to handle multiple voltage domains, we assume that the MPSoC design is partitioned into 3D-*VI*s such that cores of the same voltage are vertically aligned in the 3D stack (as shown in figure 14), similar to the 'voltage volume' concept introduced in [50] (although, we consider multiple *VI*-configurations using all possible 2D-shapes of *VIs* in our synthesis approach). This enables independent and physically disjoint 3D power supply grids to connect all cores operating at the same voltage. The points of intersection in any 3D power grid are termed as *grid-nodes* which supply power to the cores (16 grid-nodes are assumed per core as shown in figure 14). The inter-layer connections in the grid are made using power TSVs. Even though all *VI*s are contiguous, we do not restrict the *VI* shapes to rectangles (as assumed in prior works, such as [68]). It is shown in [69] that sizing of pitches and widths of the power grid does not change the on-chip voltage distribution, therefore, in our work, we assume fixed uniform power grids, as in prior works [69], [70], i.e. *grid-nodes* and the corresponding power-TSVs/power-pins are located uniformly over each core (as shown in figure 14). Note, this work considers the steady-state characterization of the thermal, PDN, communication, and computation profiles of MPSoCs. Therefore, we investigate the steady state effects of the global power grid in terms of static IR-drop, while time-varying network characteristics such as transient noise are not considered.

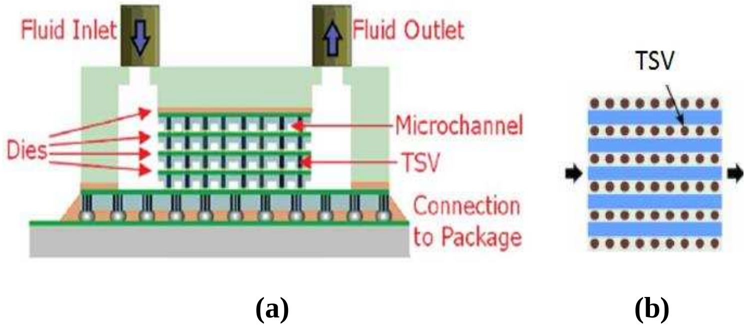


Figure 15: Inter-layer liquid cooling [43]. (a) Cross-section of a 3D chip (b) Top-view of micro-channels (in blue) with fluid flowing from left to right

3.4 Inter-tier Liquid Cooling in 3D ICs

The structure of the micro-fluidic layers with micro-channels for fluid delivery and with the possible locations of inter-tier through silicon vias (TSVs) is shown in figure 15(b). Given the effectiveness of micro-fluidic cooling, we do not need thermal-TSVs in 3D ICs, as also assumed in the other works on micro-fluidic cooling [43], [44], [74]. In this work, we use a micro-fluidic layer under each device layer as suggested in [43]. Note that as the micro-fluid within micro-channels progresses from the fluid inlet towards the fluid outlet (figure 15(a)), its capacity to absorb heat flux from device layers steadily decreases [74] from the inlet to the outlet. By increasing fluid flow-rates, it is possible for the fluid to absorb more heat flux from the cores closer to the outlet, helping to reduce their temperatures.

In this work, we assume that the hot water exiting the 3D chip is cooled (to a fixed ambient temperature) by a heat-sink with a radiator-fan assembly and fed back into the fluid-inlets of the chip (as in [63]-[65]). Higher fluid flow-rates require the fan-based heat-sink to accommodate higher heat transfer-rates (as discussed in section 3.6.5), with higher number of rotations per minute (rpm). Higher rpm values in turn increase the fan-power (P_{fan}), in fact, $P_{fan} \propto (rpm)^3$ [63]. Also, the electrical power required to pump the fluid into the 3D-chip is directly proportional to the square of the flow-rate ($P_{pump} \propto Fl^2$) [43]. Therefore, even though serious thermal problems can generally be alleviated by increasing fluid flow-rates, indiscriminately ramping up flow rates in practice can potentially lead to prohibitively high cooling power ($P_{fan}+P_{pump}$), and thus violate chip power budgets. *Consequently, the design of a micro-fluidic cooled 3D MPSoC requires careful planning during the allocation of computational and network resources on the die.*

Figure 16 shows a motivational example with power profiles for two different core-to-tile mappings onto a 2D 4×4 mesh, where a micro-channel layer below the die (device-layer) is assumed (as shown in figure 15). For the same application core-graph, cores with higher power dissipation values are mapped closer to the fluid-inlet in Mapping-I, whereas the higher power cores are mapped onto tiles closer to the fluid-outlet in Mapping-II. The steady-state thermal profiles are evaluated (using [71]) for Mapping-I and Mapping-II, for a fixed liquid flow-rate of 48ml/min, which corresponds to a total cooling-power (P_{cool}) of 1.1W [46], [65] (P_{fan}=0.8W and P_{pump}=0.3W). It is observed that the hottest tile corresponding to Mapping-I is 94^{o}C, whereas that for Mapping-II is 129^{o}C. In fact, to bring down the maximum temperature for Mapping-II below an assumed threshold of 95^{o}C, the flow-rate would need to be increased to 330ml/min, which corresponds to P_{cool}=65.9W (P_{fan}=51.7W and P_{pump}=14.2W). Such a high power overhead for cooling in 2D MPSoCs is unacceptable. In 3D ICs the required cooling power can be much greater, depending on the number of device-layers. Our proposed 3D co-synthesis framework therefore integrates thermal-awareness, optimizing for cooling power given a maximum temperature constraint.

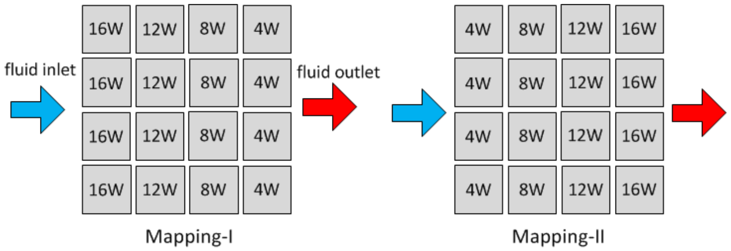


Figure 16: Motivational example for considering thermal-awareness in a liquid-cooled 2D MPSoC. Power profiles (in Watts) shown for the two mappings. For a flow-rate of 48 ml/min, thermal-aware Mapping-I produces a max. temperature of 94^{o}C and Mapping-II produces a max. temperature of 129^{o}C.

3.5 Problem Formulation

In this section we present our problem formulation. We assume the following inputs to our problem:

- A 3D IC with a regular 3D mesh NoC, with dimensions (dim_x, dim_y, dim_z) and number of tiles T = $dim_x \times dim_y \times dim_z$ with each tile containing a core and a NoC router;

- A core graph $G(V,E)$ with a set of T vertices $\{V_1,V_2, \ldots, V_T\}$ representing homogenous compute cores on which application tasks have already been mapped, and the set of M edges $\{e_1, e_2, \ldots, e_M\}$ that represent communication flow dependencies and requirements between cores;

- A minimum communication bandwidth constraint for each communication flow; and latency constraints represented as an upper bound on number of hops $\{h_1, h_2, \ldots, h_M\}$ for each communication flow;

- A set of pre-assigned triplets constituting operating voltages, operating frequencies, and maximum supply currents for the T cores $\{(v_1,f_1,i_1), (v_2,f_2,i_2), (v_3,f_3,i_3), \ldots, (v_T,f_T,i_T)\}$ in the core graph, to meet compute performance and power requirements of tasks mapped to the cores;

- A set of Ω supply voltage (V_{dd}) levels, which also represents the total number of VIs;

- *Ω* separate regular 3D power grids (each corresponding to a *VI*), with multiple power input pins for every grid;

- A maximum temperature constraint Ψ for the entire chip, and a max-IR-drop constraint Γ for the PDN.

Based on the minimum voltage/frequency requirements of the *T* cores (which in turn depends on the performance demands of tasks assigned on each core), we assume that voltage partitioning (assignment of available {voltage, frequency} pairs to the *T* cores) has already been performed using techniques from prior works e.g., [55], [24]. Note that as the (v_x,f_x,i_x) triplets for each core are fixed, total computation power in our co-synthesis framework is pre-determined. However, the manner in which compute and communication resources are mapped on the die can alter communication power, cooling power, and 3D PDN design costs.

Objective: Given the above inputs, the goal of our proposed framework is to obtain a core-to-die mapping and synthesize a regular 3D mesh NoC for a specific application, such that the application performance constraints (bandwidth and latency constraints), 3D-*VI* contiguity constraints (all cores are contiguously placed within individual 3D *VI*s), and the maximum chip temperature constraint are satisfied; while minimizing total communication power and cooling power, as well as PDN cost (represented by the maximum IR-drop). Note that our proposed framework is also applicable to 2D MPSoCs (with the vertical NoC dimension (dim_z) set to 1).

3.6 Co-Synthesis Framework Overview

We first present a brief overview of the design flow (shown in figure 17) of our co-synthesis framework (FDS-CoSyn). Based on the sizes (in terms of number of cores) of the *Ω VI*s and the dimensions of the mesh, a 2D-VI-shape library is initially generated. Given the core-assignments (v_x,f_x,i_x), this library is used to generate N core-to-tile initial mapping solutions (*IMs*). Given the core-graph *G(V,E)*, our force-directed swapping algorithm is applied on each of these *IM*s, to produce up to N feasible mapping solutions that are co-optimized for communication power, maximum temperature and max-IR-drop, while satisfying hop-constraints for all communication flows. These solutions undergo 3D routing path allocation (3D_Routing()), thermal simulation (Thermal_Eval()), and 3D PDN synthesis (PDN_Solver()) to produce a 3-tuple set of costs in terms of communication power, cooling power, and max-IR-drop.

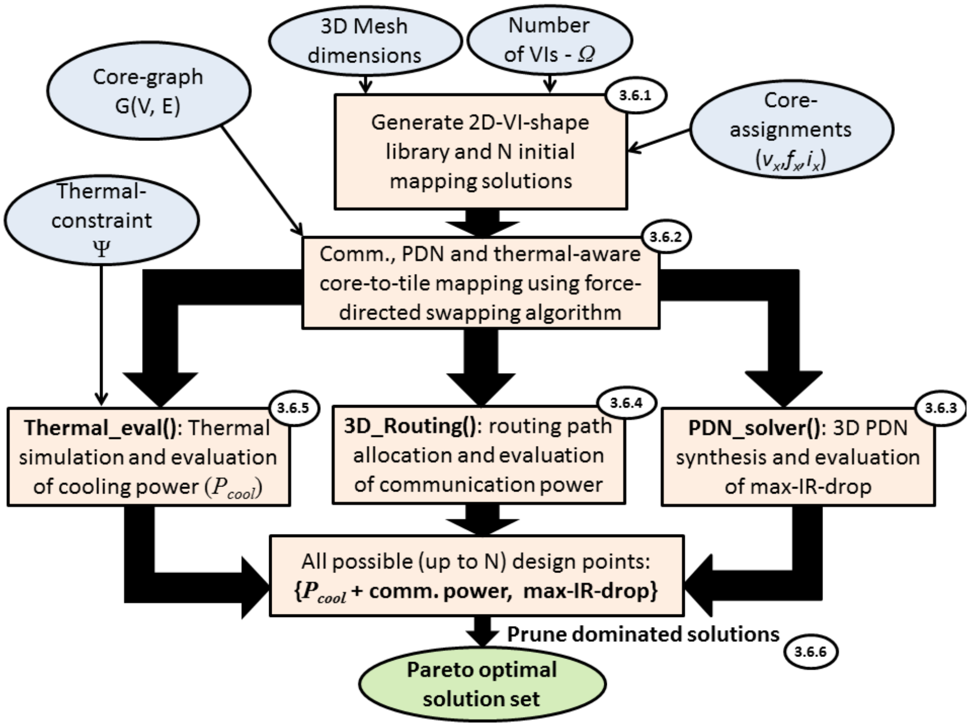

3D Mesh dimensions
Number of VIs - Ω
Core-graph G(V, E)
3.6.1
Generate 2D-VI-shape library and N initial mapping solutions
Core-assignments (v_x,f_x,i_x)
Thermal-constraint Ψ
3.6.2
Comm., PDN and thermal-aware core-to-tile mapping using force-directed swapping algorithm
3.6.5
Thermal_eval(): Thermal simulation and evaluation of cooling power (P_{cool})
3.6.4
3D_Routing(): routing path allocation and evaluation of communication power
3.6.3
PDN_solver(): 3D PDN synthesis and evaluation of max-IR-drop
All possible (up to N) design points:
{P_{cool} + comm. power, max-IR-drop}
Prune dominated solutions
3.6.6
Pareto optimal solution set

Figure 17: Design flow of our co-synthesis framework (FDS-CoSyn): block arrows indicate transfer of multiple (up to N) design solutions, small circles contain section-numbers that corresponding blocks are discussed in.

For any given core-to-tile mapping solution, our 3D_Routing() step satisfies communication constraints while optimizing NoC power; Thermal_Eval() determines the corresponding minimum cooling power required to satisfy the thermal constraint (Ψ); and PDN_Solver() synthesizes a 3D PDN and evaluates the corresponding max-IR-drop. Finally, up to N candidate co-synthesis solutions are obtained. Out of these candidate solutions, the ones that have both max-IR-drop and (communication +

cooling) power greater than some other solution are pruned to produce a set of Pareto optimal co-synthesis solutions, each optimized for the PDN, NoC, and thermal design objectives by varying degrees.

Thus, FDS-CoSyn essentially has two major stages: firstly, N initial mapping solutions are generated (as discussed in section 3.6.1); secondly, candidate final solutions (for the synthesized MPSoC) are produced from the N initial mapping solutions (discussed in sections 3.6.2 – 3.6.6). In the following subsections, we describe the algorithms used in each stage of our co-synthesis framework in detail.

3.6.1 Generation of Initial Mapping Solutions

Given that the MPSoC is partitioned into contiguous *3D-VI*s (as shown in figure 14), the number of cores within each voltage island is required to be a multiple of dim_z (number of device layers) in the 3D IC. As the number of tiles in a tier occupied by each *VI* is known a priori, in addition to mesh dimensions, a set of all possible 2D shapes of *VI*s can easily be enumerated. For example, figure 18 shows all shapes for a 2D-*VI* of 2, 3 and 4 tiles.

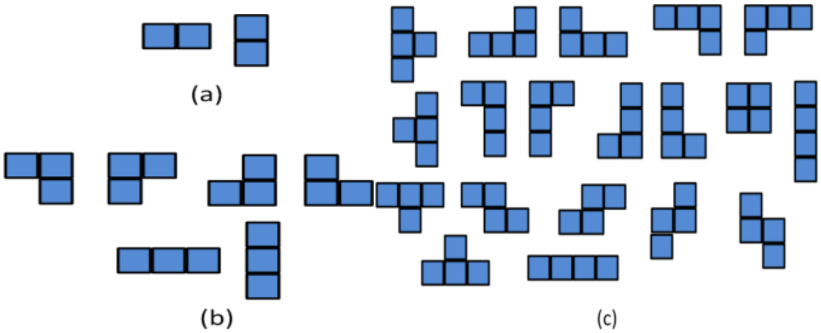


Figure 18: Library of all possible 2D-*VI*-shapes: - (a) 2 2D-*VI*-shapes of size 2 tiles (b) 6 2D-*VI*-shapes of size 3 tiles (c) 19 2D-*VI*-shapes of size 4 tiles

With the knowledge of all possible 2D shapes for all *VI*s, we create a 2D-*VI*-shape library for all *Ω* *VI*s. Using such a library, we invoke arbitrary shapes for the arbitrarily chosen *VI* and place them on the

2D mesh to generate a set of valid (non-overlapping) *VI*-configurations containing all Ω *VI*s. The pseudo-code for generating a set of N distinct *VI*-configurations is given below:

```
Algorithm 3.1: Generation of N distinct VI-configurations
inputs: 2D-VI-shape library for all $\Omega$ VIs and mesh dimensions

1: $i = 0$
2: while ($i < N$) { $j = 0$
3:     while ($j < \Omega$) {
4:         Randomly choose a VI
5:         choose 2D-VI-shapes from library for jth VI in an arbitrary order until one
           can be fitted onto the unused tiles of the current (ith) VI- configuration
6:         if no VI-shape can be fitted, drop this VI-configuration, go to step2
7:         else place the VI-shape, increment j }
8:     compare the new (ith) VI-configuration with all previous (0 to (i-1)th)
       configurations to check if it is duplicated
9:     if duplicated, drop this VI-configuration else save it and increment i }

output : N distinct 2D-VI-configurations
```

For each (ith) *VI*-configuration (**step 2**), all possible 2D-*VI*-shapes corresponding to the selected (jth) *VI* are chosen in an arbitrary order until one is found which can be fitted on to the unused tiles of the current *VI*-configuration (**step 5**). Note that the Ω *VI*s are also selected in a random order (**step 4**). If no 2D-*VI*-shape (corresponding to the size of jth*VI*) can be fitted on to the unused tiles, the current *VI*-configuration is dropped and the ith iteration is re-started (**step 6**). This procedure continues until a valid *VI*-configuration is obtained. The new found (ith) *VI*-configuration is saved only if it is different from all previously found (i-1) *VI*-configurations (**steps 8, 9**). Finally a set of N distinct *VI*-configurations are obtained.

Corresponding to the N 2D-*VI*-configurations, N 3D core-to-tile mapping solutions are arbitrarily constructed with all 3D-*VI*s contiguously placed (as shown in figure 14).

3.6.2 Core-to-tile Mapping with Force-directed Swapping

Each initial mapping solution generated in the previous stage corresponds to a distinct global *VI*-configuration. In this step, for each of the N global *VI*-configurations, we perform swaps between adjacent cores within the same *VI* (i.e. intra-*VI* swaps), without altering the *VI* mapping of each generated configuration from the previous stage (Section 3.6.1). We associate any given mapping solution (for a *VI* configuration) with a force-directed map, where every core in the 3D mesh is acted upon by various pulling forces in one or more directions.

For a given mapping, these forces are dependent on (*i*) estimated inter-core communication in the NoC, (*ii*) estimated thermal profile, and *(iii)* estimated IR-drop distribution in the corresponding PDN. The main goal of this stage is to minimize the sum total of these forces, which corresponds to reducing costs associated with communication power, cooling power, and the PDN fabric. To capture this multi-objective optimization in the force-directed map, we define the following five force-components (FC_1-FC_5) for a core C_i (core under consideration), with α, γ, β_1, and β_2 being weighting factors for 4 of these optimizations corresponding to FC_1-FC_4:

1) <u>Communication traffic optimizing force component (FC_1)</u>: The basic premise of using FC_1 is that by mapping cores such that communication flows with higher bandwidth requirements have shorter routing paths (Manhattan-distances for minimal routing), the total network traffic (and thus the NoC power) is optimized. FC_1 is derived from the communication dependencies (ingress as well as outgress) of core C_i with other cores on the 3D mesh. FC_1 is subdivided into its *x*, *y* and *z* directional components FC_{1x}, FC_{1y} and FC_{1z}:

$$FC_1\{C_i\}_x = \alpha * \sum_j MD_{jx} * BW_j \qquad \text{…. (3.1)}$$

$$FC_1\{C_i\}_y = \alpha * \sum_j MD_{jy} * BW_j \qquad \text{…. (3.2)}$$

$$FC_1\{C_i\}_z = \alpha * \sum_j MD_{jz} * BW_j \qquad \text{…. (3.3)}$$

where, FC_1 is computed over all communication flows associated with the core C_i under consideration. For a given mapping, MD_j is the Manhattan distance (in the respective direction) between the two communicating cores for the jth flow; and BW_j is the communication bandwidth of the jth flow. Note that

Manhattan distances in negative *x*, *y* and *z* directions are treated as negative numbers; thus, the directional subcomponents of FC_1 can take positive or negative values.

2) Max-IR-drop optimizing force component (FC_2): The basic premise of using FC_2 is that by mapping cores with higher supply current requirements to tiles on the 3D mesh closer to the external input power pins, the max-IR-drop can be reduced. Assuming that the power pins are at the bottom of the 3D stack (figure 15(a)), the bottom tier is indexed dim_z and top tier has an index of 1. Therefore, FC_2 for each core is always directed in the positive *z* direction (downwards, towards I/O pins), with a magnitude depending on its tier index (tier#) as well as its maximum current requirement (i_{Ci}) in relation to the lowest current requirement of any core on the 3D mesh (i_{min}):

$$FC_2\{C_i\} = \gamma * \left(dim_z - tier\#_{C_i}\right) * (i_{C_i} - i_{min}) \qquad \ldots. (3.4)$$

3) Temperature optimizing force components (FC_3 and FC_4): The central idea behind incorporating these force components is that by mapping the cores with higher power values (i_{Ci}×v_{Ci}) closer to the fluid inlet of the microfluidic-cooling system, the chip cooling power can be optimized while meeting the thermal constraints. We assume that the micro-fluidic channels are laid out along the *x*-direction, i.e., inlet at x=1 and outlet at x=dim_x. FC_3 represents the horizontal force directed in the negative-x direction (towards the micro-fluid inlet):

$$FC_3\{C_i\} = \beta_1 * dist_x * (i_{C_i} - i_{min}) \qquad \ldots. (3.5)$$

where, $dist_x$ is the distance of C_i (in the negative-*x* direction) from the leftmost (with least *x*-coordinate value) core within its own *VI*. As discussed earlier, no swaps are performed across the 3D-*VI* boundaries; therefore, the horizontal force component FC_3 is based on the location of C_i within its own *VI*. Also, for a well-distributed temperature profile (which facilitates efficient cooling), the total power dissipation on each tier should be generally balanced. Therefore, FC_4 is a force component in the vertical (up{-*z*} or down{+z}) direction for relatively higher current cores, which attempts to balance the power between adjacent tiers:

$if\ ((i_{C_i} > i_{avg})AND(Power_{tier\#} > Power_{tier\#+1}))$

$$FC_4\{C_i\}_{+z} = \beta_2 * (Power_{tier\#} - Power_{tier\#+1}) * (i_{C_i} - i_{avg}) \quad \text{…. (3.6)}$$

$else\ FC_4\{C_i\}_{+z} = 0$

Similarly,

$if\ ((i_{C_i} > i_{avg})AND(Power_{tier\#} > Power_{tier\#-1}))$

$$FC_4\{C_i\}_{-z} = \beta_2 * (Power_{tier\#} - Power_{tier\#-1}) * (i_{C_i} - i_{avg}) \quad \text{…. (3.7)}$$

$else\ FC_4\{C_i\}_{-z} = 0$

where, tier# is the tier that C_i belongs to, (tier#-1) and (tier#+1) are the tiers directly above and directly below tier#, respectively; and i_{avg} is the average value of maximum rated current out of all cores on the 3D mesh.

Note that our force-directed mapping approach can *adapt to the type of cooling mechanism employed*. For instance, if a conventional air-cooled heat-sink is used, $dist_x$ can be replaced by $dist_z$ in FC_3, to map potentially hot cores closer to the heat sink, and the power gradient between tiers can be formulated in FC_4.

4) Force component (FC_5) to satisfy hop-constraints: Force component $FC_5\{C_i\}$ goes into effect (becomes non-zero) only when at least one of the following conditions hold:

i. one or more hop-constraint violations exist on ingress or outgress flows of C_i

ii. one or more hop-constraint(s) associated with C_i are barely met, i.e., one wrong swap can introduce hop-constraint violation(s)

If condition-i holds, the value of FC_5 (for the corresponding direction) becomes infinite, so that C_i will be swapped in this direction for the violation to be corrected. If condition-ii holds, FC_5 acts as an infinite inertia force (instead of a pulling force) prohibiting C_i from being swapped in the opposite direction, thereby avoiding the introduction of a hop-constraint violation. Thus, FC_5 will only contribute to the overall force-directed mapping if a violation exists; otherwise it will constantly act as a deterrent for new violations.

Having summarized the five main force components, we now present details of our iterative force-directed swapping algorithm which optimizes the given initial mapping solution for all three objectives (communication power, cooling power, PDN cost). For any given mapping, a force-directed map (as shown in figure 19(a)) is first created which represents the net forces in *x*, *y*, and *z* directions. Each directed edge in figure 19(a) represents a force between the two cores it connects. These directed forces are computed by summing all individual directed force components (in the same direction), of the corresponding core.

$$F\{C_i\} = FC_1\{C_i\} + FC_2\{C_i\} + FC_3\{C_i\} + FC_4\{C_i\} + FC_5\{C_i\} \qquad \text{…. (3.8)}$$

Note that a positive net force in +*x*-direction is equivalent to a negative net force in –*x*-direction. As the inertia force of FC_5 only precludes any violation causing swaps and does not contribute to the force-directed map, the FC_5 term in the above equation represents just the pulling force component. Corresponding to the resulting force-directed map, reciprocal forces of attraction (RFAs) between each pair of adjacent cores on the 3D mesh are computed as shown in figure 19(b).

The pseudo-code for our force-directed swapping algorithm is shown below. The algorithm is intended to reduce the sum total force in the entire 3D mesh. It is applied to all N initial mapping solutions (*IM*s) (**step 1**). After every swap on the current mapping solution, the force-directed map is rebuilt (**step 3**). The algorithm searches for a pair of adjacent cores with the highest RFA on the current 3D force-directed map to choose the next swap (**step 4**). Any swap under consideration is executed only if the following pre-conditions are met:

1) Both cores belong to the same *VI*

2) Swap will reduce the sum of all net forces in the 3D mesh

3) Inertia force (of FC_5) does not prohibit this swap

If all three pre-conditions of the swap under consideration are not met (**step 7**), this swap is invalidated to restrict it from being considered again until a valid swap is found. On the other hand, if all pre-conditions are met (**step 6**), then the swap is executed and all previously invalidated swaps on the 3D mesh are re-validated to be considered for swapping in the next iteration. With every successive swap, this iterative swapping process continues to reduce the sum of all net forces in the 3D mesh. Finally, when all swaps are invalidated on the 3D mesh (i.e., no swap meets the 3 pre-conditions), a state of equilibrium is reached on the force-directed 3D map and the swapping algorithm terminates (for the current mapping solution). As our force-directed algorithm is applied to all N initial mapping solutions, up to N feasible candidate mapping solutions are output at the end of this stage.

Algorithm 3.2: Core mapping with force-directed swapping

inputs: core-graph G(V, E) and N initial mapping solutions

```
1: for each mapping-solution do {
2: do {
3: build the force-directed map for the current mapping solution
4: choose the pair of adjacent cores (which are not invalidated) with the
   maximum reciprocal forces of attraction (RFA)
5: if all swaps on 3D mesh are invalidated (equilibrium achieved),
   save the mapping solution and go to (step 1)
6: if the 3 pre-conditions for this swap are met, execute it and re-validate all
   swaps on the 3D mesh; go to next iteration (step 2)
7: else invalidate this swap for the current iteration; go to step 4}}
```

output: up to N feasible mapping solutions optimized for all 3objectives

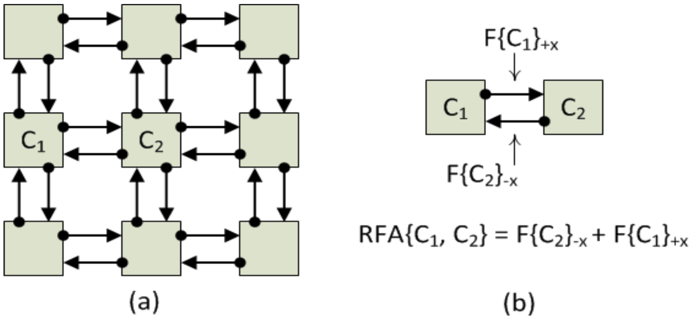


Figure 19: Forces in the force-directed swapping algorithm: (a) A force-directed map generated for any mapping solution (*z*-dimension is omitted for clarity); (b) Reciprocal force of attraction between pair of adjacent cores.

3.6.3 LP-formulation for 3D PDN Synthesis *(PDN_solver())*

As discussed earlier, there are Ω regular 3D power grids in our PDN, one for each 3D-*VI*, and the individual *VI*s (on the 2D plane) are not necessarily rectangular in shape. We assume equal number of grid-nodes (in an n×n 2D grid) supplying to each core on the MPSoC, where n=4 in figure 14. Thus, given *T* cores on the 3D mesh, the total number of grid-nodes in the PDN are $T \times n^2$. We also assume that the power pins are located below the bottom tier, i.e., grid-nodes of just the bottom tier are connected to the power pins, therefore, the total number of external power inputs are $(T \times n^2)/dim_z$ and all vertical PDN branch currents flow in the upward direction. Note that grid-nodes of the Ω power grids are not connected to each other. For any given candidate mapping solution, we use a linear programming (LP) formulation *to solve for the grid-node voltages and currents flowing in the branch resistances of the PDN with multiple 3D grids.* Our final metric of interest is the percentage max-IR-drop in the entire PDN, which we obtain from all the grid-node voltages. The reader is referred to APPENDIX I for details of our LP-formulation. Note that we employ an LP-solver [75] as it seamlessly integrates into our system level framework; nevertheless we validated our LP-based PDN synthesis approach using spice simulations.

Algorithm 3.3. Routing Path Allocation

input: core graph G(V, E), candidate-mapping-solution

```
1: Sort all communication flows in the increasing order of path lengths
   and in decreasing order of bandwidths for the same path length
2: for each communication flow in sorted list do {
3: Out of all the candidate minimal paths, choose the set of paths
   that would require the least number of inter-island link insertions
4: Out of the chosen paths, choose the subset of paths that would result in
   the minimum number of intra-island link insertions
5: Out of the chosen paths, choose the one path that uses the least number
of inter-island links
6: Insert the necessary links and allocate the current flow bandwidth
   over the chosen routing path }
```

output :Synthesized NoC with all communication flows routed

3.6.4 Routing Path Allocation *(3D_Routing())*

For inter-island communication, voltage level converters (VLCs) and frequency level converter resources (MCFIFOs) are required in the *VI* boundary routers. *These frequency and voltage conversion components incur power dissipation and delay overheads.* Thus, the main objective of our 3D routing path allocation step is to find a minimal path for each communication flow such that the number of additional inter-island link insertions is minimized. Additional inter-island links are inserted only when minimal paths cannot be found within the residual bandwidth capacities of the existing inter-island links.

Algorithm 3.3 summarizes our routing path allocation algorithm. The order in which communication flows are routed is determined in the following manner. The flows with longer minimal paths (MDs) have more choices for routing and thus have a larger scope for optimization. Also, flows with smaller bandwidths, require less residual capacities to be accommodated within existing links. Therefore, flows are sorted in the increasing order of their path lengths, in decreasing order of their bandwidths for the same path length; and considered for routing in that order (**step 1**).

For each communication flow (**step 2**), we consider all candidate minimal paths. Note that, the number of all possible minimal paths between two cores on a 3D mesh, which are *N* hops apart ($N = x + y$

+ *z*; where *x*, *y* and *z* are the number of *x*-hops, *y*-hops and *z*-hops on the 3D path) is given by $\{^{N}C_{x}\}\times\{^{(N-x)}C_{z}\}$. Here, $\{^{N}C_{x}\}$ represents the number of possible ways the *x*-path can be constructed; and $\{^{(N-x)}C_{z}\}$ represents the number of possible ways the *z*-path can be constructed, for a given *x*-path. Out of these candidate minimal paths, we choose a path based on the following optimization objectives (in that order):

(1) Minimize the total number of inter-island link-insertions needed on the path;

(2) Minimize the total number of intra-island link-insertions needed on the path; and

(3) Minimize the number of inter-island links used by the path

Note that as power optimization is the principle goal of our routing and link-insertion algorithm, the primary objectives (1) and (2) above reduce NoC power dissipation, whereas objective (3) reduces path-latencies by incurring minimal delay-overheads. To meet these objectives, we first choose paths that need the minimum total number of inter-island link insertions (**step 3**). Out of the chosen paths (with the same number of inter-island link insertions), we choose the ones that need the minimum total number of intra-island link insertions (**step 4**). Then, out of the chosen paths (with the same number of inter-island and intra-island link insertions), we choose the path which crosses the minimum number of inter-island links (**step 5**). When a path is chosen, the current communication flow (bandwidth) is allocated to its constituent links (**step 6**). Note that while routing any flow over a given path, links insertions are performed whenever the existing link(s) cannot support the bandwidth of the current flow or if no links are available. We also perform a post-processing design time cyclic dependency analysis using the algorithm in [73], to ensure freedom from cyclic dependencies that can cause deadlock at runtime. After this step, a mapped and routed 3D NoC-based MPSoC is obtained for the given application.

After routing is completed for all communication flows, the aggregate communication power dissipation and path-latencies in the NoC are computed taking into consideration the number of links inserted, link loads, router sizes, number of VLCs and MCFIFOs used, and corresponding voltage/frequency values.

3.6.5 Thermal Evaluation *(Thermal_eval())*

To perform thermal evaluation of any given mapping solution in our framework, we utilize the open-source thermal emulator 3D-ICE 2.2.5 [71], which supports steady-state thermal analysis of 3D ICs with inter-layer liquid cooling. As discussed in section 3.4, by increasing fluid flow-rates, the maximum temperature on the 3D die can be reduced. Thus, to evaluate the minimum flow-rate required to satisfy the maximum temperature design constraint Ψ, we invoke the 3D-ICE tool multiple times (on each mapping solution) while increasing the flow-rate in fixed increments until the thermal constraint is satisfied. Once the required flow-rate (*Fl*) is determined, the cooling power (power consumed by the pump and the cooling fan {$P_{cool}=P_{fan}+P_{pump}$}) corresponding to this flow-rate is calculated as follows:

(i) Evaluation of P_{pump}: The pump-power is generally defined as $P_{pump} \propto \Delta P \times Fl$, where ΔP is the pressure difference between the inlet and outlet, and *Fl* is the fluid flow-rate. Given the number and dimensions of micro-channels, and the fluid flow rate *(Fl)*, we evaluate the pump-power using the power model in [46].

(ii) Evaluation of P_{fan}: The thermal resistance (R_{th}) of the heat-sink (in Kelvin/Watt) is generally expressed as:

$$R_{th}=\Delta T/(dq/dt) \quad …….. (3.9)$$

where ΔT is the difference between the temperature of hot fluid entering the fan-based heat-sink (or exiting the 3D-chip) and the target ambient temperature that the heat-sink is required to cool the fluid to (to feed back into the chip), and *dq/dt* is the required heat-transfer rate (or thermal power in Watts) for this purpose.

Based on the definition of a calorie and assuming 1ml=1gm for water, the required heat transfer rate to achieve fixed target fluid-temperature, *dq/dt*, for the given *Fl* and ΔT can be expressed as:

$$dq/dt = 4.18 \times \Delta T \times Fl \quad ……. (3.10)$$

Once R_{th} is calculated from equations (3.9) and (3.10), the corresponding fan-speed (rpm) is found using the relationship between R_{th} and rpm from [65] for a 120mm diameter fan. We assume 6W of fan-power for a nominal fan-speed of 2000 rpm for a typical commercially available 120mm fan-based heat-sink. Thus P_{fan} is scaled based on different rpm values ($P_{fan} \propto (rpm)^3$).

Finally, the cooling-power ($P_{cool} = P_{pump} + P_{fan}$) is recorded for the mapping solution under consideration.

3.6.6 Solution Pruning

The set of up to N solution points produced by our co-synthesis flow will each represent a 3-tuple set of costs {communication power, cooling power, PDN max-IR-drop}. As our goal is to minimize the sum of communication power and cooling power, as well as the max-IR-drop in the PDN, we construct an optimized 2D Pareto front of solution points, each representing {communication power + cooling power, max-IR-drop}. To this end, solution pruning is performed to remove clearly dominated solutions. The designer is ultimately presented with a set of non-dominated Pareto solution points that trade off PDN vs. (NoC + cooling power) design objectives by varying degrees, while satisfying application performance and thermal constraints.

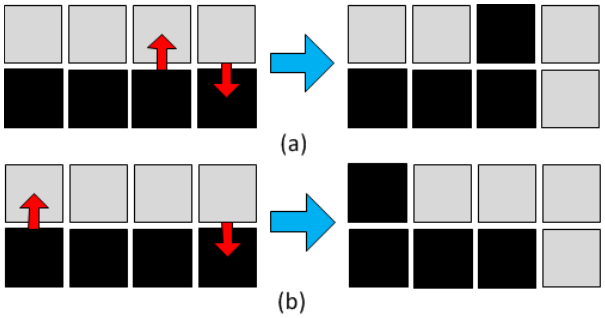


Figure 20: Example of a tile-exchange perturbation (a) an invalid perturbation where the contiguity of *VI*s is not retained (b) a valid perturbation

3.7 Co-Synthesis with Probabilistic Metaheuristic

Simulated annealing (SA) is a probabilistic metaheuristic that has found widespread application in several physical design problems [61] and is a widely accepted technique for multi-objective optimization. For the same problem formulation (as in Section 3.5) as well as for the same set of assumptions pertaining to 3D-*VI*s, 3D-PDN, and micro-fluid cooling, we also create an SA based co-synthesis framework for co-optimization of the same objectives {cooling+communication power, max-IR-drop}.

In this SA-based co-synthesis framework (SA-CoSyn), we utilize the same functions PDN_solver(), 3D_Routing() and Thermal_eval() discussed in sections 3.6.3, 3.6.4, and 3.6.5 respectively. This framework is loosely based on the SA based approach for PDN-NoC co-synthesis in our prior work [67]; however that work did not consider thermal awareness and micro-fluidic cooling, and used an SA based core-to-tile mapping approach that is different from the force-directed swapping based core-to-tile mapping proposed in this article.

We designed a dual-objective SA based algorithm. Instead of saving just the best solution at each iteration, we maintain a Pareto front of two dimensional (PDN and (NoC + cooling)) costs that are not dominated by any solution found so far. The three possible solution perturbations used in the algorithm are:

<u>*Perturbation 1*</u>: *core-swap* – Two cores randomly selected from the same 3D-*VI* are swapped. As the swap takes place within the same 3D-*VI*, *VI*-contiguity is not a concern.

<u>*Perturbation 2*</u>: *3D-VI-swap* – Two 3D-*VI*s (of equal sizes), are randomly selected to be swapped.

<u>*Perturbation 3*</u>: *tile-exchange* between two separate 3D-*VI*s – The *VI* configuration in the 3D mesh is changed by reciprocal occupation of a tile (tile-exchange) between two adjacent *VI*s (with at least 2 tiles in each *VI* being adjacent to the other *VI*) on every tier of the 3D MPSoC. A tile-exchange is valid only if both participating *VI*s retain their contiguity (figure 20).

Note that entire columns of cores (across all tiers) need to be exchanged between the 3D-*VI*s, and if this perturbation is valid on one tier it is valid on all tiers because all 3D-*VI*s are vertically aligned across tiers. The pseudo-code for our SA-based co-synthesis framework is described below.

Algorithm 3.4: SA-based co-synthesis framework

inputs: as described in section 3.5

```
1: Generate a valid initial mapping and evaluate cost (current solution)
2: Set T to $T_{init}$ and counter to 0
3: while (terminating-condition not reached) { K=0
4:     while (K < $K_{max}$) {
5:         Perturb current mapping solution in one of 3 ways to get new solution:
6:         (i) Run PDN_solver() to evaluate new mapping solution for  max-IR-drop
7:         (ii) Perform 3D_Routing() and evaluate communication power
8:         (iii) Perform Thermal_eval() and evaluate cooling power ($P_{cool}$)
9:         Compute: cost(new solution) = $\omega\times$(communication power + $P_{cool}$)
             + $\lambda\times$(max-IR-drop) + $\phi\times$(# of hop-constraint violations) +
             {if (max-IR-drop > $\Gamma$) then $\phi'\times$(max-IR-drop – $\Gamma$) else 0}
10:        If no hop-constraint violations exist and PDN constraint satisfied,
             update the Pareto front and the counter
11:        Set value of ACCEPT according to the SA-acceptance criterion
12:        If (ACCEPT=1), then cost(current solution) = cost(new solution)
13:        K= K+1 }
14:    T = $\delta\times T$ }
```

output : final set of feasible solutions on the 2D Pareto front with the associated (NoC+cooling) costs and PDN costs

An initial mapping solution is arbitrarily generated, which satisfies basic 3D-*VI* contiguity constraints. The cost of this initial solution is computed by calling the PDN_solver(), 3D_Routing() and Thermal_eval() functions. The initial solution now becomes the current solution in the SA process **(step1)**. To initiate the SA process, the SA-temperature parameter (T) is set to T_{init} **(step2)**. At each iteration, one of the three perturbations is randomly chosen to perturb the current solution. To generate enough mapping solutions for every 3D-*VI* configuration, we choose perturbations 1, 2 and 3 with probabilities 10/13, 1/13 and 2/13 respectively **(step5)**. To evaluate the new (perturbed) mapping for communication power, cooling power (P_{cool}) as well as PDN max-IR-drop, our PDN_solver(), 3D_Routing() and Thermal_eval() are invoked **(steps 6, 7, 8)**. Then, the cost of this new solution is computed **(step9)**. Here, the coefficients of the terms with total number of hop-constraint violations (ϕ)

and PDN constraint violation (ϕ’) are set to high values, in order to penalize infeasible mapping solutions. If the new solution corresponds to no violations and does not have both power cost and PDN cost greater than any solution on the current Pareto front (non-dominated solution), it is inserted into the Pareto front. Any Pareto solutions that get dominated as a result are discarded from the front. Thus, the Pareto front of solutions (with non-dominated costs), is checked for an update at every iteration of SA **(step10)**. Solutions with better (smaller) cost than the current solution are accepted, and solutions with worse costs are either accepted or rejected according to the following acceptance criteria **(step11)**:

$$if\ (r < exp\ ([cost(current\ solution) - cost(new\ solution)]/T)\ \ldots.\ (3.11)$$

$$ACCEPT\ =\ 1$$

$where, r\ is\ a\ random\ number\ between\ 0\ and\ 1$

After every K_{max} iterations, the SA-temperature is scaled by the parameter δ. Finally, when no new solution is added to the Pareto front for L number of consecutive SA-iterations, the SA process terminates (*terminating-condition*), and a set of 2D Pareto design points are produced.

3.8 Experimental Studies

3.8.1 Experimental Setup

We use the ARM Cortex-A9 multi-core processors [37] as the baseline MPSoC compute cores in our experiments. These processors support three operating voltage levels (Ω=3): 0.9V, 1.0V, and 1.1V; and corresponding operating frequencies of 1310MHz, 1550MHz, and 1775Mz; at the 45nm process technology node. The maximum current requirements for the processing cores range from 1A to 4A, based on the level of compute intensity of the tasks assigned to the respective cores. Even though we consider ARM processors in the platform used to evaluate our framework, it should be noted that our framework is applicable to multi-core platforms with any processor architecture, given the operating ranges of supply voltages, frequencies, and maximum-powers for the processor. We assume that the rated maximum supply current (and the corresponding maximum power) of each core is sufficient for the associated router as well. In our studies, we use a 60-core 3D-mesh for the MPSoC with dimensions 5×3×4 (dim_x×dim_y×dim_z). In addition, we also use a 100-core 3D-mesh of similar homogeneous cores, with dimensions 5×4×5 assuming a 32nm process technology node. The values of core-voltages, frequencies, powers, and area are scaled by factors of 0.925×, 1.1×, 0.626×, and 0.57× respectively to capture the effects of technology-scaling (from 45nm to 32nm), as suggested in [52].

Our experiments were conducted using six parallel application benchmarks from the SPLASH-2 [58] and PARSEC benchmark suites [56]: *streamcluster, ocean,* and *cholesky* for 60-cores (*fluidanimate, lu,* and *vips* for 100-cores) of low, medium, and high communication-intensities respectively. Our core-graphs are modeled based on inter-core communication characterizations given in [47]. Each vertex in a core-graph corresponds to a core and the edge weights represent inter-core communication latency and bandwidth requirements (based on our observations of traces and communication patterns between cores). Our synthesis framework ensures that the minimum communication bandwidth constraints are satisfied, with hop-constraints representing the actual communication latency constraints.

The power values of routers and links (32-bit wide) for different voltages, frequencies, and router complexities at varying communication loads, for 32nm and 45nm process nodes are obtained from ORION 2.0 [40]. As the VLCs and MCFIFOs required to enable error-free interactions between *VI*s incur a power overhead that is proportional to their voltage supply, we consider the power overhead of these components, with the actual power values based on reported overheads from existing literature [53], [59], [60]. In *Thermal_eval(),* we conservatively set the maximum thermal constraint (Ψ) to 75^{o}C and for the given mapping solution *Fl* values are increased in until Ψ is satisfied. The corresponding P_{pump} and P_{fan} values are calculated as discussed in section 3.6.5. Single-phase cooling with de-ionized water as coolant material is assumed (similar to [74]). 3D-ICE assumes identical micro-channels, uniformly placed, in all micro-fluidic layers. We set the height and the cross-sectional width of micro-channels to 100μm and

200μm respectively. Micro-channels are assumed to be horizontally separated by silicon walls, each of width 400μm (to accommodate the inter-tier signal and power TSVs). The temperature of the coolant fed back into the 3D-chip is assumed to be fixed at 30°C ambient, which would satisfy most environments. Our regular 3D-PDN power grids are modeled based on the guidelines provided in [7]. For the 60-core mesh, with 15 cores on each tier (20 cores for the 100-core mesh), a total of 240 (320 for 100-core mesh) input power pins are used with $n^2=16$ grid-nodes for each core. Thus, the total number of grid-nodes in the entire PDN is equal to 960 (1600 for 100-core mesh). For the PDN corresponding to the 60-core mesh, values of R_h=40mΩ and R_v=80mΩ are assumed (based on [7]) for the horizontal and vertical branch resistances. For the 100-core mesh, R_h=28mΩ is assumed in accordance to the reduced area, whereas the TSV height is kept unchanged across technologies (as assumed in [51]).

For the implementation of our FDS-CoSyn framework, we generate *N* initial mapping solutions from an equal number of distinct 2D-*VI*-configurations, where $N=66$ for the 60-core case ($N=68$ for the 100-core case), is the number of 2D-*VI*-configurations that can be readily found using algorithm 3.1. Values of 0.3, 0.1, 0.1, and 0.5 are used for α, β_2, β_1, and γ respectively. For the SA-based co-synthesis algorithm, we set $T_{init}=100$, $K_{max}=100$, the scaling factor $\delta=0.9$, and $L=250$.

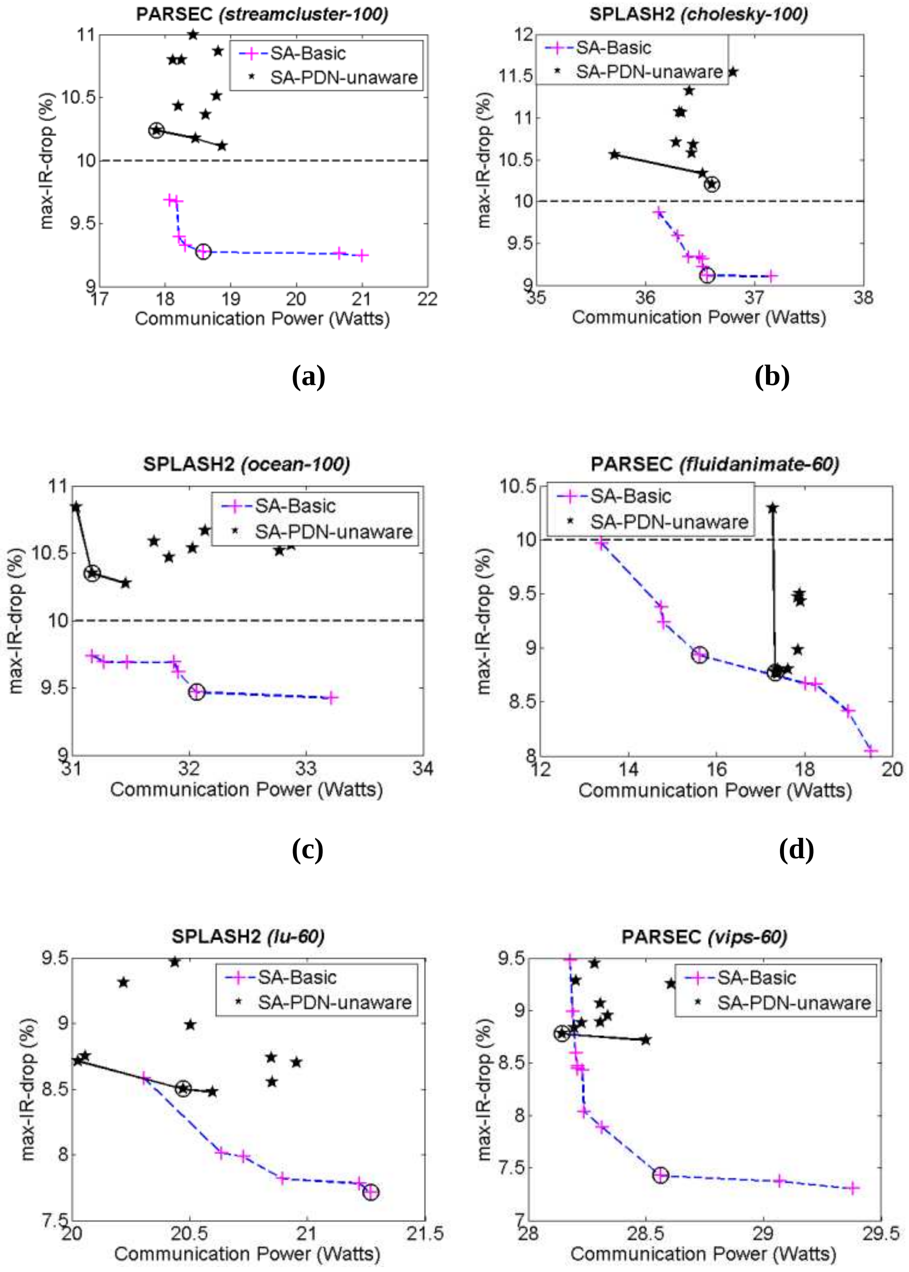
3.8.2 Experimental Results

In this section, we present the experimental results to highlight several interesting insights of our proposed framework. Sections 3.8.2.1 and 3.8.2.2 show the importance of employing PDN-awareness and thermal-awareness during synthesis of MPSoCs, respectively. Section 3.8.2.3 shows the efficacy of the proposed FDS core-mapping approach, in comparison to simulated annealing approaches. Section 3.8.2.4 shows the advantages of using our *VI*-aware routing algorithm in comparison to the traditional dimension-order routing schemes. Finally, sections 3.8.2.5 and 3.8.2.6 show the results of sensitivity analysis of the different parameters utilized in the FDS-CoSyn framework, and that of using varying power-grid granularities, respectively.

3.8.2.1 Importance of PDN-awareness

The SA-based PDN-NoC co-synthesis framework (*SA-Basic*), introduced in [67] performs PDN-NoC co-synthesis for MPSoCs. To illustrate the importance of PDN-awareness during MPSoC synthesis, we compare the approach from [67] (*SA-Basic*) which is PDN-aware with an SA-based PDN-unaware framework (*SA-PDN-unaware*), where the optimization objective is primarily to minimize NoC power. The 2D solution space generated by the two frameworks for six SPLASH-2 and PARSEC benchmarks is shown in figure 21, with each candidate solution characterized by its communication power and the percentage max-IR-drop in the PDN. The 2D Pareto front enables the designer to choose a design point for an appropriate trade-off between NoC costs and PDN costs. In the *SA-PDN-unaware* framework, we run the SA algorithm exactly 10 times (starting with different initial mappings) to generate 10 distinct solution points (shown by black stars in figure 21). The PDN unaware approach, which is representative of system-level 3D NoC synthesis approaches proposed in literature to date, optimizes for NoC power, but the solutions generated generally have much higher values of max-IR-drop. Observe that all of the solution points from *SA-PDN-unaware* corresponding to the three 100-core benchmarks (shown as black stars in figures 10(a)-(c)) violate the PDN constraint Γ (max-IR-drop of 10%), rendering infeasible final solutions.

Each design point on the Pareto front in figure 21 represents two different quantities: communication power and percentage max-IR-drop. For a quantitative comparison of the results obtained, we select the best point (knee-point) on each Pareto front. The knee-point of a 2D Pareto front is characterized by a small improvement in one objective causing a large deterioration in the other objective, thus making it unattractive to move in either direction on the front [48]. In our analysis, we define the knee-point as the point on the Pareto front with the smallest ratio of (improvement in one objective) /(deterioration in other objective), when moving in either directions. The en-circled points in figure 21 represent the knee-points associated with their respective Pareto fronts. As the goal of our co-synthesis framework is overall optimization, evaluating solutions on the basis of only power or IR-drop will be inappropriate. Therefore, while comparing knee-points of two different Pareto fronts, we consider the percentage gross-improvement: (% improvement in power) + (% improvement in max-IR-drop).



(a) **(b)**

(c) **(d)**

(e) **(f)**

Figure 21: Results for *SA-PDN-unaware* and *SA-Basic* (PDN-aware) synthesis frameworks

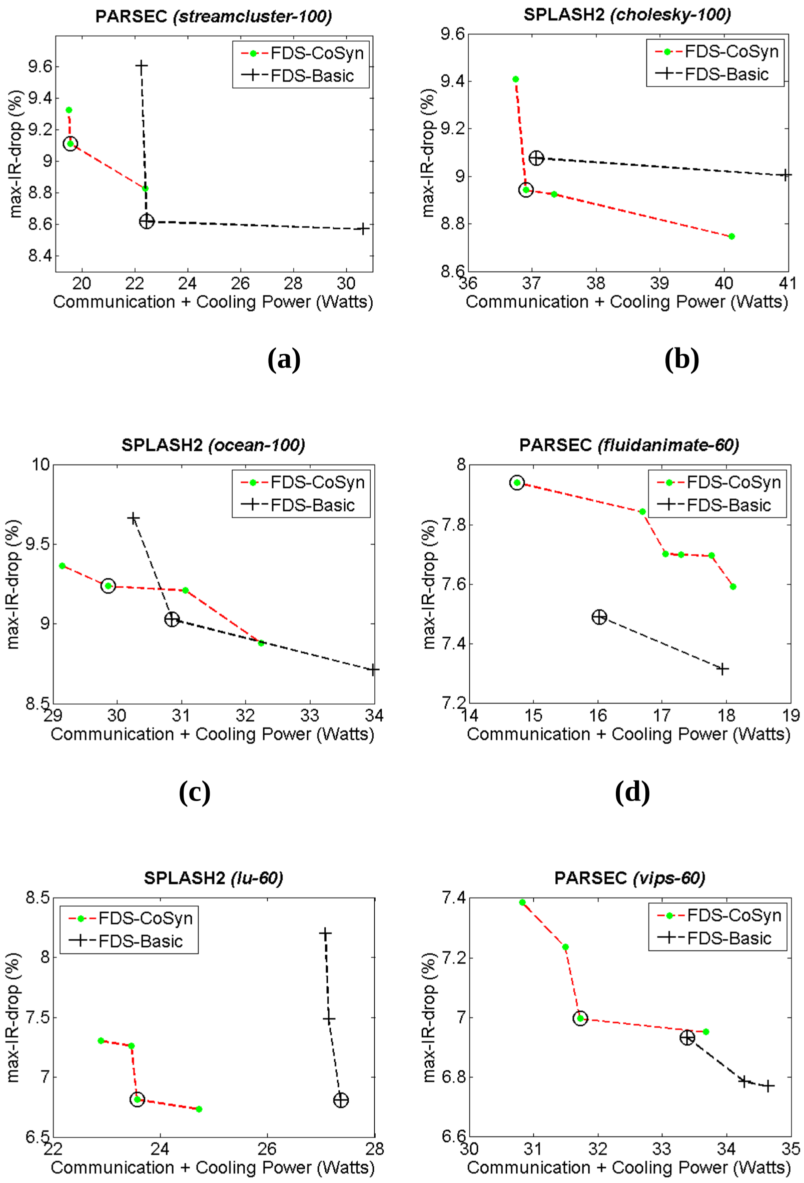
The solution-costs associated with the knee-point of each Pareto front in figure 21 for *SA-Basic* and *SA-PDN-unaware* are tabulated in Table 3. Each cell in the first two rows shows two numbers that represent communication cost (power) and PDN cost (max-IR-drop), respectively. The last row for each benchmark suite shows the percentage gross-improvement per benchmark obtained with *SA-Basic* over *SA-PDN-unaware.* This improvement ranges from 5.4% to 13.9%. *Thus, it can be observed from figure 21 and Table 3 that considering PDN-awareness not only results in feasible PDN solutions, but also solutions with better overall optimality.*

Table 3: Importance of PDN-awareness: % gross-improvements in terms of communication power and PDN cost for *SA-Basic* vs. *SA-PDN-unaware*

	streamcluster-100	*cholesky-100*	*ocean-100*	*fluidanimate-60*	*lu-60*	*vips-60*
SA-PDN-unaware	17.9,10.2	36.6,10.2	31.2,10.4	17.3,8.8	20.5,8.5	28.1,8.8
SA-Basic	18.5,9.3	36.6,9.1	32.1,9.5	15.6,8.9	21.3,7.7	28.6,7.4
% Gross Imp.	5.4	10.8	5.7	8.0	5.4	13.9

3.8.2.2 Importance of thermal-awareness

To show the significance of considering thermal-awareness in our co-synthesis framework, we use a thermal-unaware subset of our FDS-CoSyn framework (*FDS-Basic*). *FDS-Basic* has no temperature optimizing force-components (*FC_3* and *FC_4*), where we use the following coefficient values: *α=0.5, β_2=0, β_1=0,* and γ=0.5. Therefore, with *FDS-Basic*, communication power is somewhat better optimized compared to *FDS-CoSyn*, while on the other hand *FDS-CoSyn* trades-off communication power with cooling power for better overall (communication + cooling) power. The 2D solution space produced with *FDS-Basic* and *FDS-CoSyn* for six benchmarks is shown in Figure 22, with each candidate solution characterized by its (communication + cooling) power and the percentage max-IR-drop in the PDN. By optimizing cooling-power with a slight degradation in communication power, it can be seen from figure 22 that *FDS-CoSyn* produces solutions with better overall optimality compared to the thermal-unaware *FDS-Basic* framework.



(a)

(b)

(c)

(d)

(e)

(f)

Figure 22: Results for *FDS-Basic* (thermal-unaware) and *FDS-CoSyn* (thermal-aware) co-synthesis frameworks

The solution-costs associated with the knee-point of each Pareto front in Figure 22 for *FDS-Basic* and *FDS-CoSyn* are tabulated in Table 4. The power term in the solution-cost (defined in section 3.8.2.1) of each design point now includes the cooling power in addition to communication power (power = communication power + cooling power). Each entry in the second and third columns of the table corresponds to {communication + cooling} power and PDN cost. Compared to FDS-Basic, the thermal-awareness in FDS-CoSyn produces improvements in cooling power, by up to 57.8% (as shown in the fourth column of Table 4). As the communication power with FDS-CoSyn is generally slightly worse compared to FDS-Basic, lower improvements of up to 13.9% in (communication + cooling) power are obtained for FDS-CoSyn, as shown in the fifth column. Observe that benchmarks with the lowest communication-intensities (*streamcluster-100 and fluidanimate-60*) with a much weaker force-component for communication and a relatively strong thermal component in the FDS algorithm produce higher power-improvements with the thermal-aware FDS-CoSyn framework. Also, due to the stronger thermal-component for these benchmarks, max-IR-drop values obtained using FDS-CoSyn are slightly higher than that obtained using FDS-Basic (even with the same value of γ=0.5). Percentage gross-improvements of up to 13.7% (from the last column of Table 4) are obtained for *FDS-CoSyn* over *FDS-Basic*. *Thus, it can be observed from Figure 22 and Table 4 that considering thermal-awareness allows for a reduction in cooling power that translates into solutions with better overall optimality.*

Table 4: Importance of thermal-awareness: % gross-improvements in (communication+cooling) power and PDN cost for *FDS-Basic* vs. *FDS-CoSyn*

	FDS-Basic	FDS-CoSyn	%Imp.Cooling-power	%Imp. power	%Gross Imp.
streamcluster-100	{14.6+7.8= 22.5},8.6	{14.3+5.3= 19.6},9.1	32.1	12.9	7.1
cholesky-100	{31.0+6.1= 37.1},9.1	{32.3+4.6=36.9},8.9	24.6	0.4	1.9
ocean-100	{24.0+6.9= 30.9},9.0	{25.2+4.6= 29.8},9.2	33.3	3.2	0.9
fluidanimate-60	{12.6+3.5= 16.1},7.5	{12.5+2.2=14.8},7.9	37.1	8.0	2.0
lu-60	{19.1+8.3= 27.4},6.8	{20.1+3.5=23.6},6.8	57.8	13.9	13.7
vips-60	{28.0+5.4= 33.4},6.9	{27.8+3.9=31.7},7.0	27.7	5.0	4.1

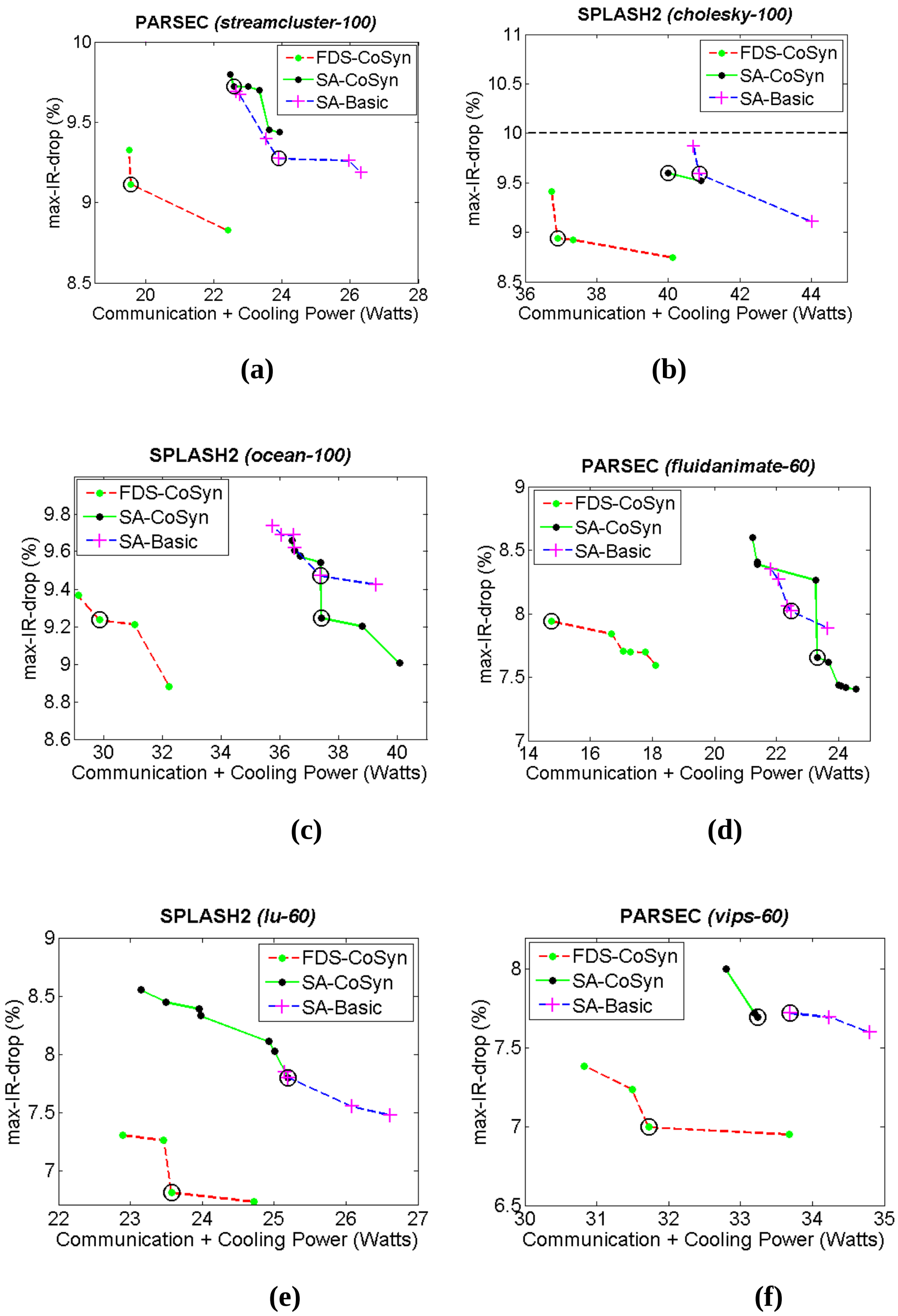


Figure 23: Results for *SA-Basic* (thermal-unaware), *SA-CoSyn*, and *FDS-CoSyn* co-synthesis frameworks

3.8.2.3 Algorithm comparison

To the authors' knowledge, this is the only work besides [67] that proposes a co-synthesis framework to co-optimize the 3D NoC fabric with the PDN fabric to produce a more efficient overall 3D MPSoC design. Therefore, in order to evaluate the efficacy of our proposed force-directed heuristic based framework (*FDS-CoSyn*) in comparison to the SA-based co-synthesis approach in [67] (*SA-Basic*), we perform a comparison study between these frameworks. We also compare these frameworks against the SA-based thermal-aware framework (*SA-CoSyn*) that is also proposed in this work and described in Section 3.7. The results of this study are summarized in Figure 23 and Table 5. Here, each candidate solution is characterized by its (communication + cooling) power and the percentage max-IR-drop in the PDN. Note that the solution points of *SA-Basic* are transferred from figure 21 to the plots of the same benchmark in Figure 23, with the added cooling power corresponding to individual solution points, and the new Pareto fronts thus formed are plotted in Figure 23. Note that our *PDN_solver()* and *3D_Routing()* in addition to *Thermal_eval()* are utilized to evaluate the max-IR-drop, communication power, and cooling power in all three framework implementations.

Table 5: Algorithm comparison: % gross-improvements in (communication + cooling) power and PDN cost for *FDS-CoSyn* over *SA-Basic* and *SA-CoSyn*

	stream-cluster-100	*cholesky-100*	*ocean-100*	*fluidanimate-60*	*lu-60*	*vips-60*
SA-Basic	23.9,9.3	40.9,9.6	37.4,9.5	22.5,8.0	25.2,7.8	33.7,7.7
SA-CoSyn	22.6,9.7	40.0,9.6	37.4,9.2	23.3,7.7	25.2,7.8	33.2,7.7
FDS-CoSyn	19.6,9.1	36.9,8.9	29.9,9.2	14.8,7.9	23.6,6.8	31.7,7.0
%Gross Imp. vs. SA-Basic	19.9	16.5	22.6	35.4	19.1	15.3
%Gross Imp. vs.SA-CoSyn	19.6	14.6	20.3	32.9	19.1	13.7

It can be observed from Figure 23 and Table 5 that our force-directed swapping algorithm based *FDS-CoSyn* framework finds excellent mapping solutions corresponding to the given initial mapping solutions. Therefore our approach of applying the swapping algorithm to numerous seeds (initial mappings) generates better overall solutions compared to an SA-based approach. Also, we observed that the SA algorithm spends considerable time searching for feasible mapping solutions meeting all hop-constraints and the PDN constraint, while FDS-CoSyn has an inherent mechanism for overcoming hop-constraint violations and the max-IR-drop optimizing force component (*FC_2*) ensures that the PDN constraint is always satisfied. Thus the force-directed swapping technique can optimize the design objectives more efficiently.

Our FDS-CoSyn framework produces solutions with much better overall optimality in terms of cooling costs, communication costs, as well as PDN costs, as shown in Figure 23. In Table 5, the last two rows show the percentage gross-improvements (per benchmark) obtained with *FDS-CoSyn* over *SA-Basic* and *SA-CoSyn*. *FDS-CoSyn produces percentage gross-improvements ranging from 13.7% to 32.9% over SA-CoSyn, and 15.3% to 35.4% over the thermal-unaware SA-Basic framework.*

As discussed earlier, the SA frameworks (*SA-Basic* and *SA-CoSyn*) execute until the solution quality fails to improve for 250 consecutive iterations. On the other hand, we observed that our FDS framework *FDS-CoSyn* runs for a maximum of 68 iterations. At each of these iterations, the thermal-unaware (Basic) frameworks execute *PDN_solver()* and *3D_Routing(),* whereas the thermal-aware (CoSyn) framework executes *Thermal_eval()* as well (although in *SA-CoSyn*, *3D_Routing()* and *Thermal_eval()* are performed only for mapping solutions that satisfy all hop-constraints as well as the PDN-constraint).

Table 6: Execution times (in seconds) of SA-CoSyn and FDS-CoSyn, and % average reduction in execution time with FDS-CoSyn over SA-CoSyn

	streamcluster-100	*ocean-100*	*cholesky-100*	*fluidanimate-60*	*lu-60*	*vips-60*
SA-Basic	19989	9392	12310	3195	3890	9100
SA-CoSyn	37210	26055	62835	20064	20834	8156
FDS-CoSyn	2934	2788	2704	1731	1761	1718

All the synthesis frameworks were simulated on a machine with Intel Core2Duo CPU running Linux OS. Table 6 shows a comparison of our FDS-CoSyn and SA-CoSyn frameworks, in terms of execution time in seconds. On average, the *FDS-CoSyn*, produces 12.9× and 4.2× reductions in execution time over *SA-CoSyn* and *SA-CoSyn*. This can be attributed to the following two factors (in addition to the reduced number of iterations for the FDS frameworks):

1) Unlike in *SA-CoSyn*, every mapping solution in *FDS-CoSyn* is optimized for the thermal profile; therefore, the intermediate mapping solutions generated in *SA-CoSyn* correspond to worse thermal profiles compared to the final candidate solutions produced by *FDS-CoSyn*. Therefore at each execution of *Thermal_eval()*, compared to *FDS-CoSyn*, *SA-CoSyn* requires on average more 3D-ICE runs of incrementing flow-rates, to satisfy the thermal constraint.

2) The FDS technique generally produces mapping solutions with shorter communication paths, thus requiring *3D_Routing()* to evaluate lesser number of candidate minimal paths on average.

3.8.2.4 Efficacy of routing path allocation

To show the effectiveness of our *VI*-aware NoC routing path allocation (discussed in section 3.6.4) in reducing power and latency overheads, we compare the results of our allocation approach with routing paths obtained by integrating traditional dimension-order routing schemes XYX and YXZ within the FDS-CoSyn framework. For NoC-latency calculations, we assume a 5 clock cycle latency for router-traversal, 1 cycle for link-traversal, 1 cycle for VLC-traversal [60], and 2-cycles for MCFIFO-traversal [53], [59]. As discussed earlier, we assume minimum BW constraints for sustaining communication-flows, and congestion is assumed to be negated by additional link-insertions. The latency of a particular communication-flow is defined as the time taken by a flit to traverse the path from source to destination. Due to the presence of *VIs*, different frequencies would be encountered on a single path. We sum-up the cycle-times along the path for all the flows and then calculate the average latency (across all flows).

Table 7 shows the NoC-power and average NoC latency results for the knee-points from the Pareto-front generated for the FDS-CoSyn framework with the corresponding routing scheme. The two numbers in every table cell indicate the NoC-power (in Watts) and average NoC-latency (in ns). Table 7 shows that power-savings of 6.2% and 6.3%, and latency savings of 2.4% and 2.5% are obtained on average, with our *VI*-aware routing over XYZ and YXZ routing schemes respectively, for the FDS-CoSyn framework. Although the savings in average latency may seem modest, savings of up to 35% (not tabulated) in the number of overhead clock-cycles (cycles spent for MCFIFOs/VLCs) are obtained, which highlights the effectiveness of our *VI*-aware-routing scheme in minimizing the latency-overheads of inter-island communication. These savings could potentially be much more pronounced in the presence of additional *VI*s (we assume three *VI*s in our experiments).

Table 7: Results for NoC power (Watts) and average-latency (ns) for FDS-CoSyn with different routing schemes: XYZ, YXZ, and our-*VI*-aware routing

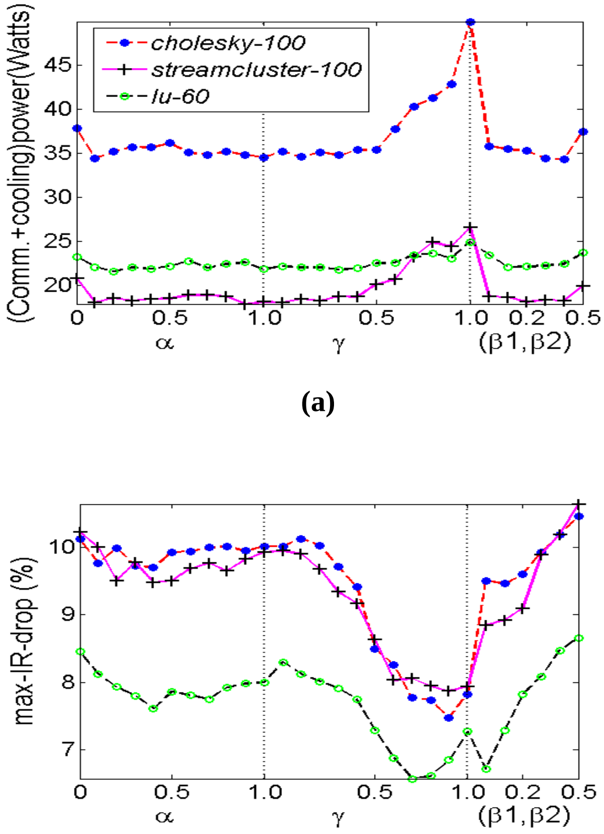
	Streamcluster-100	*ocean-100*	*cholesky-100*	*fluidanimate-60*	*lu-60*	*vips-60*
XYZ	16.8,20.3	26.3,19.8	33.0,17.8	13.9,20.0	21.8,17.0	29.5,19.8
YXZ	16.9,20.4	26.5,19.9	33.0,17.8	13.9,20.0	21.7,17.0	29.5,19.8
VI-aware	15.2,19.9	24.2,19.4	31.5,17.7	12.6,19.1	20.3,16.8	28.8,19.1

3.8.2.5 Parameter sensitivity analysis

In our experiments, we also perform a sensitivity analysis for the values of coefficients (α, β_2, β_1, and γ) of the four force-components in our *FDS-CoSyn* framework. For this purpose, we use three benchmarks: *streamcluster-100, lu-60, and cholesky-100* with low, medium, and high communication intensities respectively. The impact on solution-costs of different combinations of coefficients is shown in figure 24.

In our analysis, we sweep α and γ from 0 to 1.0, shown in the left-most and center vertical partitions respectively of figures 13(a) and 13(b). To capture the combined effect of the two temperature optimizing coefficients β_2 and β_1, we sweep them in tandem (shown in the right-most vertical partitions). Note that the coefficients that are not being swept, e.g. β_2, β_1, and γ in the left-most vertical partition of figure 24, are all set to equal values such that ($\alpha+\beta_2+\beta_1+\gamma$)=1. Also note that PDN-constraint is ignored in these analyses in order to investigate the full range of effects of changing coefficients.

As *N (66 or 68)* solution points are generated for each combination of coefficients, each solution point in figure 24 represents the solution (one out of 66 or 68) with the corresponding minimum cost. Notice in figure 24(a) that relatively low values of β_1, β_2, and α (around 0.2) are sufficient to produce well optimized values of (communication + cooling) power. Also, from figure 24(b), it can be observed that high values of γ (γ>0.7) result in the biggest optimization in max-IR-drop. Interestingly, for values of γ in excess of 0.5 the power generally increases sharply (as shown in figure 24(a)). Moreover, max-IR-drop increases rapidly with increasing values of β_1 and β_2. We use this knowledge while deciding on the coefficient values in our implementation of the *FDS-CoSyn* framework. Driven by the sensitivity analysis, we chose values of 0.3, 0.1, 0.1, and 0.5 for α, β_1, β_2, and γ respectively, so as to efficiently trade-off power costs with PDN costs for better overall optimization in *FDS-CoSyn*.



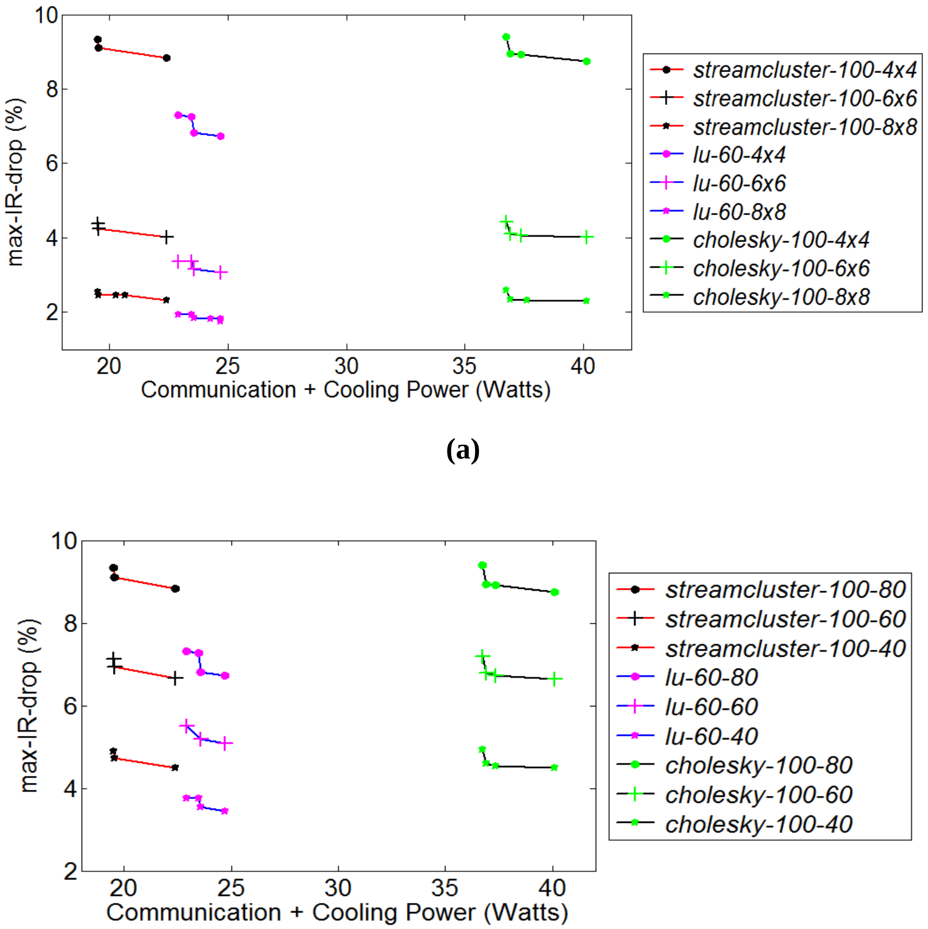
(a)

(b)

Figure 24: Results of sensitivity analysis (w.r.t. α, β_2, β_1, and γ) using FDS-CoSyn: (a) in terms (communication + cooling) power; (b) in terms of max-IR-drop. Coefficients α and γ are swept from 0 to 1.0, whereas β_2 and β_1 are varied simultaneously from 0 to 0.5; in all cases, equal values of other coefficients are used such that the sum of α, β_2, β_1, and γ always equals unity.

3.8.2.6 *Sensitivity analysis of power-grid granularity and resistance*

In our final set of experiments, we present an analysis of the sensitivity of results for FDS-CoSyn with: *(i)* varying density of power-TSVs/power pins (grid-node granularity) of the 3D regular power-grid for fixed TSV-resistance of 80mΩ (as shown in figure 25(a)), *(ii)* varying TSV-resistance values (representing varying TSV widths) for fixed granularity of 4x4 grid-nodes per core (as shown in figure 12(b)). Here, we use the same set of benchmarks that are employed in section 3.8.2.5.



(a)

(b)

Figure 25: Results of sensitivity analysis of the power-grid (w.r.t spacing and width of TSVs) using FDS-CoSyn. For each benchmark, the max-IR-drop decreases when: (a) number of 3D PDN grid-nodes per core increase from 16 (4x4) to 36 (6x6) and 64 (8x8); (b) TSV-resistance values decrease from 80mΩ to 60mΩ and 40mΩ.

Figure 25 shows that increasing power-pin/power-TSVs as well as decreasing TSV-resistances, result in decreasing max-IR-drop values. Also, observe that increasing the width of TSV-wires has a similar effect on IR-drops as reducing the PDN-pitch. However, increasing the number of power-pin/power-TSVs is only possible if chip-design constraints on the number of external power pins are satisfied. Moreover, increasing power-pin/power-TSVs or TSV-widths result in the following challenges: *(i)* reduction in process yield with increased TSV-density; and *(ii)* die area overheads. Therefore, denser (in terms of TSV-granularity or width) power grids could be employed if lesser IR-drops are desired, at the cost of the above mentioned challenges.

3.9 Conclusion

This chapter proposes an automated system-level framework for thermal-aware co-synthesis of PDN design and NoC design in 3D MPSoCs. Our co-synthesis framework (FDS-CoSyn) uses a novel force-directed swapping algorithm for generating better overall core-to-tile mapping solutions while efficiently trading-off among communication, thermal, and PDN design goals. Additionally, we propose an efficient routing algorithm that minimizes total NoC power. Our experimental results show that the proposed co-synthesis framework not only generates solutions with viable PDN designs unlike traditional approaches, but also generates better design solutions (by up to 32.9% and 35.4%) much faster (by up to 12.9× and 4.2×) when compared to simulated-annealing based thermal-aware and thermal-unaware co-optimization approaches, respectively. Therefore, our co-synthesis approach can potentially ease design effort and improve time-to-market of emerging 3D MPSoC designs.

We note that a routing methodology where long-range links are selectively inserted between cores to improve overall network latency (as suggested in [30], [57]), could also be used within our FDS frameworks. Such an approach would possibly require careful considerations of the increase in complexity of design and physical implementation due to issues such as repeater-overheads, asymmetric interconnect lengths, and more complex wire-routing.

4 PRATHAM: A Power Delivery-Aware and Thermal-Aware Mapping Framework for Parallel Embedded Applications on 3D MPSoCs

In emerging 3D-ICs, thermal hotspots and high IR-drops in the power delivery network (PDN) can significantly limit overall system performance. The high core counts required to support parallel embedded applications in these 3D-ICs also notably increases communication energy. As inter-core communication patterns, IR-drop distributions, and 3D thermal profiles all influence system performance and power, it is critical that a system-level application mapping framework consider their combined effect. In this chapter, for the first time, we propose a system-level embedded application mapping framework (PRATHAM) which integrates a holistic solution evaluation methodology that considers the impact of network-on-chip (NoC) communication, drops in supply voltage, and the on-chip thermal profile on overall system performance, and co-optimizes on-chip IR-drop, thermal, and communication profiles, for improved overall system performance and lower energy consumption.

4.1 Introduction

In emerging 3D multiprocessor system-on-chip (MPSoC) architectures, network-on-chip (NoC) fabrics enable tens to hundreds of cores to communicate with each other and memory at the intra- and inter-layer levels. NoCs with mesh-based (regular) topologies have been employed in many recent MPSoC designs, e.g. Tilera's TILE64 [5] and Intel's Single Chip Cloud Computer (SCC) [3]. But as the power dissipated in the NoC is a significant portion of the total on-chip power in these designs, optimizing communication power, in addition to computation power, has become a critical step in design methodologies for 3D MPSoCs.

Yet another critical component in 3D MPSoCs is the power delivery network (PDN), which is required to deliver a stable power supply across the chip that is within a desired voltage range and can tolerate large variations in load currents [100]. However, with increasing on-chip device densities and decreasing voltage levels, supply currents have risen but the scaling of PDN impedance has not kept up with this trend [135]. This has led to worsening IR-drops in the PDN. As circuit delay in modern CMOS technologies has a super-linear relationship to the supply voltage drop [6], high IR-drops in the PDN have increased circuit delay, limiting operating frequencies of cores and thus reducing MPSoC performance. This problem is even more severe in 3D MPSoCs, as the number of I/O pins on an *n*-layered 3D-IC is about *n* times smaller than its 2D counter-part, thus exacerbating the problem of a degraded voltage supply in 3D MPSoCs [7].

With much higher power densities in high-performance 3D MPSoCs, another challenge is to remove the heat generated in the 3D stacked dies [72]. This problem is exacerbated due to the increased thermal coupling between adjacent cores in advanced technologies [76]. In addition, the circuit delay is also strongly related to temperature. Increasing temperatures can limit the operating frequency on the chip, thereby degrading system performance [77], [78]. It is also commonly known that leakage power has a super-linear relationship with temperature and that temperature in turn depends on the power profile of the chip.

As a result of these close interactions between the PDN, die temperature, circuit performance, and power dissipation, performing thermal analysis (or PDN analysis) in isolation, without considering the influence of temperature (or supply voltage) on power and performance could lead to a grossly inaccurate estimation of system metrics. Prior works [79]-[81] have motivated a joint thermal-power-performance-PDN analysis to capture the inter-dependencies between the various design metrics. However, to the authors' knowledge, none of these prior works have considered such co-analysis for accurately evaluating the merit of any given design solution as part of an MPSoC design-time synthesis framework for 2D or 3D ICs.

This work is motivated by the interesting insight that inter-core communication patterns, IR-drop distribution in the PDN, as well as the on-chip thermal profile can vary significantly with different mapping configurations of computation and communication resources on an MPSoC. We propose an *automated system-level design-time parallel embedded application mapping framework (PRATHAM) for 3D MPSoCs that considers the impact of NoC communication, drops in supply voltage, and chip thermal profile on overall system performance.* PRATHAM co-optimizes on-chip IR-drop, thermal, and communication profiles for improved system performance and lower energy consumption. The novel contributions of our framework are summarized as follows:-

- We propose a design-time CAD/synthesis approach to map cores running parallel embedded applications to the tiles on a 3D die while co-optimizing communication, PDN design, and thermal objectives to generate an optimized 3D MPSoC design;
- We integrate the effects of temperature and supply voltage drops in the performance and energy modeling of the compute cores as well as the communication resources; and show that considering such awareness in the solution evaluation framework results in significantly improved design solutions;
- We design an algorithm for 3D routing path allocation that minimizes path-latencies and energy in the 3D NoC fabric.

4.2 Related Work

Given the promise of 3D IC technology and MPSoCs, the problem of design-time mapping of tasks and communication resources onto 3D die stacks has attracted significant research efforts, e.g., [27], [62], [66], [67], [84]. The authors in [62] propose a tool for designing irregular 3D NoCs that finds paths for communication flows and places network components on the 3D layers; while reporting savings in NoC power and delay. In [66], improvements are shown over [62] by additionally considering the impact of TSV serialization and network interface placement in application specific 3D NoCs. The authors in [84] perform thermal-aware assignment of tasks onto a 3D regular mesh followed by voltage-frequency planning to minimize power dissipation and meet application performance constraints. In [27] a floorplan-aware synthesis algorithm is presented for application-specific 3D NoC synthesis based on simulated allocation; generating topologies to optimize chip power, network latency, and temperature.

The authors in [67] motivate the need for considering PDN awareness in the design-time application mapping stage, and propose a co-synthesis framework for 3D MPSoCs, which is shown to produce more efficient solutions compared to a PDN-unaware synthesis approach. But the proposed approach fails to consider the adverse effects of IR-drops on the system performance and energy dissipation. Also, they do not consider the thermal costs associated with the generated solutions. In our work, *we propose a novel MPSoC synthesis framework to significantly improve overall system performance and energy dissipation during embedded application computation and communication mapping, while for the first time capturing the effects of IR-drops and temperature on the system profile with a holistic solution evaluation approach.*

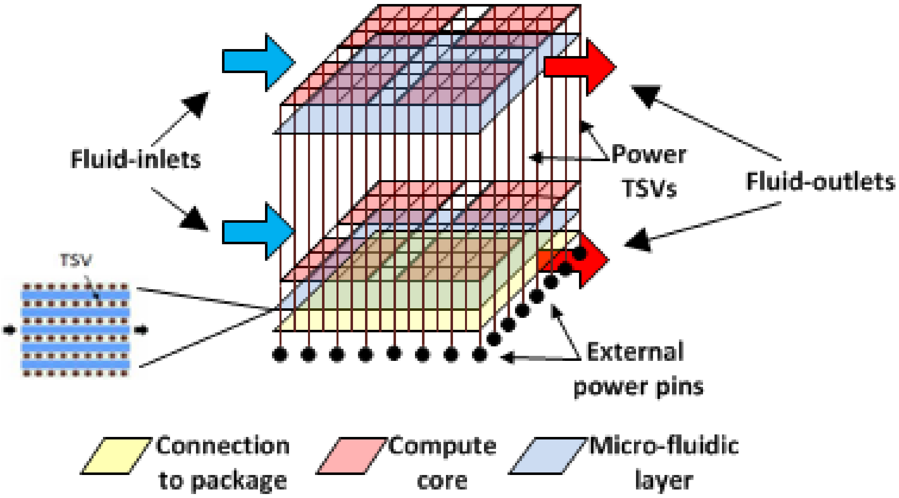


Figure 26: Example of a 3D package for 8-core MPSoC (2×2×2 3D-mesh) with: *(i)* a regular 3D power-grid with 64 external power-pins and 64 grid-points per tier (16 grid-points supplying to each core); *(ii)* a micro-fluidic cooling layer under each device layer

4.3 Background: PDN and Liquid-cooling

In this work, we consider a 3D MPSoC platform with a 3D mesh of tiles, where there is a one-to-one mapping of processing cores to these tiles. We assume a 3D package with compute cores, PDN, and micro-fluidic cooling, as shown in figure 26.

It is shown in [69] that sizing of pitches and widths of the power grid does not change the on-chip voltage distribution. Therefore in our work we assume fixed uniform power grids as in prior works [67], [69], [70]. As we primarily investigate steady state effects of the global power grid, time-varying network characteristics such as transient noise are not considered.

Our use of micro-fluidic cooling is driven by the observation that conventional air-cooled heat sinks are widely believed to be insufficient to remove the heat generated in multiple stacked tiers of high-performance 3D MPSoCs [72]. In [83], it was demonstrated that liquid cooling can provide much more efficient heat removal than conventional heat-sink based cooling solutions. We use a micro-fluidic layer under each device layer (as shown in figure 26) as suggested in [43]. The authors in [74] have shown that as the micro-fluid within micro-channels progresses from the fluid inlet towards the fluid outlet (from left to right in figure 26), its capacity to absorb heat flux from device layers steadily decreases.

Among its many contributions, the proposed PRATHAM synthesis framework brings to light the importance of employing an accurate solution-evaluation methodology that captures the co-dependence between various design metrics: power dissipation, performance, supply voltage, and temperature. To this end, we now present a simple motivational example. Figure 27(a) shows the rated power-profile of a 27-core 3D MPSoC for a given core-mapping, corresponding to nominal values of temperature, supply voltage, and operating frequency. Traditional design-time MPSoC CAD/synthesis frameworks commonly assume such a system-profile, where the adverse effects of IR-drops in the PDN and temperature are ignored. In reality, the operating frequency at which a core can be reliably clocked at depends upon both the operating temperature and voltage supply level. Also, power dissipation is intricately related to voltage, frequency, and temperature: *(i)* dynamic power linearly increases with voltage and frequency; and *(ii)* the super-linear inter-dependence of leakage power and temperature is well known. Assuming that external power pins of the PDN are closest to the bottom tier and micro-fluid flows from left to right (as depicted in figure 26), after considering the combined effects of IR-drops and temperature on system performance and power, the resulting system-profile is shown in figure 27(b) (linear-programming based

PDN-analysis and thermal-analysis using the 3D-ICE tool [71] are performed iteratively to obtain this profile).

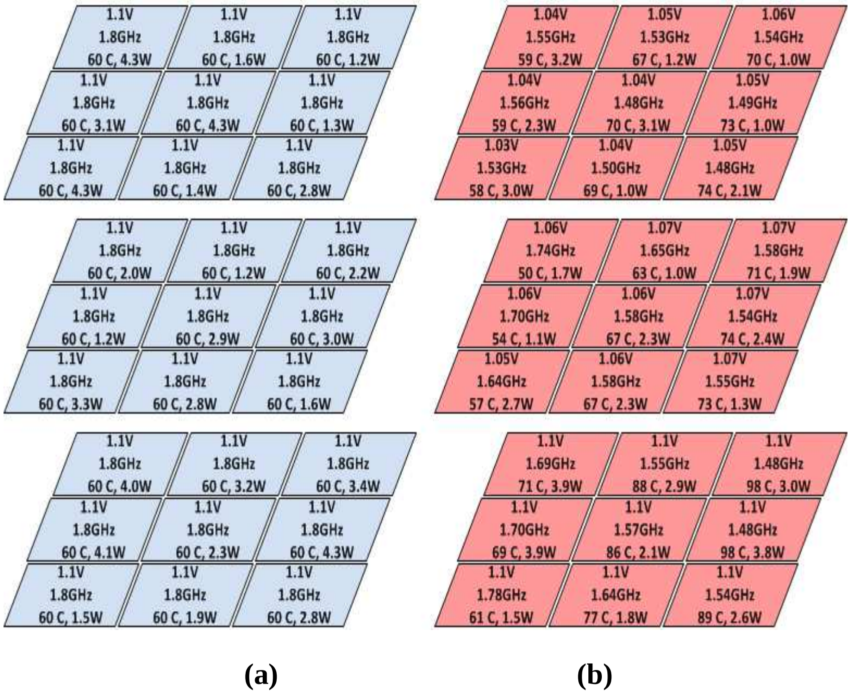


Figure 27: Example of a 27-core 3D MPSoC (3×3×3 mesh): (a) rated power-profile with nominal values of supply voltage (1.1V), operating frequency (1.8GHz), and temperature (60°C); (b) actual system-profile considering the combined effects of IR-drops and temperature on power and performance of cores. A 3D power-grid and micro-fluidic cooling are assumed (not shown) as in figure 26

Note how actual performance and power dissipation of various cores differs significantly in figure 27(b) where we have considered the impact of IR-drops and temperature on system power and performance, compared to the idealized scenario in figure 27(a) without any such considerations. The example highlights how the assumptions made in existing mapping frameworks (represented by figure 27(a)) ignore important facets of the design space influenced by IR-drops and temperature. As a result, *the inherent inaccuracies in these existing frameworks may result in the discarding of good solutions during design space pruning* and the selection of seemingly optimal solutions that are eventually found to be highly inefficient once the PDN model and thermal profiles are considered. By capturing a more holistic system view (as in figure 27(b)), MPSoC synthesis frameworks can traverse the solution search-space much more intelligently, resulting in final synthesized solutions with better overall optimality.

4.4 Problem Formulation

In this section we present our problem formulation. We assume the following inputs to our problem:

- A 3D-IC with a regular 3D mesh NoC, with dimensions (dim_x, dim_y, dim_z) and number of tiles T = $dim_x \times dim_y \times dim_z$ with each tile containing a compute core and a NoC router;
- A core graph $G(T,E)$ (corresponding to the given set of parallel embedded applications) with a set of T vertices representing homogenous compute cores on which application tasks have already been mapped, and a set of E edges that represent communication volumes between communicating cores;
- A triplet $\{V_{nom}, f_{nom}, t_{nom}\}$ constituting the nominal values of supply voltage, operating frequency, and temperature, representing nominal operating condition for all T cores;
- A set of nominal supply current values $\{I_{1\_nom}, I_{2\_nom},...,I_{T\_nom}\}$ for the T cores corresponding to nominal operating conditions;
- A set of T nominal computation-times $\{CT_{1\_nom}, CT_{2\_nom}, ..., CT_{T\_nom}\}$ corresponding to workloads executed on each core;
- A regular 3D power grid, with $n \times n$ grid-points supplying to each core, and a chip-wide maximum IR-drop constraint Γ in the PDN (relative to V_{nom});
- A fixed liquid flow-rate Ω in the micro-fluid cooling setup, with a chip-wide maximum temperature constraint Ψ.

Problem Objective: Given the above inputs, the goal of our proposed framework is to obtain a core-to-die mapping on the 3D-IC and a mapping of communication flows to a regular 3D mesh NoC for a given set of parallel embedded applications, such that energy-delay-squared product (ED^{2}P) of the system is minimized, while the design constraints Ψ and Γ are satisfied. In our ED^{2}P calculations, we define system-energy as (computation + communication) energy dissipation in the 3D die, and system-delay as the aggregate of computation and communication times for all cores. As performance is considered the principal design metric in most high-performance chips, we choose system ED^{2}P as the optimization metric, as motivated in prior work [85], with an upper bound on the peak temperature of the chip.

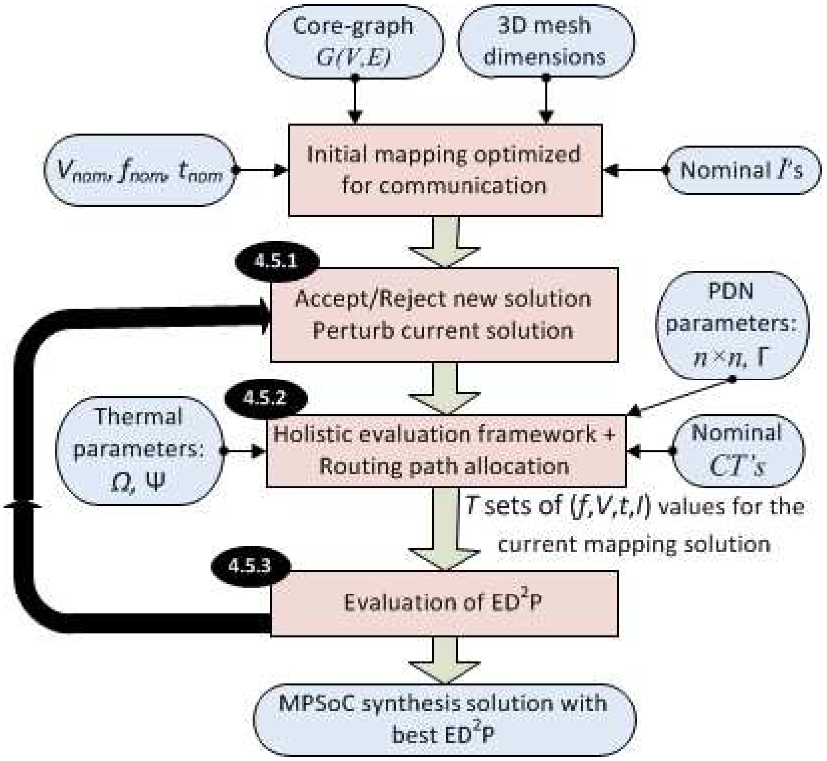

Core-graph $G(V,E)$
3D mesh dimensions
V_{nom}, f_{nom}, t_{nom}
Initial mapping optimized for communication
Nominal I's
4.5.1
Accept/Reject new solution Perturb current solution
PDN parameters: $n \times n$, Γ
4.5.2
Thermal parameters: Ω, Ψ
Holistic evaluation framework + Routing path allocation
Nominal CT's
T sets of (f,V,t,I) values for the current mapping solution
4.5.3
Evaluation of ED^{2}P
MPSoC synthesis solution with best ED^{2}P

Figure 28: Design flow of the PRATHAM synthesis framework

4.5 PRATHAM Framework: Overview

We first present a brief overview of our 3D MPSoC synthesis framework, before describing more details later in this section. The PRATHAM framework, shown in Figure 28, uses a simulated annealing (SA) based search algorithm (discussed in section 4.5.1), where the minimization objective is the system ED^{2}P. The initial mapping solution, which is optimized for communication traffic, is generated by an efficient incremental core mapping approach based on [24]. A holistic solution evaluation stage and a routing path allocation heuristic are integrated within the SA-based search algorithm. The holistic evaluation stage computes the set of T quadruplets $\{(f_1,V_1,t_1,I_1), (f_2,V_2,t_2,I_2), \ldots, (f_T,V_T,t_T,I_T)\}$ for each new mapping solution (discussed in section 4.5.2). In this manner, computation times ($CT_1,CT_2,\ldots,CT_T$) and computation energies of cores can be readily evaluated for any given mapping solution. The routing path allocation heuristic, which optimizes path latencies of communication flows, is then integrated (discussed in section 4.5.3) to produce a complete synthesis solution for the current mapping solution. Finally, the framework generates the best MPSoC synthesis solution with the least ED^{2}P. The next several subsections discuss the framework in more detail.

4.5.1 SA-based Search Algorithm

Broadly speaking, the optimization objectives of our 3D MPSoC synthesis framework are three-fold: (*i*) minimize application communication latencies and energy (by mapping cores such that the high-volume communication flows have shorter path-lengths in the NoC); (*ii*) minimize the IR-drops in the PDN (by mapping cores with higher supply current requirements to tiles on the 3D mesh closer to the external input power pins), and (*iii*) facilitate more efficient cooling of the 3D chip (by mapping cores with higher power dissipation closer to the water inlet of the microfluidic-cooling system, while at the same time maintaining a reasonably uniform power profile across all the device layers).

The single minimization objective of our SA-based search, system ED^{2}P, is representative of the combination of the above objectives. In the SA search process, the solution is perturbed by swapping two arbitrarily chosen cores on the 3D mesh. As in a traditional SA-based search, the best synthesis solution found is saved, and any newly generated solution is probabilistically accepted (or rejected) based on its solution-cost as well as the current SA-temperature. In addition to the ED^{2}P value, the number of violations (at per-core granularity) of the PDN constraint Γ and the thermal constraint Ψ are also considered for the cost-evaluation of the current solution. The SA cost function used by the search is defined as:

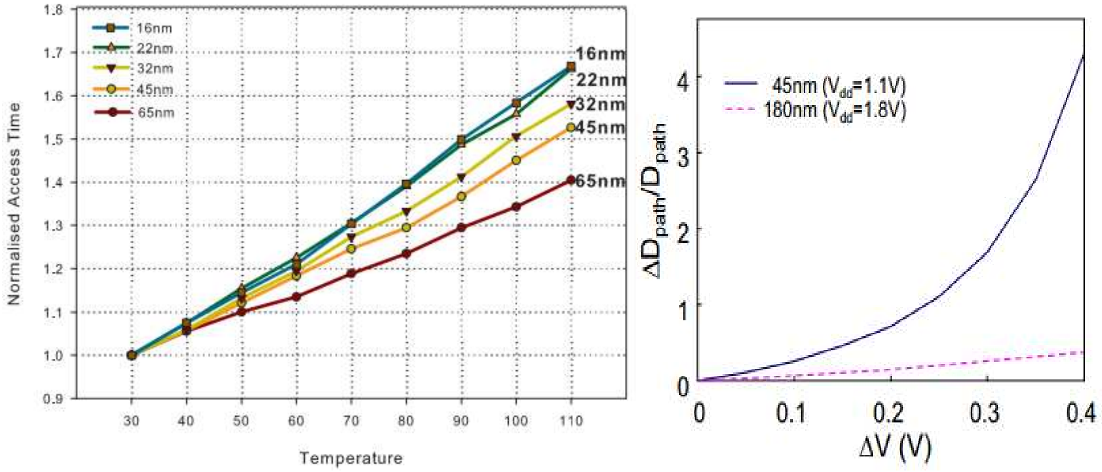
$$cost = ED^2P + c \times \{num.\ of\ (PDN\text{-}violations + thermal\text{-}violations)\}$$

where, *c* is the violation-penalty coefficient with a high integer value. The search phase terminates either when no new best solution is found for *L* number of consecutive SA-iterations or a pre-specified number of iterations have been executed.

4.5.2 Holistic Solution Evaluation Framework

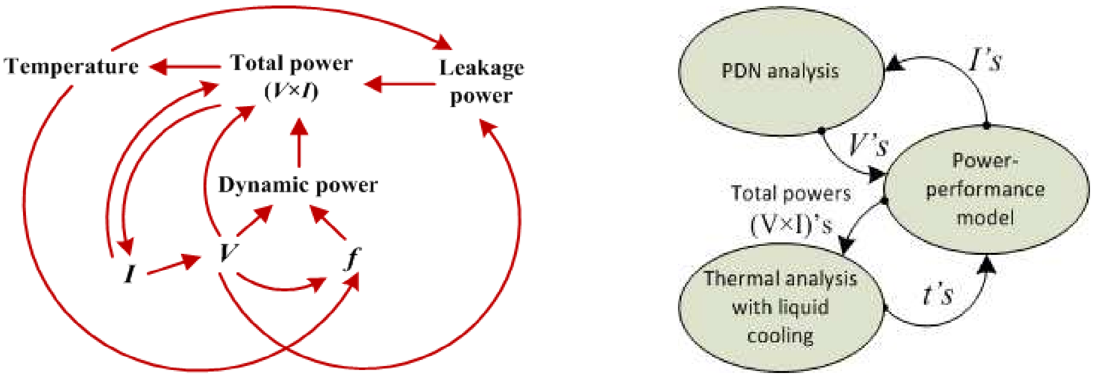
In this section, we first explain the various dependencies between different design metrics at the system-level. Later in the section, we discuss implementation details of our holistic evaluation framework, and the necessity of taking into consideration these inter-dependencies during solution evaluation.

It is well known that leakage power depends exponentially on temperature. At the same time, temperature directly depends on the power-profile of the system. Increasing temperatures contribute to significant circuit slow-down; and this degradation in performance worsens with technology scaling, as shown in figure 29(a). Additionally, circuit delay is also related to the supply voltage, as shown in figure 29(b). Supply voltage in turn depends on the IR-drop distribution in the PDN. Therefore, *the maximum frequency (f) that a core can be clocked at depends on both the supply voltage and the operating temperature of the core.* Moreover, the operating core frequency, along with the supply voltage, determines the total power dissipation *(V×I)*, which in turn determines the supply current requirement of the given core. Finally, the supply currents flowing in the PDN determine the IR-drop distribution in the PDN. This inter-dependence among power, thermal, performance, and PDN metrics is depicted pictorially with a metric dependency graph in figure 30(a).



(a) (b)

Figure 29: Performance degradation: (a) in terms of cache access time, with increasing circuit temperatures [77]; (b) in terms of circuit path-delay, with increasing drop in supply voltage [6]



(a) (b)

Figure 30: Holistic solution evaluation framework (a) metric dependency graph (b) control-flow of the evaluation framework

Our evaluation framework is applied to each new mapping candidate generated in the SA-based search. To capture the cyclic inter-dependencies among different design metrics as shown in figure 30(a), our holistic solution evaluation framework is executed iteratively until convergence to final values. Figure 30(b) shows a graphical representation (control-flow graph) of the three phases in our framework that are iteratively executed. The details of the three phases (*i*) Power-performance model (*ii*) Thermal-analysis (*iii*) PDN-analysis are discussed below:

Power-performance model: The impact of IR-drops in the PDN and chip thermal profile on performance and power dissipation is taken into consideration by the power-performance model. In each iteration of execution, updated values of core-temperatures (*t*'s) and supply voltages (*V*'s) are first utilized to update the core-frequencies (*f*'s). The *f*'s in turn are utilized (along with *V*'s) to compute the dynamic power of cores. The *t*'s and the *V*'s are used to compute the leakage powers of cores. The individual sums of dynamic and leakage powers result in total powers of the *T* cores, which are finally used to update the supply currents (*I'*s). The power-performance modeling is based on [81] (with parameters being technology-scaled based on [6] and [77]).

Thermal-analysis: To perform thermal evaluation of any given mapping solution in our framework, we utilize the open-source thermal emulator 3D-ICE 2.2.5 [71] which supports steady-state thermal analysis of 3D ICs with inter-layer liquid cooling. For the given power-profile, the tool outputs the core-temperatures (*t*'s) on the 3D die, for a fixed micro-fluid flow-rate of *Ω*.

PDN-analysis: For any given candidate mapping solution (with the supply current requirements of the *T* cores on 3D mesh), we use a linear programming (LP) based formulation to solve for the grid-point voltages and currents flowing in the 3D regular power grid. We validated the LP formulation using HSPICE simulations and used the lp_solve solver [75] to obtain results. In our LP formulation, we assume a 2D grid of $n \times n$ grid-points supplying to each core on the 3D die (as illustrated in figure 26). This phase in the evaluation framework essentially evaluates the IR-drops in the 3D resistive grid, which enables the updating of supply voltages of cores in the 3D MPSoC. Details of the design variables and constraints of our LP formulation are presented in APPENDIX I.

Algorithm 4.1 presents the pseudo-code for our holistic evaluation framework. The three phases of the framework: PDN analysis, Power-performance model, and Thermal analysis are iteratively executed

(steps 2-5) until the assumed *convergence* condition is met. When all of core-temperatures (*t'*s) change by less than 0.5°C in successive iterations of thermal analysis, we say that *convergence* is achieved **(step 6)**, and end the iterative evaluation process. We empirically chose the value of 0.5°C as a reasonable trade-off between accuracy and execution-time. Once temperature convergence is achieved, the power-profile can be safely assumed to have stabilized, and thus *V's* and *I's* of cores are also assumed to have converged to their final values. Finally, the updated (converged) values of temperatures, supply currents, operating frequencies, and supply voltages for all *T* cores of the current mapping solution are obtained. Using this information, computation times (CT_1, CT_2, …, CT_T) and computation energies of cores are evaluated for the given mapping solution **(steps 8, 9)**. For the infeasible mapping solutions that do not satisfy the chip-wide peak temperature constraint Ψ or the maximum IR-drop constraint Γ for all cores, the total number of PDN and thermal violations are recorded **(step 10)**.

Algorithm 4.1:Iterative holistic solution evaluation framework

Inputs: mapping-solution, nominal CT's, and Ω, Ψ, n×n, Γ

```
1: do {
2: PDN analysis computes $V$'s for the current set of $I$'s
3: Power-performance model updates $f$'s, dynamic and leakage power values, and $I$'s, for the
   updated set of $V$'s
4: Thermal analysis computes the $t$'s for the current set of power values
5: Power-performance model now updates $f$'s, power values, and $I$'s for the updated set of $t$'s
6: exit do loop if convergence achieved
7: go to step 2 }
8: update computation-times of cores based on the scaled (updated) $f$'s
9: calculate core computation-energies using computation-times and power $(V \times I)$ values
10: output the num. of violations if $\Psi$ or $\Gamma$ constraints have been violated
```

outputs:{ computation-energies, computation-times, updated (f,V,t,I)'s of T cores} and number of violations

One of the main contributions of this work is to highlight the significance of considering these dependencies during design-time 3D MPSoC synthesis. A holistic evaluation approach as discussed in this section is inherently more accurate, but more importantly, when integrated within the synthesis framework, it produces significantly improved design solutions by facilitating the solution search space to be explored more intelligently (corroborated with experimental results in section 4.6). The updated values of operating frequencies and voltages from this stage are utilized by the routing path allocation step, discussed in the next section.

4.5.3 3D Routing Path Allocation

The routing stage assigns routing paths to all inter-core communication flows in the application core-graph. The goal of this step is to minimize communication path-latencies and communication energy in the 3D NoC. Algorithm 4.2 (shown below) summarizes our routing path allocation algorithm.

Algorithm 4.2. Routing Path Allocation

```
input: core graph G(T, E), candidate-mapping-solution

1: Sort all communication flows in the increasing order of path lengths
   and in decreasing order of volumes for the same path length
2: for each communication flow in sorted list do {
3: Out of all the candidate minimal paths, choose a path that
   has the smallest congestion bottleneck
4: Allocate the current flow bandwidth over the chosen routing path }

output :Synthesized NoC with all communication flows routed
```

The order in which communication flows are routed is determined in the following manner. The flows with longer minimal paths (Manhattan distances) have more choices for routing and thus have a larger scope for optimization. Also, flows with smaller communication volumes require lesser link-bandwidths. Therefore, flows are sorted in the increasing order of their path lengths, in decreasing order of their communication volumes for the same path length; and considered for routing in that order (**step 1**). We consider all possible minimal paths for each flow and select the path with the smallest congestion-bottleneck, i.e., the path with lowest maximum communication volume assigned on any link along the path (**step 3**). Finally, the current communication flow is allocated to the path **(step 4)**.

4.5.4 ED^{2}P Evaluation

Once all the communication flows have been routed, the aggregate communication power dissipation in the NoC is computed taking into consideration the link loads, router sizes, and corresponding voltage/frequency values. Note that as each core operates at its own frequency (each router operates at the same frequency/voltage as the core it is associated with), the communication link operates at the lesser frequency of the pair of routers it connects. The power and latency overheads of MCFIFO based frequency converters (that are required to ensure correctness when crossing frequency domains) are included in our analysis. Path latencies are computed for each flow and used to determine the total communication time for each core. We define the core-time of the i^{th} core with q number of outgoing communication-flows as: *core-time*$_i$ = *computation-time*$_i$ + *communication-time1*$_i$ + *communication-time2*$_i$ + ... + *communication-timeq*$_i$. The average of all *core-times* is termed as *system-delay*, which is finally used to evaluate the ED^{2}P of the synthesis solution: ED^{2}P = (communication + computation) energy × (*system-delay)*^{2}. After the completion of this step, a mapped and routed 3D MPSoC is obtained for the given set of applications.

4.6 Experimental Studies

4.6.1 Experimental Setup

In our experiments, ARM Cortex A-9 processors [37] are used as the baseline MPSoC compute cores, at 32-nm technology node. For the nominal operating condition, values of V_{nom} = 1.1V, f_{nom} = 1800MHz, and t_{nom} = 45°C are assumed. We assume leakage power to be 30% of total power under nominal operating conditions, based on our analysis at 32nm. We consider 64-core and 144-core 3D MPSoC platforms, with dimensions 4×4×4 and 6×6×4 (dim_x×dim_y×dim_z). The cores are inter-connected using a 3D mesh NoC topology. Under nominal operating conditions, the maximum supply current values for the 64-core MPSoC, are assumed to be between 1A and 4A, based on the computation requirements of tasks assigned to the respective cores. As we assume both the 64-core and 144-core MPSoCs to roughly have the same area footprint, the maximum supply current values for the 144-core MPSoC are scaled to the 0.4A to 1.7A range.

We use eight parallel application benchmarks from the SPLASH-2 and PARSEC benchmark suites [58], [56]: *fft, lu, cholesky, radix, vips, dedup, streamcluster, and blackscholes.* Our core-graphs are modeled based on inter-core communication characterizations given in [47]. In order to evaluate our synthesis framework for varied combinations of parallel benchmarks, we assume concurrent execution of multiple benchmarks on the 3D MPSoCs. We consider four combinations of compute-intensive and communication-intensive benchmarks implemented on 64-core and 144-core platforms for a total of eight workloads. We assume equal number of threads for all concurrently executing benchmarks. The eight different workloads considered are: wld-1A and wld-1B: combination of compute-intensive and communication-intensive PARSEC benchmarks {*vips* + *dedup* + *streamcluster* + *blackscholes*}; wld-2A and wld-2B: combination of compute-intensive and communication intensive SPLASH2 benchmarks *{fft* + *lu* + *cholesky* + *radix};* wld-3A and wld-3B: combination of compute-intensive benchmarks *{streamcluster* + *blackscholes* + *cholesky* + *radix};* wld-4A and wld-4B: combination of communication-intensive benchmarks {*vips* + *dedup* + *fft* + *lu*}. The workloads with suffix “A” are executed on the 64-core MPSoC platform and those with suffix “B” are executed on the 144-core MPSoC platform.

In the SA-based search, an initial annealing-temperature of 100°C is scaled by a factor of 0.9 after every 50 iterations. The search continues until one of the following termination criteria are reached: i) L=350 consecutive search iterations with no improvement in the best solution cost are encountered; ii) The annealing-temperature passes below 1°C. Also, a value of 1000 is used for violation-penalty coefficient (*c*) in the SA cost function. The above parameter values for our SA-based search are empirically determined for efficient exploration of solution search space within a reasonable simulation time overhead.

In our thermal analysis, a fixed micro-fluid flow-rate *Ω*=10ml/min (constant cooling power is assumed) and maximum temperature constraint Ψ=90°C is used. The incoming coolant (de-ionized water) temperature is assumed to be 25°C. Our regular 3D-PDN power grid is modeled based on the guidelines provided in [7]. For the 64-core platform, with 16 cores on each tier, a total of 256 input power pins are used with $n^2=16$ grid-points for each core. For the 144-core platform, with 36 cores on each tier, $n^2=9$ grid-points per core are used. Values of 40mΩ and 83mΩ are used for horizontal and vertical branch resistances, based on [14]; and a maximum IR-drop constraint Γ=6.5% is assumed.

The nominal power values of NoC routers and links (32-bit wide) for different voltages, frequencies, and router complexities at varying communication loads for the 32nm node are obtained from ORION 2.0 [40]. The NoC router is assumed to be operating at the same supply voltage and temperature as the compute core it is associated with. Therefore, to integrate the effects of temperature and IR-drops in the NoC routers (which may contain MCFIFOs as needed), the leakage power of each router is scaled in proportion to the leakage power of the corresponding compute-core connected to it.

4.6.2 Experimental Results

To the authors' knowledge, [67] is the only prior work that has proposed a design-time MPSoC synthesis framework to co-optimize PDN costs and on-chip communication costs. Therefore, to evaluate the quality of solutions generated by our MPSoC synthesis framework (PRATHAM), we compare our results with the synthesis framework from [67]. Although the authors in [67] integrate PDN-awareness (in terms of worst case IR-drop constraints) in their synthesis framework, they fail to account for the effects of IR-drops on power and performance of the system. Additionally, they do not consider thermal-awareness while evaluating the cost of any given MPSoC solution. PRATHAM overcomes the above mentioned drawbacks by integrating the evaluation of the combined impact of IR-drops and temperature on the system power/performance profile into our holistic evaluation methodology. We also compare our framework with a thermal and communication-aware mapping framework for 3D mesh-based MPSoCs, proposed in [84]. The framework in [84] utilizes an incremental mapping heuristic to minimize both the peak temperature and the communication traffic in the 3D MPSoC, but it fails to consider the effects of IR-drops in the PDN.

We implemented the frameworks presented in [67], [84] to the best of our understanding and used the same platform and workload models for both implemented approaches to ensure a fair comparison. The synthesis approach in [67] outputs a Pareto front with solution-points optimized for communication energy and maximum IR-drop with varying degrees. In our implementation of [67], our holistic solution evaluation is performed only on the final set of solutions, and then the solution with the best ED^{2}P cost which meets PDN-constraints (Γ) for all cores is chosen for comparison. On the other hand, PRATHAM performs the holistic solution evaluation after every perturbation exploring the solution search-space more intelligently, while simultaneously optimizing both the communication and the computation profile of the system. Our implementation of [84] is adapted to the liquid-cooled 3D-MPSoCs, where we attempt to map the high-power compute-cores close to the liquid-inlets while minimizing the network traffic. Note that as [84] does not discuss a 3D NoC routing algorithm, we assume the same routing scheme for [84] as used in PRATHAM.

Table 8: Total number of (thermal/IR-drop) violations obtained from using [84], [67] and PRATHAM. PDN-unaware framework [84] incurs high number of PDN-violations and thermal-unaware framework [67] incur significant number of thermal-violations; whereas, PRATHAM produces solutions with no violations.

Num. of violations	wld-1A	wld-2A	wld-3A	wld-4A	wld-1B	wld-2B	wld-3B	wld-4B
[84]	0/9	0/10	0/9	0/12	0/19	4/24	0/26	0/17
[67]	1/0	7/0	5/0	8/0	0/0	16/0	10/0	1/0
PRATHAM	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0

Table 8 shows the total number of thermal/IR-drop violations (e.g., 0/9 indicates 0 thermal violations and 9 IR-drop violations) obtained when using [84], [67] and PRATHAM. It can be observed that the thermal-aware incremental mapping framework [84] satisfies thermal constraints for all cores in almost all test-cases. Although, unlike our holistic evaluation framework, [84] does not perform entirely accurate thermal-analysis. Therefore, the final solution produced may occasionally violate one or more thermal constraints (as in the case of wld-2B shown in Table 8). Also observe that as [84] is PDN-unaware, a large number of cores in its synthesized solution violate IR-drop constraints. In addition to IR-drops in the PDN being a serious system-reliability concern, as the maximum operating core-frequency is strongly related to the supply voltage, IR-drops, if left unchecked, adversely affect the operating frequencies of cores, which causes degradation in overall system-performance. Table 9 shows the average core frequencies in solutions obtained from [84], [67], and PRATHAM. Observe in table 9 that PRATHAM [67], which satisfy all IR-drop constraints (as shown in Table–I), produce final MPSoC solutions with notably higher average core frequencies. Finally, as [67] is thermal-unaware, it incurs a significant number of thermal violations (as shown in Table 8).

Table 9: Average core frequencies obtained for the final MPSoC solutions produced by [84], [67], and PRATHAM

Avg. freqs. (MHz)	wld-1A	wld-2A	wld-3A	wld-4A	wld-1B	wld-2B	wld-3B	wld-4B
[84]	1436	1422	1420	1419	1452	1426	1427	1448
[67]	1470	1450	1449	1462	1482	1462	1460	1478
PRATHAM	1473	1461	1448	1466	1495	1461	1464	1487

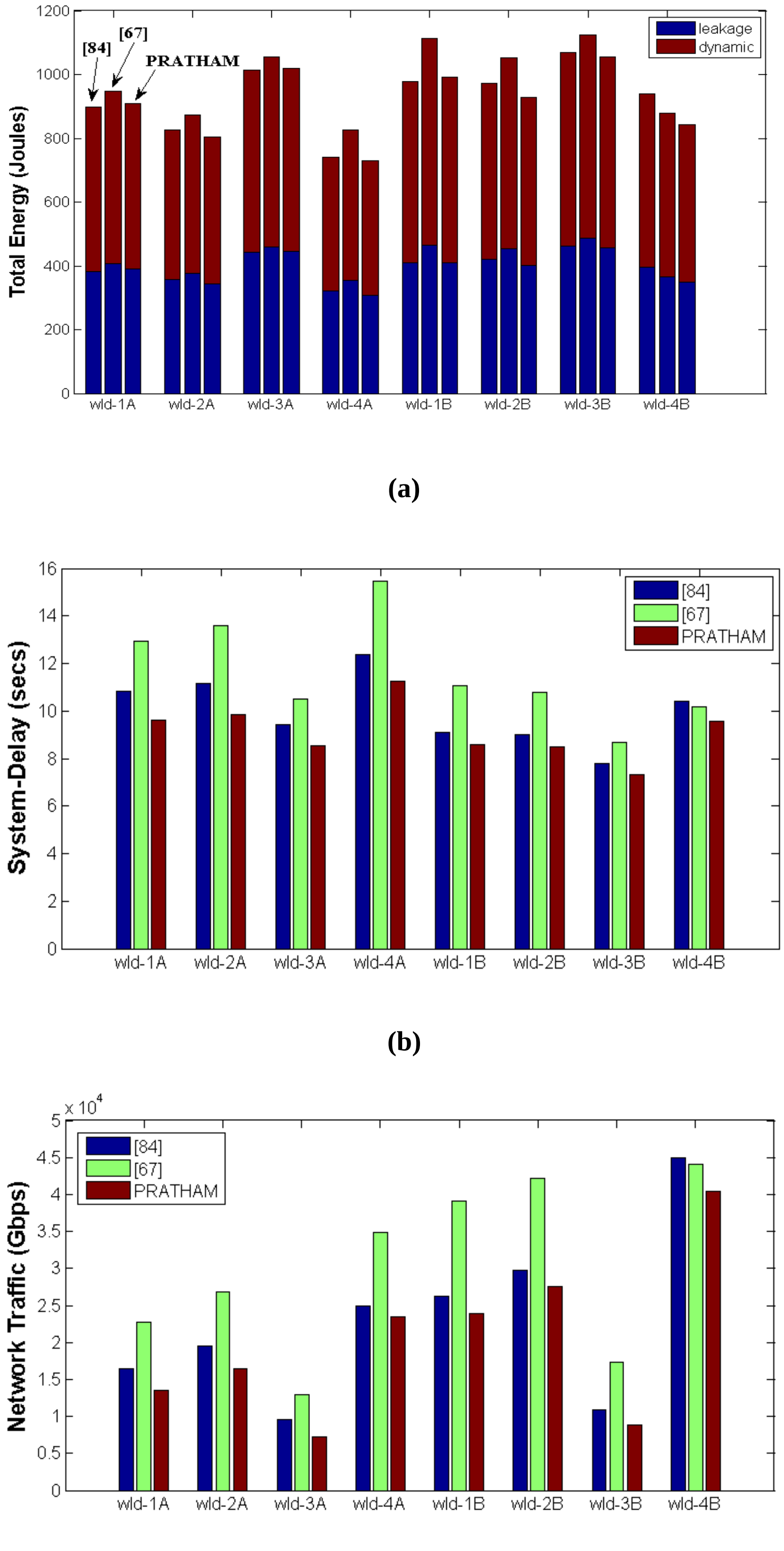
Figures 31((a)-(d)) shows results for energy consumption, system-delay, network traffic, and ED^2P for the best solutions generated by frameworks from [84], [67], and PRATHAM. The framework in [67] utilizes an efficient SA-based search algorithm, but without the knowledge of the various interdependencies between design-metrics, it is unable to accurately evaluate the solution-costs of intermediate solutions while attempting to optimize the maximum IR-drop and communication energy. Therefore, it ends up with final synthesis solutions that possess much worse network traffic compared to solutions obtained from PRATHAM (Figure 31(c)). For the framework in [84], thermal-profile is more optimized and core supply voltage values are smaller (due to higher IR-drops) compared to [67].

Therefore the leakage energy as well as the dynamic energy with [84] is significantly lower than [67] (Figure 31(a)).

PRATHAM intelligently explores the overall solution search-space, co-optimizing communication, thermal profile, and IR-drop distribution in the PDN, while satisfying all PDN and thermal constraints, and thus: *(i)* produces average improvement of 7.8% and 2.7% in leakage energy (7.5% and 2.1% in terms of total energy) over [67], [84]; and *(ii)* produces the highest average core-frequencies (Table 9) and the lowest network traffic (Figure 31(c)); resulting in significant savings in system-delay of 21.4% and 8.7% over [67], [84] respectively (Figure 31(b)).

Note that leakage power increases exponentially with supply-voltage and dynamic power increases with the square of supply-voltage, whereas core-frequency increases approximately linearly with supply-voltage, thus operating at lower voltage levels (with higher IR-drops) would generally be more energy-efficient (also noted in [82]). Despite the above observation, PRATHAM produces energy-savings of 2.1% on average over framework [84] (Figure 31(a)), which incurs numerous IR-drop violations (as shown in Table 8).

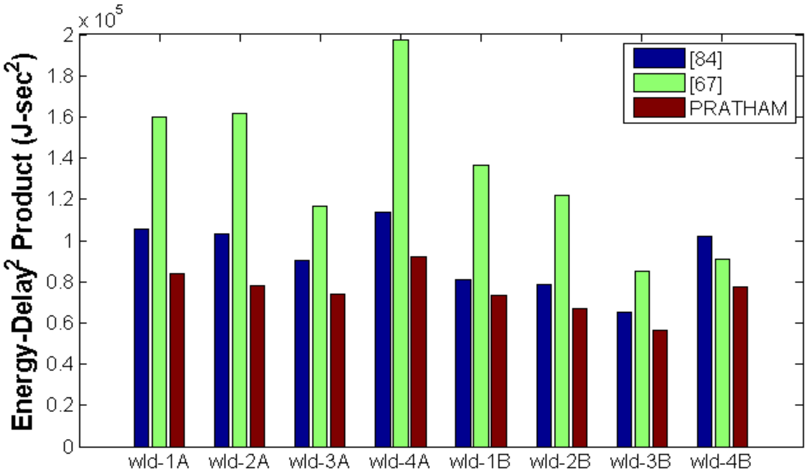
In summary, it is apparent that frameworks focusing primarily on thermal-awareness or PDN-awareness during 3D MPSoC synthesis produce infeasible solutions with IR-drop violations or thermal violations, respectively. PRATHAM integrates thermal- and PDN-awareness to generate solutions with better overall optimality. As shown in Figure 31(d), the system-ED^{2}P is reduced by 18.5% and 43.6% on average, over [84] and [67] respectively. PRATHAM derives its improvements by combining the communication and computation domains of optimization within a single solution search space, while more accurately modeling the complex set of inter-dependencies among various system metrics.


Total Energy (Joules)
0
200
400
600
800
1000
1200
[84]
[67]
PRATHAM
leakage
dynamic
wld-1A
wld-2A
wld-3A
wld-4A
wld-1B
wld-2B
wld-3B
wld-4B
System-Delay (secs)
0
2
4
6
8
10
12
14
16
[84]
[67]
PRATHAM
wld-1A
wld-2A
wld-3A
wld-4A
wld-1B
wld-2B
wld-3B
wld-4B
Network Traffic (Gbps)
x 10^4
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
[84]
[67]
PRATHAM
wld-1A
wld-2A
wld-3A
wld-4A
wld-1B
wld-2B
wld-3B
wld-4B

(a)

(b)

(c)



(d)

Figure 31: Experimental results showing improvements with PRATHAM over [84] and [67] for eight multi-application workloads that combine various benchmarks from SPLASH2 and PARSEC benchmark suites: (a) total (leakage + dynamic) energy dissipation (b) system-delay, (c) network traffic, and (d) system-ED2P.

Table 10: Simulation times (in minutes) when executing 8 different workloads with the SA-based framework [67] and PRATHAM. Simulation time for executing the incremental-mapping approach in [84] is in the order of a few of minutes, and thus omitted.

Sim. times (minutes)	wld-1A	wld-2A	wld-3A	wld-4A	wld-1B	wld-2B	wld-3B	wld-4B
[67]	240	208	232	449	297	212	221	273
PRATHAM	1708	1755	1723	1356	1934	1208	1953	1927

Finally, we present a comparison of simulation times taken by the synthesis frameworks. Table 10 shows the results of this comparison. We assume the same SA-parameters and initial solution for the implementation of [67] as used for PRATHAM, for a fair comparison. The SA-based search in [67] only performs PDN-analysis and routing path allocation on each intermediate mapping solution, whereas, PRATHAM executes multiple iterations involving fast PDN-analysis and thermal-analysis (within our holistic evaluation framework) for every intermediate mapping solution. Therefore, the execution times for PRATHAM, which is guided by a much more intelligent search algorithm, are significantly longer than that of [67]. The 1-1.5 days required for analysis with PRATHAM is however quite reasonable, given that the analysis is performed at design time where spending several weeks for system-level analysis is not uncommon.

4.7 Conclusion

As circuit performance is strongly co-related to the quality of both the voltage supply and operating temperatures, improving upon the worsening IR-drops and thermal profiles will be critical in future 3D MPSoC designs. For the first time, we propose a 3D MPSoC synthesis framework (PRATHAM) that co-optimizes on-chip IR-drops, as well as thermal and communication profiles, for better overall system performance and energy dissipation, while accurately accounting for the impact of increasing temperatures and IR-drops on system performance and power dissipation.

Our experimental results indicate that PRATHAM improves system-ED^{2}P by up to 24.1% and 53.4% (**18.5%** and **43.6%** on average) over thermal-aware and PDN-aware 3D MPSoC synthesis frameworks from prior work, while also satisfying all IR-drop and thermal constraints. PRATHAM represents one of the first efforts to tightly integrate PDN design, thermal, power, and performance objectives as part of a design-time exploration effort for 3D MPSoCs. Solutions generated by PRATHAM are also more amenable to efficient physical design because of considering PDN design issues and the impact of IR-drops and temperature on system power and performance early in the design flow. PRATHAM can also be complimented by a lightweight run-time scheme that performs energy optimization in incremental steps, by employing techniques such as DVFS and task-migration to account for the time-varying communication and computation profiles of applications.

5 Process Variation Aware Synthesis of Application-Specific MPSoCs to Maximize Yield

In contemporary semiconductor technologies, considerable unpredictability in the behavior of manufactured chips is being observed due to the effects of process variations. This unpredictability translates into variations in power and performance within these chips. At the same time, with ever shrinking power budgets and rising cooling costs, most chip designs need to satisfy a hard limit on the maximum power that the chip can dissipate. In such a scenario, the yield of a design for a given process depends on the number of chips meeting both performance and power constraints. In this chapter, we propose a novel process variation-aware MPSoC synthesis framework that performs simultaneous mapping and voltage assignment of cores to mitigate the adverse effects of process variations while maximizing yield.

5.1 Introduction

Unpredictability in system behavior due to variability in modern fabrication processes has become a serious concern for chip designers. In modern multiprocessor system-on-chip (MPSoC) designs, spatially correlated systematic within-die (WID) variations often manifest across multiple cores, leading to core-to-core (C2C) variations [86]. At the same time, die-to-die (D2D) variations remain quite significant [87]. Both WID and D2D variations have random and systematic components. One of the primary effects of such process variations is the deviation of the threshold voltage (V_T) from its nominal value [88]. A rise in the value of V_T increases circuit delay (which decreases maximum core frequency) and decreases the leakage power of the core; on the other hand, a reduction in V_T decreases circuit delay and increases leakage power. Variations can thus impact both the performance and power of MPSoC designs in undesirable ways. Therefore there is a pressing need to mitigate process variations at all levels of chip design with new design techniques, especially at the early system-level where design decisions have the biggest impact.

In addition to unpredictability from variations, rising power dissipation is another cause for concern in today’s chips. MPSoCs today almost always have an upper limit on the allowed power dissipation. Voltage islands (*VIs*), which combine cores into islands that use the same V_{DD} and ground lines, have successfully been employed to manage chip power. Also, the use of *VIs* has been shown to isolate the effects of WID process variations efficiently to boost performance [89]. One recent trend has been the rise in the power footprint of network-on-chip (NoC) fabrics. NoCs have been shown to dissipate significant power (e.g., ~30% of chip power in Intel’s 80-core teraflop [4] and ~40% of chip power in MIT RAW [5] chips). As a result, reducing not only computation but also communication power has become a high priority for chip designers. In order to minimize communication power in the presence of voltage islands, efficient *VI*-aware voltage-assignment, core-mapping and routing techniques need to be employed [33], [24], [90], where the number of VLCs (voltage level converters) and MCFIFO (multiple clock first in-first-out) queue based frequency level converters required for inter-*VI* communication are optimized in addition to network traffic.

Given the critical need to manage variations and power in MPSoCs, new frameworks are required that can optimize chip designs for any given applications. In this work we propose one such framework for application-specific variation- and energy-aware MPSoC synthesis. The proposed framework targets MPSoCs with a mesh-based NoC topology, similar to recent commercial MPSoCs, e.g., Tilera’s TILE64 [5] and Intel’s SCC [3]. Our framework improves upon prior work in several ways, with major improvements in the core voltage assignment phase. Existing work on variation-aware MPSoC synthesis (e.g., [91], [92]) typically performs core voltage assignments such that all cores meet their performance constraints for the highest (slowest) possible V_T value. But this incurs a power penalty due to higher than optimal voltage assignments to cores. To overcome the drawback of a two-step design process of first performing pessimistic voltage-assignments and then doing core-to-tile mapping as done in these works, our framework utilizes a novel variation-aware simultaneous voltage assignment and mapping technique to more efficiently reduce power while meeting performance constraints. Our experimental results show average improvements ranging from 2× to 3.8× in power-performance yield over other variation-aware MPSoC synthesis frameworks [91], [92], [96] proposed in prior literature.

5.2 Related Work

Over the past few years, the problem of variation-aware yield enhancement of MPSoCs has been studied in several works [93]-[96]. Wang et al. [93] improve performance yield by modeling task completion time as a stochastic variable and generate both task scheduling and routing procedures to optimize the probability of a given schedule meeting performance constraints. Huang et al. [95] propose to optimize performance yield by utilizing multiple test-chips representative of the spatially correlated systematic WID variations in addition to random variations. They use a simulated-annealing (SA)-based scheduling technique as well as a clustering technique to generate multiple synthesis solutions at design-time, such that one of these solutions can be selected at run-time, based on the actual variation of each chip. Bhardwaj et al. [96] propose a new design metric - power-performance yield (PPY) defined as the percentage of test chips meeting both performance and power constraints. By using an SA-based approach they trade-off the performance yield with power yield to optimize PPY. Much like prior work (e.g., [95], [96]) we consider multiple variation (V_T) maps, represented by N test chips, to capture the effects of process variations on performance and power. *But in addition to WID variations, we also consider D2D variations in our test set of N V_T-maps, which prior work [93]-[96] does not consider.*

A few prior works propose frameworks for variation-aware synthesis of MPSoCs with *VI*s, to optimize computation and communication power [91], [92]. Mazjoub et al. [91] propose a *VI*- and core-placement approach to achieve a balance between limits of spatial extent of the WID variations across cores, and communication patterns between *VI*s, by varying the shape of *VI*s to minimize total power. In our previous work [92], we consider the locations of *VI*s to alleviate the effects of process variations. Here, we make use of a single V_T-map to perform variation-aware *VI*-placement to optimize compute energy. These approaches [91], [92] tend to produce synthesis solutions with sub-optimal power because they assume pessimistic initial voltage assignments of cores corresponding to the worst-case V_T value. *In this work, for the first time, we propose a variation-aware design-time simultaneous voltage assignment and mapping based synthesis framework to reduce power while meeting performance constraints in MPSoCs with VIs. Our approach integrates novel variation-aware and VI-aware optimization techniques to produce notable improvements in PPY over prior work.*

5.3 Problem Formulation

The inputs to our problem are as follows:

- An MPSoC with a regular mesh-based 2D NoC with *T* tiles: $T = (d^2)$, where *d* is the mesh dimension, and each tile consists of a computate core, a NoC router, and a network interface (*NI*) between the router and core;
- A set of *N* V_T-maps incorporating the effects of WID and D2D variations with continuous distribution over the die;
- A core graph *G(V, E)* with a set of *T* vertices, representing homogeneous cores on which tasks have already been mapped and $\{f_1, f_2, f_3, \ldots, f_T\}$ representing rated operating frequencies of the *T* cores; and the set of *M* edges $\{e_1, e_2, e_3, \ldots, e_M\}$, where edge weights $\{w(e_1), w(e_2), \ldots, w(e_M)\}$ represent minimum bandwidth constraints between cores;
- A set *S* of candidate supply voltage (V_{dd}) levels, where each V_{dd} level can form at most one *VI*;
- A chip-wide power dissipation constraint (*PC*).

Objective: Given the above inputs, the goal of our work is to perform: *(i)* voltage assignments of cores; *(ii)* core-to-tile mapping; and *(iii)* synthesize a regular 2D mesh NoC for a specific application; such that all cores within individual *VI*s are contiguously placed while maximizing power-performance yield (PPY). We define PPY as the number of test-chips (out of *N*) that *(i)* satisfy frequency (performance) constraints for all T cores, as well as *(ii)* satisfy the power constraint (PC), considering power consumption (dynamic and leakage) in compute cores and communication resources (routers, links and VLCs/MCFIFOs).

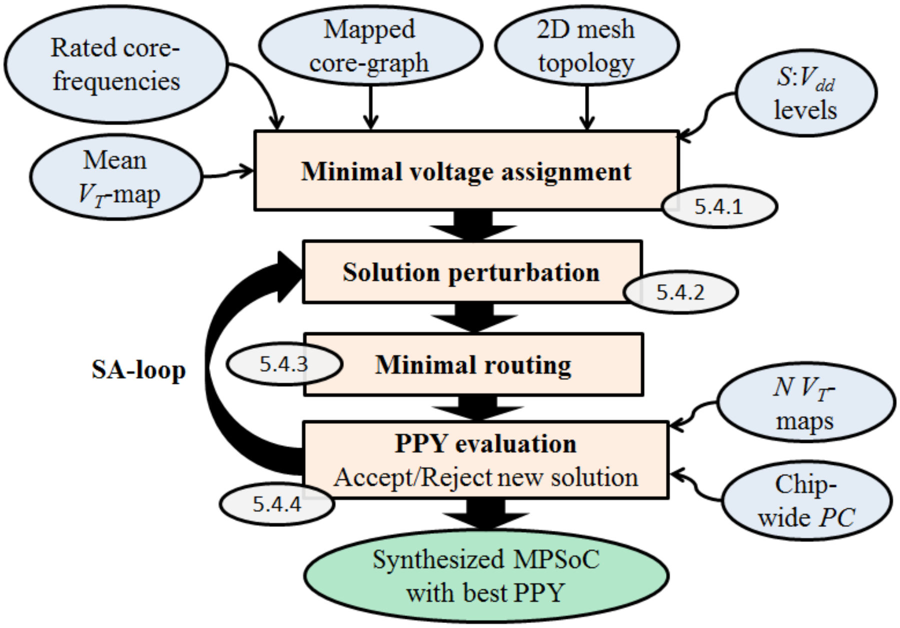


Figure 32: Design flow of the synthesis framework

5.4 Variation-Aware MPSoC Synthesis Framework

Figure 32 shows the flow of our proposed variation-aware MPSoC synthesis framework. Given the application core-graph and a 2D mesh of tiles on the MPSoC, we start with a core-to-tile mapping solution, which is optimized for communication traffic using incremental swapping (first proposed in [55]). Then contiguous *VI*s are formed (from the candidate V_{dd} levels of the input set S) with minimal voltage assignments for the current mapping, such that the rated frequencies of all cores are satisfied for the mean V_T-map (discussed in section 5.4.1). The mean V_T-map represents the test-chip (out of N test-chips) with a V_T-map which is least different from all other V_T-maps (i.e., has the least average difference in V_T values over all T tiles). A simulated annealing algorithm (SA-loop) is initialized with this synthesis solution, which iteratively perturbs the current mapping solution, performs minimal routing, and evaluates the current PPY (discussed in section 5.4.2). At each iteration within the SA-loop, our minimal routing heuristic allocates paths for communication flows such that the number of VLCs used for inter-*VI* data communication is minimized (discussed in section 5.4.3). Then, the current synthesis solution is subsequently evaluated for each of the *N* test-chips to obtain a corresponding PPY value. The PPY evaluation basically computes the number of test-chips which satisfy both the performance and power constraints for the given synthesis solution (discussed in section 5.4.4). Finally, our framework outputs the MPSoC synthesis solution with the highest PPY.

In the following sections (Sections 5.4.1 – 5.4.4), we discuss the details of each step in our synthesis framework in detail.

5.4.1 Minimal voltage assignment

The objective of this step is to assign minimal voltages to cores for the current core-to-tile mapping such that all cores can be clocked at their rated operating frequencies for the mean V_T-map. Note that the maximum operating frequency can be increased by increasing the supply voltage (V_{dd}), though at the cost of increased power. Thus, for a certain $\{V_T, V_{dd}\}$ pair, the core can operate at a certain maximum frequency, in other words, to meet the minimum operating frequency requirement of a particular core, an appropriate $\{V_T, V_{dd}\}$ pair needs to be chosen. The following equation gives the relationship of core frequency *f*, supply voltage V_{dd}, and the V_T value of the tile, where α and β are constants.

$$f = \frac{(V_{dd} - V_T)^{\alpha}}{\beta . V_{dd}} \quad \ldots (5.1)$$

A valid voltage assignment must also satisfy the *VI*-contiguity constraint (all cores within individual *VI*s are contiguously placed), and voltage levels must be chosen from the input set *S*. These objectives are accomplished in two steps: *(i)* calculate minimum core voltages to satisfy performance constraints, based on the mean V_T-map (section 5.4.1.1); *(ii)* create up to *S* contiguous *VI*s based on the continuous distribution of minimum voltages (section 5.4.1.2).

5.4.1.1 *Calculation of minimum core voltages*

The minimum core voltages are calculated such that all operating frequency constraints are satisfied for V_T values of the mean V_T-map. We first elaborate on how the mean V_T-map is chosen out of the *N* V_T-maps. A V_T-map is represented as $\{V_{T1}, V_{T2},\ldots V_{TT}\}$. For each ($\mathrm{i}^{\mathrm{th}}$) map, we find the V_T-difference value w.r.t. each of the other (j^{th}) *N*-1 maps: V_T-diff(i,j) = $|V_{T1}^{\mathrm{i}}- V_{T1}^{\mathrm{j}}|+|V_{T2}^{\mathrm{i}}- V_{T2}^{\mathrm{j}}|+\ldots+| V_{TT}^{\mathrm{i}}-V_{TT}^{\mathrm{j}}|$. Then, V_T-diff(i) is the average V_T-difference of the i^{th} map w.r.t. all other maps: V_T-diff(i)= [{V_T-diff(i,j)}+…{}]/(*N*-1) : for all j≠i. Thus, V_T-diff(i) is the quantification of how different the i^{th} V_T-map is from all other maps. After computing this V_T-difference for all *N* V_T-maps, the V_T-map with lowest V_T-diff(i) becomes the mean V_T-map.

Given the rated frequencies for each core and the mean V_T-map for the $d\times d$ mesh, a continuous distribution of minimum core voltages can be computed (using equation 5.1) for the current core-to-tile mapping.

5.4.1.2 *Formation of contiguous* VI*s*

Given the continuous distribution of minimum core voltages over the 2D mesh (example shown in figure 33(a)), and set *S* of candidate supply voltage levels; in this step, we form contiguous *VI*s (figure 33(e)). Considering all candidate voltage levels in decreasing order (starting with the highest V_{dd} level), we list all the tiles with minimum voltages less than or equal to the current V_{dd} and greater than the next lower V_{dd}. We then attempt to connect all these tiles in order to form a single contiguous *VI* of the current V_{dd} level (figure 33(b)). This can create disjoint islands of tiles with minimum voltages within the current voltage range (shown as grey tiles in figure 33(c)). We then connect each island to the biggest island (of size 2 in figure 33(c)) following either a XY or YX path between the closest pair of tiles (one from each island). Here, we choose the path with the least voltage overhead (the difference between the current V_{dd}

and minimum core voltages along the path). Note that the green path is chosen in figure 33(c) because it has a lower voltage overhead compared to the red path.

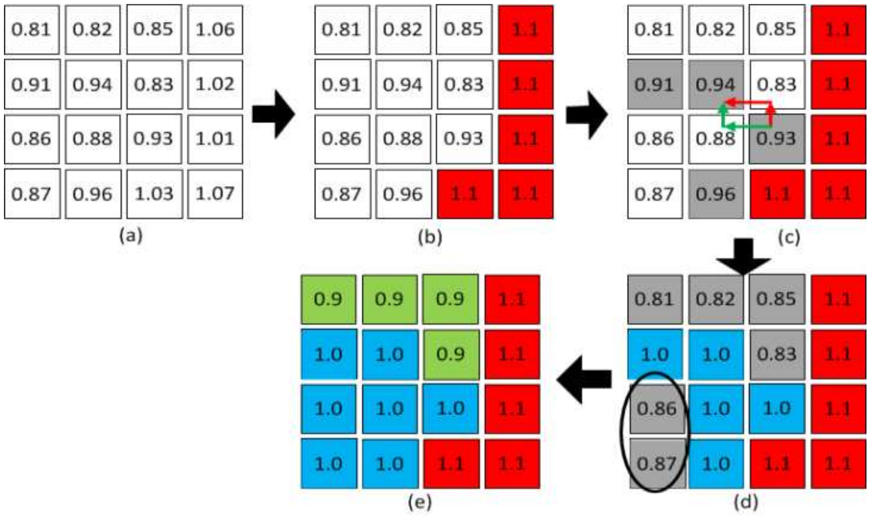


Figure 33: An illustrative example of our minimal voltage assignment: *VIs* are color-coded and annotated with V_{dd} levels, and grey tiles are marked to be connected together to form the *VI* of current V_{dd} level

In certain situations, some disjoint islands, called separated islands, may not be able to connect to the biggest island because the corresponding XY and YX paths are obstructed by previously placed *VI*s of higher V_{dd} levels. In figure 33(d), the en-circled separated island cannot be connected to the biggest island (other grey tiles) to form a single contiguous *VI* of the current V_{dd} (0.9V). Here, we assign the biggest contiguous island with the current V_{dd} (green *VI* in figure 33(e)) and connect the separated islands to their nearest previously placed *VI*s (V_{dd} = 1.0V in figure 33(e)). Note that as we consider placing *VI*s in decreasing order of V_{dd} levels, all previously placed *VI*s would be of voltage levels higher than the current V_{dd}, therefore, connecting separated islands to them will not violate minimum voltage requirements of separated islands.

Finally, the minimal voltage assignment method yields a set of contiguous *VI*s on the mesh, such that the current voltage-assigned core-to-tile mapping solution satisfies the frequency constraints of all cores for the mean V_T-map.

5.4.2 SA-loop and solution perturbations

In this step, our goal is to modify the solution from the previous step to optimize PPY. We make use of a simulated annealing (SA) algorithm in this step to iteratively perturb the current solution in one of three ways, as described later in the section, with an objective function that maximizes PPY. After every solution perturbation, our minimal routing path allocation (Section 5.4.3) and PPY evaluation (Section 5.4.4) is performed. The PPY evaluation step calculates the total (computation + communication) power dissipation for all the test-chips that satisfy the performance constraints. For the current synthesis solution, the number of test-chips (out of *N* test-chips) satisfying both power and performance constraints is the current PPY. The SA procedure ultimately outputs the synthesized MPSoC solution with the highest PPY. The three main perturbations used in our SA-loop procedure are:

Mapping perturbation: Two cores are randomly chosen to be swapped with each other. Note that once the current mapping is perturbed, *VI*-assignment for the current solution may no longer satisfy performance constraints with respect to the mean V_T-map. Therefore, we perform minimal voltage assignment (discussed in section 5.4.1) after every mapping perturbation.

VI-assignment perturbation: A tile on the mesh, which is on the periphery of a *VI* and is adjacent to another *VI* is randomly chosen. Then, the V_{dd} level for the chosen tile is either hiked or lowered in order to transfer it into the neighboring *VI*. Note that any *VI*-assignment perturbation is valid only if it does not violate the contiguity of any of the *VI*s on the mesh. Therefore, validity checks are necessary before this perturbation is executed. Figure 34 shows an example of both a valid and an invalid *VI*-assignment perturbation.

Hike highest voltage perturbation: Only when the highest V_{dd} for the current solution is lower than the highest V_{dd} in set *S*, we choose any one core from the existing highest voltage *VI* of the current solution and hike the voltage to the next higher candidate voltage level (such that this *VI* remains contiguous). Note that the *VI*-assignment perturbation does not introduce new candidate voltages from set *S*. Therefore, the hike highest voltage perturbation is occasionally applied to explore the higher candidate V_{dd} levels in the solution search-space as well.

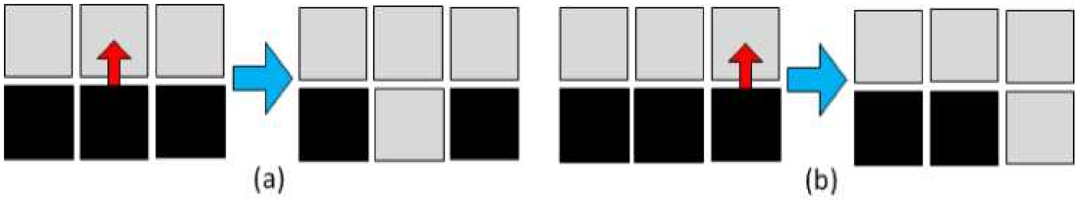


Figure 34: Illustrating valid and invalid VI-assignment perturbations: a tile from the black VI is transferred to the neighboring grey VI. (a) Example of an invalid perturbation, where the black VI no longer remains contiguous (b) Example of a valid perturbation.

5.4.3 Minimal routing

Given a mapped and voltage-assigned solution, our routing heuristic allocates minimal paths to all communication flows to generate a synthesized MPSoC solution. Note that whenever an inter-island link is inserted, voltage level converters (VLCs) are required in the corresponding NoC routers. These VLC components incur an overhead in terms of power dissipation and delay. Thus, the main objective of our routing step is to find a path for each communication flow such that the number of inter-island links is reduced. For each communication flow, we consider all candidate minimal paths. Out of these candidate minimal paths, we choose a routing path based on the following optimization objectives (in that order): 1) minimize total number of inter-island link insertions needed on the path; and 2) minimize total number of intra-island link insertions needed on the path.

As in [90], the communication flows are sorted in the increasing order of their path lengths, in decreasing order of their communication bandwidths for the same path length; and routed in that order. While routing any communication flow over a given path, links insertions are performed whenever the existing link(s) cannot support the bandwidth of the current flow or if no links are available. In summary, this routing scheme optimizes the number of inter-island links as well as intra-island links, thereby minimizing communication power.

5.4.4 PPY evaluation

Here we discuss the evaluation of performance yield (PY) as well as power-performance yield (PPY) for any given complete synthesis solution – voltage-assigned, mapped, and with routing paths allocated. The synthesis solution needs to be evaluated for satisfaction of performance and power constraints, for each of the *N* test-chips. To check if a certain test-chip satisfies the performance constraints for a given solution, we compute the maximum frequency that each of the *T* cores can be clocked at (using equation 5.1). If all *T* cores can be clocked at their rated frequencies, then, the test-chip under consideration is said to satisfy performance constraints for the given synthesis solution. Thus, the number of test-chips (out of *N*) that satisfy performance constraints, becomes the performance yield for the given synthesis solution.

In order to evaluate the PPY for a given solution, we evaluate total power dissipation of only those test-chips which satisfy the performance constraints. If the total power dissipation is found to be lower than *PC*, then power constraint is considered satisfied for that test-chip. Note that routers, MCFIFOs, and VLCs operate at the same voltages/frequencies as the cores they are associated with, and are affected by the same variations in threshold voltage. Therefore, for the given synthesis solution, communication power is also evaluated for each of the *N* test-chips. In summary, the PPY evaluation step calculates the number of test-chips (out of *N*) that satisfy both performance and power goals for a given synthesis solution.

5.5 Experiments

5.5.1 Experimental setup

Our experiments were conducted using six parallel application benchmarks: four from the SPLASH-2 benchmark suite [58] (*fft*, *raytrace*, *lu*, and *cholesky*), and two from the PARSEC benchmark suite [56] (*vips* and *dedup*). Our core graphs are modeled based on the inter-core communication characterization from [47]. Each vertex in a core-graph corresponds to a producer/consumer core and the edge weights represent inter-core communication bandwidths (based on our observations of traces and communication patterns between the respective pair of cores). The rated operating core frequencies range from 600 MHz to 1300 MHz, based on the level of compute intensity of the tasks assigned to the respective cores. We use the ARM Cortex-A9 multi-core processors [37] as the baseline MPSoC compute cores, which support five operating voltage levels (*S*=5): 0.8V, 0.9V, 1.0V, 1.1V, and 1.2V. We use 64 core and 100 core meshes (with core-graphs of same sizes) for MPSoC platforms with dimensions 8×8 and 10×10.

Much like prior work (e.g., [95]) we use a total of 1000 test-chips (*N*=1000), in our experiments. The 1000 V_T-maps are generated using the open-source tool [97] (based on systematic and random WID-variation model in [98]), for mesh sizes (8×8) and (10×10). The values of 0.4 and 0.09 are used for the statistical mean and a standard deviation of the parameter V_T respectively, and the correlation range (ϕ) of 0.5 is used (as recommended in [98]). A normally distributed V_T bias, representing the D2D variation component is superimposed onto these V_T-maps; the standard deviation of the D2D V_T is assumed to be 6%, as in [99].

The power values of routers and links (32-bit wide) for different voltages, frequencies and router complexities at varying communication loads, for 45 nm technology node are obtained from ORION 2.0 [40]. Note, the router power values obtained are for nominal V_T, and are scaled for varying V_T values. Also, as all cores operate at their own rated frequencies, inter-core communication requires MCFIFOs and VLCs for inter-island communication. We consider the power overhead of these components (proportional to the voltage supply), with the actual power values based on reported overheads from existing literature [21].

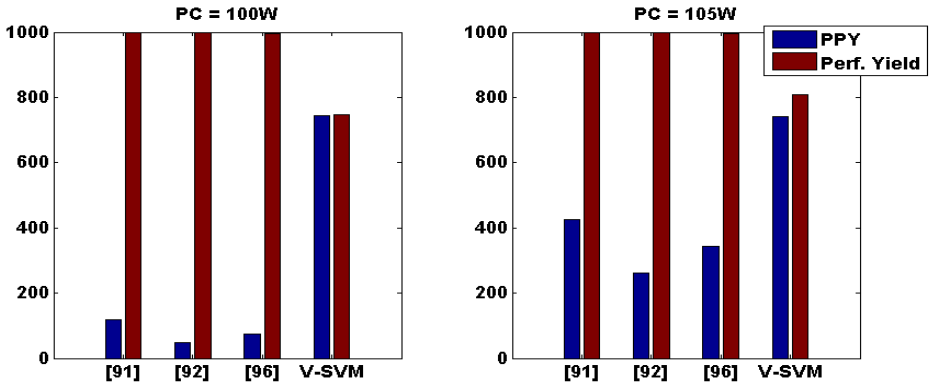
In our implementation of the SA-based algorithm, we simulate for 1000 iterations (with SA temperatures between 100 and 34), after which the solution quality does not seem to improve. To effectively explore both the mapping and *VI*-assignment solution search spaces, we invoke the three perturbations with the following probabilities: *(1)* Mapping perturbation: P_1=4/40, *(2) VI*-assignment perturbation: P_2=34/40, and *(3)* Hike highest voltage perturbation: P_3=2/40. Two chip-wide power-constraints (*PC_stringent* and *PC_nominal*) are used in our experiments. For 64 core benchmarks: *PC* values of 100W and 105W are used, and for 100 core benchmarks: *PC* values of 150W and 160W are used.

5.5.2 Results

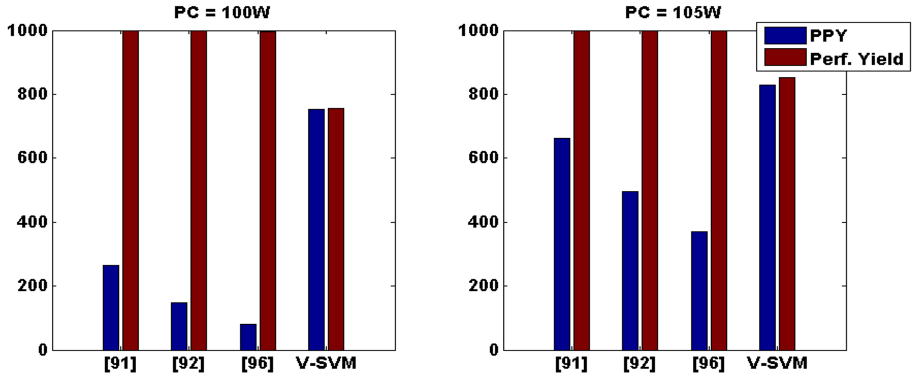
To evaluate the quality of solutions generated by our variation-aware simultaneous voltage assignment and mapping synthesis framework (hereafter referred to as *V-SVM*), we compare our results with three frameworks from prior work that we implemented along with our *V-SVM* framework.

Two of these frameworks from prior work perform process variation-aware and *VI*-aware MPSoC synthesis [91], [92]. The framework from [91] performs *VI*-assignment and core- placement to optimize the communication in the NoC, while attempting to retain the cloud-like shapes of *VI*s, in order to limit the spatial extent of systematic WID variations within individual *VI*s, thereby optimizing total energy. While [91] just considers the shapes of *VI*s, not the *VI*-locations on the mesh, the framework in [92] performs a variation-aware *VI*-placement that improves compute energy efficiency based on a single V_T-map; then core-to-tile mapping is performed within each *VI* to optimize communication energy. Both [91] and [92] do not use the knowledge of multiple variation maps, ignore chip-wide power-constraints, and optimize total power for a perfect performance yield. On the other hand, our *V-SVM* framework trades-off performance yield with power-yield for better PPY. We also compare our work against a third framework that performs variation-aware MPSoC synthesis [96]. Although, the framework in [96] does consider multiple test-chips, it does not consider *VI*s for power saving. Therefore, while implementing the approach in [96], we assume a relatively high voltage supply (1.1V) for all cores, such that almost all test-chips can satisfy all frequency constraints. Also, as [96] does not discuss the routing mechanism used, we use our minimal routing scheme in our implementation of [96] for an objective comparison with our *V-SVM* framework. We implemented the frameworks in [91], [92], [96] to the best of our understanding and used the same power and variation models for all implemented approaches to ensure a fair comparison of the algorithms used.

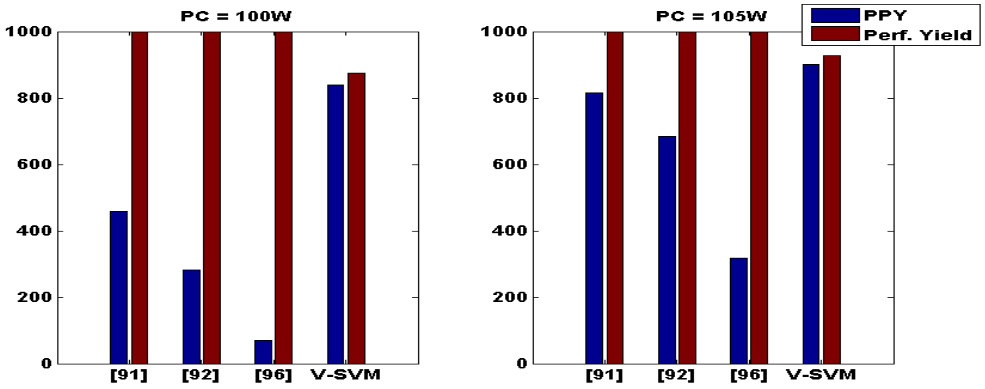
Figure 35 shows a comparison of the results obtained from using the four frameworks, in terms of performance yield and power-performance yield (PPY). Results for benchmarks with 64 core implementations are shown in figure 35(a)-(c) and for benchmarks with 100 core implementations are shown in figure 35(d)-(f)). Notice that for all the frameworks, the PPY increases as *PC* is relaxed (from left to right for any benchmark). Also note that as the primary objective of *V-SVM* is to trade-off performance yield with power-yield, the resulting PPY is generally very close to performance yield. On the other hand, for all other frameworks, even though the synthesis solutions achieve almost perfect performance yields, their PPY is quite poor.



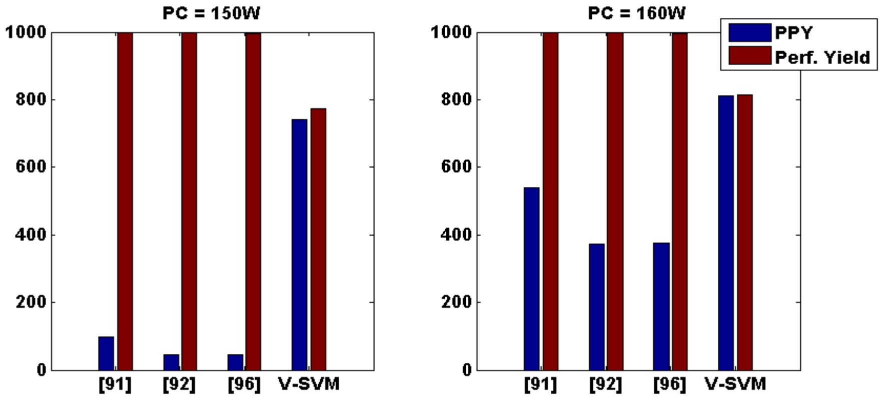
(a) SPLASH2-*fft*_64



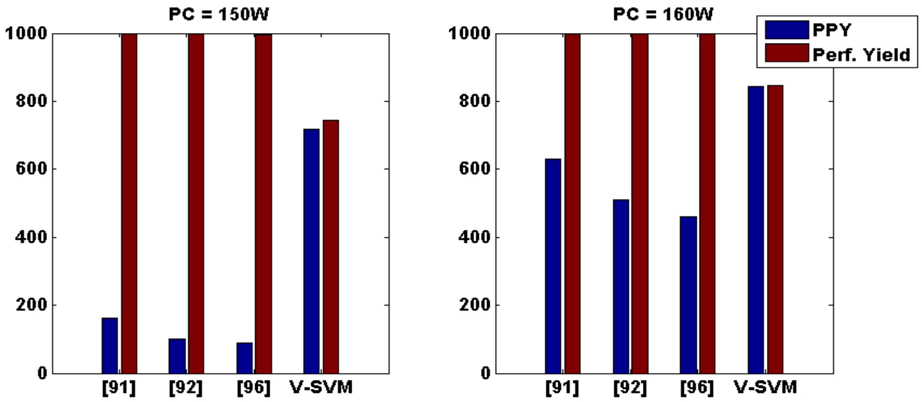
(b) SPLASH2-*raytrace*_64



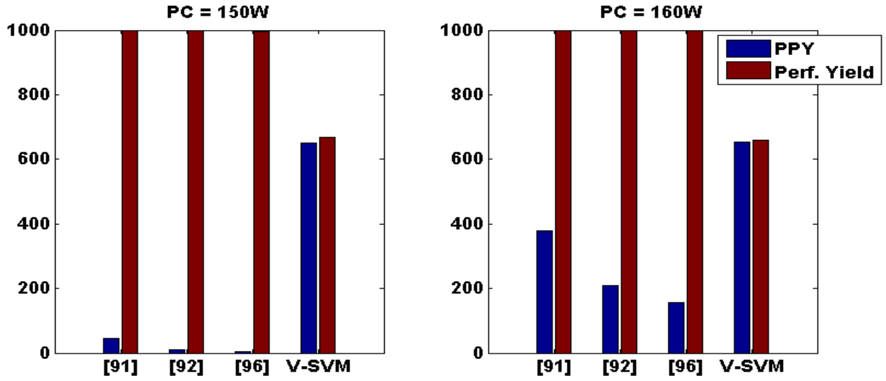
(c) PARSEC-*dedup*_64



(d) SPLASH2-*lu*_100



(e) SPLASH2-*cholesky*_100



(f) PARSEC-*vips*_100

Figure 35: Yield comparison for all four synthesis frameworks for 64 core and 100 core MPSoCs, using applications from SPLASH2 and PARSEC benchmark suites. PPY and performance yield are evaluated for each benchmark, using two power constraint (PC) values.

Both [91] and [92] assume a single WC V_T-value for all *T* tiles while performing voltage-assignments of cores, whereas, *V-SVM* combines the *VI*-assignment and mapping search-spaces to perform more power-efficient voltage assignments than [91] or [92], based on the knowledge of chip-wide *PC* as well as performance constraints corresponding to the target set of V_T-maps. Our *V-SVM* framework considers *VI*s with multiple supply voltages, which enables it to take advantage of a varied distribution of core frequencies to optimize total power. In contrast, by using a single voltage supply level for all cores, the optimization search space in [96] is quite limited, resulting in poor PPYs.

The framework from [91] outperforms [92] because [92] performs core/*VI* placement assuming only a single V_T-map, whereas [91] considers average trends during cloud shaped *VI* placement which provides better results across multiple V_T-maps. Both [91] and [92] outperform [96] as [96] does not use *VI*s (both [91] and [92] use *VI*s), which results in [91] and [92] producing lower voltage assignments that lead to higher PPYs.

Figure 35 also shows that *V-SVM* framework yields better improvements when the power constraint is more stringent. This is because a stringent power constraint lends more opportunity for our SA-loop to search for solutions with a better trade-off between performance and power. Such a trend can be readily observed in all benchmarks, by comparing the improvements obtained with *V-SVM* for different *PC* values. We can also observe a similar trend across benchmarks. For example, for a benchmark with low average power dissipation (SPLASH2-*cholesky*_100) where the given *PC* values are somewhat lax, gains in PPY with *V-SVM* (over [91], [92], [96]) are smaller compared to gains for a communication intensive, high power dissipation benchmark (PARSEC-*vips*_100) where the same *PC* values become quite stringent.

In summary, when PPYs are averaged across all 6 benchmarks and 2 PCs (i.e., across a total of 12 cases), *V-SVM* produces average PPY improvements of 2×, 2.9×, and 3.8× compared to the frameworks proposed in [91], [92], [96] respectively. Specifically, the improvements are 1.4×, 1.9×, and 2.4× with *PC_nominal*, and 3.9×, 7×, and 12× with *PC_stringent* over [91], [92], [96]. On the other hand, average performance yield obtained from using *V-SVM* is 0.79×, 0.79×, and 0.8× over that obtained from the frameworks in [91], [92], [96] respectively. Thus, rather than just aggressively optimize performance yield as done in prior work, our proposed *V-SVM* framework trades-off performance with power for a more desirable power-performance yield.

5.6 Conclusion

Due to the considerable impact of process variations in modern technologies, variation-awareness is essential even at the system-level design stage. Traditional design flows where voltage assignments are done without the core-mapping information require resorting to pessimistic voltage assignments based on worst case V_T values. This leads to higher voltage assignments and a corresponding higher overall power dissipation. For the first time, we propose a variation-aware system-level synthesis framework for simultaneous voltage assignment and mapping in MPSoCs with *VI*s. Our framework incorporates novel *VI*-aware and variation-aware optimization techniques, and produces on average 2× to 3.8× improvements in power-performance yield (PPY) over other variation-aware MPSoC synthesis frameworks. Such a framework has immense utility for emerging MPSoC designs.

6 VARSHA: Variation and Reliability-aware Application Scheduling with Adaptive Parallelism in the Dark-Silicon Era

With deeper technology scaling accompanied by a worsening power-wall, an increasing proportion of chip area on a chip multiprocessor (CMP) is expected to be occupied by dark-silicon. At the same time, design challenges due to process variations and soft-errors in integrated circuits are projected to become even more severe. It is well known that spatial variations in process parameters introduce significant unpredictability in the performance and power profiles of CMP cores. By mapping applications on to the best set of cores, process-variations could potentially be used to our advantage in the dark-silicon era. In this chapter, we propose a novel framework that leverages the knowledge of variations on the chip to perform run-time application mapping and dynamic voltage scaling to optimize system performance and energy, while satisfying dark-silicon power-constraints of the chip as well as application-specific performance and reliability constraints.

6.1 Introduction

With increasing transistor miniaturization, circuit densities have drastically increased, and the critical charge, which is the minimum charge capable of a bit-flip in a memory- or a logic-cell, has significantly decreased [8]-[10]. This phenomenon has caused newer process technologies to be more susceptible to transient-faults due to the effects of radiation, e.g., alpha-particle and neutron strikes. The rate of such transient-faults at run-time has thus been increasing with technology scaling [11]. Studies have also shown that hardened flip-flops are only 30%-50% more resilient than unprotected ones, at sub-40nm nodes [107], i.e., circuit-level hardening may not sufficiently suppress soft-errors. Thus, system-level run-time approaches to cope with transient faults and complement circuit-level techniques will be increasingly essential in newer process technologies.

Simultaneously, unpredictability in leakage power and circuit-delay due to variability in modern fabrication processes has become a serious concern. In emerging chip-multiprocessors (CMPs), spatially correlated systematic within-die (WID) variations manifest across multiple cores, creating core-to-core (C2C) variations [86]. At the same time, die-to-die (D2D) variations remain quite significant [87]. Both WID and D2D variations have random and systematic components, and it is difficult to predict the variation-profile of a fabricated chip at design-time [112]. It is however possible to extract the variation-map of a chip at run-time from the chip-frequency-profile obtained using ring-oscillator based delay-sensors [113], [114]. Thus, run-time approaches to mitigate adverse effects of process variations become possible, and will be vital for scaled technologies.

The slowdown of power scaling with technology scaling, due to leakage and reliability concerns [14], [15], has led to a rise in chip power-densities, giving rise to the dark-silicon phenomenon – a significant fraction of the chip needs to be shut-down at any given time to satisfy the chip power-budget. With the extent of dark-silicon increasing every technology-generation (30%-50% for 22nm) [16], [17], designs are becoming increasingly power-limited rather than area-limited. Run-time power-saving techniques such as dynamic voltage scaling (DVS) are thus becoming increasingly important.

Given these multiple daunting design challenges, there is a critical need for a system-level solution that can simultaneously and adaptively manage the constraints imposed by dark silicon, process variations, and soft-error reliability, while executing applications. In this work, we address this need by proposing a novel run-time application scheduling framework (VARSHA) that employs dynamically adaptable application degrees of parallelism (DoPs) to minimize average application service times and energy, while meeting a chip-wide dark-silicon power constraint (DS-Pc) and application performance and reliability constraints, in the presence of process variations. Our novel contributions are summarized below:

- We design a novel run-time application-mapping methodology for the emerging dark-silicon-constrained-design-regime, improving over traditional mapping approaches optimized for the area-constrained-design-regime;

- Our framework simultaneously manages all dynamically arriving applications while adapting application-DoPs to optimally utilize the system-power-slack (difference between DS-Pc and current system power dissipation);
- We design a novel heuristic to integrate within the application-mapping process a DVS mechanism that is constrained not just by performance but also by application-reliability requirements;
- Our combined mapping and DVS approach is also enhanced to take advantage of both D2D and WID variations, performing WID variation-aware mapping on to cores with the optimal power/performance characteristics, and D2D variation-aware chip-wide DVS where faster (leakier) chips would need lower V_{dd} levels and slower chips may run at higher V_{dd} levels.

6.2 Related Work

A few recent works have explored dark-silicon aware CMP design methodologies. Allred et al. [16] and Turakhia et al. [110] propose design-time frameworks for synthesis of heterogeneous CMPs to extract better energy-efficiency and performance in the dark-silicon regime. However, these works do not consider the effects of reliability and impact of process variations on CMP performance and power profiles. Raghunathan et al. [17] exploit the variation-profile for run-time application-mapping on to a homogeneous CMP die, to maximize performance and reduce leakage-power given a fixed dark-silicon power-budget. But the authors do not consider overheads of inter-core communication and application reliability requirements. Chou et al. [35] and Fattah et al. [116] consider run-time mapping of a queue of incoming applications, and propose mapping techniques to fit the maximum number of applications possible on a homogeneous CMP die, while minimizing inter-core communication distances. However, these approaches do not consider reliability and system power, and are geared towards traditional area-constrained designs.

Some recent works advocate varying the degree of parallelism (DoP) of multithreaded applications at run-time, to adapt to changes in the execution environment (power/performance-profiles, core-availability etc.) of a CMP while optimizing metrics such as power and energy-delay product (EDP). Given enough parallelism within the application, [14] showed that increasing DoP is a more energy-efficient way of boosting performance, compared to hiking the core-frequency/voltage values. Variable DoPs can be implemented by saving multiple versions of application-code at compile-time, and choosing the DoP at run-time that is most appropriate for the execution environment; or via more sophisticated techniques [111]. The variable-DoP run-time scheduling frameworks in [108] and [109] report improvements over scheduling techniques that employ statically fixed DoPs, and search for the best combination of voltage, frequency, number of cores, and number of threads, to optimize power and performance of a single application on a CMP. Our proposed VARSHA framework is different from these efforts in that we assume such information to be pre-profiled at design-time and the focus is on run-time management of application-DoPs in a multi-application power-constrained system. Besides, unlike VARSHA, the frameworks in [108], [109] consider neither the effects of process variations nor the design implications of system-reliability.

6.3 Motivational Example

In this section, we illustrate the advantages of some of the key aspects of our VARSHA framework with the help of a small example. We consider a scenario in which applications arrive at run-time at a service-queue to be served. The example assumes applications App-1 and App-2 arriving and being mapped on the CMP at time t=0s, and a third application App-3 arriving at t=1s.

Prior works such as [35], [116] search for square or near-convex regions on the CMP die to map arriving application at run-time, so that a maximum number of applications can fit on the die-area, while also minimizing inter-core communication distances. In this work, we note that in the modern dark-silicon era, performance is generally not limited by the chip-area but by the dark-silicon power-constraint (DS-Pc) of the chip. Therefore, while frameworks proposed in prior works would map App-1 and App-2 at t=0 as shown in figure 36(a) with a nominal DoP of 8, our framework, which adapts application-DoPs (app-

DoPs) to extract maximum system-performance (i.e., minimum application service-times) for a given DS-Pc, increases the DoP of App-1 to 16 (as shown in figure 36(b)). When App-3 arrives at t=1s, our framework maps it with a reduced DoP of 4 with a zero-wait time while meeting the DS-Pc of 60W (figure 36(b)), whereas [35] and [116] map App-3 at its nominal DoP of 8 (not shown), and require waiting until App-1 finishes at t=4.1s, so that the DS-Pc is not violated. Thus in our framework, app-DoPs are hiked from their nominal values opportunistically, to minimize runtimes, and app-DoPs are reduced to cut-down on wait times thereby minimizing average application service times (service time=runtime + wait time).

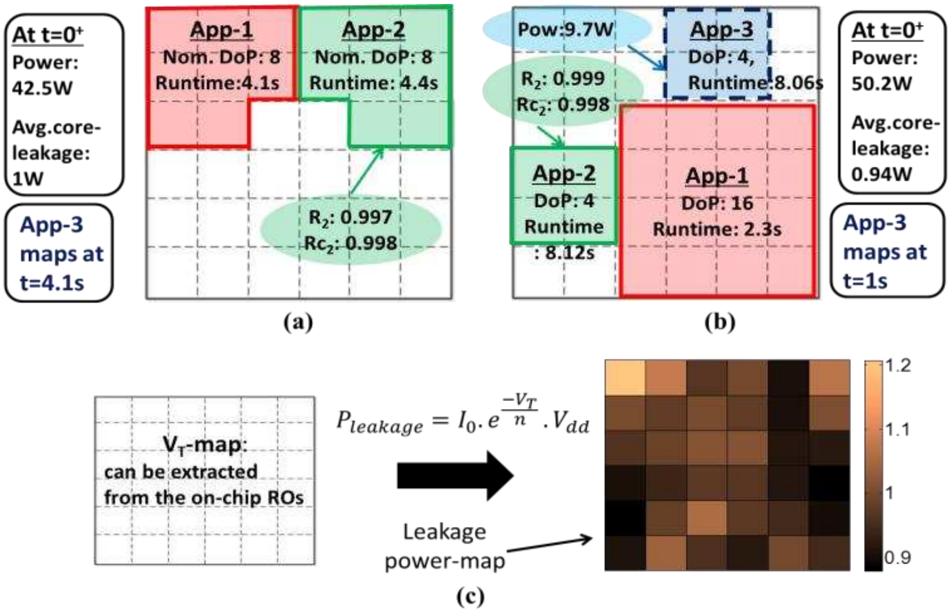


Figure 36: Motivational example using a 6×6 CMP where App-1 and App-2 are simultaneously mapped at time t=0 and App-3 arrives at t=1s. A DS-Pc = 60W is assumed: (a) Application-scheduling representative of prior works [35], [116]; (b) Application-scheduling obtained by our proposed VARSHA framework; (c) Extracted V_T-map and corresponding estimated leakage-power profile (in Watts) of the chip.

We define reliability of the i^{th} application as R_i = {1-Probability of one or more soft-errors during the execution of App-i}. In this work, we assume applications with different minimum reliability constraints running simultaneously on a CMP. Rc_i represents the reliability constraint of the i^{th} application. The application-reliability (R_i) primarily depends on V_{dd} and frequency of the cores (as shown in equations

(6.1)-(6.3) in section 6.4). Additionally, R_i depends upon the app-DoP – although application execution-time typically reduces with higher DoP (as long as the DoP is below an application-specific performance saturation point), the chip-area susceptible to soft-errors increases. Therefore, for fixed values of voltage and frequency, soft-error probability increases (i.e., application-reliability decreases) with increasing app-DoP. Our framework exhibits reliability-awareness by reducing the DoP of App-2 (with a stringent reliability-constraint) from the nominal value of 8 to 4, thereby meeting Rc_2 (figure 36(b)). However, Rc_2 is violated when the reliability-unaware frameworks [35], [116] are employed (figure 36(a)).

Furthermore, due to the variation-awareness of our framework, applications are mapped to regions with minimum core-leakage powers (see figure 36(c)). Therefore, the average leakage power per core is 1W for the variation-unaware frameworks (figure 36(a)), and 0.94W for our framework (figure 36(b)). Even though the initial allocation’s power dissipation is higher with our framework (at t=0, in figure 36), our lower service time and variation-aware mapping results in less overall energy consumption for the applications - 184J compared to 213J for [35], [116]. Our framework also employs an energy-saving mechanism to opportunistically scale V_{dd}-levels while meeting performance and reliability constraints of all applications. This feature is omitted from the above simple example for brevity.

6.4 Problem Formulation

6.4.1 Reliability Modeling

Computer systems are susceptible to both transient and permanent faults, the latter being caused by fabrication defects or wear-out. We only consider the impact of transient faults on reliability and do not consider permanent faults in this work. It is assumed that permanent faults could either be detected during the testing phase or are mitigated by hardware-redundancy techniques.

To model the dependence of raw soft error rate, raw-SER (λ), in a hardware component (core or router) on voltage and frequency values, we use the relationship proposed in [11]:

$$\lambda(\omega_j) = \lambda_0 . 10^{\frac{d.(1-\omega_j)}{1-\omega_{min}}} \qquad \text{..... (6.1)}$$

where λ_0 is the SER corresponding to the highest voltage and frequency values (ω_{max}), and ω_j is the average of the normalized values of the jth combination of voltage and frequency (such that ω_{max}=1). For any compute-core, we use a value of 10^{-6} errors/sec for λ_0 and assume d=3 as in [104], [106]. However, we use $\lambda_0 = 10^{-6}/3$ for NoC routers, given that our router-area is roughly a third of the area of a compute-core. The reliability of a core or a router is given by:

$$\mathbb{R}(\omega_j) = e^{-\lambda(\omega_j).\tau} \qquad \text{..... (6.2)}$$

where τ is the execution time of the component (time-duration that the component stays active). Assuming n = app-DoP, reliability of the ith application running on a CMP can be given as the product of all 2*n* (*n* routers and *n* compute-cores) component-reliabilities:

$$R_i = \prod_{2n} \mathbb{R}(\omega_j) \qquad \text{..... (6.3)}$$

Prior works [8], [9] have shown that at technology nodes of 32nm and below, process variations have almost no effect on SER. Therefore, in this work, we assume no dependence of V_T-variations on SER, instead exploiting variations for speed/power benefits only.

6.4.2 Inputs, Assumptions, and Problem Objective

We assume the following inputs to our problem:

- A CMP with a regular mesh-based 2D network-on-chip (NoC), with T tiles: $T = (d^2)$, where d is the mesh dimension, and each tile consists of a compute core and a NoC router;
- A set S of candidate supply voltage (V_{dd}) levels for the chip;
- A set of N V_T-maps (N test-chips) incorporating the effects of WID and D2D variations with continuous distribution over the die; the estimated leakage power-profile for a given value of V_{dd} can be obtained from a V_T-map (using the relation shown in figure 36).

- Application sequence s of length ℓ, made up of η different applications, with arbitrary application inter-arrival times;
- Application task graphs for the set $P = \{P_1, P_2, \ldots P_\eta\}$ of DoPs for all applications; an application i possesses $|P_i|$ viable DoPs; an application has a maximum DoP value beyond which performance does not improve (or gets worse) - such DoP values are ignored;
- Vertices of each task-graph with execution-times of compute cores and edges with inter-task communication volumes; execution time and volume values are assumed available from offline profiling;
- Energy-optimal frequency constraint $\{f_1, f_2, \ldots, f_\eta\}$ and minimum reliability-constraints $\{Rc_1, Rc_2, \ldots, Rc_\eta\}$ for all η applications;
- A chip-wide dark-silicon power dissipation constraint (DS-Pc).

We make the following assumptions in our work:

- There exists one-to-one mapping between tasks and cores;
- Applications are mapped contiguously on non-overlapping rectangular regions of the CMP die, for inter-application isolation;
- All cores executing an application run at the same frequency, to avoid imbalances during multi-threaded execution [17];
- Variation-map data for a chip is available at run-time, in terms of the threshold-voltage (V_T) distribution, from the chip-frequency-profile obtained using ring-oscillator based delay sensors [113];
- A chip wide supply voltage that can be scaled using DVS at run-time, as the overheads of implementing DVS at a per–core granularity are deemed to be prohibitively expensive [115].

Problem Objective: Given the above inputs and assumptions, our objective is to perform run-time application-scheduling and DVS on a given CMP platform such that the average application service-time and average energy (across all N test-chips) are minimized, while all application-specific operating frequency- and reliability-constraints, as well as CMP platform-specific DS-Pc are satisfied.

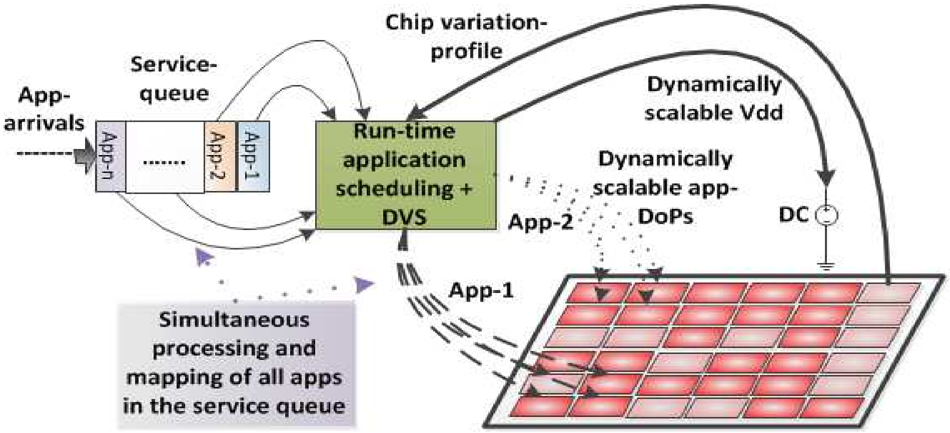
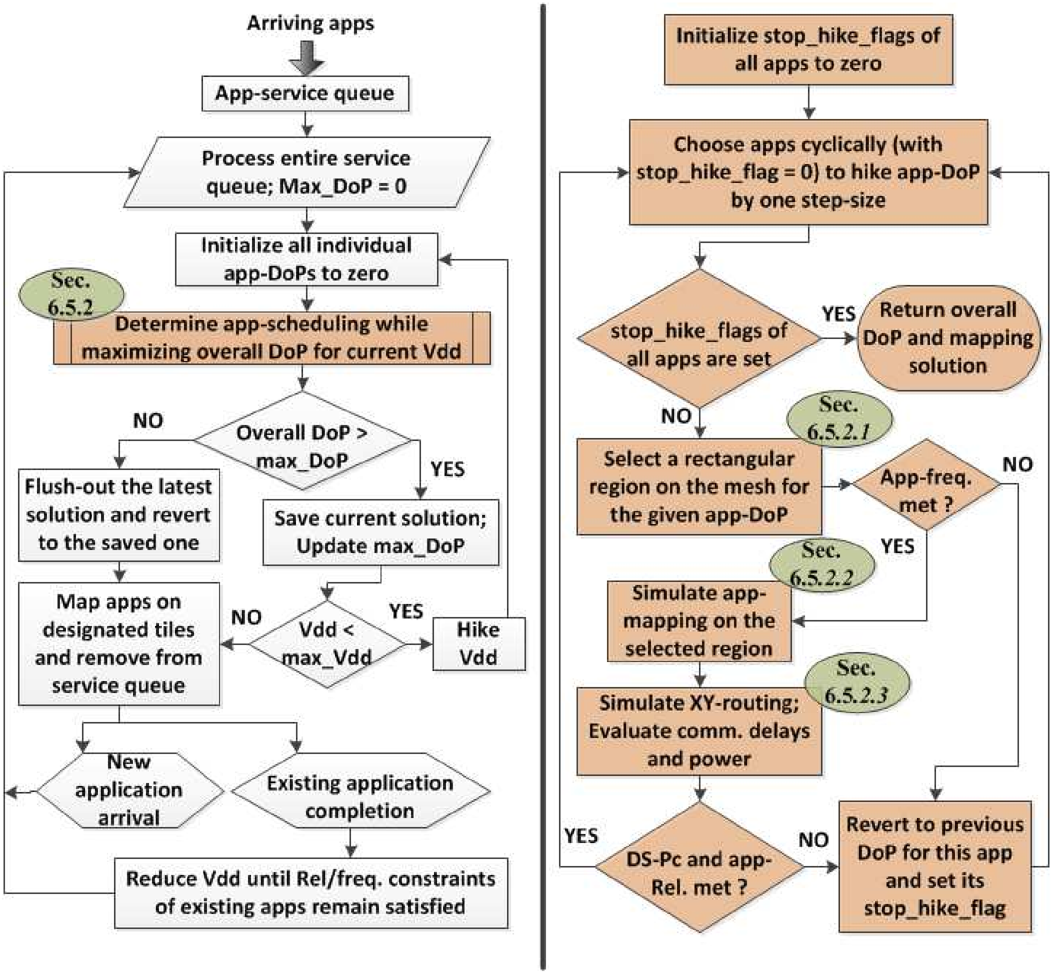

Chip variation-profile
Service-queue
App-arrivals
App-n
....
App-2
App-1
Run-time application scheduling + DVS
Dynamically scalable Vdd
Dynamically scalable app-DoPs
App-2
DC
App-1
Simultaneous processing and mapping of all apps in the service queue

Figure 37: Overview of our application-scheduling + DVS framework

6.5 VARSHA Framework: Overview

Figure 37 illustrates the key aspects of our proposed VARSHA framework. The knowledge of the chip-variation profile is continuously utilized in the scheduling and DVS steps. Assuming equal priority for all incoming applications, the application-scheduling step consists of *(i)* determining the DoP (out of the $|P_i|$ DoPs) for each waiting application in the service-queue, and *(ii)* mapping the appropriate task-graphs on to the tiles of the CMP. For a given V_{dd}, scaling-up of application-DoPs (*app-DoPs*) is constrained by the available power-slack (difference between DS-Pc and current system-power), application-reliability constraints, and the available tiles meeting the application-frequency constraints. At any given time, the scaling-down of V_{dd} (to save power/energy) is constrained by the frequency and reliability-constraints of the applications running on the CMP, whereas scaling-up of V_{dd} (to boost *app-DoPs*) is constrained by the DS-Pc for the CMP. The VARSHA framework is effectively executed in two nested procedures: *(i)* V_{dd}-level selection (outer loop), triggered on an arrival or a departure of any application; and *(ii)* determination of application-schedule for the current V_{dd}-level (inner loop). These procedures are discussed in detail in sections 6.5.1 and 6.5.2 respectively, and the corresponding design-flows for the procedures are shown in figure 38(a) and figure 38(b), respectively.


Arriving apps
App-service queue
Process entire service queue; Max_DoP = 0
Initialize all individual app-DoPs to zero
Sec. 6.5.2
Determine app-scheduling while maximizing overall DoP for current Vdd
Overall DoP > max_DoP
NO
YES
Flush-out the latest solution and revert to the saved one
Save current solution; Update max_DoP
Map apps on designated tiles and remove from service queue
Vdd < max_Vdd
NO
YES
Hike Vdd
New application arrival
Existing application completion
Reduce Vdd until Rel/freq. constraints of existing apps remain satisfied
Initialize stop_hike_flags of all apps to zero
Choose apps cyclically (with stop_hike_flag = 0) to hike app-DoP by one step-size
stop_hike_flags of all apps are set
YES
Return overall DoP and mapping solution
NO
Sec. 6.5.2.1
Select a rectangular region on the mesh for the given app-DoP
App-freq. met ?
NO
YES
Sec. 6.5.2.2
Simulate app-mapping on the selected region
Sec. 6.5.2.3
Simulate XY-routing; Evaluate comm. delays and power
YES
DS-Pc and app-Rel. met ?
NO
Revert to previous DoP for this app and set its stop_hike_flag

(a) (b)

Figure 38: Design-flows for the VARSHA framework: (a) Vdd-level selection (discussed in section 6.5.1); (b) Determination of application-schedule for the current Vdd-level (discussed in section 6.5.2).

6.5.1 V_{dd}-level Selection

To extract maximum performance from the applications being considered for mapping at any time instant, the first-order objective in VARSHA is to maximize overall DoP of the system (sum-total of all *app-DoPs*). Recall that an application typically has a maximum viable DoP, and higher DoPs can cause performance to degrade (due to high synchronization overheads) – such higher DoP configurations are ignored (section 6.4.2). Our V_{dd}-selection heuristic (figure 38(a)) selects the V_{dd}-level that yields the maximum overall DoP. As a second-order power/energy saving objective, on completion of any application, our framework reduces V_{dd} to the lowest allowable level that would not introduce any violations in frequency and reliability constraints of existing (already running) applications.

We assume all incoming applications are buffered in a service-queue. On arrival or completion of any application, the V_{dd}-selection heuristic is triggered, which processes the entire service-queue. The V_{dd}-selection heuristic iteratively invokes the application-schedule determination procedure (discussed in section 6.5.2), which produces the mapping-solution with the highest overall DoP corresponding to the current V_{dd}-level. The V_{dd}-level is hiked (in increments of 0.1V) until either the overall DoP reduces, in which case the immediately preceding solution with the highest overall DoP is reverted to, or the maximum allowable V_{dd}-level (max_V_{dd}) is reached. Note that the overall DoP may increase with increasing V_{dd}-levels, as more applications satisfy frequency and reliability constraints for higher DoPs; at the same time, the chip-power will reach the DS-Pc quicker at higher V_{dd}-levels, thereby limiting overall DoP. Therefore, our search for the optimal V_{dd}-level culminates when the increase in overall DoP is limited by the DS-Pc. Finally, the best application-mapping solution with the highest overall DoP is mapped to the CMP. The voltage supply is hiked to the selected V_{dd}-level, and the mapped applications are then removed from the service-queue.

6.5.2 Determination of Application-schedule

Given a specific execution environment for the CMP (including the V_{dd}–level, available power-slack, and variation-profile), the objective of the application-schedule determination heuristic is to maximize the overall DoP, while simultaneously considering all applications in the service-queue and satisfying application-frequency and -reliability constraints. Figure 38(b) shows the design-flow of this heuristic. Starting at the least possible DoP value of zero (DoP of zero leaves the application unmapped at the current time) applications are considered cyclically for hiking of DoP to their next higher valid DoP-level. Here, to extract maximum performance from the CMP, we choose applications for hiking of DoP in order of their compute-intensiveness, because of the relatively smaller communication delay and power overheads for compute-intensive applications at higher DoPs. Also, we hike *app-DoPs* symmetrically across all applications because execution of an application is generally more energy-efficient at lower DoPs (due to lower parallelization- and communication-overheads). For instance running four applications simultaneously, each with DoP=4, is typically more energy-efficient than running them one after the other with DoP=16 each.

To produce an optimal mapping for the application under consideration (with a specific DoP), the following three steps are performed: *(i)* rectangular-region selection on the CMP for the current DoP (section 6.5.2.1); *(ii)* mapping of the corresponding task-graph to the selected region (section 6.5.2.2); *(iii)* communication-flow routing and delay/power analysis (section 6.5.2.3). After the above steps, the mapping is evaluated for overall power footprint and reliability of the applications. Satisfaction of the application-frequency constraints are checked during the rectangular-region-selection step. As shown in figure 38(b), DoP-hike of any application could fail due to potential violation(s) in application-frequency and –reliability constraints or DS-Pc. When an attempted DoP-hike is stalled for any application, its *stop_hike_flag* is set to preclude it from future DoP-hike consideration, and the feasible mapping with the preceding DoP is finalized for this application.

6.5.2.1 Rectangular Region Selection

We consider application-mapping on rectangular regions, of pre-determined dimensions corresponding to each possible DoP value in set *P*. Therefore, all intra-application communication is contained within a rectangular region. This provides inter-application isolation, and eliminates communication cross-interference overhead. Given the V_T-map, the maximum frequency that each core can be reliably clocked at depends upon the V_T and V_{dd} values, given by:

$$f_{max} = \frac{\mu(V_{dd} - V_T)^{\alpha}}{C_0 . V_{dd}} \quad \textbf{.....} \text{ (6.4)}$$

where α and μ are technology-dependent constants, and C_0 is switching capacitance of the critical path [98]. In our region selection method, we utilize the knowledge of both frequency and leakage-power profiles of the chip. Note that dynamic-power remains unaffected by V_T-variations and is thus not considered in this step. Our objective is to find the region on the mesh (of pre-defined dimensions) that dissipates the least total leakage-power and all cores within which satisfy the frequency-constraint of the application being mapped. To this end, we perform a simple exhaustive search over all tiles on the mesh as the left-upper corner of the given rectangular region. If the rectangle is not a square, then both of its orientations need to be checked to find the optimal rectangular region. Figure 39 shows an illustration of our region-selection method for a 4×4 mesh.

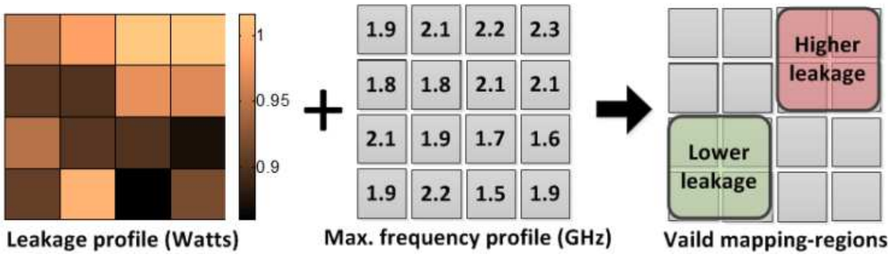


Figure 39: Two feasible regions are shown for the application frequency constraint of 1.9 GHz. Our method chooses the one with lower leakage.

Theoretical time-complexity analysis: At most *T* tiles (total tiles on the chip) are considered for the prospective rectangular region. For non-squares, e.g., a 2×4 rectangle, rectangles of both orientations need to be evaluated at each tile. Note that *app-DoP* (relatively small integer *c* – treated as constant) number of tiles are to be evaluated for frequency and leakage-power at each of these iterations. This gives a linear-time complexity for the region-selection step: O(2*cT*).

6.5.2.2 Application Mapping

After the rectangular region (of size equal to app-DoP) on the mesh has been selected, our mapping heuristic maps the appropriate application-task-graph on to the CMP tiles. We employ an incremental-mapping approach that is carried out in two steps:

1) Starting with the task with highest communication-volume, list tasks in decreasing order of inter-task communication volumes with all preceding tasks in the list;

2) Map tasks on the selected region, in the order of this list – starting with one of the center tiles on the rectangular-region. To map any task, select the tile that would incur least traffic foot-print with the already mapped tasks. We define communication-traffic foot-print of a core (mapped task) as: $\sum MD_j \times vol_j$, where vol_j is the communication-volume of the j^th^ flow entering or leaving the core and MD_j is the Manhattan distance between the source and destination of the j^th^ flow.

Theoretical time-complexity analysis: Step (1) sorts *app-DoP* tasks – the time-complexity could be bounded by O((*app-DoP*)^2). In step (2), *app-DoP* number of tasks are mapped and each could consider a maximum of *app-DoP* number of tile-locations. Thus this step could be loosely bounded by O((*app-DoP*)^2). As *app-DoP* is a relatively small integer, it can be considered a constant (between 4 and 16 in our experiments). Thus, our application-mapping has constant time-complexity. Note that mapping-information can potentially even be saved with the task-graph information at compile-time, thus saving valuable run-time resources and overhead.

6.5.2.3 Communication Delay and Power Estimation

Similar to numerous prior works such as [116], we use the XY-routing scheme to route the communication-flows of applications. The communication-delays with congestion-overheads are calculated from the application-frequency (routers and links run at the rated application-frequency) and link BWs. Profiling of compute- and communication-delays could potentially be performed at design-time. Dynamic powers of routers and links (corresponding to different voltages, frequencies, communication-loads, and router-sizes), are assumed to be saved in the read-only (or non-volatile) memory on the CMP die. Router leakage powers, estimated from the chip-variation-profile, are added to the appropriate dynamic power values to produce the total communication-power for running each application. Based on the active-times (execution-times) of routers and compute-cores, the application-reliability is computed using equations (6.1)-(6.3). For our analyses, the energies and run-times of applications are calculated from component powers and active-times.

6.5.3 Complexity Analysis of VARSHA Framework

Our application schedule determination heuristic (discussed in section 6.5.2), which maximizes the DoP of all applications being processed at the current V_{dd}-level, attempts to find mapping solutions for the *w* waiting applications for up to $|P_i|$ DoP-levels. This heuristic could be required to execute for at most $|S|$ (total number of candidate V_{dd}-levels) times. $|S|$ and $|P_i|$ are small constant integers in our experiments. Also, at any given time, only a small number of applications (up to *w*) are generally expected to be processed given the DS-Pc. Therefore, whenever the service-queue is processed in our VARSHA framework, the number of times that our region-selection heuristic would need to be invoked is bounded by a relatively small constant integer. As the region-selection step, which has the highest theoretical time-complexity in the VARSHA design-flow, finishes in linear-time (as discussed in section 6.5.2.1), the time-complexity of our framework is linear, with respect to the CMP mesh-size, *T*.

6.6 Experiments

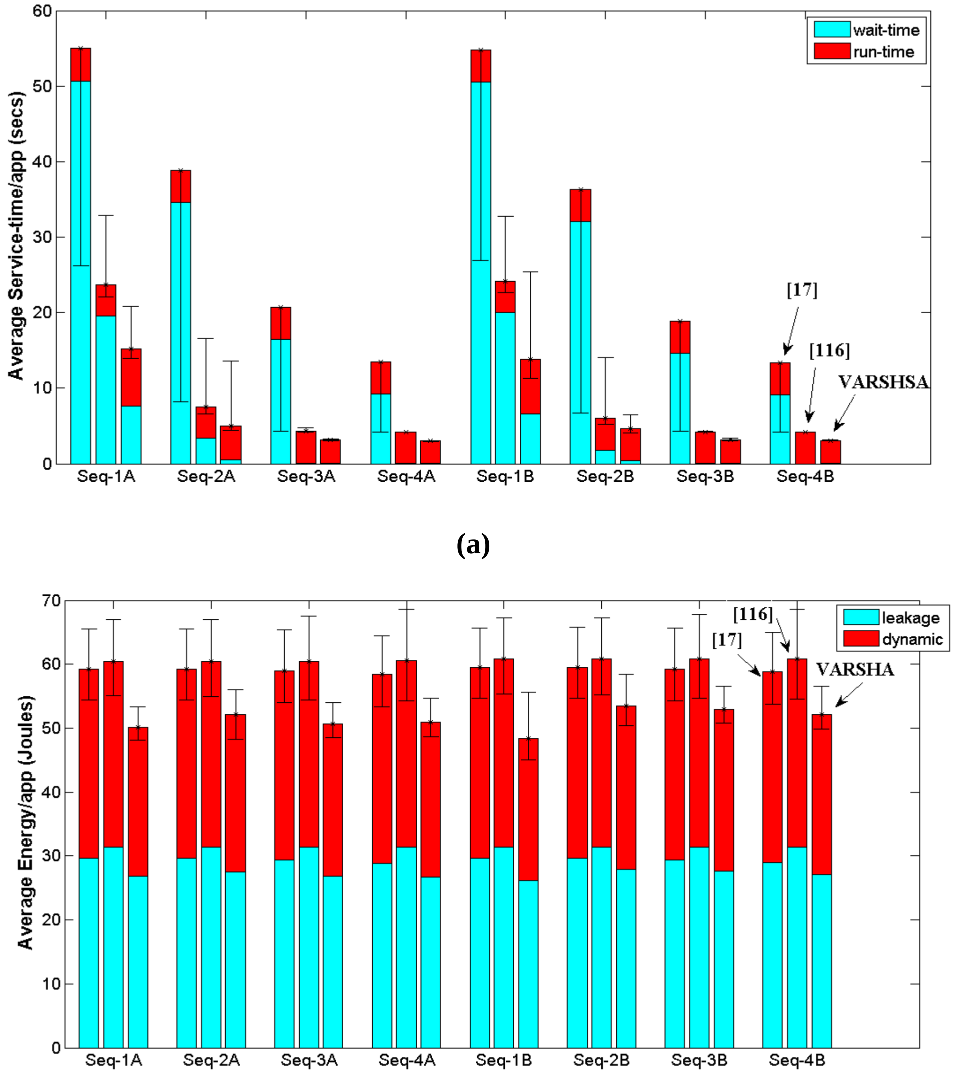
Our experiments were conducted using η=14 different parallel application benchmarks: seven from the SPLASH-2 benchmark suite [58] (*cholesky, fft, lu, ocean, radix, radiosity, and raytrace*), and seven from the PARSEC benchmark suite [56] (*vips, swaptions, fluidanimate, dedup, streamcluster, canneal, and blackscholes*). We consider DoPs that are multiples of 4, up to 16, where a DoP of 8 is considered the nominal DoP value, as a reasonable trade-off between speed and energy. As discussed earlier, every application has a unique maximum viable DoP beyond which further performance gains cannot be achieved. Our task-graphs for all applications at different DoPs are modeled based on the system-traces generated by running topology-agnostic gem5 simulations [117]. The reliability constraints of different applications are set in the range: 0.99 to 0.999. We assume the ARM Cortex-A9 processors [37] as the baseline CMP compute cores, which support five operating voltage levels ($|S|$=5): 0.8V, 0.9V, 1.0V, 1.1V, and 1.2V. The application-specific energy-optimal core frequencies range from 1300 MHz to 1900 MHz, based on the level of compute intensity of the tasks assigned to cores. We use a 100-core mesh for the CMP platform with dimensions 10×10. The dark-silicon power-constraint (DS-Pc) is set at 100W. The delay-overhead for V_{dd}-scaling (PLL lock time) is estimated to be less than 5μs, similar to [105], which is negligible compared to the granularity of application-runtimes (2-9 seconds each) in our experiments.

To investigate the applicability of our approach to CMP dies with diverse variation-profiles, we use 1000 test-chips (N=1000), in our experiments. The 1000 V_T-maps are generated using the open-source tool [97] (based on systematic and random WID-variation model in [98]). The values of 0.3 and 0.09 are used for the statistical mean and a standard deviation of the parameter V_T respectively, and a correlation range (ϕ) of 0.5 is used (as recommended in [98]). A normally distributed V_T bias, representing the D2D variation component is superimposed onto these V_T-maps; the standard deviation of the D2D V_T is assumed to be 6%, as in [99]. Each V_T-map represents a 30×30 grid of points corresponding to 9 ring oscillator test-sites for each core on the CMP. For a given V_T-map, the maximum core-frequencies are calculated by using the V_T-max values and the core-leakage powers are calculated using the V_T-avg values (out of the 9 V_T values per core). The power values of routers and links (32-bit wide) for different voltages and frequencies at varying communication loads, for the 32 nm technology node are obtained from ORION 2.0 [40]. Note that the router power values obtained are for nominal V_T, and are scaled for varying V_T values.

6.6.1 Results

We compare the results obtained from our VARSHA framework with those obtained from using run-time application mapping frameworks proposed in recent prior works [17] and [116]. A variation- and dark-silicon-aware mapping technique is proposed in [17], whereas [116] advocates for a traditional area-constrained design approach. We implemented these prior works to the best of our understanding. Our experiments considered two unique application-sequences (Seq-A and Seq-B) that represent an ordering of arriving application instances, with instances randomly chosen from among the 14 applications considered. For each sequence, we vary the inter-arrival times of application-instances randomly within the following ranges: 0 to 1 seconds (Seq-1A and Seq-1B), 0 to 2 seconds (Seq-2A and Seq-2B), 0 to 4 seconds (Seq-3A and Seq-3B), and 0 to 8 seconds (Seq-4A and Seq-4B). We assume ℓ=100 application-instances in any application-sequence. Also, our results (as shown in figure 40 and Table 11) show the mean-values across a 1000 test-chips.

The prior works [17] and [116] assume fixed nominal app-DoPs. Our framework adapts app-DoPs in accordance with the application inter-arrival rates to minimize the application service-times. Observe in Table 11 that for both sequences, the average app-DoP reduces with increasing inter-arrival-rates for VARSHA. At higher inter-arrival rates when applications with nominal DoPs cannot be quickly serviced due to the DS-Pc constraint, our VARSHA framework cuts down application wait-times significantly by reducing DoPs (as shown in figure 40(a) - Seq-1A,B and Seq-2A,B), although the application run-times tend to increase due to the reduction in DoPs. On the other hand, at lower inter-arrival rates, with on average fewer applications to be serviced simultaneously, our framework opportunistically hikes the application-DoPs to minimize run-times (as shown in figure 40(a) - Seq-3A,B and Seq-4A,B). In comparison with [116], we obtain 27%-43% savings in average service-times with our VARSHA framework. Note that maximum savings are obtained when the inter-arrival-rates are most stringent, as shown for Seq-1A and Seq-1B in figure 40(a). The communication-unaware framework in [17] maps applications on to large rectangular regions of non-contiguous tiles, resulting in longer run-times due to longer communication-latencies. Compared to [17], we obtain 70% - 87% savings in average service-times with our VARSHA framework.



(a)

(b)

Figure 40: Results of VARSHA framework vs. [17] and [116]: (a) Average service-time per application-instance (wait-time + run-time); (b) Average energy per application-instance (leakage + dynamic). The bars represent mean values of service-times and energies across 1000 test-chips, while variation in service-times and energies is shown by confidence intervals.

Table 11: Mean values across all 1000 test-chips for average DoP per application-instance and weighted-average Vdd for VARSHA framework

	Seq-1A	Seq-2A	Seq-3A	Seq-4A	Seq-1B	Seq-2B	Seq-3B	Seq-4B
DoP	4.5	10.3	14.1	14.5	5.1	10.7	13.7	14.2
$V_{w\text{-}avg}$	1.09	1.10	1.07	1.07	1.07	1.11	1.09	1.07

Our framework also opportunistically lowers V_{dd}-levels while ensuring that frequency and reliability constraints remain satisfied for the set of existing applications. The weighted-average V_{dd} metric ($V_{w\text{-}avg}$) represents the average value of V_{dd} throughout the execution of the entire application-sequence for a given chip, and is defined as:

$$V_{w-avg} = \frac{((V_{\Delta t1} \times \Delta t1) + (V_{\Delta t2} \times \Delta t2) + \cdots + (V_{\Delta tp} \times \Delta tp))}{\sum_{i=0}^{i=p} \Delta ti}$$

where, Δti is the i^{th} time-interval during which the V_{dd}-level stays constant at a value of $V_{\Delta ti}$, and the execution of the entire application-sequence *s*, takes *p* such time-intervals. Table 11 shows the mean $V_{w\text{-}avg}$ across all 1000 test-chips. In the absence of DVS in [17] and [116], in our implementation of these frameworks, we assume the highest voltage of 1.2V, which is just high enough to meet all application-frequency constraints for the 1000 test-chips. Despite the high V_{dd} and much longer communication paths, [17] saves leakage-power by performing variation-aware mapping. In the absence of such variation-awareness, [116] consumes the highest energy (as shown in figure 40(b)). In our VARSHA framework, energy-efficiency decreases with increasing app-DoPs (due to increased communication and sub-linear increase in computation-performance), while decreasing V_{dd}-levels generally cause an increase in energy-efficiency. Therefore, in figure 40(b), we observe variation in energy values with different inter-arrival rates. The maximum energy savings are obtained for the most stringent inter-arrival rates, as shown for Seq-1A and Seq-1B in figure 40(b), due to opportunistic reduction of DoPs. We find from figures 40(a), (b) that VARSHA produces on average, **13%** and **15%** savings in mean energy (8% and 14% savings in mean leakage energy) and, **80%** and **35%** savings in mean application service time (93% and 66% savings in mean application wait-times), over [17] and [116] respectively.

Our experiments indicate that for the frameworks from [17] and [116], where a fixed high value of V_{dd} is used, slower chips (usually with lesser leakage) could prove to be advantageous as they would be left with greater power-slack to accommodate more applications at any given time (less %dark-silicon), thus resulting in better service-times and energies. Conversely, in our VARSHA framework, given a DS-Pc, faster chips (that also dissipate higher leakage power) can usually utilize lower V_{dd}-levels to minimize energy, and slower chips (that dissipate lower leakage power) can utilize higher V_{dd}-levels to improve performance. We observed that test-chips with moderate leakage and performance profiles produce desirable service-times and energy results, whereas chips that are too leaky or too slow yield worse results, as indicated by the upper-bounds of the confidence-intervals in figure 40(a) and 40(b) for the VARSHA framework.

Table 12: Reliability-constraint (Rc) violations (average per sequence)

	Seq-A	Seq-B
[17]	8	9.5
[116]	6	8

For applications with relatively more stringent reliability-constraints, it may not be possible to support nominal-DoP even at the highest V_{dd}-level (1.2V). Therefore, when using reliability-unaware frameworks with no DoP-adaptivity, reliability-constraints (Rc) of such applications may be violated. Table 12 shows the Rc-violations across 1000 test-chips for schedules produced by [17] and [116]. With longer routing distances, [17] consumes far more routing resources compared to [116], thus even with the same app-DoPs (nominal DoPs), the estimated failure-rates are greater for [17]. Our reliability-aware VARSHA framework, results in no Rc-violations because of its ability to dynamically reduce app-DoPs as well as hike V_{dd}-levels in accordance with application reliability-requirements.

6.7 Conclusion

VARSHA represents one of the first efforts to integrate reliability and variation-awareness in a run-time variable degree-of-parallelism (DoP) application-scheduling methodology to enhance performance of multi-core systems in the new dark-silicon era. Tailored for deeply scaled technologies, our proposed VARSHA framework generates highly optimized application-mapping solutions, while exhibiting linear run-time complexity. Our experimental results show that VARSHA produces savings of **35%-80%** in application service-times, **13%-15%** in energy, and avoids reliability violations unlike state of the art prior works that suffer from reliability violations in up to **11%** of application instances arriving at run-time.

7 A Runtime Framework for Robust Application Scheduling with Adaptive Parallelism in the Dark-Silicon Era

With deeper technology scaling accompanied by a worsening power-wall, an increasing proportion of chip area on a chip multiprocessor (CMP) is expected to be occupied by dark-silicon. At the same time, design challenges due to process variations and soft-errors in integrated circuits are projected to become even more severe. It is well known that spatial variations in process parameters introduce significant unpredictability in the performance and power profiles of CMP cores. By mapping applications on to the best set of cores, process-variations can potentially be used to our advantage in the dark-silicon era. Additionally, the probability of occurrence of soft-errors during execution of any application has been found to be strongly related to the supply voltage and operating frequency values, thus necessitating reliability awareness within run-time voltage scaling schemes in contemporary CMPs. In this chapter, we present a novel framework that leverages the knowledge of variations on the chip to perform run-time application mapping and dynamic voltage scaling to optimize system performance and energy, while satisfying dark-silicon power-constraints of the chip as well as application-specific performance and reliability constraints. Our experimental results show average savings of 10%-71% in application service-times and 13%-38% in energy consumption, compared to prior work.

7.1 Introduction

With increasing transistor miniaturization, circuit densities have drastically increased, and the critical charge, which is the minimum charge capable of a bit-flip in a memory- or a logic-cell, has significantly decreased [8]-[10]. This phenomenon has caused newer process technologies to be more susceptible to transient-faults due to the effects of radiation, e.g., alpha-particle and neutron strikes. The rate of such transient-faults at run-time has thus been increasing with technology scaling [11]. Studies have also shown that hardened flip-flops are only 30%-50% more resilient than unprotected ones, at sub-40nm nodes [107], i.e., circuit-level hardening may not sufficiently suppress soft-errors. Thus, system-level run-time approaches to cope with transient faults and complement circuit-level techniques will be increasingly essential in newer process technologies.

Simultaneously, unpredictability in leakage power and circuit-delay due to variability in modern fabrication processes has become a serious concern. In emerging chip-multiprocessors (CMPs), spatially correlated systematic within-die (WID) variations manifest across multiple cores, creating core-to-core (C2C) variations [86]. At the same time, die-to-die (D2D) variations remain quite significant [87]. Both WID and D2D variations have random and systematic components. Although several prior works such as [92], [95], [136] have proposed variation-aware design-time application-mapping frameworks, it is generally quite difficult to predict the variation-profile of a fabricated chip at design-time [112]. It is however possible to extract the variation-map of a chip at run-time from the chip-frequency-profile obtained using ring-oscillator based delay-sensors [113], [114]. Thus, run-time approaches to mitigate adverse effects of process variations become possible, and will be vital for scaled technologies.

The slowdown of power scaling with technology scaling, due to leakage and reliability concerns [14], [15], has led to a rise in chip power-densities, and created the *dark-silicon* phenomenon – a significant fraction of the chip needs to be shut-down (i.e., “dark”) at any given time to satisfy the chip power-budget. With the extent of dark-silicon increasing every technology-generation (30%-50% for 22nm) [16], [17], designs are becoming increasingly power-limited rather than area-limited. Run-time power-saving techniques such as dynamic voltage scaling (DVS) are thus becoming increasingly important.

Given these multiple daunting design challenges, there is a critical need for a system-level solution that can simultaneously and adaptively manage the constraints imposed by dark silicon, process variations, and soft-error reliability, while executing applications. In this chapter, we address this need by proposing a novel run-time variation- and reliability-aware application scheduling framework that employs dynamically adaptable application degrees of parallelism (DoPs) to minimize average application service times and energy, while meeting a chip-wide dark-silicon power constraint (DS-Pc) and application performance and reliability constraints, in the presence of process variations. Our novel contributions in this work are:

- We design a novel run-time application-mapping framework for the emerging dark-silicon-constrained-design-regime, improving over traditional mapping approaches that are typically optimized for the area-constrained-design-regime;
- Our framework simultaneously manages all dynamically arriving applications while adapting application-DoPs to optimally utilize the system-power-slack (difference between DS-Pc and current system power dissipation);
- We design a novel heuristic to integrate within the application-mapping process a DVS mechanism that is constrained not just by performance but also by application-reliability requirements;
- The traditionally disjoint design steps of region-selection and task-to-tile mapping are seamlessly integrated in our framework to co-optimize memory-communication and leakage-power while exhibiting awareness of the run-time CMP-environment;
- Our combined mapping and DVS approach is also enhanced to take advantage of both D2D and WID variations, performing WID variation-aware mapping on to cores with the optimal power and performance characteristics, and D2D variation-aware chip-wide DVS where faster (leakier) chips would need lower V_{dd} levels and slower chips may run at higher V_{dd} levels.

7.2 Related Work

A few recent works have begun to focus on dark-silicon aware design methodologies for emerging CMPs. Allred et al. [16] and Turakhia et al. [110] propose design-time frameworks for the synthesis of heterogeneous CMPs to extract better energy-efficiency and performance in the dark-silicon regime. However, these works do not consider the effects of reliability and the impact of process variations on

CMP performance and power dissipation. Raghunathan et al. [17] exploit the variation-profile for run-time application-mapping on to a homogeneous CMP die, to maximize performance and reduce leakage-power given a fixed dark-silicon power-budget. But the authors do not consider overheads of inter-core communication or application reliability requirements. Chou et al. [35] and Fattah et al. [116] consider run-time mapping of a queue of incoming applications, and propose mapping techniques to fit the maximum number of applications possible on a homogeneous CMP die, while minimizing inter-core communication distances. However, these approaches do not consider reliability and system power dissipation, and are geared towards traditional area-constrained designs.

Several other efforts (e.g., [104], [106], [137]) have proposed design-time fault-tolerant scheduling frameworks that assume fault-detection mechanisms implemented on the multi-core platform. Hardening techniques such as task-re-execution and replication are utilized in these works to probabilistically meet task-completion deadlines for real-time tasks of varying criticalities. However, these efforts do not consider dark silicon and process variation challenges, run-time adaptation support, or a dynamically parallel application workload.

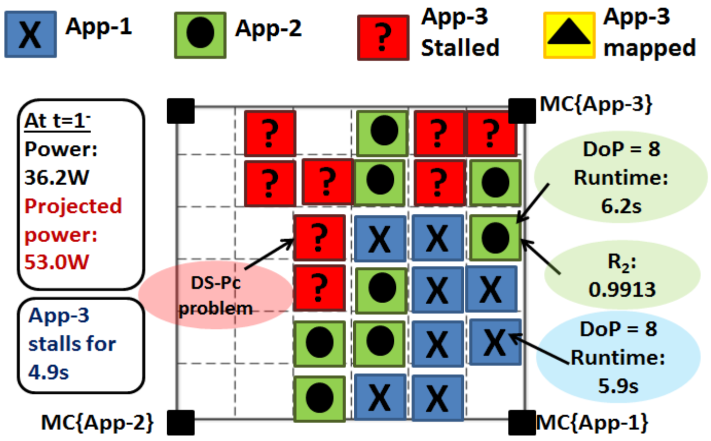
Some recent works advocate varying the degree of parallelism (DoP) of multithreaded applications at run-time to adapt to changes in the execution environment (power/ performance-profiles, core-availability etc.) of a CMP while optimizing metrics such as power and energy-delay product. Given enough parallelism within the application, [14] showed that increasing DoP is a more energy-efficient way of boosting performance, compared to hiking the core-frequency/voltage values. Variable DoPs can be implemented by saving multiple versions of application-code at compile-time, and selecting the DoP at run-time that is most appropriate for the execution environment; or via more sophisticated techniques [111]. The variable-DoP run-time scheduling frameworks in [108] and [109] report improvements over scheduling techniques that employ statically fixed DoPs, and search for the best combination of voltage, frequency, number of cores, and number of threads, to optimize power and performance of a single application on a CMP. Our proposed framework is different from these efforts in that we assume such information to be pre-profiled at design-time and the focus is on run-time management of application-DoPs in a multi-application power-constrained system. Moreover, unlike the frameworks in [108] and [109], our work also considers the effects of process variations as well as the design implications of system-reliability.

More recently, a few works have proposed reliability- and variation-aware run-time application mapping and dynamic voltage scaling frameworks for multicore systems. M. Saheli et al. [138] advocates using different pre-compiled code-versions for every task with varying levels of reliability, and maximizes the reliability of the currently mapped task given the power budget available at run-time. But this work does not consider multithreaded applications or DoP adaptation, nor does it consider any inter-task communication. Our prior work, VARSHA [139], aims to minimize average application-service times by performing DoP adaptations and exhibiting awareness of the run-time CMP-environment (variation-profile, application-reliability/performance, and dark-silicon). This framework performs application mapping on contiguous and non-overlapping rectangular regions on the CMP-die. As in most prior works, VARSHA assumes that the traffic generated in the network-on-chip (NoC) due to accesses to off-chip memory, i.e., communication-traffic between compute-cores and on-chip memory-controllers (MCs), is negligible. Under such an assumption, mapping applications within rectangular regions could simplify design and enable communication-isolation. However, such an approach that restricts application-mapping onto rectangular regions exclusively has the following disadvantages: (i) the scope of power optimization is restricted while selecting cores that dissipate lower leakage power and meet performance constraints; and (ii) performing region selection without the awareness of memory-traffic could potentially result in much longer communication routing paths to MCs, which in turn would induce network congestion and increase communication latencies.

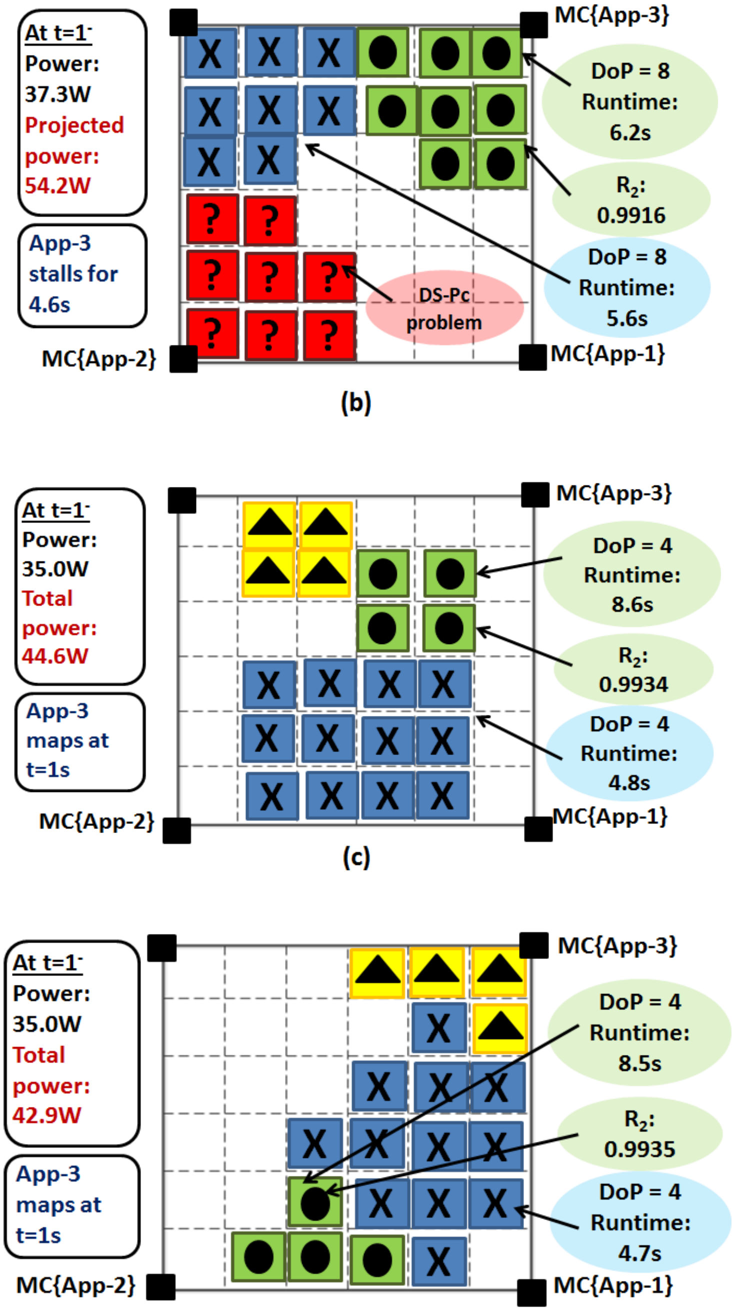
In this chapter, we improve upon the above mentioned shortcomings of the VARSHA framework by: *(i)* abandoning the restrictive region selection methodology and *(ii)* considering memory-traffic in the tile-selection (region-selection) and mapping scheme. Our improved tile-selection approach essentially balances leakage-power savings with communication optimization in the presence of memory-traffic in the NoC. In other words, we select CMP-tiles that dissipate lower leakage power while simultaneously minimizing the traffic footprint of the mapped application. The integrated tile-selection and task-to-tile mapping approach proposed in this article (discussed in section 5) is much better suited to effectively trade-off between both the objectives.

7.3 Motivational Example

In this section, we illustrate the advantages of some of the key aspects of our framework with the help of a small motivational example. We consider a scenario in which applications arrive at run-time at a service-queue to be served. The example assumes applications App-1 and App-2 arriving and being mapped on the CMP at time t=0s, and a third application App-3 arriving at t=1s. In this section, we show (in figures 41(a)-(c)) how different approaches proposed in prior works perform the application-mapping in comparison to our proposed approach (shown in figure 41(d)). The locations of memory-controllers (MCs) on the chip-corners serving the respective applications are also shown in figure 41. A DS-Pc of 45W is assumed for the 6×6 CMP utilized in this example.



(a)


At t=1-
Power:
37.3W
Projected
power:
54.2W
App-3
stalls for
4.6s
MC{App-3}
MC{App-2}
MC{App-1}
DoP = 8
Runtime:
6.2s
R2:
0.9916
DoP = 8
Runtime:
5.6s
DS-Pc
problem

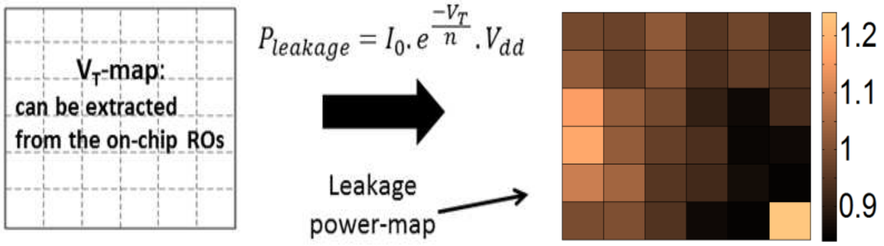
(b)

At t=1-
Power:
35.0W
Total
power:
44.6W
App-3
maps at
t=1s
MC{App-3}
MC{App-2}
MC{App-1}
DoP = 4
Runtime:
8.6s
R2:
0.9934
DoP = 4
Runtime:
4.8s

(c)

At t=1-
Power:
35.0W
Total
power:
42.9W
App-3
maps at
t=1s
MC{App-3}
MC{App-2}
MC{App-1}
DoP = 4
Runtime:
8.5s
R2:
0.9935
DoP = 4
Runtime:
4.7s

(d)



(e)

Figure 41: Motivational example using a 6×6 CMP where App-1 and App-2 are simultaneously mapped at time t=0 and App-3 arrives at t=1s. A DS-Pc = 45W is assumed. Application-mapping solutions obtained with: (a) [17]; (b) [116]; (c) VARSHA [139]; (d) our proposed VARSHA++ framework; (e) extracted V_T-map and corresponding leakage-power profile (in Watts) of the chip.

Raghunathan et al. [17] exploit the variation-profile (shown in figure 41(e)) of the CMP-die and select the tiles with the least leakage power dissipation (with highest effective V_T values) as shown in figure 41(a). But this work does not consider the inter-core communication in the NoC fabric, thereby producing mapping solutions with high communication power and latency overheads. On the other hand, prior works such as [35], [116] search for square or near-convex regions on the CMP die to map arriving application at run-time, so that a maximum number of applications can fit on the die-area, while also minimizing inter-core communication distances (as shown in figure 41(b)). Clearly, such a variation-unaware approach is suited for traditional area-constrained designs. Therefore, frameworks proposed in prior works [17] and [116] would map App-1 and App-2 at t=0 as shown in figures 41(a), 41(b) with a nominal DoP of 8. At t=1s, mapping of App-3 at its nominal DoP of 8 is stalled (until App-1 finishes) because the projected power values exceed the DS-Pc of 45W.

Mapping solutions produced with our prior work VARSHA [139] and the proposed VARSHA++ framework in this article are shown in figures 41(c) and 41(d). Our frameworks that adapt application-DoPs (*app-DoP*s) to extract maximum system-performance (i.e., minimum application service-times) for a given DS-Pc, increase the DoP of App-1 to 12. When App-3 arrives at t=1s, our frameworks map it with a reduced DoP of 4 with a zero-wait time while meeting the DS-Pc. Thus in our frameworks, *app-DoP*s are hiked from their nominal values opportunistically, to minimize runtimes, and *app-DoP*s are reduced to cut-down on wait times thereby minimizing average application service times (service time = run-time + wait-time).

We define reliability of the ith application as R_i = *{1-Probability of one or more soft-errors during the execution of App-i}*. In this work, we assume applications with different minimum reliability constraints running simultaneously on a CMP. Rc_i represents the reliability constraint of the i^{th} application. The application-reliability (R_i) primarily depends on V_{dd} and frequency of the cores (as shown in equations (1)-(3) in section 4). Additionally, R_i depends on the *app-DoP*. Although application execution-time typically reduces with higher DoP (as long as the DoP is below an application-specific performance saturation point), the chip-area susceptible to soft-errors increases. Therefore, for fixed values of voltage and frequency, soft-error probability increases (i.e., application-reliability decreases) with increasing *app-DoP*. Our frameworks (shown in figures 41(c) and (d)) exhibit reliability-awareness by reducing the DoP of App-2 (with a relatively stringent reliability-constraint) from the nominal value of 8 to 4, thereby meeting Rc_2 of 0.993 (figure 41(b)). However, Rc_2 is violated when the reliability-unaware frameworks [17] and [116] are employed (shown in figures 41(a) and (b)).

In summary, the variation-aware framework, [17], optimizes computation leakage energy by mapping onto low leakage tiles (resulting in overall energy of 89.9J in this example), while the variation-unaware framework, [116], optimizes communication energy by mapping applications within contiguous square-like regions (resulting in overall energy of 90.9J); whereas, VARSHA [139] finds contiguous rectangular regions on the die with minimum estimated leakage power, thereby cutting down on both communication and computation energy (resulting in overall energy of 74.5J). On the other hand, our proposed VARSHA++ framework in this article improves on VARSHA by: *(i)* further optimizing NoC communication-profiles with memory-traffic awareness, *(ii)* simultaneously producing better leakage profiles by relaxing the constraint of mapping onto contiguous regions (resulting in overall energy of 69.7J). Note that our frameworks also employ an energy-saving mechanism to opportunistically scale V_{dd}-

levels while meeting performance and reliability constraints of all applications. This feature is omitted from the above example for brevity.

7.4 Models and Problem Formulation

7.4.1 Reliability Modeling

Computer systems are susceptible to both transient and permanent faults, the latter being caused by fabrication defects or wear-out. We only consider the impact of transient faults on reliability and do not consider permanent faults in this work. It is assumed that permanent faults could either be detected during the testing phase or are mitigated by hardware-redundancy techniques. To model the dependence of raw soft (transient) error rate, raw-SER (λ), in a hardware component (core or NoC router) on voltage and frequency values, we use the relationship proposed in [11]:

$$\lambda(\omega_j) = \lambda_0 . 10^{\frac{d.(1-\omega_j)}{1-\omega_{min}}} \qquad \text{..... (7.1)}$$

where λ_0 is the SER corresponding to the highest voltage and frequency values (ω_{max}), and ω_j is the average of the normalized values of the jth combination of voltage and frequency (such that ω_{max}=1). For any compute-core, we use a value of 10^{-6} errors/sec for λ_0 and assume *d*=3 as in [104], [106]. However, we use $\lambda_0 = 10^{-6}/3$ for NoC routers, given that our router-area is roughly a third of the area of a compute-core. The reliability of a core or a NoC router is given by:

$$\mathbb{R}(\omega_j) = e^{-\lambda(\omega_j).\tau} \qquad \text{..... (7.2)}$$

where τ is the execution time of the component (time-duration that the component stays active). Assuming *n* = *app-DoP*, reliability of the *i*th application running on a CMP can be given as the product of all 2*n* (*n* routers and *n* compute-cores) component-reliabilities:

$$R_i = \prod_{2n} \mathbb{R}(\omega_j) \qquad \text{..... (7.3)}$$

Prior works [8], [9] have shown that at technology nodes of 32nm and below, process variations have almost no effect on SER. Therefore, in this chapter, we also assume no dependence of V_T-variations on SER, instead exploiting variations for speed and power benefits only.

7.4.2 Inputs, Assumptions, and Problem Objective

We assume the following inputs to our problem:

- A CMP with a regular mesh-based 2D network-on-chip (NoC), with T tiles: $T = (d^2)$, where d is the mesh dimension, and each tile consists of a compute core and a NoC router;
- A set S of candidate supply voltage (V_{dd}) levels for the chip;
- A set of N V_T-maps (N test-chips) incorporating the effects of WID and D2D variations with continuous distribution over the die; the estimated leakage power-profile for a given value of V_{dd} can be obtained from a V_T-map (using the relation shown in figure 41(e)).
- Application sequence s of length l, made up of η different applications, with arbitrary application inter-arrival times;
- Application task graphs for the set $P = \{P_1, P_2, \ldots P_\eta\}$ of DoPs for all applications; an application i possesses $|P_i|$ viable DoPs; an application has a maximum DoP value beyond which performance does not improve (or gets worse). Such sub-optimal DoP values are ignored;
- Vertices of each task-graph with execution-times of compute cores and edges with inter-task communication volumes; execution time and volume values are assumed available from offline profiling;
- Energy-optimal frequency constraint $\{f_1, f_2, \ldots, f_\eta\}$, minimum reliability-constraints $\{Rc_1, Rc_2, \ldots, Rc_\eta\}$, and designated MCs $\{MC_1, MC_2, \ldots, MC_\eta\}$ for all η applications;

- A chip-wide dark-silicon power dissipation constraint (DS-Pc).

We make the following assumptions in our work:

- There exists one-to-one mapping between tasks and cores;
- Variation-map data for a chip is available at run-time, in terms of the threshold-voltage (V_T) distribution, from the chip-frequency-profile obtained using distributed ring-oscillator based delay sensors [113];
- There exists a chip wide supply voltage that can be scaled using DVS at run-time, as the overheads of implementing DVS at a per-core granularity are deemed to be prohibitively expensive [115];
- As the applications are not necessarily mapped in contiguous regions of the CMP die, there exists inter-application interference, which is quantified in our work with cycle-accurate NoC-simulation;
- Memory controllers (MCs) are placed at the four corners of the CMP-die; all memory traffic in the NoC for any application is directed to and from a single MC;
- All compute-cores executing an application run at the same frequency, to avoid imbalances during multi-threaded execution [17], while the NoC fabric runs at a fixed NoC-frequency.

Problem Objective: Given the above inputs and assumptions, our objective is to perform run-time application-scheduling and DVS on a given CMP platform such that the average application service-time and average energy (across all N test-chips) are minimized, while all application-specific operating frequency- and reliability-constraints, as well as CMP platform-specific *DS-Pc* are satisfied.

7.5 VARSHA++ Framework: Overview

Figure 42 illustrates the key aspects of our VARSHA++ framework. The knowledge of the chip-variation profile is continuously utilized in the scheduling and DVS steps. Assuming equal priority for all incoming applications, the application-scheduling step consists of *(i)* determining the DoP (out of the $|P_i|$ DoPs) for each waiting application in the service-queue, and *(ii)* mapping the appropriate task-graphs on to the tiles of the CMP. For a given V_{dd}, scaling-up of application-DoPs (*app-DoPs*) is constrained by the available power-slack (difference between *DS-Pc* and current system-power dissipation), application-reliability constraints, and available tiles that meet the application-frequency constraints. At any given time, the scaling-down of V_{dd} (to save power/energy) is constrained by the frequency and reliability-constraints of the applications running on the CMP, whereas scaling-up of V_{dd} (to boost *app-DoPs*) is constrained by the *DS-Pc* for the CMP.

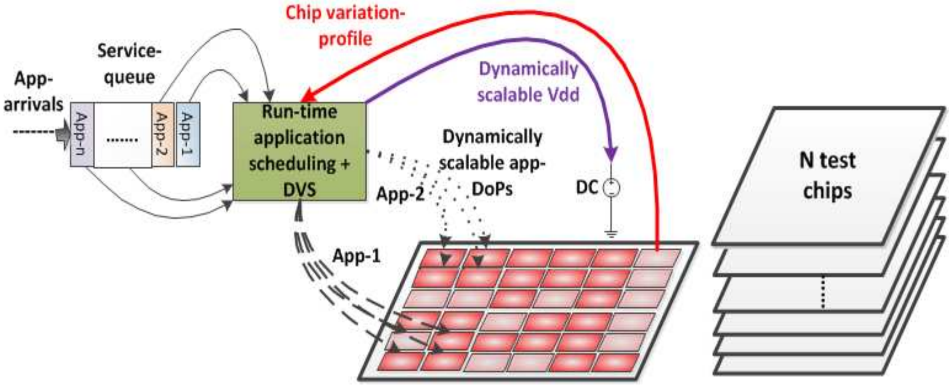


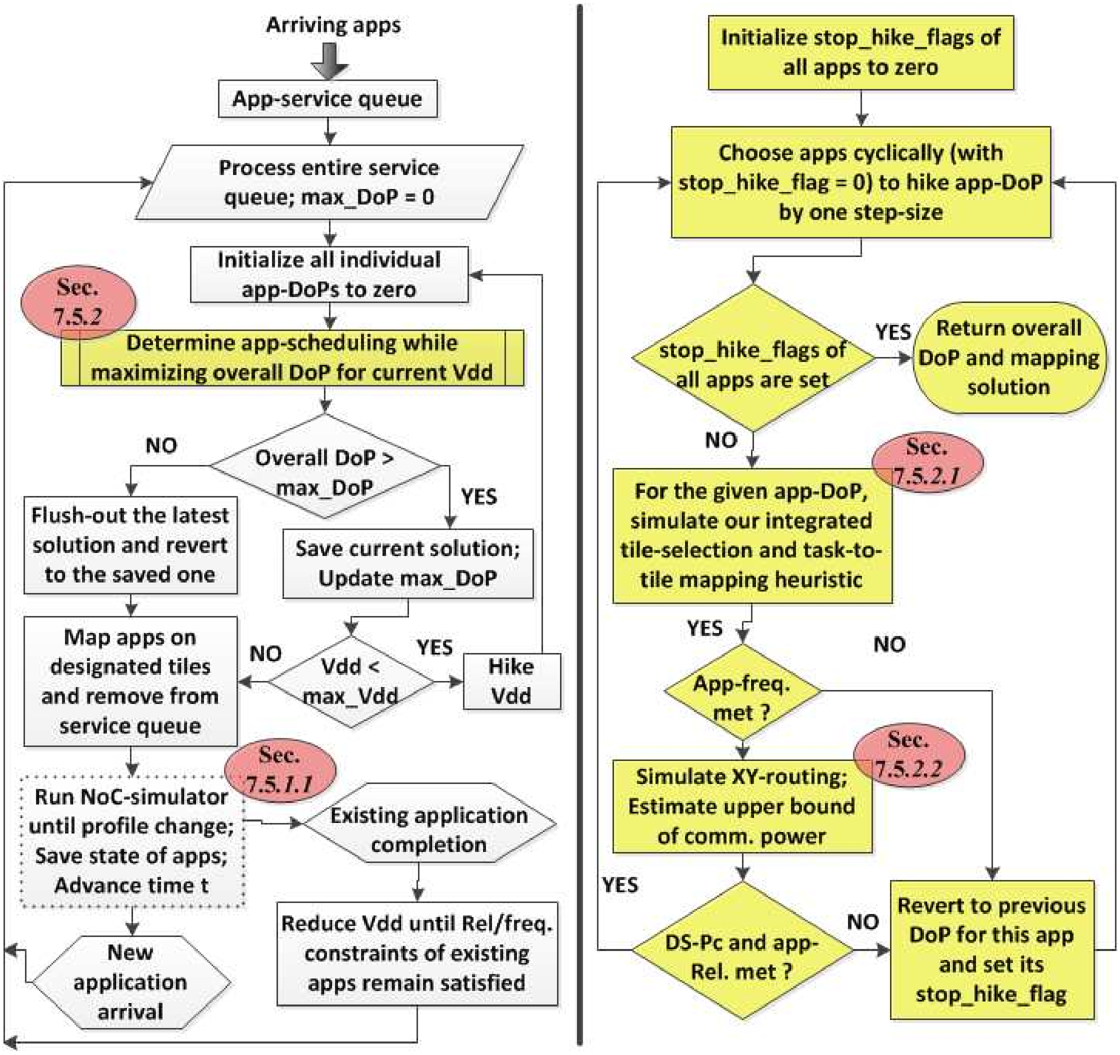
Figure 42: High level overview of our proposed application-scheduling + DVS framework (VARSHA++) for CMPs

The proposed framework is effectively executed in two nested procedures: *(i)* V_{dd}-level selection (outer loop), triggered on an arrival or a departure of any application; and *(ii)* determination of application-schedule for the current V_{dd}-level (inner loop). These procedures are discussed in detail in sections 7.5.1 and 7.5.2 respectively, and the corresponding design-flows for the procedures are shown in figure 43(a) and figure 43(b), respectively.

7.5.1 V_{dd}-level Selection

To extract maximum performance from the applications being considered for mapping at any instant of time, the first-order objective is to maximize overall (aggregate) DoP for all applications in the system, at the current time. Recall that an application typically has a maximum viable DoP, and higher DoPs can cause performance to degrade (e.g., due to high synchronization overheads) – such higher DoP configurations are ignored by our framework (section 7.4.2). Our V_{dd}-level selection heuristic (figure 43(a)) selects the V_{dd}-level that yields the maximum overall DoP. As a second-order power/energy saving objective, on completion of any application, the heuristic also reduces V_{dd} to the lowest allowable level that would not introduce any violations in frequency and reliability constraints of existing (already running) applications.

We assume all incoming applications are buffered in a service-queue (*App-service queue* in figure 43(a)). On arrival or completion of any application, the V_{dd}-selection heuristic is triggered, which processes the entire service-queue. The V_{dd}-selection heuristic iteratively invokes the application-schedule determination procedure (discussed in section 7.5.2), which produces the mapping-solution with the highest overall DoP corresponding to the current V_{dd}-level. The V_{dd}-level is hiked iteratively (in increments of 0.1V) and the highest *overall DoP* (sum-total of *DoPs* of all executing applications) obtained thus far is recorded at each step (*max_DoP*). This continues until either the overall DoP reduces compared to *max_DoP*, in which case the immediately preceding solution with the highest overall DoP is reverted to, or the maximum allowable V_{dd}-level (max_V_{dd}) is reached. Note that the overall DoP may increase with increasing V_{dd}-levels, as more applications satisfy frequency and reliability constraints for higher DoPs; at the same time, the chip-power will reach the *DS-Pc* quicker at higher V_{dd}-levels, thereby limiting overall DoP. Therefore, our search for the optimal V_{dd}-level culminates when the increase in overall DoP is limited by the *DS-Pc*. Finally, the best application-mapping solution with the highest overall DoP is mapped to the CMP. The voltage supply is changed to the selected V_{dd}-level, and the mapped applications are then removed from the service-queue.



Arriving apps
App-service queue
Process entire service queue; max_DoP = 0
Initialize all individual app-DoPs to zero
Sec. 7.5.2
Determine app-scheduling while maximizing overall DoP for current Vdd
NO
Overall DoP > max_DoP
YES
Flush-out the latest solution and revert to the saved one
Save current solution; Update max_DoP
Map apps on designated tiles and remove from service queue
NO
Vdd < max_Vdd
YES
Hike Vdd
Sec. 7.5.1.1
Run NoC-simulator until profile change; Save state of apps; Advance time t
Existing application completion
Reduce Vdd until Rel/freq. constraints of existing apps remain satisfied
New application arrival
Initialize stop_hike_flags of all apps to zero
Choose apps cyclically (with stop_hike_flag = 0) to hike app-DoP by one step-size
stop_hike_flags of all apps are set
YES
Return overall DoP and mapping solution
NO
Sec. 7.5.2.1
For the given app-DoP, simulate our integrated tile-selection and task-to-tile mapping heuristic
YES
NO
App-freq. met ?
Sec. 7.5.2.2
Simulate XY-routing; Estimate upper bound of comm. power
YES
DS-Pc and app-Rel. met ?
NO
Revert to previous DoP for this app and set its stop_hike_flag

Figure 43: Design-flows for the proposed VARSHA++ framework: (a) V_{dd}-level selection (section 7.5.1); (b) Determination of application-schedule for the current V_{dd}-level (section 7.5.2).

7.5.1.1 NoC Simulation

To accurately estimate latencies (and thus application completion times) due to intra-application communication as well as the memory-traffic in the presence of possible inter-application interference (note that individual NoC routers route packets from multiple applications) on the mesh-based NoC fabric, we employ a cycle-accurate NoC simulator [54]. An XY routing scheme is assumed. In the event of either arrival or completion of an application, the state of the computation and communication profile of the CMP could potentially change. Therefore, at this time, the current NoC-simulation instance is terminated, time *t* of the simulation engine is advanced appropriately, and the current state of CMP is saved i.e., the number of packets received by the destination core of each communication flow is recorded. This enables the NoC-simulation to accurately resume at its next invocation. Note that the dotted outline of the NoC-simulation block (in figure 43(a)) signifies that this step simulates the runtime NoC-behavior and is not needed when utilizing our framework on real hardware, where NoC latency, in the presence of possible run-time congestion, for a communication event can be easily obtained after the communication event finishes.

7.5.2 Determination of Application-schedule

Given a specific execution environment for the CMP (including the V_{dd}–level, available power-slack, and variation-profile), the objective of the application-schedule determination heuristic is to maximize the overall DoP, while simultaneously considering all applications in the service-queue and satisfying application-frequency and -reliability constraints. Figure 43(b) shows the design-flow of this heuristic. Starting at the least possible DoP value of zero (DoP of zero leaves the application unmapped at the current time) applications are considered cyclically for hiking of DoP to their next higher valid DoP-level. Here, to extract maximum performance from the CMP, we choose applications for hiking of DoP in order of their compute-intensiveness, because of the relatively smaller communication delay and communication power overheads for compute-intensive applications at higher DoPs. Also, we hike *app-DoPs* symmetrically across all applications because execution of an application is generally more energy-efficient at lower DoPs (due to lower parallelization and communication overheads), i.e., running four applications simultaneously, each with DoP=4, is typically more energy-efficient than running them one after the other with DoP=16 each.

To produce an optimal mapping for the application under consideration (with a specific DoP), the following steps are performed: *(i)* integrated tile-selection and task-to-tile mapping given the task-graph for the current DoP (section 5.2.1); and *(ii)* communication-flow routing and delay/power analysis (section 7.5.2.2). After the above steps, the mapping is evaluated for overall power footprint and reliability of the applications. Satisfaction of the application-frequency constraints are also checked during the tile-selection and application-mapping step. As shown in figure 43(b), the DoP-hike of any application could fail due to potential violation(s) in application-frequency constraints, application-reliability constraints, or *DS-Pc*. When an attempted DoP-hike is stalled for any application, its *stop_hike_flag* is set to preclude it from future DoP-hike consideration, and the feasible mapping with the preceding DoP is finalized for this application.

7.5.2.1 Integrated Tile-selection and Mapping Heuristic

We first motivate the need for an integrated tile-selection and task-to-tile mapping approach with a small example. Given the V_T-map for a die, the maximum frequency that each core on the die can be reliably clocked at depends upon the V_T as well as the V_{dd} values, and the relationship can be expressed as:

$$f_{max} = \frac{\mu(V_{dd}-V_T)^{\alpha}}{C_0.V_{dd}} \qquad \textbf{.....} \text{ (7.4)}$$

where, α and μ are technology-dependent constants, and C_0 is switching capacitance of the critical path [98]. In our approach, we utilize the knowledge of both frequency and leakage-power profiles of the chip. Note that dynamic-power remains unaffected by V_T-variations and is thus not considered in this step. Our objective is to find the set of tiles on the mesh such that leakage-power as well as communication latency and energy are minimized, while satisfying the frequency-constraint of the application being mapped.

Figure 44 shows an illustration of our approach for a 4×4 mesh-based CMP. The application of DoP equal to 4 (shown in figure 44(b), with a frequency constraint of 1.9GHz can only be mapped on to feasible tiles that satisfy its minimum frequency constraints (as shown in figure 44(a)). Figure 44(c) shows a mapping that exclusively minimizes the communication overheads in terms of latency and energy by minimizing the communication Manhattan distances (MDs) for inter-task communication as well as memory (MC) communication. On the other hand, figure 44(d) shows an application-mapping solution that exclusively minimizes the computation leakage-power power by choosing tiles with minimum leakage from the chip-leakage-profile shown in figure 44(a).

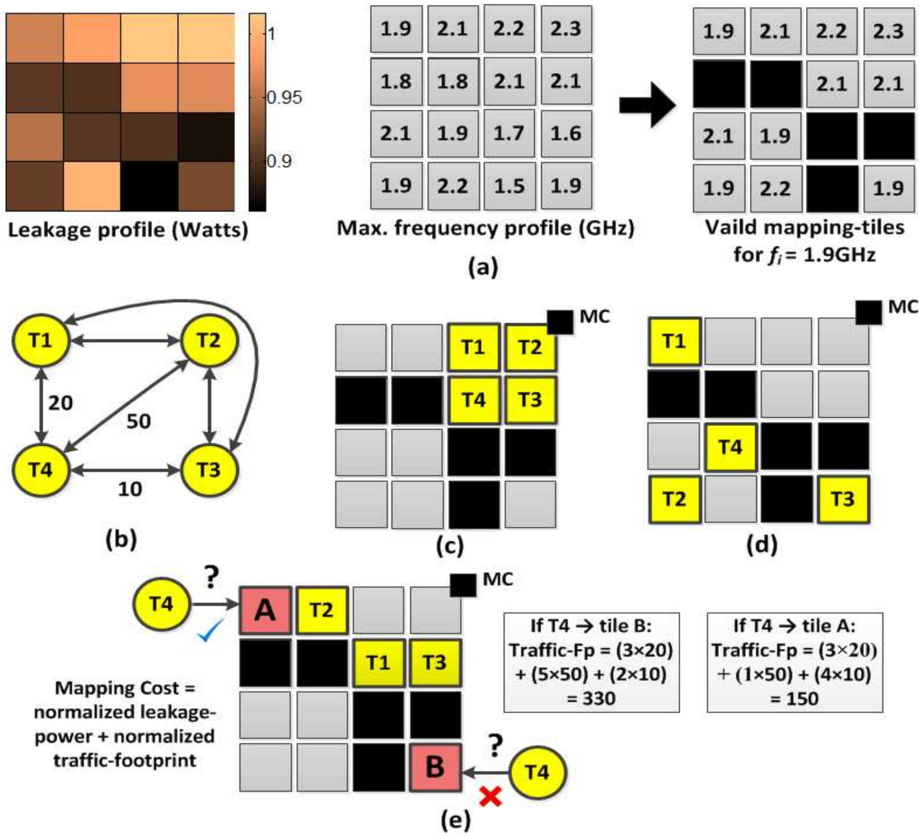


Figure 44: Example of our integrated tile-selection and application-mapping heuristic: (a) leakage power map and feasible mapping tiles for application frequency constraint of 1.9 GHz; (b) application task-graph with *app-DoP* = 4; (c) a solution minimizing communication energy/latency; (d) a solution minimizing computation energy; (e) our proposed solution.

Figure 44(e) presents our solution. Let us assume that tasks T1, T2, and T3 have already been mapped (as shown) using a mapping cost function that considers both the core leakage power and the resulting communication overheads. Now, tile A and tile B, both of which correspond to the same MD (in terms of number of hops) from the appropriate MC and similar leakage powers, are feasible candidates to map the task T4. But the two candidates produce significantly different traffic-footprint (*traffic-fp*) on the underlying NoC fabric as shown in figure 44(e), where *traffic-fp* is calculated as:

$$traffic\text{-}fp = \sum_{\forall flows} MD \times comm.\ volume \quad \textbf{.....}\ (7.5)$$

Mapping T4 to tile A reduces the communication latency and thereby the application-runtimes. Note that in this example the communication volumes to and from the MCs are ignored as the MDs of tile A and tile B from the appropriate MC are equal. Note also that the *traffic-fp* metric can only consider communication flows associated with tasks that have already been mapped, in addition to communication flows to and from the appropriate MC. Our approach therefore is able to intelligently trade-off communication overheads with computation energy, considering leakage power as well as NoC traffic footprint during the tile selection step.

Now, we present the details of our integrated tile-selection and mapping approach. The pseudo code of the heuristic is shown in Algorithm 7.1. The problem-objective is to optimize both the computation and the communication profiles of the application *i* to be mapped, given the application characteristics (task-graph, *app-DoP*, f_i, MC_i) and the CMP-platform characteristics (power-slack, V_{dd}, variation-profile) at current time *t*. We first sort the tasks in decreasing order of communication volumes (ingress and egress volumes), to be mapped in that order (step 1). In order to limit the memory traffic in the NoC, we restrict our mapping search space to a square region in the CMP-corner adjoining MC_i (step 2). Note that the application is not necessarily mapped contiguously within the square region. The size of this square region, *Sq_size*, is determined by the *app-DoP* or the number of tiles that application *i* maps to, and can be expressed as:

$$Sq\_size = \max(\left\lceil\left(\sqrt{c \times DoP}\right)\right\rceil^2, \min\_sq\_size) \qquad \textbf{.....}\ (7.6)$$

where *c* and *min_sq_size* are constants. To avoid frequent application-stalls due to unavailability of tiles when mapping the smallest *app-DoP*s, we use *min_sq_size* as the smallest region-size of the search-space.

Now, we map the first task from the sorted list on to an unmapped tile that satisfies the f_i constraint with the smallest cost C1 (step 3). For mapping the rest of the tasks (step 4), the *traffic-fp* corresponding to each of the available tiles (meeting the f_i constraint) within the square region is calculated (step 5) as explained earlier in this section (Eq. 7.5). Using these *traffic-fp* values, the mapping cost C2 is calculated for each valid tile, and the task is mapped onto the tile with minimum cost C2 (step 6).

Algorithm 7.1: Integrated tile-selection and mapping heuristic

inputs: Application task graph of the appropriate DoP and CMP platform characteristics at current time t

```
1: sort the tasks in the order of decreasing communication volumes – to
   be mapped sequentially in this order
2: designate a square region (of size given by Eq. 7.6) in the CMP-corner
   adjoining the appropriate MC for mapping the application
3: map the first task onto an available tile on the square region with the
   least cost C1= normalized leakage-power + normalized MD from MC
4: for all remaining tasks in the sorted list, do {
5:   compute the traffic-fp w.r.t. already mapped tasks (and MC)
     corresponding to mapping the current task onto all available tiles
6:   map the task onto the tile on the square region with the least cost:
     C2 = normalized leakage-power + normalized traffic-fp
7: }
```

output: mapping solution of application task graph on to the CMP die

<u>*Theoretical time-complexity analysis*</u>: The size of the square region to be evaluated (for cost C1 or C2) for mapping each task is proportional to the *app-DoP* and there are *app-DoP* number of tasks to be mapped. Also, to calculate the *traffic-fp* of each candidate tile, up to *app-DoP* communication flows may have to be evaluated. Thus, the time complexity for our integrated mapping heuristic would be of the order of O((*app-DoP*)^3). Note that *app-DoP* is typically a small integer (between 4 and 16 in our experiments) and can be treated as a constant.

7.5.2.2 Communication Power Estimation

Similar to numerous prior works e.g., [116], we use low-cost XY-routing to route the communication-flows of applications. Average dynamic power of NoC routers and links (corresponding to different voltages, communication-loads, and router-sizes), are assumed to be saved in the read-only (or non-volatile) memory on the CMP die, and accessible by our framework at run-time. Router leakage powers, estimated from the chip-variation-profile, are added to the appropriate dynamic power values to produce the total communication-power for running each application. Based on the active-times (execution-times) of routers and compute-cores, the application-reliability is computed using equations (7.1)-(7.3). For our analyses, the energies and run-times of applications are calculated from component powers and active-times.

7.5.3 Complexity Analysis of VARSHA++ Framework

Our application schedule determination heuristic (discussed in section 7.5.2), which maximizes the DoP of all applications being processed at the current $\mathrm{V_{dd}}$-level, attempts to find mapping solutions for the *w* waiting applications for up to $|P_i|$ DoP-levels. This heuristic could be required to execute for at most $|S|$ (total number of candidate V_{dd}-levels) times. $|S|$ and $|P_i|$ are small constant integers in our experiments. Also, at any given time, only a small number of applications (up to *w*) are generally expected to be processed given the DS-Pc. Therefore, whenever the service-queue is processed in our framework, the number of times that our integrated tile-selection and application-mapping heuristic would need to be invoked is bounded by a relatively small constant integer. As this step, which has the highest theoretical time-complexity in the design-flow, finishes in constant time, O((*app-DoP*)^3) (as discussed in section 7.5.2.1), the time-complexity of our framework can be bounded in constant time. In other words, our framework is scalable with respect to the CMP-size (number of tiles *T*) and thus amenable for use at runtime.

7.6 Experiments

Our experiments were conducted using η=14 different parallel application benchmarks: seven from the SPLASH-2 benchmark suite [58] (*cholesky, fft, lu, ocean, radix, radiosity, and raytrace*), and seven from the PARSEC benchmark suite [56] (*vips, swaptions, fluidanimate, dedup, streamcluster, canneal, and blackscholes*). We consider DoPs that are multiples of 4, up to 16, where a DoP of 8 is considered the nominal DoP value, as a reasonable trade-off between speed and energy. As discussed earlier, every application has a unique maximum viable DoP beyond which further performance gains cannot be achieved. Our task-graphs are modeled based on the inter-core communication characterization from [47] and based on our observations of traces and communication patterns between the respective pair of cores. The reliability constraints of different applications are set in the range: 0.99 to 0.999.

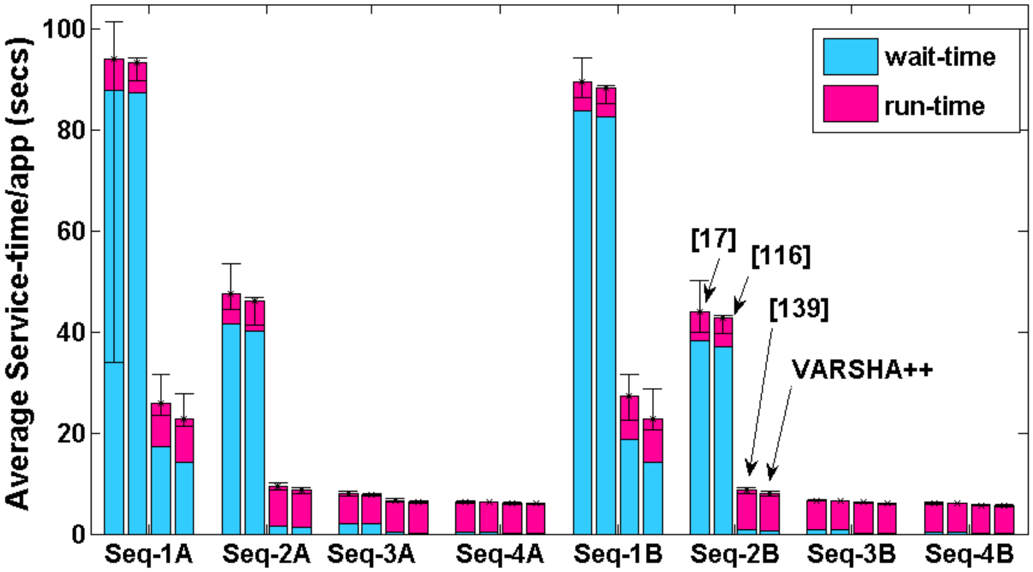
We assume the ARM Cortex-A9 processors [37] as the baseline CMP compute cores, which support five operating voltage levels ($|S|$=5): 0.8V, 0.9V, 1.0V, 1.1V, and 1.2V. The application-specific energy-optimal core frequencies range from 1300 MHz to 1900 MHz, based on the level of compute intensity of the tasks assigned to cores. We consider an 81-core mesh based CMP platform with dimensions 9×9 for our experiments. The dark-silicon power-constraint (*DS-Pc*) is set to 50W. In our integrated tile-selection and mapping heuristic (discussed in section 7.5.2.1), we use a value of 1.5 for the constant *c* (in Eq. 7.6) when computing the size of the square region to map any application. However, a minimum size (*min_sq_size*) of 16 tiles is used for this region to avoid excessive stalling of applications at the highest arrival rates. The delay-overhead for V_{dd}-scaling (PLL lock time) is estimated to be less than 5μs, similar to [105], which is negligible compared to the granularity of application-runtimes (2-9 seconds each) in our experiments.

To investigate the applicability of our approach to CMP dies with diverse variation-profiles, we use either 100 or 1000 test-chips (N=100 or N=1000), in different experiments. The V_T-maps corresponding to these test chips are generated using the open-source tool [97] (based on systematic and random WID-variation model in [98]). The values of 0.3 and 0.09 are used for the statistical mean and a standard deviation of the parameter V_T respectively, and a correlation range (ϕ) of 0.5 is used (as recommended in [98]). A normally distributed V_T bias, representing the D2D variation component is superimposed on to these V_T-maps; the standard deviation of the D2D V_T is assumed to be 6%, as in [99]. Each V_T-map represents a 27×27 grid of points corresponding to 9 ring oscillator test-sites for each core on the CMP. For a given V_T-map, the maximum core-frequencies are calculated by using the V_T-max values and the core-leakage powers are calculated using the V_T-avg values (out of the 9 V_T values per core). The power values of routers and links (32-bit wide) for different voltages and frequencies at varying communication loads, for the 32nm technology node are obtained from ORION 2.0 [40]. Note that the router power values obtained are for nominal V_T, and are scaled for varying V_T values.

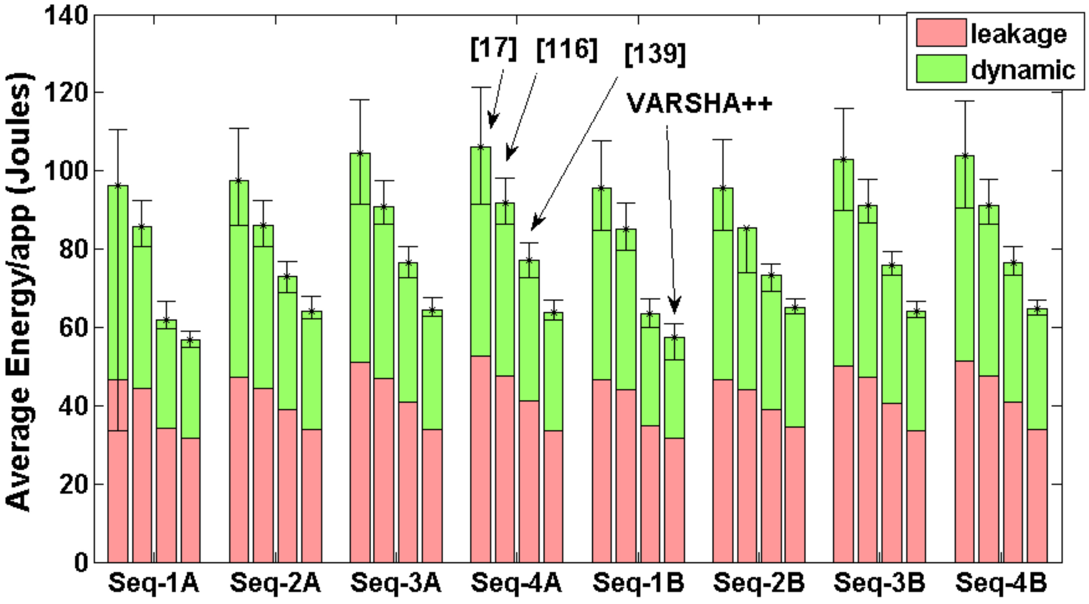
7.6.1 Experimental Results

We compare the results obtained from our proposed VARSHA++ framework with those obtained from using run-time application mapping frameworks proposed in recent prior works [17], [116], and [139] (VARSHA). A variation- and dark-silicon-aware mapping technique is proposed in [17], whereas [116] advocates for a traditional area-constrained design approach. We implemented these prior works to the best of our understanding. Our experiments considered two unique application-sequences (Seq-A and Seq-B) that represent an ordering of arriving application instances, with instances randomly chosen from among the 14 applications considered. For each sequence, we vary the inter-arrival times of application-instances randomly within the following ranges: 0 to 2 seconds (Seq-1A and Seq-1B), 0 to 4 seconds (Seq-2A and Seq-2B), 0 to 8 seconds (Seq-3A and Seq-3B), and 0 to 16 seconds (Seq-4A and Seq-4B). We assume l=100 application-instances in any application-sequence. In the first set of experiments, we consider a set of 100 test-chips, representing different variation-profiles; our results (in figure 45, Table 13, and Table 14) show the mean-values across all 100 test-chips.

Figure 45 presents results comparing VARSHA++ with the frameworks from [17], [116], and [139]. The prior works [17] and [116] assume fixed nominal *app-DoPs*. Our VARSHA++ framework as well as [139] adapt *app-DoP*s in accordance with the application inter-arrival rates to minimize the application service-times. Observe in Table 13 that for both sequences, the average *app-DoP* reduces with increasing inter-arrival-rates for both of our frameworks. At higher inter-arrival rates when applications with nominal DoPs cannot be quickly serviced due to the *DS-Pc* constraint, our frameworks cut down application wait-times significantly by reducing DoPs (as shown in figure 45(a) - Seq-1A,B and Seq-2A,B), although the application run-times tend to increase due to the reduction in DoPs. On the other hand, at lower inter-arrival rates, with on average fewer applications to be serviced simultaneously, our framework opportunistically hikes the application-DoPs to minimize run-times (as shown in figure 45(a) - Seq-3A,B and Seq-4A,B). In comparison with [116], we obtain on average **71%** savings in average service-times with our VARSHA++ framework. Note that maximum savings are obtained when the inter-arrival-rates are most stringent, as shown for Seq-1A and Seq-1B in figure 45(a). The communication-unaware framework in [17] maps applications on to regions of non-contiguous tiles to optimize computation power exclusively (without consideration of communication traffic in the NoC), resulting in longer run-times due to longer communication-latencies. Compared to [17], we obtain **71.5%** savings in average service-times with our VARSHA++ framework.



(a)



(b)

Figure 45: Results of our proposed framework (VARSHA++) vs. frameworks in [17], [116], [139]: (a) Average service-time per application-instance (wait-time + run-time); (b) Average energy per application-instance (leakage + dynamic). The bars represent mean values of service-times and energies across 100 test-chips, while variation in service-times and energies is shown by confidence intervals.

Our frameworks also opportunistically lower V_{dd}-levels while ensuring that frequency and reliability constraints remain satisfied for the set of existing applications. The weighted-average V_{dd} metric ($V_{w\text{-}avg}$) represents the average value of $\mathrm{V_{dd}}$ throughout the execution of the entire application-sequence for a given chip, and is defined as:

$$V_{w-avg} = \frac{((V_{\Delta t1} \times \Delta t1) + (V_{\Delta t2} \times \Delta t2) + \cdots + (V_{\Delta tp} \times \Delta tp))}{\sum_{i=0}^{i=p} \Delta ti} \quad \text{..... (7.7)}$$

where Δti is the i^{th} time-interval during which the V_{dd}-level stays constant at a value of $V_{\Delta ti}$, and the execution of the entire application-sequence s, takes p such time-intervals. Table 14 shows the mean $V_{w\text{-}avg}$ across the 100 test-chips for both of our frameworks. In the absence of DVS in [17] and [116], in our implementation of these frameworks, we assume the highest voltage of 1.2V, which is just high enough to meet all application-frequency constraints across all test-chips.

Table 13: Mean values across 100 test-chips for average DoP per application-instance for [139] vs. VARSHA++

	Seq-1A	Seq-2A	Seq-3A	Seq-4A	Seq-1B	Seq-2B	Seq-3B	Seq-4B
[139]	4.21	5.84	8.54	9.57	4.10	5.47	8.98	10.32
VARSHA++	4.24	6.49	9.13	9.86	4.13	6.15	9.41	10.70

Table 14: Mean values across 100 test-chips for weighted-average V_{dd} ($V_{w\text{-}avg}$) for [139] vs. VARSHA++

	Seq-1A	Seq-2A	Seq-3A	Seq-4A	Seq-1B	Seq-2B	Seq-3B	Seq-4B
[139]	1.10	1.15	1.09	0.97	1.13	1.16	1.08	0.95
VARSHA++	1.10	1.13	1.07	0.96	1.10	1.13	1.06	0.94

Although [17] saves on leakage-power dissipation in computation cores by performing variation-aware mapping, it produces solutions with much longer communication paths along with high V_{dd}, resulting in highest service times and energy values. In comparison, the variation-unaware framework, [116], consumes higher computation power but minimizes communication latency and energy by mapping applications onto contiguous regions (as shown in figures 5(a) and 5(b)). With variable *app-DoP*s in our frameworks, energy-efficiency decreases with increasing app-DoPs (due to increased communication and sub-linear increase in computation-performance), while decreasing V_{dd}-levels generally cause an increase in energy-efficiency. Therefore, in figure 45(b), we observe variation in energy values with different inter-arrival rates. The maximum energy savings are obtained for the most stringent inter-arrival rates, as shown for Seq-1A and Seq-1B in figure 45(b), due to opportunistic reduction of DoPs. We find from figures 5(a), (b) that our VARSHA++ framework produces on average, **37.6%** and **29.3%** savings in mean energy (32% and 27% savings in mean leakage energy) and, **71.5%** and **71%** savings in mean application service time (88% and 87% savings in mean application wait-times), over [17] and [116] respectively.

In general, our proposed VARSHA++ framework, which co-optimizes NoC-communication (including memory traffic) in addition to leakage-power, results in even higher CMP-performance compared to [139] due to the following reasons: *(i)* higher leakage-power savings generate higher power slacks enabling execution of applications at higher DoPs on average, thereby cutting down on application service times, and *(ii)* shorter communication-routing paths produce shorter application run-times, which in turn enables application-reliability constraints to be satisfied at lower V_{dd}-levels (thus saving even more power) and higher app-DoPs (thus boosting performance further). The higher average app-DoPs (by up to **12.4%**) and lower $V_{w\text{-}avg}$ values (by up to **2.6%**) obtained with VARSHA++ in comparison to [139] are shown in Table 13 and Table 14 respectively. We obtain improvements of **10.1%** in mean application service time (**21.7%** in mean wait time) and **13.4%** in mean energy (**14.1%** in mean leakage energy), in comparison to the framework proposed in [139].

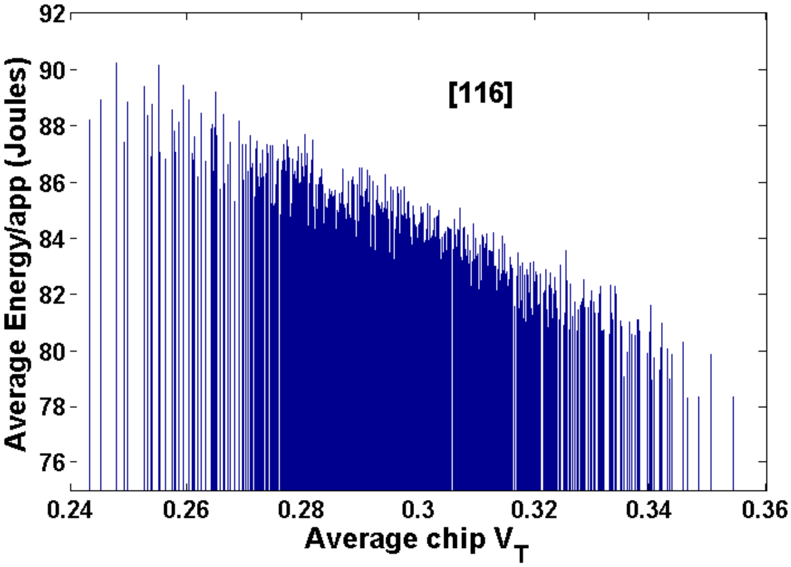
We note that for applications with relatively more stringent reliability-constraints, it may not be possible to support nominal-DoP even at the highest V_{dd}-level (1.2V). Therefore, when using reliability-unaware frameworks with no DoP-adaptivity, reliability-constraints (*Rc*) of such applications may be violated. Table 15 shows the average number of application-instances incurring *Rc*-violations, across the 100 test-chips for schedules produced by [17] and [116]. With longer routing distances, [17] consumes more routing resources compared to [116], thus even with the same app-DoPs (nominal DoPs), the estimated failure-rates are greater for [17]. Observe that higher number of *Rc* violations may occur for high inter-arrival rates. This is due to the higher levels of NoC-traffic-congestion and higher execution times of hardware components with increased die-utilization at higher application inter-arrival rates. Our reliability-aware frameworks, both [139] and VARSHA++ framework, result in no *Rc*-violations because of their ability to dynamically reduce app-DoPs as well as hike V_{dd}-levels in accordance with application reliability-requirements.

Table 15: Average number of reliability-constraint (Rc) violations across the 100 test-chips obtained using the reliability-unaware frameworks [17] and [116]

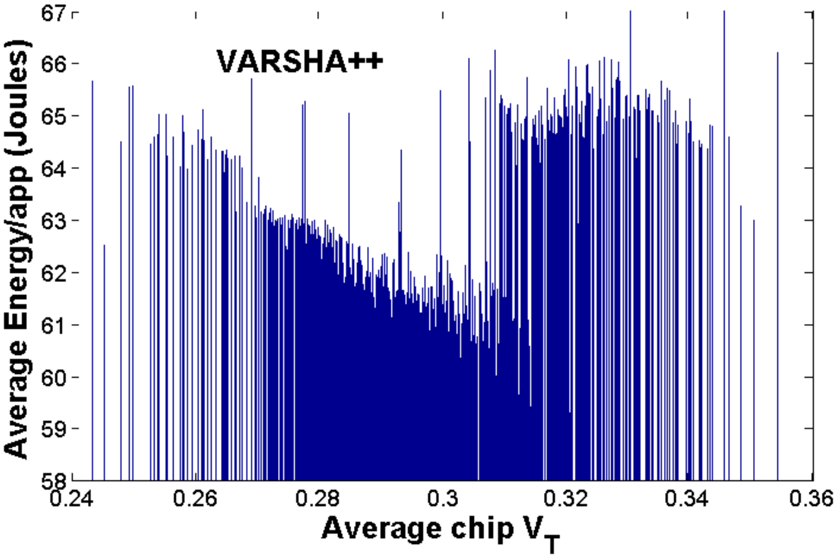
	Seq-1A	Seq-2A	Seq-3A	Seq-4A	Seq-1B	Seq-2B	Seq-3B	Seq-4B
[17]	13.9	13.9	13.4	13.4	13.8	13.7	13.7	13.4
[116]	13.4	13.4	7.15	6.95	12.8	12.8	7.0	5.0

We also perform another set of experiments to investigate the effects of D2D variations in terms of energy-efficiency obtained using different frameworks. As discussed earlier, our frameworks, both [139] and VARSHA++, combine a DVS scheme with an application mapping methodology. This facilitates the mitigation of the adverse effects of D2D variations by enabling faster chips to utilize lower V_{dd} values and slower chips to utilize higher V_{dd} values while satisfying reliability and frequency constraints at all times. On the other hand, prior works, such as [116] and [17], do not employ DVS in their application-mapping methodology, instead utilizing a fixed high value of V_{dd} for all test-chips; thereby resulting in significantly worse energy-efficiencies over the entire set of test-chips. Thus, in this set of experiments, we select one of our frameworks (VARSHA++) that employs DVS in order to mitigate effects of D2D variations, and a prior work ([116]) that does not employ DVS to analyze the impact of how D2D variations are addressed by these frameworks. Also, here, we consider a much bigger test set (of a 1000 test-chips) while considering just one application-sequence, Seq-2A.

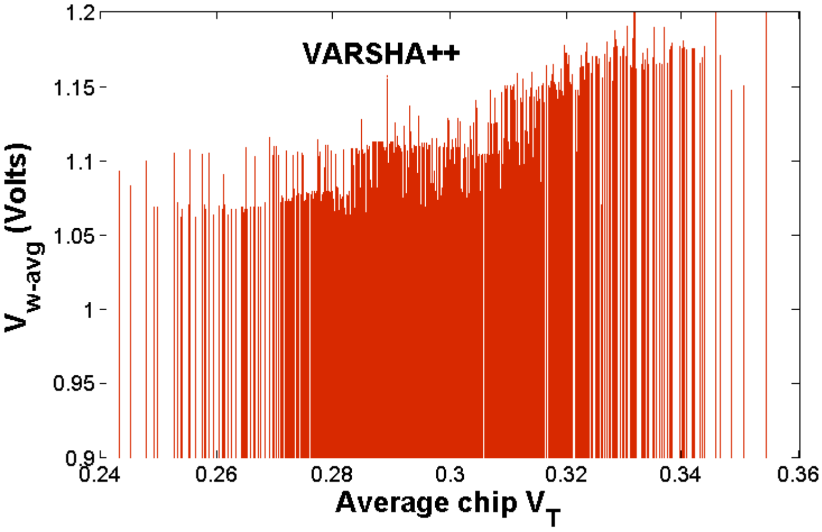
Figure 46 shows the average energy consumed per application for all 1000 test-chips across VARSHA++ and [116]. The experimental results indicate that for the framework from [116] (figure 46(a)), where a fixed high value of V_{dd} is set pessimistically for the worst case (highest) V_T values, slower chips (those with high V_T and lower leakage) could prove to be advantageous as they would be left with greater power-slack to accommodate more applications at any given time (i.e., less % of dark-silicon), thus resulting in better service-times and energy profiles. We observe that the energy efficiency obtained using [116] generally steadily reduces as the leakage power of test-chips increases (going from right to left in figure 46(a)).



(a)



(b)



(c)

Figure 46: Effects of D2D variation in terms of average energy/app obtained for 1000 test-chips for Seq-2A. (a) With [116], performance degradation due to D2D variations cannot be mitigated efficiently with high V_{dd}, faster chips generally result in much worse energy-efficiency; (b) Due to intelligent DVS scheme integrated within the VARSHA++ framework, the best energy efficiency is obtained for test-chips with moderate leakage and performance profiles; (c) $V_{w\text{-}avg}$ distribution with VARSHA++.

Conversely, the VARSHA++ framework can intelligently mitigate the adverse effects of D2D variations resulting in much higher energy efficiency, as shown in figure 46(b). With VARSHA++, given a *DS-Pc*, faster chips (that also dissipate higher leakage power) can usually utilize lower V_{dd}-levels to minimize energy, and slower chips (that dissipate lower leakage power) can utilize higher V_{dd}-levels to improve performance, as shown in figure 46(c). We observe that test-chips with moderate leakage and performance profiles produce desirable energy results, whereas chips that are too leaky or too slow yield worse results, as shown in figure 46(b). Thus, VARSHA++ can more intelligently mitigate the adverse effects of D2D variations resulting in much higher energy efficiency.

7.7 Conclusion

In this chapter, we proposed an application-mapping and DVS framework which represents one of the first efforts to integrate reliability and variation-awareness in a run-time variable degree-of-parallelism

(DoP) based application-scheduling methodology to enhance performance of multi-core systems in the dark-silicon era. We combine the traditionally disjoint steps of region-selection and task-to-tile mapping to co-optimize memory-communication, NoC traffic, and leakage power. Tailored for deeply scaled technologies, our light-weight runtime framework generates highly optimized application-mapping solutions. Our experimental results show that the proposed framework produces average savings of **10%-71%** in application service-times, **13%-38%** in energy, and avoids reliability violations unlike state of the art prior works that suffer from reliability violations in up to **14%** of application instances arriving at run-time.

8 ARTEMIS: An Aging-Aware Runtime Application Mapping Framework for 3D NoC-based Chip Multiprocessors

In emerging 3D NoC-based chip multiprocessors (CMPs), aging in circuits due to bias temperature instability (BTI) stress is expected to cause gate-delay degradation that, if left unchecked, can lead to untimely failure. Simultaneously, the effects of electromigration (EM) induced aging in the on-chip wires, especially those in the 3D power delivery network (PDN), are expected to notably reduce chip lifetime. A commonly proposed solution to mitigate circuit-slowdown due to aging is to hike the supply voltage; however this increases current-densities in the PDN due to the increased power consumption on the die, which in turn expedites PDN-aging. We thus note that mechanisms to enhance lifetime reliability in 3D NoC-based CMPs must consider circuit-aging together with PDN-aging. In this chapter, we propose a novel runtime framework (*ARTEMIS*) for intelligent dynamic application-mapping and voltage-scaling to simultaneously manage aging in circuits and the PDN, and enhance the performance and lifetime of 3D NoC-based CMPs. We also propose an aging-enabled routing algorithm that balances the degree of aging between NoC routers and cores, thereby increasing the combined lifetime of both. Our framework also considers dark silicon power constraints that are becoming a major design challenge in scaled technologies, particularly for 3D stacked CMPs.

8.1 Introduction

Bias Temperature Instability (BTI) is the most dominant physical phenomenon that degrades the maximum switching rate of transistors under long periods of voltage stress [118]. BTI causes gradual circuit slowdown over the operational lifetime of an electronic chip. For systems manufactured at technology nodes below 45nm, BTI-induced delay-degradation can be quite significant [119], [120]. The principal effect of such a circuit-aging mechanism is to increase circuit-threshold voltage (V_T), which results in higher circuit-delay. From a system-level perspective, such V_T-degradation causes slowdown in critical paths of processor-cores and NoC routers, thereby limiting overall system performance.

Additionally, electromigration (EM) in metal wires on the chip leads to increased interconnect resistance over time. This phenomenon is most dominant in power delivery network (PDN) wires that carry larger unidirectional currents compared to signal wires [12], [13]. The increased resistance of the power-grid results in higher IR-drops in the PDN, which causes further circuit slowdown due to degradation of supply voltage [6]. These adverse effects of EM are expected to be particularly severe in 3D chip-multiprocessors (CMPs) that possess limited number of power-pins and higher current densities [121]. Also, with process technology scaling, this problem is exacerbated due to the reduction in cross-sections of metal wires, which causes further increase in PDN current-densities [12].

To mitigate BTI-induced delay degradation, while maintaining circuit operation at a minimum clock frequency (i.e., minimum performance level), one solution is to hike the supply voltage [124] adaptively over time based on the degree of circuit-aging. However, doing so increases current-densities in the PDN due to the increased power dissipated in the chip as a result of the voltage-hike. High current densities end up causing faster EM-induced PDN-aging [122], hastening circuit-slowdown. Hiking supply voltage also increases V_T-degradation, further increasing the rate of circuit-aging.

As aging reduces the viable lifetime of current and emerging CMPs, it is becoming increasingly important to consider it during the design process. Unfortunately, designers today are more focused on meeting performance requirements, and resort to either using costly hardware guardbands to minimize the effect of performance variations on a die, or employing large supply voltage guardbands to ensure a reliable voltage supply, that ends up increasing current densities and shortening chip lifetime. Practical and low-cost solutions to enhance lifetime are thus becoming essential, especially in dense 3D CMPs fabricated in scaled technologies. As noted earlier, such solutions must also consider the interdependence between BTI-induced circuit aging, supply voltage, and EM-induced PDN-aging.

Yet another challenge facing CMP designers that cannot be ignored is the rise in on-chip power dissipation. The slowdown of power scaling with technology scaling, due to leakage and reliability concerns [14], [15], has led to high chip power-densities, giving rise to the dark-silicon phenomenon, whereby a non-negligible fraction of the chip must be shut down at any given time to satisfy the chip power-budget. With the extent of dark-silicon increasing with every technology-generation [16], [17], designs are becoming increasingly power-limited rather than area-limited. Therefore, runtime power-saving techniques such as dynamic voltage scaling (DVS) are of paramount importance to extract much needed performance given a stringent chip-wide power-budget.

To simultaneously address all of the abovementioned challenges related to aging, power dissipation, and performance facing chip designers, in this work we propose a novel *runtime aging-aware application-mapping framework called ARTEMIS.* Our framework is intended for 3D NoC-based CMPs and aims to extend the operational lifetime of these chips, while meeting the dark-silicon power-budget (DS-PB) and application performance goals. The novel contributions of our work are summarized below:

- We propose a novel runtime application-mapping and DVS-scheduling framework that can adapt to different aging scenarios to extend the lifetime of a 3D NoC-based CMP;
- As the impacts of PDN-aging and circuit-aging (for cores and NoC routers) on system-performance are correlated, our framework considers aging in these key components, unlike any prior work, while making mapping decisions to alleviate system aging;
- Our methodology to evaluate system-aging and the resulting maximum-attainable performance accounts for progressive effects of IR-drops due to PDN-aging, V_T-degradation due to circuit-aging, as well as temperature profiles over the lifetime of the chip;
- We design a novel symmetric aging-enabled routing path allocation (SAR) heuristic to produce a balanced core-router aging profile to extend the lifetime of the NoC. In addition, SAR efficiently trades-off aging with network-congestion in the NoC;
- Our framework also meets chip-wide power constraints, thus finding applicability in contemporary power-constrained (dark-silicon afflicted) multicore designs.

8.2 Related Work

In recent years, several researchers have proposed run-time and design-time application mapping techniques to address the problem of circuit-aging in CMPs. Tiwari et al. [124] suggest mapping high-power tasks onto faster (less-aged) cores and low-power tasks onto slower cores, thereby “hiding” the aging in the chip. At the same time, they propose to lessen aging by scaling the supply voltage or the threshold voltage. Feng et al. [125] perform “local wear-leveling” by scheduling tasks on cores while considering circuit-aging in sub-core components. Such wear-leveling approaches where “younger” cores are prioritized over aged cores without considering application-performance (frequency) requirements lead to higher leakage power dissipation as faster cores are also leakier, which expedites aging. Thus, in the dark-silicon regime where performance is closely tied to power, wear-leveling techniques are usually sub-optimal.

More recent works propose aging-aware frameworks that discretize target lifetime of the chip into finer lifetime constraints, and perform runtime management to satisfy a pre-defined target lifetime and system performance goal. For instance, Mintarno et al. [120] use frequency, voltage, and cooling power as control parameters to optimize the energy-efficiency of a system while meeting lifetime targets. Paterna et al. [126] propose a linear-programming based task-allocation solution to optimize energy, while [127] performs voltage tuning over shorter time-intervals to meet aging constraints over longer time-intervals. But these techniques are either too time consuming to be viable for runtime decision making, or require precise knowledge of future application characteristics, which may not be available in many environments where CMPs are used. *To the best of the authors’ knowledge, our work is the first work on lifetime-aware application mapping at runtime that considers the impact of PDN-aging on the lifetime of the chip, and is also tailored for the power-constrained dark-silicon design regime.*

Aging is also a concern for NoC fabrics. But very few works have investigated design-techniques that extend the service life of the NoC fabric. In particular, Bhardwaj et al. [102] have proposed an aging-aware adaptive routing algorithm that routes packets along the paths that are both less congested and experience smaller aging stress. But the authors do not consider aging in compute-cores. *Our work represents one of the first efforts to extend lifetime of the entire chip by producing a balanced core-router aging profile, with our proposed symmetric aging-enabled routing path allocation (SAR) heuristic.*

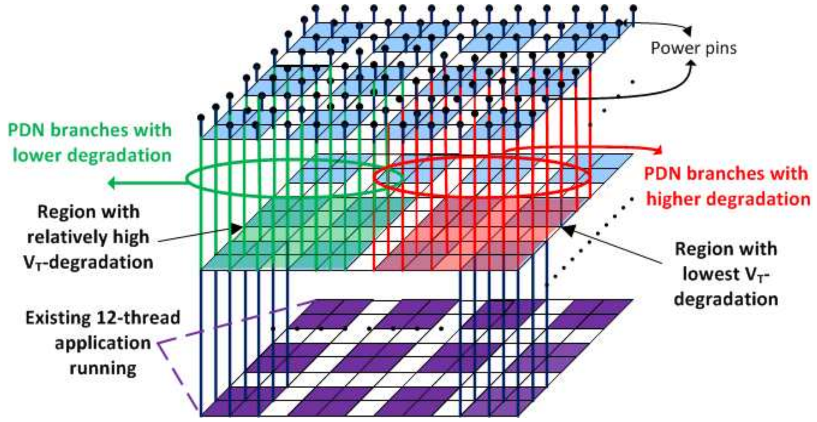

Power pins
PDN branches with lower degradation
PDN branches with higher degradation
Region with relatively high V_T-degradation
Region with lowest V_T-degradation
Existing 12-thread application running

Figure 47: Example of a 3D package for a 36-core CMP (4x3x3 3D-mesh) with a regular 3D power-grid that has 108 external power-pins and 108 grid-points per tier (16 grid-points supplying to each core). Not all vertical branches of PDN are shown, for brevity.

8.3 Motivation

In this section, we illustrate the advantages of our *ARTEMIS* framework with the help of a small example. We consider a scenario in which applications arrive at runtime to be executed on a 36-core CMP with a core capable of executing a single thread (task) at a time. The example assumes a 4-thread application being mapped on to the cores of a 3D-CMP at time *t*, when a 12-thread application is already executing on the bottom tier (shown in purple in figure 47). When an aging-aware wear-leveling technique based on prior work [124], [125] is used, the application would be mapped to the rectangular region (shown in red) containing cores with the least V_T-degradation, i.e., the region with the youngest cores. Observe that the vertical PDN branches supplying to this region have high degradation (higher resistance values due to past stress). Alternatively, although the green rectangular region has more V_T-degradation, the PDN IR-drops sustained by it are lower than the red region. By always prioritizing mapping of applications on to the youngest cores, without considering the resulting impact on EM-induced degradation in the PDN, resistances of already stressed PDN-wires would be further increased, thus exacerbating PDN-degradation. Therefore, *ARTEMIS* considers both V_T-degradation and PDN-degradation while making mapping decisions to limit PDN-degradation while guaranteeing that performance and power constraints are met on a 3D NoC-based CMP.

Additionally, the maximum frequency of a core is affected by both the PDN IR-drops (which affects V_{dd}) as well as V_T-degradation:

$$f_{max} = \frac{\mu(V_{dd}-V_T)^{\alpha}}{C_0.V_{dd}} \qquad \text{….. (8.1)}$$

where α and μ are technology-dependent constants, and C_0 is switching capacitance of the critical path [98]. Thus, any mapping solution obtained without consideration of the IR-drops experienced by cores can potentially lead to undesirable timing-errors.

In summary, the wear-leveling based application-mapping approach (i.e., always choosing the youngest cores) that is used in several prior works would increase leakage power, and hence temperature, thus resulting in higher circuit-aging. In addition, higher leakage power dissipation would cause increased supply currents to be drawn from the PDN resulting in higher PDN-aging. In contrast, *ARTEMIS* prioritizes older (slower) cores that can support application frequency constraints without requiring hiking of V_{dd}-levels.

8.4 Problem Formulation

8.4.1 Modeling BTI-induced Circuit Aging

In this work, we model circuit aging effects arising from BTI-induced circuit degradation, as BTI has been found to be one of the most dominant aging mechanisms in emerging semiconductor technologies. But, our framework is capable of supporting models of other aging-mechanisms (HCI, TDDB etc.) as well. Velamala et al. [128], [129] have shown that a Trapping/Detrapping (TD) based BTI model is capable of accurately predicting the degradation under a sequence of V_{dd}'s used in the DVS operation. They have noted that when the supply voltage is changed from a higher V_{dd} to lower V_{dd}, the circuit-degradation undergoes recovery; this recovery behavior is not captured by conventional Reaction-Diffusion (RD) models. Therefore, our analysis of circuit-aging over the CMP-lifetime is based on the long-term aging prediction model proposed in [128], which accounts for different V_{dd}-levels over time. We estimate the effective ΔV_T increase that a component (core or NoC router) experiences over a time-interval of t using Eq. (8.2) and (8.3):

$$\Delta V_T(t) = L.[A + Blog(1 + Ct)] \quad .. (8.2)$$

$$L = K_1.\exp\left(\frac{-E_0}{kT}\right).\left\{exp\left(\frac{\beta V_1}{T_{ox}kT}\right).\alpha_1 + \cdots + exp\left(\frac{\beta V_S}{T_{ox}kT}\right).\alpha_S\right\} \quad .. (8.3)$$

where V_i is the i^{th} V_{dd}-level utilized by the component for a time-duration α_i, S is total number of allowable V_{dd}-levels, and $\{\alpha_1+\alpha_2+...+\alpha_S\} \leq t$. T is the average temperature of the component during corresponding α_i. We use parameter-values in Eq. (8.2) and (8.3) from [118], [128]-[130] that are validated against silicon data.

8.4.2 Modeling EM-induced PDN Aging

To model the phenomena of void nucleation and void growth in every horizontal and vertical on-chip wire, particularly those in the PDN that are under the most stress, we use the EM model proposed in [131] for copper (Cu) interconnects in the power grid. Equation 7.4 gives the time t_n at which the void nucleates:

$$t_n = \frac{K_t}{D_{eff}}, \quad K_t = \frac{\pi}{4}\left(\frac{(\sigma_c - \sigma_{th})^2 \Omega k_B T}{\left(eZ_{eff}\rho j\right)^2 B}\right) \quad ..... (8.4)$$

where, D_{eff} is the effective diffusivity, σ_c is the critical stress, σ_{th} is the thermal stress, eZ_{eff} is the effective charge, ρ is the resistivity of copper, j is the current density in the wire, and B is the effective bulk modulus for the Cu-dielectric system. Once the void nucleates at time t_n, then at an observation time t_0, the length of the void L_{void} is:

$$L_{void}(t_0) = \left(\frac{D_{eff}}{k_B T}\right) eZ_{eff}\rho j(t_0 - t_n) \quad ..... (8.5)$$

The length of the void in a copper wire increases with the product of electrical current and the time-duration for which the current flows through it. The length of the void in turn determines the increased resistance (*ΔR*) or degradation of the wire, which is given by:

$$\Delta R = c.R_0\left(\frac{L_{void}}{L_{wire}}\right) \text{ ….. (8.6)}$$

where R_0 is the resistance, constant *c* depends upon the resistivity and cross-sectional area, and L_{wire} is the interconnect wire-segment length.

8.4.3 Inputs, Assumptions, and Problem Objective

We have the following inputs to our problem:

- A 3D NoC-based CMP with a 3D mesh NoC, of dimensions (dim_x, dim_y, dim_z) and number of tiles $N = dim_x \times dim_y \times dim_z$ with each tile containing a compute core and a NoC router;
- A set *S* of candidate supply voltage (V_{dd}) levels for the chip;
- A chip-wide dark-silicon power budget (DS-PB);
- An application task graph for each application: vertices with task execution-times on compute cores and edges with inter-task communication volumes; execution time and volume values are assumed available from offline profiling;
- Degree of parallelism (DoP) of each application, and a set of *n* admissible rectangular/cuboidal shapes (*x*, *y*, and *z* dimensions) of regions that it could be mapped to {$B_1, …, B_n$}; e.g., a tuple set {2×4×1, 4×2×1, 2×2×2} for an application with DoP=8;
- A minimum operating frequency and a maximum execution time constraint for each application;
- A regular 3D power grid, with $p \times p$ grid-points supplying to each core and $dim_x \times dim_y \times p \times p$ power-pins at the top of the 3D-die, and an air-cooled heat sink at the bottom of the 3D-die;

We make the following assumptions in our work:

- There exists one-to-one mapping between tasks and cores, i.e., a core can execute only one task (thread) at any given time;

- Applications are mapped contiguously on non-overlapping rectangular (on single tier) or cuboidal (across multiple tiers) shaped regions of the 3D CMP for inter-application isolation and more optimized communication-profiles (as recommended in prior works such as [116]); and thread-migration is not considered;
- A chip-wide supply voltage exists that can be scaled using DVS at runtime, as the overheads of implementing DVS at a per–core granularity are high for systems with very large core counts [115];
- Similar to prior-works (e.g., [120], [125], [127]) we assume presence of on-chip aging-sensors [133], [134] that provide aging information of compute-cores and NoC routers to our framework; we also assume voltage-sensors [132] at each power-input (PDN-grid-point) of a CMP-tile that track the severity of IR-drops (i.e., the degree of PDN-degradation on PDN-paths supplying to that core);
- From the on-chip sensors, runtime sensed data (at a per-tile granularity), in terms of threshold-voltage (V_T) distribution and maximum IR-drops, is available after every *epoch*; the aging models (discussed in sections 8.4.1 and 8.4.2) emulate runtime readings from sensors on real chips. We define an *epoch* as the time-period during which the aging profile of the chip can be assumed to be constant;
- The 3D-CMP chip is rendered unusable (end of lifetime) when an incoming application is unable to be executed (i.e., when its minimum application frequency requirement cannot be supported) on any of the allowed rectangular or cuboidal regions, when there are no other applications running at that time.

Objective: Given the above inputs and assumptions, our objective with the *ARTEMIS* framework is to perform runtime application-mapping and DVS-scheduling on a given 3D NoC-based CMP platform such that the total number of applications executed over the lifetime of the chip is maximized, while all application-specific minimum operating frequency- and maximum runtime-constraints, as well as CMP platform-specific DS-PB constraints are satisfied.

8.5 ARTEMIS Framework: Overview

The key aspects of our proposed framework are illustrated in Figure 48. The knowledge of the chip-aging profile (updated at the end of each epoch) is continuously utilized in the application-mapping and DVS scheduling steps. The application-mapping at runtime consists of mapping the application task-graph on to a chosen rectangular- or cuboidal-shaped region of tiles on the 3D CMP admissible for the application (from the list $\{B_1, \ldots, B_n\}$), as well as performing routing path allocation of the intra-application communication-flows on the 3D NoC. At any given time, the scaling-down of V_{dd} via DVS-scheduling to save power and to limit aging is constrained by the frequency-constraints of the applications running on the 3D NoC-based CMP, whereas scaling-up of V_{dd} is constrained by the DS-PB.

The *ARTEMIS* framework is executed in two nested procedures: *(i)* Aging-aware application mapping and DVS (inner loop); and *(ii)* Circuit- and PDN-aging analyses (outer loop). These procedures are discussed in sections 8.5.1 and 8.5.2 respectively, and the design flows for the two procedures are shown in figures 49(a) and 49(b).

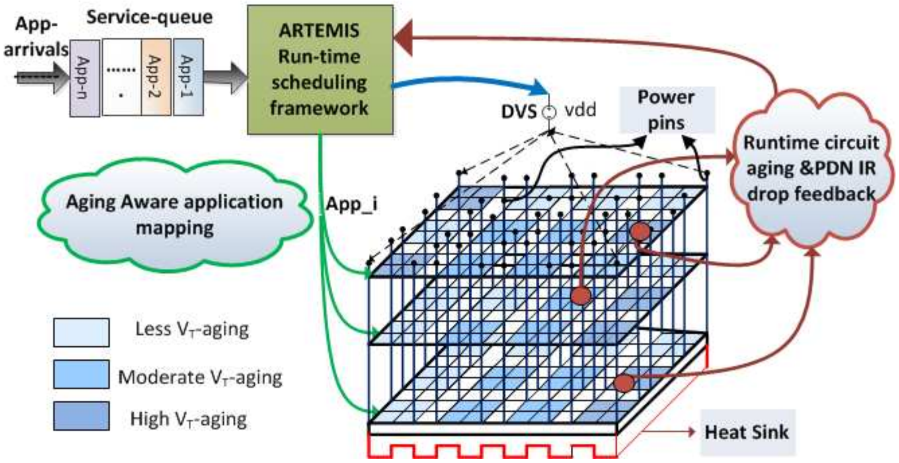
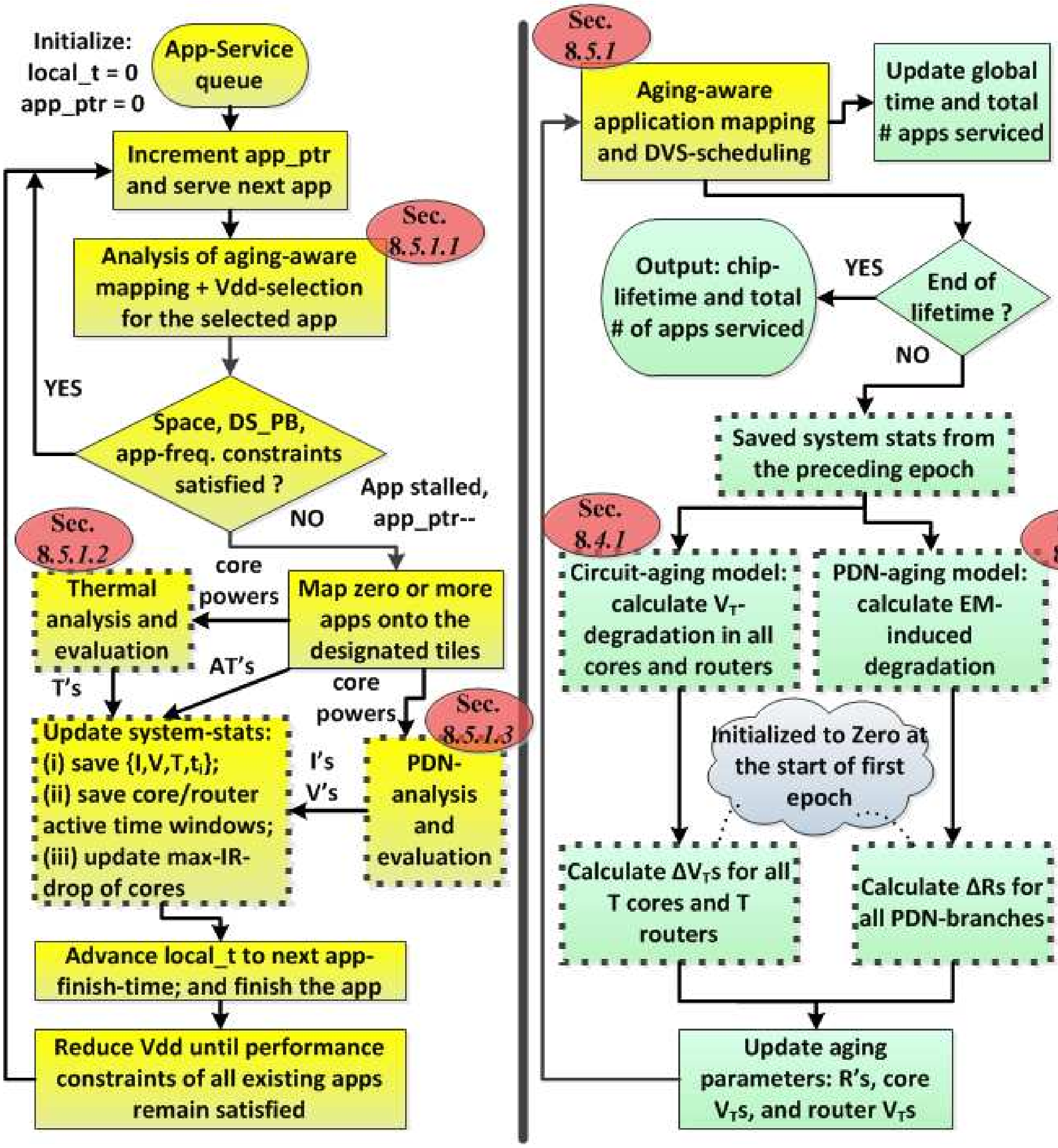


Figure 48: Overview of ARTEMIS runtime aging-aware application-mapping and DVS-scheduling framework

8.5.1 Aging-aware Application Mapping and DVS Scheduling

During each epoch at runtime, we assume that applications arrive for execution on the 3D NoC-based CMP. Suppose a sequence of ℓ applications arrives in an epoch. *ARTEMIS* is responsible for mapping these ℓ applications onto the CMP during the epoch. If an application cannot be mapped immediately after it arrives, it is kept in a service queue, and mapped at a later time. We assume processing of the service-queue on a first-come-first serve basis (although a priority-based processing approach could also be used). At the end of the epoch, aging information is updated from the on-chip circuit-aging sensors as well as the voltage sensors to be utilized by *ARTEMIS* during the next epoch. Subsequently, a new application sequence is serviced at the start of each new epoch, and this process continues until the end of the lifetime of the 3D NoC-based CMP.

To keep track of cumulative CMP-execution-time, we utilize a local time counter for the application-mapping and DVS scheduling procedure (inner loop) that is reset at the start of each epoch, and a global time (outer loop) that is augmented by the local time at the end of each epoch. At the start of an epoch, the application-service queue is initialized to point to the first application (*app_ptr*=0), and the local time is initialized to zero, as shown in figure 49(a). During the execution of the application-sequence, the service-queue is triggered, (i.e., new applications are serviced) when an application arrives or an existing one terminates. Once the service queue is triggered, an application instance is removed from the front of the queue and processed in the aging-aware mapping and V_{dd} selection phase (figure 49(a); discussed in section 8.5.1.1). Applications from the queue continue to be processed one-by-one until an "application stall" is detected. An application in a service queue can be stalled only due to the following reasons: *(i)* available tile constraints on the 3D-die; *(ii)* DS-PB constraint; *(iii)* application frequency constraints for the given degradation profile of the 3D-CMP. Note that if an application is stalled when there are no other applications running, i.e., the chip-degradation (V_T- and PDN-degradation combined) precludes it from meeting the application-frequency constraints, the 3D NoC-based CMP is considered as no longer usable and has reached its end of life.


Initialize:
local_t = 0
app_ptr = 0
App-Service queue
Increment app_ptr and serve next app
Sec. 8.5.1.1
Analysis of aging-aware mapping + Vdd-selection for the selected app
YES
Space, DS_PB, app-freq. constraints satisfied ?
NO
App stalled, app_ptr--
Sec. 8.5.1.2
Thermal analysis and evaluation
core powers
Map zero or more apps onto the designated tiles
T's
AT's
core powers
Sec. 8.5.1.3
Update system-stats: (i) save {I,V,T,t_i}; (ii) save core/router active time windows; (iii) update max-IR-drop of cores
I's
V's
PDN-analysis and evaluation
Advance local_t to next app-finish-time; and finish the app
Reduce Vdd until performance constraints of all existing apps remain satisfied
Sec. 8.5.1
Aging-aware application mapping and DVS-scheduling
Update global time and total # apps serviced
Output: chip-lifetime and total # of apps serviced
YES
End of lifetime ?
NO
Saved system stats from the preceding epoch
Sec. 8.4.1
Sec. 8.4.2
Circuit-aging model: calculate V_T-degradation in all cores and routers
PDN-aging model: calculate EM-induced degradation
Initialized to Zero at the start of first epoch
Calculate ΔV_Ts for all T cores and T routers
Calculate ΔRs for all PDN-branches
Update aging parameters: R's, core V_Ts, and router V_Ts

(a) **(b)**

Figure 49: ARTEMIS design-flow: (a) Aging-aware application-mapping and DVS scheduling (inner loop; section 8.5.1); (b) Circuit- and PDN-aging analyses (outer loop; section 8.5.2). The boxes with dotted outlines are used as part of our aging-simulation framework; however these steps are not required on real hardware where runtime aging information is assumed to be available from on-chip sensors.

When an "application stall" is detected or the application-service-queue becomes empty for the current epoch, the application(s) that have been processed by the mapping/selection phase (discussed in section 8.5.1.1) are mapped on to the appropriate tiles chosen by the phase. At this time (local time), either one or more new applications are mapped on to the 3D-CMP or an application just ended (which triggered the service-queue), thus the steady-state computation-profile (i.e., tile-power values, the resulting supply currents in the PDN, and thermal-profile) of the 3D-CMP changes. To evaluate the new computation-profile, given the tile-power-distribution, thermal-analysis is performed to re-evaluate the thermal-profile (at per tile granularity) and PDN-analysis is performed to evaluate all the branch currents and voltage-drops at all grid-points in the PDN. Also, a worst case IR-drop value (WC-IR-drop, which is the maximum voltage-drop out of all grid-points supplying to a tile) is evaluated for each of the *N* tiles. The WC-IR-drop value is updated for each tile (at every change of computation-profile) over the chip-lifetime and continuously used to calculate the maximum-frequency of the tile (for a given V_{dd}) in the application-mapping step. The thermal- and PDN-analysis is discussed in sections 8.5.1.2 and 8.5.1.3, respectively.

After the mapping/selection phase, thermal-analysis, and PDN-analysis, the updated computation-profile including current (*I*), voltage (*V*), and temperature (*T*) values, as well as the WC-IR-drop values in the time-window t_i for this (i^{th}) computation-profile, is saved in the *system-stats*, as shown in figure 49(a). Additionally, if one or more applications are mapped at the current local time, the active-times (AT's) of compute-cores and NoC routers are calculated for each newly mapped application. For each tile, these AT's could be represented as $\{C_j, R_j, t_j\}$, where C_j and R_j take values of '1' or '0' depending on whether the corresponding compute-core or router is active during the time-window t_j. These AT's for compute-cores and routers are also saved in *system-stats*. The system-stats for all time-windows for the entire epoch are eventually utilized for aging-analyses (in the outer loop) at the end of the current epoch.

After updating system-stats, local time is advanced to the next application finish-time, and the corresponding application is completed, (figure 49(a)). As part of our DVS strategy to save power and limit aging, on completion of any application, we reduce V_{dd} to the lowest allowable level that would not introduce any violations in frequency constraints of existing (already running) applications.

8.5.1.1 Application-specific mapping and V_{dd}-selection

For the application under consideration, this phase consists of three steps: *(i)* circuit- and PDN-aging aware region selection and V_{dd}-selection, *(ii*) communication-aware task-to-tile mapping, and *(iii)* NoC routing path allocation. We describe each of these steps below.

(i) Circuit- and PDN-aging aware region selection and voltage-selection: In our framework, an application with a given DoP can be mapped on to rectangular or cuboidal regions on the 3D CMP, with shapes to be chosen from a pre-defined list $\{B_1, \ldots, B_n\}$ for that application. All intra-application communication is contained within t closed region, thus application-isolation is maintained and communication cross-interference is eliminated. Our heuristic in this step utilizes the V_T-degradation profile and the WC-IR-drop profile of the 3D CMP. The objective is to find the region on the 3D-mesh (with one of the admissible shapes) so as to: *(a)* minimize leakage-power; *(b)* minimize EM-induced degradation of PDN-paths supplying to cores with high WC-IR-drops; *(c)* satisfy the frequency-constraint of the application by all cores within the region; *(d)* satisfy the DS-PB. In other words, we search for CMP-regions with most circuit-aging that satisfy minimum application-frequency constraints and have least WC-IR-drops. To this end, we define the following cost-function (Ψ) for joint optimization of leakage-power and PDN-degradation:

$$\Psi = \sum_{k=1}^{k=DoP} \left\{ \alpha . \left(\frac{\max\_V_T - V_{Tk}}{\max\_V_T - nom\_V_T} \right) + \beta \left(\frac{WC\_IR\_drop_k}{\max\_IR\_drop} \right) \right\} \quad \ldots.. (8.7)$$

where, V_{Tk} is the effective V_T and *WC-IR-drop*$_k$ is the WC-IR-drop of the k^{th} core within the region of DoP cores; *nom_*V_T is the nominal (lowest) effective V_T-value of a core with no aging; and α and β are weighting coefficients. We define *max_*V_T as the maximum V_T value that the core can support for an ideal (zero) WC-IR-drop (at highest V_{dd}) while meeting the frequency-constraint of the application. Similarly, *max_IR_drop* is the maximum tolerable WC-IR-drop for a core for nominal V_T and highest V_{dd}. The pseudo code of our region (shape)-selection and V_{dd}- selection heuristic is shown in Algorithm 8.1.

Algorithm 8.1: Aging-aware region selection and V_{dd}-selection heuristic

Inputs: V_T-profile, WC-IR-drop profile, $\{B_1, \ldots, B_n\}$

```
1: while ($V_{dd} \leq max\_V_{dd}$) do {
2:   for each tile on the 3D NoC-based CMP do {
3:     assume this tile to be the minimum $x, y, z$ coordinates of the region
4:     for each shape in $\{B_1, \ldots, B_n\}$ do {
5:       check if all tiles (compute-cores and routers) satisfy app-frequency
6:        if not, go to next shape $B_i$ (step 4)
7:        check if DS-PB is satisfied
8:        calculate $\Psi$, choose this shape if least $\Psi$ AND DS-PB satisfied
9:      } // end for each shape …
10:  } // end for each tile …
11:  if (no valid region found AND no DS-PB violation)  hike $V_{dd}$
12: } //end while
13:  if no valid region found, then stall this application
```

output: a valid region to map the application and V_{dd}-level, or "stall"

The heuristic performs a simple exhaustive search over all tiles on the 3D-mesh and over all admissible shapes $\{B_1, \ldots, B_n\}$ for the application. The V_T-profile and WC-IR-drop profile inputs are used for calculating the value of Ψ. The region with the least Ψ value that satisfies the frequency-constraints (with maximum frequency for the selected V_{dd} level calculated using Eq. (8.1)) and at the same time does not violate the DS-PB (given that existing applications have been running), is selected for mapping the application under consideration. If no region on the 3D-mesh is found to satisfy the frequency-constraints, we repeat the search for successively higher V_{dd}-levels (which can allow using a higher frequency as per Eq. (8.1) with a better probability of meeting frequency-constraints) until either a valid region with minimal Ψ is found or the DS-PB is violated. If no valid region is found, an "application stall" event is initiated.

We now present the theoretical time-complexity of our heuristic. At most N tiles (total tiles on the 3D NoC-based CMP) are considered for the prospective mapping region. Note that DoP of the application (relatively small integer c – treated as a constant) number of tiles are to be evaluated for frequency and leakage-power at each of these iterations. As, the number of candidate V_{dd}-levels $|S|$ as well as the number of admissible shapes *n* are expected to be small constant integers, our region-selection step effectively runs in linear-time complexity with respect to the number of tiles, *N*: O(*cn*|*S*|*N*).

(ii) Communication-aware task-to-tile mapping: After the region on the 3D CMP has been selected (of size equal to application-DoP), our mapping heuristic maps the appropriate application-task-graph on to the CMP tiles. We use a fast and efficient communication-intensive incremental-mapping approach (similar to that used in prior works such as [84], [24]) suitable for runtime application.

(iii) Symmetric aging-enabled routing path allocation (SAR): In this step, we map the communication-flows of the current application on to the designated cuboidal region on the 3D NoC-based CMP. We propose an aging-enabled and congestion-aware routing scheme (SAR) to produce a balanced core-router aging profile and extend the lifetime of the 3D NoC. The main objective of SAR is to minimize the number of runtime scenarios where application-mapping on a given cuboidal region is precluded due to aging in routers. Note that an application can be mapped only if all tiles (each tile has a compute-core and a NoC-router) within the region under consideration satisfy the minimum application-frequency constraint. Prior work on aging- enabled routing (such as [102]) considers the aging in NoC-routers but does not consider the aging in compute-cores. Such an approach could lead to a somewhat imbalanced aging within tiles of the CMP, thus potentially preventing application mapping onto desirable CMP regions due to excessive aging in NoC routers. SAR on the other hand enables symmetric aging on individual tiles of the 3D-CMP to extend the service life of NoC routers. Additionally, SAR efficiently trades-off aging with network-congestion in the NoC by choosing routing paths to maximize NoC-lifetime while leveraging the knowledge of maximum execution time constraints of applications, i.e., the aging metric in the routing cost function is prioritized by varying degrees, given the time-slack available for application-completion.

To ensure a low-overhead implementation, path diversity, and deadlock freedom, our routing algorithm builds on the 4N-First turn model [101] for 3D-mesh NoCs. This routing algorithm is partially adaptive, and hence allows the flexibility to potentially select from among multiple next hop directions, at each router. We designed a cost-function for next-hop selection during routing that considers the difference between router-aging and core-aging (*router_V_T – core_V_T*) values to ensure balanced aging in CMP tiles. Moreover, as congestion in the NoC-links leads to excessive routing delays and thus longer application-runtimes, we prefer allocating flows to links with lesser communication-volumes. The following routing cost function (Rt_{cost}), which is a linear combination of the two normalized metrics, is used to make routing decisions at each hop along the path:

$$Rt_{cost} = \alpha_R . \frac{(V_T\ difference) - (minimum\ V_T\ difference\ )}{range\ of\ V_T\ difference} + \beta_R . \frac{(volume) - (minimum\ volume)}{range\ of\ volume} \quad \ldots\ldots (8.8)$$

where, α_R and β_R are weighting coefficients, *V_T difference* represents (*router_V_T – core_V_T*) of the candidate next hop router, and volume represents the existing communication-volume (already allocated while routing previous flows) on the link. SAR selects the next hop with the minimum routing-cost, Rt_{cost}, given in Eq. (8.8).

Communication delays are calculated from the application-frequency and NoC link bandwidths, and thus the current application-delay can be estimated from the already routed communication-flows. NoC routers and links in an application region run at the same frequency as the cores in the region (application-frequency). Note that the goal of SAR is to extend NoC lifetime while meeting application execution time constraints. Thus, the values of coefficients in Eq. (8.8), α_R and β_R, are re-evaluated after routing each flow, as shown below:

$$\beta_R = \{current\ app.\ delay\}/\{\delta.(app.\ execution\ time\ constraint)\}\ and\ \alpha_R = (1 - \beta_R) \quad \ldots\ldots (8.9)$$

Before any application communication flows are mapped to the NoC routers, we start with values α_R =1 and β_R =0. As flows are mapped and the estimated application-delay increases, the value of β_R increases (α_R decreases) proportionally until the application-delay reaches a significant fraction (δ) of the application execution time constraint. At this point (β_R =1 and α_R =0), SAR ceases to be aging-aware and routes on paths with minimum congestion exclusively, to meet the execution time constraint of the given application. Algorithm 8.2 below summarizes our symmetric-aging enabled routing scheme.

```
Algorithm 8.2: Symmetric aging-enabled routing path allocation
Inputs: Task-graph, execution time constraints, minimum frequency, task-
mapping of current application, $V_T$-profile of compute-cores and routers

1: Initialize $\alpha_R$=1 and $\beta_R$ =0
2: for all communication-flows do {
3:   for all hops on the minimal path do {
4:     choose the next hop with the least $Rt_{cost}$ (equation 8)
5:   } update $\alpha_R$ and $\beta_R$ (equation 9)
6: }
output: all flows of the application allocated on the cuboidal CMP- region
```

Note that the given application is executed on the actual 3D-CMP platform only after the analysis for routing path allocation is performed. The turn model rules are implemented in each router using simple combinational logic. The next hop selection information at each NoC router is stored in small next-hop routing tables that enable quick selection of the most appropriate next-hop direction based on the source and destination of a packet. Even for the largest sized, 32-threaded applications mapped onto a {4x4x2} cuboid on the 3D-CMP platforms we considered, we found that the upper bound on number of communication-flows (with unique source-destination pairs) needed to be routed through any router is 64, with our 3D turn-model based minimal routing scheme. Thus, a NoC router on the 3D CMP would need a next hop table of up to 64 entries. Assuming 2 bits for the output port and 6 bits for the source and destination each, the footprint of the NoC routing table is only 896 bits. Thus, the hardware overheads of implementing SAR are quite reasonable.

8.5.1.2 Thermal-analysis and evaluation

To perform thermal evaluation of a given computation-profile in our framework, we utilize the open-source thermal emulator 3D-ICE 2.2.5 [71] which supports steady-state thermal analysis of 3D ICs with a conventional air-cooled heat-sink. For the given power-profile, the tool outputs the core-temperatures (*T*'s) on the 3D die.

8.5.1.3 PDN-analysis and evaluation

The supply current drawn by each core is calculated from the core-power and current V_{dd}-level. Given the supply current requirements of the N cores on the 3D-CMP, we created a linear programming (LP) formulation and used lp_solve [75] to solve for the grid-point voltages and currents flowing in the 3D regular power grid. This enables the updating of {V's, I's} in the power-grid and WC-IR-drops of cores in the 3D CMP, for the given time-window (t_i) of the computation-profile. APPENDIX I gives a more detailed discussion of our LP formulation, including elaboration of constraints and equations.

8.5.2 Circuit- and PDN-Aging Analyses

In the outer loop of our framework (figure 49 (b)), we utilize *system-stats* generated by the inner loop over the last epoch to perform aging-analysis at the end of the epoch. Given the system-stats for the last epoch, this analysis is used to calculate the rise in effective V_T values (ΔV_T's) of all cores and NoC routers on the 3D CMP, as well as the rise in resistance values (ΔR's) of all vertical and horizontal PDN-branches, using the circuit- and PDN-aging information. The BTI-induced circuit-aging of compute-core and NoC router components are calculated (discussed in section 8.4.1) using the V's and T's experienced by these components during all of their AT's over an entire epoch. The EM-induced PDN-degradation in PDN-branches (discussed in section 8.4.2) is calculated using I's for all computation-profiles of the epoch. As effects of EM are far less dominant in signal interconnects compared to PDN-interconnects [12], [13], we ignore EM-induced aging in the NoC-links.

Note that the active-time windows of compute-cores and routers {C_j's, R_j's, t_j's} may not be aligned with the chip-wide computation-profile windows {V's, T's, I's, t_i's}; therefore, in circuit-aging calculations, the component AT's are required to be split into multiple time-windows where computation-profiles change. Also, at the start of the very first epoch, the R's and V_T's are initialized with nominal values representing no degradation and the ΔR's and ΔV_T's are initialized to zero-values. Lastly, the updated aging profiles are leveraged to make mapping decisions in the next epoch. When the end of lifetime is encountered (discussed in section 8.5.1), the aging analyses procedure outputs the lifetime of the 3D-CMP in terms of both the total system-execution-time (global time) and the total number of applications serviced during this time (figure 49 (b)).

8.6 Experimental Studies

8.6.1 Experimental Setup

Our experiments were conducted using 13 different parallel application benchmarks taken from the well-known SPLASH-2 [58] and PARSEC [56] benchmark suites. We profiled the execution-time, power dissipation, and degree of memory-intensity of each application for different application-DoPs by performing multicore simulations using the open-source tools SNIPER [123] and McPAT [103]. For each benchmark, the DoP resulting in highest performance was obtained from this profiling study and selected as the fixed DoP value for that benchmark. These DoP values ranged from 4 to 32. Note that increasing DoP beyond this baseline value for each benchmark resulted in lower performance, due to factors such as high inter-thread synchronization and communication overheads.

We categorized the 13 benchmarks into two groups: *(i)* communication-intensive benchmarks - {*cholesky, fft, radix, raytrace, dedup, canneal, and vips*}; and *(ii)* compute-intensive benchmarks – {*swaptions, fluidanimate, streamcluster, blackscholes, radix, bodytrack, and radiosity*}. As *radix* has properties of both, we use it in both groups. In our analyses, we employ three types of application sequence groups as inputs to our framework: communication-intensive, compute-intensive, and mixed (using all 13 applications). We assume each application-sequence to have 100 randomly ordered application-instances selected from the respective group. To enhance the statistical significance of our results, we averaged results for five different randomly generated application-sequences for each group. To simulate the chip-lifetime within a reasonable time, we extrapolate the effects of aging over 500 such sequences, making the total number of application-instances executed within an epoch to be approximately ℓ = 50,000. Simulation times for *ARTEMIS* to simulate one lifetime were between 6 and 10 hours. Communication-intensive application workloads typically entailed larger simulation times because of longer chip lifetimes (see section 8.6.2), compared to the computation-intensive workloads.

We consider a 60-core 3D-mesh NoC based CMP platform, with dimensions 5×4×3 ($dim_x \times dim_y \times dim_z$). Our SNIPER simulations for application-profiling capture performance and power consumption at the 22nm node. Seven operating voltage levels are used, ($|S|$=7): 0.7V, 0.75, 0.8V, 0.85V, 0.9V, 0.95V, and 1.0V. Frequency-requirements of different applications are set between 1.5GHz and 2GHz. The following region-dimensions-lists $\{B_1, \ldots, B_n\}$ for applications (for the given DoPs) are employed: {2×2×1} for DoP = 4, {4×2×1, 2×4×1, 2×2×2} for DoP = 8, {4×2×2, 2×4×2, 4×4×1} for DoP = 16, and {4×4×2} for DoP = 32. The dark-silicon power-budget (DS-PB) is conservatively set at 85W. The regular 3D-PDN power grid is modeled based on guidelines provided in [7]. With 20 cores on each tier, a total of 320 input power pins are used with $n^2=16$ grid-points for each core. Nominal (initial non-aged) values of branch resistances are assumed to be 50mΩ [7], with 25 μm^2 cross-sectional area.

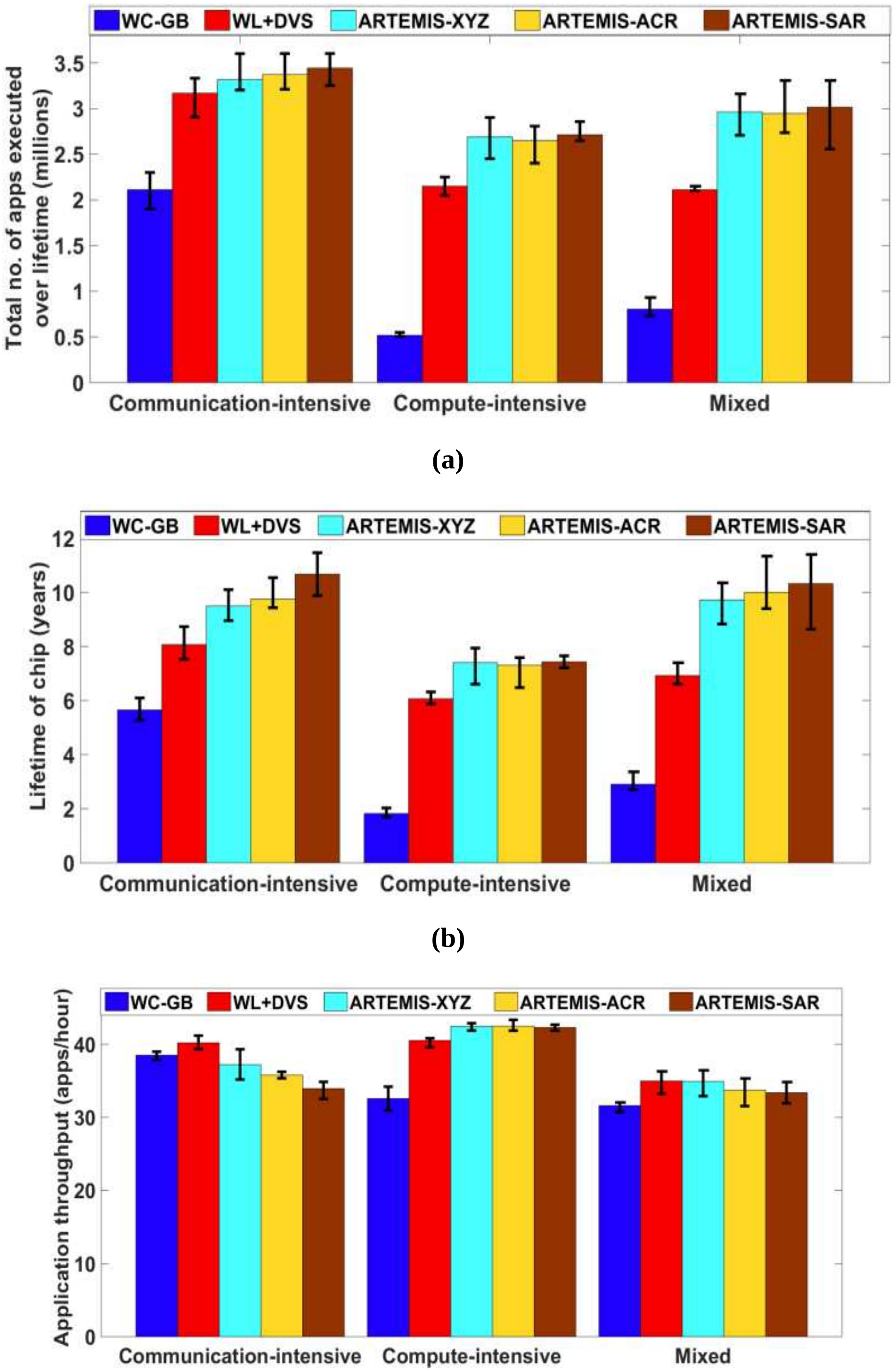
For our circuit-aging calculations, we assume a nominal effective V_T of 0.3V for un-aged cores and routers. In our combined cost function calculations (Ψ in E q. (7)) for the aging-aware region-selection heuristic, we use α=β=0.5 (empirically derived to achieve the longest lifetimes); *max_IR_drop* and *max_*V_T are set to 0.3V and 0.5V respectively, based on Eq. (8.1), with operating frequency requirement of 2GHz. In our SAR heuristic we use δ=0.6, to calculate the value of β_R, for an appropriate trade-off between application performance and aging. In our experiments, an epoch interval can range between 25 to 35 days, depending on the power profile, execution-times, and average DoPs of the application workload, as well as the degree of aging in the chip. Given the relatively slow rate of aging, such an aging-measurement interval has been found to be appropriate for runtime frameworks [120]. Also, as the overheads incurred due to employing aging sensors have been reported to be quite small (power dissipation of 84.7nW, sensing-latency of 100μs, and area of 77.3μm^2 per sensor at 45nm technology node) in [133], we ignore them in our calculations.

8.6.2 Experimental Results

Our experiments compare three variants of the proposed *ARTEMIS* framework with two other runtime mapping approaches derived from prior work. These prior works are designed for 2D CMPs, so we extend them to 3D CMPs for a fair comparison.

To investigate the effectiveness of the circuit-aging (leakage) and PDN-aging aware region- and voltage-selection/mapping techniques, we adapt our *ARTEMIS* framework to use an XYZ-routing scheme (*ARTEMIS-XYZ*) and compare the results obtained with two other mapping techniques that also use the same XYZ routing scheme: *(i) traditional worst-case guard-banding approach (WC-GB)*: In this approach, region selection is done based on the runtime area constrained mapping approach from [116] that attempts to fit the maximum number of applications on the chip. To satisfy the application-performance requirements for an extended period of time, a high V_{dd}=1.0V is used at all times. This approach does not assume runtime inputs from aging-sensors to make mapping decisions and thus is not aging-aware; *(ii) wear-leveling approach with DVS (WL+DVS)*: In this approach, region-selection for application-mapping is always done based on the lowest average V_T–degradation in cores, as proposed in [124], [125]; in addition, V_{dd} is opportunistically reduced when possible and adaptively hiked with aging to meet application performance constraints.

Additionally, we adapt our *ARTEMIS* framework to use an aging- and congestion-aware routing scheme (ACR) obtained from prior work in [102]. Finally, we also include results for our *ARTEMIS* framework with the proposed symmetric aging-enabled routing (SAR) scheme. Thus the comparison between *ARTEMIS-XYZ*, *ARTEMIS-ACR*, and *ARTEMIS-SAR* allows us to determine the most effective 3D NoC routing scheme that can help improve lifetime in 3D NoC-based CMPs while meeting performance and power constraints.



(a)

(b)

(c)

Figure 50: Results comparing ARTEMIS framework variants with other mapping approaches from prior work, for workloads that combine various SPLASH-2 and PARSEC benchmarks: (a) Total number of applications serviced over lifetime, (b) lifetime (years), (c) application-throughput over lifetime (applications/hour)

Figure 50(a) shows the total number of applications serviced over the chip lifetime, figure 50(b) shows total CMP-lifetime (total system-execution-time), and figure 50(c) shows the application-throughput extracted over the service-life of the CMP for all the compared frameworks, across the three different types of application-input-sequences. The error-bars in all our plotted results represent the range of results across simulations with five different randomly generated application-sequences (with individual applications derived from the SPLASH-2 and PARSEC benchmarks, as discussed earlier).

As expected, the WL+DVS framework outperforms the WC-GB approach that does not perform DVS. By intelligently selecting application-regions on the 3D-die with its region-selection heuristic, our *ARTEMIS* frameworks (*ARTEMIS-XYZ, ARTEMIS-ACR,* and *ARTEMIS-SAR*) achieve a further reduction in leakage-power dissipation and stress on the more highly degraded PDN-paths. The *ARTEMIS* frameworks produce 9%–40% (25% average) improvement in the total number of applications serviced over the next best framework, WL+DVS, as well as significant improvements in total CMP-lifetime, as can be seen from figures 50(a)-(b).

For communication-intensive applications, we observed far less percentage of dark-silicon, approximately 0%–15% (depending on V_{dd}-levels and workload profiles), compared to compute-intensive applications where dark-silicon is approximately 10%–33%. A lower percentage of dark-silicon is indicative of more active cores running with less stress, whereas a higher percentage of dark-silicon indicates fewer active cores that are running with greater stress. Thus, communication-intensive applications experience less aggressive aging (because of their lower %dark-silicon), which results in more of these applications being executed over the chip lifetime and also a higher lifetime compared to compute intensive applications. Figures 50(a)-(b) corroborate this observation. Also, most communication-intensive applications generate relatively low current-densities in the PDN, i.e., PDN-degradation is slower relative to circuit-degradation, which limits the improvements obtained by the *ARTEMIS* frameworks for such applications, as can be seen from figure 50(a).

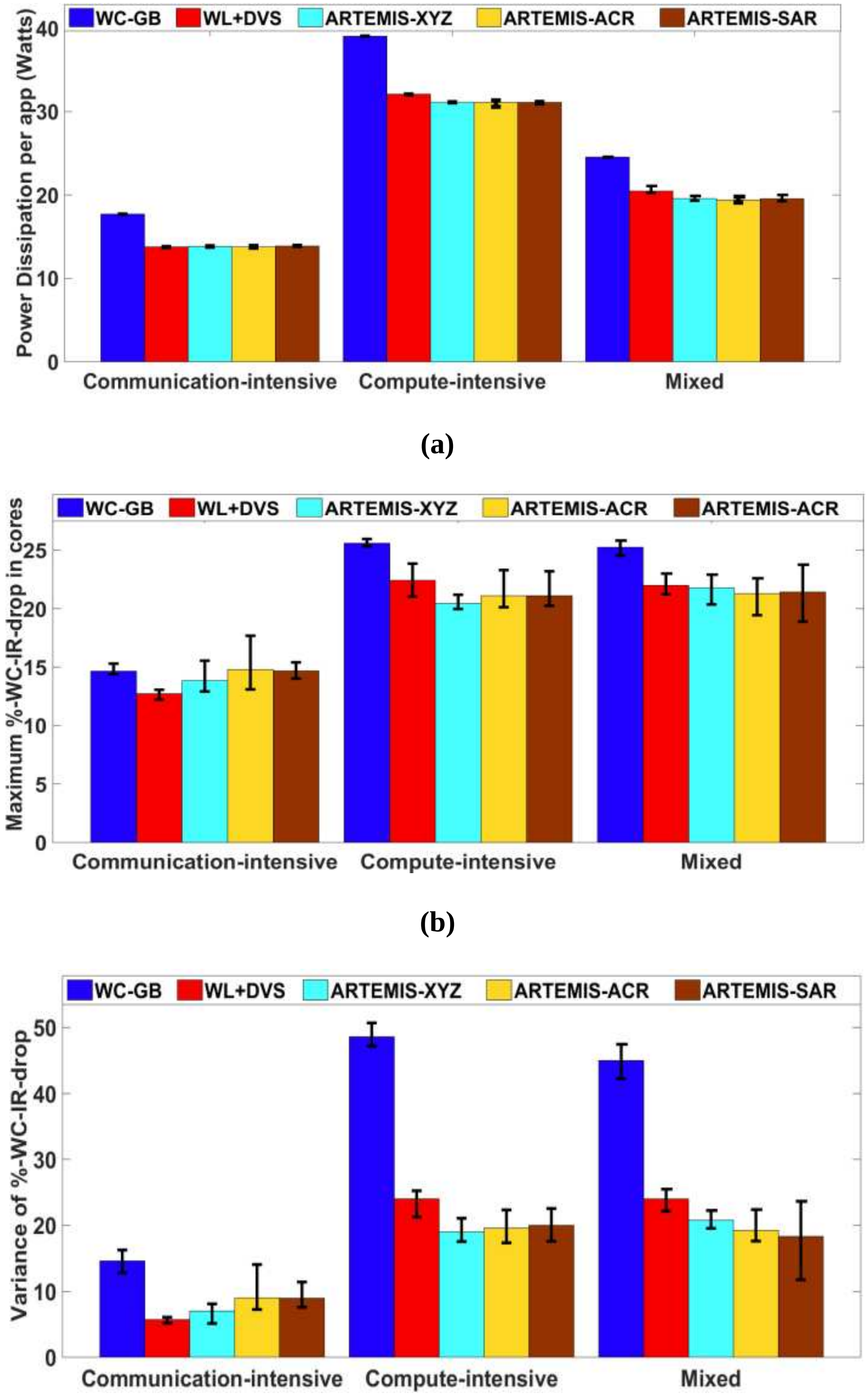
Next, we present an analysis of lifetime improvements obtained when our proposed symmetric aging-aware routing path allocation (SAR) heuristic is used with *ARTEMIS* (*ARTEMIS-SAR)*, compared to the *ARTEMIS-XYZ* and *ARTEMIS-ACR* frameworks. Our SAR heuristic enables better balancing of aging between compute-cores and their associated NoC-routers. While executing communication-intensive workloads exclusively, where the rate of aging in routers is comparable to that of core-aging, SAR minimizes the number of runtime scenarios when mapping of an application is stalled due to aged routers, thereby extending the system-lifetime. Observe in figure 50(a) that *ARTEMIS-SAR* produces notable improvements in number of applications executed over lifetime, compared to *ARTEMIS-XYZ* (by 4%) and *ARTEMIS-ACR* (by 2.2%) with comparable application-throughput (as shown in figure 50(c)), for communication-intensive workloads. However, the choice of routing scheme has very little effect on lifetimes for compute-intensive and mixed workloads, where NoC-aging does not determine the service-life of the chip.

We also show experimental results related to the power dissipation, PDN performance, and V_T degradation profile on the 3D CMP when using the different mapping frameworks, in Figure 51.

A comparison of the average power-dissipated per application over the chip lifetime is shown in figure 51(a). As expected, the WC-GB framework, which does not utilize DVS, dissipates significantly more power. The leakage-optimizing mapping in *ARTEMIS* results in up to a 5.5% improvement for compute-intensive workloads (2.8% on average for all workloads) in total power/application over WL+DVS.

We also analyze the distribution of percentage worst-case IR-drops (%WC-IR-drops) at the end of lifetime with different frameworks. Figure 51(b) shows the maximum %WC-IR-drops obtained for different frameworks at the end of chip lifetime. The aging-unaware WC-GB framework maps applications such that some cores are more heavily loaded than others, thus resulting in the shortest lifetimes with high maximum WC-IR-drops. With our strategy to prioritize mapping on cores with less

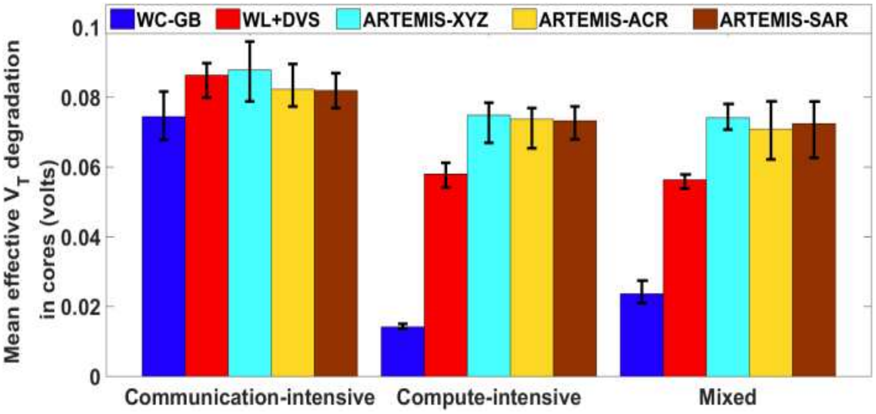
WC-IR-drops, *ARTEMIS* frameworks produce lower maximum %WC-IR-drop-values (by up to 9% lower), compared to WL+DVS, despite *ARTEMIS* having a longer lifetime and servicing a higher number of applications.



(a)

(b)

(c)



(d)

Figure 51: Results showing improvements for our circuit-aging (leakage) and PDN-aging aware region selection and Vdd selection heuristic in the ARTEMIS frameworks: (a) power dissipation per application, (b) maximum %WC-IR-drop at end of lifetime, (c) Variance of %WC-IR-drop at end of lifetime, (d) Mean effective V_T-degradation in compute-cores at end of lifetime

Figure 51(c) shows the variance in the WC-IR-drop-distribution on the 3D chip obtained at the end of lifetime with different frameworks. A smaller variance of IR-drops with *ARTEMIS* frameworks (up to 24% lower compared to WL+DVS) signifies efficient management of PDN-aging that aides in improving the longevity of the PDN.

Figure 51(d), shows the mean effective V_T-degradation in compute-cores at the end of lifetime, and provides additional insights into the lifetime improvements obtained with our circuit-aging (leakage) and PDN-aging aware region selection and V_{dd} selection heuristic. As discussed earlier, the V_T-values of circuit components increase with aging. Given the nominal-V_T of 0.3V at the start of lifetime, observe in figure 51(d) that the mean V_T-degradation values at the end of lifetime for *ARTEMIS* frameworks are significantly higher (by up to 30% for compute-intensive workloads) compared to the WL+DVS framework. By restricting the EM-induced PDN-degradation, *ARTEMIS* can extend the tolerable degree of circuit-aging (V_T-degradation) in compute-cores, while meeting the same performance constraints. Thus the 3D CMP remains functional for much higher V_T-degradation with *ARTEMIS* compared to other approaches.

8.7 Conclusion

In this chapter, we proposed an aging-aware application-mapping and DVS scheduling framework (*ARTEMIS*) that considers PDN-aging of 3D NoC-based CMPs in addition to circuit-aging (in NoC and cores) in both the performance and aging evaluation stages, and the lifetime optimization methodology. Compared to a framework based on the best known prior works on aging-aware mapping techniques, *ARTEMIS* is able to service **25%** more applications (on average) over the chip lifetime, which highlights its promise for emerging 3D-CMPs.

9 Summary and Future Work

9.1 Research Summary

To mitigate contemporary and future design-challenges, this thesis proposes a set of design-time and run-time frameworks with novel system-level optimization techniques for voltage island-partitioning, voltage-scaling, application-mapping, and routing path allocation for the design of network-on-chip (NoC)-based multicore SoCs.

The first part of the thesis (chapters 2 to 5) contributes with design-time synthesis frameworks for multiprocessor-system-on-chips (MPSoCs) - MPSoCs are a class of multi-processors that are designed to run embedded applications whose computation and communication profiles are known at design-time. The proposed frameworks exhibit awareness of process variations, voltage-islands, communication- and computation-latencies, thermal profiles, voltage-drop profiles, power, and energy-dissipation in the multicore system. Our holistic solution evaluation methodology that accurately captures the impact of increasing temperatures and voltage-drops on system performance and power dissipation, leads to more intelligent exploration of the solution search-space. We show that such a holistic approach for multi-objective optimization is critical to produce feasible synthesis-solutions with better overall optimality. Our frameworks are able to produce significant improvements across different metrics such as design-yield, voltage-drops in the power-delivery network (PDN), power- and energy-dissipation, NoC-traffic, and system energy-delay-squared product (ED^{2}P). The solutions generated are also more amenable to efficient physical design because of considering PDN design issues and the impact of process variations, voltage-drops, and temperature on system power and performance early in the design flow.

The second part of the thesis (chapters 6, 7, and 8) contributes with novel run-time adaptations for chip multiprocessors (CMPs) – CMPs are a class of multi-processors built for general purpose computing. Our frameworks integrate lifetime- and soft-error-reliability, and variation-awareness in a run-time variable degree-of-parallelism (DoP) application-scheduling methodology to enhance CMP-performance in the new dark-silicon era. We consider PDN-aging of 3D NoC-based CMPs in addition to circuit-aging (in NoC and cores) in both the performance and aging evaluation stages, and the lifetime optimization methodology. Tailored for deeply scaled technologies, our proposed frameworks generate highly optimized application-mapping solutions, while exhibiting linear run-time complexity.

9.2 Future Work

In modern power-constrained designs with increasing levels of dark-silicon, operating at lower voltages and frequencies could potentially be quite beneficial. Firstly, given a fixed power budget, power savings would translate to higher available power-slacks on the CMP-die thereby possibly yielding higher performance by either simultaneously executing additional applications or running applications at higher app-DoPs. Secondly, low power techniques could potentially retard the rate of aging on the die thereby extending useful lifetime of the chip.

At the same time, the soft-error rate (SER) exponentially increases as we attempt to reduce the rate of circuit-aging with DVFS. Higher SER as well as higher app-DoP increases the probability of occurrence of one or more errors during the execution of an application (decreases the application reliability). Even with sophisticated re-execution techniques using check-pointing and rollback, occurrence of soft-errors could potentially incur significant overheads in terms of application runtimes, power dissipation and application aging foot-print - the very metrics that were expected to improve by employing low-power DVFS techniques.

Additionally, as discussed in chapters 6 and 7, varying the degree of parallelism (DoP) of applications at runtime to adapt to the execution environment of the CMP could potentially produce significant benefits in terms of both application service-times and energy dissipation. Besides application run-times, application-DoPs (app-DoPs) also significantly influence application-soft-error reliability, aging foot-print, and chip power budget. Moreover, note that each run-time aging profile produced during the lifetime of the CMP-die corresponds to unique power/performance profile as well; therefore, different task-to-core mappings correspond to unique values of maximum operating frequency, power dissipation, and aging foot-print of the application.

In our future work, we bring to light the intricate interdependence of the above mentioned seemingly unrelated optimization metrics (power, performance, and reliability), design-knobs (voltage, frequency, app-DoP, and task-to-core mapping) and their effects on physical phenomena (soft-errors and circuit-aging). Also, we will propose a novel run-time soft-error reliability- and lifetime-reliability-aware application mapping + DVFS framework that employs dynamically adaptable application degrees of parallelism (DoPs) to maximize the number of applications serviced over the target lifetime (in years) of a CMP that meet their completion deadlines, while meeting a chip-wide dark-silicon power constraint (DS-Pc) and application performance (minimum frequency) constraints. For the first time, this work will consider the combined effects of runtime management techniques on lifetime-reliability and soft-error reliability of the CMP, while integrating an error check-pointing and rollback scheme for applications to recover from runtime soft-errors.

References

[1] http://betanews.com/2013/10/15/breaking-moores-law/ - last referenced: Apr. 21, 2015.

[2] Intel FAER series lecture, "Introduction to Multi-Core", http://www.faer.ac.in/treach/pdf/Introduction_to_Multi_Core.pdf - last referenced: Apr. 20, 2015.

[3] T. G. Mattson, R. F. Van der Wijngaart, M. Riepen, T. Lehnig, P. Brett, W. Haas, P. Kennedy, J. Howard, S. Vangal, N. Borkar, G. Ruhl, S. Dighe, "The 48-core SCC processor: the programmer's view," *ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1-11, Nov. 2010.

[4] S. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, P. Iyer, A. Singh, T. Jacob, S. Jain, S. Venkataraman, Y. Hoskote, N. Borkar, "An 80-Tile 1.28 TFLOPS Network-on-Chip in 65nm CMOS," *IEEE International Solid-State Circuits Conference*, pp. 98– 589, Feb. 2007.

[5] Tilera Corporation. TILE64^{TM} Processor. Product Brief. 2007.

[6] T. Okumura, F. Minami, K. Shimazaki, K. Kuwada, M. Hashimoto, "Gate delay estimation in STA under dynamic power supply noise," *IEEE Asia and South Pacific Design Automation Conference*, pp. 775–780, Jan. 2010,

[7] N. H. Khan, S. M. Alam, S. Hassoun, "Power delivery design for 3-D ICs using different through-silicon via (TSV) technologies," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 19, no. 4, pp. 647–658, Apr. 2011.

[8] A. Kaouache, F. Wrobel, F. Saigne, A. D. Touboul, R. D. Schrimpf, "Analytical method to evaluate soft error rate due to alpha contamination," *IEEE Trans. Nucl. Sci.*, vol. 60, no. 6, pp. 4059-4066, Dec. 2013.

[9] N. Gaspard, S. Jagannathan, Z. Diggins, A. V. Kauppila, T. D. Loveless, J. S. Kauppila, B. L. Bhuva, L. W. Massengill, W. T. Holman, A. S. Oates, Y-P. Fang, S-J. Wen, R. Wong, "Effect of

threshold voltage implants on single-event error rates of D flip-flops in 28-nm bulk CMOS*,"* *IEEE International Reliability Physics Symposium*, pp. SE.7.1–SE.7.3, Apr. 2013.

[10] S. Abe, Y. Watanabe, N. Shibano, N. Sano, H. Furuta, M. Tsutsui, T. Uemura, T. Arakawa, "Neutron-induced soft error analysis in MOSFETs from a 65 to a 25 nm design rule using multi-scale Monte-carlo simulation method," *IEEE International Reliability Physics Symposium*, pp.SE.3.1-SE.3.6, Apr. 2012.

[11] D. Zhu, R. Melhem, D. Mosse, "The effects of energy management on reliability in real-time embedded systems," *IEEE/ACM International Conference on Computer Aided Design*, pp. 35-40, Nov. 2004.

[12] X. Huang, T. Yu, V. Sukharev, S. X.-D. Tan, "Physics-based electromigration assessment for power grid networks," *IEEE/ACM Design Automation Conference*, pp. 1-6, June 2014.

[13] S. S. Sapatnekar, "What happens when circuits grow old: Aging issues in CMOS design," *International Symposium on VLSI Technology, Systems, and Applications*, pp. 1-2, Apr. 2013.

[14] J. Lee, N. Sung Kim, "Optimizing total power of many-core processors considering voltage scaling limit and process variations," *ACM/IEEE International Symposium on Low Power Electronics and Design*, pp. 201-206, July 2009.

[15] S. Borkar, "Design perspectives on 22nm CMOS and beyond," *IEEE/ACM Design Automation Conference*, pp. 93-94, July 2009.

[16] J. Allred, S. Roy, K. Chakraborty, "Designing for dark silicon: a methodical perspective on energy efficient systems," *ACM/IEEE International Symposium on Low Power Electronics and Design*, pp. 255-260, July 2012.

[17] B. Raghunathan, Y. Turakhia, S. Garg, D. Marculescu, "Cherry-picking: Exploiting process variations in dark-silicon homogeneous chip multi-processors," *Design Automation and Test in Europe Conference*, pp. 39-44, Mar. 2013.

[18] L. Benini, G. De-Micheli, "Networks on Chip: A New SoC Paradigm," *IEEE Computer*, vol. 49, no. 1, pp. 70-78, Jan. 2002.

[19] J. M. Rabaey, Digital Integrated Circuits. Prentice Hall, 1996.

[20] F. Fallah, M. Pedram, "Standby and active leakage current control and minimization in CMOS VLSI circuits," *IEICE Trans. on Electronics*, vol. 88, no. 4, pp. 509–519, Apr. 2005.

[21] T. Chelcea, S. Nowick, "A low latency FIFO for mixed-clock system," *IEEE Workshop on VLSI*, pp. 119-126, Apr. 2002.

[22] P. Choudhary, D. Marculescu, "Hardware based frequency/ voltage control of voltage frequency island systems," *International Conference on Hardware/Software Codesign and System Synthesis*, pp. 34-39, Oct. 2006.

[23] D. Lackey, P. S. Zunchowski, T. R. Bednar, D. W. Stout, S. W. Gould**,** J. M. Cohn, "Managing Power and Performance for System-on-Chip Designs using Voltage Islands," *IEEE/ACM International Conference on Computer Aided Design*, pp. 195-202, Nov. 2002.

[24] W. Jang, D. Ding, D. Pan, "A Voltage-Frequency Island Aware Energy Optimization Framework for Networks-on-Chip," *IEEE/ACM International Conference on Computer Aided Design*, pp. 264 – 269, Nov. 2008.

[25] S. Murali, P. Meloni, F. Angiolini, D. Atienzat, S. Carta, L. Benini, G. De-Micheli, L. Raffo, "Designing Application-Specific Networks on Chips with Floorplan Information," *IEEE/ACM International Conference on Computer Aided Design*, pp. 355-362, Nov. 2006.

[26] C. Seiculescu, S. Murali, L. Benini, G. De-Micheli, "NoC Topology Synthesis for Supporting Shutdown of Voltage Islands in SoCs," IEEE/ACM *Design Automation Conference*, pp. 822-825, July 2009.

[27] P. Zhou, P. Yuh, S. Sapatnekar, "Application-Specific 3D Network-on-Chip Design Using Simulated Allocation," *IEEE Asia and South Pacific Design Automation Conference*, pp. 517-522, Jan. 2010.

[28] K. Srinivasan, K. Chatha, "A low complexity heuristic for design of custom network-on-chip architectures," *Design Automation and Test in Europe*, pp. 130-135, March 2006.

[29] K. Srinivasan, K. Chatha, G. Konjevod, “Linear-Programming-Based Techniques for Synthesis of Network-on_Chip Architectures,” *IEEE Trans. on VLSI systems*, vol. 14, no. 4, pp. 407-420, Apr. 2006.

[30] U. Ogras, R. Marculescu, “It’s a Small World After All”: NoC Performance Optimization Via Long-Range Link Insertion,” *IEEE Trans. on VLSI systems (TVLSI)*, vol. 14, no. 7, pp. 693-706, July 2006.

[31] L. Leung, C. Tsui, “Energy-Aware Synthesis of Networks-on-Chip Implemented with Voltage Islands,” *Design Automation Conference*, pp. 128-131, June 2007.

[32] P. Ghosh, A. Sen, “Efficient Mapping and Voltage Islanding Technique for Energy Minimization in NoC under Design Constraints,” *Symposium on Applied Computing*, pp. 535–541, Mar. 2010.

[33] U. Ogras, R. Marculescu, P. Choudhary, D. Marculescu, “Voltage-Frequency Island Partitioning for GALS-based Networks-on-Chip”, *Design Automation Conference*, pp. 110-115, June 2007.

[34] J. Hu, R. Marculescu, “Communication and task scheduling of application-specific networks-on-chip,” *IEE Computers and Digital Techniques*, vol. 152, no. 5, pp. 643-651, Sep. 2005.

[35] C. Chou, U. Ogras, R. Marculescu, “Energy and Performance-Aware Incremental Mapping for Networks on Chip with Multiple Voltage Levels,” *IEEE Transactions on CAD (TCAD)*, vol. 27, no. 10, pp. 1866–1879, Oct. 2008.

[36] J. Hu, R. Marculescu, “Energy and Performance-Aware Mapping for Regular NoC Architectures,” *IEEE Transactions CAD*, vol. 24, no. 4, pp. 551-562, Apr. 2005.

[37] http://www.arm.com/products/processors/selector.php - last referenced: April 12, 2015.

[38] http://ziyang.eecs.umich.edu/~dickrp/tgff/ - last referenced: April 12, 2015.

[39] S. Murali, M. Coenen, A. Radulescu, K. Goossens, G. De-Micheli, “Mapping and configuration methods for multi-use-case networks on chips,” *IEEE Asia and South Pacific Design Automation Conference*, pp. 146-151, Jan. 2006.

[40] A. Kahng, B. Li, L.-S. Peh, K. Samadi, "ORION 2.0: A Fast and Accurate NoC Power and Area Model for Early-Stage Design Space Exploration," *Design Automation and Test in Europe*, pp. 423–428, Apr. 2009.

[41] T. Chelcea, S. Nowick, "A Low-Latency for Mixed-Clock Systems," *IEEE Computer Society Workshop on VLSI*, pp. 119–126, Apr. 2000.

[42] W. Dally, B. Towles, Principles and Practices of Interconnection Networks. Morgan Kauffman, 2005.

[43] F. Zanini, M. Sabry, D. Atienza, G. De-Micheli, "Hierarchical thermal management policy for high-performance 3D systems with liquid cooling," *IEEE Journal on Emerging and Selected Topics on Circuits and Systems*, vol. 1, no. 2, pp. 88-101, June 2011.

[44] A. Sridhar, A. Vincenzi, M. Ruggiero, T. Brunschwiler, D. Atienza, "3D-ICE: Fast compact transient thermal modeling for 3D-ICs with inter-tier liquid cooling," *IEEE/ACM International Conference on Computer Aided Design*, pp. 463-470, Nov. 2010.

[45] H. Matsutani, M. Koibuchi, I. Fujiwara, T. Kagami, Y. Take, T. Kuroda, P. Bogdan, R. Marculescu, H. Amano, "Low-latency wireless 3D NoCs via randomized shortcut chips," *Design Automation and Test in Europe*, pp. 1-6, Mar. 2014.

[46] B. Shi, A. Srivastava, "Optimized micro-channel design for stacked 3-D-ICs," *IEEE Trans. Computer Aided Design*, vol. 33, no. 1, pp. 90-100, Jan. 2014.

[47] N. Barrow-Williams, C. Fensch, S. Moore, "A communication characterization of Splash-2 and Parsec," *IEEE International Symposium on Workload Characterization*, pp. 86-97, Oct. 2009.

[48] P. M. Chaudhari, R. V. Dharaskar, V. M. Thakare, "Computing the most significant solution from Pareto front obtained in multi-objective evolutionary," *International Journal of Advanced Computer Science and Applications*, vol. 1, no. 4, pp. 63-68, Oct. 2010.

[49] P. Falkenstern, Y. Xie, Y. Chang, Y. Wang, "Three-dimensional integrated circuits (3D IC) floorplan and power/ground network co-synthesis," *IEEE Asia and South Pacific Design Automation Conference*, pp. 169-174, Jan. 2010.

[50] H. Chen, H. Lin, Z. Wang, T. Hwang, “A new architecture for power network in 3D IC,” *Design Automation and Test in Europe*, pp. 401-406, Mar. 2011.

[51] K. Ghosh, J. Zhang, L. Zhang, Y. Dong, H.Y. Li, C.M. Tan, G. Xia, C.S. Tan, “Strategy for TSV scaling with consideration on thermo-mechanical stress and acceptable delay,” *International Microsystems, Packaging, Assembly and Circuits Technology Conference*, pp. 49-51, Oct. 2012.

[52] W. Huang, K. Rajamani, M.R. Stan, K. Skadron, “Scaling with design constraints: predicting the future of big chips,” *IEEE Micro*, vol. 31, no. 4, pp. 16-29, Aug. 2011.

[53] R. Apperson, Z. Yu, M.J. Meeuwsen, T. Mohsenin, B. Baas, “A scalable dual-clock FIFO for data transfers between arbitrary and haltable clock domains,” *IEEE Trans.VLSI*, vol. 15, no. 10, pp. 1125-1134, Oct. 2007.

[54] Nirgam simulator, http://nirgam.ecs.soton.ac.uk/ - last referenced: April 18, 2015.

[55] N. Kapadia, S. Pasricha, “VISION: A framework for voltage island aware synthesis of interconnection networks-on-chip,” *ACM Great Lakes Symposium on VLSI*, pp. 31-36, May 2011.

[56] C. Bienia, S. Kumar, J.P. Singh, K. Li, “The PARSEC benchmark suite: characterization and architectural implications,” *ACM International Conference on Parallel Architectures and Compilation Techniques*, pp. 72-81, Oct. 2008.

[57] A. Ganguly, K. Chang, S. Deb, P.P. Pande, B. Belzer, C. Teuscher, “Scalable hybrid wireless network-on-chip architectures for multicore systems,” *IEEE Trans. on Computers*, vol. 60, no. 10, pp.1485-1502, Oct. 2011.

[58] S.V. Woo, M. Ohara, E. Torrie, J.P. Singh, A. Gupta, “The SPLASH-2 programs: characterization and methodological characterization,” *IEEE/ACM International Symposium on Computer Architecture*, pp. 24-36, May 1995.

[59] M.E.S. Elrabaa, “A new FIFO for transferring data between two unrelated clock domains,” *International Journal of Electronics*, vol. 99, no. 8, pp. 1-12, Aug. 2012.

[60] P. Zhao, J. B. McNeely, P. K. Golconda, S. Venigalla, N. Wang, M. A. Bayoumi, W. Kuang, L. Downey, “Low-power clocked-pseudo-NMOS flip-flop for level conversion in dual supply systems,” *IEEE Trans. on VLSI Systems*, vol. 17, no. 9, pp. 1196-1202, Sept. 2009.

[61] C. Alpert, D. Mehta, S. Sapatnekar, “Handbook of algorithms for physical design automation,” CRC Press, 2009.

[62] C. Seiculescu, S. Murali, L. Benini, G. De Micheli, “SunFloor 3D: A tool for networks on chip topology synthesis for 3D systems on chips,” *Design Automation and Test in Europe*, pp. 9-14, Apr. 2009.

[63] D. Shin, S.W. Chung, E. Chung, N. Chang, “Energy-optimal dynamic thermal management: computation and cooling power co-optimization,” *IEEE Trans. on Industrial Informatics*, vol. 6, no. 3, pp. 340-351, Aug. 2010.

[64] X. Wei, G. Goth, P. Kelly, R. Zoodsma, A. VanDeventer, “Air-water hybrid cooling for computer servers: a case study for optimum cooling energy allocation,” *IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems*, pp. 568-573, May 2014.

[65] J. Chang, H.S. Park, J.I. Jo, S. Julia, “A system design of liquid cooling computer based on the micro cooling technology,” *IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems*, pp. 157-160, May 2006.

[66] S. Pasricha, “A framework for TSV serialization-aware synthesis of application specific 3D networks-on-chip”, *IEEE VLSI Design Conference*, pp. 268-273, Jan. 2012.

[67] N. Kapadia, S. Pasricha, “A co-synthesis methodology for power delivery and data interconnection networks in 3D ICs,” *IEEE International Symposium on Quality Electronic Design*, pp. 73-79, Mar. 2013.

[68] W. Lee, D. Marculescu, Y. Chang, “Post-floorplanning power/ground ring synthesis for multiple-supply-voltage designs,” *ACM International Symposium on Physical Design*, pp. 5-12, Apr. 2009.

[69] Q. Zhou, J. Shi, B. Liu, Y. Cai, “Floorplanning considering IR drop in multiple supply voltages island designs,” *IEEE Trans. on VLSI Systems*, vol. 19, no. 4, pp. 638-646, Apr. 2011.

[70] S. Kose, E. Friedman, “Fast algorithms for power grid analysis based on effective resistance,” *IEEE International Symposium on Circuits and Systems*, pp. 3661-3664, May 2010.

[71] 3D-ICE open-source tool: http://esl.epfl.ch/3d-ice.html - last referenced: April 20, 2015.

[72] A. Coskun, J. Meng, D. Atienza, M.M. Sabry, “Attaining single-chip, high-performance computing through 3D systems with active cooling,” *IEEE Micro*, vol. 31, no. 4, pp. 63-75, July-Aug. 2011.

[73] D. Starobinksi, M. Karpovsky, L. Zakrevski, “Application of network calculus to general topologies using turn-prohibition,” *ACM/IEEE Trans. on Networking*, vol. 11, no. 3, pp. 411-421, June 2003.

[74] M.M. Sabry, A.K. Coskun, D. Atienza, T.S. Rosing, T. Brunschwiler, “Energy-efficient multiobjective thermal control for liquid-cooled 3-D stacked architectures,” *IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems*, vol. 30, no. 12, pp. 1883-1896, Dec. 2011.

[75] lp_solve 5.5.2.0, http://lpsolve.sourceforge.net/5.5/ - last referenced: April 20, 2015.

[76] M. Janicki, J. H. Collet, A. Louri, A. Napieralski, “Hot spots and core-to-core thermal coupling in future multi-core architectures,” *IEEE Semiconductor Thermal Measurement and Management Symposium*, pp. 205-210, Feb. 2010.

[77] S. Ganapathy, R. Canal, A. Gonzalez, A. Rubio, “Circuit propagation delay estimation through multivariate regression-based modeling under spatio-temporal variability,” *Design Automation and Test in Europe Conference*, pp. 417-422, Mar. 2010.

[78] A. T. Winther, W. Liu, A. Nannarelli, S. Vrudhula, “Temperature dependent wire delay estimation in floorplanning,” *NORCHIP*, pp. 1-4, Nov. 2011.

[79] H. Su, F. Liu, A. Devgan, E. Ecar, S. Nassif, “Full chip leakage-estimation considering power supply and temperature variations,” *ACM/IEEE International Symposium on Low Power Electronics and Design*, pp. 78-83, Aug. 2003.

[80] M. Pedram, S. Nazarian, “Thermal Modeling, Analysis, and Management in VLSI Circuits: Principles and Methods,” *Proc. of the IEEE*, vol. 94, no. 8, pp. 1487-1501, Aug. 2008.

[81] W. Liao, L. He, K. M. Lepak, “Temperature and supply voltage aware performance and power modeling at microarchitecture level,” *IEEE Trans. on Computer Aided Design of Circuits and Systems*, vol. 24, no. 7, pp. 1042-1053, July 2005.

[82] S. Herbert, S. Garg, D. Marculescu, “Exploiting process variability in voltage/frequency control,” IEEE Trans. on VLSI Systems, vol. 20, no. 8, pp. 1392-1404, Aug. 2012.

[83] A. Coskun, T.S. Rosing, J. Ayala, D. Atienza, “Modeling and dynamic management of 3D multicore systems with liquid cooling,” *IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, pp. 35-40, Oct. 2009.

[84] M. Arjomand, H. Sarbazi-Azad, “Voltage-frequency planning for thermal-aware, low-power design of regular 3-D NoCs,” *IEEE International conference on VLSI Design*, pp. 57-62, Jan. 2010.

[85] A. J. Martin, “Towards an energy complexity of computation,” *Information Processing Letters*, vol. 77, no. (2-4), pp. 181-187, Feb. 2001.

[86] E. Humenay, D. Tarjan, K. Skadron, “Impact of process variations on multicore performance symmetry,” *Design Automation and Test in Europe Conference*, pp. 1653-1658, Apr. 2007.

[87] L. Pang, K. Qian, C. J. Spanos, B. Nikolic, “Measurement and Analysis of Variability in 45 nm Strained-Si CMOS Technology,” *IEEE Journal of Solid –State Circuits*, vol. 44, no. 8, pp. 2233-2243, Aug. 2009.

[88] S. Reda, S. Nassif, “Analyzing the impact of process variations on parametric measurements: novel models and applications,” *Design Automation and Test in Europe Conference*, pp. 375-380, Apr. 2009.

[89] S. Garg, D. Marculescu, “System-level process variation driven throughput analysis for single and multiple voltage-frequency island designs,” *Design Automation and Test in Europe Conference*, pp. 1-6, Apr. 2007.

[90] N. Kapadia, S. Pasricha, “A Framework for Low Power Synthesis of Interconnection Networks-on-Chip with Multiple Voltage Island,” *Integration, the VLSI Journal*, vol. 45, no. 3, pp. 271-281, June 2012.

[91] S. Majzoub, R. Saleh, S. Wilton, R. Ward, “Energy Optimization for Many-Core Platforms: Communication and PVT Aware Voltage-Island Formation and Voltage Selection Algorithm”, *IEEE Trans. Computer-Aided Design of Circuits and Systems*, vol. 29, no. 5, pp. 816-829, May 2010.

[92] N. Kapadia, S. Pasricha, “VERVE: A Framework for Variation-Aware Energy Efficient Synthesis of NoC-based MPSoCs with Voltage islands,” *IEEE International Symposium on Quality Electronic Design*, pp. 603-610, Mar. 2013.

[93] F. Wang, Y. Chen, C. Nicopoulos, X. Wu, Y. Xie, N. Vijaykrishnan, “Variation-Aware Task and Communication Mapping for MPSoC Architecture,” *IEEE Trans. on Computer Aided Design of Circuits and Systems*, vol. 30, no. 2, pp. 295-307, Feb. 2011.

[94] D. Mirzoyan, B. Akesson, K. Goossens, “Process-variation aware mapping of real-time streaming applications to MPSoCs for improved yield,” *IEEE International Symposium on Quality Electronic Design*, pp. 41-48, Mar. 2012.

[95] L. Huang, Q. Xu, “Performance Yield-driven Task Allocation and Scheduling for MPSoCs under Process Variation,” *IEEE/ACM Design Automation Conference*, pp. 326-331, June 2010.

[96] K. Bhardwaj, S. Roy, K. Chakraborty, “Power-Performance Yield Optimization for MPSoCs Using MILP,” *IEEE International Symposium on Quality Electronic Design*, pp. 764-771, Mar. 2012.

[97] VARIUS model, http://web.cse.ohio-state.edu/~teodores/arch/tools/varius/varius.html - last referenced: April 20, 2015.

[98] S. Sarangi, B. Greskamp, R. Teodorescu, J. Nakano, A. Tiwari, J. Torrellas, “VARIUS: A Model of Process Variation and Resulting Timing Errors for Microarchitects,” *IEEE Trans. on Semiconductor Manufacturing*, vol. 21, no. 1, pp. 3-13, Feb. 2008.

[99] B. Li, L. Peh, P. Patra, “Impact of process and temperature variation on network-on-chip design exploration,” *IEEE/ACM International Symposium on Networks-on-chip*, pp. 117-126, Apr. 2008.

[100] B. Amelifard, M. Pedram, “Optimal design of the power delivery network for multiple voltage-island system-on-chips,” *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 6, pp. 888–900, May 2009.

[101] S. Pasricha, Y. Zou, “A low overhead fault tolerant routing scheme for 3D Networks-on-Chip,” *IEEE International Symposium on Quality Electronic Design*, pp. 1-8, Mar. 2011.

[102] K. Bhardwaj, K. Chakraborty, S. Roy, “Towards graceful aging degradation in NoCs through an adaptive routing algorithm,” *ACM/IEEE Design Automation Conference*, pp. 382-391, June 2012.

[103] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, N. P. Jouppi, “McPAT: An integrated power, area, and timing modeling framework for multicore and many-core architectures,” *IEEE/ACM International Symposium on Microarchitecture*, pp. 469-480, Dec. 2009.

[104] A. Das, A. Kumar, B. Veeravalli, C. Bolchini, A. Miele, “Combined DVFS and mapping exploration for lifetime and soft-error susceptibility improvement in MPSoCs,” *Design Automation and Test Europe Conference*, pp. 1-6, Mar. 2014.

[105] A. Bashir, J. Li, K. Ivatury, N. Khan, N. Gala, N. Familia, Z. Mohammed, “Fast lock scheme for phase-locked loops,” *IEEE Custom Integrated Cicuits Conference*, pp. 319–322., Sep. 2009.

[106] M. Haque, H. Aydin, D. Zhu, “Energy-aware task replication to manage realiability for periodic real-time applications on multicore platforms,” *International Green Computing Conference*, pp. 1-11, June 2013.

[107] T. D. Loveless, S. Jagannathan, T. Reece, J. Chetia, B. L. Bhuva, M. W. McCurdy, L. W. Massengill, S. J. Wen, R. Wong, D. Rennie, “Neutron- and proton-induced single event upsets for D- and DICE-flip/flop designs at a 40 nm technology node,” *IEEE Trans. on Nuclear Science*, vol. 58, no. 3, pp. 1008-1014, June 2011.

[108] J. Li, J. F. Martinez, “Dynamic power-performance adaptation of parallel computation on chip multiprocessors,” *International Symposium on High Performance Computer Architecture*, pp. 77-87, Feb. 2006.

[109] Y. Ding, M. Kandemir, P. Raghavan, M. J. Irwin, “A helper thread based EDP reduction scheme for adapting application execution in CMPs,” *IEEE International Symposium on Parallel and Distributed Processing*, pp. 1-14, Apr. 2008.

[110] Y. Turakhia, B. Raghunathan, S. Garg, D. Marculescu, “HADES: Architectural synthesis for heterogeneous dark silicon chip multi-processors,” *ACM/IEEE Design Automation Conference*, pp. 1-7, June 2013.

[111] A. Raman, H. Kim, T. Oh, J. W. Lee, D. I. August, “Parallelism orchestration using DoPE: the degree of parallelism executive,” ACM *Conference on Programming Language Design and Implementation*, pp. 26-37, June 2011.

[112] L. Zhang, L. S. Bai, R. P. Dick, L. Shang, R. Joseph, “Process variation characterization of chip-level multiprocessors,” *ACM/IEEE Design Automation Conference*, pp.694-697, July 2009.

[113] X. Wang, M. Tehranipoor, S. George, D. Tran, L. Winemberg, “Design and analysis of a delay sensor applicable to process/environmental variations and aging measurements,” *IEEE Trans. on VLSI Systems*, vol. 20, no. 8, pp. 1405-1418, Aug. 2012.

[114] A. A. M. Bsoul, N. Manjikian, L. Shang, “Reliability- and process variation-aware placement for FPGAs,” *Design Automation and Test in Europe Conference*, pp. 1809-1814, Apr. 2010.

[115] S. Dighe, S. R. Vangal, P. Aseron, S. Kumar, T. Jacob, K. A. Bowman, J. Howard, J. Tschanz, V. Erraguntla, N. Borkar, V. K. De, S. Borkar, “Within-die variation-aware dynamic-voltage-frequency-scaling with optimal core allocation and thread hopping for the 80-core TeraFLOPS processor,” *IEEE Journal of Solid-state Circuits*, vol. 46, no. 1, pp. 184-193, Jan. 2011.

[116] M. Fattah, M. Daneshtalab, P. Liljeberg, J. Plosila, “Smart hill climbing for agile dynamic mapping in many-core systems,” *IEEE/ACM Design Automation Conference*, pp. 1-6, June 2013.

[117] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, D. A. Wood, "The gem5 simulator," *ACM SIGARCH Computer Architecture*, vol. 39, no. 2, pp. 1-7, May 2011.

[118] V. B. Kleeberger, M. Barke, Christoph Werner, D. Schmitt-Landsiedel, U. Schlichtmann, "A compact model for NBTI degradation and recovery under use-profile variations and its application to aging analysis of digital integrated circuits," *Microelectronics Reliability*, vol. 54, no. 6, pp. 1083-1089, July 2014.

[119] D. Bergstrom, M. Hattendorf, J. Hicks, J. Jopling, J. Maiz, S. Pae, C. Prasad, J. Wiedemer, "Intel's 45 nm CMOS technology," *Intel Technology Journal*, vol. 12, no. 2, pp. 77-156, June 2008.

[120] E. Mintarno, J. Skaf, R. Zheng, J. B. Velamala, Y. Cao, S. Boyd, R. W. Dutton, S. Mitra, "Self-tuning for maximized lifetime energy-efficiency in the presence of circuit aging," *IEEE Trans. on Computer Aided Design of Circuits and Systems*, vol. 30, no. 5, pp. 760-773, May 2011.

[121] N. Khan, S. M. Alam, H. Hassoun, "System-level comparison of power delivery design for 2D and 3D ICs," *IEEE International Conference on 3D System Integration*, pp. 1-7, Sept. 2009.

[122] W. Chan, A. B. Kahng, S. Nath, "Methodology for electromigration signoff in the presence of adaptive voltage scaling," *ACM/IEEE International Workshop on System level Interconnect Prediction*, pp. 1-7, June 2014.

[123] T. E. Carlson, W. Heirman, L. Eeckhout, "Sniper: Exploring the level of abstraction for scalable and accurate parallel multi-core simulation," *ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1-12, Nov. 2011.

[124] A. Tiwari, J. Torrellas, "Facelift: Hiding and slowing down aging in multicores," *IEEE/ACM International Symposium on Microarchitecture*, pp. 129-140, Nov. 2008.

[125] S. Feng, S. Gupta, A. Ansari, S. Mahlke, "Maestro: Orchestrating lifetime reliability in chip multiprocessors," *International Conference on High Performance Embedded Architectures and Compilers*, pp. 186-200, Jan. 2010.

[126] F. Paterna, A. Acquaviva, L. Benini, “Aging-aware energy-efficient workload allocation for mobile multimedia platforms,” *IEEE Trans. on Parallel and Distributed Systems*, vol. 24, no. 8, pp. 1489-1499, Aug. 2013.

[127] P. Mercati, A. Bartolini, F. Paterna, T. S. Rosing, L. Benini, “Workload and user experience-aware dynamic reliability management in multicore processors,” *IEEE/ACM Design Automation Conference*, pp. 1-6, May 2013.

[128] J. Velamala, K. Sutaria, H. Shimizu, H. Awano, T. Sato, Y. Cao, “Statistical aging under dynamic voltage scaling: a logarithmic model approach,” *IEEE Custom Integrated Circuits Conference*, pp. 1-4, Sept. 2012.

[129] J. Velamala, K. Sutaria, T. Sato, Y. Cao, “Physics matters: statistical aging prediction under trapping/detrapping,” *IEEE/ACM Design Automation Conference*, pp. 139-144, June 2012.

[130] J. Velamala, K. B. Sutaria, H. Shimizu, H. Awano, T. Sato, G. Wirth, Y. Cao, “Compact modeling of statistical BTI under trapping/detrapping,” *IEEE Trans. Electron Devices*, vol. 60, no. 11, pp. 3645-3654. Nov. 2013.

[131] V. Mishra, S. Sapatnekar, “The impact of electromigration in copper interconnects on power grid integrity,” *IEEE/ACM Design Automation Conference*, pp. 1-6, May 2013.

[132] S. Chen, M. Chang, W. Hsieh, W. Hwang, “Fully on-chip temperature, process, and voltage sensors,” *IEEE International Symposium on Circuits and Systems*, pp. 897-900, June 2010.

[133] P. Singh, E. Karl, D. Sylvester, D. Blaauw, “Dynamic NBTI management using a 45nm multi-degradation sensor,” *IEEE Custom Integrated Cicuits Conference*, pp. 1-4, Sept. 2010.

[134] S. Alessandro, M. Petricca, M. Poncino, E. Macci, “A fully standard-cell delay measurement circuit for timing variability detection,” *International Workshop on Power and Timing Modeling, Optimization and Simulation*, pp. 239-242, Sept. 2013.

[135] P. Jain, T.-H. Kim, J. Keane, C. H. Kim, “A multi-story power delivery technique for 3D integrated circuits,” *ACM/IEEE International Symposium on Low Power Electronics and Design*, pp. 57–62, Aug. 2008.

[136] N. Kapadia, S. Pasricha, “Process variation aware synthesis of application-specific MPSoCs to maximize yield”, IEEE Intl. Conf. on VLSI Design (VLSID), pp: 270-275, Jan. 2014.

[137] S. Kang, H. Yang, S. Kim, I. Bacivarov, S. Ha, L. Thiele, “Static mapping of mixed-critical applications for fault-tolerant MPSoCs,” ACM/EDAC/IEEE Design Automation Conference (DAC), pp: 1-6, June 2014.

[138] M. Salehi, M. Shafique, F. Kriebel, S. Rehman, M. K. Tavana, A. Ejlali, J. Henkel, “dsReliM: Power-constrained reliability management in dark-silicon many-core chips under process variations,” ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES), pp: 75-82, Oct. 2015.

[139] N. Kapadia, S. Pasricha, “VARSHA: Variation and reliability-aware application scheduling with adaptive parallelism in the dark-silicon era,” ACM/IEEE Design, Automation and Test in Europe Conference (DATE), pp: 1060-1065, Mar. 2015.

Appendix I

In this appendix, we present our linear programming (LP) formulation to solve for the grid-point voltages and currents flowing in a 3D regular power grid. We assume a 2D grid of $n \times n$ grid-points supplying to each core on the 3D die (as illustrated in figures 14 and 26). Thus, given T cores on the 3D mesh, the total number of grid-points in the PDN is $T \times n^2$. We also assume that the power pins are located below the bottom tier, i.e., grid-points of just the bottom tier are connected to the power pins, therefore, the total number of external power inputs are $(T \times n^2)/dim_z$ and all vertical PDN branch currents flow in the upward direction.

For a given core-to-tile mapping solution, when the x^{th} core on the core-graph is mapped on to the tile co-ordinates $\{i, j, k\}$ of the 3D mesh, we represent the operating voltage level (v_x) and the maximum current requirement (i_x) of the x^{th} core by $C_{i,j,k}$ *and* $CI_{i,j,k}$ respectively at the co-ordinates $\{i, j, k\}$. Thus, we are given a set of T tile co-ordinates $T_{i,j,k}$, for $0 \leq i \leq dim_x$-1, $0 \leq j \leq dim_y$-1, $0 \leq k \leq dim_z$-1 on a 3D mesh with dimensions $\{dim_x, dim_y, dim_z\}$, with the respective $C_{i,j,k}$ *and* $CI_{i,j,k}$ values. Values of horizontal and vertical branch resistances (R_h, R_v) are defined for a uniform grid, where R_v is typically higher due to the added TSV resistance.

The key *design variables* for our LP problem formulation are summarized as follows:

- Horizontal PDN branch currents leaving the grid-points in d $\{pos\_x, neg\_x, pos\_y, neg\_y\}$ direction: $I_{i,j,k,l,m\text{-}d}$, where co-ordinates $\{i, j, k\}$ represent the 3D location of the corresponding tile on the 3D mesh and $\{l, \mathrm{m}\}$ represent the location of the grid-point on the tile;
- Vertical PDN branch currents respectively entering or leaving the grid-points: $I_{i,j,k,l,m}$ or $I_{i,j,k-1,l,m}$
- PDN grid-point voltages: $V_{i,j,k,l,m}$

The goal of the LP formulation is to determine grid-point voltages and currents flowing in the branch resistances of the PDN with multiple 3D grids.

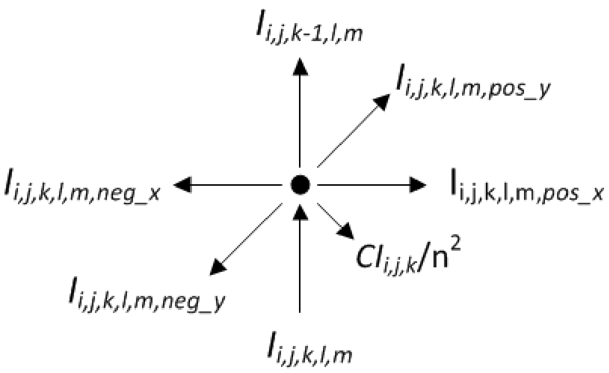


Figure 52: Currents flowing through a PDN grid-point at co-ordinates {i,j,k,l,m}. CIi,j,k/n2 is current supplied to the core. All four horizontal currents are positive when leaving (and negative when entering) the grid-point

The major constraints of our LP formulation are summarized as follows:

Constraint 1: Grid Nodal (Point) Equation (as shown in figure 47) by Kirchoff's Current Law (KCL):

$$I_{i,j,k,l,m} - I_{i,j,k,l,m,pos\_x} - I_{i,j,k,l,m,neg\_x} - I_{i,j,k,l,m,pos\_y} - I_{i,j,k,l,m,neg\_y} - I_{i,j,k-1,l,m} = CI_{i,j,k}/n^2$$

where n^2 is the number of grid-points supplying to each core. Note that the supply current requirement of the each core is assumed to be uniformly distributed over all its n^2 grid-points.

Constraint 2: Grid-point voltages in the bottom-tier ((dim_z -1)th tier) are equal to the rated core supply voltages:

$$V_{i,j,dimz-1,l,m} = C_{i,j,k}$$

Constraint 3: Voltage drops across all vertical branch resistances are defined by the value of R_v used:

$$I_{i,j,k-1,l,m} = \frac{V_{i,j,k,l.m} - V_{i,j,k-1,l,m}}{R_v}$$

Constraint 4: Voltage drops across all horizontal branch resistances (*R*$_h$) corresponding to the four directions are defined with the following equations:

$$I_{i,j,k,l,m-pos\_x} = \frac{V_{i,j,k,l.m} - V_{i+1,j,k,l,m}}{R_h}$$

$$I_{i,j,k,l,m-neg\_x} = \frac{V_{i,j,k,l.m} - V_{i-1,j,k,l,m}}{R_h}$$

$$I_{i,j,k,l,m-pos\_y} = \frac{V_{i,j,k,l.m} - V_{i,j+1,k,l,m}}{R_h}$$

$$I_{i,j,k,l,m-neg\_y} = \frac{V_{i,j,k,l.m} - V_{i,j-1,k,l,m}}{R_h}$$

Constraint 5: If the maximum IR-drop constraint Γ is set to *x*%, i.e., grid-point voltages can be no smaller than (100-*x*)% of the corresponding rated core voltage:

$$V_{i,j,k,l,m} \geq \frac{(100 - x)}{100} * C_{i,j,k}$$

The above constraints are fed to an LP solver [75] to extract exact values of all grid-point voltages and currents in the 3D PDN, which enable the updating of supply voltages of cores, for a given core-to-tile mapping.