Dark Silicon and the End of Multicore Scaling

Hadi Esmaeilzadeh† Emily Blem‡ Renée St. Amant§ Karthikeyan Sankaralingam‡ Doug Burger◇
†University of Washington ‡University of Wisconsin-Madison
§The University of Texas at Austin ◇Microsoft Research
hadianeh@cs.washington.edu blem@cs.wisc.edu stamant@cs.utexas.edu karu@cs.wisc.edu dburger@microsoft.com

ABSTRACT

Since 2005, processor designers have increased core counts to exploit Moore's Law scaling, rather than focusing on single-core performance. The failure of Dennard scaling, to which the shift to multicore parts is partially a response, may soon limit multicore scaling just as single-core scaling has been curtailed. This paper models multicore scaling limits by combining device scaling, single-core scaling, and multicore scaling to measure the speedup potential for a set of parallel workloads for the next five technology generations. For device scaling, we use both the ITRS projections and a set of more conservative device scaling parameters. To model single-core scaling, we combine measurements from over 150 processors to derive Pareto-optimal frontiers for area/performance and power/performance. Finally, to model multicore scaling, we build a detailed performance model of upper-bound performance and lower-bound core power. The multicore designs we study include single-threaded CPU-like and massively threaded GPU-like multicore chip organizations with symmetric, asymmetric, dynamic, and composed topologies. The study shows that regardless of chip organization and topology, multicore scaling is power limited to a degree not widely appreciated by the computing community. Even at 22 nm (just one year from now), 21% of a fixed-size chip must be powered off, and at 8 nm, this number grows to more than 50%. Through 2024, only 7.9× average speedup is possible across commonly used parallel workloads, leaving a nearly 24-fold gap from a target of doubled performance per generation.

Categories and Subject Descriptors: C.0 [Computer Systems Organization] General — Modeling of computer architecture; C.0 [Computer Systems Organization] General — System architectures

General Terms: Design, Measurement, Performance

Keywords: Dark Silicon, Modeling, Power, Technology Scaling, Multicore

1. INTRODUCTION

Moore's Law [23] (the doubling of *transistors* on chip every 18 months) has been a fundamental driver of computing. For the past three decades, through device, circuit, microarchitecture, architecture, and compiler advances, Moore's Law, coupled with Dennard scaling [11], has resulted in commensurate exponential performance increases. The recent shift to multicore designs has aimed to increase the number of cores along with transistor count increases, and continue the proportional scaling of performance. As a result, architecture researchers have started focusing on 100-core and 1000-core chips and related research topics and called for changes to the undergraduate curriculum to solve the parallel programming challenge for multicore designs at these scales.

With the failure of Dennard scaling–and thus slowed supply voltage scaling–core count scaling may be in jeopardy, which would leave the community with no clear scaling path to exploit continued transistor count increases. Since future designs will be power limited, higher core counts must provide performance gains despite the worsening energy and speed scaling of transistors, and given the available parallelism in applications. By studying these characteristics together, it is possible to predict for how many additional technology generations multicore scaling will provide a clear benefit. Since the energy efficiency of devices is not scaling along with integration capacity, and since few applications (even from emerging domains such as recognition, mining, and synthesis [5]) have parallelism levels that can efficiently use a 100-core or 1000-core chip, it is critical to understand how good multicore performance will be in the long term. In 2024, will processors have 32 times the performance of processors from 2008, exploiting five generations of core doubling?

Such a study must consider devices, core microarchitectures, chip organizations, and benchmark characteristics, applying area and power limits at each technology node. This paper considers all those factors together, projecting upper-bound performance achievable through multicore scaling, and measuring the effects of *non-ideal* device scaling, including the percentage of "dark silicon" (transistor under-utilization) on future multicore chips. Additional projections include best core organization, best chip-level topology, and optimal number of cores.

We consider technology scaling projections, single-core design scaling, multicore design choices, actual application behavior, and microarchitectural features together. Previous studies have also analyzed these features in various combinations, but not all together [8, 9, 10, 14, 15, 20, 22, 27, 28]. This study builds and combines three models to project performance and fraction of "dark silicon" on fixed-size and fixed-power chips as listed below:

- Device scaling model (*DevM*): area, frequency, and power requirements at future technology nodes through 2024.
- Core scaling model (*CorM*): power/performance and area/performance single core Pareto frontiers derived from a large set of diverse microprocessor designs.
- Multicore scaling model (*CmpM*): area, power and perfor-

ISCA'11, June 4–8, 2011, San Jose, California, USA.

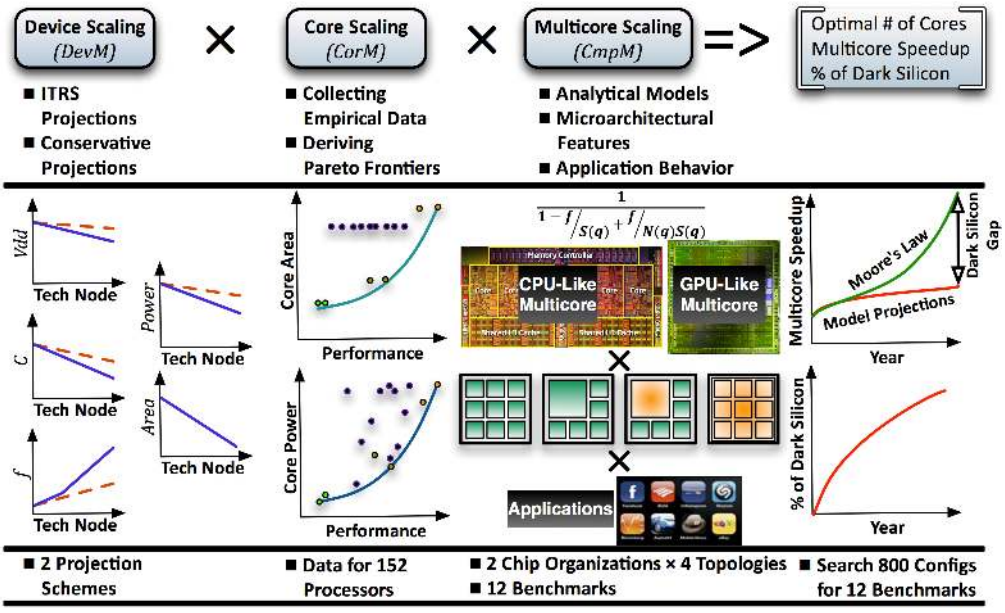


Figure 1: Overview of the models and the methodology

mance of any application for "any" chip topology for CPU-like and GPU-like multicore performance.

- *DevM* × *CorM*: Pareto frontiers at future technology nodes; any performance improvements for future cores will come only at the cost of area or power as defined by these curves.
- *CmpM*×*DevM*×*CorM* and an exhaustive state-space search: maximum multicore speedups for future technology nodes while enforcing area, power, and benchmark constraints.

The results from this study provide detailed best-case multicore performance speedups for future technologies considering real applications from the PARSEC benchmark suite [5]. Our results evaluating the PARSEC benchmarks and our upper-bound analysis confirm the following intuitive arguments:

i) Contrary to conventional wisdom on performance improvements from using multicores, over five technology generations, only 7.9× average speedup is possible using ITRS scaling.

ii) While transistor dimensions continue scaling, power limitations curtail the usable chip fraction. At 22 nm (i.e. in 2012), 21% of the chip will be dark and at 8 nm, over 50% of the chip will not be utilized using ITRS scaling.

iii) Neither CPU-like nor GPU-like multicore designs are sufficient to achieve the expected performance speedup levels. Radical microarchitectural innovations are necessary to alter the power/performance Pareto frontier to deliver speed-ups commensurate with Moore's Law.

2. OVERVIEW

Figure 1 shows how this paper combines models and empirical measurements to project multicore performance and chip utilization. There are three components used in our approach:

Device scaling model (*DevM*)**:** We build a device-scaling model that provides the area, power, and frequency scaling factors at technology nodes from 45 nm to 8 nm. We consider ITRS Roadmap projections [18] and conservative scaling parameters from Borkar's recent study [7].

Core scaling model (*CorM*)**:** The core-level model provides the maximum performance that a single-core can sustain for any given area. Further, it provides the minimum power (or energy) that must be consumed to sustain this level of performance. To quantify, we measure the core performance in terms of SPECmark. We consider empirical data from a large set of processors and use curve fitting to obtain the Pareto-optimal frontiers for single-core area/performance and power/performance tradeoffs.

Multicore scaling model (*CmpM*)**:** We model two mainstream classes of multicore organizations, multi-core CPUs and many-thread GPUs, which represent two extreme points in the threads-per-core spectrum. The CPU multicore organization represents Intel Nehalem-like, heavy-weight multicore designs with fast caches and high single-thread performance. The GPU multicore organization represents NVIDIA Tesla-like lightweight cores with heavy multithreading support and poor single-thread performance. For each multicore organization, we consider four topologies: symmetric, asymmetric, dynamic, and composed (also called "fused" in the literature).

Symmetric Multicore: The symmetric, or homogeneous, multicore topology consists of multiple copies of the same core operating at the same voltage and frequency setting. In a symmetric multicore, the resources, including the power and the area budget, are shared equally across all cores.

Asymmetric Multicore: The asymmetric multicore topology consists of one large monolithic core and many identical small cores. The design leverages the high-performing large core for the serial portion of code and leverages the numerous small cores as well as the large core to exploit the parallel portion of code.

Dynamic Multicore: The dynamic multicore topology is a variation of the asymmetric multicore topology. During parallel code portions, the large core is shut down and, conversely, during the serial portion, the small cores are turned off and the code runs only on the large core [8, 26].

Composed Multicore: The composed multicore topology consists of a collection of small cores that can logically fuse together to compose a high-performance large core for the execution of the serial portion of code [17, 19]. In either serial or parallel cases, the large core or the small cores are used exclusively.

Table 1 outlines the design space we explore and explains the roles of the cores during serial and parallel portions of applica-

Table 1: CPU and GPU topologies (ST Core: Single-Thread Core and MT: Many-Thread Core)

		Symmetric	Asymmetric	Dynamic	Composed
CPU Multicores	Serial	1 ST Core	1 Large ST Core	1 Large ST Core	1 Large ST Core
	Parallel	*N* ST Cores	1 Large ST Core + *N* Small ST Cores	*N* Small ST Cores	*N* Small ST Cores
GPU Multicores	Serial	1 MT Core (1 Thread)	1 Large ST Core (1 Thread)	1 Large ST Core (1 Thread)	1 Large ST Core (1 Thread)
	Parallel	*N* MT Cores (Multiple Threads)	1 Large ST Core + *N* Small MT Cores (1 Thread) (Multiple Threads)	*N* Small MT Cores (Multiple Threads)	*N* Small MT Cores (Multiple Threads)

Table 2: Scaling factors for ITRS and Conservative projections.

	Year	Tech Node (nm)	Frequency Scaling Factor (/45nm)	Vdd Scaling Factor (/45nm)	Capacitance Scaling Factor (/45nm)	Power Scaling Factor (/45nm)
ITRS	2010	45*	1.00	1.00	1.00	1.00
	2012	32*	1.09	0.93	0.7	0.66
	2015	22†	2.38	0.84	0.33	0.54
	2018	16†	3.21	0.75	0.21	0.38
	2021	11†	4.17	0.68	0.13	0.25
	2024	8†	3.85	0.62	0.08	0.12
	31% frequency increase and 35% power reduction per node					
Conservative	2008	45	1.00	1.00	1.00	1.00
	2010	32	1.10	0.93	0.75	0.71
	2012	22	1.19	0.88	0.56	0.52
	2014	16	1.25	0.86	0.42	0.39
	2016	11	1.30	0.84	0.32	0.29
	2018	8	1.34	0.84	0.24	0.22
	6% frequency increase and 23% power reduction per node					

∗: Extended Planar Bulk Transistors, †:Multi-Gate Transistors

tions. Single-thread (ST) cores are uni-processor style cores with large caches and many-thread (MT) cores are GPU-style cores with smaller caches; both are described in more detail in Section 5.

This paper describes an analytic model that provides system-level performance using as input the core's performance (obtained from *CorM*) and the multicore's organization (CPU-like or GPU-like). Unlike previous studies, the model considers application behavior, its memory access pattern, the amount of thread-level parallelism in the workload, and microarchitectural features such as cache size, memory bandwidth, etc. We choose the PARSEC benchmarks because they represent a set of highly parallel applications that are widely studied in the research community.

Heterogeneous configurations such as AMD Fusion and Intel Sandybrige combine CPU and GPU designs on a single chip. The asymmetric and dynamic GPU topologies resemble those two designs, and the composed topology models configurations similar to AMD Bulldozer. For GPU-like multicores, this study assumes that the single ST core does not participate in parallel work. Finally, our methodology implicitly models heterogeneous cores of different types (mix of issue widths, frequencies, etc.) integrated on one chip. Since we perform a per-benchmark optimal search for each organization and topology, we implicitly cover the upper-bound of this heterogeneous case.

3. DEVICE MODEL

We consider two different technology scaling schemes to build a device scaling model. The first scheme uses projections from the ITRS 2010 technology roadmap [18]. The second scheme, which we call conservative scaling, is based on predictions presented by Borkar and represents a less optimistic view [7]. The parameters used for calculating the power and performance scaling factors are summarized in Table 2. For ITRS scaling, frequency is assumed to scale linearly with respect to FO4 inverter delay. The power scaling factor is computed using the predicted frequency, voltage, and gate capacitance scaling factors in accordance with the $P = \alpha C V_{dd}^2 f$ equation. The ITRS roadmap predicts that multi-gate MOSFETs, such as FinTETs, will supersede planar bulk at 22 nm [18]. Table 2 also highlights the key difference between the two projections. Details on how we handle the partitioning between leakage power and dynamic power is explained in Section 4.2.

4. CORE MODEL

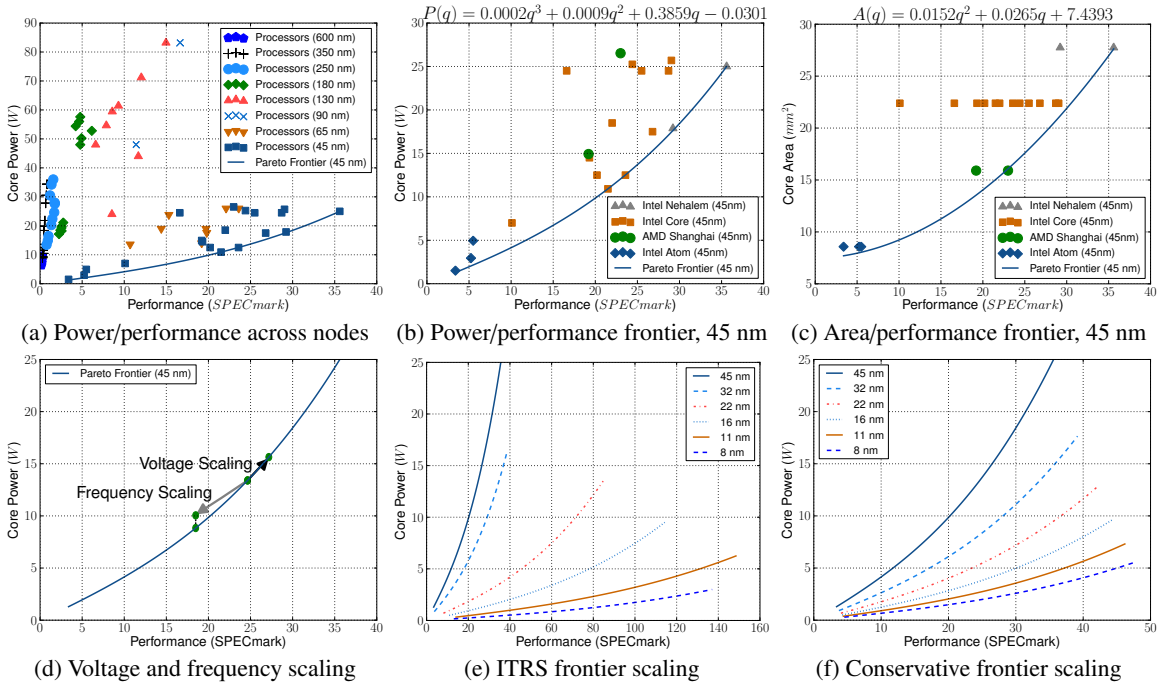
This paper uses Pareto frontiers to provide single-core power/performance and area/performance tradeoffs at each technology node while abstracting away specific details of the cores. The Pareto-optimal core model provides two functions, $A(q)$ and $P(q)$, representing the area/performance and power/performance tradeoff Pareto frontiers, where q is the single-threaded performance of a core measured in SPECmarks. These functions are derived from the data collected for a large set of processors. The power/performance Pareto frontier represents the optimal design points in terms of power and performance [16]. Similarly, the area/performance Pareto frontier represents the optimal design points in the area/performance design space. Below, we first describe why separate area and power functions are required. Then, we describe the basic model and empirical data used to derive the actual Pareto frontier curves at 45 nm. Finally, we project these power and area Pareto frontiers to future technology nodes using the device scaling model.

4.1 Decoupling Area and Power Constraints

Previous studies on multicore performance modeling [8, 9, 10, 15, 20, 22, 28] use Pollack's rule [6] to denote the tradeoff between transistor count and performance. Furthermore, these studies consider the power consumption of a core to be directly proportional to its transistor count. This assumption makes power an area-dependent constraint. However, power is a function of not only area, but also supply voltage and frequency. Since these no longer scale at historical rates, Pollack's rule is insufficient for modeling core power. Thus, it is necessary to decouple area and power into two independent constraints.

4.2 Pareto Frontier Derivation

Figure 2(a) shows the power/performance single-core design space. We populated the depicted design space by collecting data for 152 real processors (from P54C Pentium to Nehalem-based i7) fabricated at various technology nodes from 600 nm through 45 nm. As shown, the boundary of the design space that comprises the power/performance optimal points constructs the Pareto frontier. Each processor's performance is collected from the SPEC website [25] and the processor's power is the TDP reported in its datasheet. Thermal design power, TDP, is the chip power budget, the amount of power the chip can dissipate without exceeding transistors junction temperature. In Figure 2(a), the x-axis is the SPEC CPU2006 [25]



(a) Power/performance across nodes (b) Power/performance frontier, 45 nm (c) Area/performance frontier, 45 nm

(d) Voltage and frequency scaling (e) ITRS frontier scaling (f) Conservative frontier scaling

Figure 2: Deriving the area/performance and power/performance Pareto frontiers

score (SPECmark) of the processor, and the y-axis is the core power budget. All SPEC scores are converted to SPEC CPU2006 scores.

Empirical data for the core model: To build a technology-scalable model, we consider a family of processors at one technology node (45 nm) and construct the frontier for that technology node. We used 20 representative Intel and AMD processors at 45 nm (Atom Z520, Atom 230, Atom D510, C2Duo T9500, C2Extreme QX9650, C2Q-Q8400, Opteron 2393SE, Opteron 2381HE, C2Duo E7600, C2Duo E8600, C2Quad Q9650, C2Quad QX9770, C2Duo T9900, Pentium SU2700, Xeon E5405, Xeon E5205, Xeon X3440, Xeon E7450, i7-965 ExEd). The power/performance design space and the cubic Pareto frontier at 45 nm, $P(q)$, are depicted in Figure 2(b).

To derive the quadratic area/performance Pareto frontier (Figure 2(c)), die photos of four microarchitectures, including Intel Atom, Intel Core, AMD Shanghai, and Intel Nehalem, are used to estimate the core areas (excluding level 2 and level 3 caches). The Intel Atom Z520 with a 2.2 W total TDP represents the lowest power design (lower-left frontier point), and the Nehalem-based Intel Core i7-965 Extreme Edition with a 130 W total TDP represents the highest performing (upper-right frontier point). Other low-power architectures, such as those from ARM and Tilera, were not included because their SPECmark were not available for a meaningful performance comparison.

Since the focus of this work is to study the impact of power constraints on logic scaling rather than cache scaling, we derive the Pareto frontiers using only the portion of chip *power budget* (TDP) allocated to each core. To compute the power budget of a single core, the power budget allocated to the level 2 and level 3 caches is estimated and deducted from the chip TDP. In the case of a multicore CPU, the remainder of the chip power budget is divided by the number of cores, resulting in the power budget allocated to a single core (1.89 W for the Atom core in Z520 and 31.25 W for each Nehalem core in i7-965 Extreme Edition). We allocate 20% of the chip power budget to leakage power. As shown in [24], the transistor threshold voltage can be selected so that the maximum leakage power is always an acceptable ratio of the chip power budget while still meeting the power and performance constraints. We also observe that with 10% or 30% leakage power, we do not see significant changes in optimal configurations.

Deriving the core model: To derive the Pareto frontiers at 45 nm, we fit a cubic polynomial, $P(q)$, to the points along the edge of the *power/performance* design space. We fit a quadratic polynomial (Pollack's rule), $A(q)$, to the points along the edge of the *area/performance* design space. We used the least square regression method for curve fitting such that the frontiers enclose all design points. Figures 2(b) and 2(c) show the 45 nm processor points and identify the power/performance and area/performance Pareto frontiers. The power/performance cubic polynomial $P(q)$ function (Figure 2(b)) and the area/performance quadratic polynomial $A(q)$ (Figure 2(c)) are the outputs of the *core model*. The points along the Pareto frontier are used as the search space for determining the best core configuration by the multicore-scaling model. We discretized the frontier into 100 points to consider 100 different core designs.

Voltage and frequency scaling: When deriving the Pareto frontiers, each processor data point was assumed to operate at its optimal voltage (Vdd_{min}) and frequency setting ($Freq_{max}$). Figure 2(d) shows the result of voltage/frequency scaling on the design points along the power/performance frontier. As depicted, at a fixed Vdd setting, scaling down the frequency from $Freq_{max}$, results in a power/performance point inside of the optimal Pareto curve, or a suboptimal design point. Scaling voltage up, on the other hand, and operating at a new Vdd_{min} and $Freq_{max}$ setting, results in a different power-performance point along the frontier. Since we investigate all the points along the Pareto frontier to find the optimal multicore configuration, voltage and frequency scaling does not require special consideration in our study. If an application dissipates less than the power budget, we assume that the voltage and frequency scaling will be utilized to achieve the highest possible performance with the minimum power increase. This is possible since voltage and frequency scaling only changes the operating condition in a Pareto-optimal fashion. Hence, we do not need to measure per-benchmark power explicitly as reported in a recent study [12].

Table 3: $CmpM_U$ equations: corollaries of Amdahl's Law for power-constrained multicores.

Symmetric
$N_{Sym}(q) = min(\frac{DIE_{AREA}}{A(q)}, \frac{TDP}{P(q)})$
$Speedup_{Sym}(f,q) = \frac{1}{\frac{(1-f)}{S_U(q)} + \frac{f}{N_{Sym}(q) S_U(q)}}$
Asymmetric
$N_{Asym}(q_L, q_S) = min(\frac{DIE_{AREA} - A(q_L)}{A(q_S)}, \frac{TDP - P(q_L)}{P(q_S)})$
$Speedup_{Asym}(f,q_L,q_S) = \frac{1}{\frac{(1-f)}{S_U(q_L)} + \frac{f}{N_{Asym}(q_L,q_S) S_U(q_S) + S_U(q_L)}}$
Dynamic
$N_{Dyn}(q_L, q_S) = min(\frac{DIE_{AREA} - A(q_L)}{A(q_S)}, \frac{TDP}{P(q_S)})$
$Speedup_{Dyn}(f,q_L,q_S) = \frac{1}{\frac{(1-f)}{S_U(q_L)} + \frac{f}{N_{Dyn}(q_L,q_S) S_U(q_S)}}$
Composed
$N_{Composd}(q_L, q_S) = min(\frac{DIE_{AREA}}{(1+\tau)A(q_S)}, \frac{TDP - Pq_L}{P(q_S)})$
$Speedup_{Composed}(f,q_L,q_S) = \frac{1}{\frac{(1-f)}{S_U(q_L)} + \frac{f}{N_{Composed}(q_L,q_S) S_U(q_S)}}$

4.3 Device Scaling × Core Scaling

To study core scaling in future technology nodes, we scaled the 45 nm Pareto frontiers to 8 nm by scaling the power and performance of each processor data point using the projected *DevM* scaling factors and then re-fitting the Pareto optimal curves at each technology node. Performance, measured in SPECmark, is assumed to scale linearly with frequency. This is an optimistic assumption, which ignores the effects of memory latency and bandwidth on the performance. Thus actual performance through scaling is likely to be lower. Figures 2(e) and 2(f) show the scaled Pareto frontiers for the ITRS and conservative scaling schemes. Based on the optimistic ITRS roadmap predictions, scaling a microarchitecture (core) from 45 nm to 8 nm will result in a 3.9× performance improvement and an 88% reduction in power consumption. Conservative scaling, however, suggests that performance will increase only by 34%, and power will decrease by 74%.

5. MULTICORE MODEL

We first present a simple upper-bound ($CmpM_U$) model for multicore scaling that builds upon Amdahl's Law to estimate the speedup of area- and power-constrained multicores. To account for microarchitectural features and application behavior, we then develop a detailed chip-level model ($CmpM_R$) for CPU-like and GPU-like multicore organizations with different topologies. Both models use the $A(q)$ and $P(q)$ frontiers from the core-scaling model.

5.1 Amdahl's Law Upper-bounds: $CmpM_U$

Hill and Marty extended Amdahl's Law [1] to study a range of multicore topologies by considering the fraction of parallel code in a workload [15]. Their models describe symmetric, asymmetric, dynamic, and composed multicore topologies, considering area as the constraint and using Pollack's rule–the performance of a core is proportional to the square root of its area–to estimate the performance of multicores. We extend their work and incorporate power as a primary design constraint, independent of area. Then, we determine the optimal number of cores and speedup for topology. The $CmpM_U$ model does not differentiate between CPU-like and GPU-like architectures, since it abstracts away the chip organization.

Per Amdahl's Law [1], system speedup is $\frac{1}{(1-f)+\frac{f}{S}}$ where f represents the portion that can be optimized, or enhanced, and S represents the speedup achievable on the enhanced portion. In the case of parallel processing with perfect parallelization, f can be thought of as the parallel portion of code and S as the number of processor cores. Table 3 lists the derived corollaries for each multicore topology, where TDP is the chip power budget and DIE_{AREA} is the area budget. The q parameter denotes the performance of a single core. Speedup is measured against a baseline core with performance $q_{Baseline}$. The upper-bound speedup of a single core over the baseline is computed as $S_U(q) = q/q_{Baseline}$.

Symmetric Multicore: The parallel fraction (f) is distributed across the $N_{Sym}(q)$ cores each of which has $S_U(q)$ speedup over the baseline. The serial code-fraction, $1 - f$, runs only on one core.

Asymmetric Multicore: All cores (including the large core), contribute to execution of the parallel code. Terms q_L and q_S denote performance of the large core and a single small core, respectively. The number of small cores is bounded by the power consumption or area of the large core.

Dynamic Multicore: Unlike the asymmetric case, if power is the dominant constraint, the number of small cores is not bounded by the power consumption of the large core. However, if area is the dominant constraint, the number of small cores is bounded by the area of the large core.

Composed Multicore: The area overhead supporting the composed topology is τ. Thus, the area of small cores increases by a factor of $(1 + \tau)$. No power overhead is assumed for the composability support in the small cores. We assume that τ increases from 10% up to 400% depending on the total area of the composed core. We assume performance of the composed core cannot exceed performance of a scaled single-core Nehalem at 45 nm. The composed core consumes the power of a same-size uniprocessor core.

5.2 Realistic Performance Model: $CmpM_R$

The above corollaries provide a strict upper-bound on parallel performance, but do not have the level of detail required to explore microarchitectural features (cache organization, memory bandwidth, number of threads per core, etc.) and workload behavior (memory access pattern and level of multithread parallelism in the application). Guz et al. proposed a model to consider first-order impacts of these additional microarchitectural features [13]. We extend their approach to build the multicore model that incorporates application behavior, microarchitectural features, and physical constraints. Using this model, we consider single-threaded cores with large caches to cover the CPU multicore design space and massively threaded cores with minimal caches to cover the GPU multicore design space. For each of these multicore organizations, we consider the four possible topologies.

The $CmpM_R$ model formulates the performance of a multicore in terms of chip organization (CPU-like or GPU-like), frequency, CPI, cache hierarchy, and memory bandwidth. The model also includes application behaviors such as the degree of thread-level parallelism, the frequency of load and store instructions, and the cache miss rate. To first order, the model considers stalls due to memory dependences and resource constraints (bandwidth or functional units). The input parameters to the model, and how, if at all, they are impacted by the multicore design choices are listed in Table 4.

Microarchitectural Features

Multithreaded performance ($Perf$) of an Multithreaded performance ($Perf$) of an either CPU-like or GPU-like multicore running a fully parallel ($f = 1$) and multithreaded application is calculated in terms of instructions per second in Equation (1) by multiplying the number of cores (N) by the core utilization (η) and scaling by the ratio of the processor frequency to CPI_{exe}:

$$Perf = \min\left(N\frac{freq}{CPI_{exe}}\eta, \frac{BW_{max}}{r_m \times m_{L1} \times b}\right) \quad (1)$$

Table 4: $CmpM_R$ parameters with default values from 45 nm Nehalem

Parameter	Description	Default	Impacted By
N	Number of cores	4	Multicore Topology
T	Number of threads per core	1	Core Style
$freq$	Core frequency (MHz)	3200	Core Performance
CPI_{exe}	Cycles per instruction (zero-latency cache accesses)	1	Core Performance, Application
C_{L1}	L1 cache size per core (KB)	64	Core Style
C_{L2}	L2 cache size per chip (MB)	2	Core Style, Multicore Topology
t_{L1}	L1 access time (cycles)	3	-
t_{L2}	L2 access time (cycles)	20	-
t_{mem}	Memory access time (cycles)	426	Core Performance
BW_{max}	Maximum memory bandwidth (GB/s)	200	Technology Node
b	Bytes per memory access (B)	64	-
f	Fraction of code that can be parallel	varies	Application
r_m	Fraction of instructions that are memory accesses	varies	Application
α_{L1}, β_{L1}	L1 cache miss rate function constants	varies	Application
α_{L2}, β_{L2}	L2 cache miss rate function constants	varies	Application

The CPI_{exe} parameter does not include stalls due to cache accesses, which are considered separately in the core utilization (η). The core utilization is the fraction of time that a thread running on the core can keep it busy. It is modeled as a function of the average time spent waiting for each memory access (t), fraction of instructions that access the memory (r_m), and the CPI_{exe}:

$$\eta = \min\left(1, \frac{T}{1 + t\frac{r_m}{CPI_{exe}}}\right) \tag{2}$$

The average time spent waiting for memory accesses is a function of the time to access the caches (t_{L1} and t_{L2}), time to visit memory (t_{mem}), and the predicted cache miss rate (m_{L1} and m_{L2}):

$$t = (1 - m_{L1})t_{L1} + m_{L1}(1 - m_{L2})t_{L2} + m_{L1}m_{L2}t_{mem} \tag{3}$$

$$m_{L1} = \left(\frac{C_{L1}}{T\beta_{L1}}\right)^{1-\alpha_{L1}} \text{ and } m_{L2} = \left(\frac{C_{L2}}{NT\beta_{L2}}\right)^{1-\alpha_{L2}} \tag{4}$$

The $Perf$ part of the $CmpM_R$ model is based on Guz et al.'s model [13], and is summarized by Equations (1)-(4).

One of the contributions of this work is incorporating real application behavior and realistic microarchitectural features into the multithreaded speedup formulation at each technology node. The parameters listed in Table 4 define the microarchitectural design choices for a multicore topology, while taking into account the application characteristics and behavior. To compute the overall speedup of different multicore topologies using the $CmpM_R$ model, we calculate the baseline multithreaded performance for all benchmarks by providing the $Perf$ equation with the inputs corresponding to a Quad-core Nehalem at 45 nm.

Multi-level caches: To model a second level of cache, we add a miss rate prediction function similar to that for the single layer of cache. This extension is completely modeled by a small modification to the average memory access time, as shown in Equation (3).

Obtaining frequency and CPI_{exe} from Pareto frontiers: To incorporate the Pareto-optimal curves into the $CmpM_R$ model, we convert the SPECmark scores into an estimated CPI_{exe} and core frequency. We assume the core frequency scales linearly with performance, from 1.5 GHz for an Atom core to 3.2 GHz for a Nehalem core. Each application's CPI_{exe} is dependent on its instruction mix and use of hardware optimizations (e.g., functional units and out-of-order processing). Since the measured CPI_{exe} for each benchmark at each technology node is not available, we use the $CmpM_R$ model to generate per benchmark CPI_{exe} estimates for each design point along the Pareto frontiers. With all other model inputs kept constant, we iteratively search for the CPI_{exe} at each processor design point. We start by assuming that the Nehalem core has a CPI_{exe} of ℓ. Then, the smallest core, an Atom processor, should have a CPI_{exe} such that the ratio of its $CmpM_R$ performance to the Nehalem core's $CmpM_R$ performance is the same as the ratio of their SPECmark scores. Since the performance increase between any two points should be the same using either the SPECmark score or $CmpM_R$ model, we continue in this fashion to estimate a per benchmark CPI_{exe} for each processor design point. We observe CPI_{exe}s greater than 10ℓ for the Atom node and that CPI_{exe} decreases monotonically to ℓ for the Nehalem node. We assume CPI_{exe} does not change with technology node, while frequency scales as discussed in Section 4.3. This flexible approach allows us to use the SPECmark scores to select processor design points from the Pareto optimal curves and generate reasonable performance model inputs.

Application Behavior

To characterize an application, the required input parameter models are cache behavior, fraction of instructions that are loads or stores, and fraction of parallel code. For the PARSEC applications, we obtain this data from two previous studies [4, 5]. To obtain f, the fraction of parallel code, for each benchmark, we fit an Amdahl's Law-based curve to the reported speedups across different numbers of cores from both studies. This fit shows values of f between 0.75 and 0.9999 for individual benchmarks.

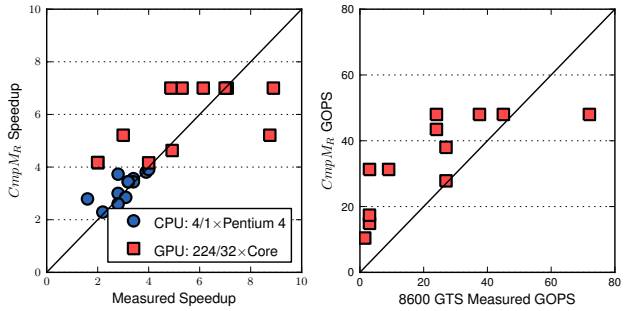
Multicore Topologies

Table 1 described the combinations of ST (single-thread) and MT (many-thread) cores constructing each of the four CPU/GPU topologies. The number of cores that fit in the chip's area and power budget is chosen using the equations for N in Table 3. For each design point, we compute serial performance ($Perf_S$) using Equations (1)-(4) with a single thread and the serial core parameters and parallel performance ($Perf_P$) using Equations (1)-(4) with the number of parallel cores and the parallel core parameters. We also compute the performance of a single Nehalem core at 45 nm as our baseline ($Perf_B$). The serial portion of code is thus sped up by $S_{R,Serial} = Perf_S/Perf_B$ and the parallel portion of the code is sped up by $S_{R,Parallel} = Perf_P/Perf_B$. The overall speedup is computed below; this formulation captures the impact of parallelism on all four topologies:

$$Speedup_R = 1/\left(\frac{1-f}{S_{R,Serial}} + \frac{f}{S_{R,Parallel}}\right) \tag{5}$$

Table 5: Effect of assumptions on $CmpM_R$ accuracy. Assumptions lead to ↑ (slightly higher), ⇑ (higher) or ↓ (slightly lower) predicted speedups (or have no effect (—)).

	Assumption	Impact on CPU Speed	Impact on GPU Speed
μarch	Memory Contention: 0	↑	⇑
	Interconnection Network Latency: 0	⇑	⇑
	Thread Swap Time: 0	↑	↑
Application	Cache Hit Rate Function	↑ or ↓	↑ or ↓
	Thread Synch & Communication: 0	↑	⇑
	Thread Data Sharing: 0	↓	—
	Workload Type: Homogeneous	↑	↑



(a) Speedup (b) Performance

Figure 3: $CmpM_R$ validation

Physical Constraints

A key component of the detailed model is the set of input parameters that model the microarchitecture of the cores. As discussed, we model two styles of cores: single-thread (ST) and many-thread (MT). For single-thread cores, we assume each core has a 64 KB L1 cache, and chips with only ST cores have an L2 cache that is 30% of the chip area. Many-thread cores have small L1 caches (32 KB for every 8 cores), support multiple hardware contexts (1024 threads per 8 cores), a thread register file, and no L2 cache. From Atom and Tesla die photo inspections, we estimate that 8 small MT cores, their shared L1 cache, and their thread register file can fit in the same area as one Atom processor. We assume a similar correspondence with power, discussed in Section 7.6. We further assume that off-chip bandwidth (BW_{max}) increases linearly as process technology scales down while the memory access time is constant.

Model Assumptions

The model's accuracy is limited by our assumptions which are optimistic. Thus, the model only over-predicts performance, making our speedup projections optimistic. This model allows us to estimate the first-order impact of caching, parallelism, and threading under several key assumptions. It optimistically assumes that the workload is homogeneous, work is infinitely parallel during parallel sections of code, and no thread synchronization, operating system serialization, or swapping overheads occur. We also assume memory accesses never stall due to a previous access. Each of these assumptions could cause the model to overpredict performance, making the model and hence projected speedups optimistic. Cache behaviors may lead to over- or under-prediction. The model assumes that each thread effectively only sees its own slice of the cache and the cache hit rate function may over or underestimate. Table 5 qualitatively describes the impact of these assumptions.

Model Validation

To validate the $CmpM_R$ model, we compare the speedup projections from the model to measurement and simulation results for both CPU and GPU organizations. For the CPU case, we compare the model's predicted speedup to measured PARSEC speedup on a quad-Pentium 4 multicore [4]. The model is configured to match this system. We validate GPU speedup projections by comparing the model's output to GPGPUSim [3] simulation results. Both model and simulator compare speedups of a 224-core GPU over a 32-core GPU. We use GPGPUSim's 12 CUDA benchmarks since GPU implementations of PARSEC are not available. Figure 3(a), which includes both CPU and GPU data, shows that the model is optimistic. $CmpM_R$ underpredicts speedups for two benchmarks; these speedups are greater than 7× (the increase in number of cores).

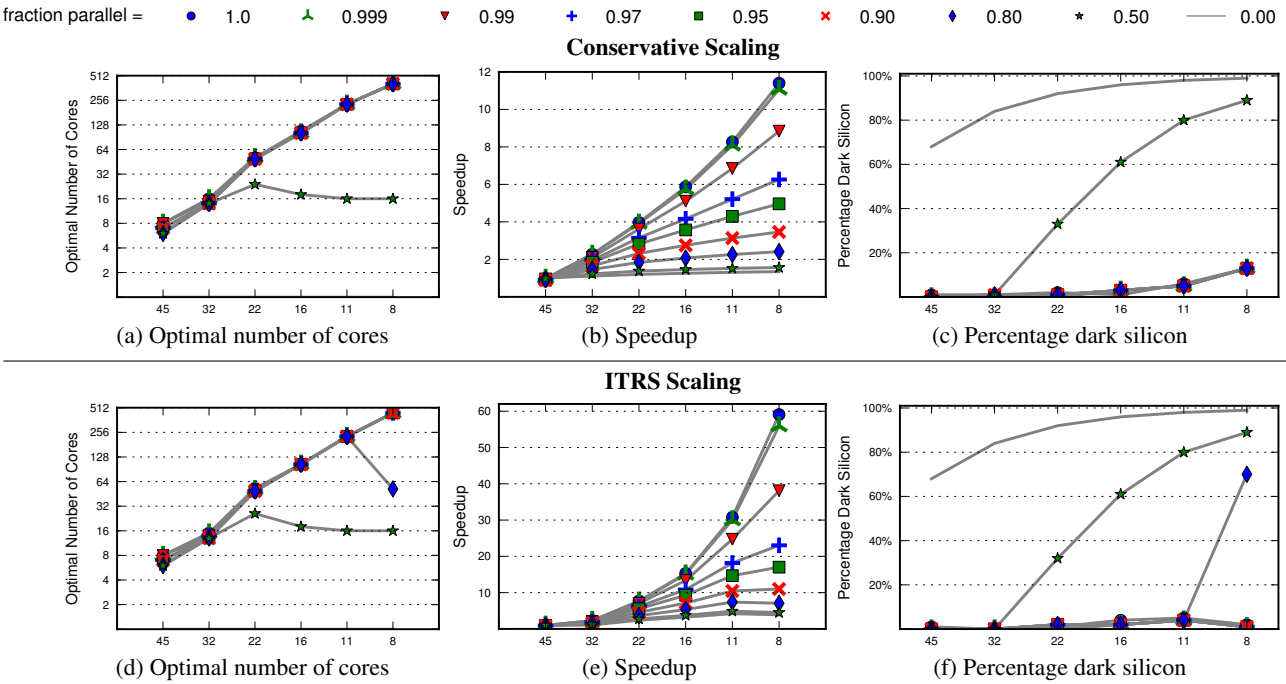
To strongly advance our GPU claim, we also need to prove the model's raw performance projection is accurate or optimistic. As depicted in Figure 3(b), the model's GPU performance projection is validated by comparing its output to the results from a real system, NVIDIA 8600 GTS, using the data from [3]. Except for a known anomaly that also occurs in GPGPUsim, $CmpM_R$ consistently overpredicts raw performance.

Using our model, we find 4× geometric-mean and 12× maximum speedup for PARSEC benchmarks on Tesla compared to a quad-core Nehalem. While our results are impressively close to Intel's empirical measurements using similar benchmarks [21], the match in the model's maximum speedup prediction (12× vs 11× in the Intel study) is an anomaly. Our model does not account for specialized compute units, which contribute to the speedup in [21].

6. DEVICE × CORE × CMP SCALING

We now describe how the three models are combined to produce projections for optimal performance, number of cores, and amount of dark silicon. To determine the best core configuration at each technology node, we consider only the processor design points along the area/performance and power/performance Pareto frontiers as they represent the most efficient design points. The following outlines the process for choosing the optimal core configuration for the symmetric topology at a given technology node (the procedure is similar for the other topologies):

- The area/performance Pareto frontier is investigated, and all processor design points along the frontier are considered.
- For each area/performance design point, the multicore is constructed starting with a single core. We add one core per iteration and compute the new speedup and the power consumption using the power/performance Pareto frontier.
- Speedups are computed using the Amdahl's Law corollaries ($CmpM_U$ model) to obtain an upper-bound or our $CmpM_R$ model for more realistic performance results using the PARSEC benchmarks. The speedup is computed over a quad-core Nehalem at 45 nm.
- After some number of iterations, the area limit is hit, or power wall is hit, or we start seeing performance degradation. At this point the optimal speedup and the optimal number of cores is found. The fraction of dark silicon can then be computed by subtracting the area occupied by these cores from the total die area allocated to processor cores.
- The above process is repeated for each technology node using the scaled Pareto frontiers. The core power budget (excluding level 2 and level 3 caches) is held constant at 125 W (core power budget of a 4-core Nehalem-based multicore at 45 nm), and the core area budget is held constant at 111 mm^2 (area of 4 Nehalem cores at 45 nm, excluding level 2



(a) Optimal number of cores (b) Speedup (c) Percentage dark silicon

(d) Optimal number of cores (e) Speedup (f) Percentage dark silicon

Figure 4: Amdahl's law projections for the dynamic topology. Upperbound of all four topologies (x-axis: technology node).

and level 3 caches). *The reported projections of dark silicon are for the area budget that is solely allocated to the cores, not caches and other 'uncore' components.*

This exhaustive search is performed separately for Amdahl's Law $CmpM_U$, CPU-like $CmpM_R$, and GPU-like $CmpM_R$ models. We optimistically add cores until either the power or area budget is reached. We also require that doubling the number of cores increases performance by at least 10%.

7. SCALING AND FUTURE MULTICORES

We begin the study of future multicoredesigns with an optimistic upper-bound analysis using the Amdahl's Law multicore-scaling model, $CmpM_U$. Then, to achieve an understanding of speedups for real workloads, we consider the PARSEC benchmarks and examine both CPU-like and GPU-like multicore organizations under the four topologies using our $CmpM_R$ model. We also describe sources of dark silicon and perform sensitivity studies for cache organization and memory bandwidth.

7.1 Upper-bound Analysis using Amdahl's Law

Figures 4(a)-(c) show the multicore scaling results comprising the optimal number of cores, achievable speedup, and dark silicon fraction under conservative scaling. Figures 4(d)-(f) show the same results using ITRS scaling. The results are only presented for the dynamic topology, which offers the best speedup levels amongst the four topologies. These results are summarized below.

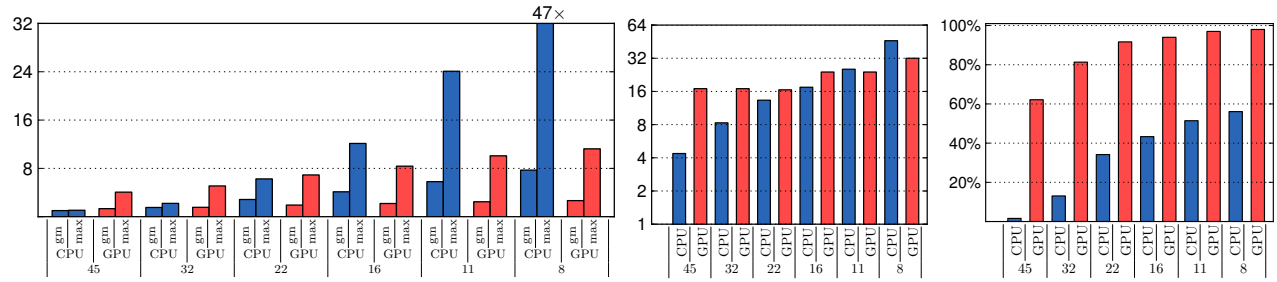
Characteristic	**Conservative**	**ITRS**
Maximum Speedup	11.3×	59×
Typical # of Cores	< 512	< 512
Dark Silicon Dominates	—	2024

The 59× speedup at 8 nm for highly parallel workloads using ITRS predictions, which exceeds the expected 32×, is due to the optimistic device scaling projections. We consider scaling of the Intel Core2 Duo T9900 to clarify. At 45 nm, the T9900 has a SPECmark of 23.58, frequency of 3.06 GHz, TDP of 35 W and per-core power of 15.63 W and are of 22.30 mm^2. With ITRS scaling at 8nm, T9900 will have SPECmark of 90.78, frequency of 11.78 GHz, core power of 1.88 W, and core area of 0.71 mm^2. With the 125 W power budget at 8nm, 67 such cores can be integrated. There is consensus that such power efficiency is unlikely. Further, our $CmpM_U$ model assumes that performance scales linearly with frequency. These optimistic device and performance assumptions result in speedups exceeding Moore's Law.

7.2 Analysis using Real Workloads

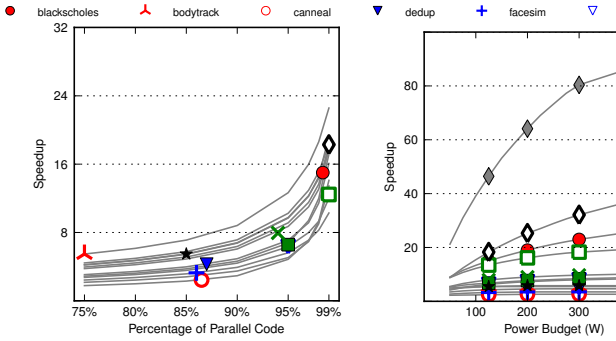
We now consider PARSEC applications executing on CPU- and GPU-like chips. The study considers all four symmetric, asymmetric, dynamic, and composed multicore topologies (see Table 1) using the $CmpM_R$ realistic model. As discussed before, the model captures microarchitectural features as well as application behavior. To conduct a fair comparison between different design points, all speedup results are normalized to the performance of a quad-core Nehalem multicore at 45 nm. In Figure 5, we present the geometric mean of speedup, best-case speedup, geometric mean of the optimal number of cores, and geometric mean of the percentage dark silicon using optimistic ITRS scaling. The symmetric topology achieves the lower bound on speedups; with speedups that are no more than 10% higher, the dynamic and composed topologies achieve the upper-bound. The results are presented for both CPU-like and GPU-like multicore organizations. Details for all applications and topologies are presented in Figure 8. The results are summarized below.

	Conservative		**ITRS**	
Characteristic	CPU	GPU	CPU	GPU
Symmetric GM Speedup	3.4×	2.4×	7.7×	2.7×
Dynamic GM Speedup	3.5×	2.4×	7.9×	2.7×
Maximum Speedup	10.9×	10.1×	46.6×	11.2×
Typical # of Cores	< 64	< 256	< 64	< 256
Dark Silicon Dominates	2016	2012	2021	2015



(a) Speedup: geomean and best case (b) Number of cores: geomean (c) Percent dark silicon: geomean

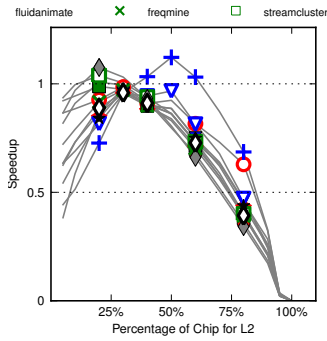
Figure 5: Speedup and number of cores across technology nodes using symmetric topology and ITRS scaling



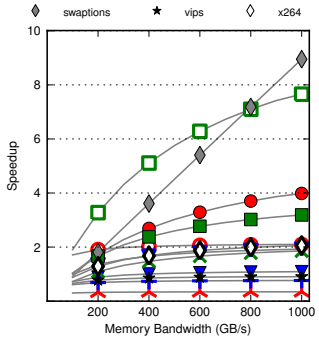
(a) Parallelism (f actual at marker)

(b) Power

Figure 6: Dark silicon bottleneck relaxation using CPU organization and dynamic topology at 8 nm with ITRS Scaling



(a) L2 size (CPU)



(b) Memory bandwidth (GPU)

Figure 7: Sensitivity studies of L2 size and memory bandwidth using symmetric topology at 45 nm

The optimal number of cores projected by our study seems small compared to chips such as the NVIDIA Fermi, which has 512 cores at 45 nm. There are two reasons for this discrepancy. First, in our study we are optimizing for a fixed power budget, whereas with real GPUs the power has been slightly increasing. Second, our study optimizes core count and multicore configuration for general purpose workloads similar to the PARSEC suite. We assume Fermi is optimized for graphics rendering. When we applied our methodology to a graphics kernel (ray tracing) in an asymmetric topology, we obtained higher speedups and an optimal core count of 4864 at 8 nm, with 8% dark silicon.

7.3 Sources of Dark Silicon

To understand whether parallelism or power is the primary source of dark silicon, we examine our model results with power and parallelism levels alone varying in separate experiments as shown in Figure 6 for the 8 nm node (2018). First, we set power to be the "only" constraint, and vary the level of parallelism in the PARSEC applications from 0.75 to 0.99, assuming programmer effort can somehow realize this. As shown in Figure 6(a), which normalizes speedup to a quad-core Nehalem at 45 nm, we see performance improves only slowly as the parallelism level increases, with most benchmarks reaching a speedup of about only 15× at 99% parallelism. The markers show the level of parallelism in their current implementation. If power was the only constraint, typical ITRS-scaling speedups would still be limited to 15×. With conservative scaling, this best-case speedup is 6.3×.

We then see what happens if parallelism alone was the constraint by allowing the power budget to vary from 50 W to 500 W (our default budget is 125 W) in Figure 6(b). Eight of twelve benchmarks show no more than 10X speedup even with practically unlimited power, i.e. parallelism is the primary contributor to dark silicon. Only four benchmarks have sufficient parallelism to even hypothetically sustain Moore's Law level speedup, but dark silicon due to power limitations constrains what can be realized.

7.4 Sensitivity Studies

Our analysis thus far examined "typical" configurations and showed poor scalability for the multicore approach. A natural question is, *can simple configuration changes (percentage cache area, memory bandwidth, etc.) provide significant benefits?* Our model allows us to do such studies, and shows that only small benefits are possible from such simple changes. We elaborate on two representative studies below.

L2 cache area: Figure 7(a) shows the optimal speedup at 45 nm as the amount of a symmetric CPU's chip area devoted to L2 cache varies from 0% to 100%. In this study we ignore any increase in L2 cache power or increase in L2 cache access latency. Across the PARSEC benchmarks, the optimal percentage of chip devoted to cache varies from 20% to 50% depending on benchmark memory access characteristics. Compared to a 30% cache area, using optimal cache area only improves performance by at most 20% across all benchmarks.

Memory bandwidth: Figure 7(b) illustrates the sensitivity of PARSEC performance to the available memory bandwidth for symmetric GPU multicores at 45 nm. As the memory bandwidth increases, the speedup improves as the bandwidth can keep more threads fed with data; however, the increases are limited by power and/or parallelism and in 10 out of 12 benchmarks speedups do not increase by more than 2× compared to the baseline, 200 GB/s.

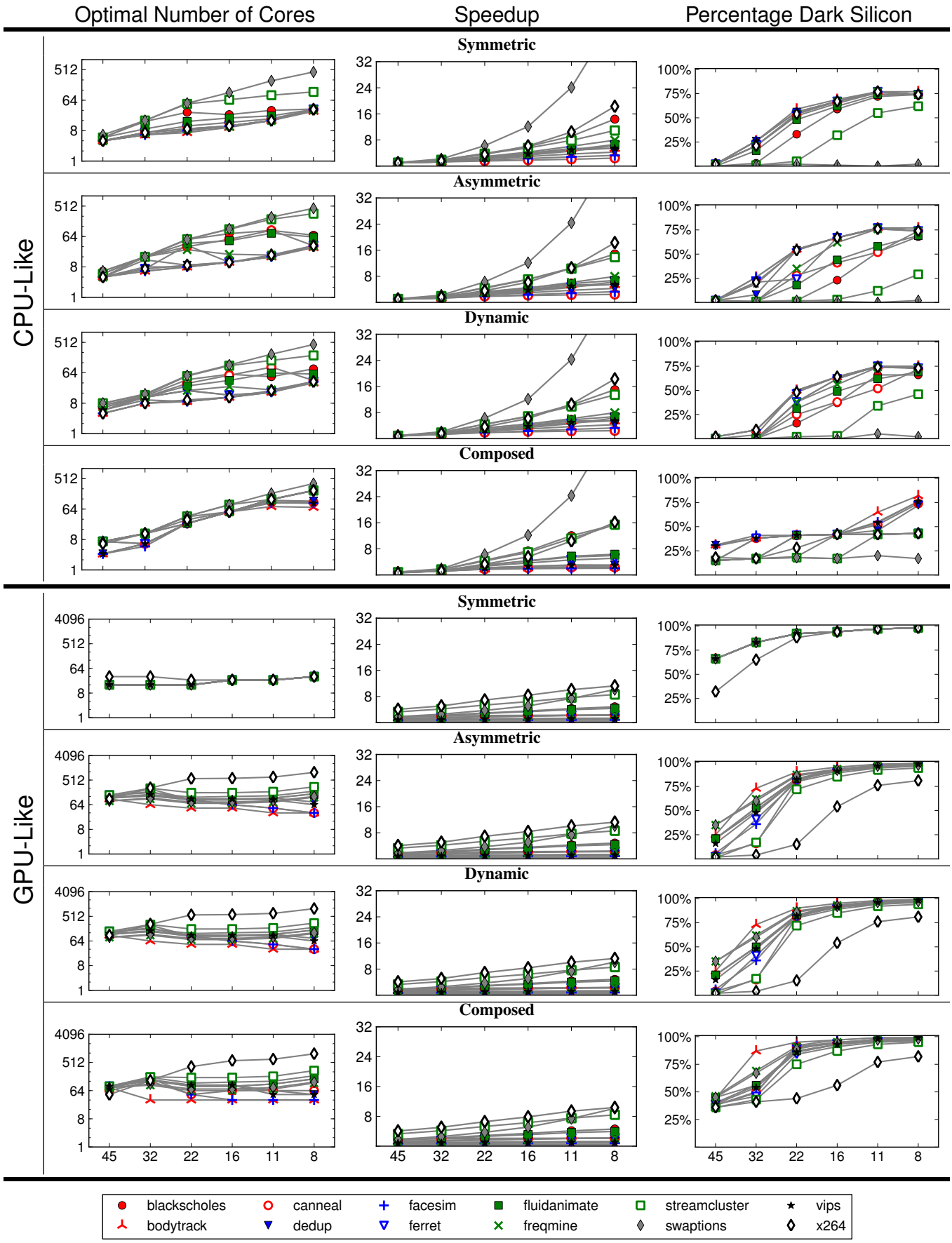
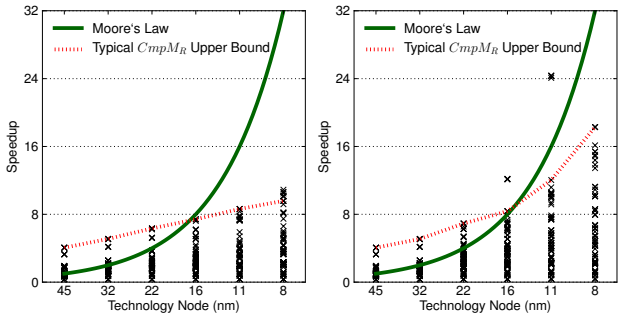


Figure 8: Optimal number of cores, speedup over quad-Nehalem at 45 nm, and percentage dark silicon under ITRS scaling projections using the $CmpM_R$ realistic model.



(a) Conservative Scaling (b) ITRS Scaling

Figure 9: Speedup across process technology nodes over all organizations and topologies with PARSEC benchmarks

7.5 Summary

Figure 9 summarizes all the speedup projections in a single scatter plot. For every benchmark at each technology node, we plot the eight possible configurations, (CPU, GPU) × (symmetric, asymmetric, dynamic, composed). The solid curve indicates *performance* Moore's Law or doubling performance with every technology node. As depicted, due to the power and parallelism limitations, a significant gap exists between what is achievable and what is expected by Moore's Law. Results for ITRS scaling are slightly better but not by much. With conservative scaling a speedup gap of at least 22× exists at the 8 nm technology node compared to Moore's Law. Assuming ITRS scaling, the gap is at least 13× at 8 nm.

7.6 Limitations

Our modeling includes certain limitations, which we argue do not significantly change the results. To simplify the discussion, we did not consider SMT support for the processors (cores) in the CPU multicore organization. SMT support can improve the power efficiency of the cores for parallel workloads to some extent. We studied 2-way, 4-way, and 8-way SMT with no area or energy penalty, and observed that speedup improves with 2-way SMT by 1.5× in the best case and decreases as much as 0.6× in the worst case due to increased cache contention; the range for 8-way SMT is 0.3-2.5×.

Our GPU methodology may over-estimate the GPU power budget, so we investigated the impact of 10%-50% improved energy efficiency for GPUs and found that total chip speedup and percentage of dark silicon were not impacted.

We ignore the power impact of "uncore" components such as the memory subsystem. There is consensus that the number of these components will increase and hence they will further eat into the power budget, reducing speedups.

We do not consider ARM or Tilera cores in this work because they are designed for different application domains and their SPECmark scores were not available for a meaningful comparison. For highly parallel applications, these lightweight cores may achieve higher speedups, but similar to the GPU case, they will likely be limited by bandwidth and available parallelism.

We acknowledge that we make a number of assumptions in this work to build a useful model. Questions may still linger on the model's accuracy and whether its assumptions contribute to the performance projections that fall well below the ideal 32×. First, in all instances, we selected parameter values that would be favorable towards performance. Second, our validation against real and simulated systems (Section 5.2) shows the model always under-predicts performance.

8. RELATED WORK

Hill and Marty applied Amdahl's Law to a range of multicore topologies, including symmetric, asymmetric, and dynamic multicore designs and conclude dynamic topologies are superior [15]. Their models used area as the primary constraint, using Pollack's rule (Performance $\propto \sqrt{Area}$ [6]), to estimate performance. Extensions have been developed for modeling 'uncore' components, such as the interconnection network and last-level cache [22], computing core configuration optimal for energy [9, 20], and leakage power [28]. These studies all model power as a function of area (neglecting frequency and voltage's direct effect on power), making power an area-dependent constraint.

Chakraborty considers device-scaling alone and estimates a simultaneous activity factor for technology nodes down to 32 nm [8]. Hempstead et al. introduce a variant of Amdahl's Law to estimate the amount of specialization required to maintain 1.5× performance growth per year, assuming completely parallelizable code [14]. Using ITRS projections, Venkatesh et al. estimate technology-imposed utilization limits and motivate energy-efficient and application- specific core designs [27]. Chung et al. study unconventional cores including custom logic, FPGAs, or GPUs in heterogeneous single-chip design [10]. They rely on Pollack's rule for the area/performance and power/performance tradeoffs. Using ITRS projections, they report on the potential for unconventional cores, considering parallel kernels.

Azizi et al. derive the energy/performance Pareto frontiers for single-core architectures using statistical architectural models combined with circuit-level energy-performance tradeoff functions [2]. Our core model derives these curves using measured data for real processors. Esmaeilzadeh et al. perform a power/energy Pareto efficiency analysis at 45 nm using total chip power measurements in the context of a retrospective workload analysis [12]. In contrast to the total chip power measurements, we use only the power budget allocated to the cores to derive the Pareto frontiers and combine those with our device and chip-level models to study the future of multicore design and the implications of technology scaling.

Previous work largely abstracts away processor organization and application details. This study considers the implications of process technology scaling, decouples power/area constraints, and considers multicore organizations, microarchitectural features, and real applications and their behavior.

9. CONCLUSIONS

For decades, Dennard scaling permitted more transistors, faster transistors, *and* more energy efficient transistors with each new process node, justifying the enormous costs required to develop each new process node. Dennard scaling's failure led the industry to race down the multicore path, which for some time permitted performance scaling for parallel and multitasked workloads, permitting the economics of process scaling to hold. But as the benefits of multicore scaling begin to ebb, a new driver of transistor utility must be found, or the economics of process scaling will break and Moore's Law will end well before we hit final manufacturing limits. An essential question is how much more performance can be extracted from the multicore path in the near future.

This paper combined technology scaling models, performance models, and empirical results from parallel workloads to answer that question and estimate the remaining performance available from multicore scaling. Using PARSEC benchmarks and ITRS scaling projections, this study predicts best-case average speedup of 7.9 times between now and 2024 at 8 nm. That result translates into a 16% annual performance gain, *for highly parallel workloads* and

assuming that each benchmark has its ideal number and granularity of cores.

However, we believe that the ITRS projections are much too optimistic, especially in the challenging sub-22 nanometer environment. The conservative model we use in this paper more closely tracks recent history. Applying these conservative scaling projections, half of that ideal gain vanishes; the path to 8nm in 2018 results in a best-case average 3.7× speedup; approximately 14% per year *for highly parallel codes* and *optimal per-benchmark configurations*. The returns will certainly be lower in practice.

Currently, the broader computing community is in consensus that we are in "the multicore era." Consensus is often dangerous, however. Given the low performance returns assuming conservative (and to some degree ITRS) scaling, adding more cores will not provide sufficient benefit to justify continued process scaling. If multicore scaling ceases to be the primary driver of performance gains at 16nm (in 2014) the "multicore era" will have lasted a mere nine years, a short-lived attempt to defeat the inexorable consequences of Dennard scaling's failure.

Clearly, architectures that move well past the Pareto-optimal frontier of energy/performance of today's designs will be necessary. Given the time-frame of this problem and its scale, radical or even incremental ideas simply cannot be developed along typical academic research and industry product cycles. On the other hand, left to the multicore path, we may hit a "transistor utility economics" wall in as few as three to five years, at which point Moore's Law may end, creating massive disruptions in our industry. Hitting a wall from one of these two directions appears inevitable. There is a silver lining for architects, however: At that point, the onus will be on computer architects–and computer architects only–to deliver performance and efficiency gains that can work across a wide range of problems. It promises to be an exciting time.

10. ACKNOWLEDGMENTS

We thank Shekhar Borkar for sharing his personal views on how CMOS devices are likely to scale. Support for this research was provided by NSF under the following grants: CCF-0845751, CCF-0917238, and CNS-0917213. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

References

[1] G. M. Amdahl. Validity of the single processor approach to achieving large-scale computing capabilities. In *AFIPS '67*.

[2] O. Azizi, A. Mahesri, B. C. Lee, S. J. Patel, and M. Horowitz. Energy-performance tradeoffs in processor architecture and circuit design: a marginal cost analysis. In *ISCA '10*.

[3] A. Bakhoda, G. L. Yuan, W. W. L. Fung, H. Wong, and T. M. Aamodt. Analyzing CUDA workloads using a detailed GPU simulator. In *ISPASS '09*.

[4] M. Bhadauria, V. Weaver, and S. McKee. Understanding PARSEC performance on contemporary CMPs. In *IISWC '09*.

[5] C. Bienia, S. Kumar, J. P. Singh, and K. Li. The PARSEC benchmark suite: Characterization and architectural implications. In *PACT '08*.

[6] S. Borkar. Thousand core chips: a technology perspective. In *DAC '07*.

[7] S. Borkar. The exascale challenge. Keynote at International Symposium on VLSI Design, Automation and Test (VLSI-DAT), 2010.

[8] K. Chakraborty. *Over-provisioned Multicore Systems*. PhD thesis, University of Wisconsin-Madison, 2008.

[9] S. Cho and R. Melhem. Corollaries to Amdahl's law for energy. *Computer Architecture Letters*, 7(1), January 2008.

[10] E. S. Chung, P. A. Milder, J. C. Hoe, and K. Mai. Single-chip heterogeneous computing: Does the future include custom logic, FPGAs, and GPUs? In *MICRO '10*.

[11] R. H. Dennard, F. H. Gaensslen, V. L. Rideout, E. Bassous, and A. R. LeBlanc. Design of ion-implanted mosfet's with very small physical dimensions. *IEEE Journal of Solid-State Circuits*, 9, October 1974.

[12] H. Esmaeilzadeh, T. Cao, Y. Xi, S. M. Blackburn, and K. S. McKinley. Looking back on the language and hardware revolutions: measured power, performance, and scaling. In *ASPLOS '11*.

[13] Z. Guz, E. Bolotin, I. Keidar, A. Kolodny, A. Mendelson, and U. C. Weiser. Many-core vs. many-thread machines: Stay away from the valley. *IEEE Computer Architecture Letters*, 8, January 2009.

[14] M. Hempstead, G.-Y. Wei, and D. Brooks. Navigo: An early-stage model to study power-contrained architectures and specialization. In *MoBS '09*.

[15] M. D. Hill and M. R. Marty. Amdahl's law in the multicore era. *Computer*, 41(7), July 2008.

[16] M. Horowitz, E. Alon, D. Patil, S. Naffziger, R. Kumar, and K. Bernstein. Scaling, power, and the future of CMOS. In *IEDM '05*.

[17] E. Ipek, M. Kirman, N. Kirman, and J. F. Martinez. Core fusion: accommodating software diversity in chip multiprocessors. In *ISCA '07*.

[18] ITRS. International technology roadmap for semiconductors, 2010 update, 2011. URL `http://www.itrs.net`.

[19] C. Kim, S. Sethumadhavan, M. S. Govindan, N. Ranganathan, D. Gulati, D. Burger, and S. W. Keckler. Composable lightweight processors. In *MICRO '07*.

[20] J.-G. Lee, E. Jung, and W. Shin. An asymptotic performance/energy analysis and optimization of multi-core architectures. In *ICDCN '09*, .

[21] V. W. Lee et al. Debunking the 100X GPU vs. CPU myth: an evaluation of throughput computing on CPU and GPU. In *ISCA '10*, .

[22] G. Loh. The cost of uncore in throughput-oriented many-core processors. In *ALTA '08*.

[23] G. E. Moore. Cramming more components onto integrated circuits. *Electronics*, 38(8), April 1965.

[24] K. Nose and T. Sakurai. Optimization of VDD and VTH for low-power and high speed applications. In *ASP-DAC '00*.

[25] SPEC. Standard performance evaluation corporation, 2011. URL `http://www.spec.org`.

[26] A. M. Suleman, O. Mutlu, M. K. Qureshi, and Y. N. Patt. Accelerating critical section execution with asymmetric multi-core architectures. In *ASPLOS '09*.

[27] G. Venkatesh, J. Sampson, N. Goulding, S. Garcia, V. Bryksin, J. Lugo-Martinez, S. Swanson, and M. B. Taylor. Conservation cores: reducing the energy of mature computations. In *ASPLOS '10*.

[28] D. H. Woo and H.-H. S. Lee. Extending Amdahl's law for energy-efficient computing in the many-core era. *Computer*, 41(12), December 2008.