Dark Silicon: The Beginning of the End

Hailang Liou, Edric Kusuma
March 5, 2020

Amdahl's Law in the Multicore Era

Motivations

How should we scale multi-core designs? More cores or better cores?

Develop a simple analytical model to capture general trends of approaches.

Provide an “upper bound” on performance, without getting bogged down by μ-arch.

What can we learn by applying Amdahl’s Law to these models?

What direction should future research and architectures move in based on this paper’s results?

Amdahl’s Law

I hope you know this by now, *but just in case...*

$$\text{Speedup}_{\text{parallel}}(f,n) = \frac{1}{(1-f) + \frac{f}{n}}$$

The Simple Model

Maximum budget of *n* “base core equivalents” (BCE), representative of the complexity of some baseline core. Budget acts as a proxy for die area

Create a multiprocessor chip with *x* cores, each with the power of *y* BCEs.

Explore this design space! How many cores, and are they all the same? More on this later...

Assume single-core serial performance scales by $\sqrt{y}$ (*Pollack’s rule*) to create a tradeoff between sequential vs parallel execution.

Symmetric Multicore Chips

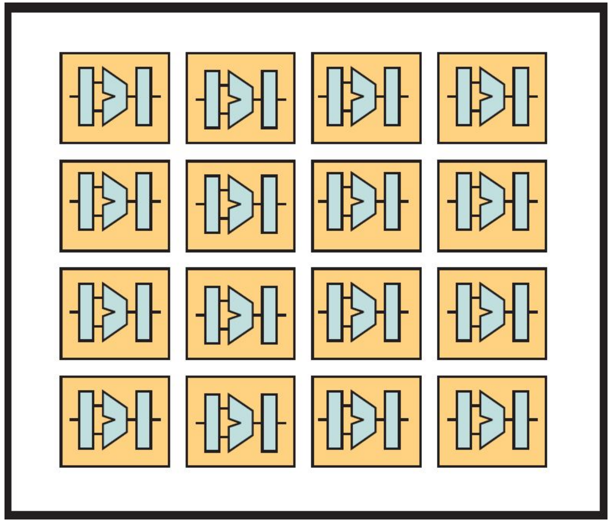
Each core is the same size, so split the total resources available *evenly*.

Sequential component executed on one of the cores, while parallel execution is done on *all* cores.

Amdahl's law gives:

$$\text{Speedup}_{\text{symmetric}}(f, n, r) = \frac{1}{\frac{1-f}{perf(r)} + \frac{f \cdot r}{perf(r) \cdot n}}$$

When would it be beneficial to increase core complexity?

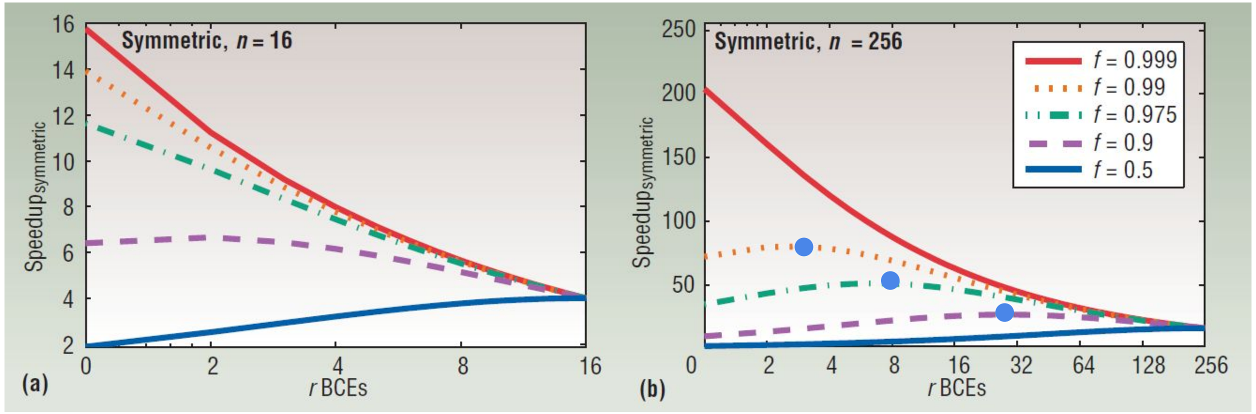


Symmetric Multicore Takeaways

More *f* gives more speedup by augmenting the effect of having more cores.

As *f* approaches 1, the ideal core size becomes minimal… but *f* is usually not 1.

The more compute resources there are, the optimal size (●) becomes larger.



Asymmetric Multicore Chips

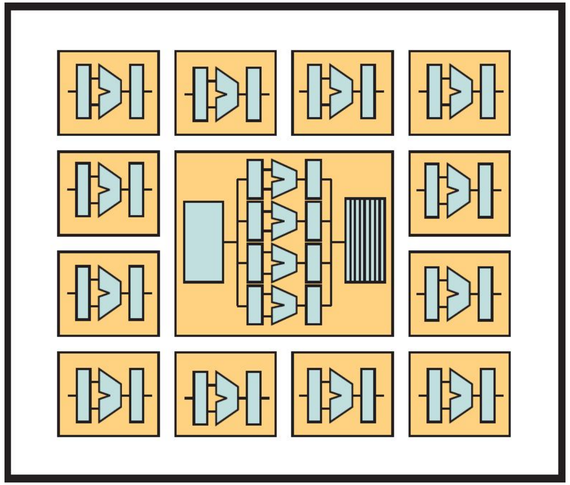
Similar to “big little” architectures

One big core of *r* BCEs and *n-r* little cores of 1 BCE each

Sequential portion runs on big core, parallel portion runs on all cores

Amdahl’s law gives:

$$\text{Speedup}_{\text{asymmetric}}(f,n,r) = \frac{1}{\frac{1-f}{perf(r)} + \frac{f}{perf(r)+n-r}}$$



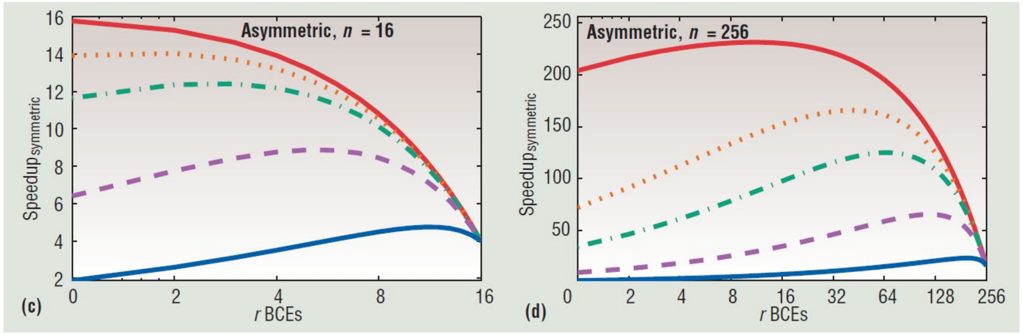
Asymmetric Multicore Takeaways

Strictly higher upper-bound on speedup compared to symmetric chips

With more area available, the optimal size and speedup of the big core increases

Amdahl's law -- make the sequential bottleneck fast with high parallelism

What are some challenges when coordinating cores with workloads that have differing amounts of parallelism?



Dynamic Multicore Chips

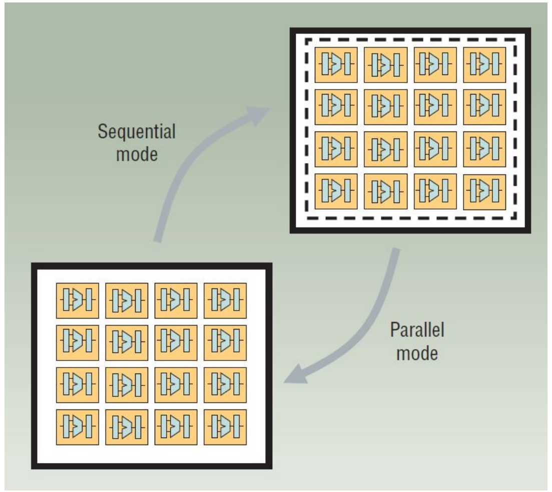
Not real, idealized multiprocessor. *How idealized?*

Freely combine resources to create a composite “big” core to run serial portion

Parallel portion uses all resources as minimal sized parallel cores

Amdahl’s law gives:

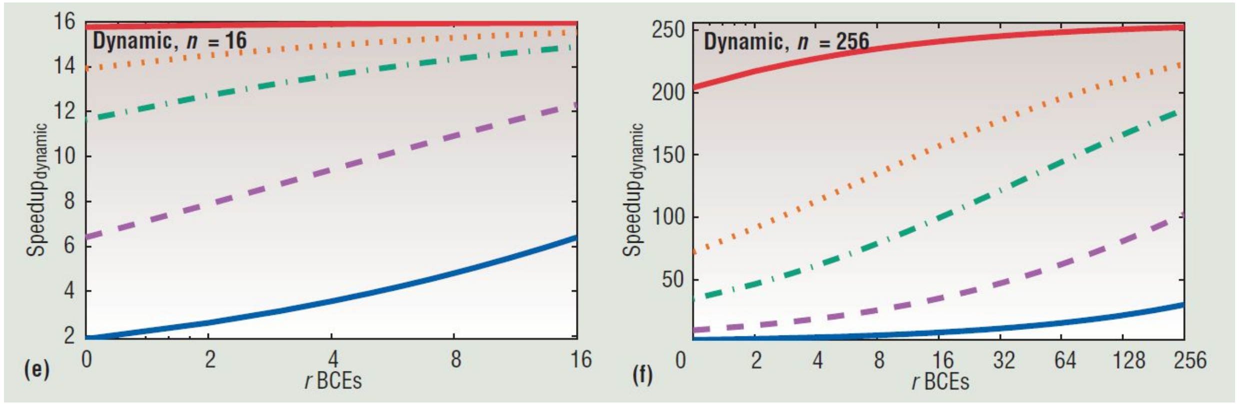
$$\text{Speedup}_{\text{dynamic}}(f, n, r) = \frac{1}{\frac{1-f}{perf(r)} + \frac{f}{n}}$$



Dynamic Multicore Takeaways

Seems to be perfectly optimal design

- Authors suggest exploring approximations for dynamic multicore
- Is it worth it?



Final thoughts

Model may be simple and unrealistic but authors claim purpose is to “make people think” and “question assumptions”

Can we expand this model?

Does ignoring other aspects of architecture affect the conclusions that can be drawn?

Dark Silicon and the End of Multicore Scaling

“Dark Silicon”

Before 2006, transistor scaling (Moore’s Law) has mostly been followed by voltage scaling (Dennard scaling).

Around 2006, Dennard scaling failed such that it cannot follow Moore’s Law.

The extra transistors brought by Moore’s Law can no longer be powered on because it would violate the *thermal design power* (TDP) constraint 🔥

These unpowered/unused transistors are “**dark silicon**”.

Motivations

How much farther can multicore scaling take us?

What limits multicore scaling?

Take the models from the previous paper… and explore them rigorously

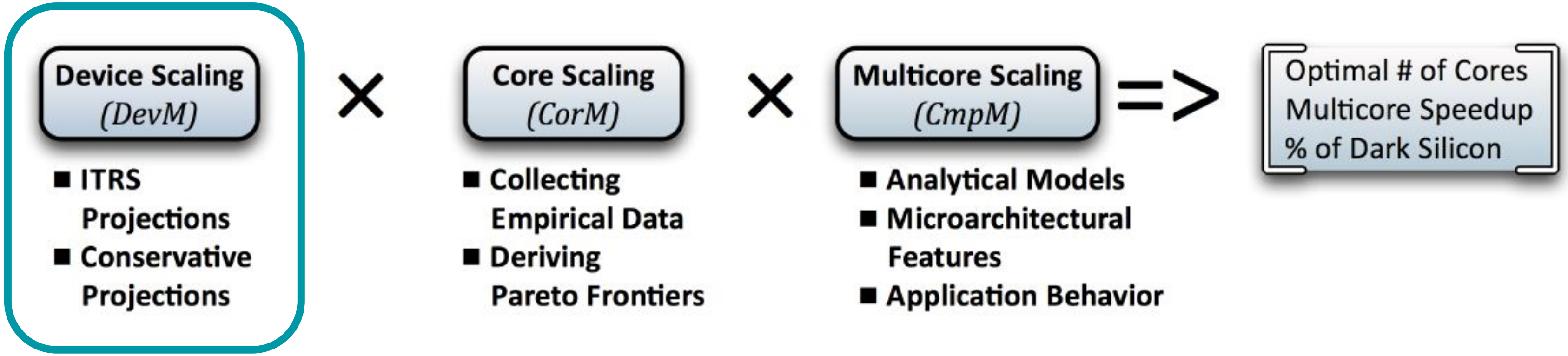
Use existing microarchitectures and future projections to assess scalability

How does silicon at nanometer scale process nodes affect scaling?

Models and Methodology: Device Scaling

How do transistors behave as the process node shrinks?

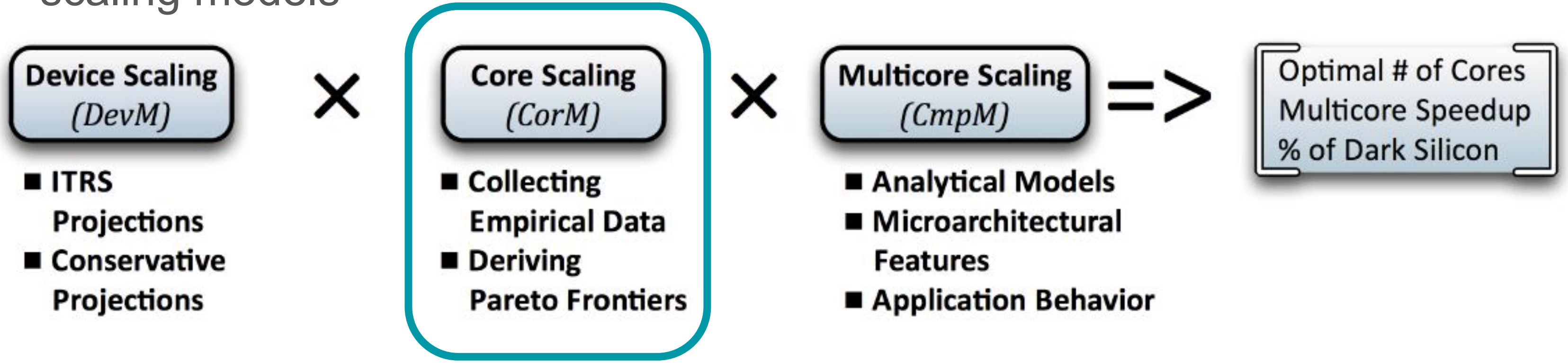
- ITRS projections on behavior from 45nm down to 8nm
- "Conservative Projections" assume much less performance increase and power decrease per node



Models and Methodology: Core Scaling

What does a single core look like at each process node?

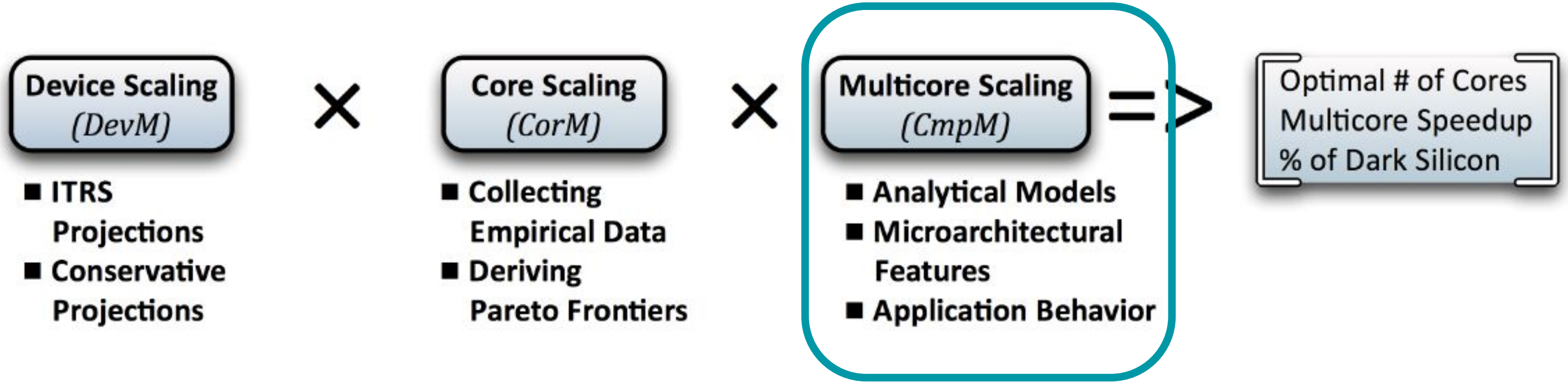
- Collect performance (SPECmark), power (TDP), and area (die shots) metrics fro, a wide variety of processors.
- Generate Pareto frontiers at the existing process node using the data
- Can derive theoretical Pareto frontiers at future process nodes using device scaling models



Models and Methodology: Multicore Scaling

How can performance from one core to many?

- Using the simple model in the other paper, an upper-bound for multicore speedup can be determined, $CmpM_U$
- By factoring in more practical parameters like cache access times, bandwidth, etc, a more realistic model can be created, $CmpM_R$



Models and Methodology: Multicore Scaling

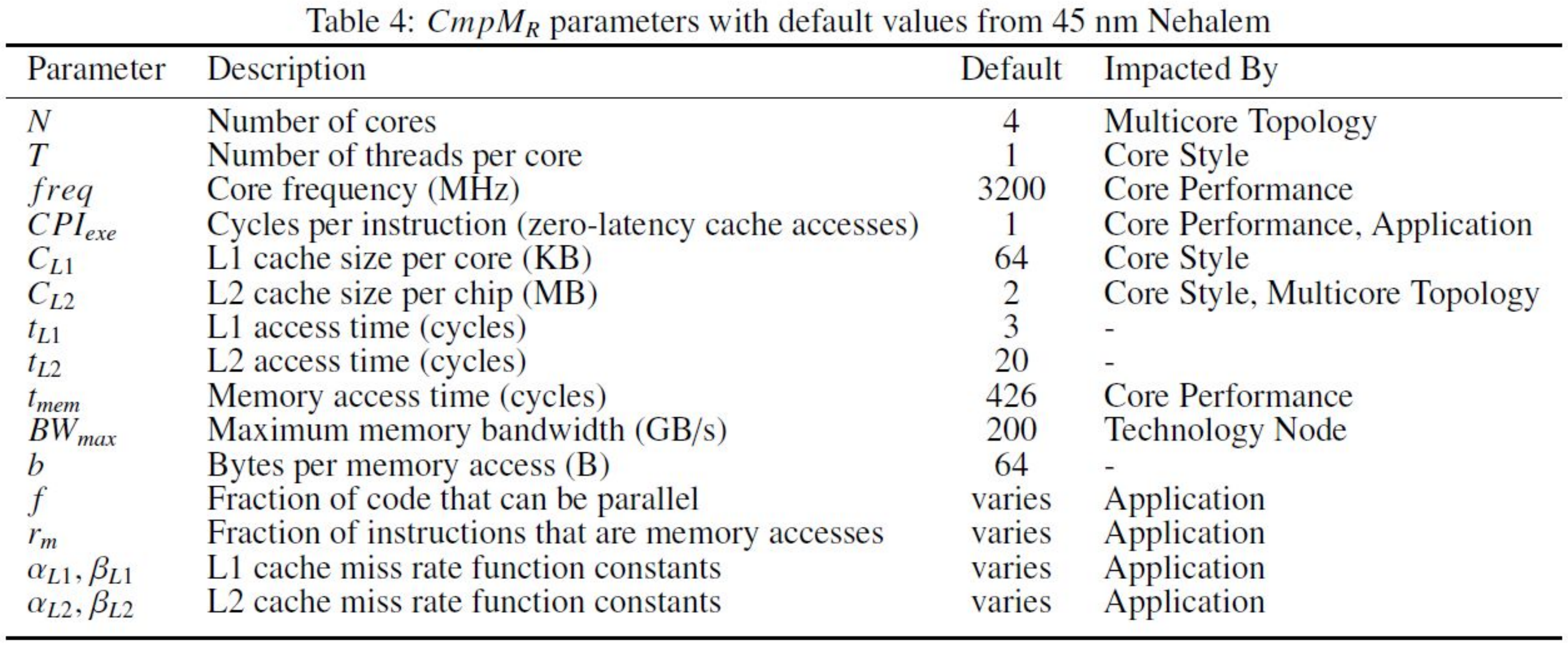


Table 4: $CmpM_R$ parameters with default values from 45 nm Nehalem

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

Putting it all together: Device x Core x CMP

Produce “projections for optimal performance, number of cores, and amount of dark silicon” using the three models.

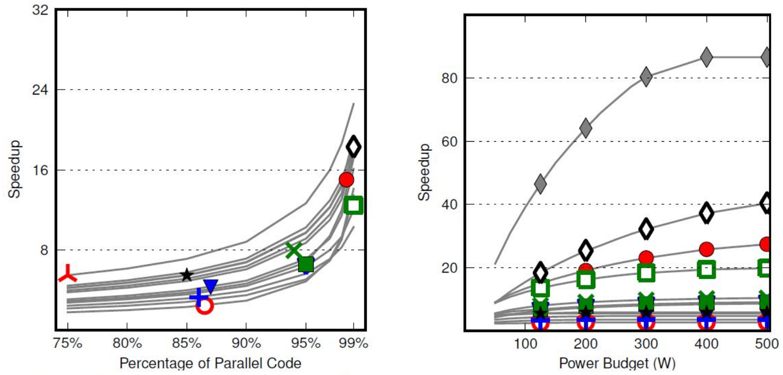
1. Examine the design of a processor that is along the Pareto frontier.
2. Gradually add a core to the system, and compute the speedup (either upper-bound or realistic) with these parameters.
3. Stop after some number of iterations, when an area or power limit is hit, or noticeable performance degradation occurs. This is the design with the optimal speedup and number of cores. Amount of dark silicon can be determined.
4. Repeat steps for all designs on the Pareto frontier.

Important Results

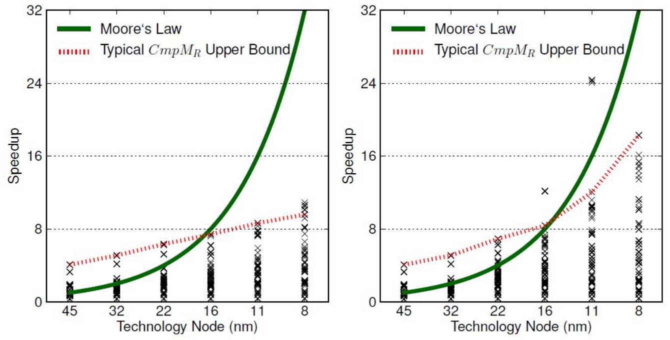
Where is dark silicon coming from?

- Could be either power, or parallelism
- Figures imply parallelism the primary contributor

When comparing with expected speedup as implied by Moore's Law, dark silicon is shown to be a problem.



(a) Parallelism (f actual at marker) (b) Power



(a) Conservative Scaling (b) ITRS Scaling

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