Amdahl's Law in the Multicore Era

Mark D. Hill and Michael R. Marty

- Varun Shankar

Goal of the paper

- Adapt Amdahl's law to take into account multicore 'revolution'.

- Determine how to distribute limited resources among many cores 'optimally'.

- Elicit discussion on future research directions, hopefully elicit research.

Remember...

- Amdahl's Law for speedup S (S single core processors)

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

Amdahl's law applies broadly and has important corollaries such as:

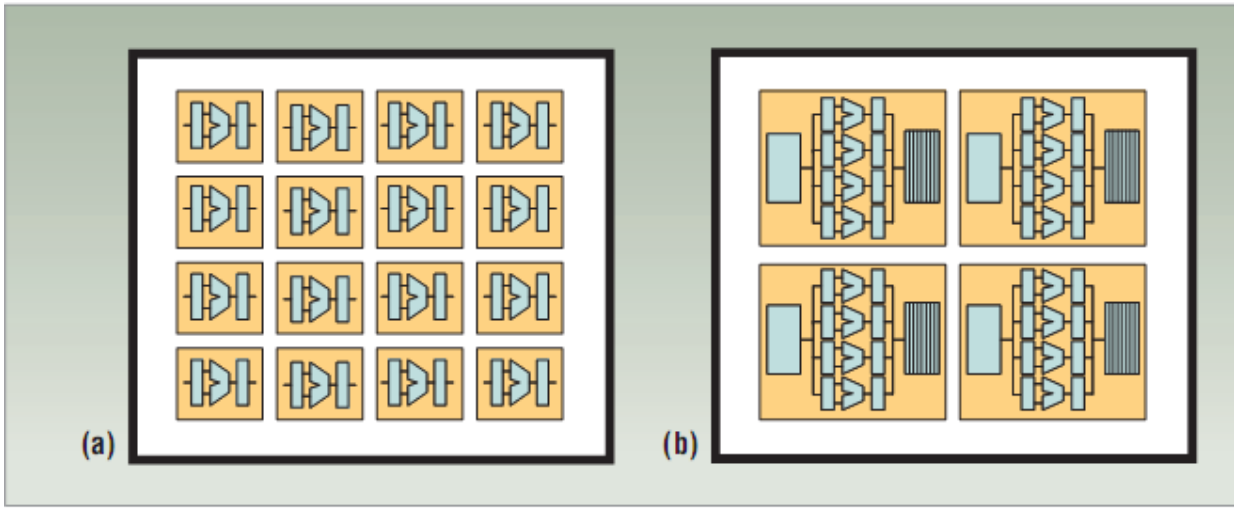
- Attack the common case: When f is small, optimizations will have little effect.
- The aspects you ignore also limit speedup: As S approaches infinity, speedup is bound by $1/(1 - f)$.

Things to keep in mind

- This paper fixes total cost (resources) and chooses how to spend that cost on the available cores.
- BCE: Generic Unit of Cost- could be area, power etc., or a combination of these factors.
- Totally n BCEs available on one chip. Expending r BCEs on a core results in sequential performance perf(r).
- Paper picks perf(r) = sqrt(r).
- Now, how do we distribute our n BCEs?

Symmetric multicore chips

- Each core must use the same number of BCEs.
- Say n=16 BCEs. Then, we can have n/r cores of r BCEs each (16 of 1 each or 4 of 4 each).

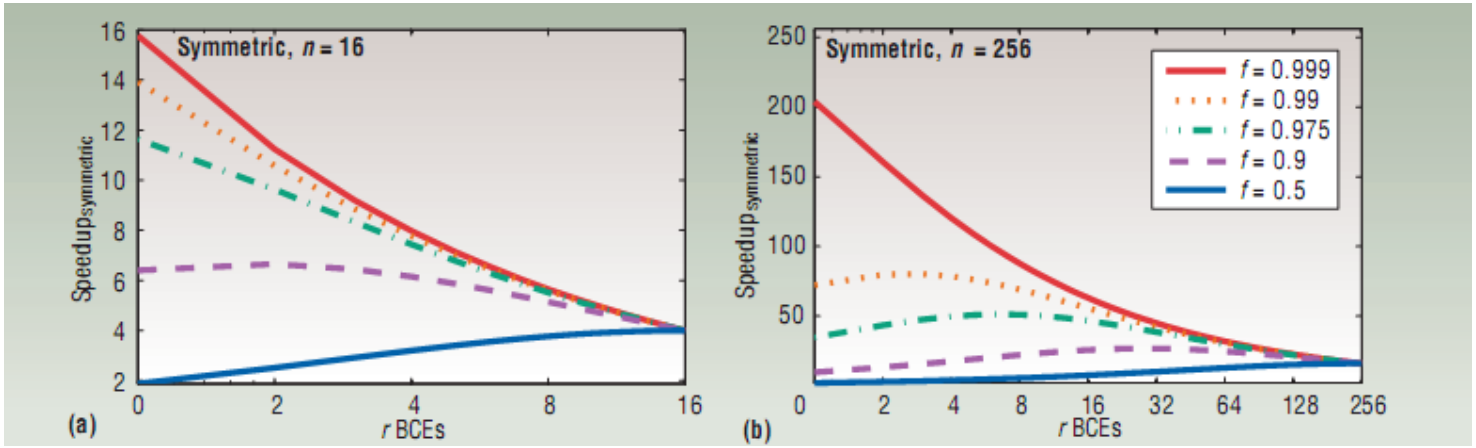


Symmetric multicore chips

- Serial Fraction 1-f uses 1 core at rate perf(r)
- Serial time = (1 – f) / perf(r)
- Parallel Fraction uses n/r cores at rate perf(r) each
- Parallel time = f / (perf(r) * (n/r)) = f*r / perf(r)*n
- Then, we have first modification to Amdahl's Law:

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

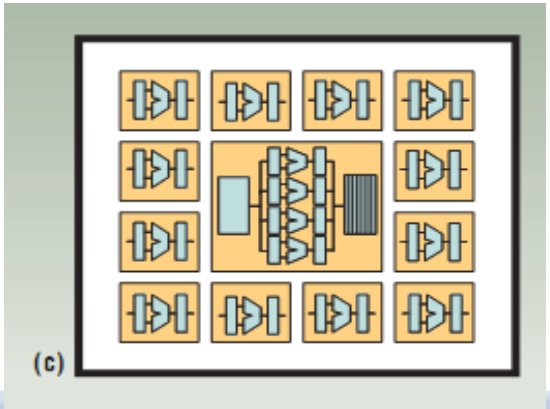
Symmetric multicore chips



- Fraction f should be as high as possible(just as followed from traditional Amdahl's Law).
- Having r>1 BCEs per core can be beneficial (for n=256, f=0.975, maximum speedup at 7.1 BCEs per core).

Asymmetric multicore chips

- Some cores more powerful than others (paper studies the case of 1 core more powerful than the others).
- If one core is larger (more BCEs) and uses r BCEs, it leaves n-r BCEs for the others to use. Chip can therefore have 1+n-r cores. For n=16, as before, we could have a 4 BCE core and 12 1-BCE cores.
- How does this stack up against the symmetric distribution?

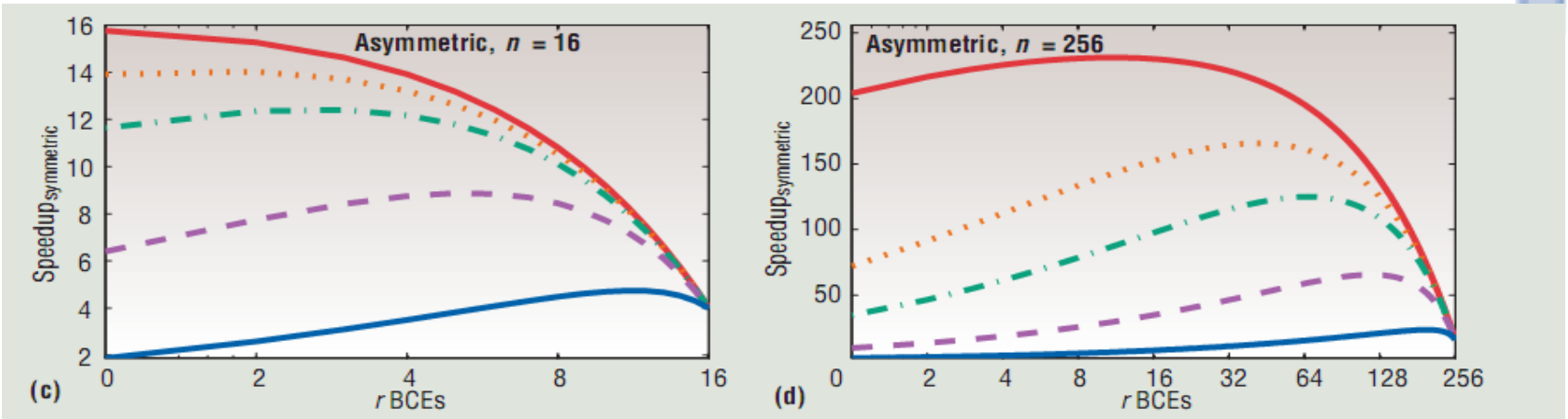


Asymmetric multicore chips

- Serial time = (1 – f) / perf(r), as before.
- In parallel, 1 core at rate perf(r), n-r cores at rate 1.
- Parallel time = f/(perf(r) + n-r)

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

Asymmetric multicore chips



- Asymmetric chips can be as good as or much better than symmetric (look at n=256 and f=0.975).
- Denser chips can increase both the benefit of asymmetric chips *and* the optimal performance of the large core (look at n=1024 and f =0.975).
- So investigate even locally inefficient sequential speedup factors (can reduce phase when other processors are idle).

Dynamic multicore chips

- Dynamically combine r cores into 1 core to boost sequential performance. In sequential mode, get perf(r).
- In parallel mode, get performance of n using all base cores in parallel.
- Better than asymmetric?

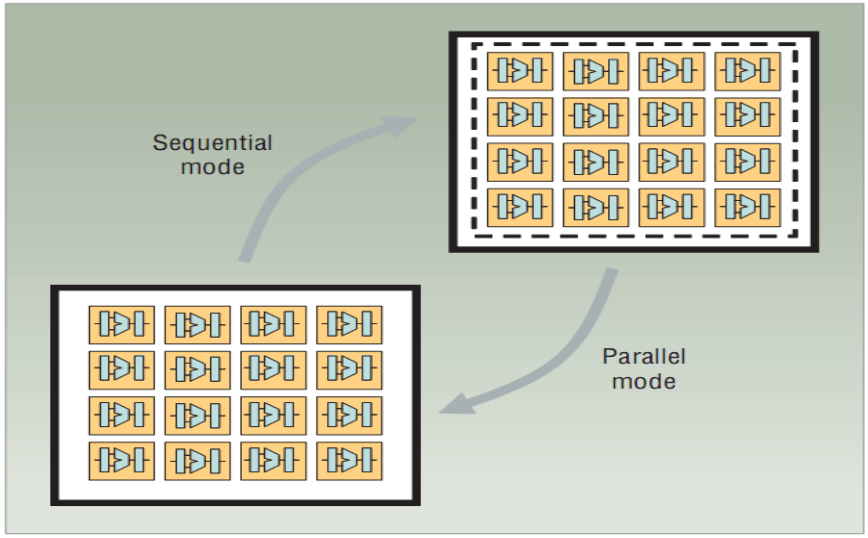


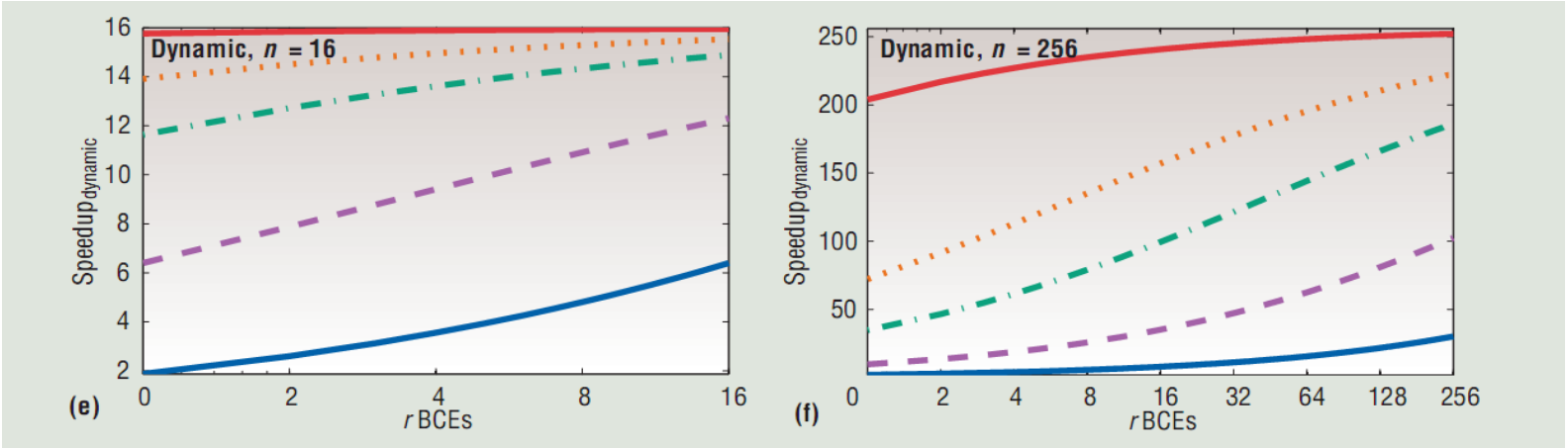
Figure 3. Dynamic multicore chip with 16 one-BCE cores.

Dynamic multicore chips

- Serial time = (1 – f) / perf(r), as before.
- In parallel, n cores at rate 1.
- Parallel time = f/(n)

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

Dynamic multicore chips



- Dynamic chips(of the future) can as good or better than the asymmetric case for large f, provided switching between serial and parallel is very fast.
- Look at f=0.99 and n=256, the speedup is 233 if all cores are harnessed(difficult to achieve in practice, but considerably faster than asymmetric).
- So investigate dynamic harnessing techniques- thread-level speculation, for example.

Summing up...

- Serial fraction and parallel fraction are not entirely serial and parallel. Corollaries do not take this into account.
- Memory system design and interconnect are not explored.
- Still do not know how to dynamically gang up cores in a reasonable way (Eg. Microsoft research's E2 is an attempt in this direction)
- Scheduling tasks on non-symmetric systems may be difficult.
- But authors did manage to write corollaries to Amdahl's Law that can point us in the right way. Good first step.

Fin