Luca Tornatore - I.N.A.F.

“**Foundation of HPC” course**

DATA SCIENCE &
SCIENTIFIC COMPUTING
2021-2022 @ Università di Trieste

The end of “Free lunch” era

Why your life would have been easier 30 years ago and why it is not so

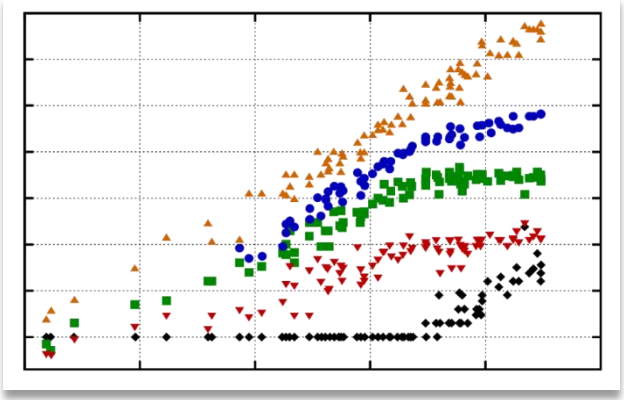
The deep roots of the shift in computer architecture

The “free lunch”

From ‘60s to mid of ‘90s the performance gain of a software was due essentially to the **constant increase** of:

1. the **clock** speed of the CPUs
2. the “**capability**”of CPUs

The "Free lunch" (*)



" *Moore's law* " predicted exponential growth of transistor density on chips:

every 18 months the density of transistors will double *at the same manufacturing cost* (*).

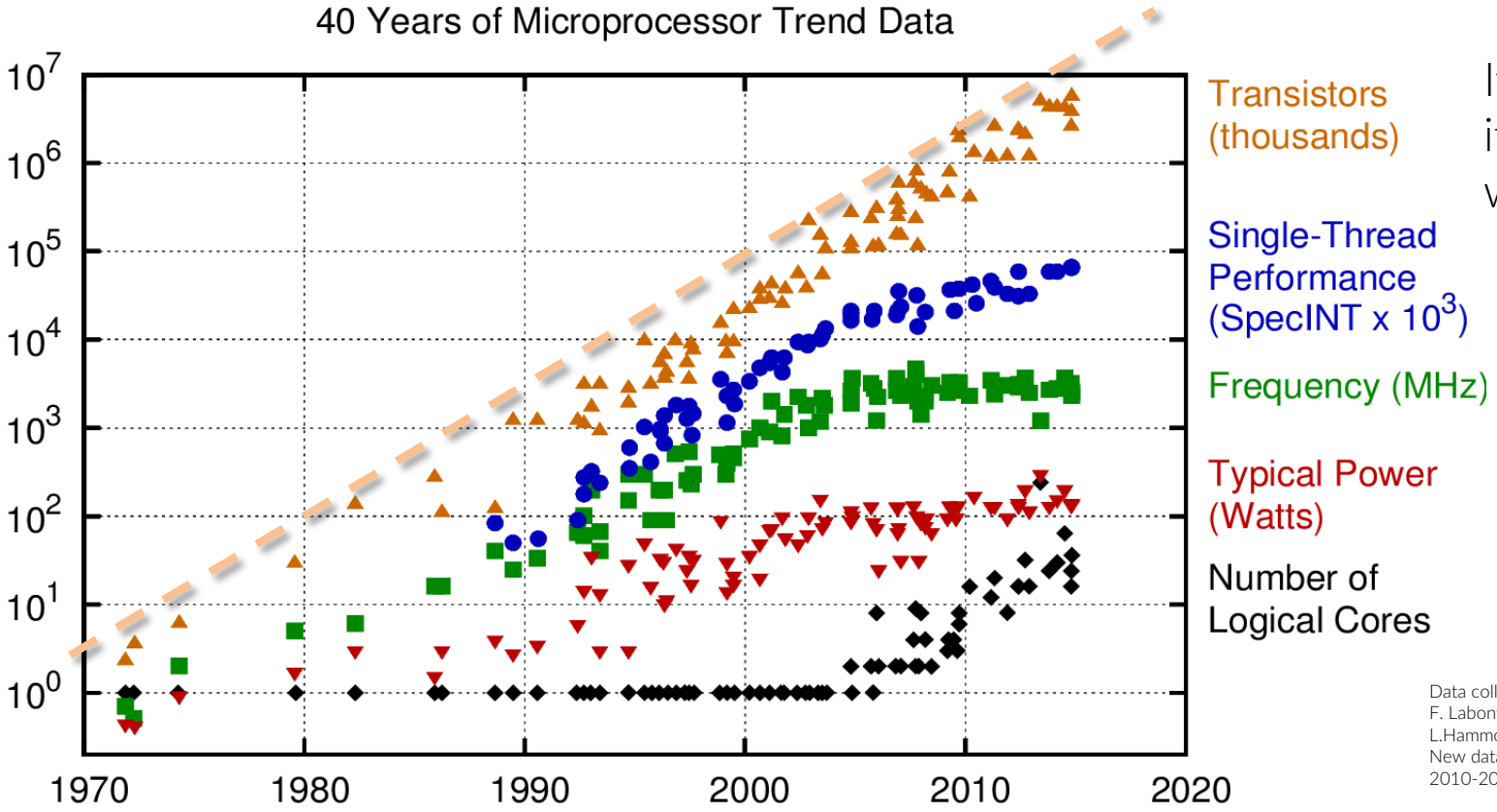
meaning that every 18 months you could buy a commodity CPUs with 2× logics than the previous generation, at the same cost

(*) there have been even faster growths in other fields, for instance in data storage density

All exponential growths get to their end..

(*) from an article by H. Sutter in *Dr. Dobb's Journal*, 2005

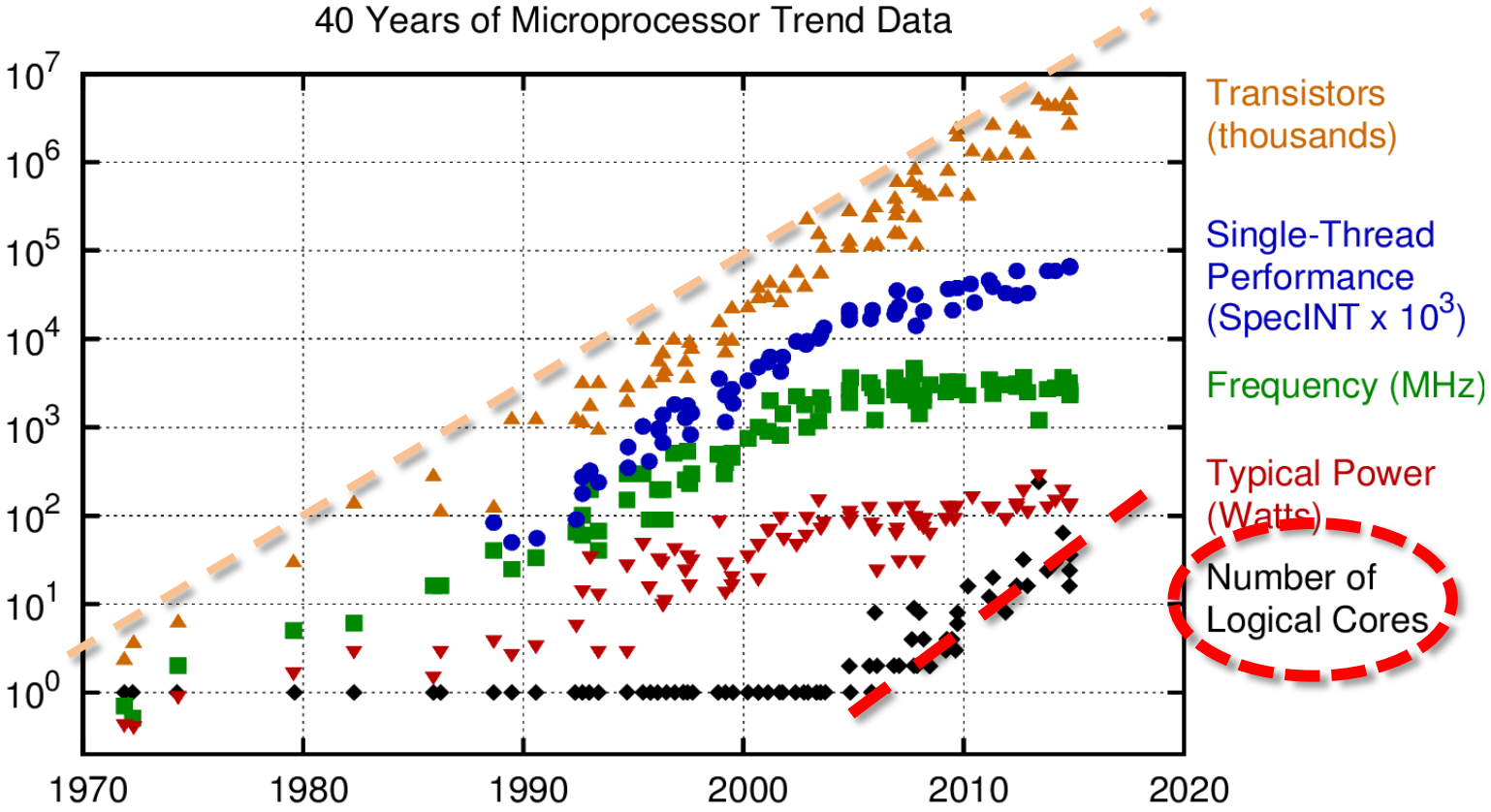
Moore's law, is it over ?



It seems that it is still working

Data collected up to 2010 by M.Horowitz, F. Labonte, O. Schacham, K. Olukotun, L.Hammond and C. Batlen.
New data up to 2015 collected by 2010-2015 by K. Rupp.

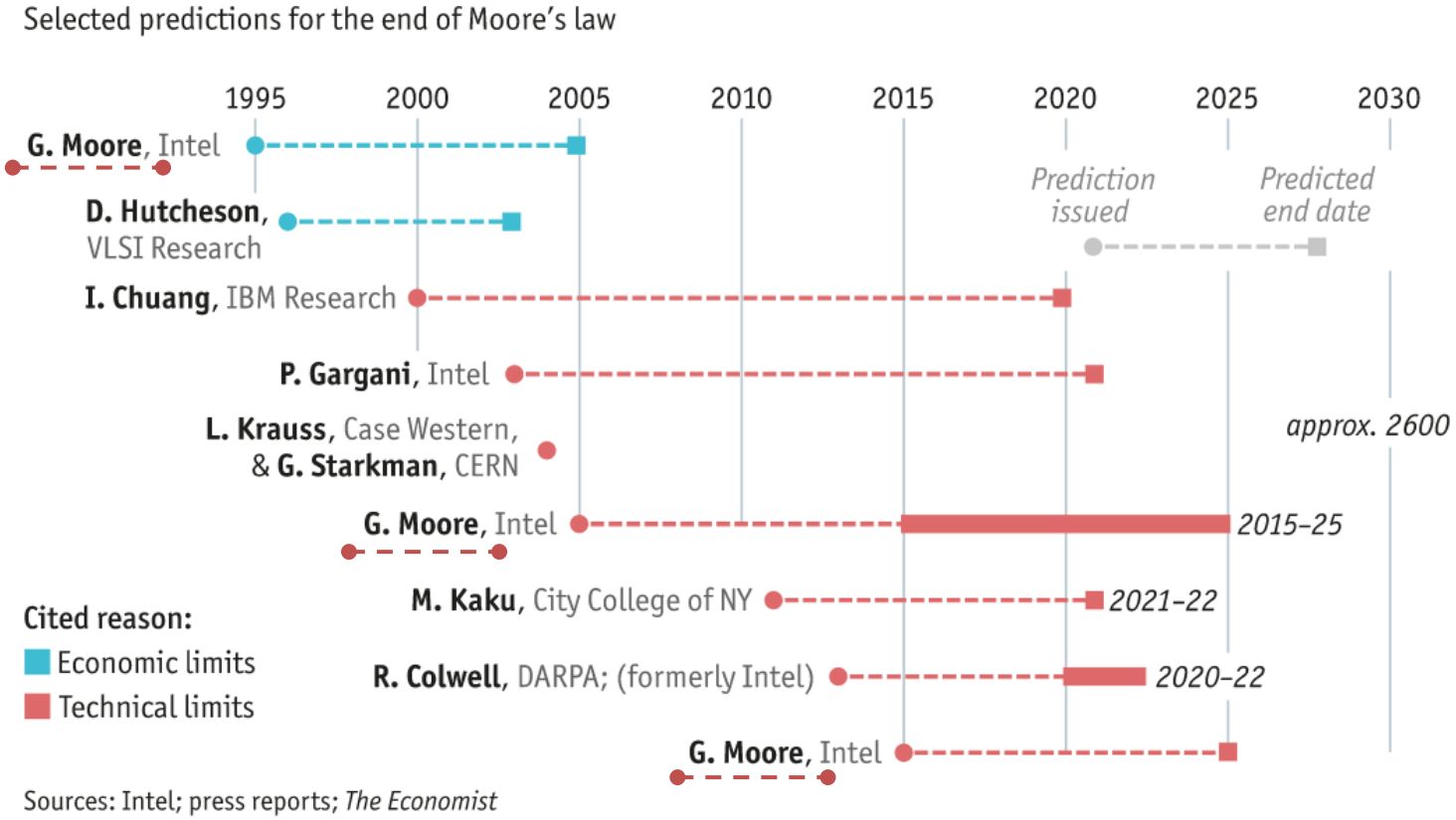
Moore's law, is it over ?



Moore's law, is it over ?

Not yet.

But there have often been rumours about that in the past



GOOD NEWS:
processors has continued to become “more powerful”

OTHER NEWS:
They are “differently” more powerful

The transition from the “free lunch”

Until half of the ‘00s, engineers succeeded in gaining performance by essentially 3 ways:

1. Increasing **clock** speed
2. Optimizing **execution**
3. Enlarging/improving **cache**

The transition from the "free lunch"

Until half of the '00s, engineers succeeded in gaining performance by essentially 3 ways:

1. Increasing **clock** speed →
2. Optimizing **execution**
3. Enlarging/improving **cache**

You get more cycles per unit time; more or less that means doing the same bunch of instructions faster

The transition from the “free lunch”

Until half of the ‘00s, engineers
gaining performance by essent

1. Increasing **clock** speed
2. Optimizing **execution** →
3. Enlarging/improving **cache**

- More powerful instructions
- Pipelining
- Branch predictions
- Out-of-order execution
- …

Under enormous pressure, CPUs manufacturers risked (and did) to break the semantic of your code. Or introduce horrible bugs..
(*have you heard about **Meltdown** and **Spectre** ?)*

The transition from the "free lunch"

Until half of the '00s, engineers
gaining performance by essent

1. Increasing **clock** speed
2. Optimizing **execution** →

Core 1	Core 2
`mov [X], value`	`mov [Y], value`
`mov reg1, [Y]`	`mov reg2, [X]`

- More powerful instructions
- Pipelining
- Branch predictions
- Out-of-order execution
- …

Under enormous pressure, CPUs manufacturers risked (and did) to break the semantic of your code. Or introduce horrible bugs..
(*have you heard about* ***Meltdown*** *and* ***Spectre*** *?)*

The transition from the “free lunch”

Until half of the ‘00s, engineers succeeded in gaining performance by essentially 3 ways:

1. Increasing **clock** speed
2. Optimizing **execution**
3. Enlarging/improving **cache** → More on that later..

Why there is no more “free lunch”?

Applications no longer get more performance for free without significant re-design, since $\gtrsim$15 years

Since 15 years, the gain in performance is essentially due to **fundamentally different factors**:

1. Multi-core + **Multi-threads**
2. Enlarging/improving **cache**
3. Hyperthreading *(smaller contribution)*

Why there is no more “free lunch”?

For instance:

2 Cores at 3GHz are basically 1 Core at 6GHz.. ?

False

- × Cores coordination for cache-coherence
- × Threads coordination
- × Memory access
- × Increased algorithmic complexity

Since 15 years, the gain in performance is essentially due to **fundamentally different factors**:

1. Multi-core + **Multi-threads**
2. Enlarging/improving **cache**
3. Hyperthreading *(smaller contribution)*

Why there is no more “free lunch”?

Moore’s law	Dennard’s scaling (MOSFET)
Manufacturing cost/area is constant while the transistors’ dimension halves every ~2 years → The number of transistors doubles in a CPU every ~2 years	- *voltage*, *capacitance*, *current* scale with λ - Transistor power scales as λ^2 → Power density remains constant

Power consumption:

$$P \propto C \cdot V^2 \cdot f$$

Why there is no more “free lunch”?

$$P \propto C \cdot V^2 \cdot f$$

C is the capacitance, scales as the area
V is the voltage, scales as the linear dimension
f is the frequency

so, the linear size of transistors shrinks and so do the voltage; if the area remains equal (more transistor on the same die), then the frequency can become larger

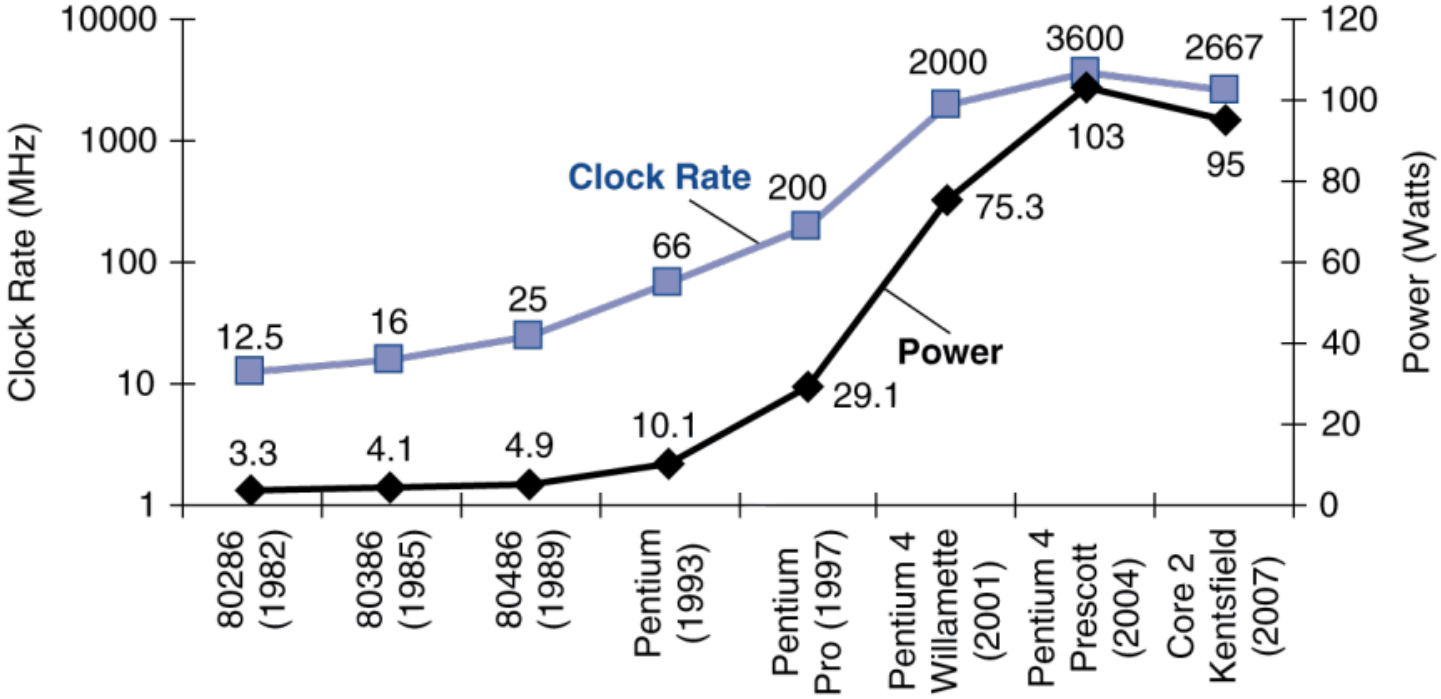
Why there is no more “free lunch”?

$$P \propto C \cdot V^2 \cdot f$$

× ~ 1000

× ~ 30

5V → 1V



Why there is no more “free lunch”?

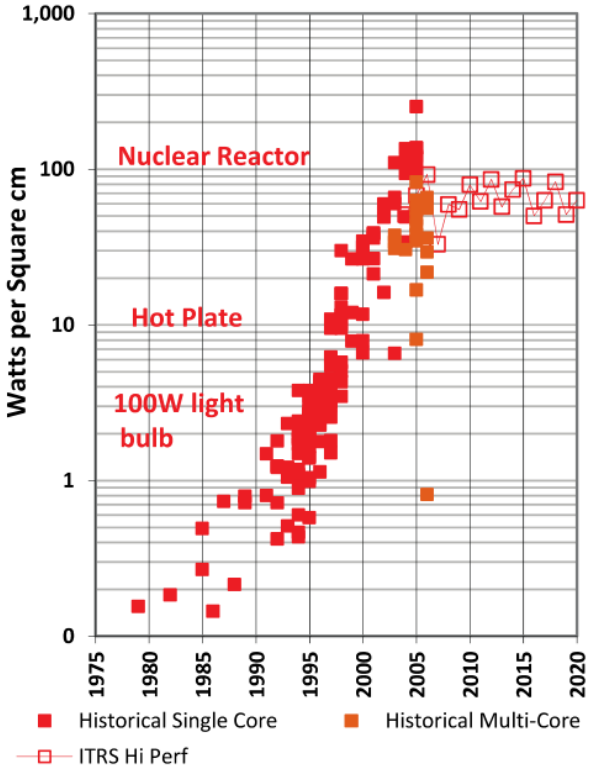
In summary, due essentially to quantum effects

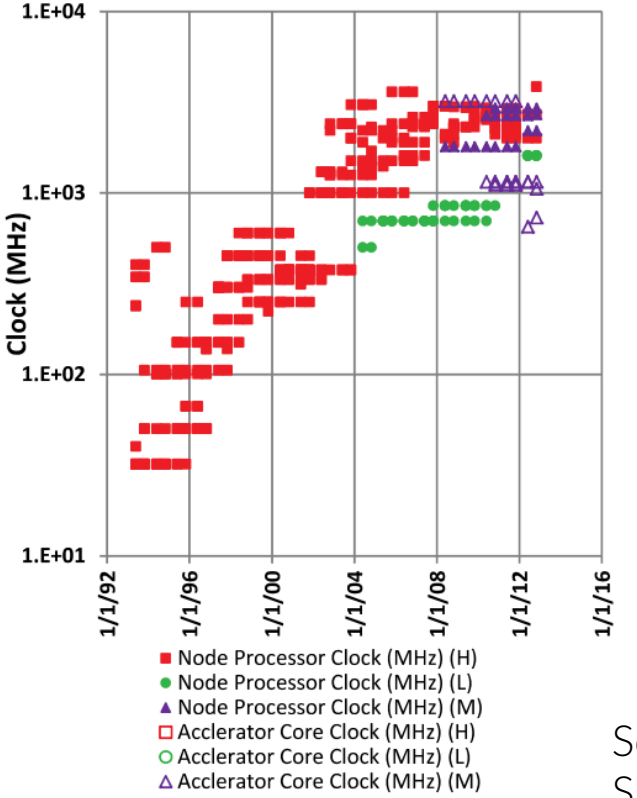
- Leakage current
- Threshold voltage
- Physical limits ad atomic scales

the *Dennard’s scaling is broken*, while the Moore’s law could still work (for a while at least).
So, as transistors get smaller **the power density actually increases**.
The result is a **“power wall”** that prevented the clock frequency to increase beyond ~3GHz since ~12 years

A dramatic change in the technological paradigm would be needed to revert the trend.

Why there is no more "free lunch"?





Source: Kogge and Shalf, IEEE CISE

"Free lunch" is over

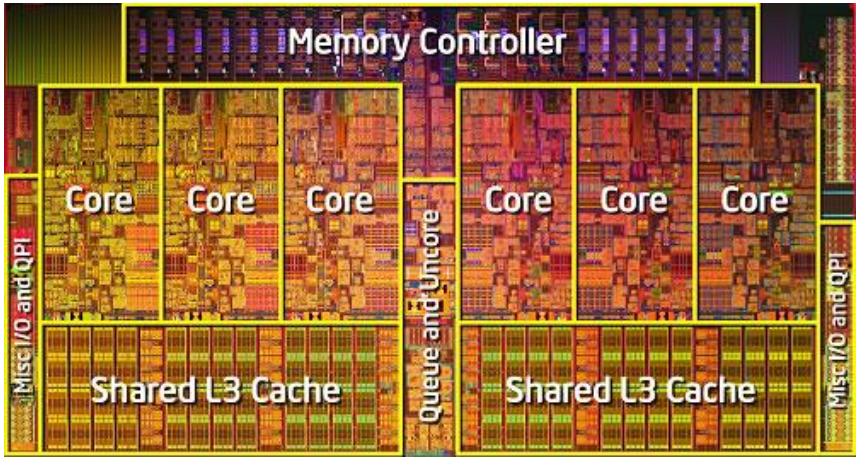
In the Top 500 the performance is missing, when it comes to consider "real" applications

	Rpeak (Pflop/s)	HPL / Rpeak	HPCG / Rpeak
FUGAKU	514	0.8	0.026
SUMMIT	201	0.74	0.015
SIERRA	126	0.75	0.014
Sunway	125	0.74	0.0038

From the top500 ranking, Fall 2019

Back to the future

Take-home message
Many-cores CPUs are here to stay



- Concurrency-based model programming
  (different than both *parallel* and *ILP*): work subdivision in as many independent tasks as possible
- Specialized, heterogeneous cores
- Multiple memory hierarchies

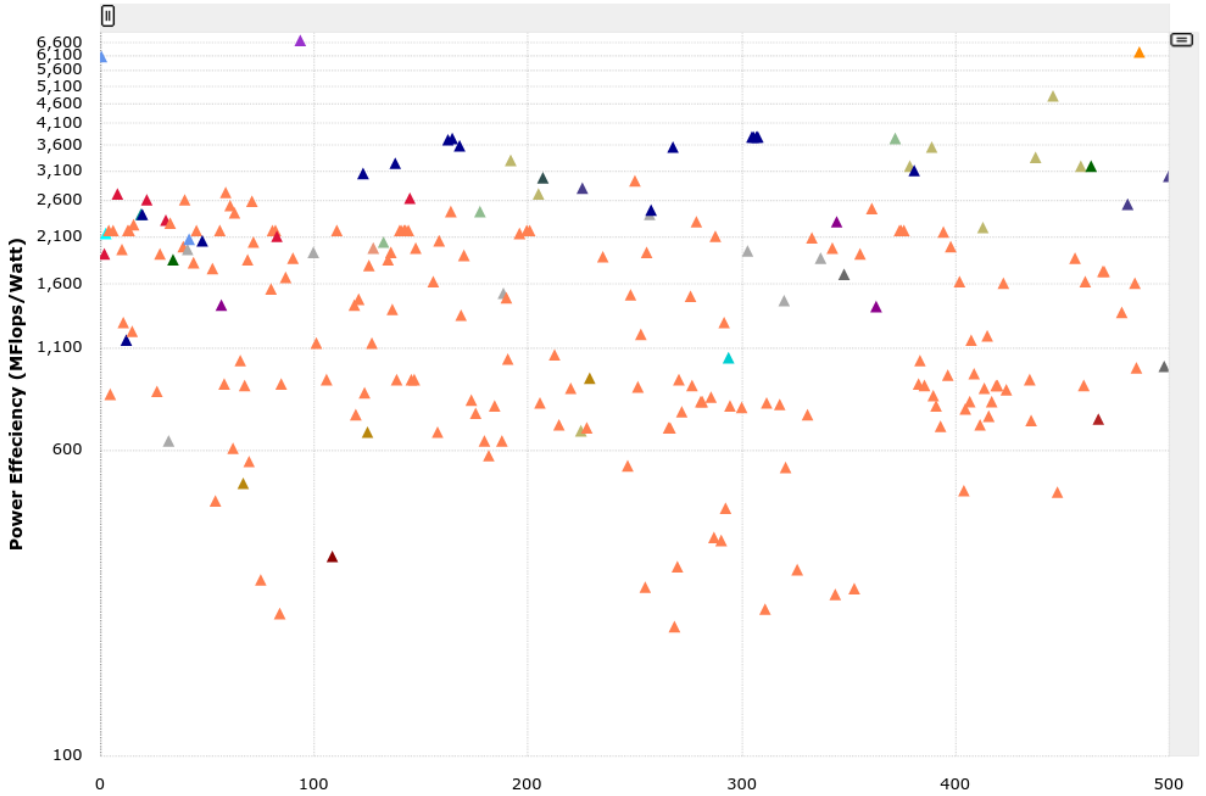
The Energy Challenge

Sunway performs for some apps at ≈ 10 Pflop/s consuming ≈ 18MW.

Simply rescaling to Eflop/s, it would consume ≈1.8GW.

The exa-scale goal is to reach Eflop/s at 20MW of electric power, i.e. 50 Gflops/W

Rule of thumb: 1MW = $1M / yr

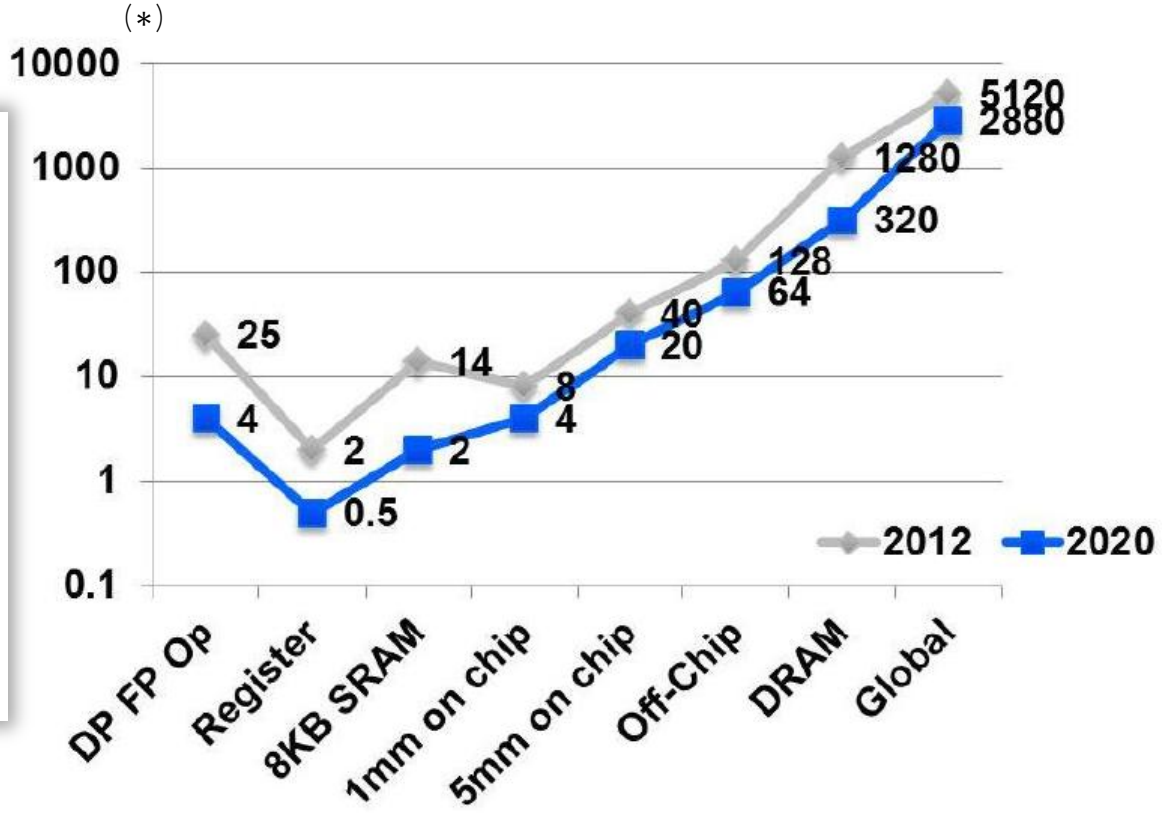


The Energy Challenge

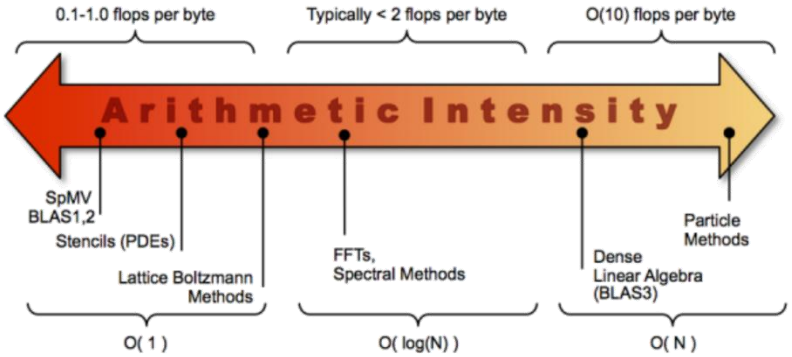
Message II

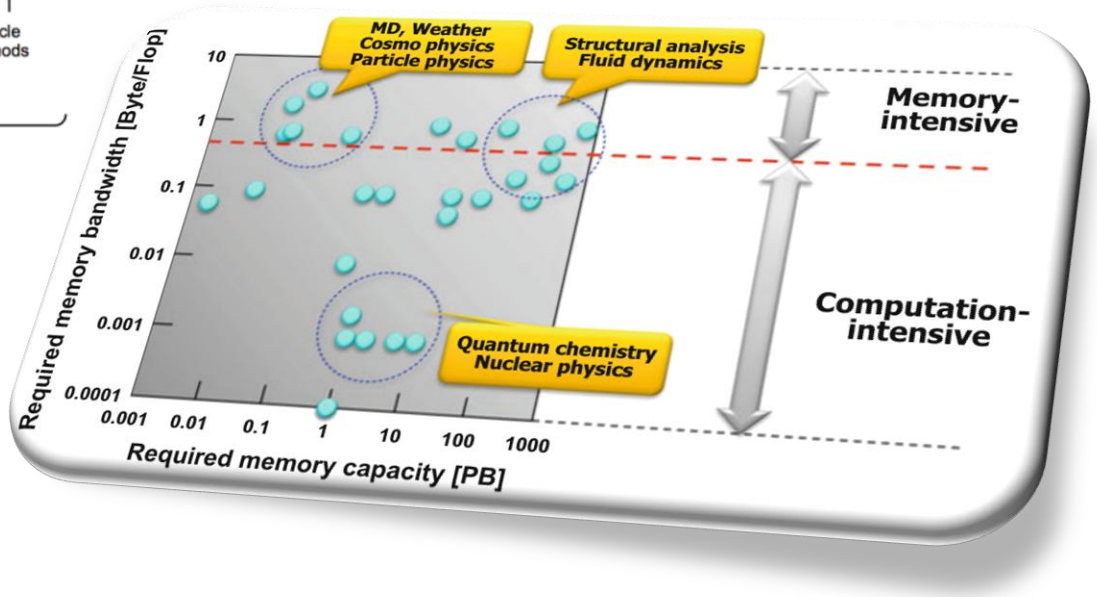
Moving memory is among the most expensive operation.
A FP op could cost ~4pJ while reading the data to be operated from memory ~700pj.

Data and projections:
R. Leland et al., 2014



The Energy Challenge

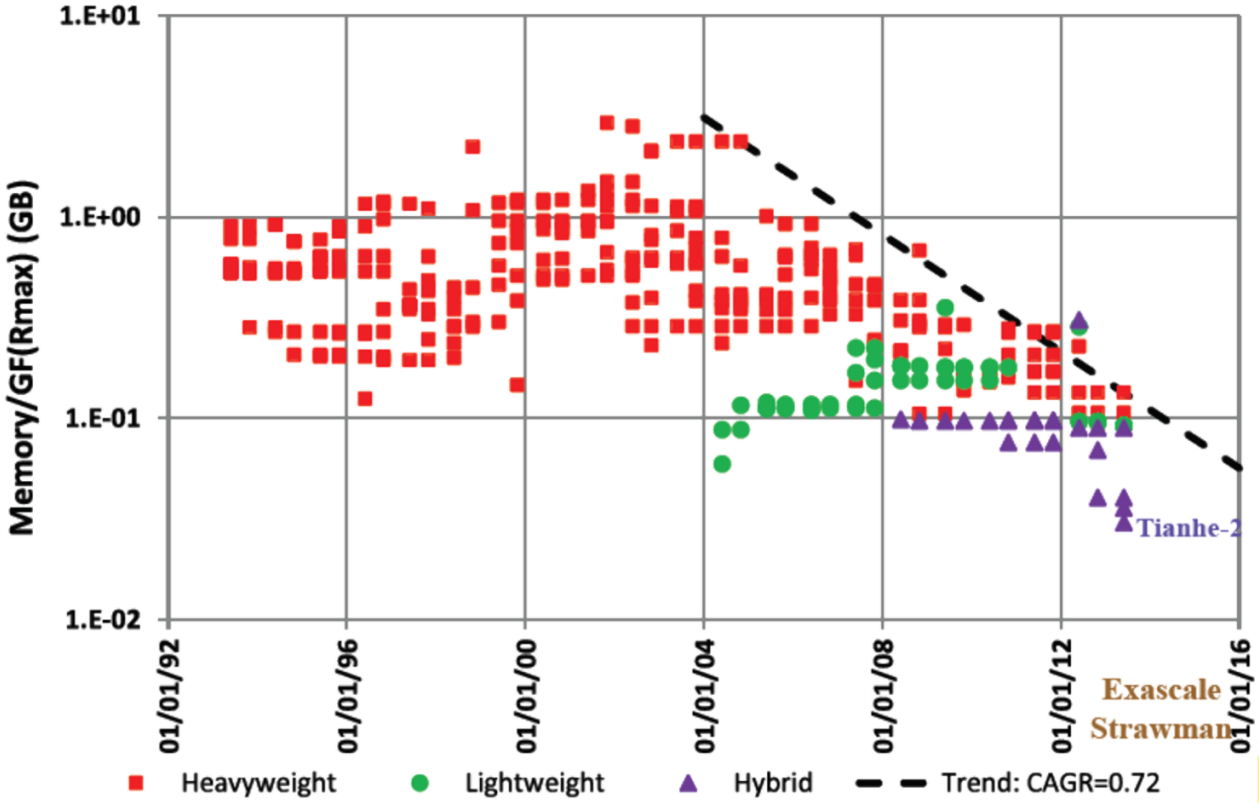




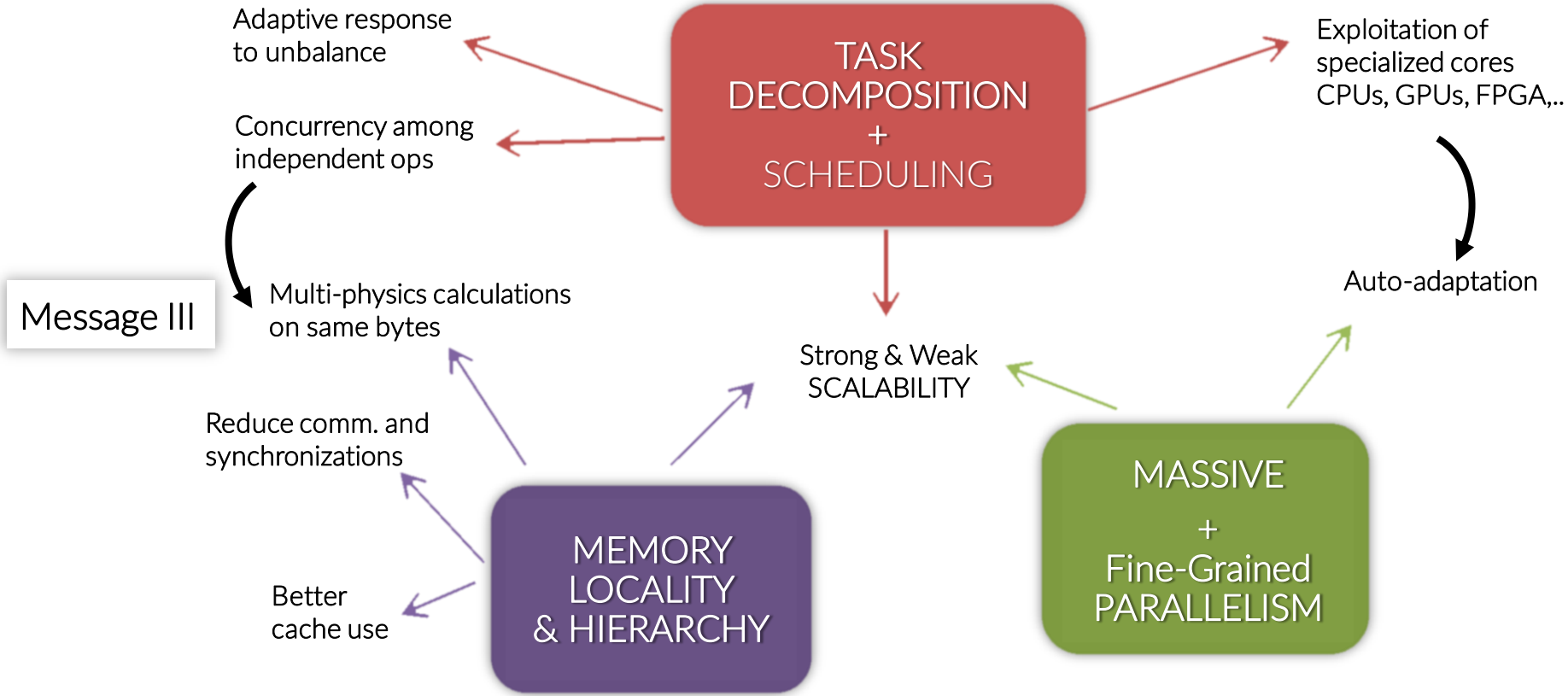
The Energy Challenge

The machines at the top of the TOP500 do not have sufficient memory to match historical requirements of 1B/Flop, and the situation is getting worse.

This is a big change: it places the burden increasingly on strong-scaling of applications for performance, rather than on weak-scaling like in tera-scale era.



The Energy Challenge



that's all, have fun