# Energy-driven algorithms

*… or energy-aware computing...*

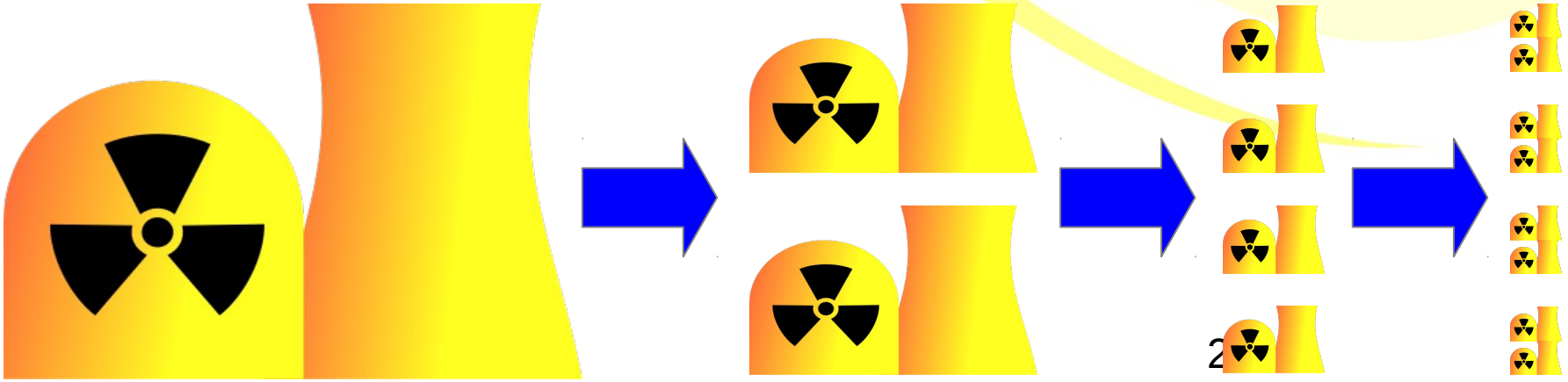
**Vincent Keller**

cadmos

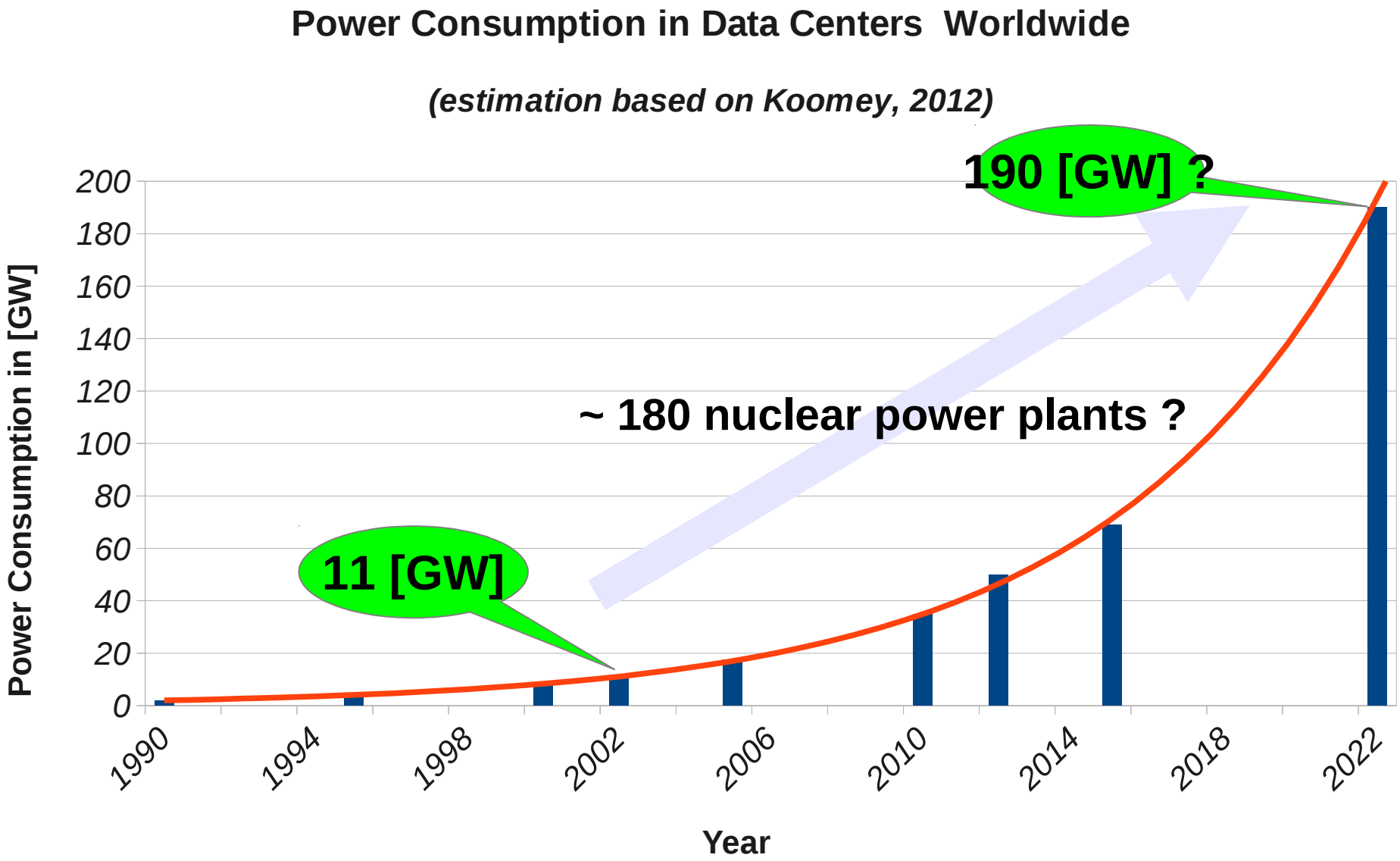
center for advanced modeling science

# Facts (2005)

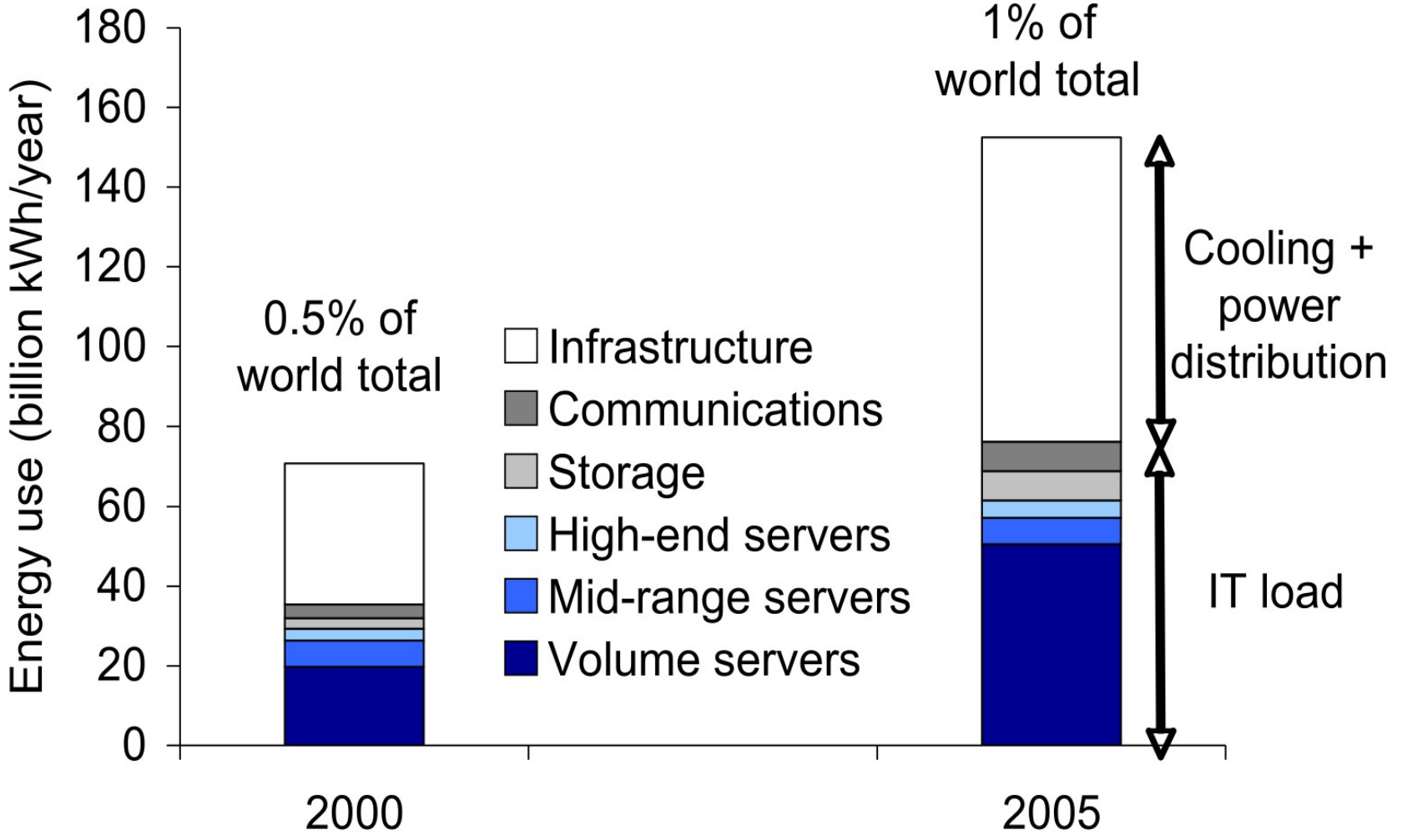
- Top 5 machines in the TOP500: 1-10 [MW]
- Power consumption for Data centers worldwide: 17,4 [GW]
- 1 mid-size nuclear power plant: 1 [GW]
- Power consumption needed by Data Centers doubles every 5 years



# (Pessimistic) Estimation 2002 – 2022



# Energy in Data Centers
# Where is it used ?



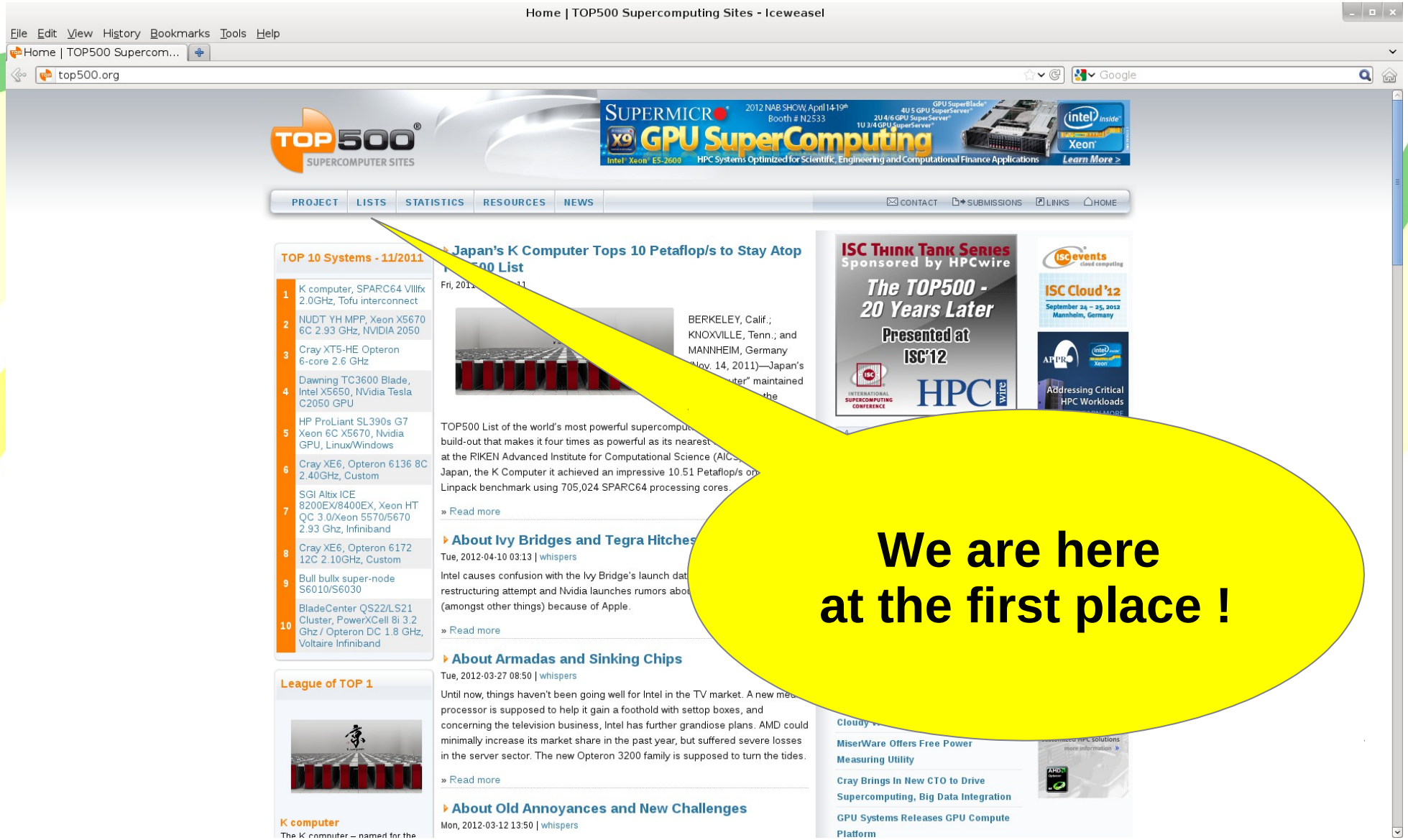



(Jonathan G. Koomey, 2012)

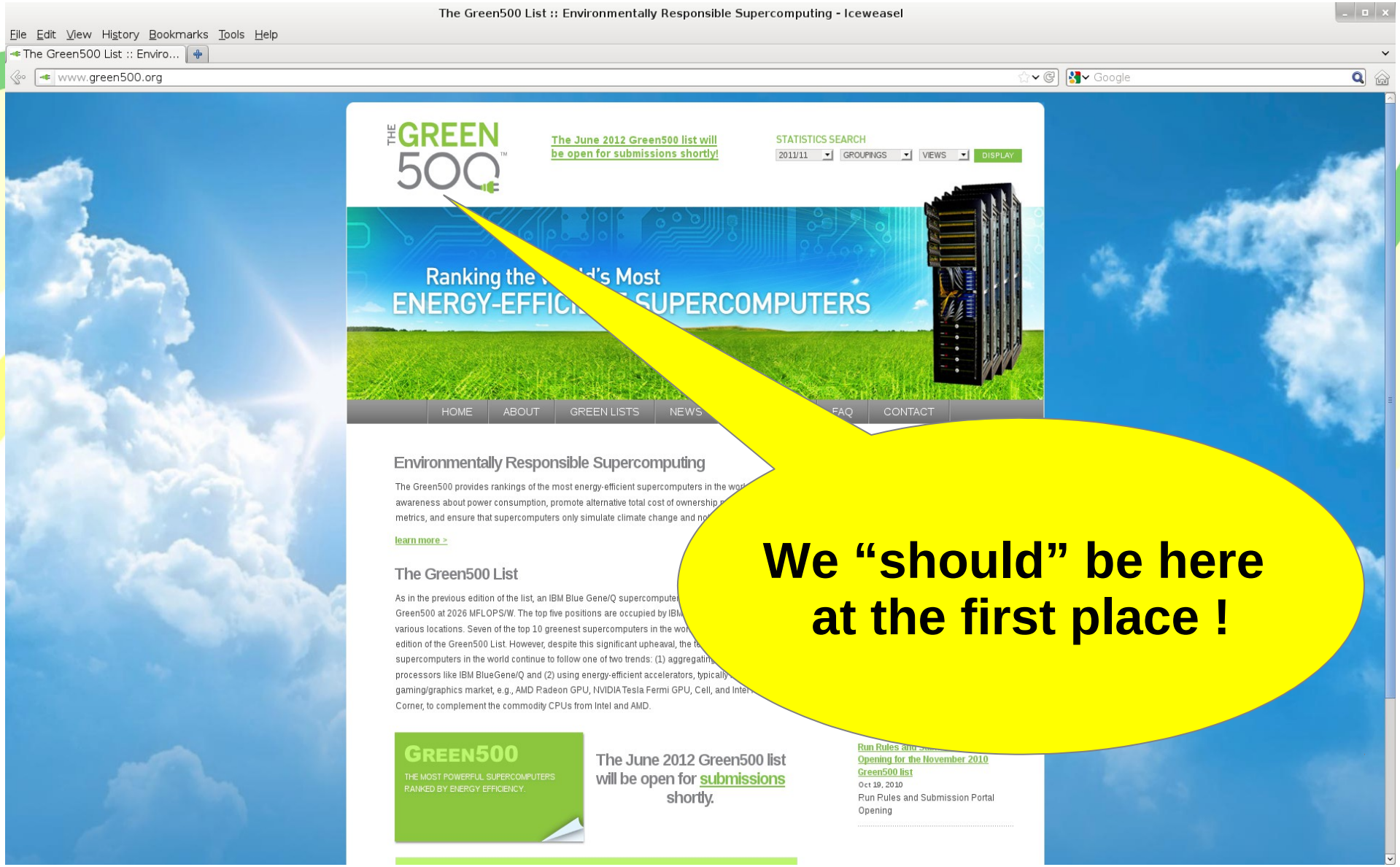
**Goal/Dream/Ads: Exascale ($10^{18}$) in 2020**

**How ? (and why ?)**

# What vendors say



# What vendors say (*Al Gore version)*

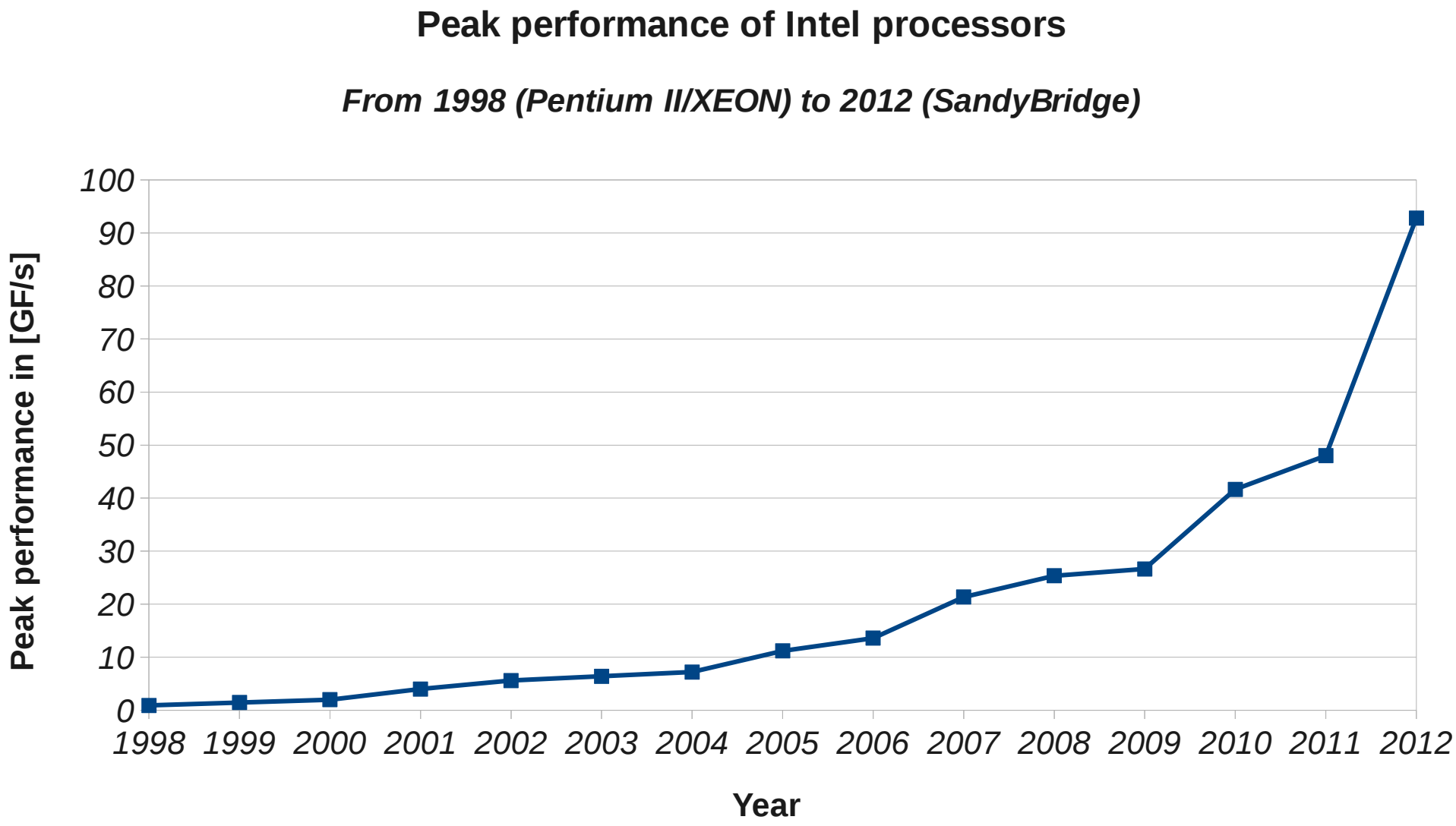


**The vendors base all their forecasts (and R&D) on the performance of HPL (High Performance LINPACK) for the TOP500 and GREEN500 lists**

***HPL is a highly optimized application's kernel***

***HPL is purely CPU-bound: its performance ($R_{max}$) is very close to the CPU peak performance***

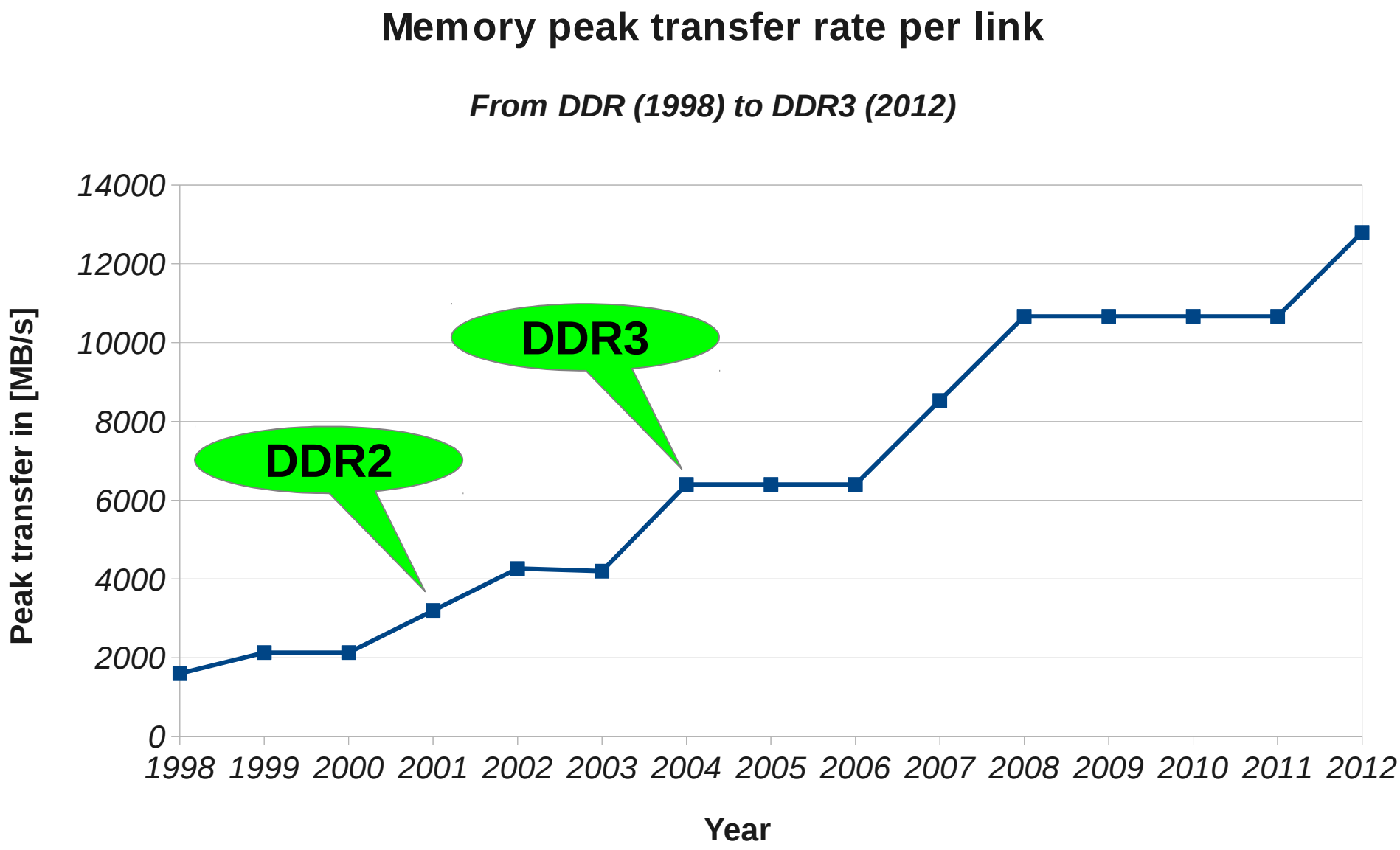
# Processor evolution ($R_{max}$)
# The Intel example



**OK, the CPU performance follows the Moore's law for HPL.**

**What about the memory ?**

# Memory performance evolution



**Nice correlated data, isn't it ?**

**No ! CPU performance grows exponentially while memory linearly.**

**So ... are our applications CPU or memory access dominated ?**

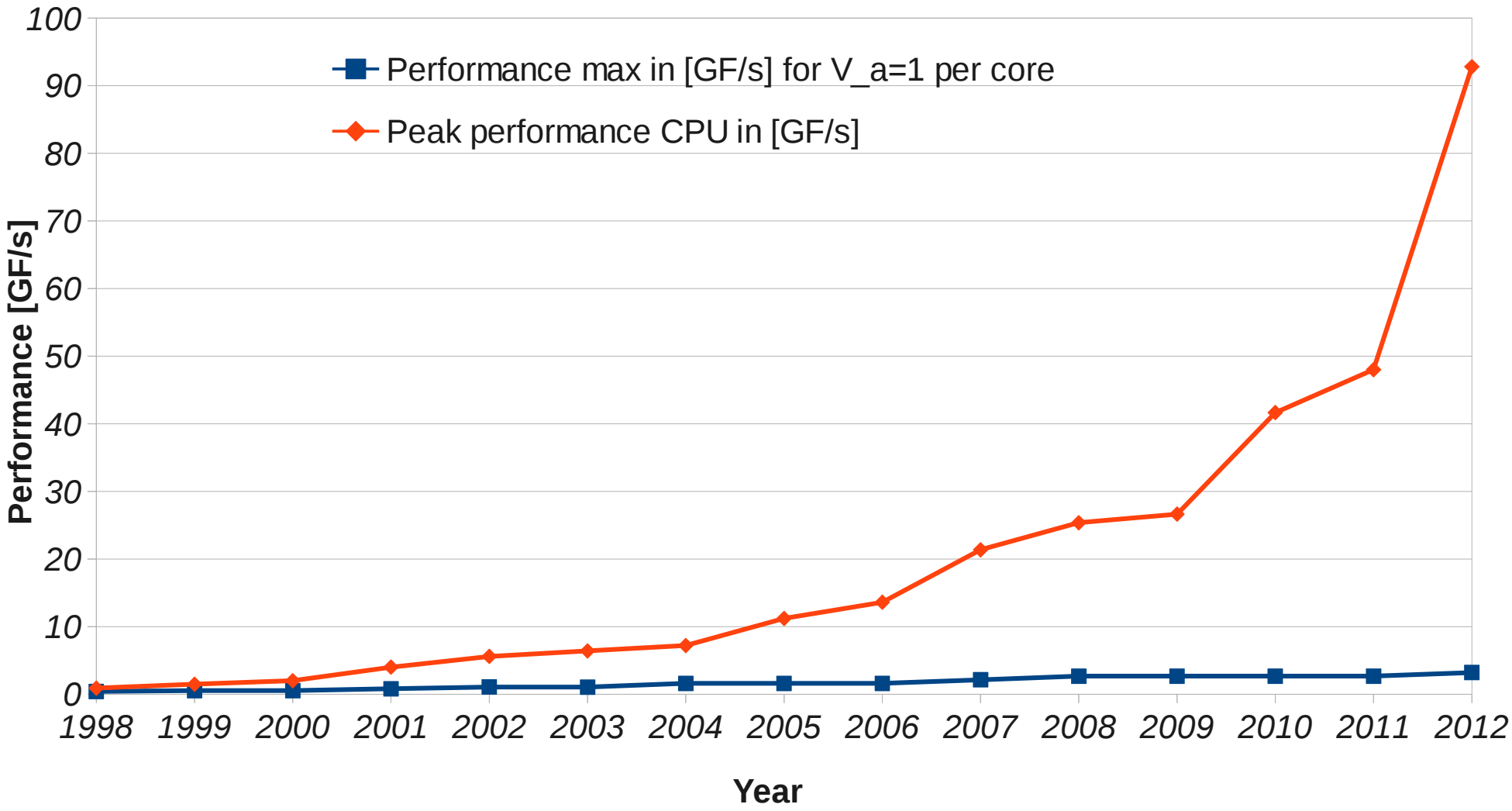
# Ax = b

- Many simulation codes can be “summarized” to the resolution of Ax = b. A is (probably) sparse and dim(x) « large enough »
- Let me define $V_a$, the “vectorization ratio”: the number of operations that can be performed by the CPU per memory access (LOAD or STORE)
- If $V_a = 1$ , each operand must be loaded from memory.
- This is the case for Ax = b

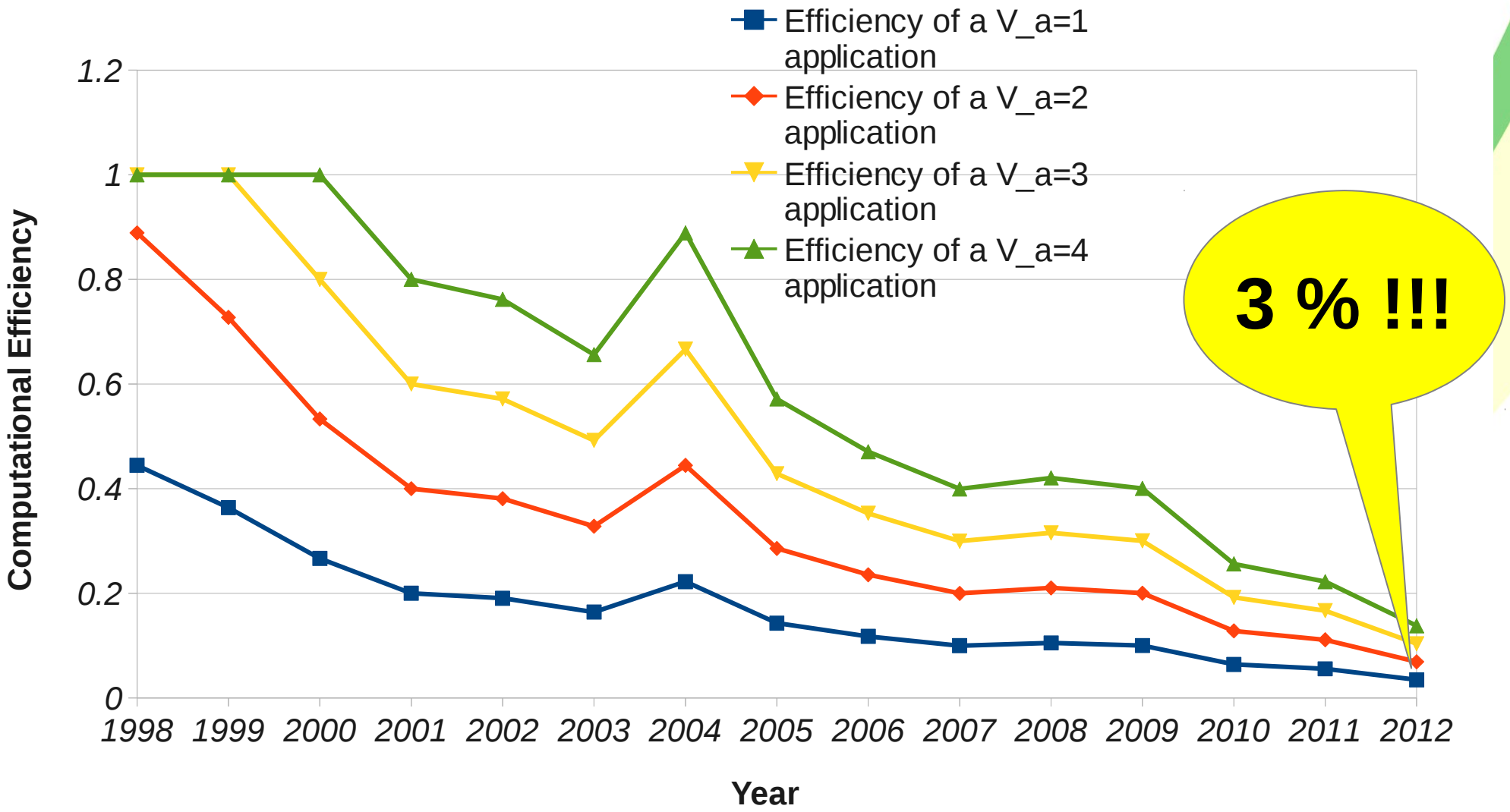
# Consequence

- If $V_a = 1$ (or close), the application is purely memory bounded.
- Memory bandwidth and latency are the bottleneck
- CPU performance is not an issue

# Comparison HPL and a memory-bound application



# Computational efficiencies of a $V_a$={1,2,3,4} application



# Energy speaking...

- 3% of computational efficiency means:
  - 3% of the CPU is used for what it is built for
  - 97% of the CPU is not used (but the CPU does not stop)
- 97 % of the energy brought to the CPU is used to heat the air (or the water) around the data center

**And we want to go for Exascale ?**

**OK, that is for a “pure” $V_a$=1 application.**

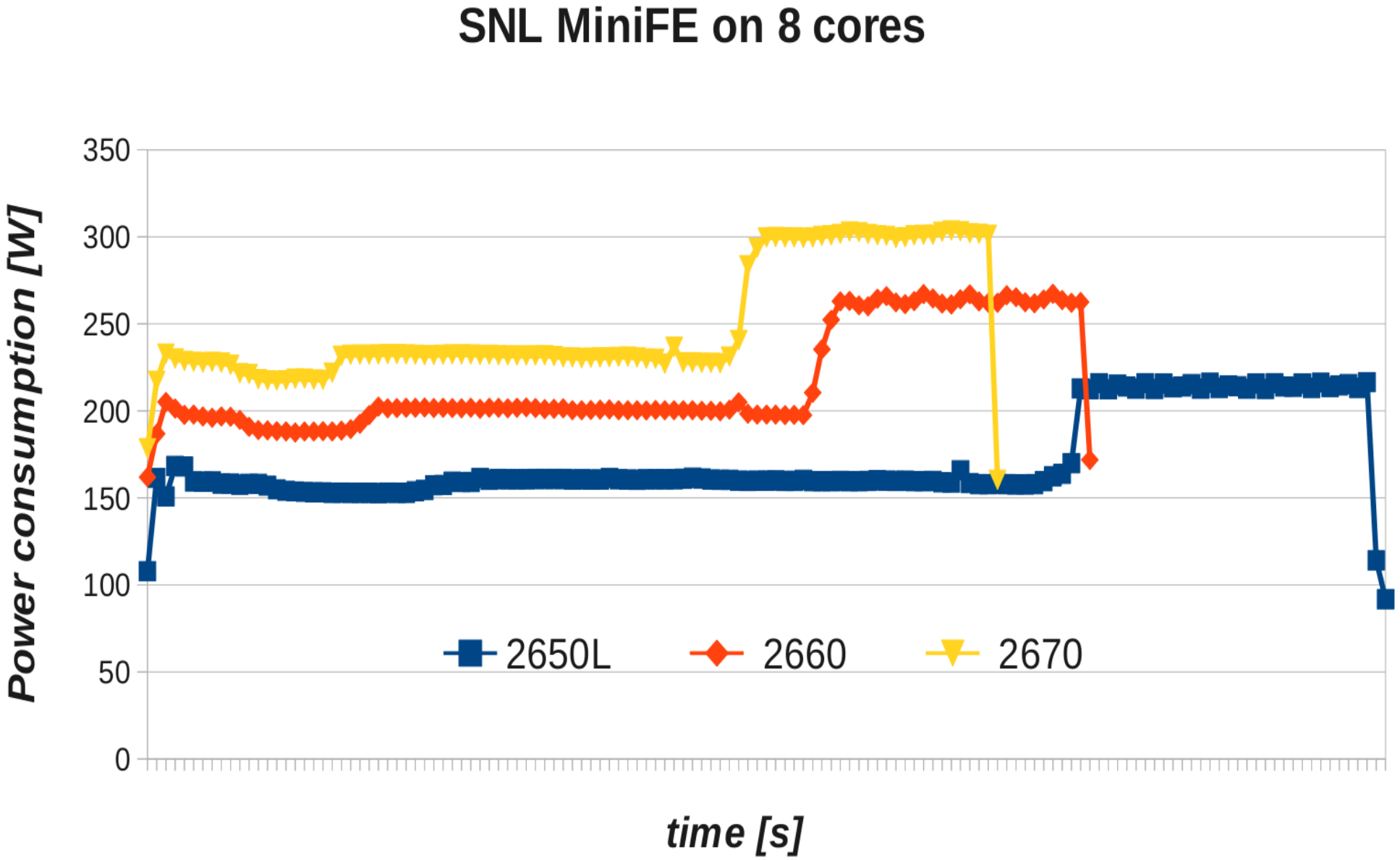
**What about REAL applications Energy footprint ?**

# Test case

- Three applications
  - MiniFE (FEM)
  - CPMD (Car-Parinello)
  - GEAR (n-body)
- Three 2012 CPU models:
  - CPU1 is “low power”
  - CPU2 is “middle class”
  - CPU3 is “high performance”
- All three CPUs have the same instruction set and the same memory banks connected.

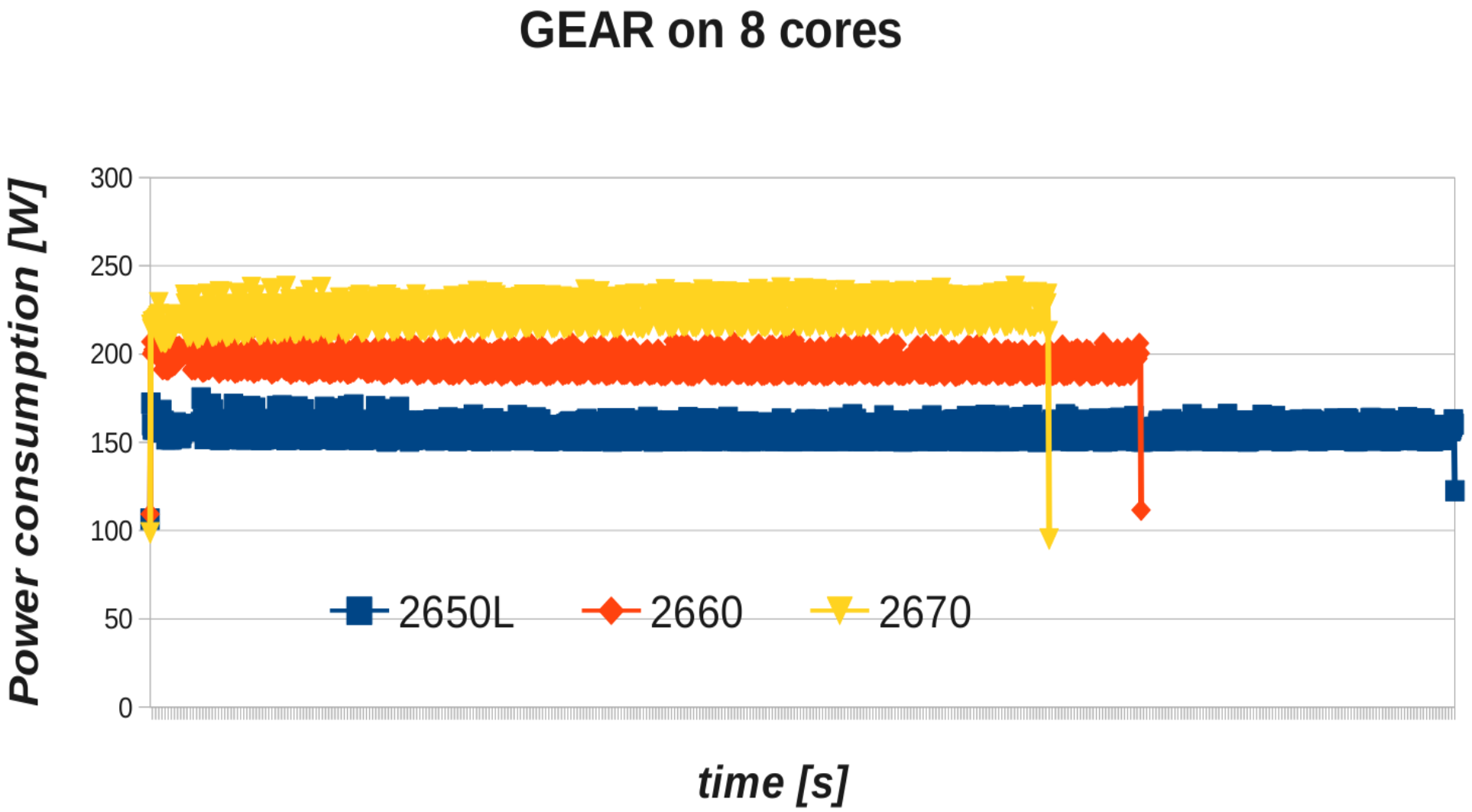
**Power consumption is measured during execution...**

# An example: SNL MiniFE

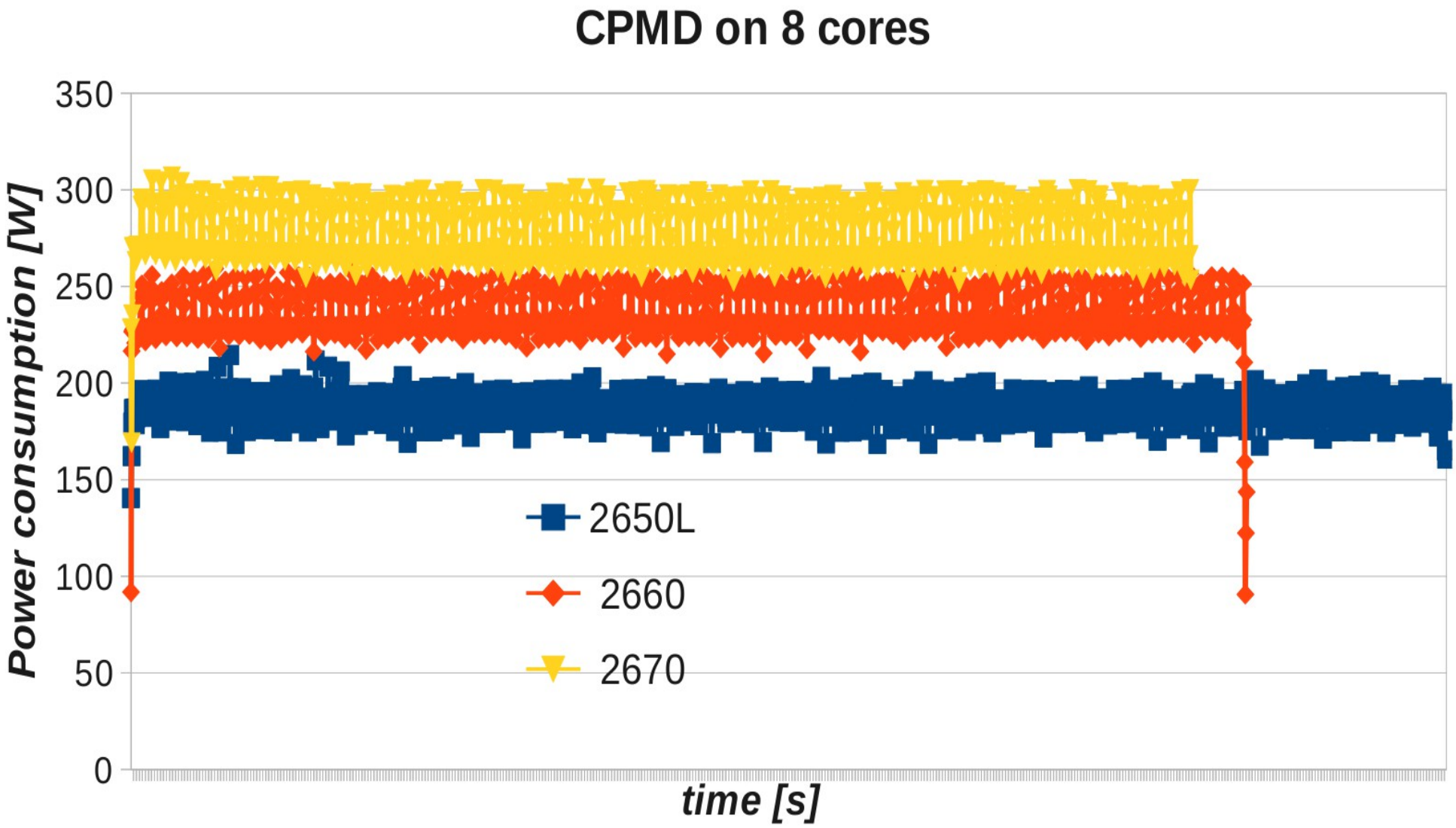



**MiniFE: “best approximation to an unstructured implicit finite element or finite volume application, but in 8000 lines or fewer.”**

# Another example: n-body GEAR



# A last example: Car-Parinello CPMD



# Energy To Solution

- E = P·t ( $E = \int p(t)\,dt$ )
- Energy-driven computing → choice of the most efficient machine for a given algorithm
- TTS (Time-to-Solution) can differ. We strictly deal

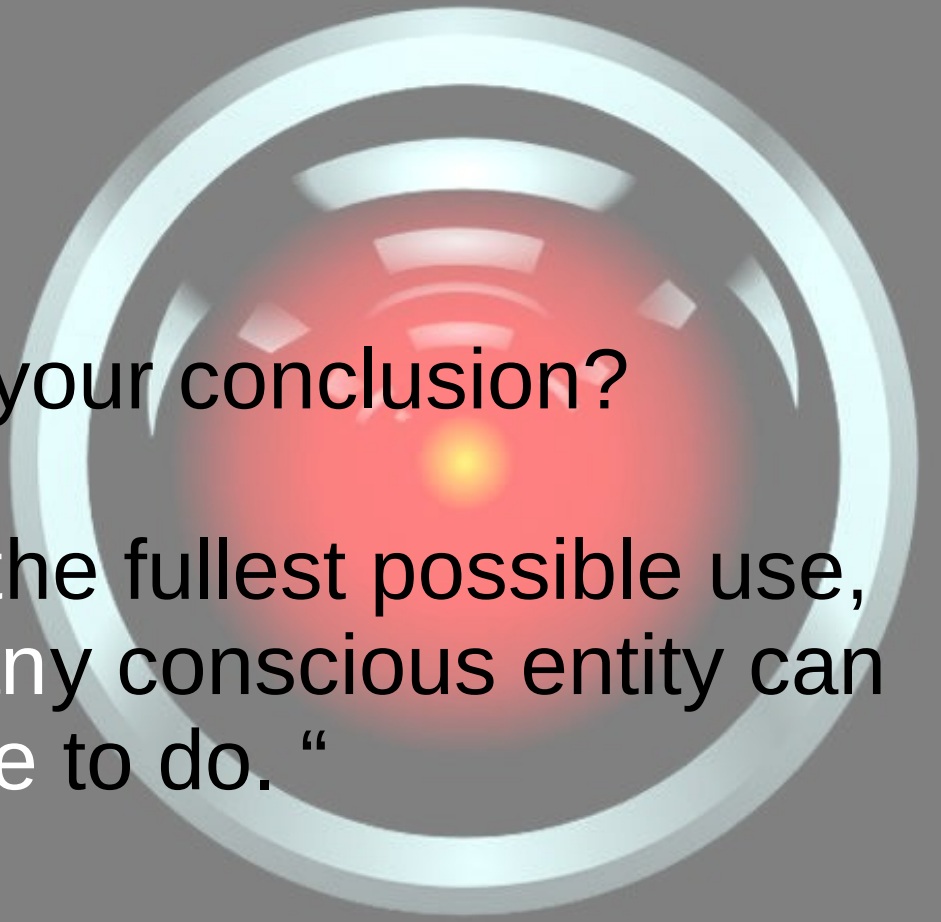
with Energy

# Energy-to-solution (in KJ)

| Application | Number of cores | CPU1 | CPU2 | CPU3 |
|---|---|---|---|---|
| **MiniFE** | **2 cores** | 54.01 | **48.225** | 48.823 |
| | **4 cores** | 29.571 | **27.695** | 28.672 |
| | **8 cores** | 23.072 | **22.206** | 23.087 |
| | **16 cores** | **18.547** | 19.425 | 21.761 |
| **GEAR** | **1 core** | 1223.269 | 1010.254 | **982.307** |
| | **2 cores** | 734.507 | 644.846 | **636.732** |
| | **4 cores** | 420.228 | **382.701** | 385.091 |
| | **8 cores** | 258.579 | **246.42** | 254.397 |
| | **16 cores** | **224.916** | 231.221 | 252.747 |
| **CPMD** | **4 cores** | 666.9 | **658.952** | 698.769 |
| | **8 cores** | **407.248** | 433.001 | 477.28 |
| | **16 cores** | **350.794** | 364.845 | 467.274 |

So HAL, what is your conclusion?

“I am putting myself to the fullest possible use, which is all I think that any conscious entity can ever hope to do. “



# Conclusion

- Choose the best suited machine.
  - Ex: For a memory-bound application ($V_a$=1), a **vector architecture** is more suited
- Change the algorithm if your problem can be solved differently
- Help CPU founders to produce suited chips with a balanced FSB/Core frequency: **CO-DESIGN**

# Conclusion

- Exascale in 2018 : probably possible.
- But (if we want to use it) :
  - Not with the current technology
  - Not without a tremendous improvement in the memory performance
  - Energy is not free. Energy-to-Solution metric will become more and more important with respect to Time-to-Solution
- Do we need that much power to make Science ?