Rebound effect in High Performance Computing

Denis Trystram
Nguyen Kim Thang
LIG, Grenoble INP, UGA

Rebound effect
in High Performance Computing

Denis Trystram
Nguyen Kim Thang
LIG, Grenoble INP, UGA

Improve the performance of a technology
but it turns out to be worse

DES SOLUTIONS MAIS ATTENTION À L'EFFET REBOND…

Src: Idris, Rafael Medeiros – ORAP – 2021

L'effet rebond caractérise un effet pervers et paradoxal des progrès en matière d'efficacité énergétique.

Christophe Biernacki, Frugalis 2024

Rebound effect
in High Performance Computing

Improve the performance of a technology
but it turns out to be worse

Rebound effect
in High Performance Computing

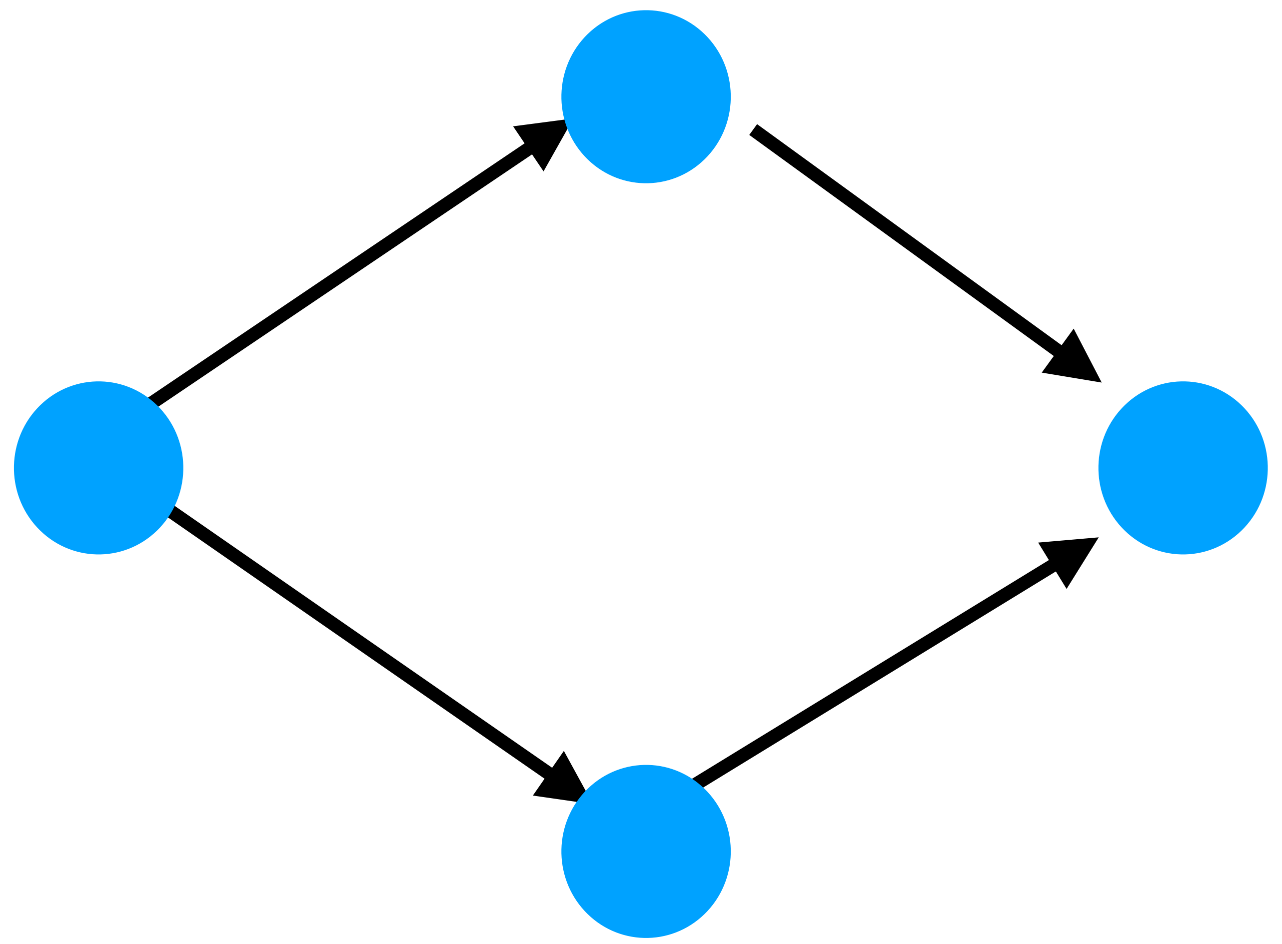
Improve the performance of a technology
but it turns out to be worse

- Principle cause: usage

Rebound effect
in High Performance Computing

Improve the performance of a technology
but it turns out to be worse

- Principle cause: usage

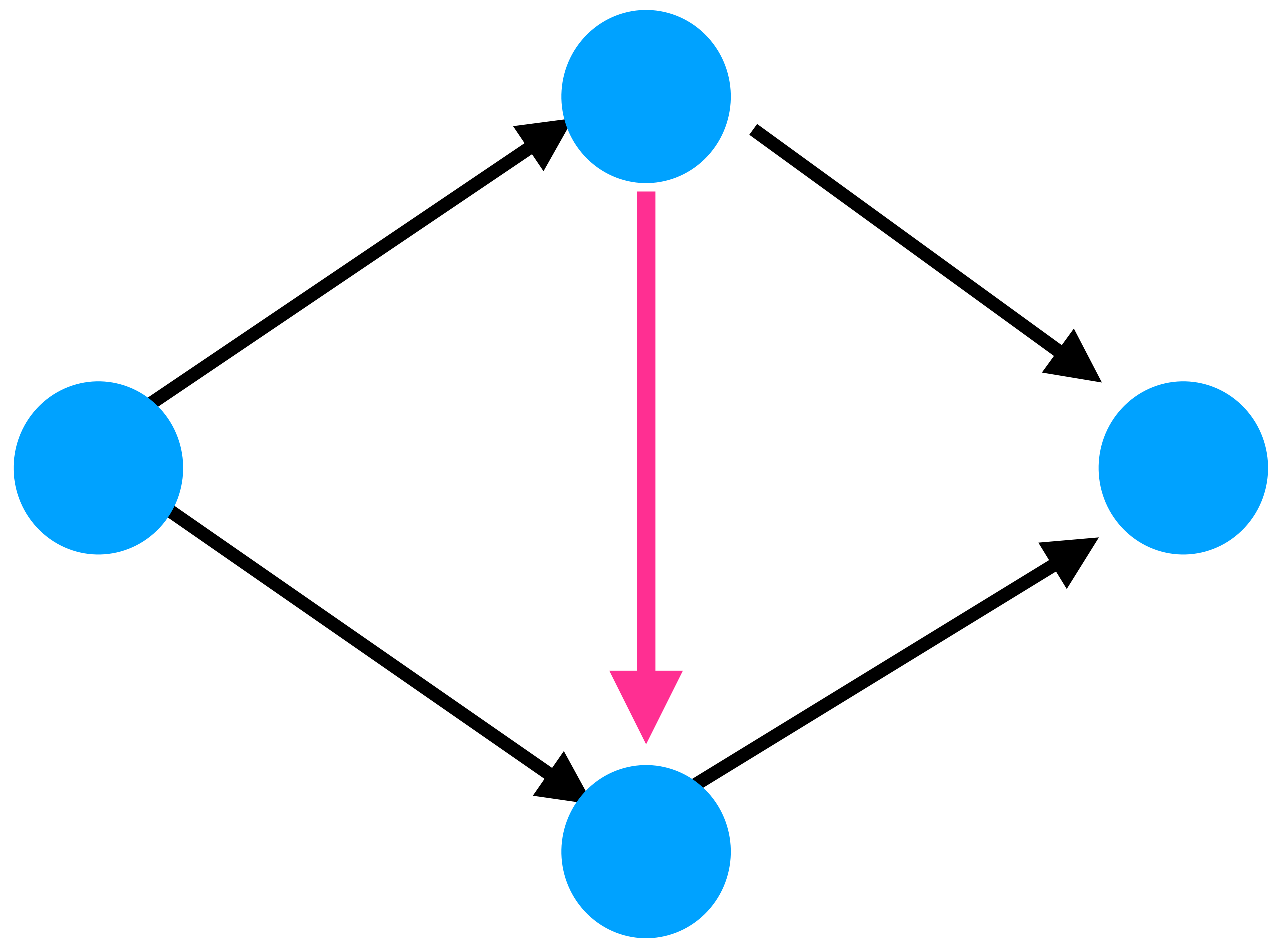


Braess paradox
(transportation in Seoul, New York, electronic networks, electron systems)

Rebound effect
in High Performance Computing

Improve the performance of a technology
but it turns out to be worse

- Principle cause: usage



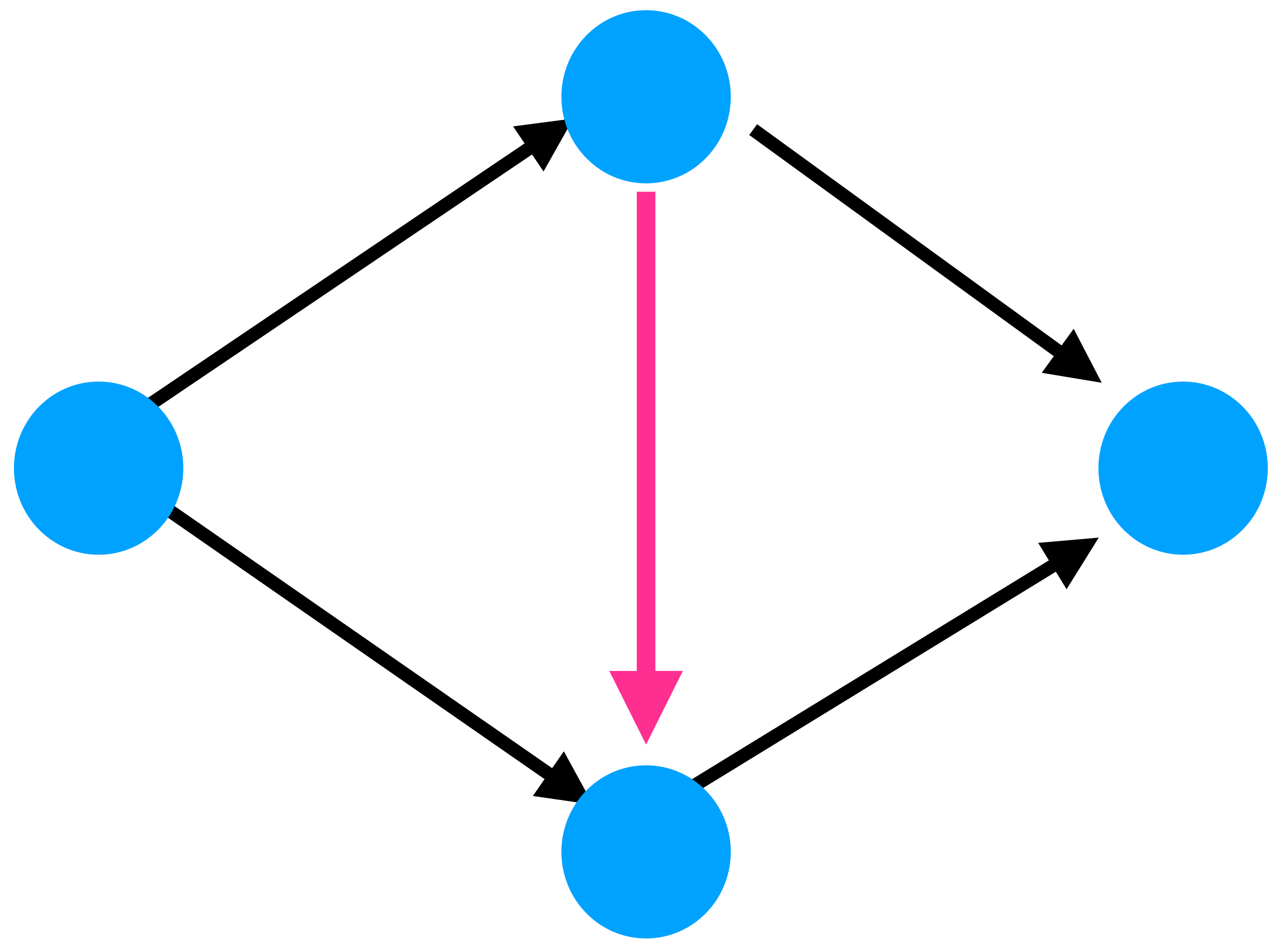
Braess paradox
(transportation in Seoul, New York, electronic networks, electron systems)

Rebound effect in High Performance Computing

Improve the performance of a technology
but it turns out to be worse

- Principle cause: usage

- Model + Theoretical Analysis:
  Scheduling + Game Theory



Braess paradox
(transportation in Seoul, New York, electronic networks, electron systems)

Model

- n jobs and m machines
- each job: work quantity, release date and required deadline
- **social objective:** minimize the energy

Machines: energy = $\int_0^{\infty} s(t)^{\alpha} dt, \ (2 \leq \alpha \leq 3)$

Model

- n jobs and m machines
- each job: work quantity, release date and required deadline
- **social objective:** minimize the energy

Machines: energy = $\int_0^\infty s(t)^\alpha dt, \; (2 \leq \alpha \leq 3)$

distribute energy to jobs

Model

- n jobs and m machines
- each job: work quantity, release date and required deadline
- **social objective:** minimize the energy

<u>Machines</u>: energy = $\int_0^\infty s(t)^\alpha dt,\ (2 \le \alpha \le 3)$

distribute energy to jobs

<u>New technology:</u>

decrease α

Model

- n jobs and m machines
- each job: work quantity, release date and required deadline
- **social objective:** minimize the energy

Machines: energy = $\int_0^\infty s(t)^\alpha dt,\ (2 \leq \alpha \leq 3)$

distribute energy to jobs

New technology:

decrease α

Jobs: rational, i.e., minimizes

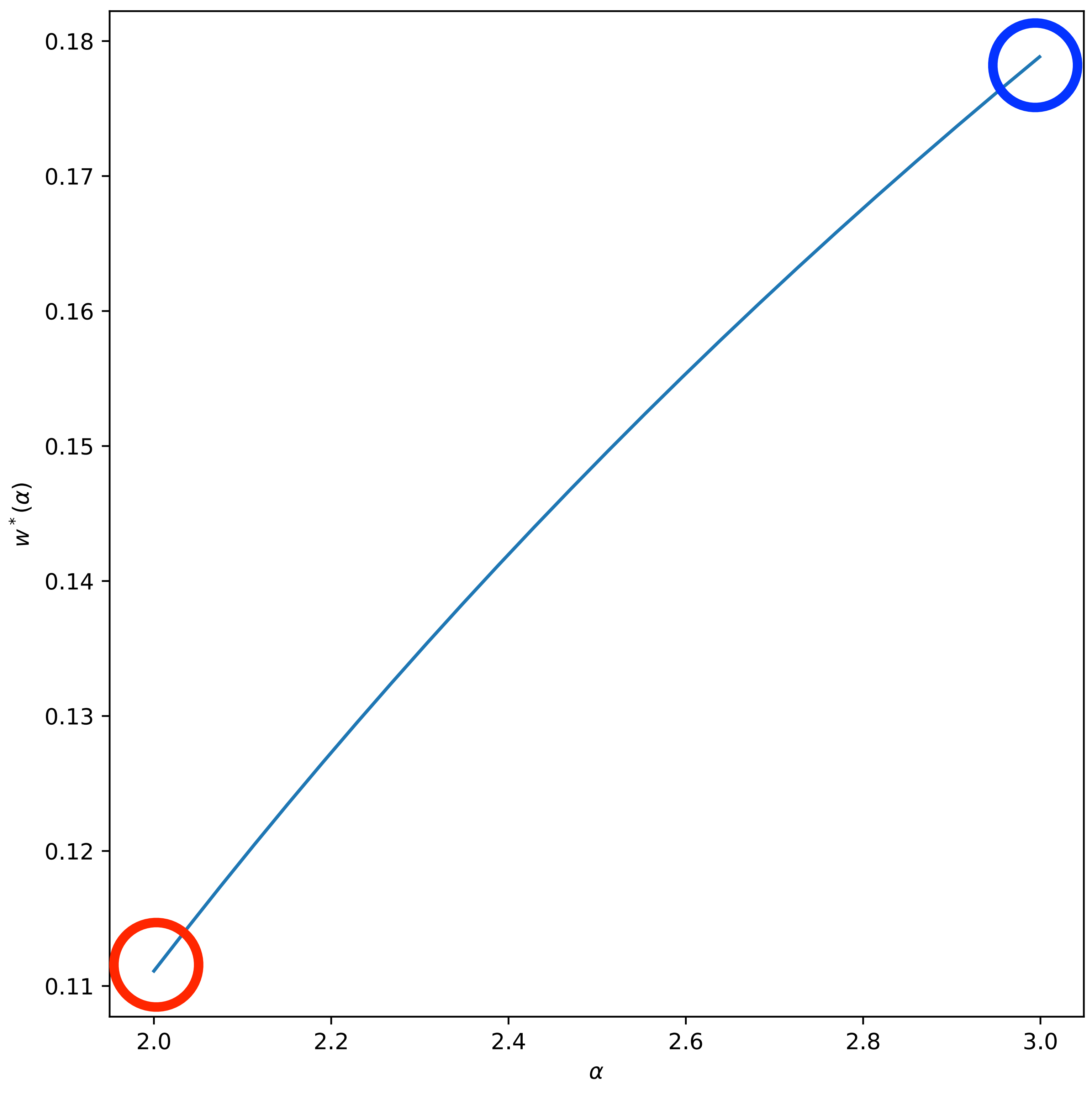
its own cost: **energy** + **deadline**

Observations/Theorems

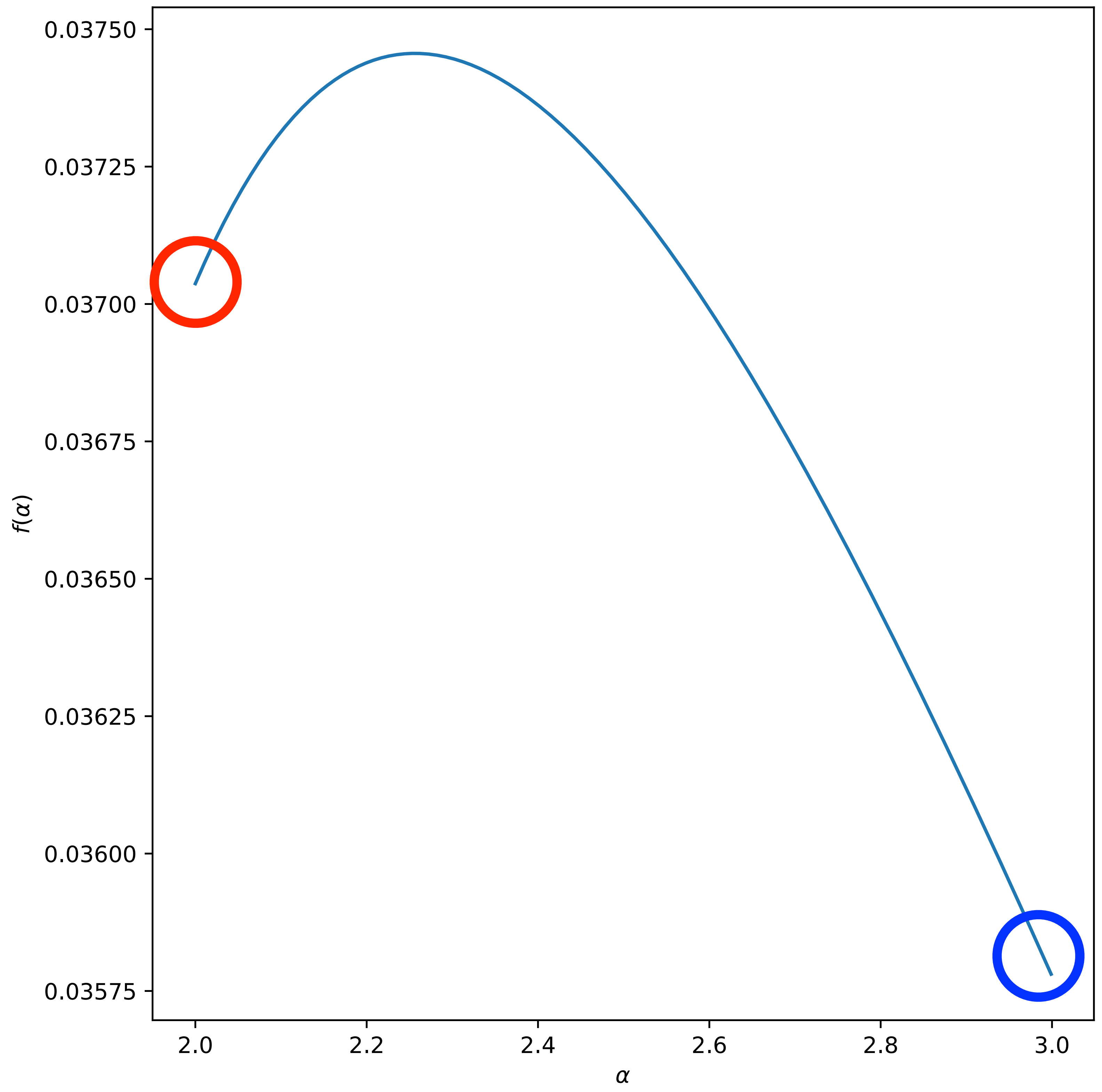
- “New form” of rebound effect: work quantity is reduced but the demand is more urgent leading to a larger energy consumption.

Observations/Theorems

- "New form" of rebound effect: work quantity is reduced but the demand is more urgent leading to a larger energy consumption.



work quantity



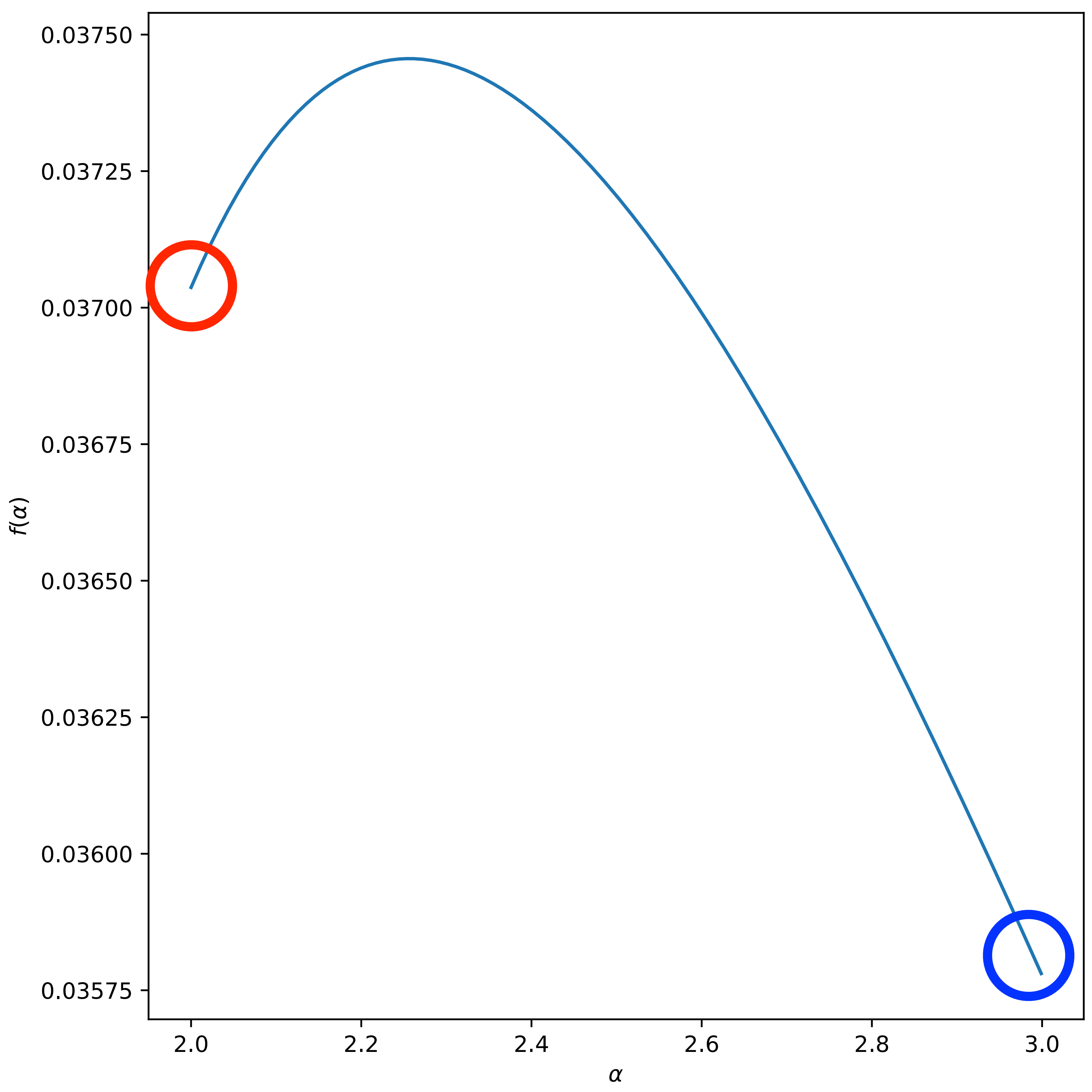
energy

A closer look/take-home message

- User chooses strategically

deadline $$\left[(\alpha-1)w^{\alpha-1}\right]^{\frac{1}{\alpha}}$$

workload $$\left[\frac{\alpha-1}{(2\alpha-1)(\alpha-1)^{\frac{1}{\alpha}}}\right]^{\frac{\alpha}{\alpha-1}}$$



energy

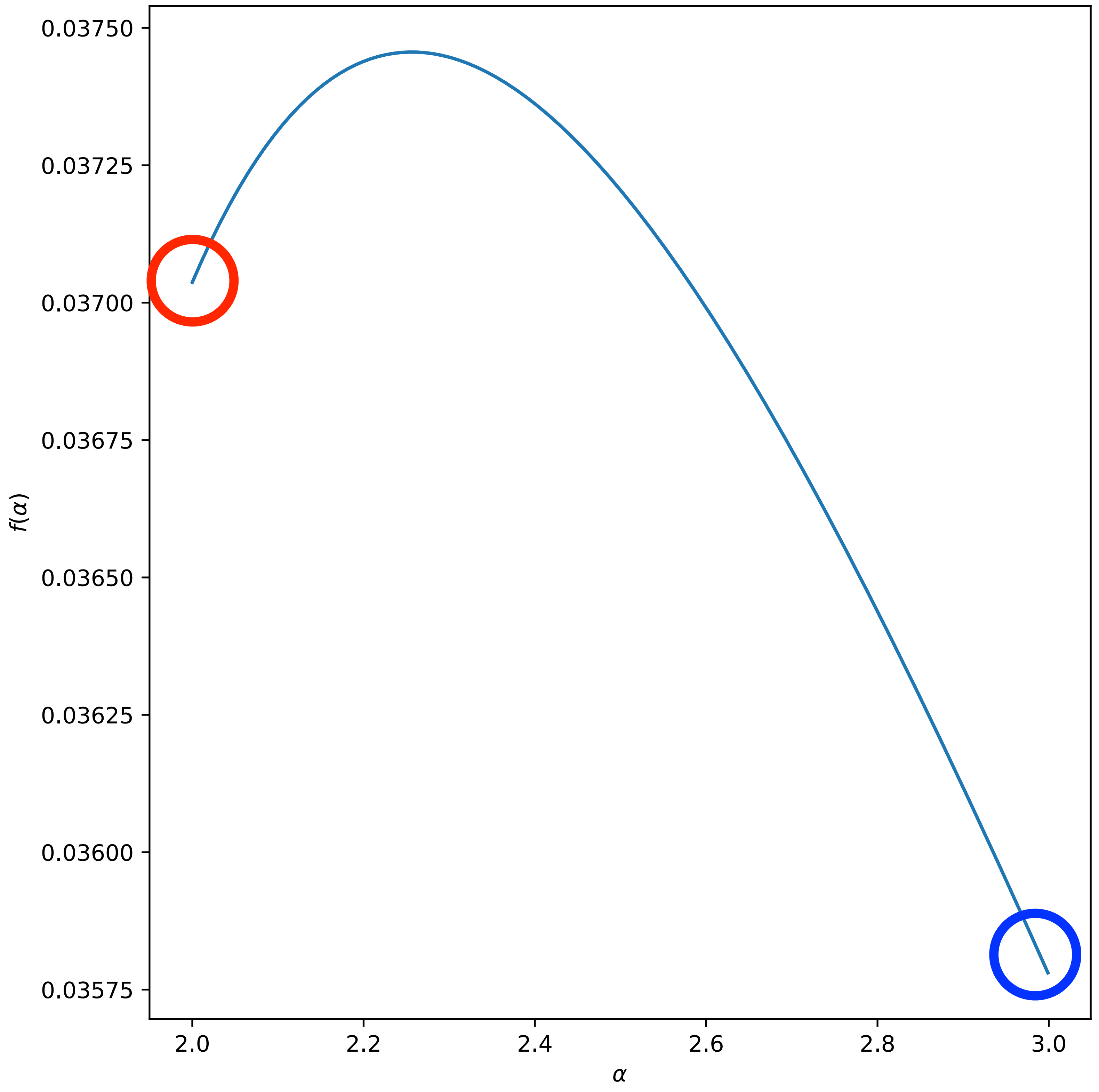
- Our analysis can be applied to HPC, in particular the machines IBM Summit (OLCF-4, USA) and Tianhe-2 (TH-2, China)

A closer look/take-home message

- User chooses strategically

deadline $\left[(\alpha-1)w^{\alpha-1}\right]^{\frac{1}{\alpha}}$

workload $\left[\frac{\alpha-1}{(2\alpha-1)(\alpha-1)^{\frac{1}{\alpha}}}\right]^{\frac{\alpha}{\alpha-1}}$



energy

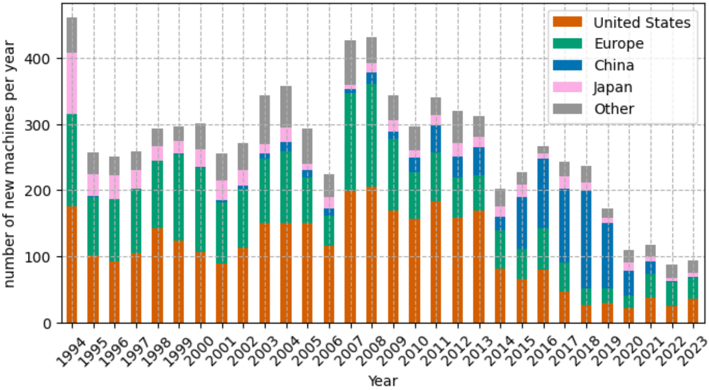
- Our analysis can be applied to HPC, in particular the machines IBM Summit (OLCF-4, USA) and Tianhe-2 (TH-2, China)

Several forms of rebound effects exists

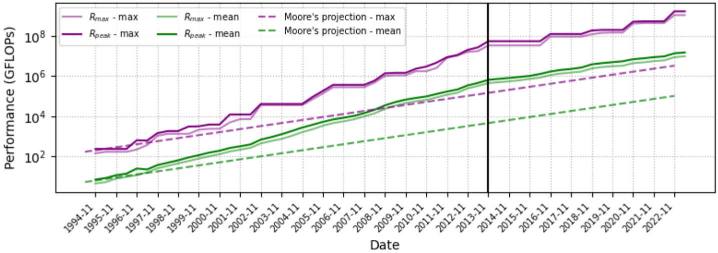
Mechanism/Algorithm design for a better use

TOP500

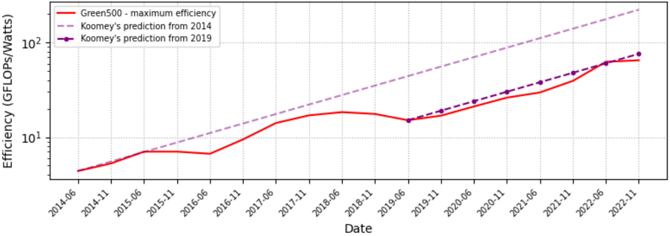
Ranking list of HPC machines every 6 months based on LINPACK benchmark.



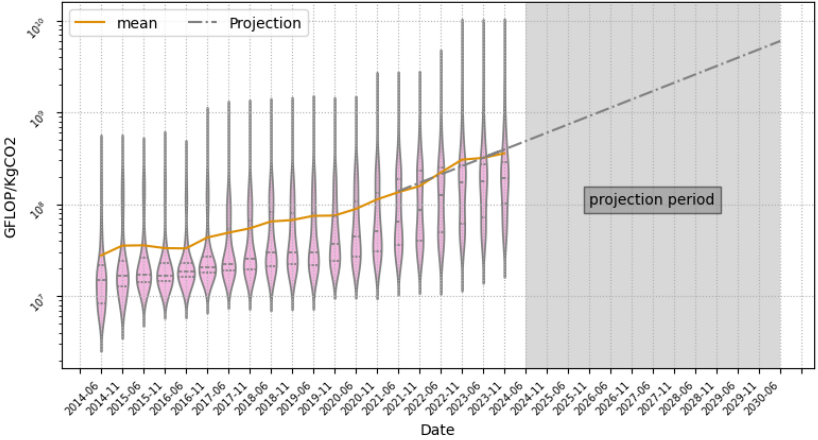
performance increase



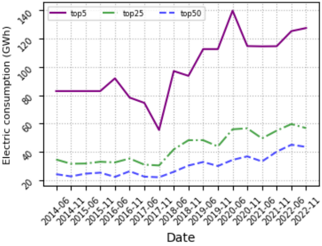
Energy efficiency

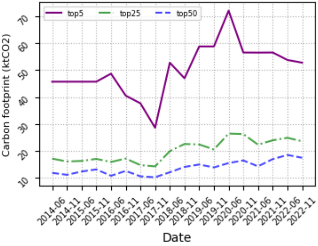


Carbon efficiency



Emission per machine



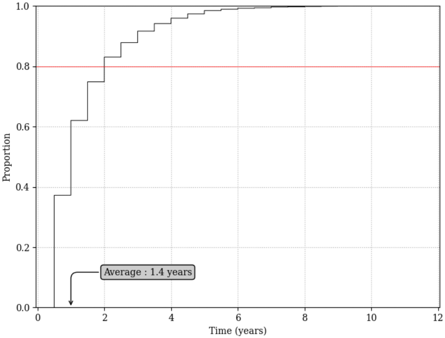


Age of HPC

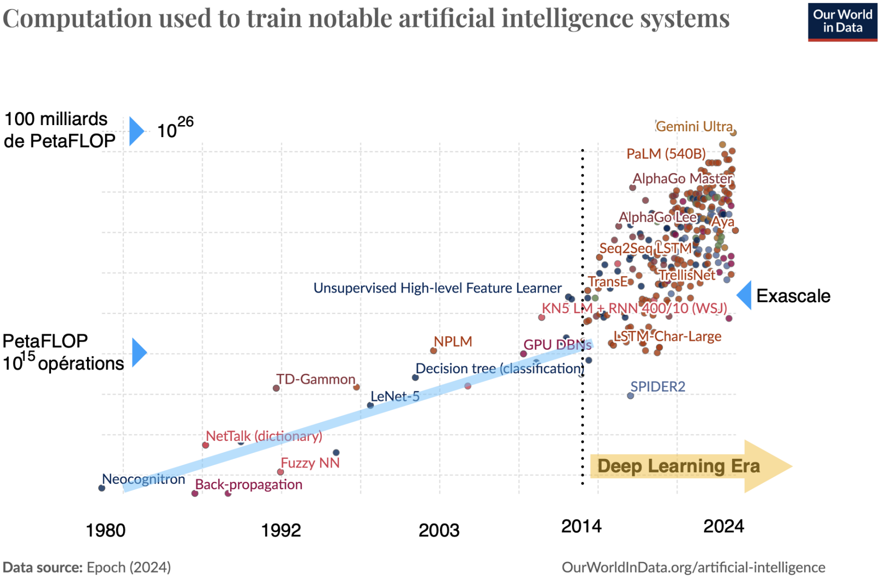
Mean time in TOP500 of 1.4 years

80% below 2 years

But how many times do they last?
How many times per CPU/GPU?



IA trends



Computation used to train notable artificial intelligence systems
Our World in Data
100 milliards de PetaFLOP
10^{26}
PetaFLOP 10^{15} opérations
Exascale
Gemini Ultra
PaLM (540B)
AlphaGo Master
AlphaGo Lee
Aya
Seq2Seq LSTM
TransE
TrellisNet
Unsupervised High-level Feature Learner
KN5 LM + RNN 400/10 (WSJ)
LSTM-Char-Large
NPLM
GPU DBNs
Decision tree (classification)
SPIDER2
TD-Gammon
LeNet-5
NetTalk (dictionary)
Fuzzy NN
Neocognitron
Back-propagation
Deep Learning Era
1980
1992
2003
2014
2024
Data source: Epoch (2024)
OurWorldInData.org/artificial-intelligence

IA vs HPC

Training time on Top1 HPC machine