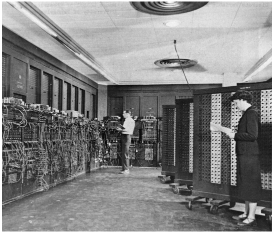
High Performance Computing: Towards Better Performance Predictions and Experiments

Tom Cornebize

2 June 2021, PhD defense

No science without computing

Arithmometer (1851)

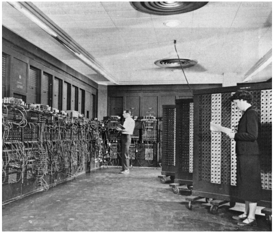


ENIAC (1945)

Fugaku (2021)

No science without computing

Arithmometer (1851)



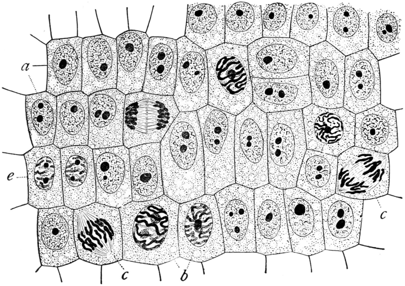
ENIAC (1945)

Fugaku (2021)

Last decades:

- Exponential performance improvements (e.g. sequencing an entire human genome costed $100,000,000 in 2001, $1000 now)
- At the price of complexity (both software and hardware)

Experimental study of computer performance



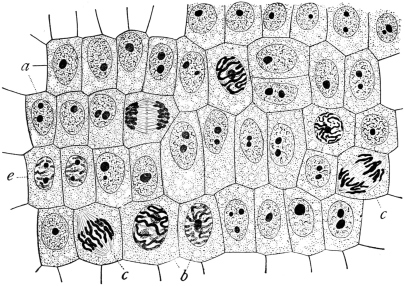
Similar to natural sciences

Complexity ⇒ Variability and Opacity

⇒ No perfect model

⇒ Need for experiments

Experimental study of computer performance



Similar to natural sciences

Complexity ⇒ Variability and Opacity

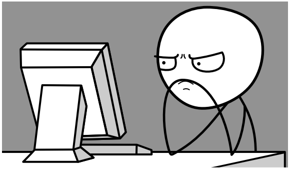
⇒ No perfect model

⇒ Need for experiments

Experiments can be carried in reality or in simulation

Context

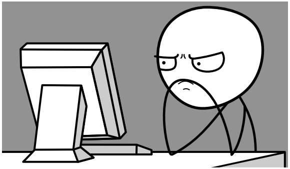
Typical Performance Evaluation Questions (Given my application and a supercomputer)



- Before running
  - How many nodes?
  - For how long?
  - Which parameters?

Context

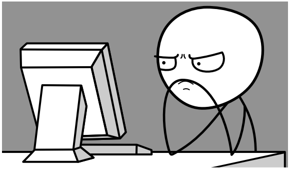
Typical Performance Evaluation Questions (Given my application and a supercomputer)



- Before running
  - How many nodes?
  - For how long?
  - Which parameters?
- After running
  - Performance as “expected”?
  - Problem in the app or the platform?

Context

Typical Performance Evaluation Questions (Given my application and a supercomputer)

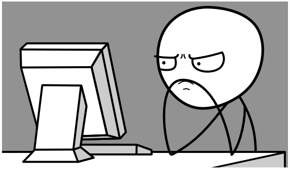


- Before running
  - How many nodes?
  - For how long?
  - Which parameters?
- After running
  - Performance as “expected”?
  - Problem in the app or the platform?

So many large-scale runs, solely to tune performance?!?

Context

Typical Performance Evaluation Questions (Given my application and a supercomputer)



- **Before** running
  - How many nodes?
  - For how long?
  - Which parameters?
- **After** running
  - Performance as “expected”?
  - Problem in the app or the platform?

So many large-scale runs, solely to tune performance?!?

Holy Grail: Predictive Simulation on a “Laptop”

Capture the **whole application** and **platform complexity**

Sim(Em)ulation: The SMPI Approach

Full reimplementation of MPI on top of SIMGRID

- C/C++/F77/F90 codes run unmodified out of the box
- Simply replace mpicc/mpirun by smpicc/smpirun

Sim(Em)ulation: The SMPI Approach

Full reimplementation of MPI on top of SIMGRID

- C/C++/F77/F90 codes run unmodified out of the box
- Simply replace mpicc/mpirun by smpicc/smpirun

Emulation: how?

- Computations run for real on a laptop
- Communications are faked, good fluid network models
- Performance model for the target platform

Goal: predict the performance of a parallel application

Goal: predict the performance of a parallel application

Thesis contribution

- Case study: High Performance Linpack (HPL)
- Skip the expensive computations (mostly `dgemm`) and replace them by a performance model
- Extensive (in)validation, comparing simulations with reality