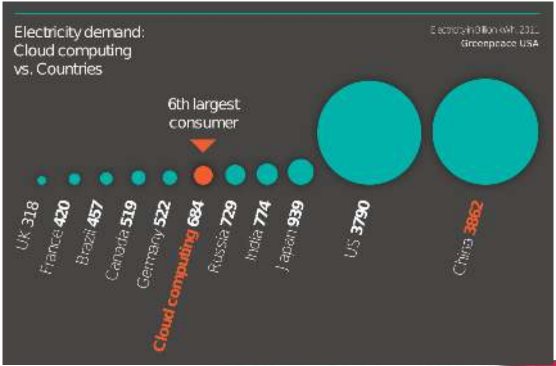
# Intro on Multi-objective resources optimization Performance- and Energy-aware HPC and Clouds

Georges Da Costa

Yerevan, Armenian National Academy of Sciences

# IT impact on electricity

- Recent datacenters: 40000 servers, 500000 services (virtual machines). Google, Facebook > 1 million servers
- One major power consumer
    - 2000 : 70 TWh
    - 2007 : 330 TWh, 2% of $CO_2$ world production
    - 2011 : $6^{th}$ electricity consumer in the world
    - 2020 : 1000 TWh
- Rising
    - 2014 to 2016: 90% of datacenters will need hardware upgrades



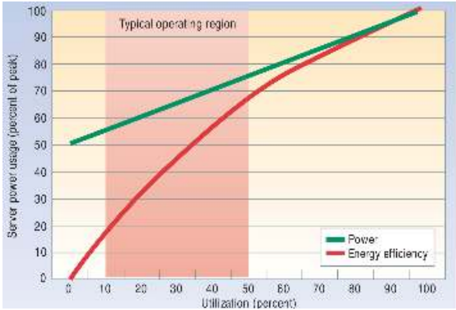
# Sustainable datacenters

- Action can be done at several different levels
  - Hardware level: changing servers or cooling system
    - If entropy is constant, theoretical energy consumtion is 0 !
  - Application level: rewrite applications while changing paradigm*or library
  - Middleware level: manages servers and services/applications
- Middleware: minimal cost, maximal impact
  - OpenStack: 30% of market share in 2014
  - OpenSource solutions: 43% (+72% in 2 years)


* Georges et al. *Exascale machines require new programming paradigms and runtimes*, SFI journal, 2015

# Low utilization = high electrical waste

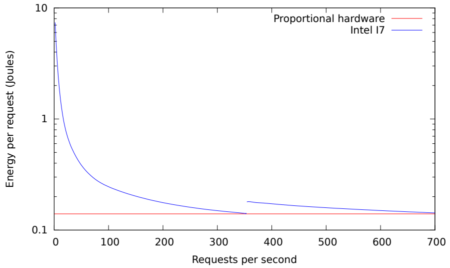
- In large organizations, computers are usually working between 10 to 50% load
- Idle power is half of max power




Barroso 2007

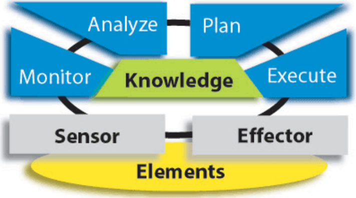
# Low utilization = high electrical waste

- In large organizations, computers are usually working between 10 to 50% load
- Idle power is half of max power
- Problem: On low load, Watt/Request is bad



# Middlewares

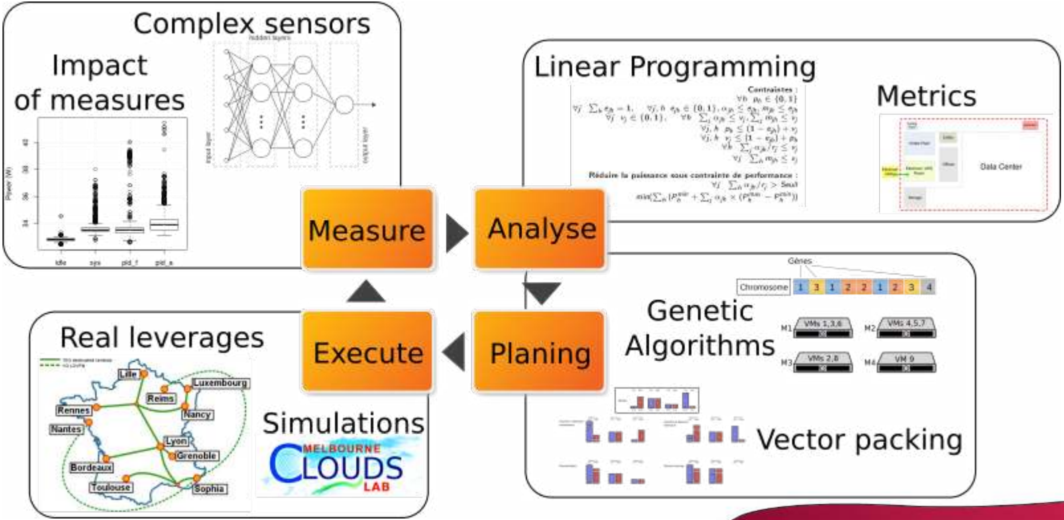
- Two goals:
  - Managing (needs, errors, faults, overheating)
  - Optimizing (Energy, performance)
- Leverages
  - Switching on/off, DVFS
  - Migration (x86/ARM)*, reduction of allocated resources, suspend
- Methods
  - Often in the real world: Humans or rules
  - In research: autonomic loop




MAPE-K loop ©IBM

* Violaine et al., *Big, Medium, Little : Reaching Energy Proportionality with Heterogeneous Computing Scheduler*, Parallel Processing Letters, journal, 2015.

# Autonomic loop



# Outline: How to efficiently manage a datacenter

- Efficiently?
  - It is necessary to be able to compare (**models & metrics**)
- Managing means deciding
  - **Measure** tools
  - **Evaluation tools** : Experiments, simulation
  - Exact approaches and heuristics for **decision**
- **Evolution** of datacenters
  - Datacenter federations
  - Multi-levels optimization

# Plan

# Model a system

To manage a system, we need to:

- Know all possible actions
- Know which is(are) the best one(s)

It can be translated into:

- Modeling impact and means (time, energy,...) of these actions
- Being able to compare two scenarios

# Impact of leverages, an example with DVFS

**Dynamic electric power consumed by a CMOS component:**

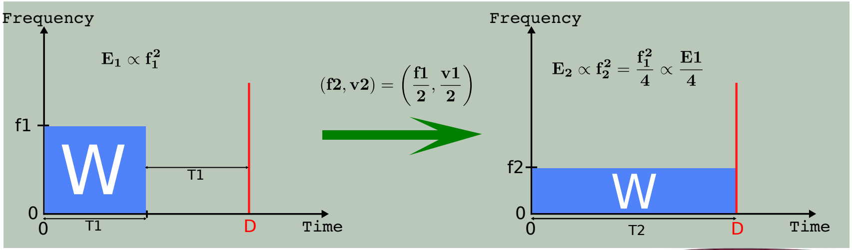
$P_{cmos} = C_{eff} \times V^2 \times f$

with, $C_{eff}$ the effective capacitance *, $V$ the voltage and $f$ the frequency

* physical quantity: capacity of a component to resist to the change of voltage between its pins

**Energy consumed for each tasks:**

$$E = P * T \propto T * V^3 \text{ , avec } V \propto f \text{ et } T \propto 1/F\text{, alors } E \propto f^2$$



# Even models are complex

Electrical power models for a single server:

- Classical : linear (error E∼10-15%)

$$Power = P_{min} + Load \times (P_{max} - P_{min})$$

# Even models are complex

Electrical power models for a single server:

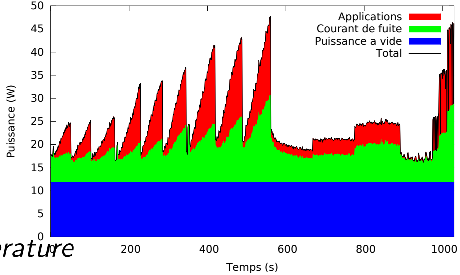
- Classical : linear (error E∼10-15%)
- Finer : Processor voltage/frequency (E∼5-9%)

$$Power = P_{min} + Load \times \alpha Voltage^2 Frequency$$

# Even models are complex

Electrical power models for a single server:

- Classical : linear (error E∼10-15%)
- Finer : Processor voltage/frequency (E∼5-9%)
- Even finer: Processor temperature (E∼4-7%)



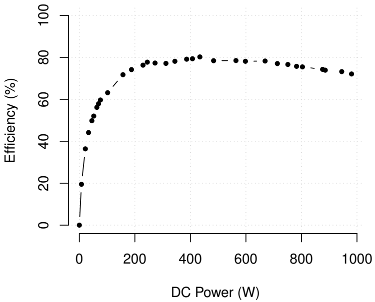

$$Power = P_{min} + Load \times \alpha Voltage^2 Frequency + \lambda Temperature$$

# Even models are complex

Electrical power models for a single server:

- Classical : linear (error E∼10-15%)
- Finer : Processor voltage/frequency (E∼5-9%)
- Even finer: Processor temperature (E∼4-7%)
- Do not forget about bias: **power supply unit E∼2-3%**, cooling, ...

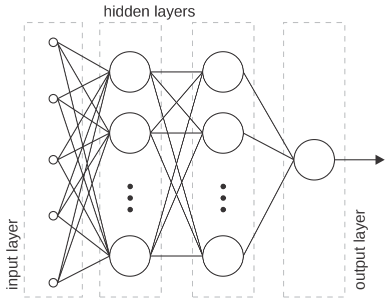
$$Power_{DC} = \omega_0 + \omega_1 Power_{AC} + \omega_2 Power_{AC}^3$$



# Even models are complex

Electrical power models for a single server:

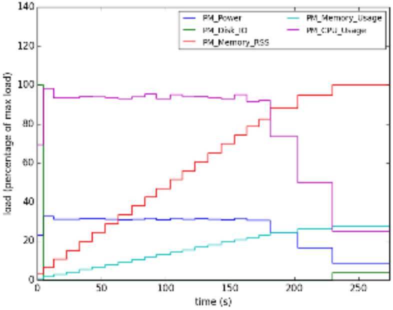
- Classical : linear (error E∼10-15%)
- Finer : Processor voltage/frequency (E∼5-9%)
- Even finer: Processor temperature (E∼4-7%)
- Do not forget about bias: **power supply unit E∼2-3%**, cooling, ...
- Learning methods (neural networks, E∼2%) *




* Leandro et al., *Towards a generic power estimator*, CSRD journal, 2015

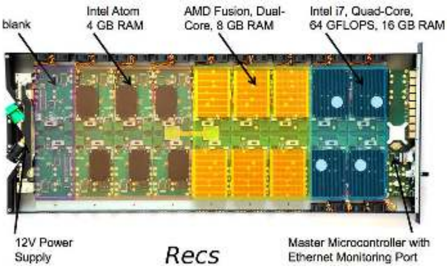
# Modeling a datacenter is a complex task

- Large number of elements
  - Applications
    - Process: Traces, high-level monitoring then abstraction



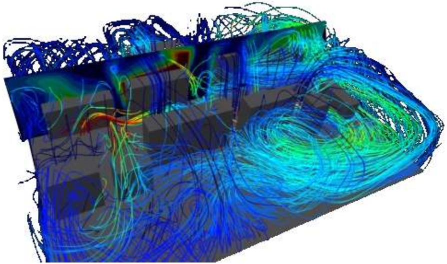
# Modeling a datacenter is a complex task

- Large number of elements
  - Applications
    - Process: Traces, high-level monitoring then abstraction
  - Servers



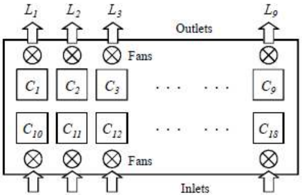
# Modeling a datacenter is a complex task

- Large number of elements
  - Applications
    - Process: Traces, high-level monitoring then abstraction
  - Servers
  - Infrastructure



# Modeling a datacenter is a complex task

- Large number of elements
  - Applications
    - Process: Traces, high-level monitoring then abstraction
  - Servers
  - Infrastructure
- And their interactions
  - Thermal (D-Matrix)*
  - Between applications

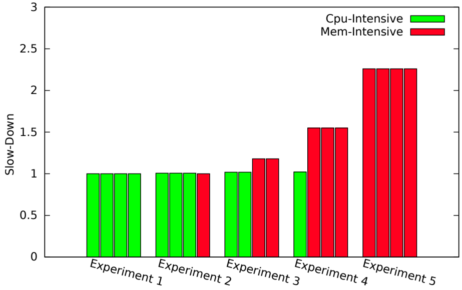



$$d_{x,k} = \begin{cases} 1, & \text{if } x = k \\ 0.84, & \text{if } x = k + 9 \\ 0, & \text{otherwise} \end{cases}$$

* Hong Yang et al.,*Energy-efficient and thermal-aware resource management for heterogeneous datacenters*, SUSCOM journal, 2014.

# A "simple" interaction of applications

- Two types of mono-thread applications
  - Application 1 : Cpu-Intensive, limited by the processor
  - Application 2 : Mem-Intensive, limited by memory
- Execution on a quad-core
  - Applications 1 : Independent
  - Applications 2 : Strong cross-impact



# Metrics : A complex landscape

- HPC
  - Improve performance, throughput
  - Steady and known workload
- Cloud systems
  - Improve cost efficiency
  - Varying workload, difficult to predict
- Two main questions :
  - How to program*them at large scale?
  - How to manage them at runtime?

* Georges et al. *Exascale machines require new programming paradigms and runtimes*, SFI journal, 2015

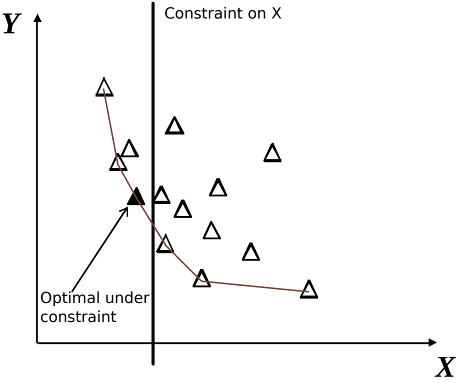
# Metrics

- Direct values:
  - Processor and memory load, power, temperature,...
  - Objective: Does the system works? Comparing two datacenters, middleware, software,...
- 40000 servers, 500000 services → Need of simple metrics
  - Consumption **and** performance
- Difficult to standardize, mainly performance
  - Depends of the service, its implementation,...
- Classical metric: PUE

Georges et al. *Data Centres Sustainability Cluster Activities Task 3*. Rapport de recherche 3. European Commission, 2014

# Problem of multi-objective

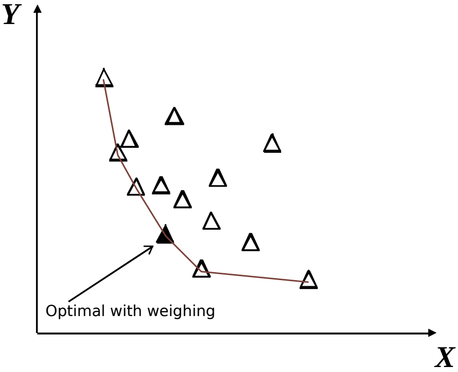
- Impossible to define in absolute, need a context, a goal
- Formalize simple metrics* : Dynamism, Energy, Performance, Resilience
- Several classical methods
  - Constraint optimization




* Tom et al., *Quality of Service Modeling for Green Scheduling in Clouds*, SUSCOM journal, 2014.

# Problem of multi-objective

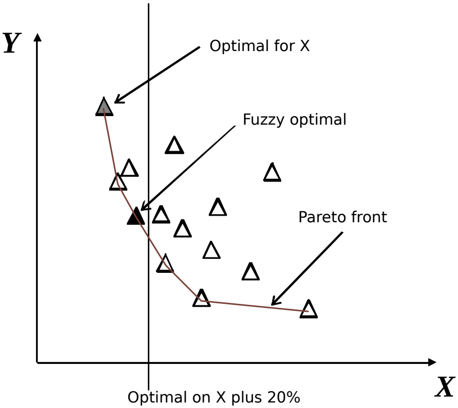
- Impossible to define in absolute, need a context, a goal
- Formalize simple metrics* : Dynamism, Energy, Performance, Resilience
- Several classical methods
  - Constraint optimization
  - Objective weighing




* Tom et al., *Quality of Service Modeling for Green Scheduling in Clouds*, SUSCOM journal, 2014.

# Problem of multi-objective

- Impossible to define in absolute, need a context, a goal
- Formalize simple metrics* : Dynamism, Energy, Performance, Resilience
- Several classical methods
  - Constraint optimization
  - Objective weighing
  - Fuzzy weighing† (Constraining by relaxation of optimal)




† Hong Yang et al. *Energy-efficient and thermal-aware resource management for heterogeneous datacenters*, SUSCOM journal, 2014.

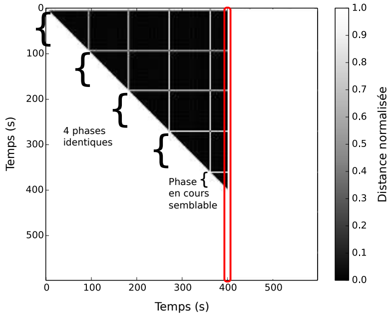
# Measure Infrastructure

- Basis for taking decision
- Basis for metric evaluation
  - Classical Infrastructure (*nagios*, *ganglia*, ...)
  - Problem for scaling
    - Most values are unused of aggregated late
  - Some measures (processor, memory), but no **knowledge**
    - Need of higher level measures
    - What type of (phase of an) application
    - Electric power consumed by applications

# Which (phase of an) application is running

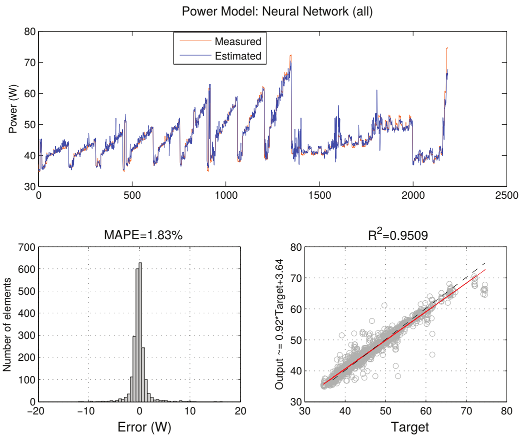
- A phase : behavior locally regular
  - Equivalent as a constant resource consumption
  - System measures constants
- Detection then identification
  - Signature of a phase
  - Same Phase ∼ Same Impact

Landry et al., *Exploiting performance counters to predict and improve energy performance of HPC systems*, FGCS journal, 2014.

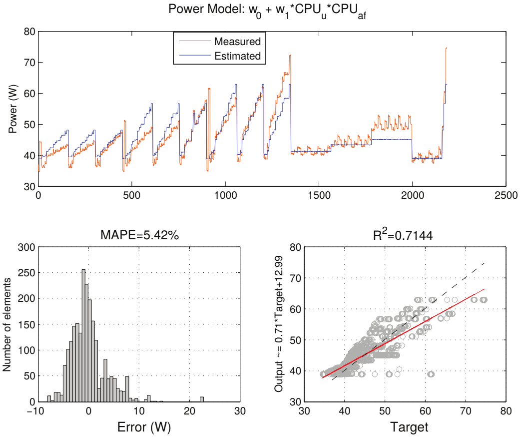



Matrix of similar system measures (WRF : Weather Research and Forecasting)

# Power of servers and applications




**(a)** ANN regression using all KPIs.




**(b)** Calibrated capacitive model.

# Evaluation tools: Experimentation, Simulation

- To improve, comparison is necessary
- Three main methods
  - Mathematical models
  - Simulation
  - Experiments

# Linear programming

- Describe all constraints with linear equations

**Example : A task is on a unique server**

- Let $e_{jh}$ the fact that task $j$ runs on server $h$
- $e_{jh} = 1$ iif task $j$ is on server $h$
- $\forall j, h \quad e_{jh} \in \{0, 1\}$,
  $\forall j \quad \sum_h e_{jh} = 1$

# Linear programming

- Describe all constraints with linear equations
- Describe the objective as a function to minimize

**Example : Minimize the total power consumed**

- $P_h^{stat}$ et $P_h^{dyn}$ : static and dynamic power of server $h$ (linear model)
- Let $\alpha_{jh}$ the processor fraction of task $j$ on server $h$
- $min \sum_h (P_h^{stat} + \sum_j \alpha_{jh} P_h^{dyn})$

# Linear programming

- Describe all constraints with linear equations
- Describe the objective as a function to minimize
- Formalize leverages and their impact
- Approximation of real world (quadratic phenomena)
- Exact resolution for small cases

Constraints :

$$\forall h \quad p_h \in \{0,1\}$$
$$\forall j \quad \sum_h e_{jh} = 1, \quad \forall j,h \quad e_{jh} \in \{0,1\}, \alpha_{jh} \leq e_{jh}, m_{jh} \leq e_{jh}$$
$$\forall j \quad v_j \in \{0,1\}, \quad \forall h \quad \sum_j \alpha_{jh} \leq v_j, \sum_j m_{jh} \leq v_j$$
$$\forall j,h \quad p_h \leq (1 - e_{jh}) + v_j$$
$$\forall j,h \quad v_j \leq (1 - e_{jh}) + p_h$$
$$\forall h \quad \sum_j \alpha_{jh}/r_j \leq v_j$$
$$\forall j \quad \sum_h m_{jh} \leq v_j$$

Minimize power under performance constraints:

$$\forall j \quad \sum_h \alpha_{jh}/r_j > Threshold$$
$$min(\sum_h (P_h^{min} + \sum_j \alpha_{jh} \times (P_h^{max} - P_h^{min}))$$

Damien et al., *Energy-Aware Service Allocation*, FGCS journal, 2011

# Simulation

- Large number of simulators: SimGrid, DCWorms, CloudSim, ...
- Particular needs for our research
  - Cloud models (migration, Over-allocation of resources, federation†)
  - DVFS
  - Electrical power
  - Temperature
- Situation is steadily improving
  - DVFS and fine-grained management of clouds in CloudSim
  - Thermal simulation in DCWorms*
  - DVFS and energy in SimGrid

* Wojtek et al., *Energy and thermal models for simulation of workload and resource management in computing systems*, SMPT journal, 2015. † Thiam et al., *Cooperative Scheduling Anti-load balancing Algorithm for Cloud*, CCTS workshop, 2013

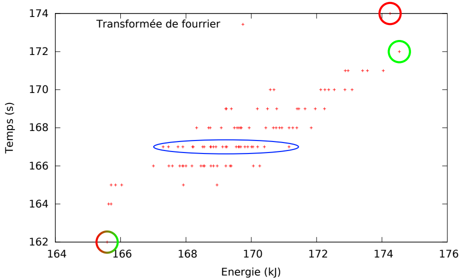
# Experimentation

- A model is always an approximation
- Final validation by experiment
- Complex because of the need to have electrical measures
  - At ENS-Lyon, they where one of the first to experiment with watt-meters at large scale (GreenNet)*
- Problem of distributed measures, electrical conversions, impact of measures (performance counters)
- Reproducibility problem

* Da Costa, *The green-net framework: Energy efficiency in large scale distributed systems*, IPDPS, 2009

# No stability of experiments

- *Simple* experiment of Fast Fourier Transform (NPB)
- 100 experiments on exactly the same hardware (Grid'5000)
- Large variations
  - Time: 12s, 7% (Std. Dev. 3.2s)
  - Energy: 9.3kJ, 5.5% (3kJ)
- For the same time, 167s
  - Difference of 4kJ
- Time ≠ Energy



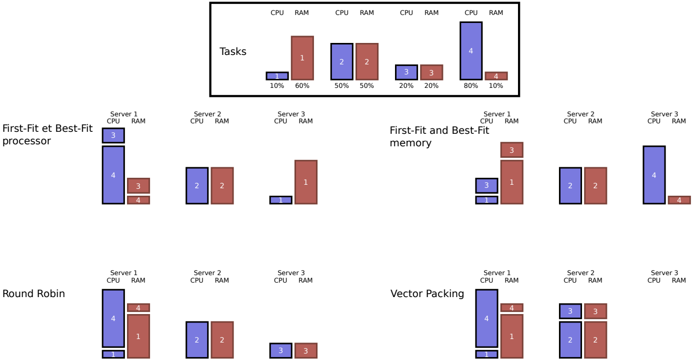
# Plan

# Exact Approaches and heuristics

- Two problems
  - Placement
  - Temporality
- Classical heuristics for placement
  - Greedy: Best Fit, First Fit
  - Vector Packing (*Gourmet* Greedy)
  - Genetic algorithms

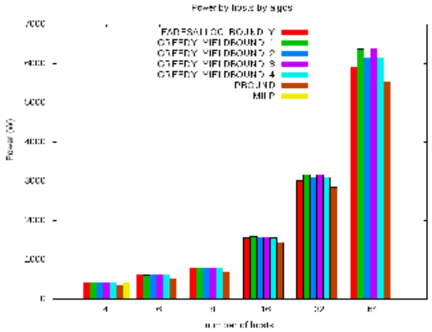
# Classical greedy algorithms

- Characteristics
  - Memory
  - Processor
- Sort services
- Sort servers
- No coming back on previous decisions



# *Gourmet* Vector packing

- 4 objectives in the sort function
  - Server is attractive from an energy point of view
  - Add the task do not overload the server
  - Server already switched on
  - The tasks brings back the balances of resources
- Time "only"in $\mathcal{O}(J \times H \ln(H))$
- But the solution of the *Gourmet* is difficult to qualify

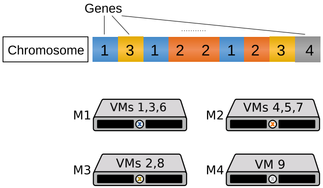


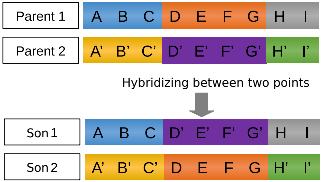

Damien et al., *Energy-Aware Service Allocation*, FGCS journal, 2012.

# Genetic Algorithms

- Chromosome = Allocation
- Initial random generation
- At each generation:
  - Hybridizing and mutation
  - Sort on the objective metric
  - Keep only the best

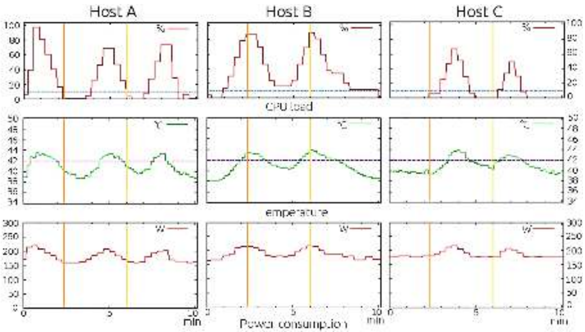
Tom et al., *Quality of Service Modeling for Green Scheduling in Clouds*, SUSCOM journal, 2014



# Why power/energy is unique

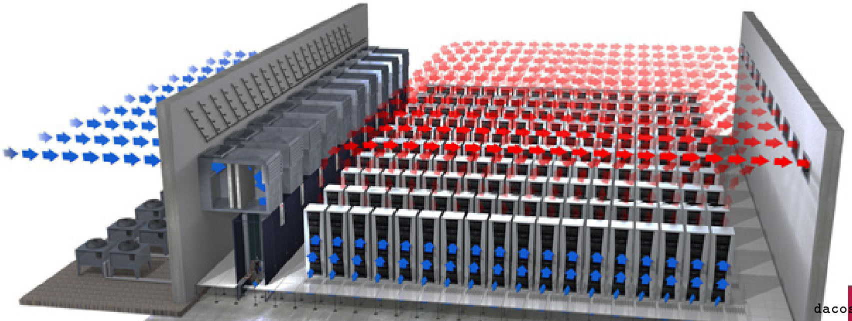
- The temporal point of view
  - Inertia due to temperature
  - Switching on/off servers
    - Over- or Under-reservation
  - Cycles are sometime good
- Non-linearities
  - Equation of power
- Feedback loops
  - Cooling system




Violaine et al., *Thermal-aware cloud middleware to reduce cooling needs*, WETICE workshop, 2014

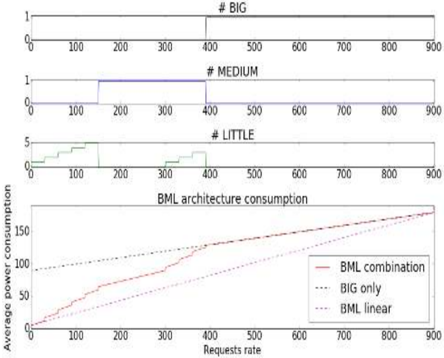
# Follow the sun, Follow the moon

- Classical approach
  - Consolidation between datacenters
  - Coordinated management of Quality of Service (ex: CDN)
- Follow the state of datacenters
  - During night, less cooling cost
  - During day, more renewable energy production



# At the scale of the *Data center in the box*

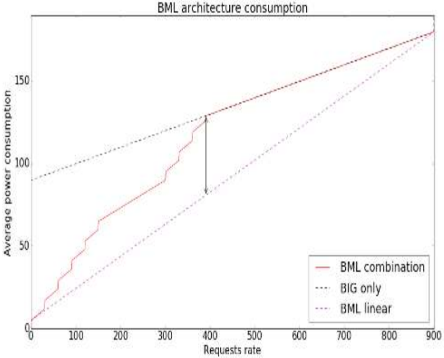
- Rack level
- Low number of services: High variance
- Perfect adaptation to the load
- Currently costly: Initial overhead
  - Cooling
  - Everything which is negative in the PUE
- *Proportional* architecture




Violaine et al., *Big, Medium, Little : Reaching Energy Proportionality with Heterogeneous Computing Scheduler*, Parallel Processing Letters, journal, 2015.

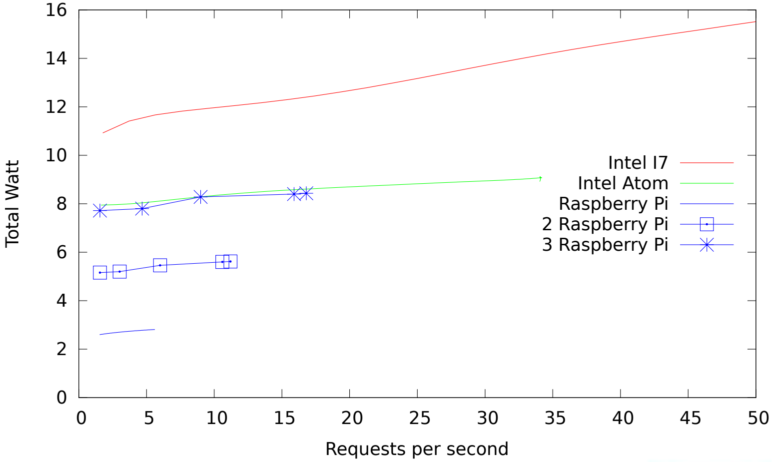
# At the scale of the *Data center in the box*

- Rack level
- Low number of services: High variance
- Perfect adaptation to the load
- Currently costly: Initial overhead
  - Cooling
  - Everything which is negative in the PUE
- *Proportional* architecture

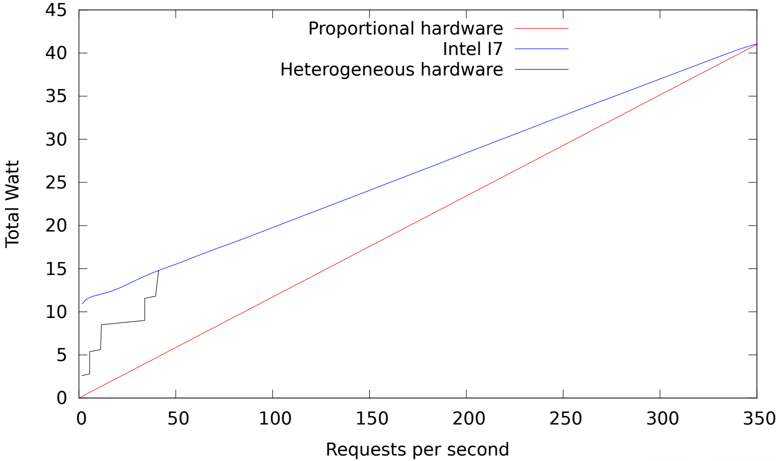



Violaine et al., *Big, Medium, Little : Reaching Energy Proportionality with Heterogeneous Computing Scheduler*, Parallel Processing Letters, journal, 2015.

# A virtual computer based on Atom, I7 and Raspberry

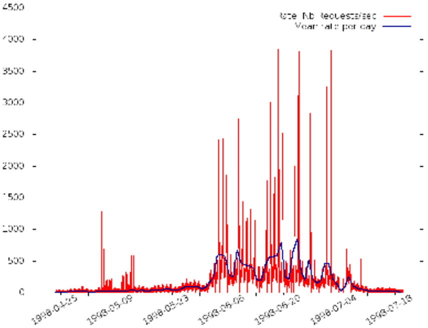


# Still far far away...

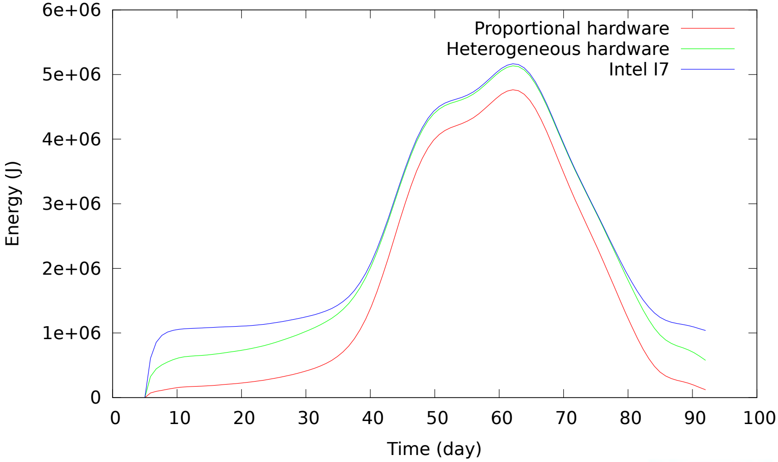


# 98 Football World Cup

- Available data
- 92 days of web server access logs
- Workload precise at the second level
- Four geographic locations, three in US, one in France
- Several phases
  - Low phases, first 40 days and last 10 days
  - High phase, during the competition

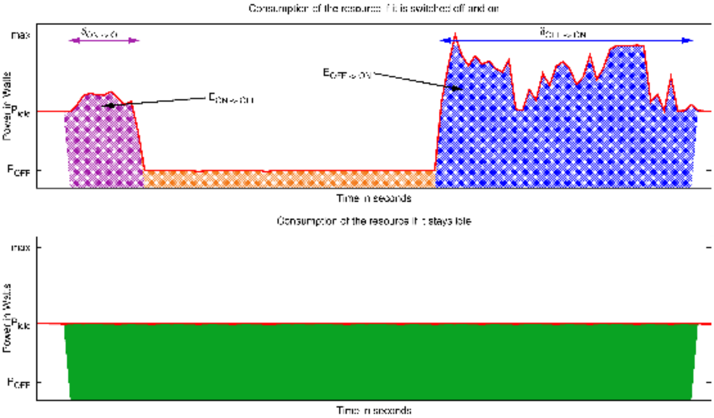


# Comparison if using one single data-center



# Adding management heterogeneity

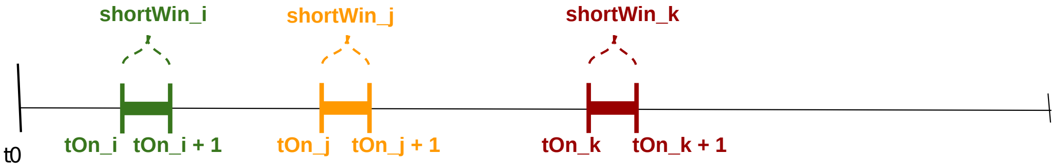
- Switching on/off nodes take time
- Switching on/off nodes consumes energy
    - For application reconfiguration
    - For switching on/off the server




Lefevre and al., Supercomputing 2008

# Several methods to manage servers

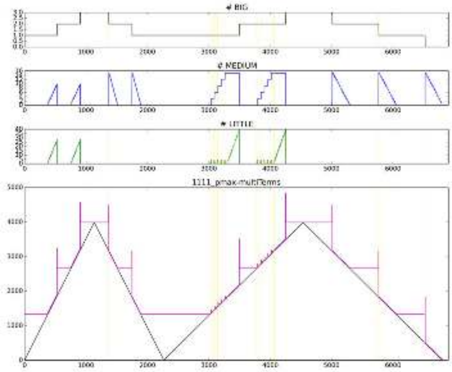
- Exact approach, linear programming
- Heuristics
  - Reactive
    - When overloaded, start new servers, when under-loaded stop some
  - Pro-active
    - Predict the future and decide which server to use




Violaine et al. *Energy Aware Dynamic Provisioning for Heterogeneous Data Centers*, SBAC-PAD 2016

# Pro-active Heuristics

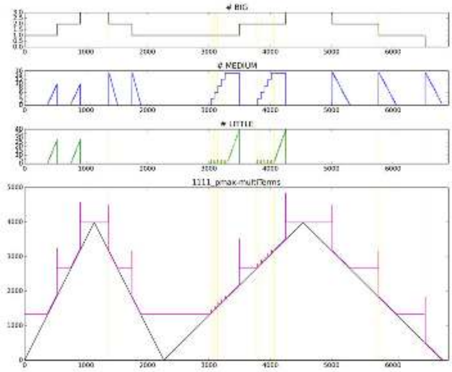
- Predict load and switch on nodes to guaranty QoS




Violaine et al., *Energy Proportionality in Heterogeneous Data Center Supporting Applications with Variable Load*, ICPDS 2016

# Pro-active Heuristics

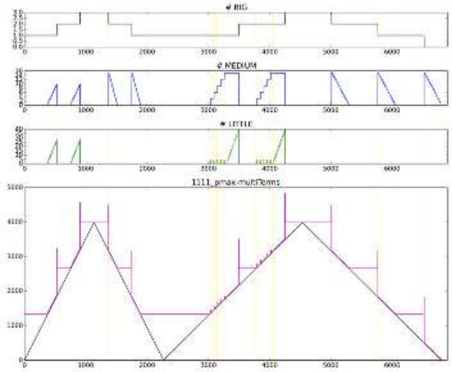
- Predict load and switch on nodes to guaranty QoS
- Switching on can be followed by switching off




Violaine et al., *Energy Proportionality in Heterogeneous Data Center Supporting Applications with Variable Load*, ICPDS 2016

# Pro-active Heuristics

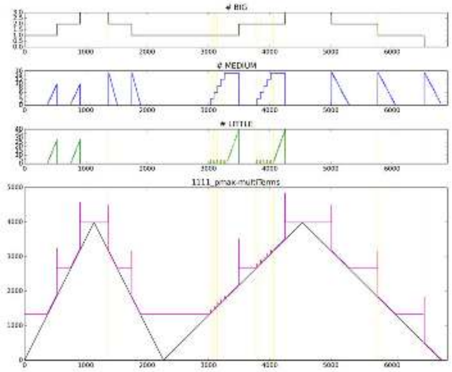
- Predict load and switch on nodes to guaranty QoS
- Switching on can be followed by switching off
- When load decrease, switch off servers accordingly




Violaine et al., *Energy Proportionality in Heterogeneous Data Center Supporting Applications with Variable Load*, ICPDS 2016

# Pro-active Heuristics

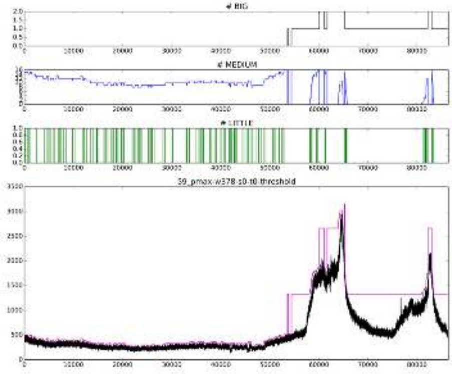
- Predict load and switch on nodes to guaranty QoS
- Switching on can be followed by switching off
- When load decrease, switch off servers accordingly
- Stay near the optimal repartition




Violaine et al., *Energy Proportionality in Heterogeneous Data Center Supporting Applications with Variable Load*, ICPDS 2016

# Pro-active Heuristics

- Predict load and switch on nodes to guaranty QoS
- Switching on can be followed by switching off
- When load decrease, switch off servers accordingly
- Stay near the optimal repartition




Violaine et al., *Energy Proportionality in Heterogeneous Data Center Supporting Applications with Variable Load*, ICPDS 2016

# Plan

# Toward the future

- A larger number of datacenters
  - Lots of smaller ones (hybrid management)
    - The knowledge: critical resource
  - A large number of diverse sizes
  - Some larger (2016 : 6 300 000 $m^2$)
- Overall, datacenters will be more integrated in their environment
  - Electrical aspects
  - Thermal aspects

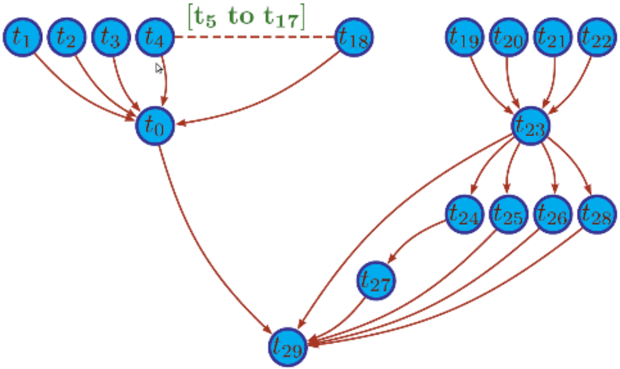
# At the level of a node

- Three temporality
  - Large-grained (minute) : Optimal frequency in function of the task graph*
    - 13% of energy savings
  - Medium-grained (second) : Phase detection†
    - 20% of energy savings, 3% of time increase
  - Fined-grained (1/10s) : Frequency policy at the kernel level‡
    - 25% of energy savings, 1% of time decrease
- No coordination between the three temporality, no objectives

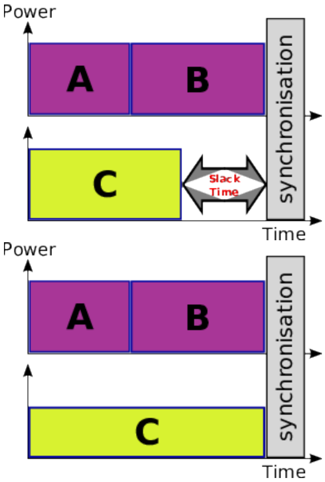
* Tom et al., *Energy-aware simulation with DVFS*, SMPT journal, 2013 †Landry et al., *Exploiting performance counters to predict and improve energy performance of HPC systems*, SUSCOM journal, 2014 ‡Georges et al., *DVFS governor for HPC: Higher, Faster, Greener*, PDP conférence, 2015

# At the scale of a node: Large-grained

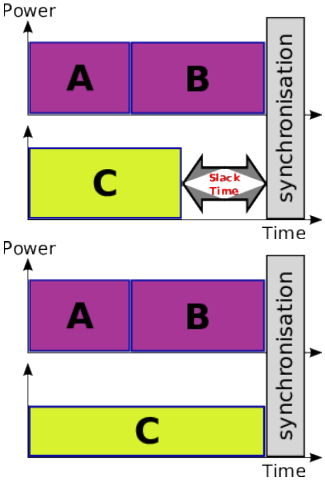
- Use of contextual external information
- Example at the scheduler level: Task DAG



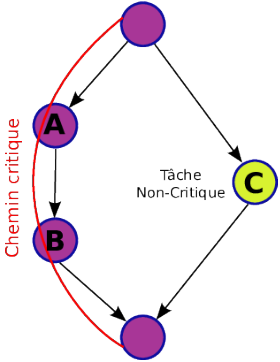
# Coordination of node speeds



# Coordination of node speeds




- Generalization toward critical path



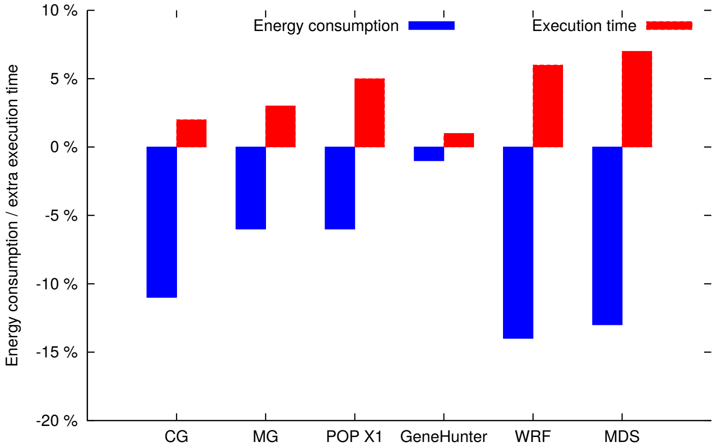
# At the level of a node: Medium-grained

- React at medium latency at the level of the node
  - Change the processor frequency
  - Change the hard drive mode
  - Reconfigure the network card
- Detection of the current phase
- React in function of this profile
- Light impact on the infrastructure

# Decision method

| Phase label | Possible reconfiguration decisions |
|---|---|
| compute-intensive | switch off memory banks; send disks to sleep; scale the processor up; put NICs into LPI mode |
| memory -intensive | scale the processor down; decrease disks or send them to sleep; switch on memory banks |
| mixed | switch on memory banks; scale the processor up send disks to sleep; put NICs into LPI mode |
| communication intensive | switch off memory banks; scale the processor down switch on disks |
| IO-intensive | switch on memory banks; scale the processor down; increase disks, increase disks (if needed) |

# Energy and performance, 28 node



# Fine-grained = DVFS ?

## Relative values between *performance* and *ondemand* governors

| Benchmark | FT | SP | BT | EP | LU | IS | CG |
|---|---|---|---|---|---|---|---|
| Time increase (%) | 0 | -3 | -1 | 1 | -2 | 2 | 0 |
| Energy increase (%) | 0 | -3 | -1 | -1 | -2 | -1 | -1 |

- HPC applications are rarely in Idle... Surprise !
- MPI libraries are spinning

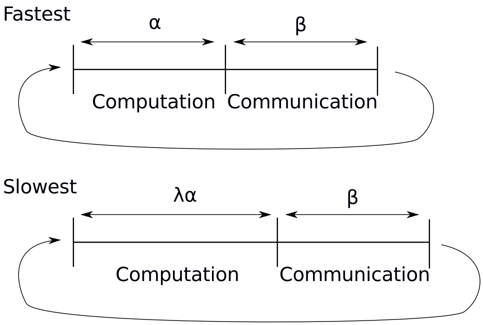
Classical HPC benchmarks from NPB (Nas Parallel Benchmark)

# HPC Hypothesis

- State of applications
  - Computing
  - Communications
  - Disk I/O
  - Idle

# HPC Hypothesis

- State of applications
  - Computing
  - Communications



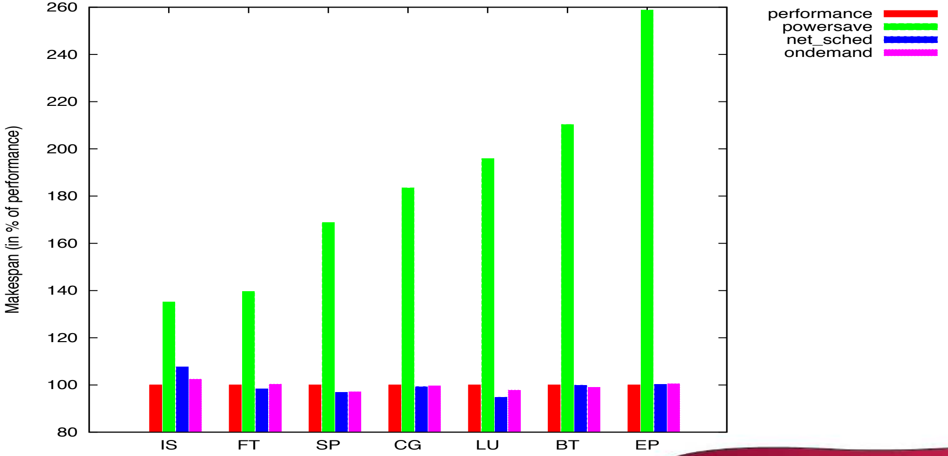
# Adding an hysteresis for adding inertia

**NetSched Algorithm**

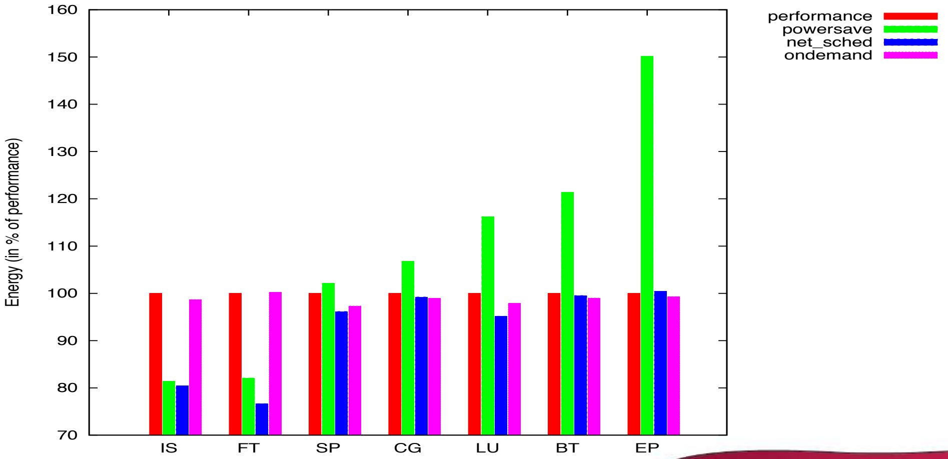
- Each $10^{th}$ of a second, do:
  - If Current_Frequency = Slowest frequency and IBR $\leq .9B_1$
    - Change frequency toward Fastest
  - If Current_Frequency = Fastest frequency and IBR $\geq 1.1B_2$
    - Change frequency toward Slowest
  - Else, do nothing

IBR : Incoming Byte Rate

# Results: Makespan



# Results: Energy-to-solution



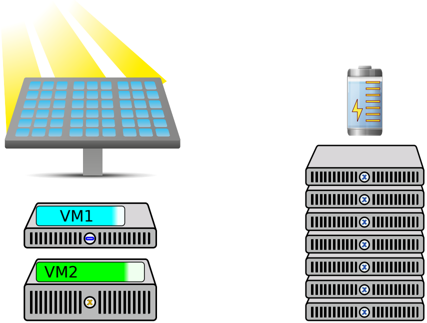
# Plan

# The missing link between levels

- An “handmade”work
  - A large number of inter-dependent middlewares
  - Human manipulations
- Toward a decentralized cooperation

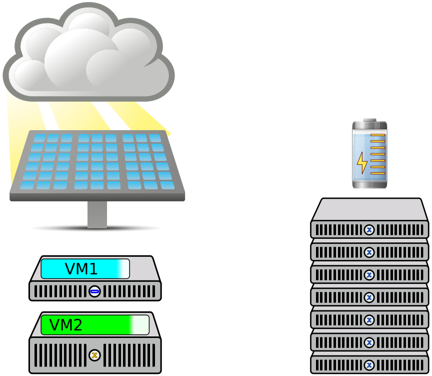
# Cooperation between decision levels

1. Initial situation is stable



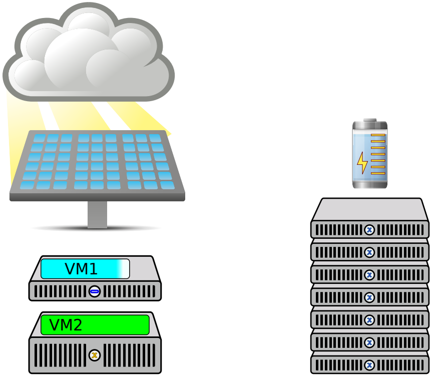
# Cooperation between decision levels

1. Initial situation is stable
2. Decrease of solar production



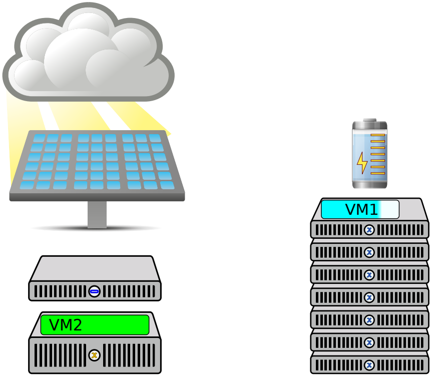
# Cooperation between decision levels

1. Initial situation is stable
2. Decrease of solar production
3. Non-critical task: Aggressive DVFS
3. Critical task: unavailable dynamism



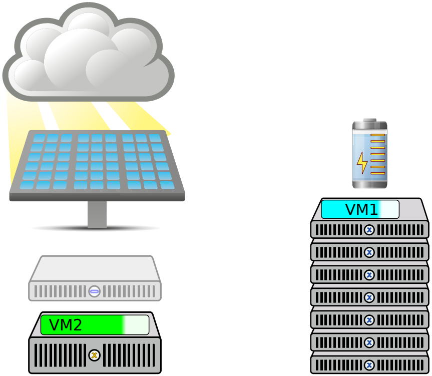
# Cooperation between decision levels

1. Initial situation is stable
2. Decrease of solar production
3. Non-critical task: Aggressive DVFS
3. Critical task: unavailable dynamism
4. Critical task: looks for an adequate location



# Cooperation between decision levels

1. Initial situation is stable
2. Decrease of solar production
3. Non-critical task: Aggressive DVFS
3. Critical task: unavailable dynamism
4. Critical task: looks for an adequate location
5. Switching off a server
6. Less aggressive DVFS



# Open research questions

- Programming paradigms
  - Ability to describe parallelism intuitively
  - Remove the burden from developer
- Runtimes
  - Capability to adapt to particular profiles and their interactions
  - Ability to change kernels in function of context
- Communication between these two levels
- Cooperation between operators
  - Cloud federation
  - Cloud and HPC systems