Introduction to Scheduling of parallel computer systems (clusters, grids, and clouds)

Andrei Tchernykh

CICESE Research Center, Ensenada, Baja California, México

chernykh@cicese.mx

http://usuario.cicese.mx/~chernykh/

- Head of “Parallel Computing Laboratory” at CICESE, Mexico
- Head of “Laboratory of Problem-Oriented Cloud Computing” at South Ural State University. Chelyabinsk, Russia.
- Adjunct professor of Ivannikov Institute for System Programming of the Russian Academy of Sciences, Moscow, Russia

São Paulo University, Institute of Mathematics and Statistics (IME),

São Paulo, Brasil, April 19-26, 2023

Importance

Modern distributed computer systems offer fundamentally new opportunities to increase computing power

- scalability,
- ability to flexibly manage the load,
- reliability and fault tolerance,
- extensibility,

etc.

- But there is significant instability during resource access and utilization.
- This creates additional challenges
    - for end users, resource providers, service providers, and scheduling systems.

Importance

Imperfect methods and models of job management

- lead to a **significant underutilization** of the capabilities of computing systems and high energy consumption.

Scheduling can

- **Ensure resource efficiency,**
- **overcome the negative consequences of non-stationarity,**

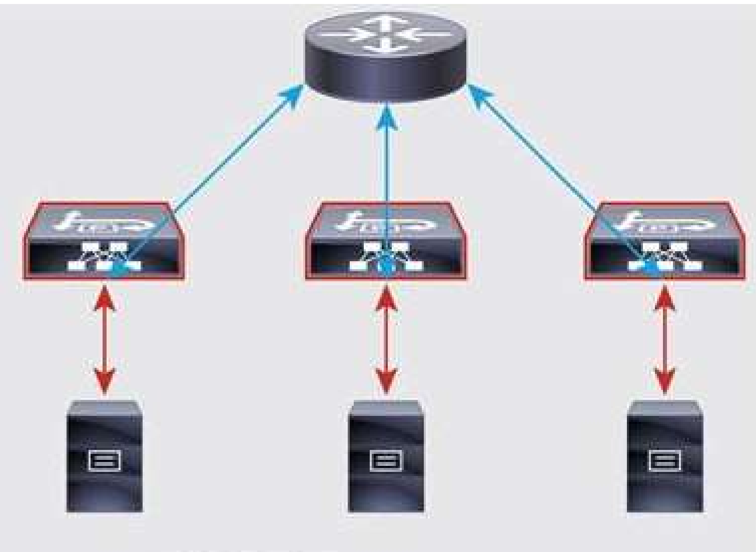
,

We need

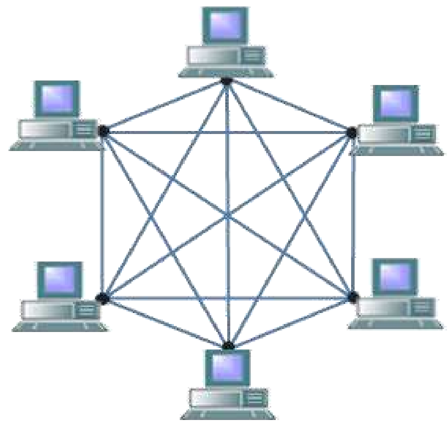
- scientific fundamentals of nonstationary resource scheduling ,
- mathematical models that consider the lack of accurate knowledge in the formation of the work plan.
- development of new adaptive algorithms for various scenarios.

Data Centers

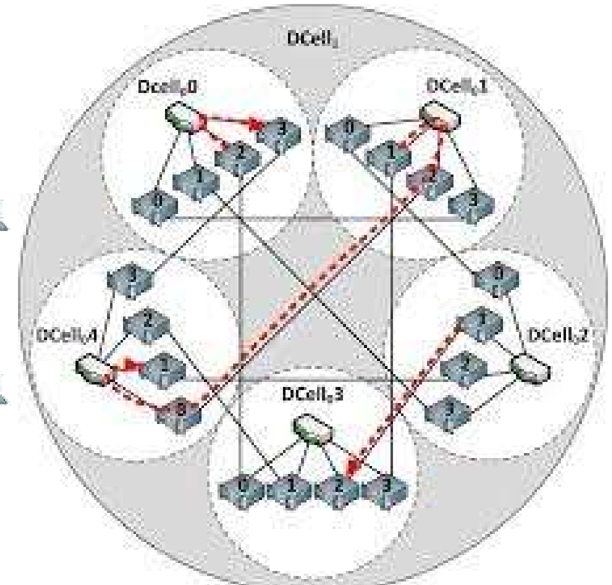
Switch-centric



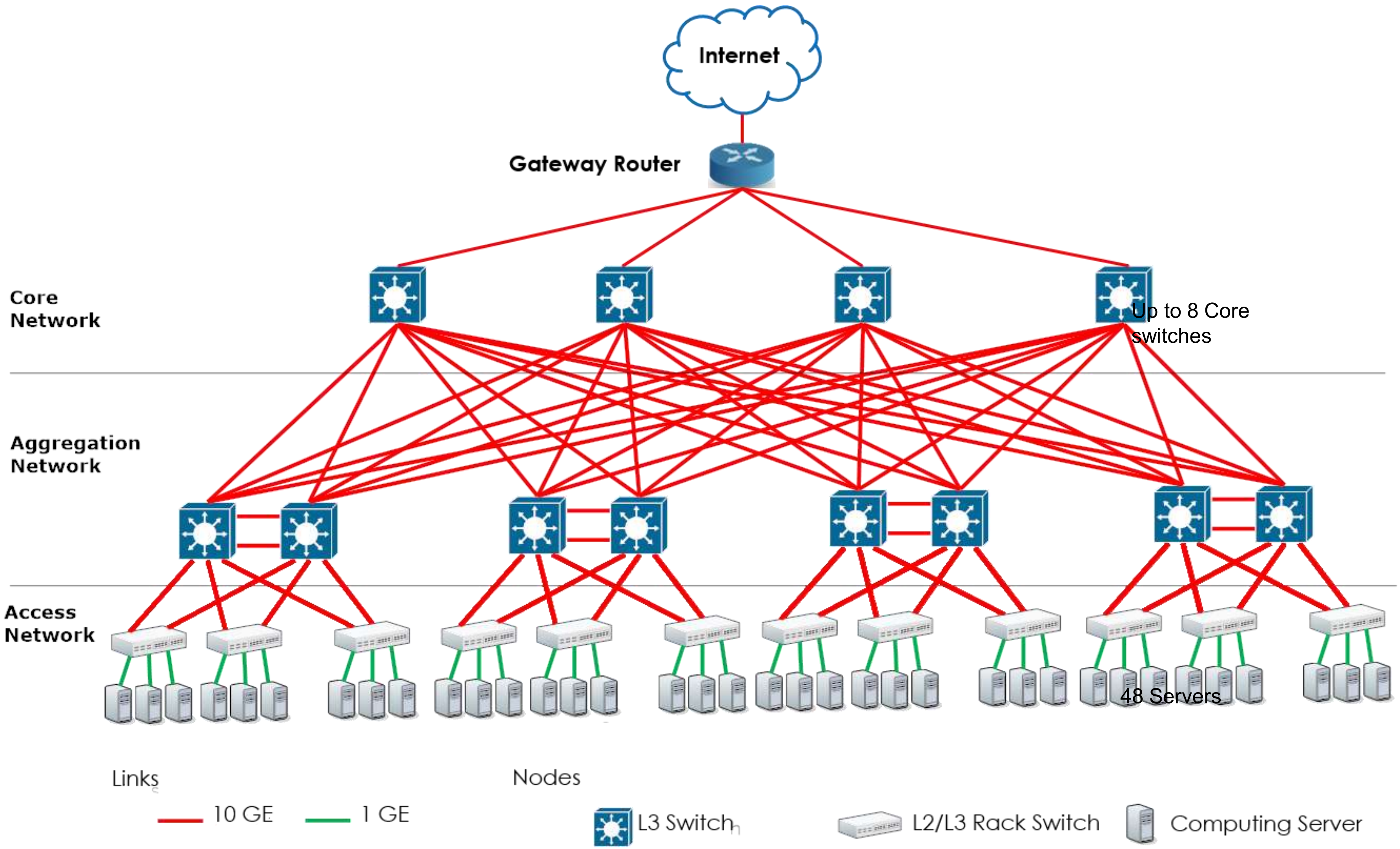
Server-centric



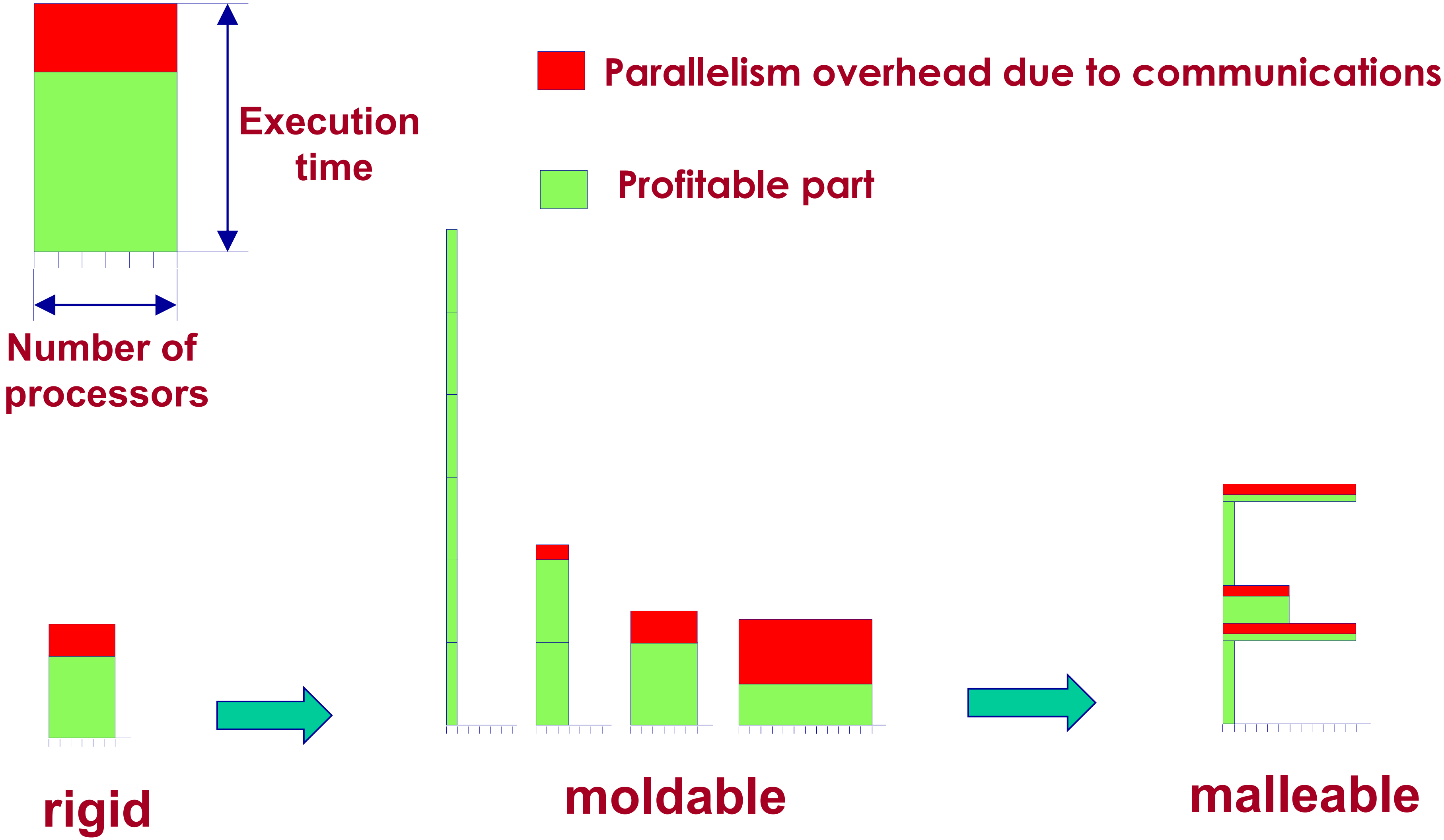
Hybrid



Three-tier topology

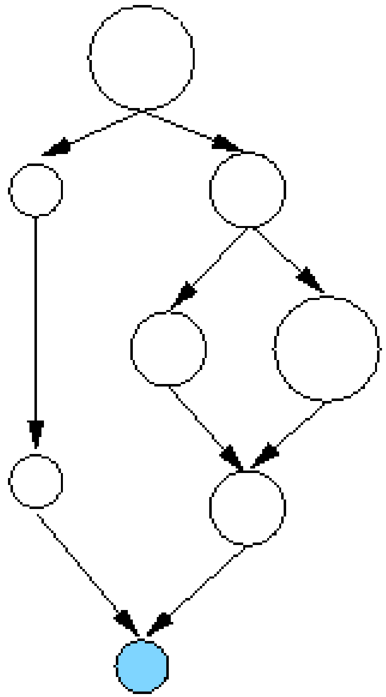


Scheduling: type of jobs

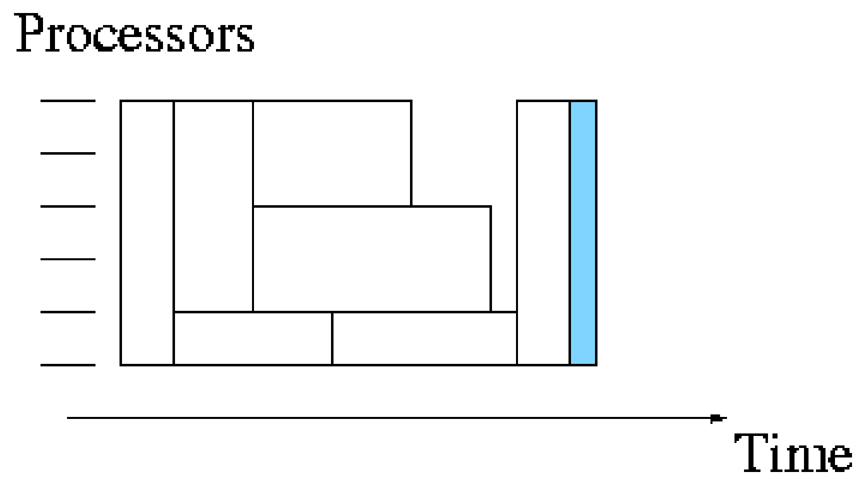


Parallel Tasks Scheduling

by Denis Trystram



MT Graph



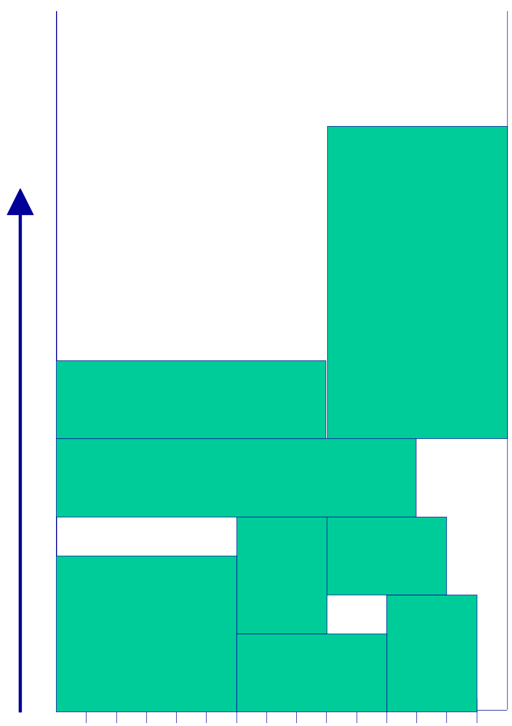
MT Scheduling

Scheduling: on-line vs off-line

by Denis Trystram

On-line: no knowledge about the future

We take the scheduling decision while other jobs arrive

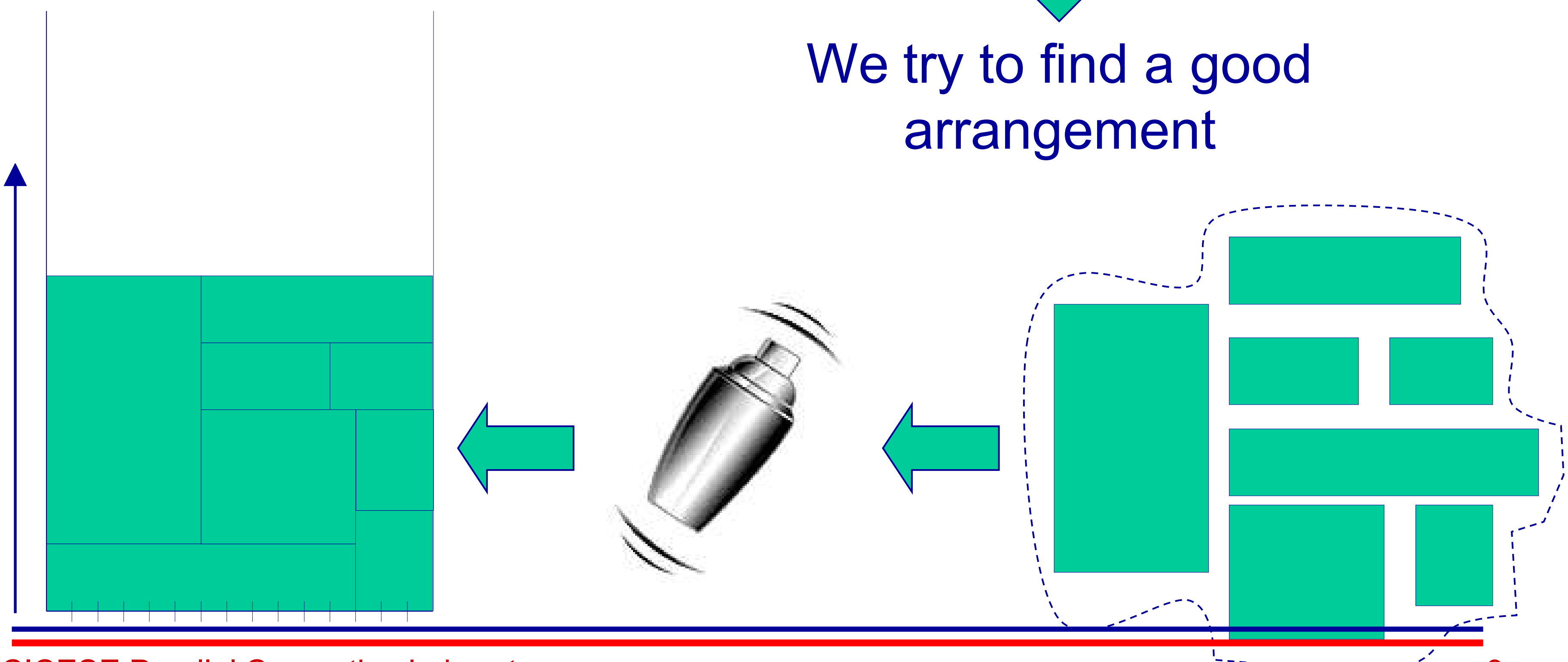


Scheduling: on-line vs off-line

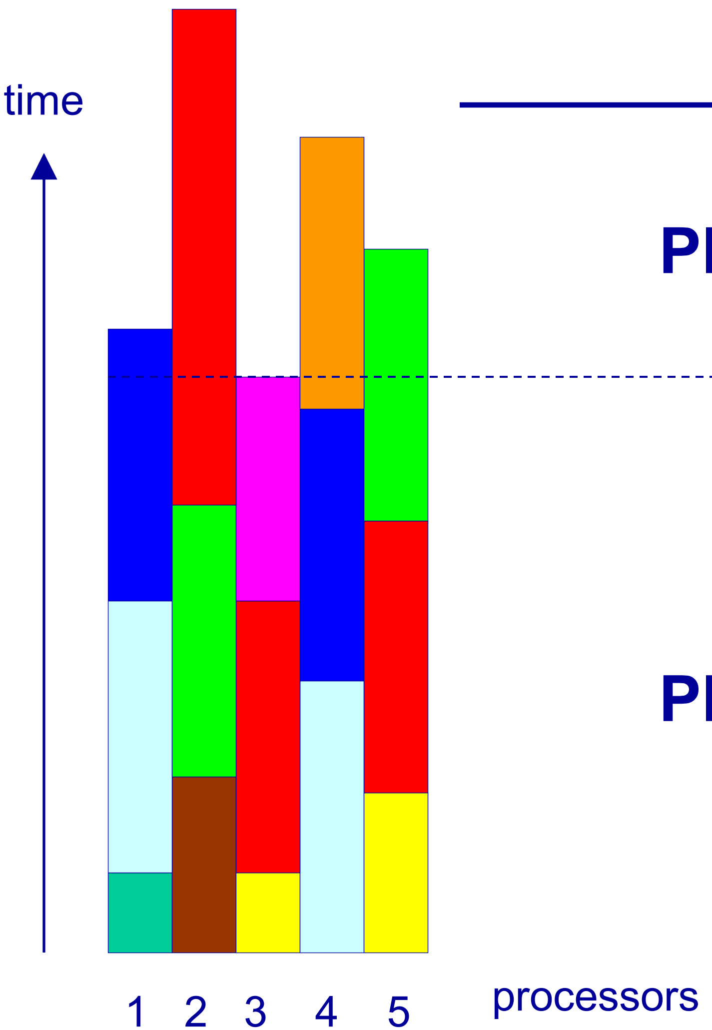
by Denis Trystram

Off-line: we have a finite set of works

We try to find a good arrangement



A generic scheme



List of jobs

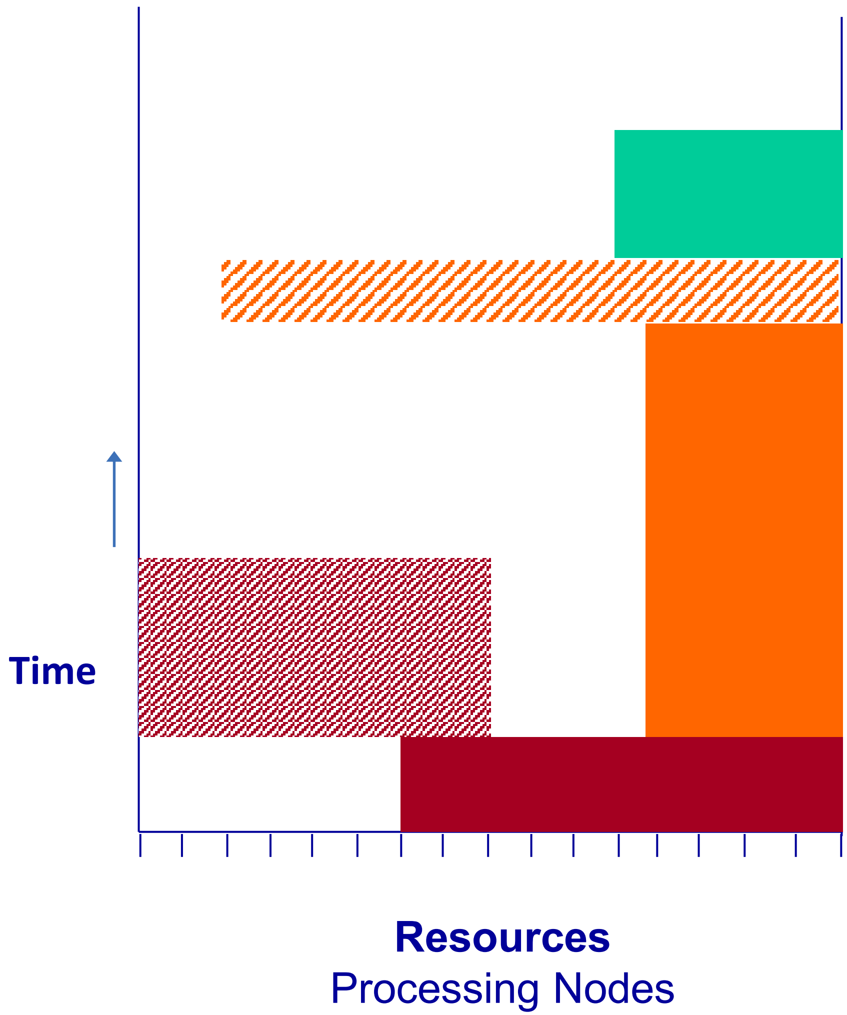
Phase 2 rigid, moldable or malleable
less than m jobs

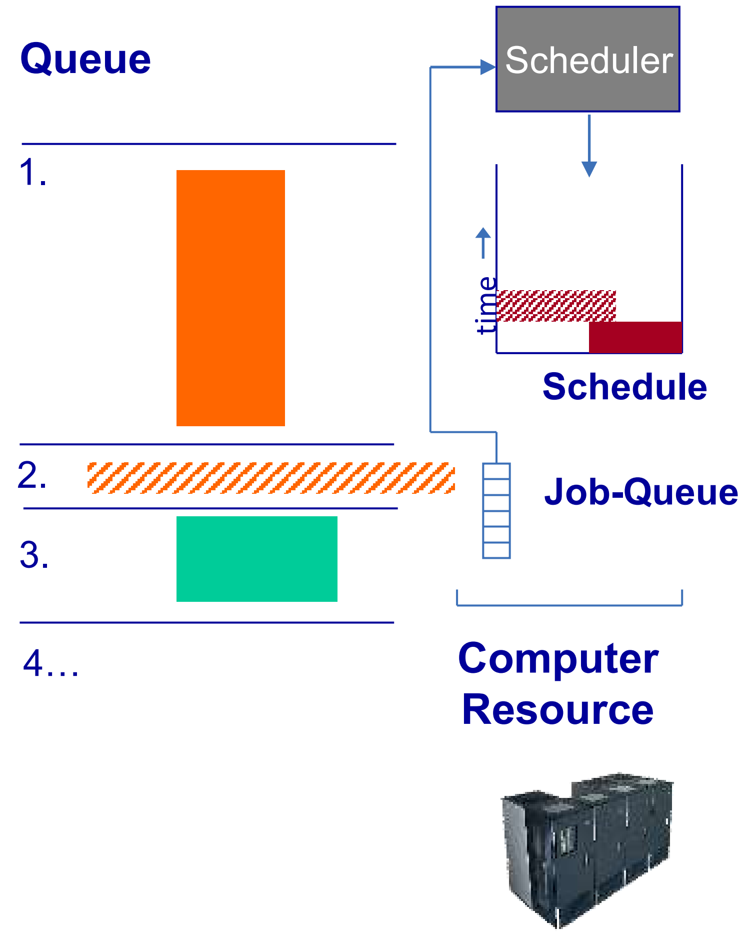
Phase 1

- On-line scheduling
- Processing time is not known until the release time

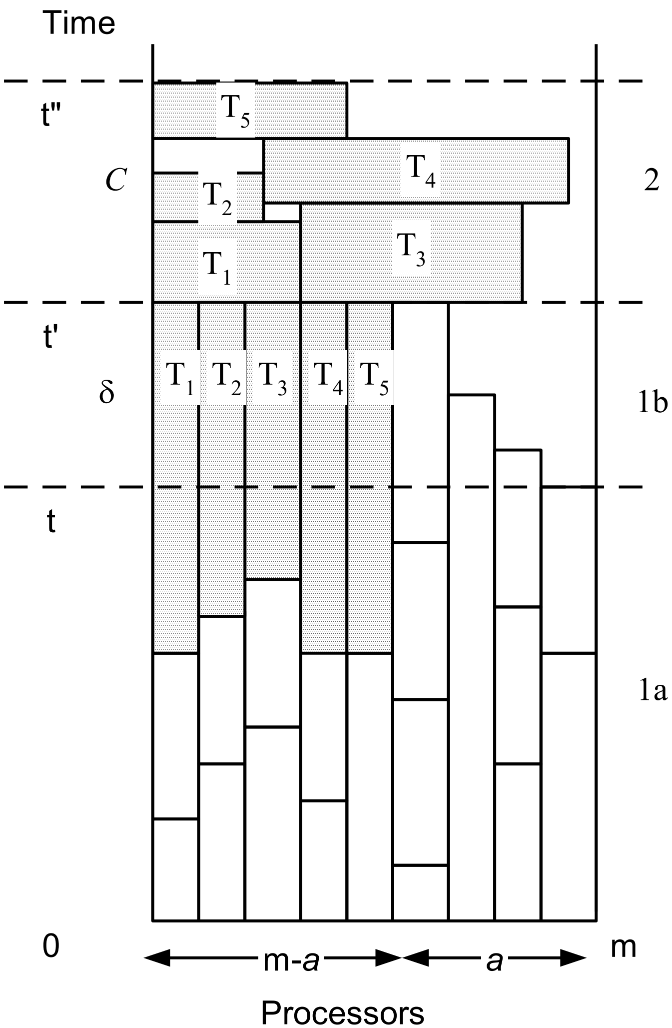
FCFS Schedule

by Ramin Yahyapour





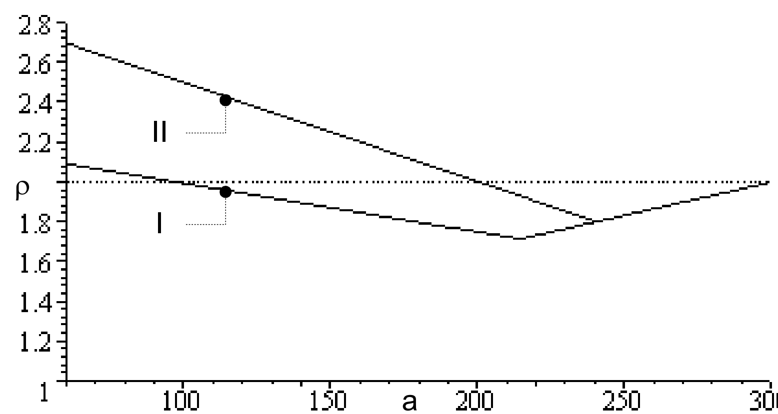
Idle Regulation for Rigid Jobs



$$\rho^{seq} = \max\{\rho^1, \rho^2\}$$

$$\rho^1 \leq 1 + \frac{a-1}{m}$$

$$\rho^2 \leq \frac{a}{m} + \frac{1}{m - k_{\max} + 1}\left(\mu_{\max}(m-a) + \frac{\mu_{\min}}{k_{\min}}(m - k_{\max})\right)$$

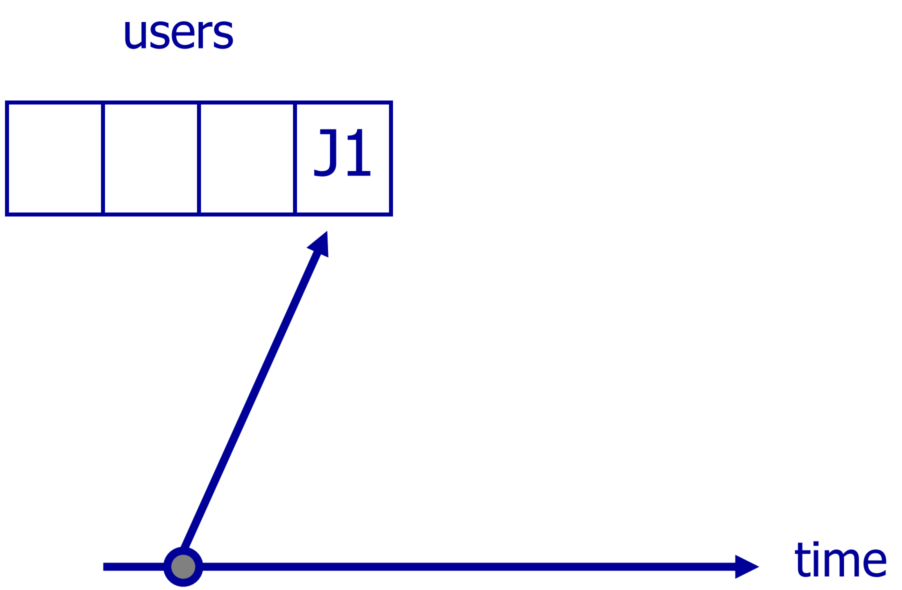


(I) Linear (logarithmic shape of a job speed-up function),

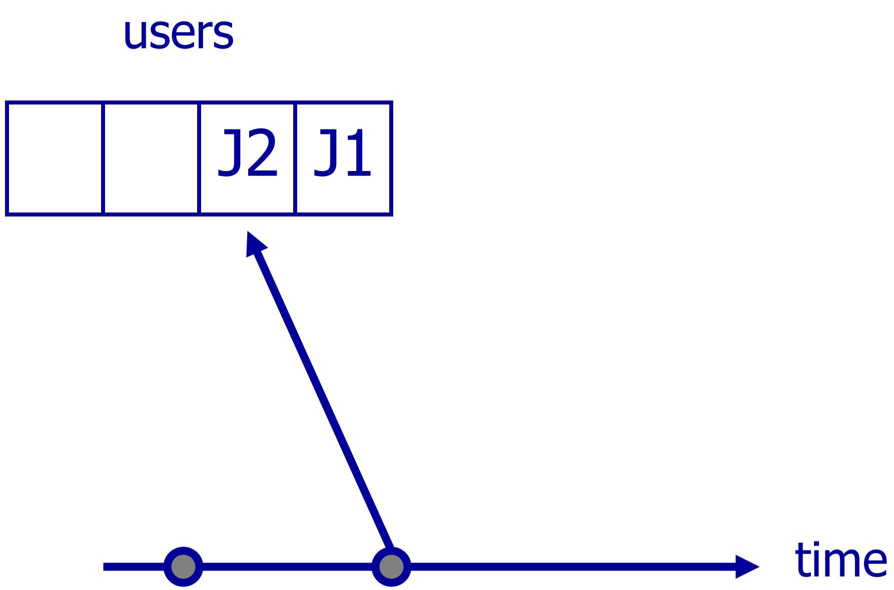
(II) Constant (linear m/2 shape of a job speed-up function of *k*).

Job submission

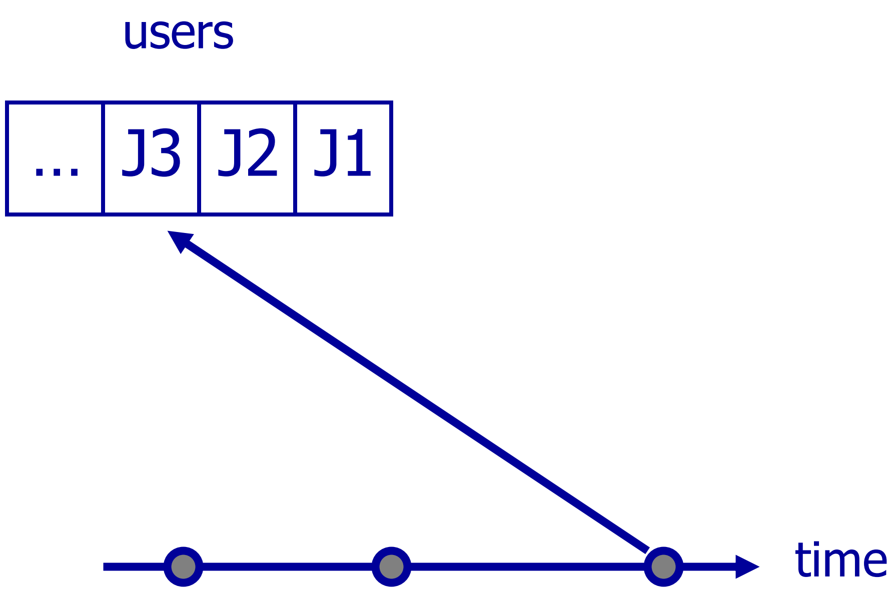
by Denis Trystram



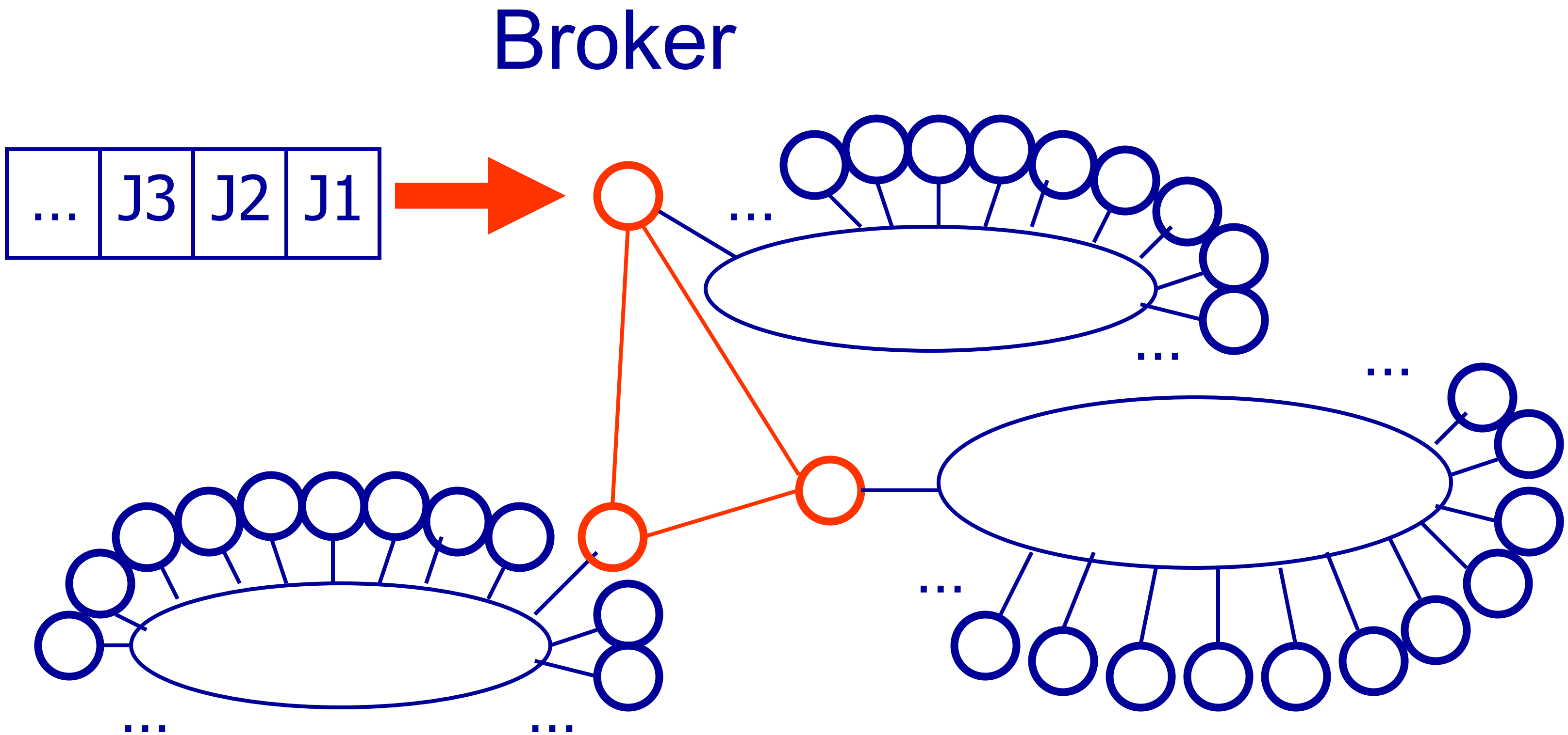
Job submission



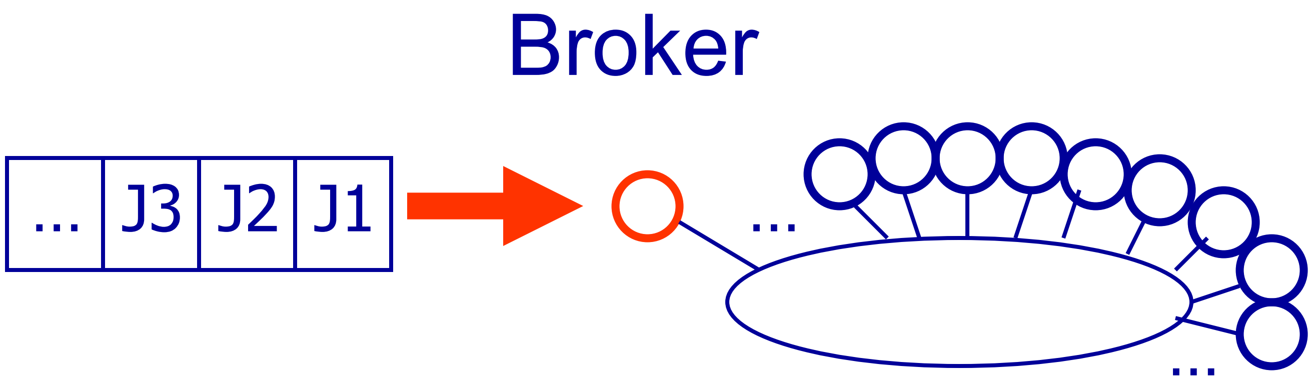
Job submission



Job allocation



Job allocation

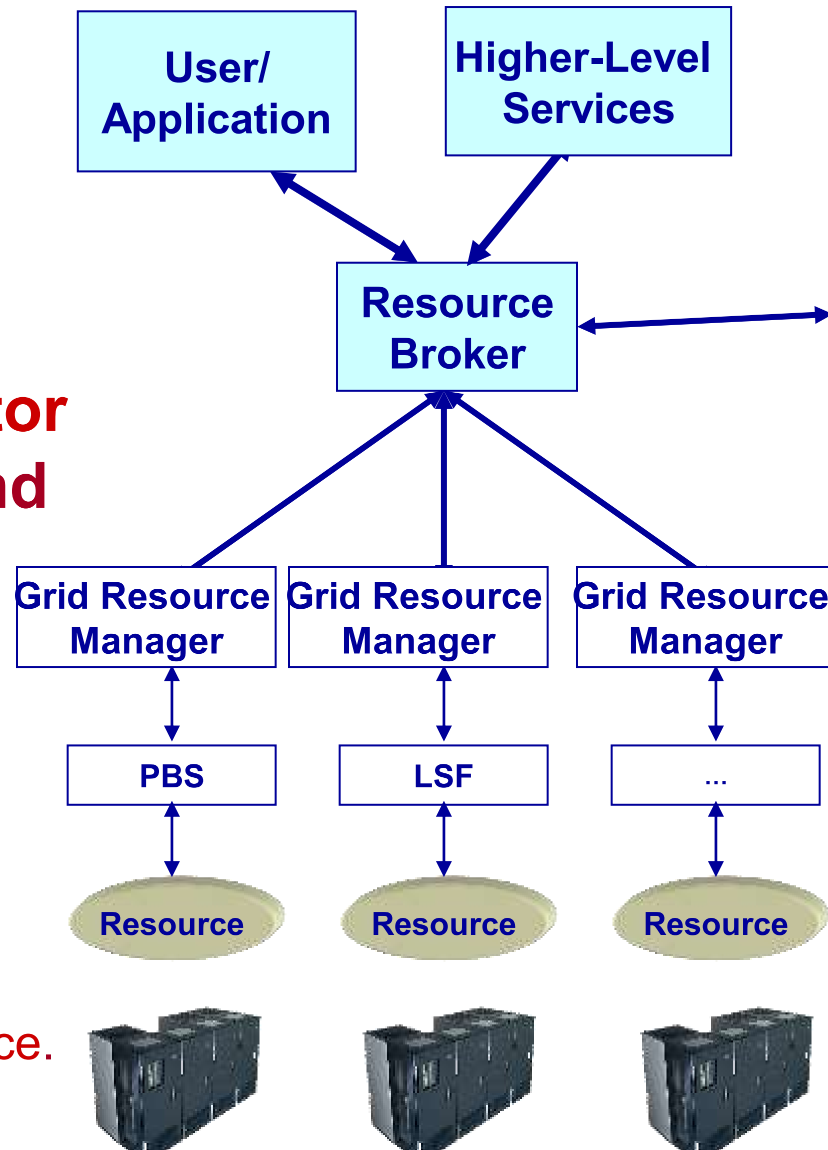


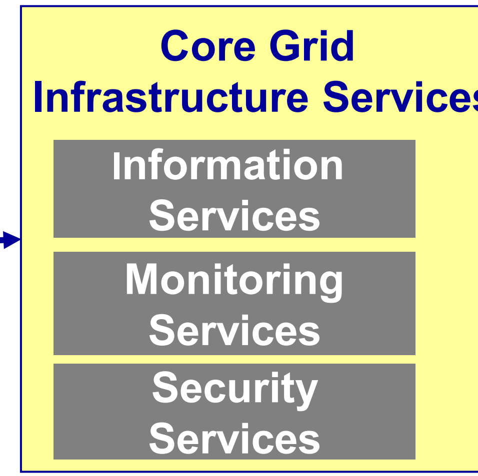
Middleware

by Ramin Yahyapour

Grid Middleware

acts as a mediator between user and resources



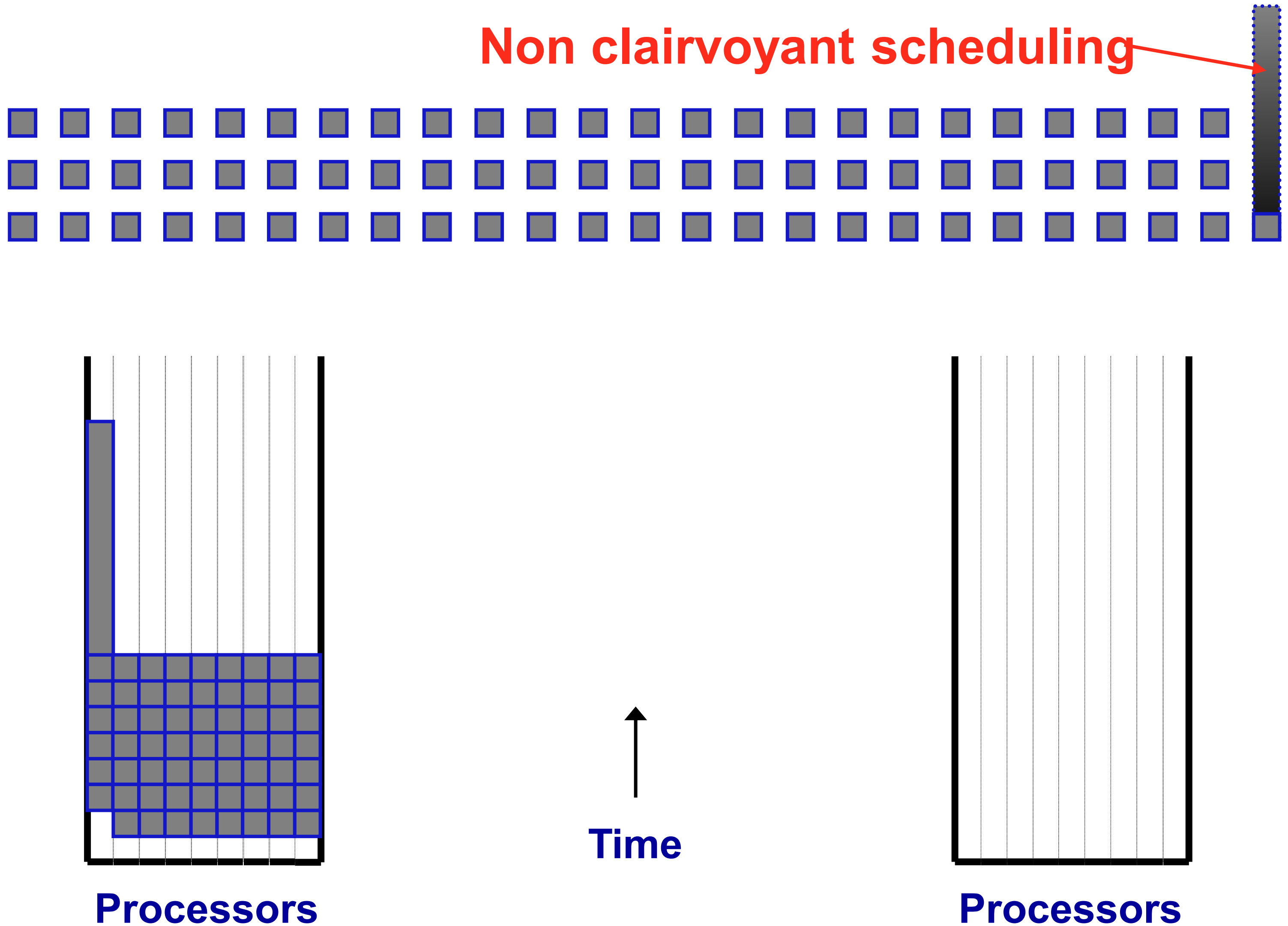


Local Resource Management

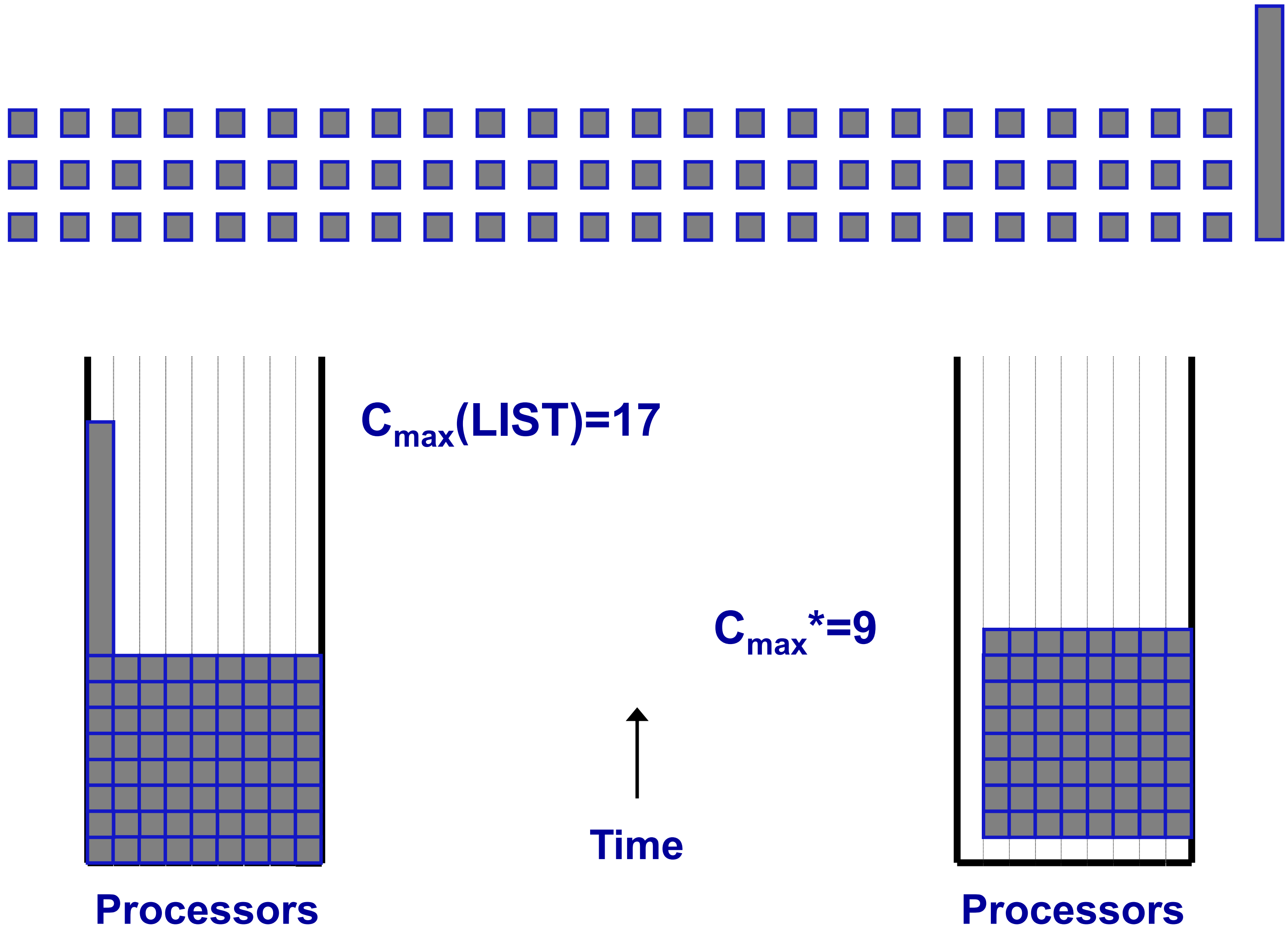
Presenting the Grid to the user as a single, unified resource.

- Resource discovery,
- Resource selection,
- Binding of software,
- Data, and hardware,
- Initiating computations
- Adapting to changes in Grid resources

List Scheduling

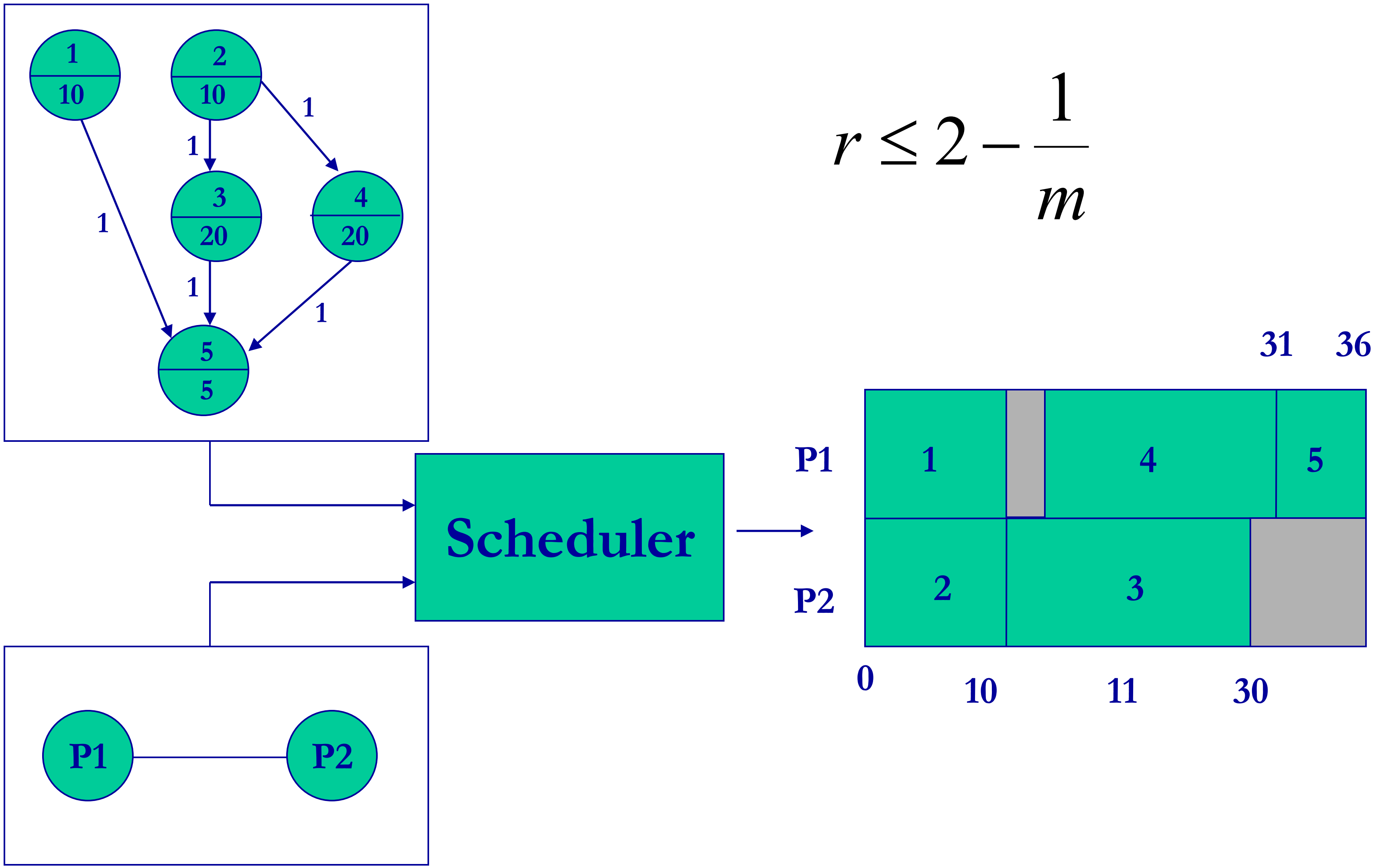


List Scheduling

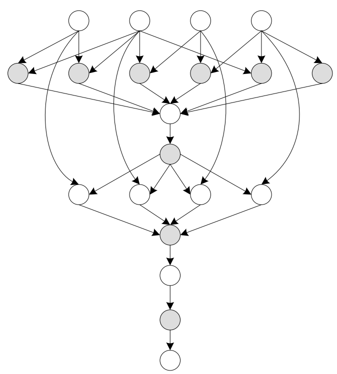


Sequential Tasks Scheduling

$$r \le 2 - \frac{1}{m}$$

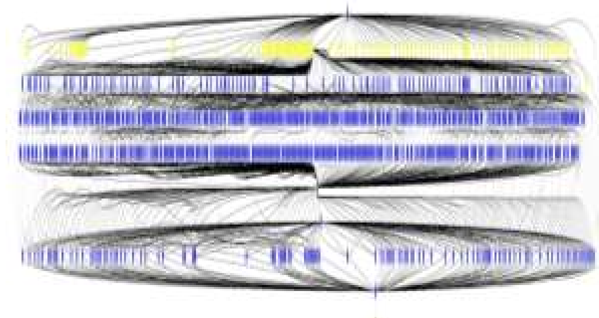


Scientific workflows

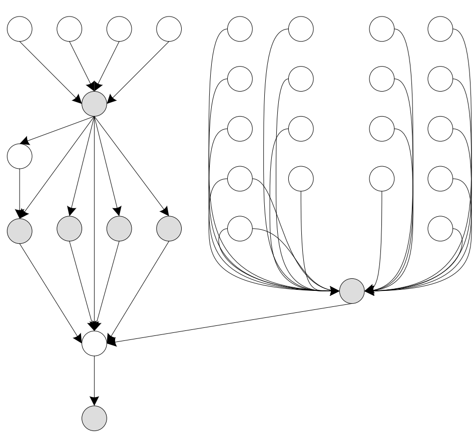


Montage

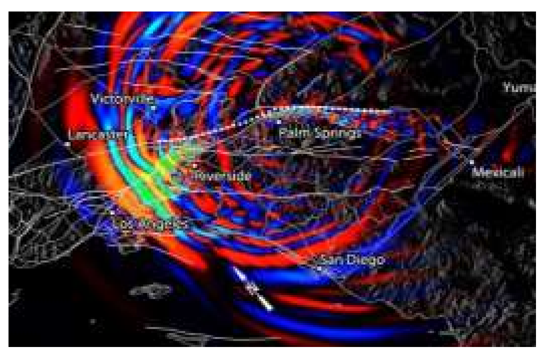
Space mosaics



Montage



CyberShake



Earthquake

Other aplications:
-Epigenomics,
-Genome,
-LIGO
-SIPHT

PEGASUS

Scheduling

- Two categories of scheduling
  - Economic-based
  - Performance-based

Performance-based optimization criteria

Mean waiting time	$t_w = \frac{1}{n}\sum_{j=1}^{n}(s_j - r_j)$
Mean bounded slowdown	$SD_b = \frac{1}{n}\sum_{j=1}^{n}\frac{t_w^j + p_j}{\max\{10, p_j\}}$
Sum of weighted completion times	$SWCT_w = \frac{1}{n}\sum_{j=1}^{n}(c_j \cdot w_j)$

Algorithm centric
User centric
System centric

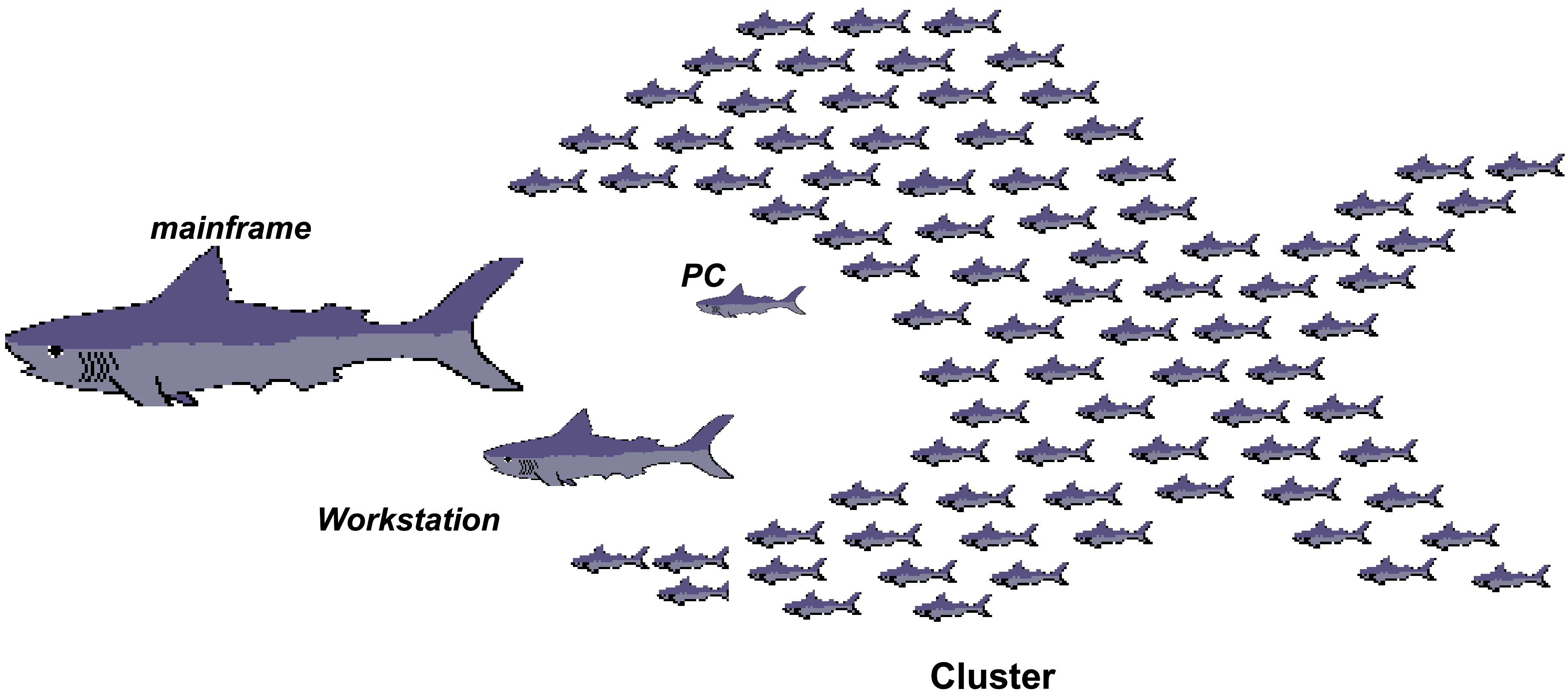
- **Makespan**
- **Competitive factor**
- **Mean Turnaround**
- **Sum waiting times**
- **Utilization**
- **Throughput**
- **Load Balance**

2-level strategies

	Model	Parameters	Description
I	WGS_Alloc+PS	*PRIO* = FCFS PS = EASY	9 strategies: MLp, MaxAR, MLB, MWT, MCT, MST, CPOP, Rand y HEFT
II	MPS_Alloc+PRIO+ PS	WGS_Label∈{DR, CR}, WGS_Alloc= {MLp, MaxAR, MLB, MST}, *PRIO*∈{SCF, LCF}, PS = EASY	16+3 best from 1: DR+MLp+LCF, DR+MaxAR+LCF, DR+MLB+LCF, DR+MST+LCF, DR+MLp+SCF, DR+MaxAR+SCF, DR+MLB+SCF, DR+MST+SCF, CR+MLp+LCF, CR+MaxAR+LCF, CR+MLB+LCF, CR+MST+LCF CR+MLp+SCF, CR+MaxAR+SCF, CR+MLB+SCF, CR+MST+SCF, MLp, MaxAR y MLB

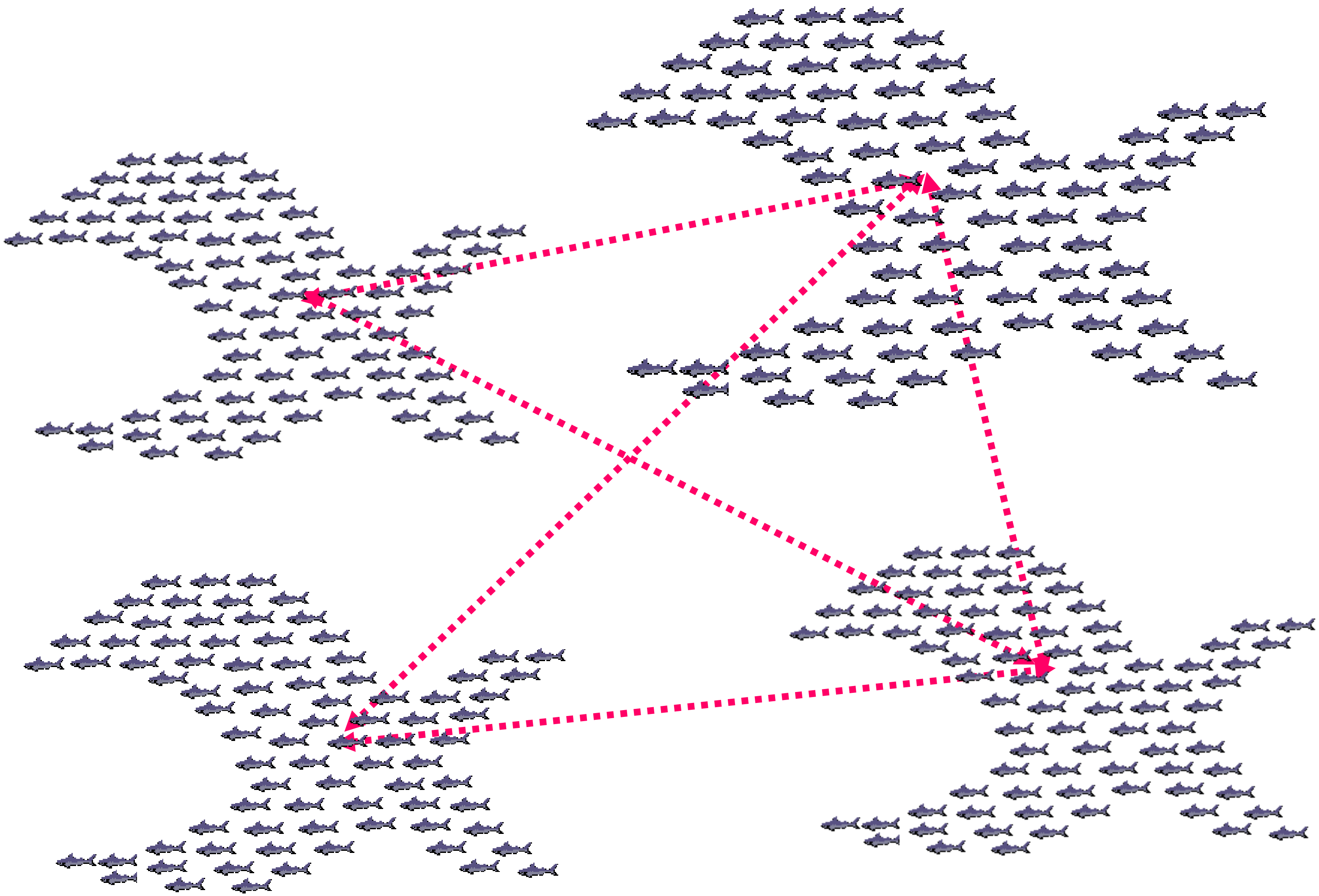
Cloud Computing

Grid Computing



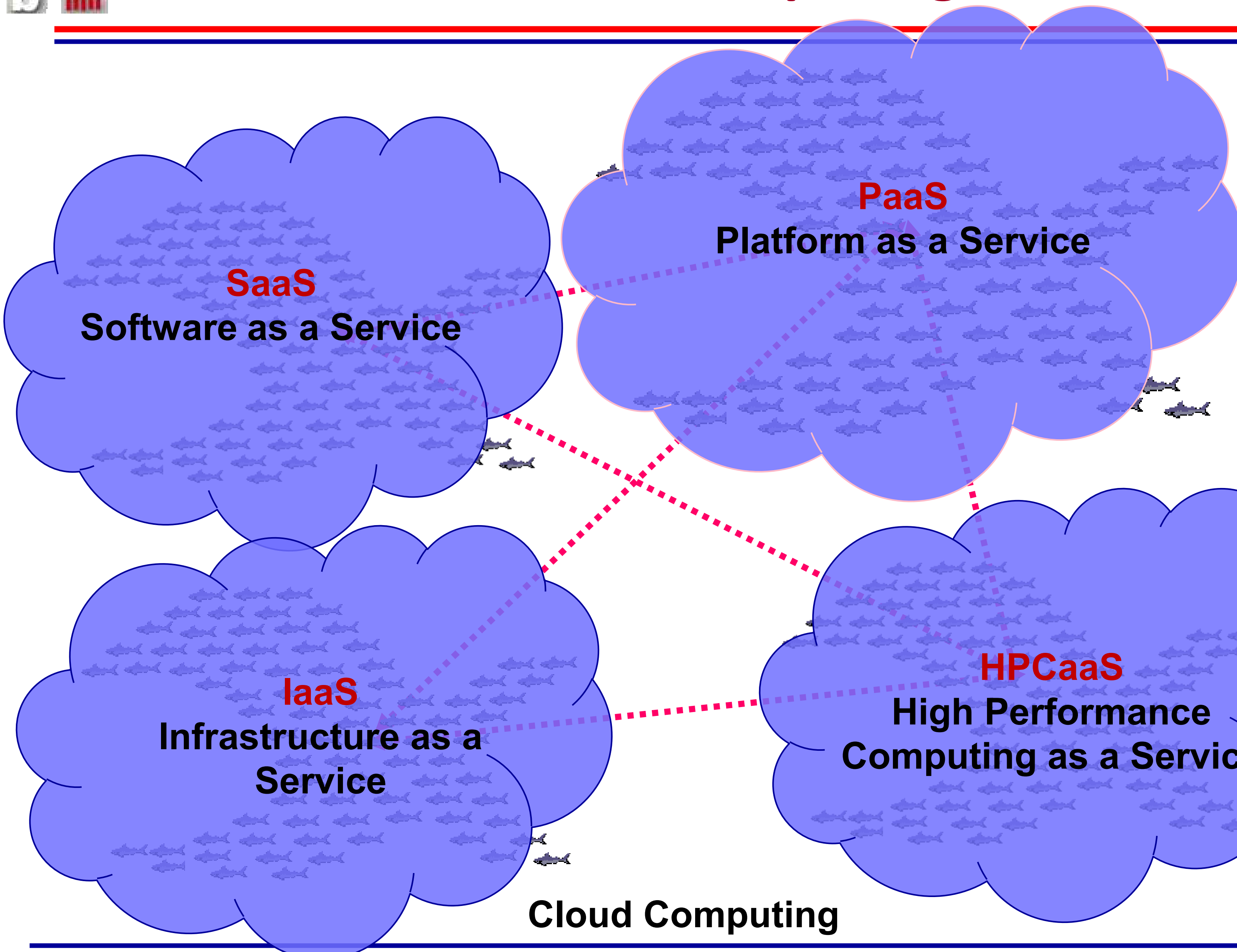
(by Christophe Jacquet)

Grid Computing



Computational GRID

Cloud Computing



Challenges in cloud computing.

Load balancing

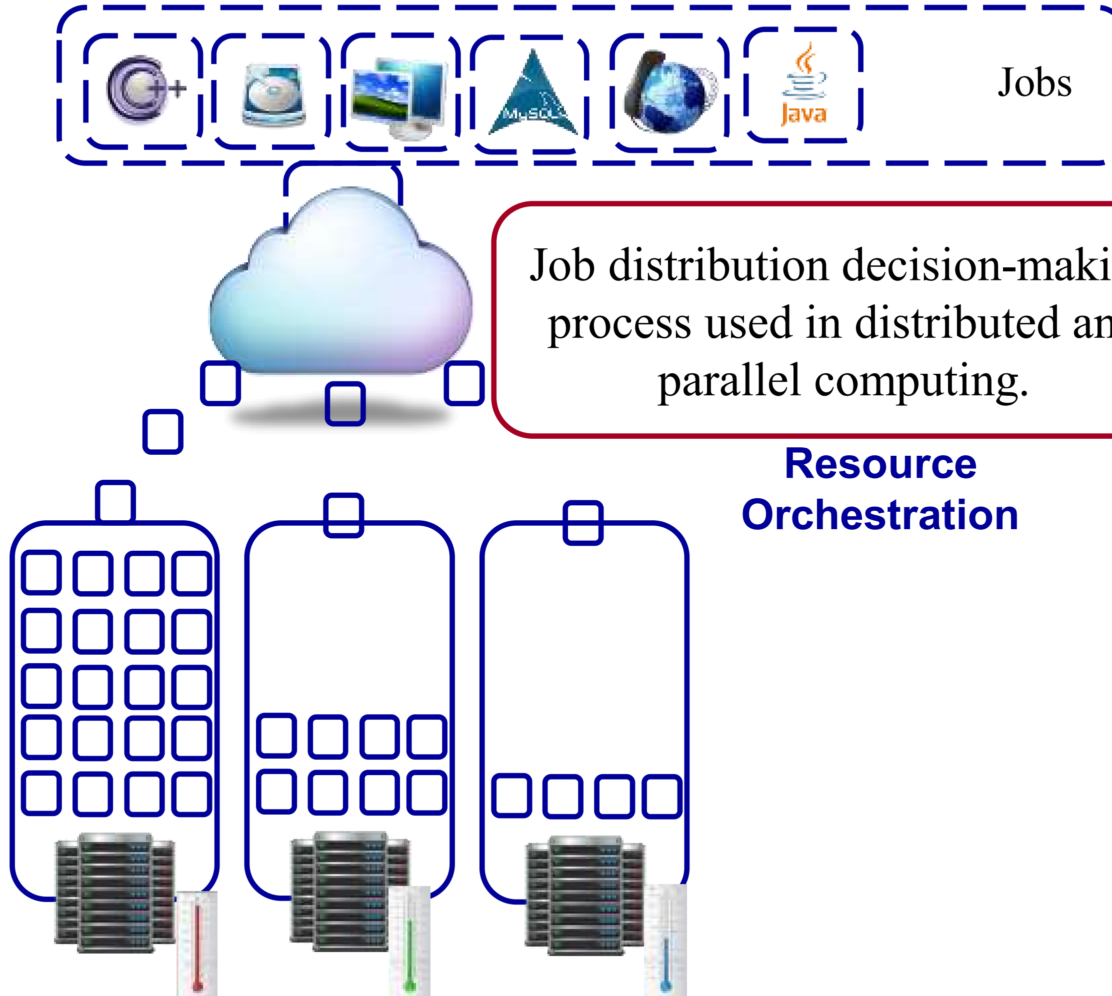
- Increases provider's profits.
- Achieves higher user satisfaction.
- Enables scalability.
- Avoids bottlenecks.

Quality of service

- Ensures sufficient amount of resources.
- Service Level Agreements.

Energy efficiency

- Impacts the users in terms of resource usage costs.
- Hardware efficiency.
- Jobs running on the system.



Job distribution decision-making process used in distributed and parallel computing.

Cloud Computing

Dynamic Resource Provisioning

- Elastic
- Efficient
- Green

Provider goals

- Cost reduction
- Customer satisfaction

Resource Provisioning

Allocate

- Processors
- Storage
- Network

Optimize

- Load balance
- Performance
- Costs

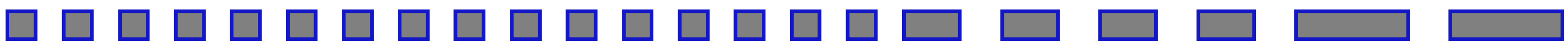
- Online and offline scheduling

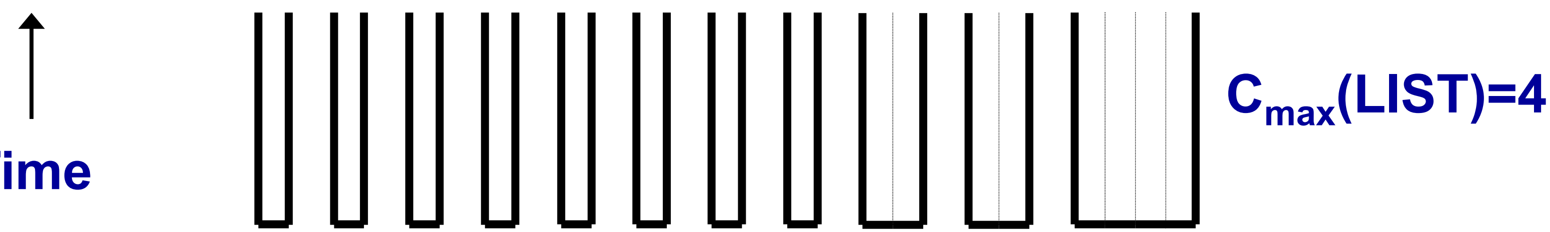
https://www.cloudberrylab.com/dedup-server.aspx

Knowledge-free

non-clairvoyant

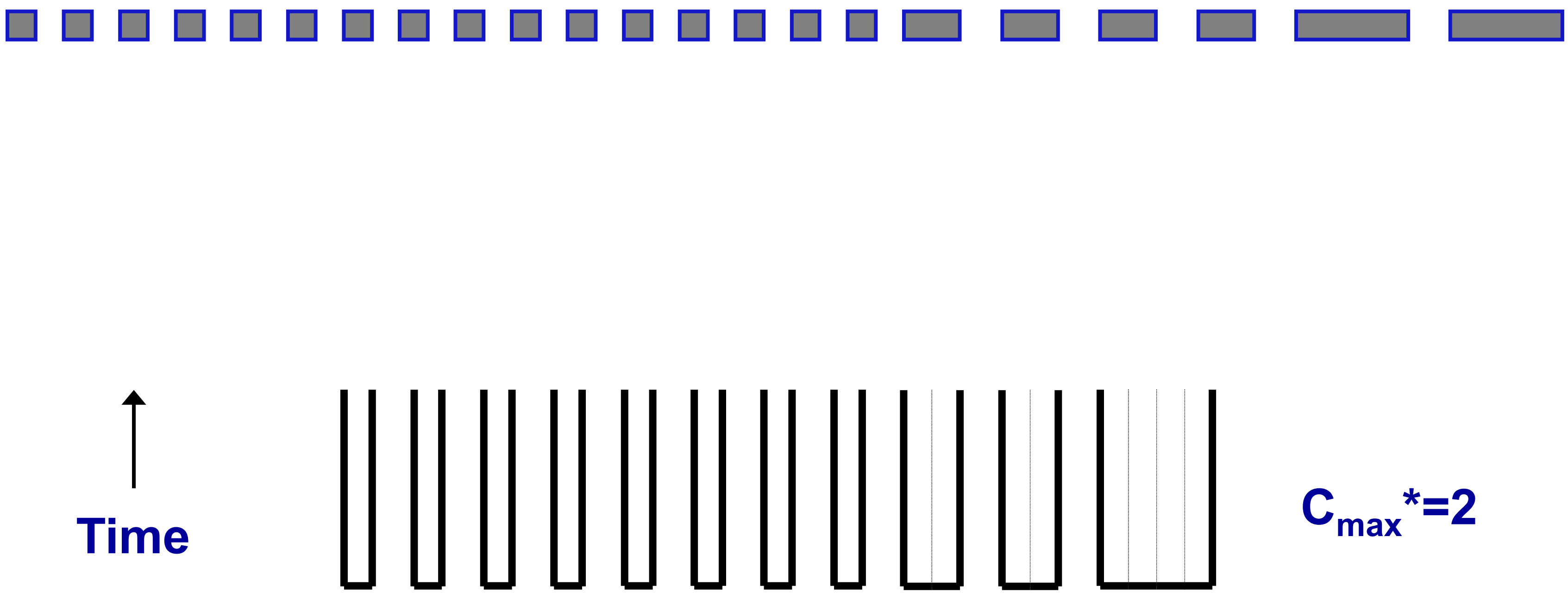
List Scheduling





Machines with different numbers of processors

List Scheduling



Time

$C_{max}^{*}=2$

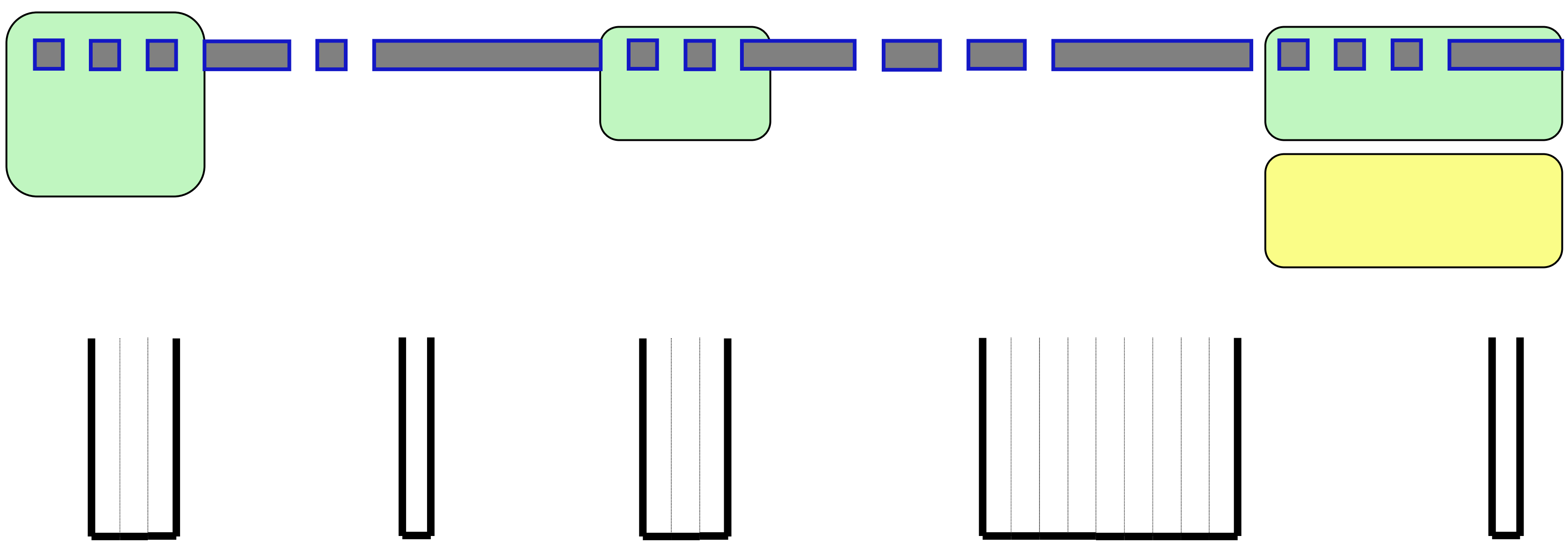
Machines with different numbers of processors

Scheduling Algorithm

1. The machines are arranged in ascending order of processor numbers.

2. A job is assigned to the first machine that can execute it.

- Group A: **>=** half of the processors on this machine are required.
- Group B: **<** half of the processors on this machine are required.



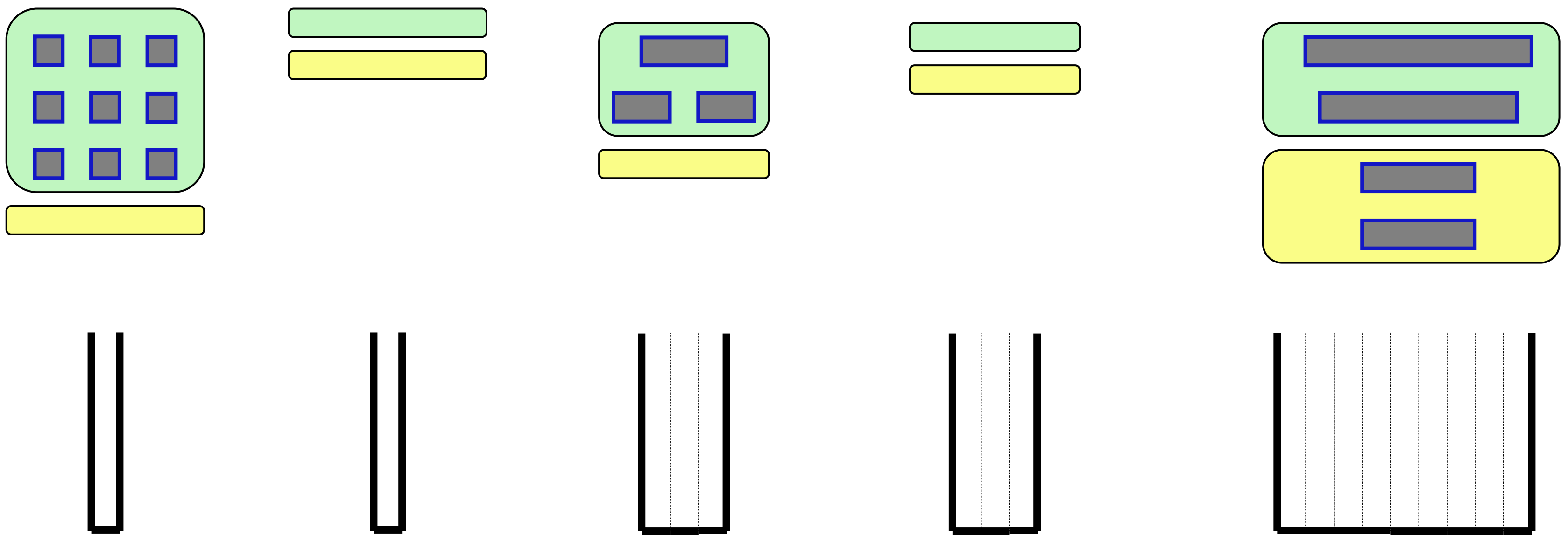
Scheduling Algorithm

3. Any machine applies a priority order when selecting jobs for execution:

Jobs of its group A

Jobs of its group B

Jobs that are enabled for execution on its previous machine.



Performance of the Algorithm

- Theoretical evaluation
  - $C_{max}(LIST)/C_{max}^* < 3$ in the **offline** case
  - $C_{max}(LIST)/C_{max}^* < 5$ in the online case

Improved by …

(Klaus Jansen, Denis Trystram et. al…)
5/2, 7/3, 2 + ε, 2 –approximations

IEEE IPDPS, 2008

Workload Uncertainty

Adaptive Admissible Allocation

Andrei Tchernykh — **CICESE Research Center**
José Luis González-García — **Mexico**
Vanessa Miranda-López

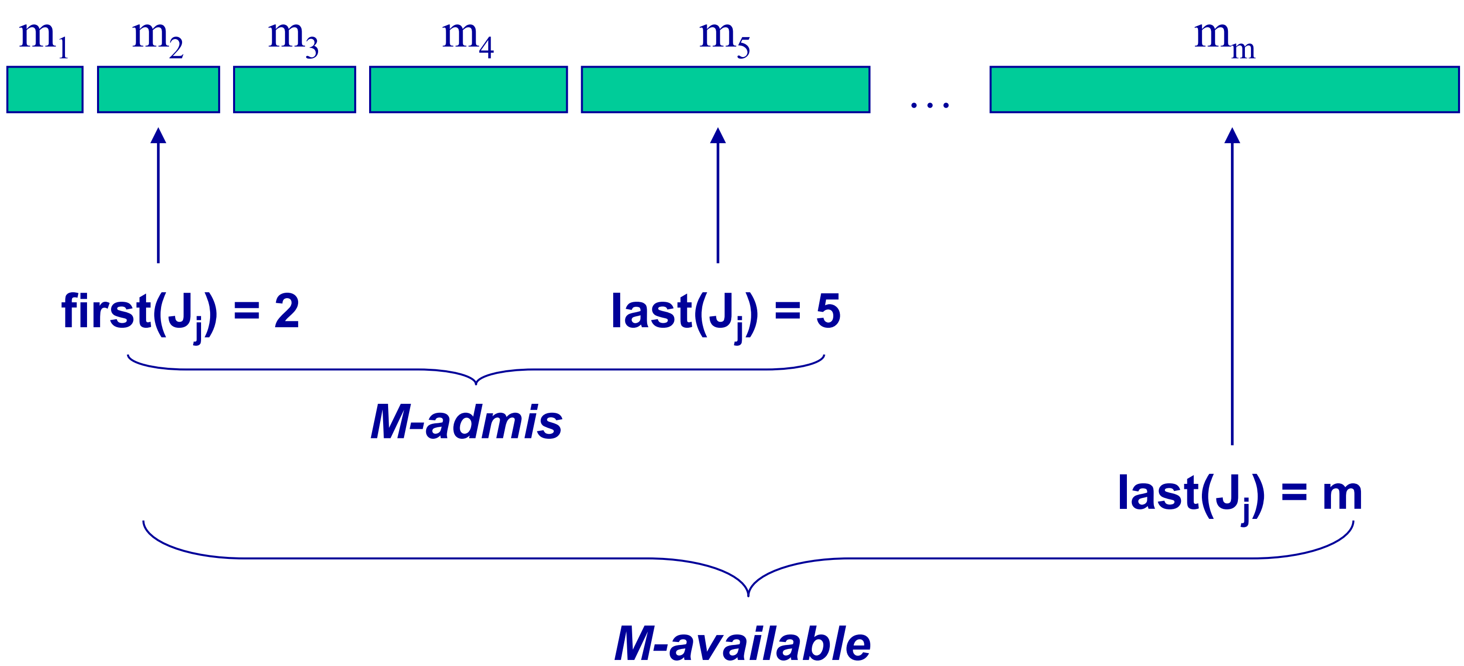
Uwe Schwiegelshohn — **University of Dortmund**
Germany

Ramin Yahyapour — **University of Göttingen**
Germany

Future Generation Computer Systems 2012
Journal of Scheduling, 2010

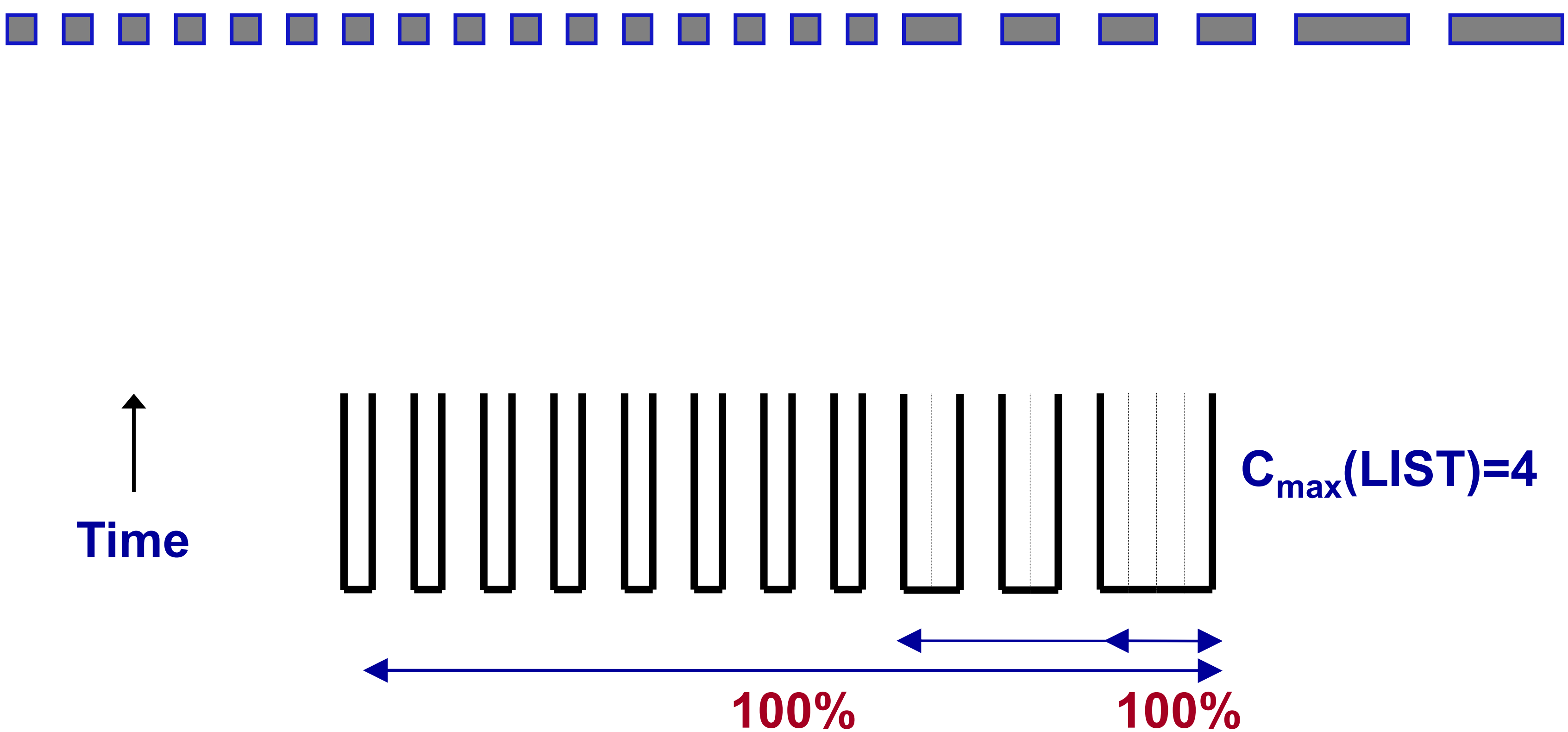
Allocation uncertainty

If last is the minimum *r* such that $\sum_{i=first(j_j)}^{r} m_i \geq a \sum_{i=first(j_j)}^{m} m_i$



List Scheduling

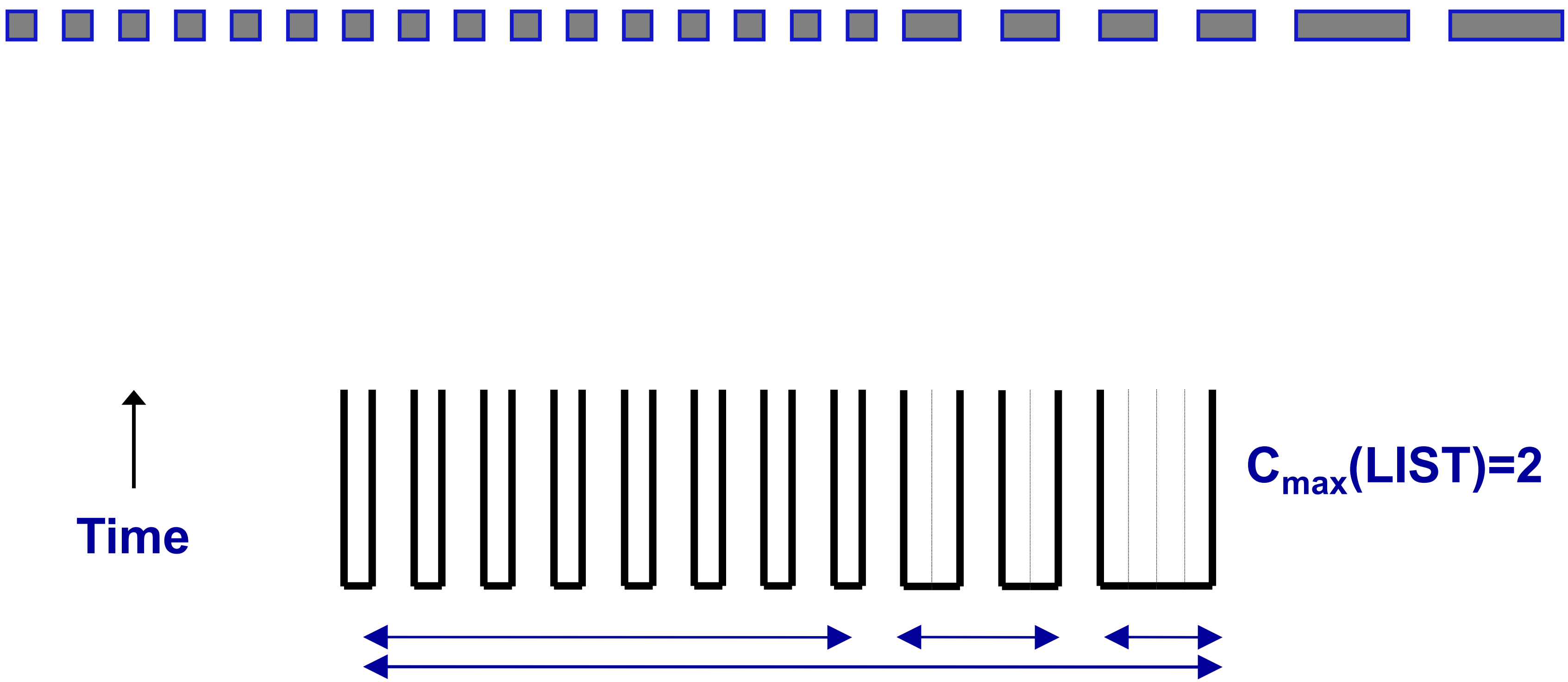
a=1



Machines with different numbers of processors

Admissible Allocation

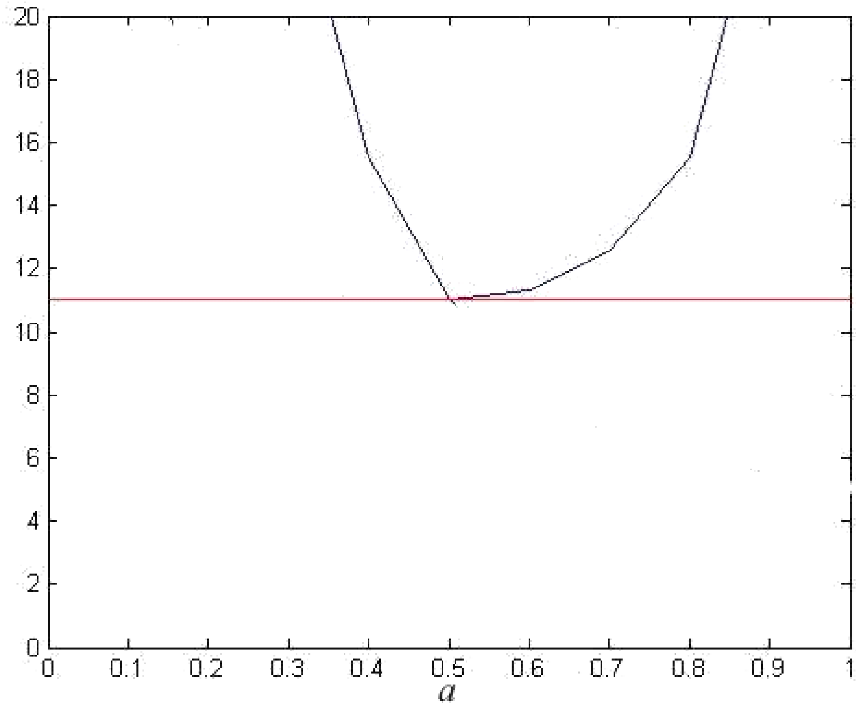
a=0.5



Machines with different numbers of processors

Adaptive optimization

For a set of machines with identical processors, and for a set of rigid jobs with admissible range $0 \le a \le 1$



Competitive factor (on-line)

Min_LB-a + Best_PS

$$\rho \le \begin{cases} 3 + \dfrac{2}{a^2} & \text{para } a \le \dfrac{m_{f,r}}{m_{f_0,m}} \\ 3 + \dfrac{2}{a(1-a)} & \text{para } a > \dfrac{m_{f,r}}{m_{f_0,m}} \end{cases}$$

Approximation factor (off-line)

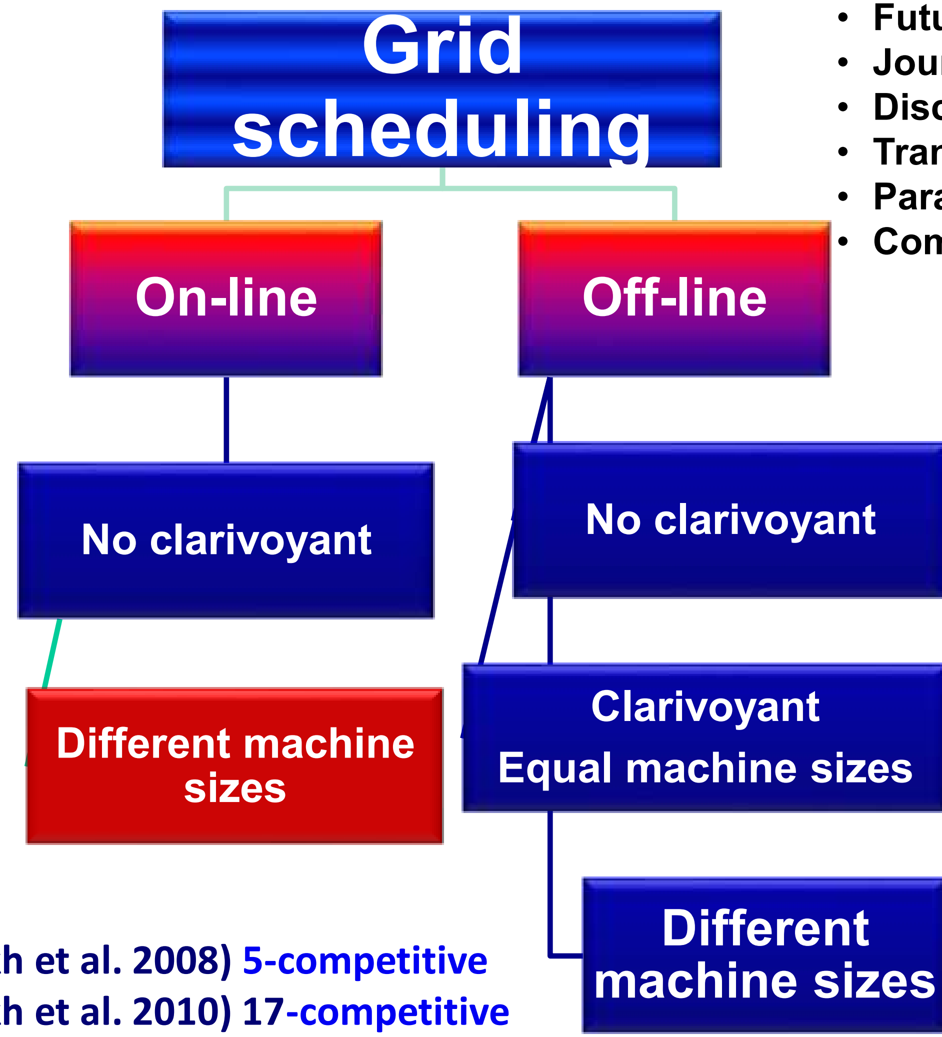
Min_LB-a + Best_PS

$$\rho \le \begin{cases} 1 + \dfrac{2}{a^2} & \text{para } a \le \dfrac{m_{f,r}}{m_{f_0,m}} \\ 1 + \dfrac{2}{a(1-a)} & \text{para } a > \dfrac{m_{f,r}}{m_{f_0,m}} \end{cases}$$

Tchernykh, et al 2012
Future Generation Computer Systems, Elsevier
Tchernykh, et al 2010
Journal of Scheduling, Springer

Theoretical Evaluation

- **Future Generation Computer Systems**, Elsevier
- **Journal of Scheduling**, Springer
- **Discrete Applied Mathematics**, Elsevier
- **Tran Fund Elec, Comm. & Comp. Science**, IEICE
- **Parallel and Distributed Processing**, IEEE
- **Computers & Industrial Engineering**, Elsevier



(Schwiegelshon et al. 2008) **3--approximation**

(Pascual et al. 2008) **4--approximation**

(Klaus Jansen, Denis Trystram) 5/2, 7/3, 2 + ε, 2 –approximation

(Zhuk et al. 2004) **10--approximation**

(Tchernykh et al. 2005) **10—approximation**

(Tchernykh et al. 2012) **3—approximation**

(Tchernykh et al. 2008) 5-competitive

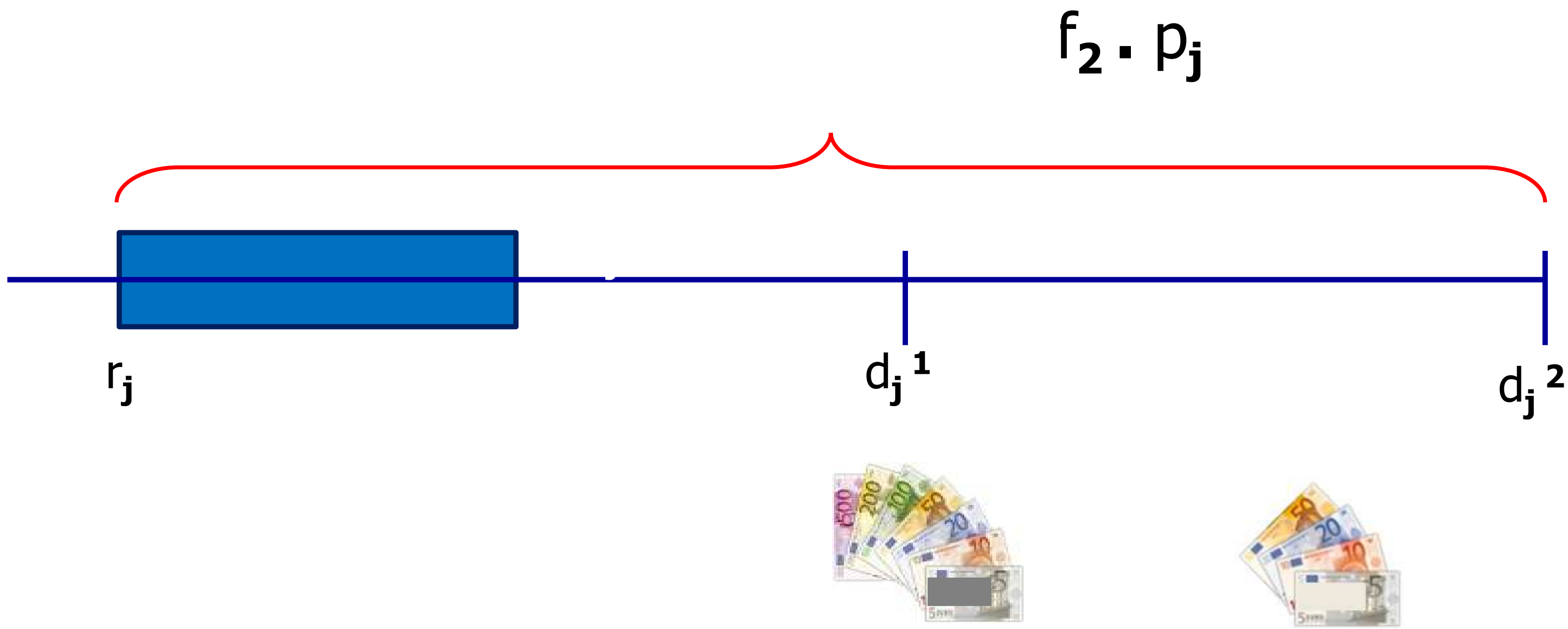
(Tchernykh et al. 2010) 17-competitive

(Schwiegelshon 2010) (2*e*+1)-competitive

(Tchernykh et al. 2012) 5-competitive

Scheduling for Cloud Computing with Quality of Service

Quality of Service



$f_2 = 4$: guarantees to deliver at least 25% of power

The provider guarantees to deliver the requested processing time within a certain time frame: **slack or stretch factor** f_i

Quality of Service

- Response time in relation to the requested processing time

$$d_j = r_j + f_i \times p_j$$

Deadline

Service Level (slack factor)

Execution time

price per time unit

Profit

$$g_j = x_j \cdot p_j \cdot u_i$$

$$x_j = \begin{cases} 1 - accepted \\ 0 - rejected \end{cases}$$

$$(f_i \geq 1)$$

Competitive Factor

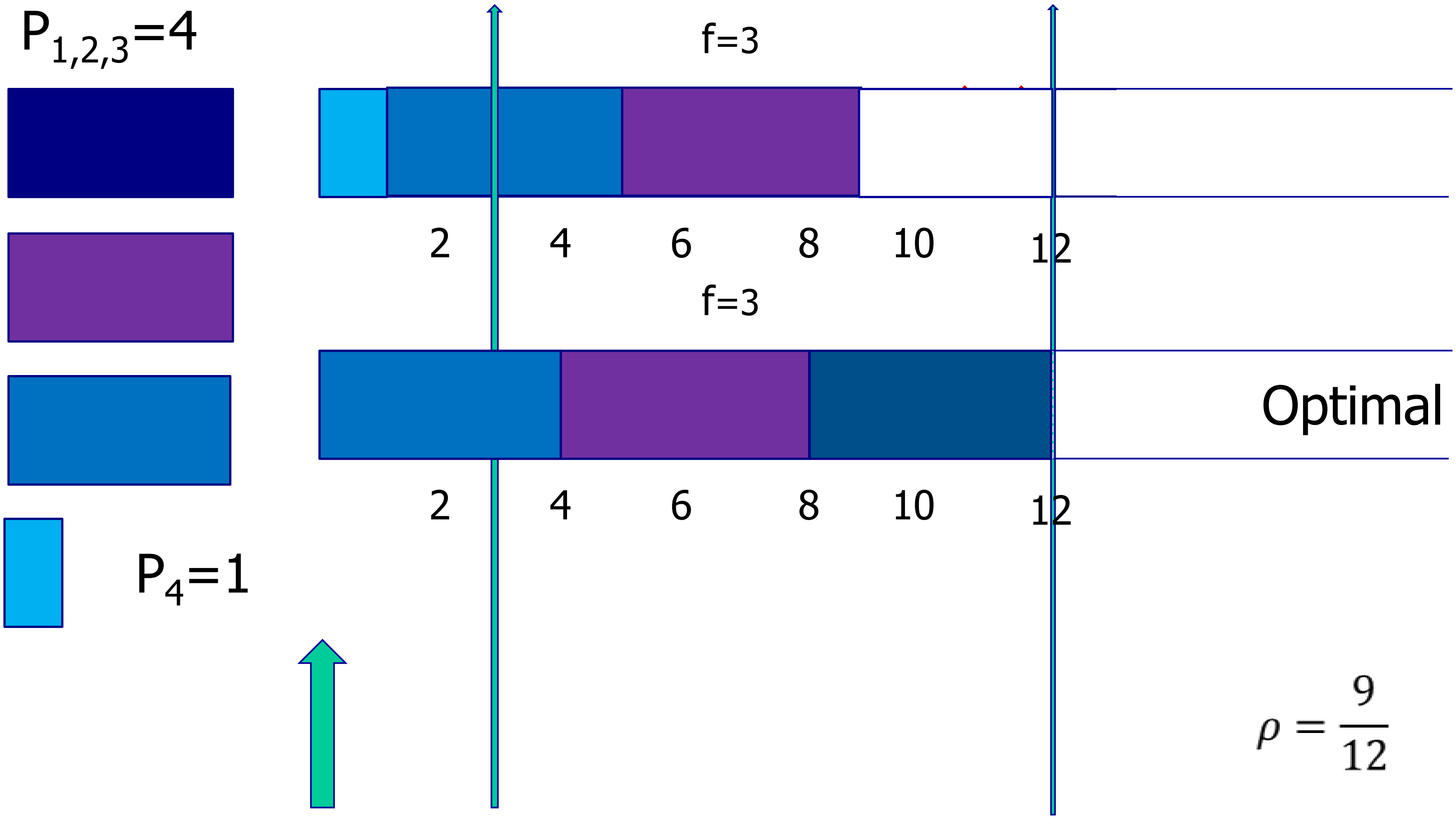
Obtained Income

$$\rho = \frac{\sum(p_j \times u_j)}{V^*} \leq 1$$

Competitive Factor

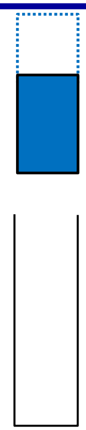
Optimal income

Scheduling



$r_1=r_2=r_3=r_4=0$ $d_1=3$ $d_2=d_3=d_4=12$

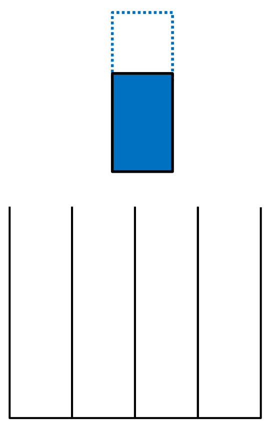
Competitive Factor



SSL-SM

$$\rho \le 1 - \left(1 - \frac{p_{min}}{p_{max}}\right)\frac{1}{f}$$

Das Gupta and Palis, 2001



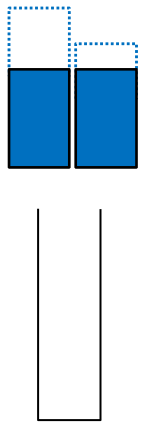
SSL-MM

$$\rho \le \frac{f}{1 + f\left(1 - \frac{p_{min}}{p_{max}}\right)}$$

Schwiegelshohn,Tchernykh 2012

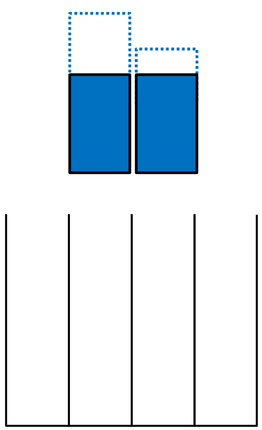
Competitive Factor

MSL-SM



$$\rho \leq max\{\frac{\frac{p_{min}}{p_{max}}}{(f_I - 1)}, \frac{f_I - 1 + \frac{p_{min}}{p_{max}}}{f_I - 1 + \frac{u_I}{u_{II}}}$$

MSL-MM



$$\rho \leq \frac{u_{II}}{u_I}(1 - \frac{1}{f_I})$$

Schwiegelshohn,Tchernykh, IPDPS 2012

Green computing

Green Computing?

- Global networks is growing at a rapid pace.
- Need to be kept constantly running in order to be available on-demand
- Their [illegible]nption grows.

Such new technologies have the power to do significant damage to our ecosystems.

Green Computing?

Traditional heuristic-based approaches to resource optimization become **insufficient**

Efficient eco-friendly power-aware computing resources optimization

- **reducing the environmental impact**
- **reducing costs**

Important issues – fossil fuels

Average desktop computer with monitor requires

- **10 times its weight in chemicals and fossil fuels to produce**
- **266 kg of fossil fuel for LCD monitor**
- **4 litres of oil for laser toner cartridge**

Important issues – electronic-waste

- **Over 130,000 PCs dumped in US homes & businesses…each day**
- **Less than 10% of electronics are recycled**
- **Est. 50 million tons of e-waste is generated globally each year**

Electronic Waste

Important issues – toxic waste

Electronic waste

- up to 70% of all hazardous waste.
- many toxic materials (heavy metals, plastics)
- can easily leach into ground water and bio-accumulate

Chip manufacturing uses some of the deadliest gases and chemicals

- CRT – graphite/zinc leachate (monitors are hazardous waste)
- Lead (plumbum): can attack proteins and DNA
- LCD – 4-12 mg mercury /unit

Important issues – wasting electricity

PC wastes half the power

- approximately one-third of their power as heat

The more powerful the machine,

  - the more cool air needed to keep it from overheating.

Cooling towers

Important issues – Improving reliability

by Rajkumar Buyya

For every 10°C increase in temperature, the failure rate of a system doubles

- **Reliability of Supercomputer**

System	CPUs	Reliability
ASCI Q	8,192	**MTBI: 6.5 hrs.** HW outage sources: storage, CPU , memory.
ASCI White	8,192	**MTBF: 5 hrs ('01) and 40 hrs ('03).** HW outage sources: storage, CPU, 3rd-party HW.
PSC Lemieux	3,016	**MTBI: 9.7 hours.**
Google	15,000	**20 reboots/day; 2-3% machines replaced/year.** HW outage sources: storage, memory.

MTBF/I: mean time between failures/interrupts

- **Estimated Cost of an hour of system downtime**

Service	Cost of One Hour of Downtime
Brokerage Operations	$6,450,000
Credit Card Authorization	$2,600,000
eBay	$225,000
Amazon.com	$180,000
Package Shipping Services	$150,000
Home Shopping Channel	$113,000
Catalog Sales Center	$90,000

The way out

- Energy-efficient manufacturing of computer parts
- Replacing petroleum-filled plastic with bioplastics
- Best use of the device by upgrading and repairing in time
- Avoiding the discarding: less e-waste
- Power-sucking displays can be replaced with green light displays made of OLEDs, or organic light-emitting diodes
- Toxic materials can be replaced by silver and copper making recycling of computers more effective
- Use of non-toxic material make the worker safe from health problem

Green computing

- minimizes the energy consumption
- saves the resource of the country as a whole.
- In the long term - green equipment will be less costly without any hidden cost of waste

The way out

- **More-efficient processors**
- **Setting the Power Options of computer to sleep mode**

- **It is better to do computer-related tasks during blocks of time**
- **Flat panel monitors**
- **Smaller form factor (e.g. 2.5 inch) hard disk drives**
- **Solid-state drives store data in flash memory or DRAM (no moving parts, power consumption may be reduced)**

Sophisticated power management
Operating system support
Power supply

Storage, Display
Video card
Materials recycling

The way out

Algorithmic efficiency

- has an impact on the amount of computer resources required for any given computing function (**consolidation**)

Resource allocation

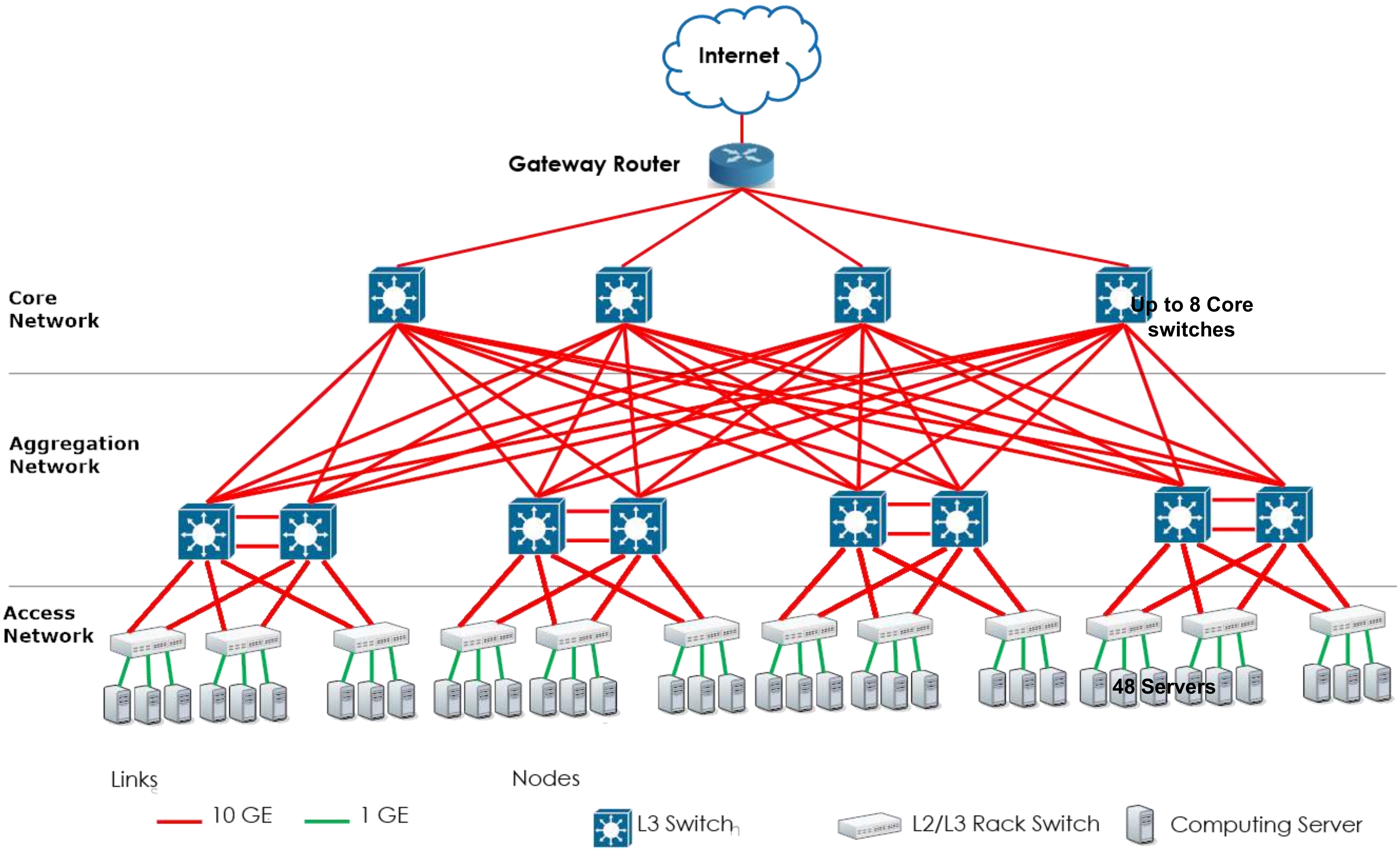
- cut energy usage by routing traffic and resource usage

Virtualization

- Use what you need (*Cloud computing*)

Adaptive Consolidation for Energy Saving

Three-tier topology



Definition

- $DC3t|r_j, l_j^{cp}, l_j^{cm}\ |E^{IT}, S$ Scheduling model
  - $DC3t$ three-tier data center, identical processors, different power consumption profiles.
  - r_j release time
  - l_j^{cp}, l_j^{cm} computational and communication requirements for job *j* given in **MIPS** and **Mbps** respectively

.

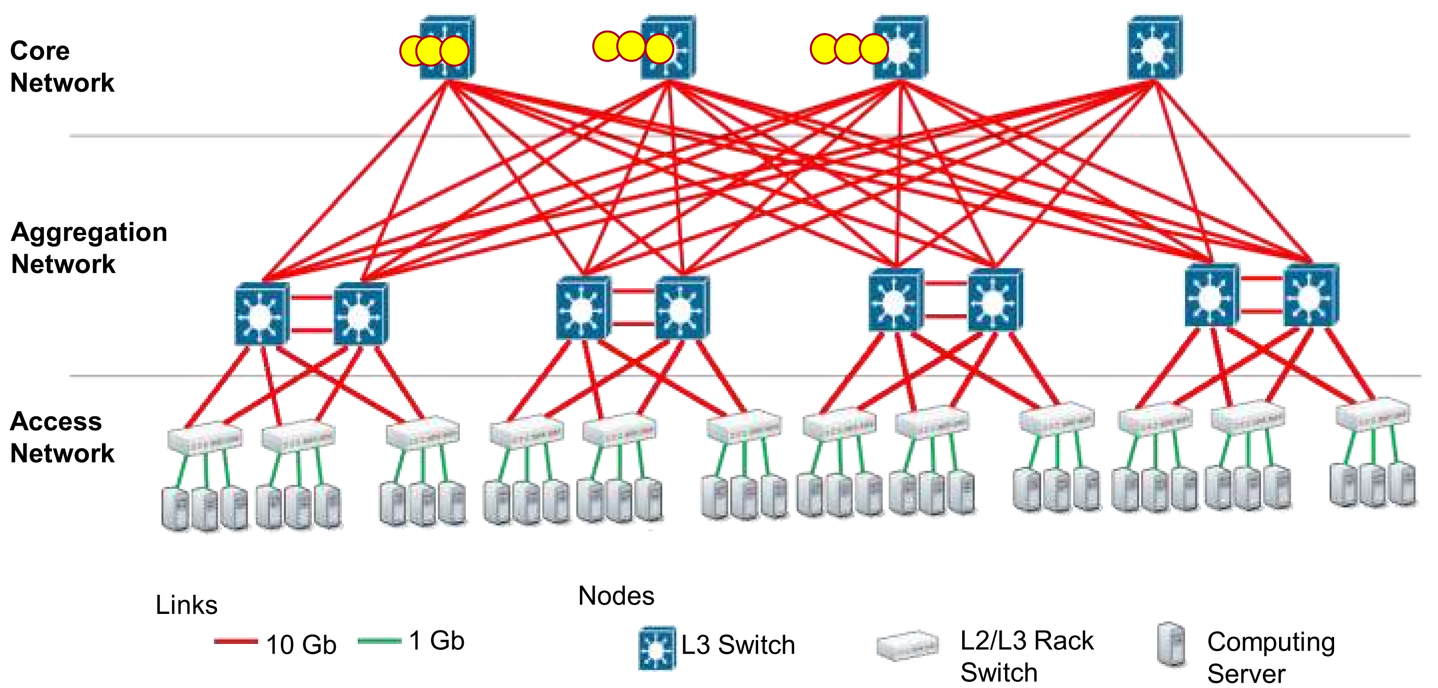
  - E^{IT} amount of energy consumed by IT equipment in data center.
  - S mean SLA violations.

$S = \frac{V_{Mbps}}{amount\ of\ jobs}$, V_{Mbps} number of jobs that didn't meet **Mbps** requirements

Consolidation

Most of energy saving is due to consolidation procedures.

Increase number of server that can be put into “sleep” mode.

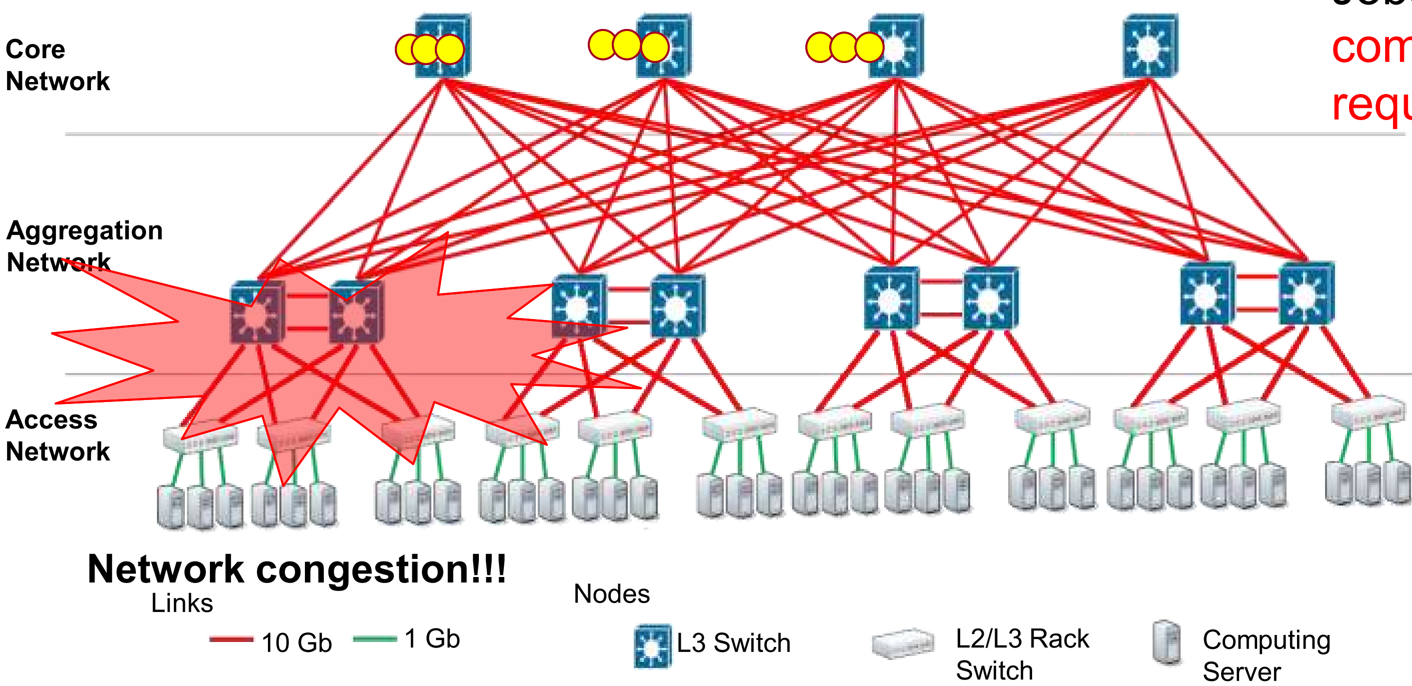


Uncertainty of communication

Most of energy saving is due to consolidation procedures.

Increase number of server that can be put into “sleep” mode.

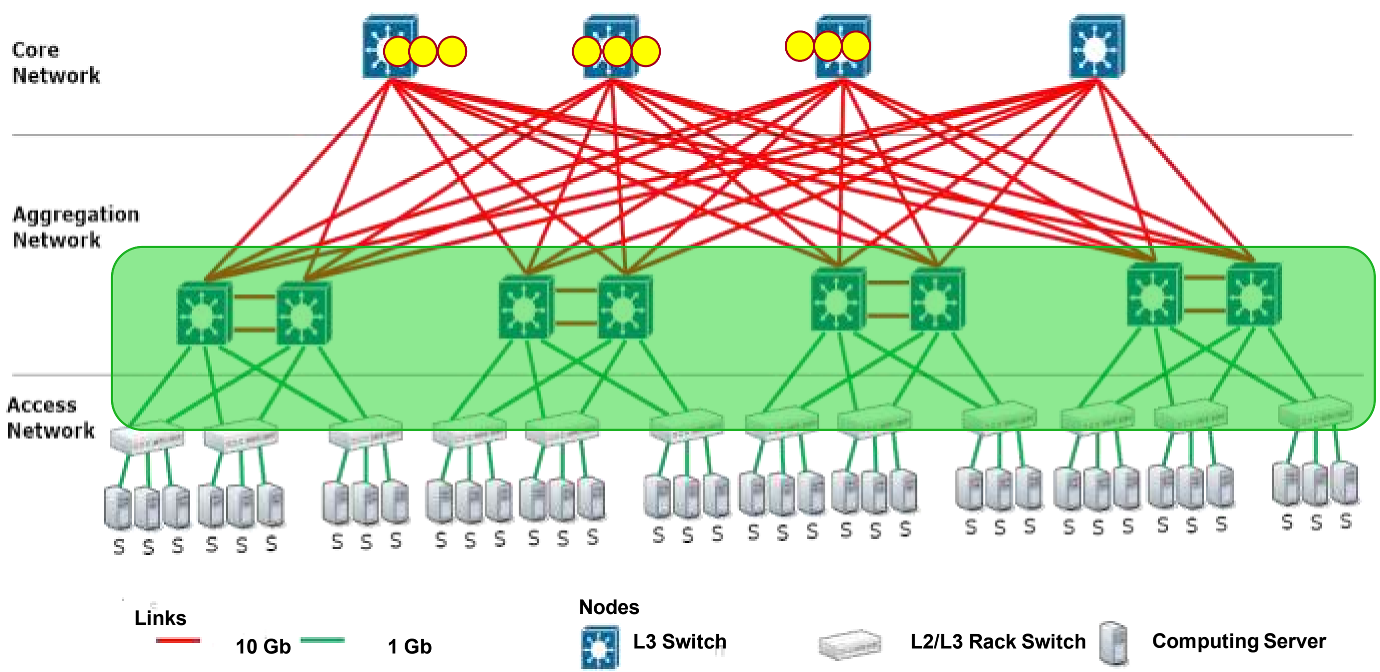
Jobs with high communication requirement



Network balancing

Scheduler should tradeoff workload concentration with load balancing of network traffic

Network is balanced !!!



- $f_i = \alpha f_i^{cp} + (1-\alpha) f_i^{cm}$
- $f_i^{cp} = \beta \overline{f_i} + (1-\beta)\mathrm{EPC}_i^{cp}$
  - $\overline{f_i}$ - function of server load $l_i^{cp}(t)$
  - EPC_i^{cp} - Energy proportionality of machine I
- $f_i^{cm\ 1} = \gamma\left(1 - \frac{1}{1+e^{-10 l_i^p}}\right) + (1-\delta)\mathrm{EPC}_i^{cm}$
  - $\mathrm{EPC}_i^{cm} = \frac{1}{n}\sum_{k=0}^{n} \mathrm{EPC}_{s_k}$
  - EPC_{s_k} value of EPC of switch $s_k \in p_i \longrightarrow \mathcal{G}$
  - n number of switches in the path
- EPC - Energy Proportionality Coefficient
  - $\mathrm{EPC_i} = 1\ (increasing\ server\ load\ \rightarrow increasing\ energy)$
  - $\mathrm{EPC_i} = -1\ (increasing\ server\ load\ \rightarrow decreasing\ energy)$
  - $\mathrm{EPC_i} = 0$ (energy consumption does not depend on the load)
- Allocate jobs to the suitable server i with the highest f_i
- α, β, γ can be tuned or **dynamically adapted**

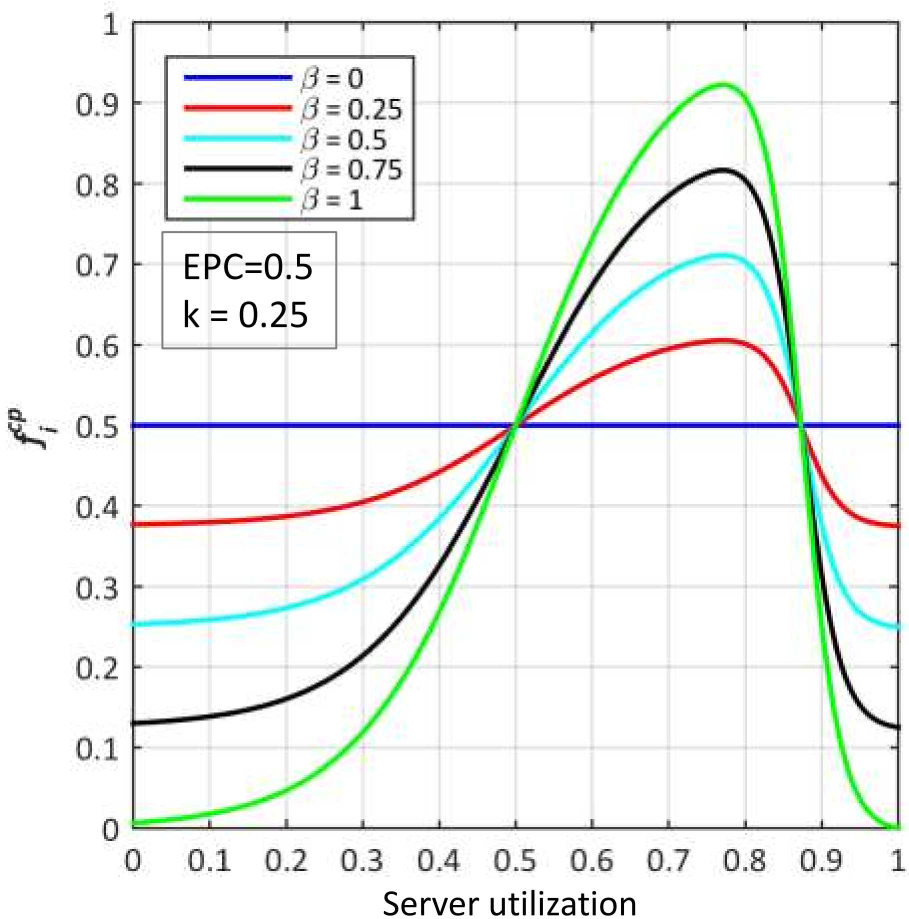
Network Balance ⟵ α ⟶ Energy save (0 … 1)

Efficient Server ⟵ β ⟶ Server Utilization (0 … 1)

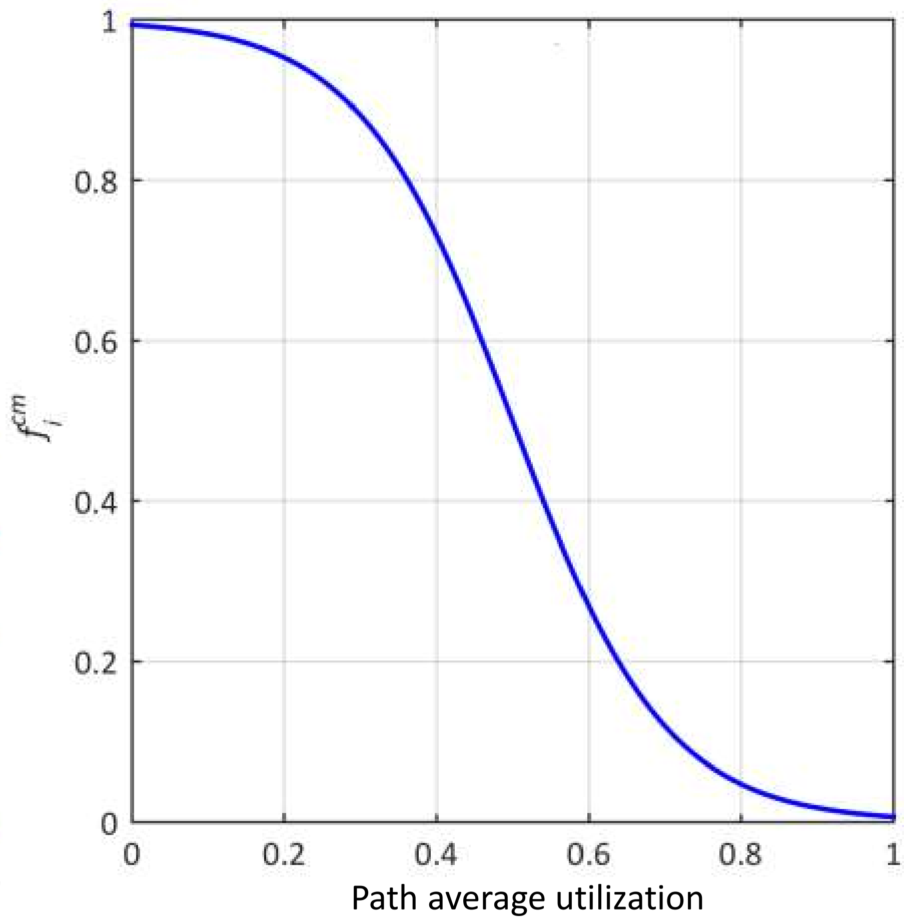
Efficient Switches ⟵ γ ⟶ Low Loaded Path (0 … 1)

Score function

Computation component f_i^{cp}

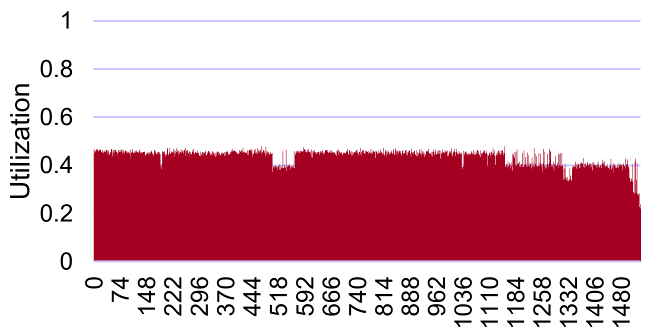


Communication component f_i^{cm}



Balancing

SERVERS

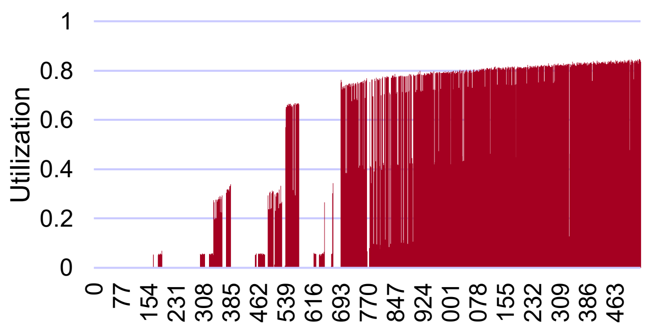


α - β

0.25-1 (**Network balancing**)

Energy 5220 Wh

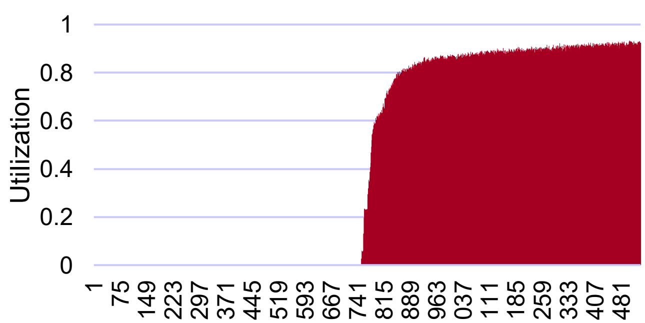
SLA violation rate 0



0.75-0.75

Energy 4455 Wh

SLA violation rate 0



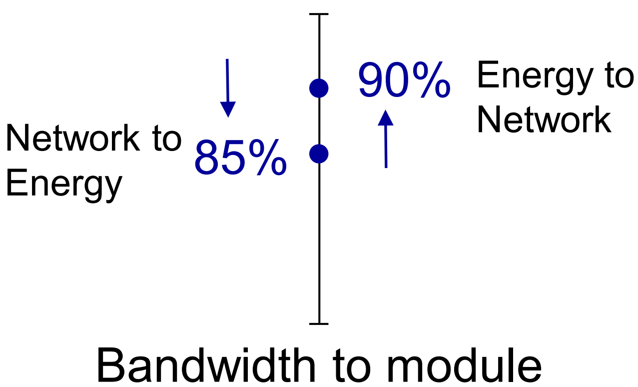
1-0 (**Consolidation**)

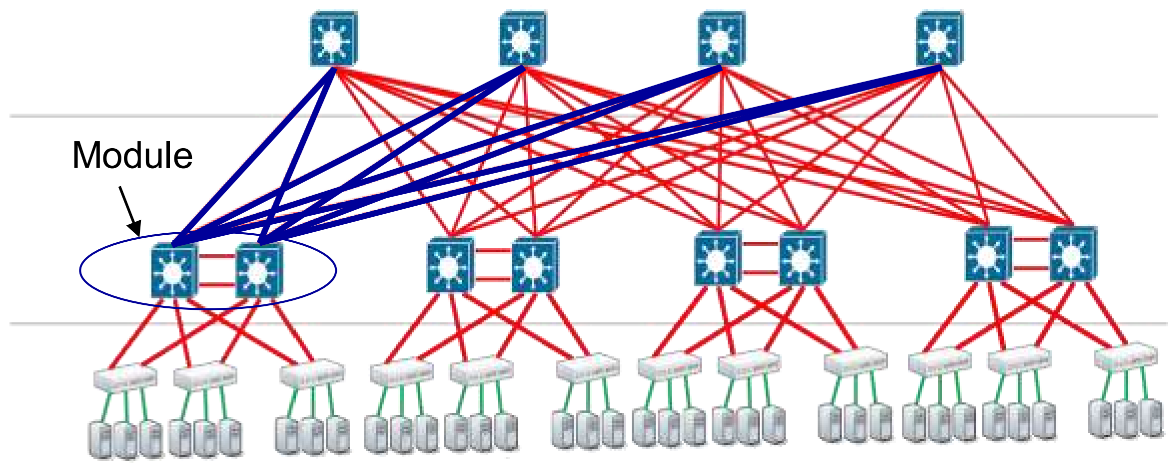
Energy 4204 Wh

SLA violation rate 0.31

Adaptive approach

- Adaptation criteria
  - Amax-ACCURATE (Am-ACCURATE). If Max bandwidth > 90%
  - Aaverage-ACCURATE (Aa-ACCURATE). If Average bandwidth > 90%





Adaptive consolidation
by
Job type Concentration

Motivation

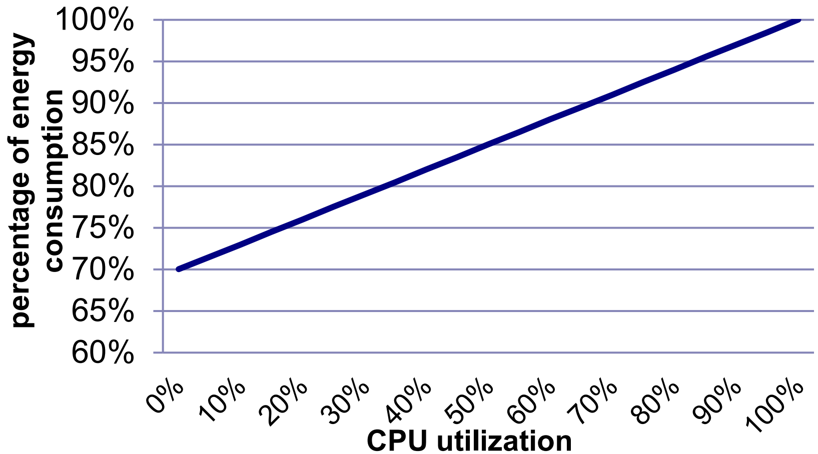
CPU intensive CI	scientific computation, encryption and decryption, compression and decompression
Disk I/O intensive DI	file serving, data mining applications
Memory intensive MI	in-memory caching servers, in-memory database servers
Network I/O intensive NI	Web servers, as well as network load balancers

Resource contention results in a poor performance and high energy consumption

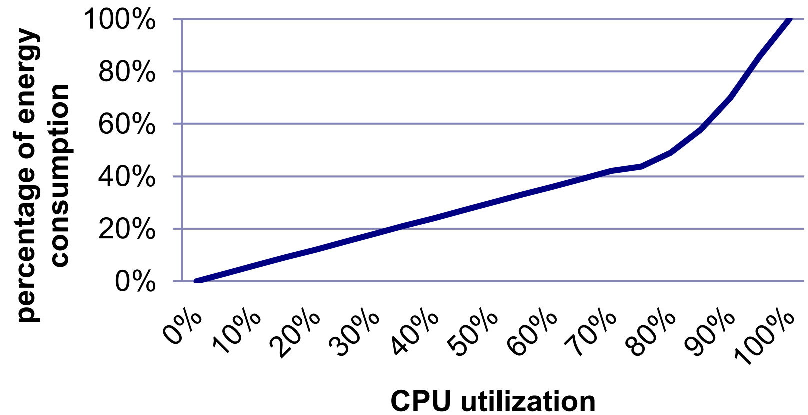
Benchmarks

Benchmark	CI	MI	NI	DI
LINPACK	●			
STREAM		●		
SysBench	●	●		●
iperf			●	
IOR				●
IOzone				●
NPB	●	●		●
Netperf			●	
SPEC	●	●		

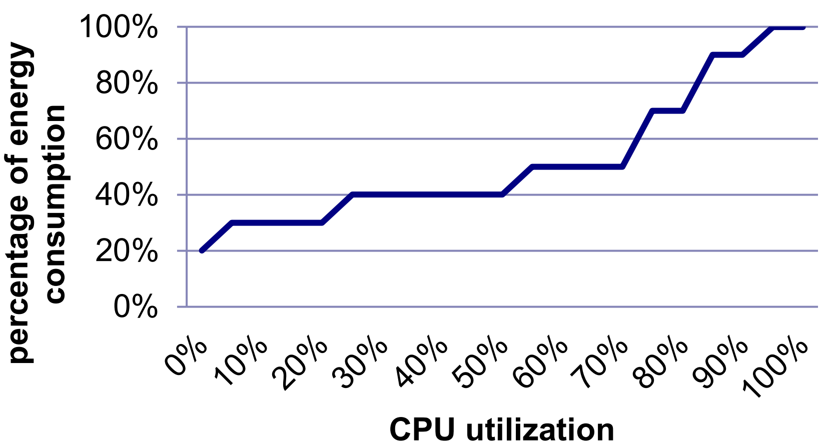
Typical energy models



A. Beloglazov, et.al “Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing” 2012.

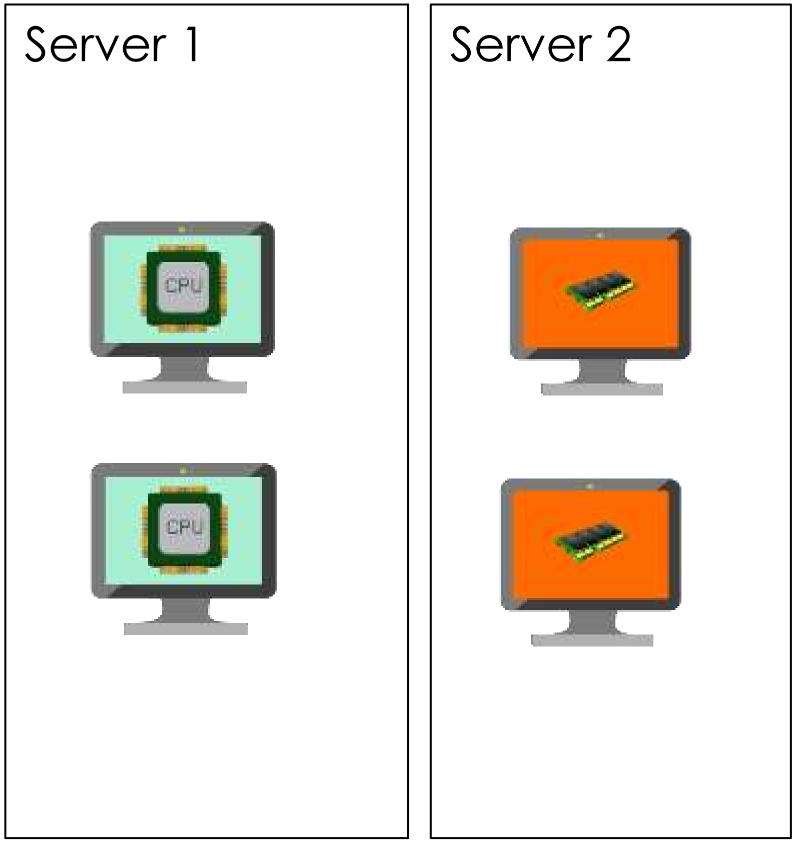


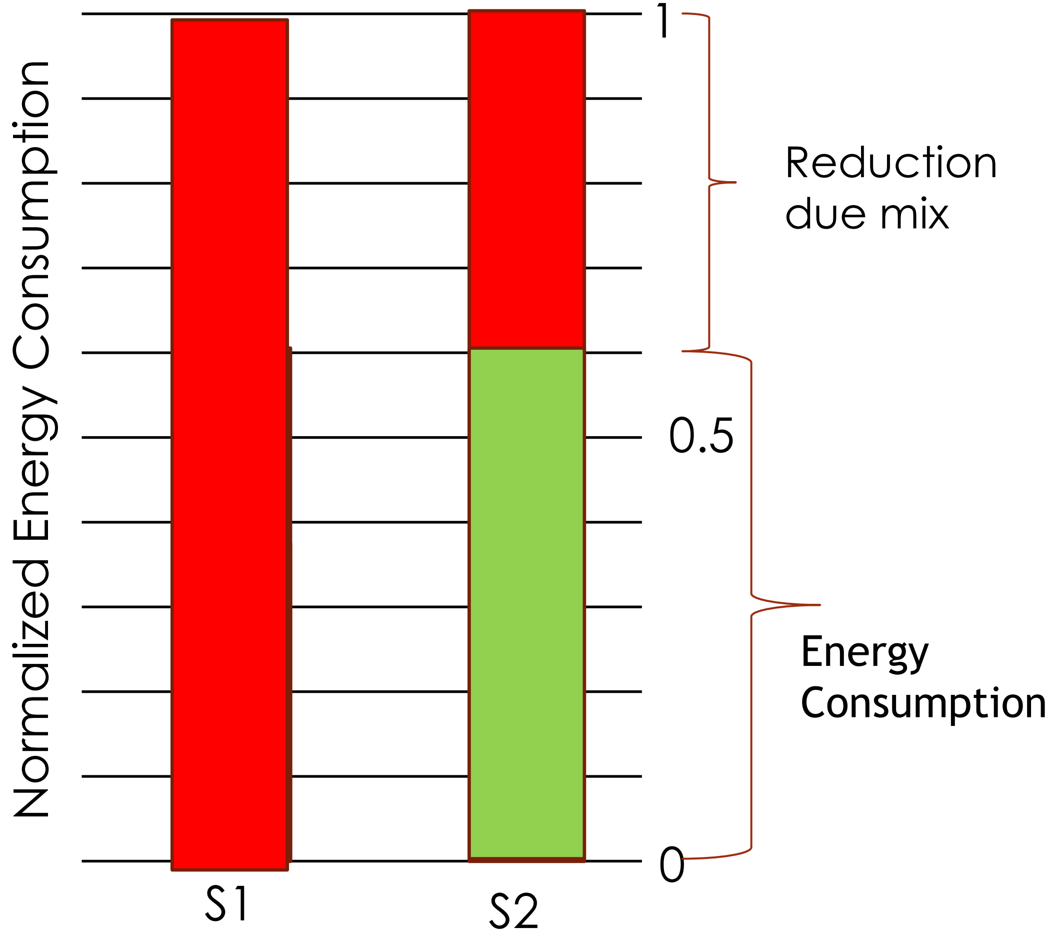
Y. Gao, et. al “An Energy and Deadline Aware Resource Provisioning, Scheduling and Optimization Framework for Cloud Systems,” 2013.



C.-H. Hsu, et. al, “Optimizing Energy Consumption with Task Consolidation in Clouds,” 2014.

Concentration





Proposed energy model

Processor's power consumption depends on

- **Utilization**
- **Job type combination (Contention)**

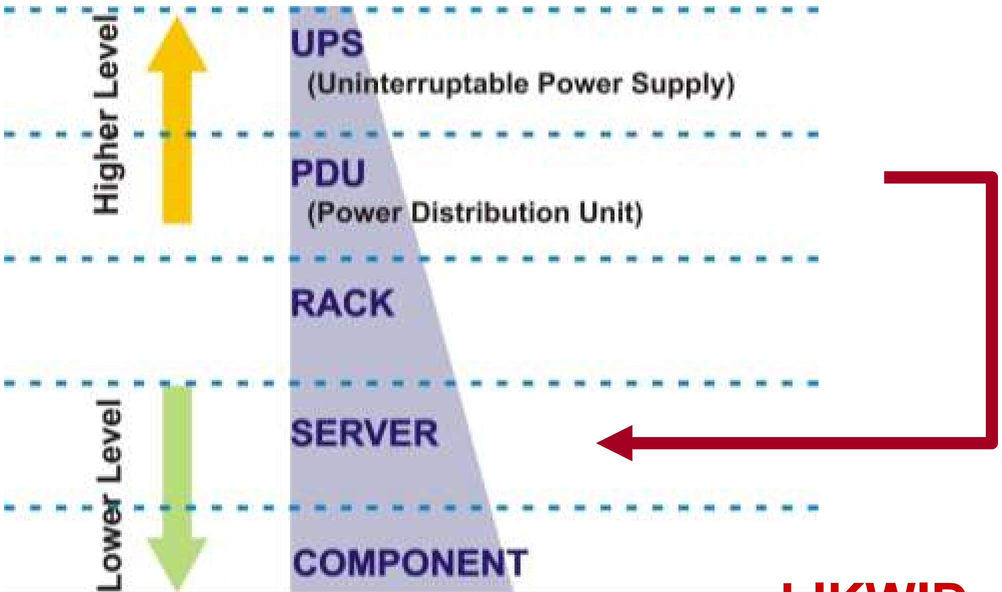
$$e(t) = o(t)\left(e_{idle} + e_{used}(t)\right)$$

$$e(t) = o(t)\left(e_{idle} + (e_{max} - e_{idle}) * \boldsymbol{F}(\boldsymbol{t}) * \boldsymbol{g}\left(\boldsymbol{\alpha}_{\boldsymbol{a}_{\boldsymbol{i}}}(\boldsymbol{t})\right)\right)$$

$$g\left(\alpha_{a_i}(t)\right) = 1$$ If **no job combination** is considered

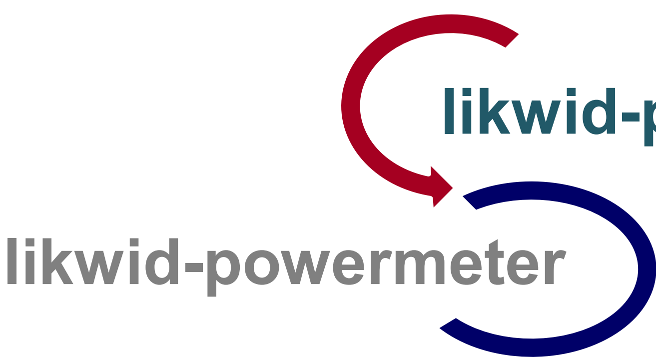
To consider job combinations, **we use "job concentration" approach**

Power distribution



Benchmark: SysBench

LIKWID

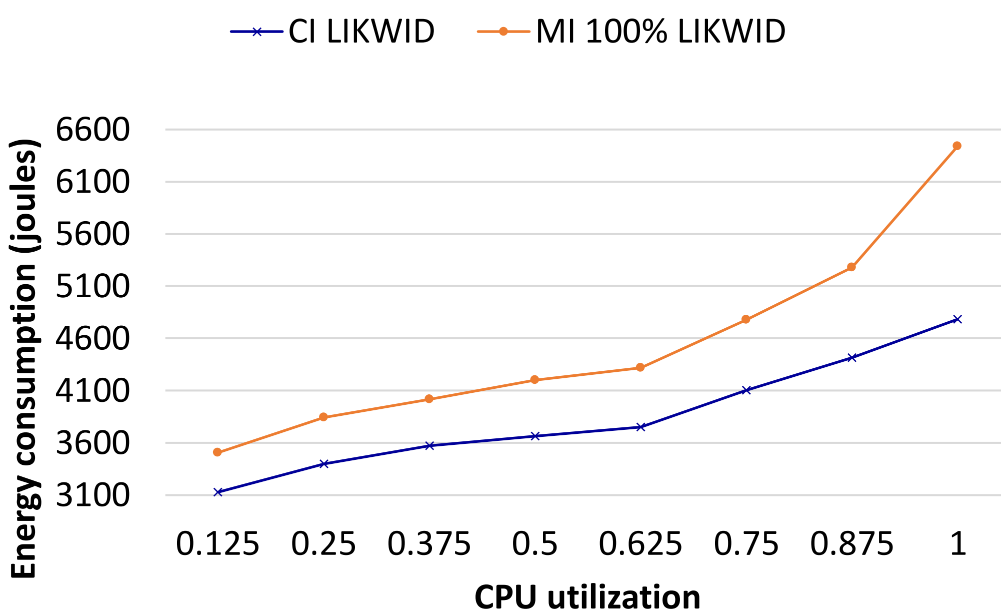


Power Distribution Unit (**PDU**)

Utilization function $F(t)$

$f_d(U_d(t))$ - fraction of power consumption when a CI or MI application is executed

$$F(t) = \sum_{\forall d} f_d(\, U_d(t)) \;, 0 \leq F(t) \leq 1, d \in \{CI, MI\}$$



$U_T(t)$ - the total CPU utilization at time t:

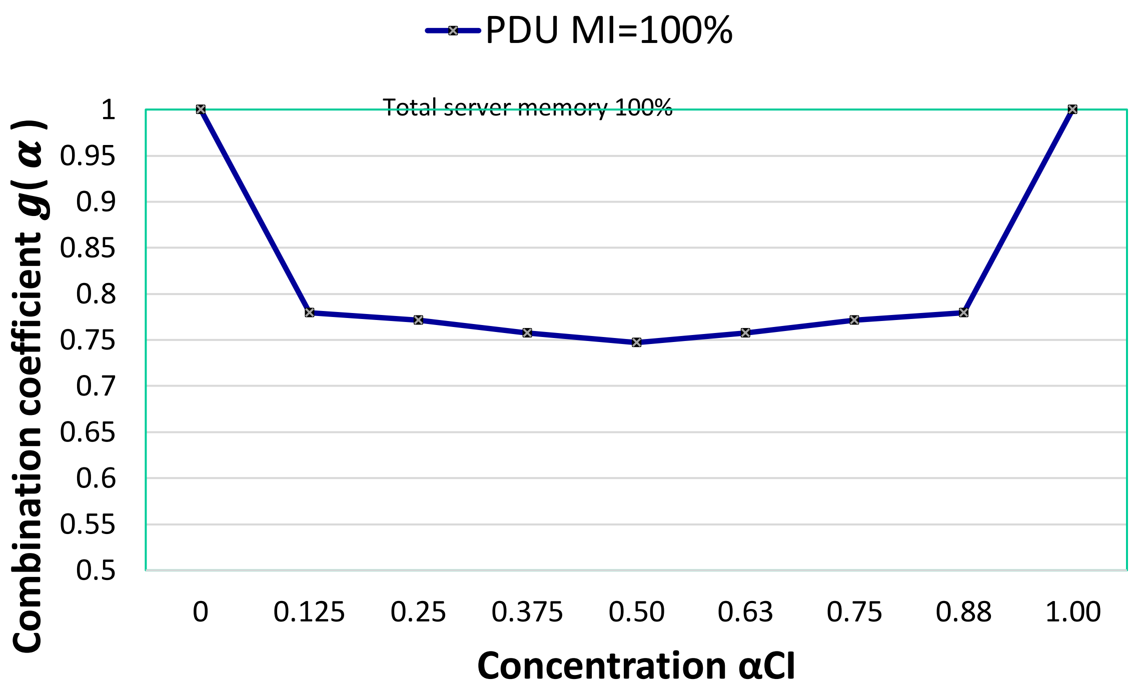
$$U_T(t) = \; U_{CI}\,(t) + U_{MI}(t)$$

$g(\alpha_{a_i}(t))$

$$U_T\ (t) = \sum_{\forall\ a_i \in A} U_{a_i}(t)$$

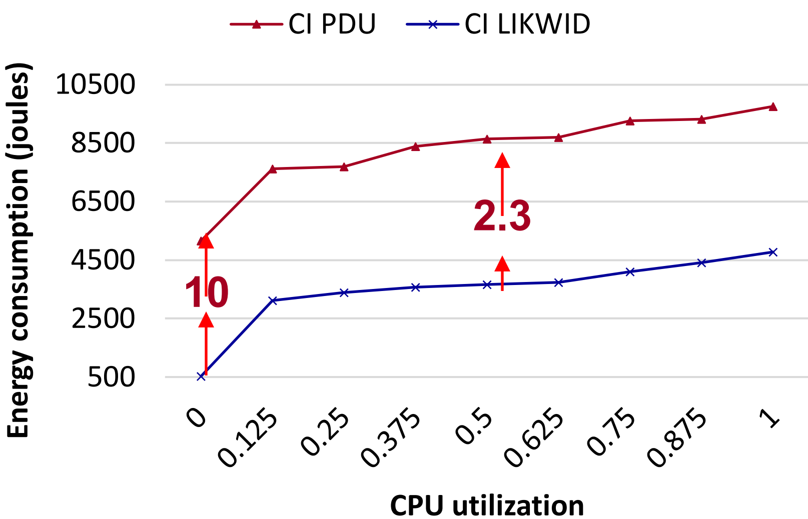
$$\alpha_{a_i}(t) = \frac{U_{a_i}(t)}{U_T(t)}$$

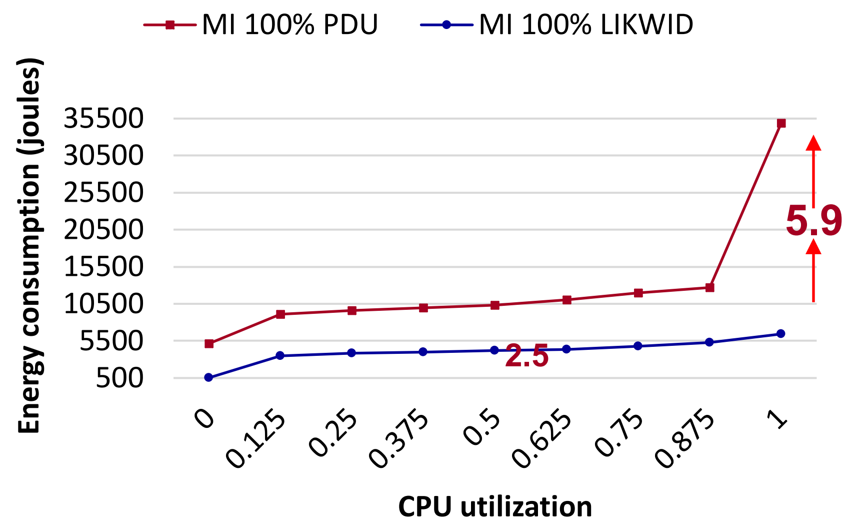
$g(\alpha_{CI}(t))$

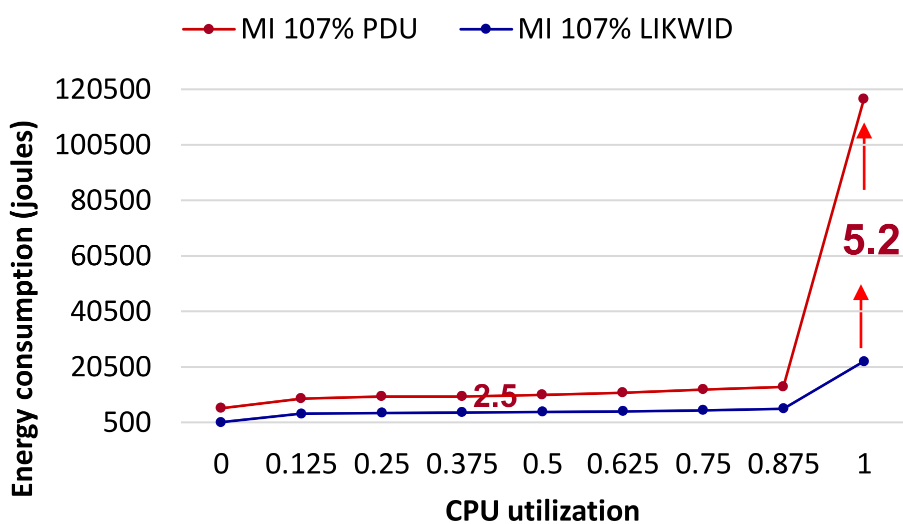


$$\alpha_{MI}(t) = 1 - \alpha_{CI}(t).$$

Energy consumption PDU vs LIKWID







Job allocation strategies

Type	Strategy	Description
Knowledge Free	*Rand*	Allocates job j to a suitable machine randomly using a uniform distribution in the range $[1..m]$.
	FFit (First Fit)	Allocates job j to the first machine available and capable to execute it.
	RR (Round Robin)	Allocates job j to the machine available and capable to execute by Round Robin strategy
Energy -aware	*Min_e* (Min-energy)	Allocates job j to the machine with minimum power consumption at time r_j : $min_{i=1..m}\left(e_i^{proc}(r_j)\right)$
Utilization Aware	*Min_u* (Min-utilization)	Allocates job j to the machine with minimum total utilization at time r_j $min_{i=1..m}(u_i^{proc})$
	Max_u (Max-utilization)	Allocates job j to the machine with maximum total utilization at time r_j $max_{i=1..m}(u_i^{proc})$
Job type	*MinU_MinC* (Min utilization and Min concentration)	Allocates job j to the machine in the subset of machines with minimum total utilization at time r_j $min_{i=1..m}(u_i^{proc})$ and minimum concentration of jobs of the same type.
	MaxU_MinC *(Max utilization and Min concentration)*	Allocates job j to the machine in the subset of machines with maximum total utilization at time r_j $max_{i=1..m}(u_i^{proc})$ and minimum concentration of jobs of the same type.
	Min_ujt (Min-util_job_type)	Allocates job j to the machine with minimum utilization of jobs of the same type at time r_j
	Min_c (Min-concentration)	Allocates job j to the machine with minimum concentration of jobs of the same type at time r_j

Modeling applications with communications and uncertainty

Modeling Applications

How to model applications with communication processes?

Two known approaches:

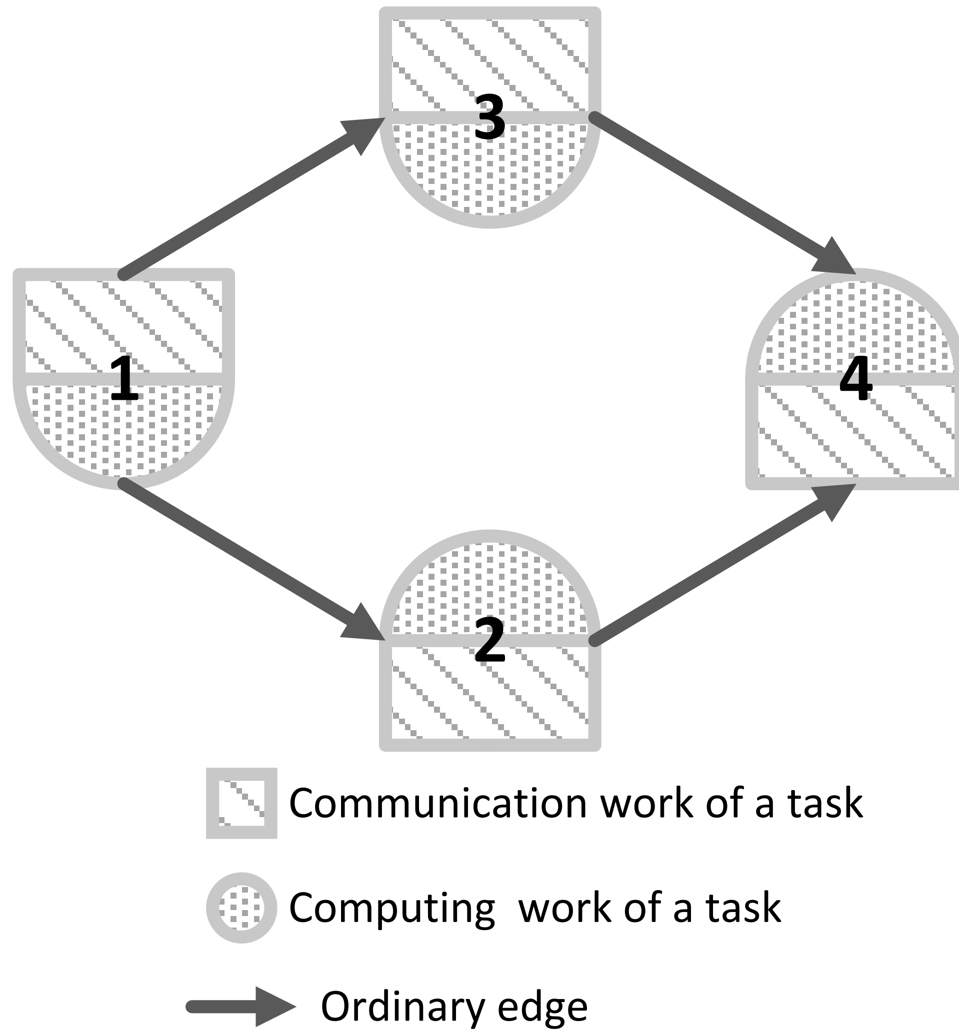
- **CU-DAG** Communication-unaware model
- **EB-DAG** Edges-based model

New approach:

- **CA-DAG** **Communication-aware model**

Communication-unaware model

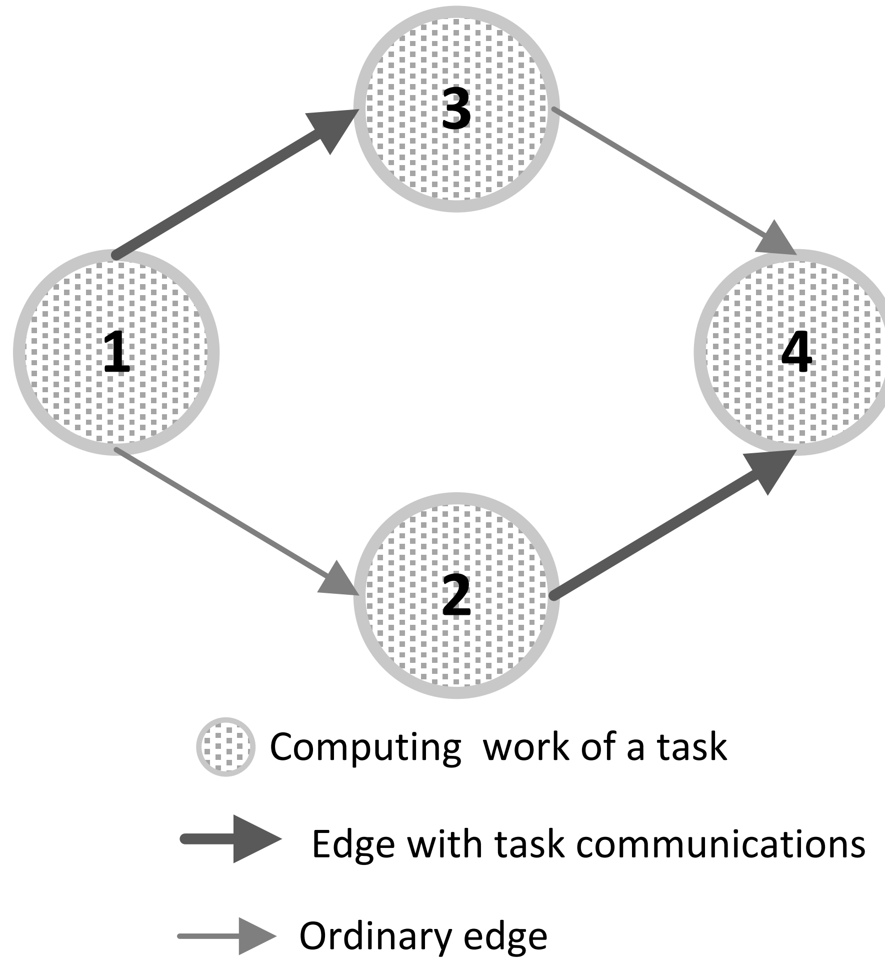
- vertex represents both computing and communication
- Edges: dependencies



- Main drawback
  - Difficult to make separate scheduling decisions

Edge-based model

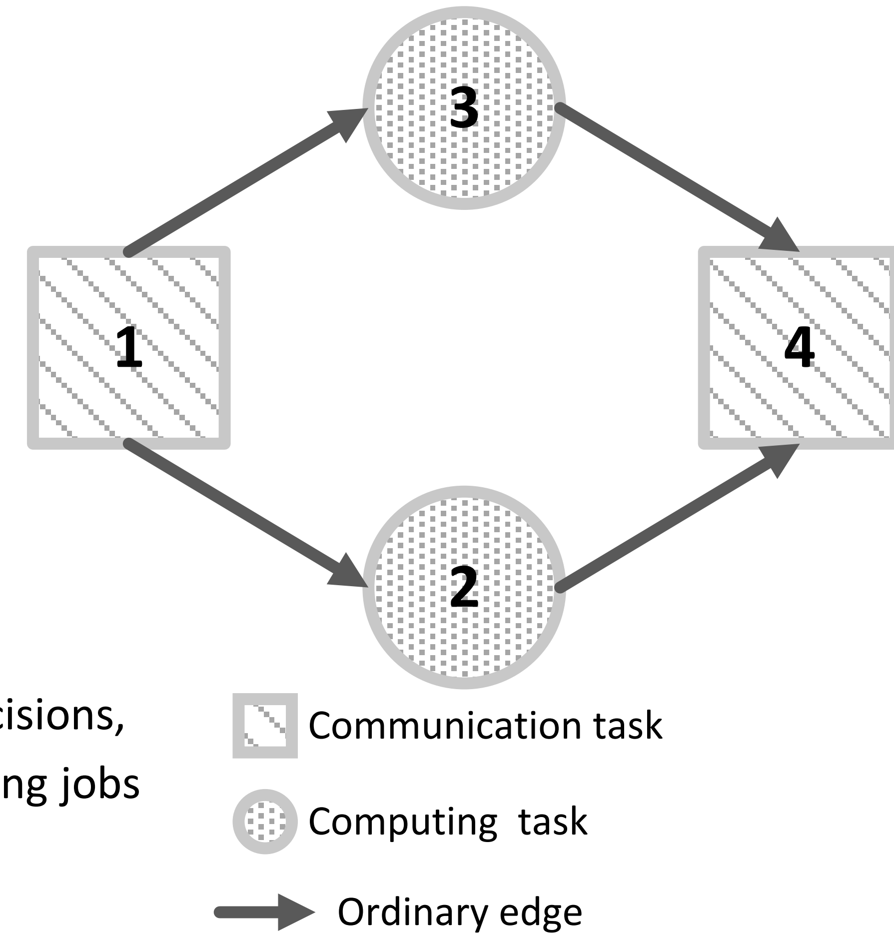
– Vertex represents computing
– Edges represent communication



- Main drawback
  – Two computing tasks cannot have the same data transfer to input
  – singe edge cannot lead to two different vertices

CA-DAG: Communication-Aware model

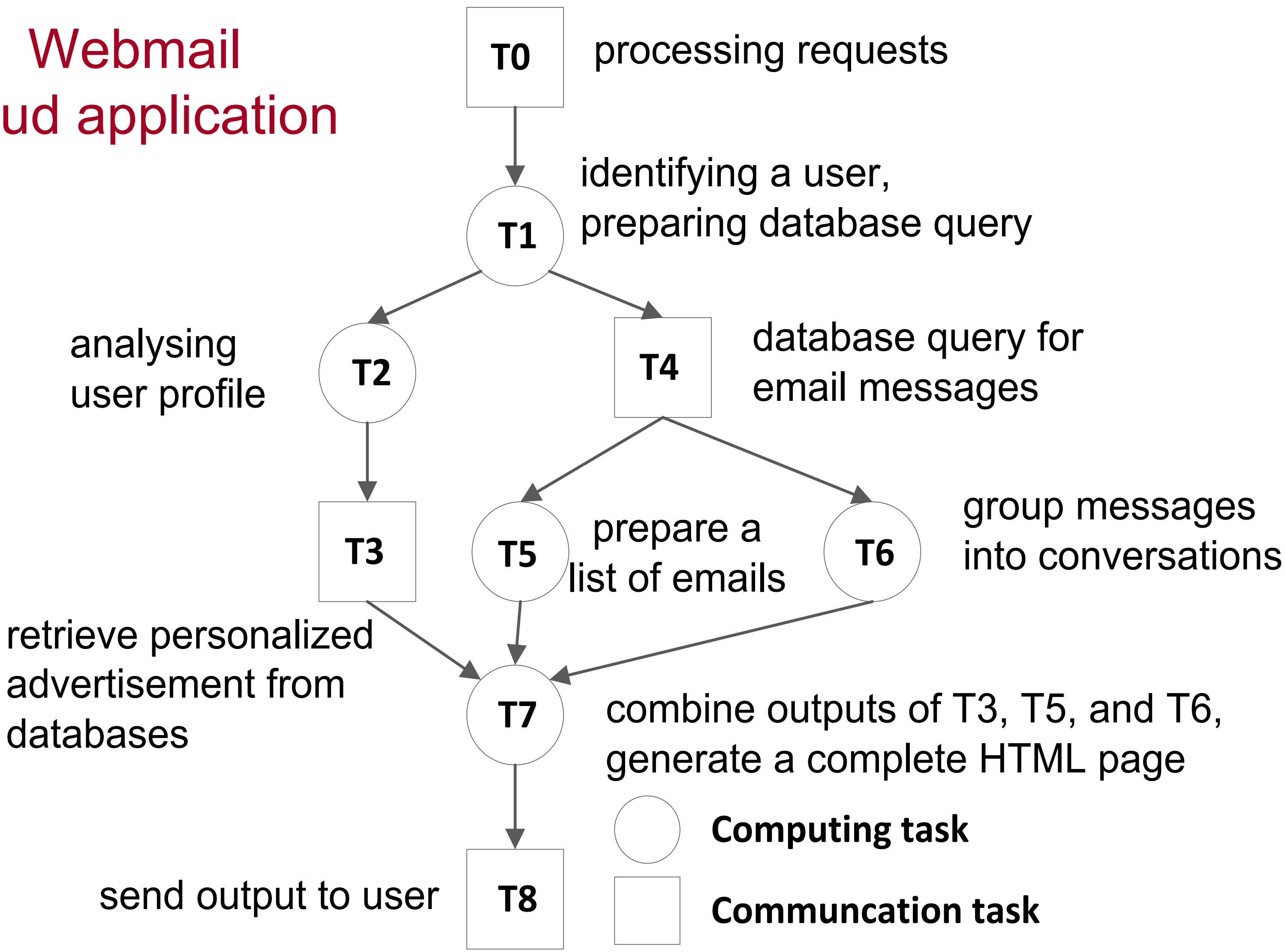
– Two types of vertices:
  - one for computing
  - one for communications
– Edges define dependences between tasks and order of execution



- Main advantage
  – Allows separate resource allocation decisions,
  – assigning processors to handle computing jobs
  – network resources for information transmissions

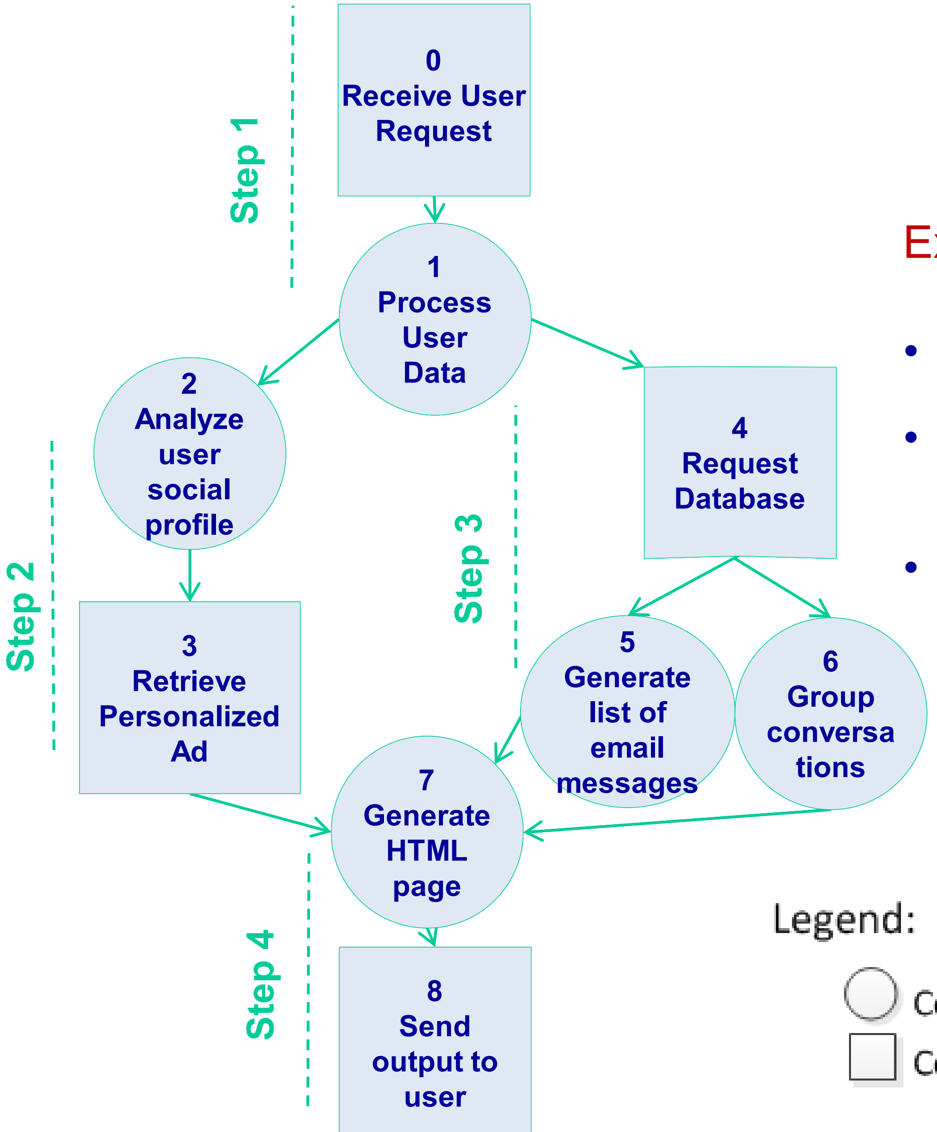
CA-DAG: Communication-Aware DAG

Webmail
cloud application



CA-DAG: Communication-Aware DAG

by Dzmitry Kliazovich IEEE Cloud'13



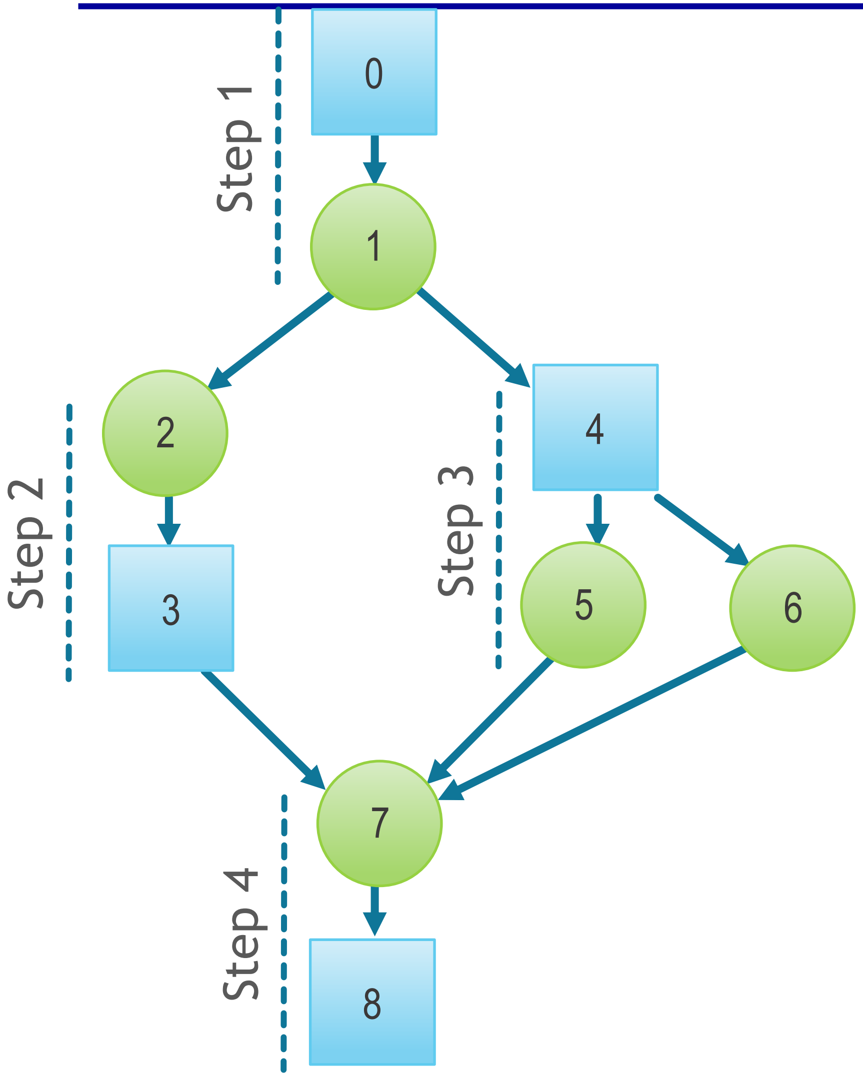
Example of webmail cloud application

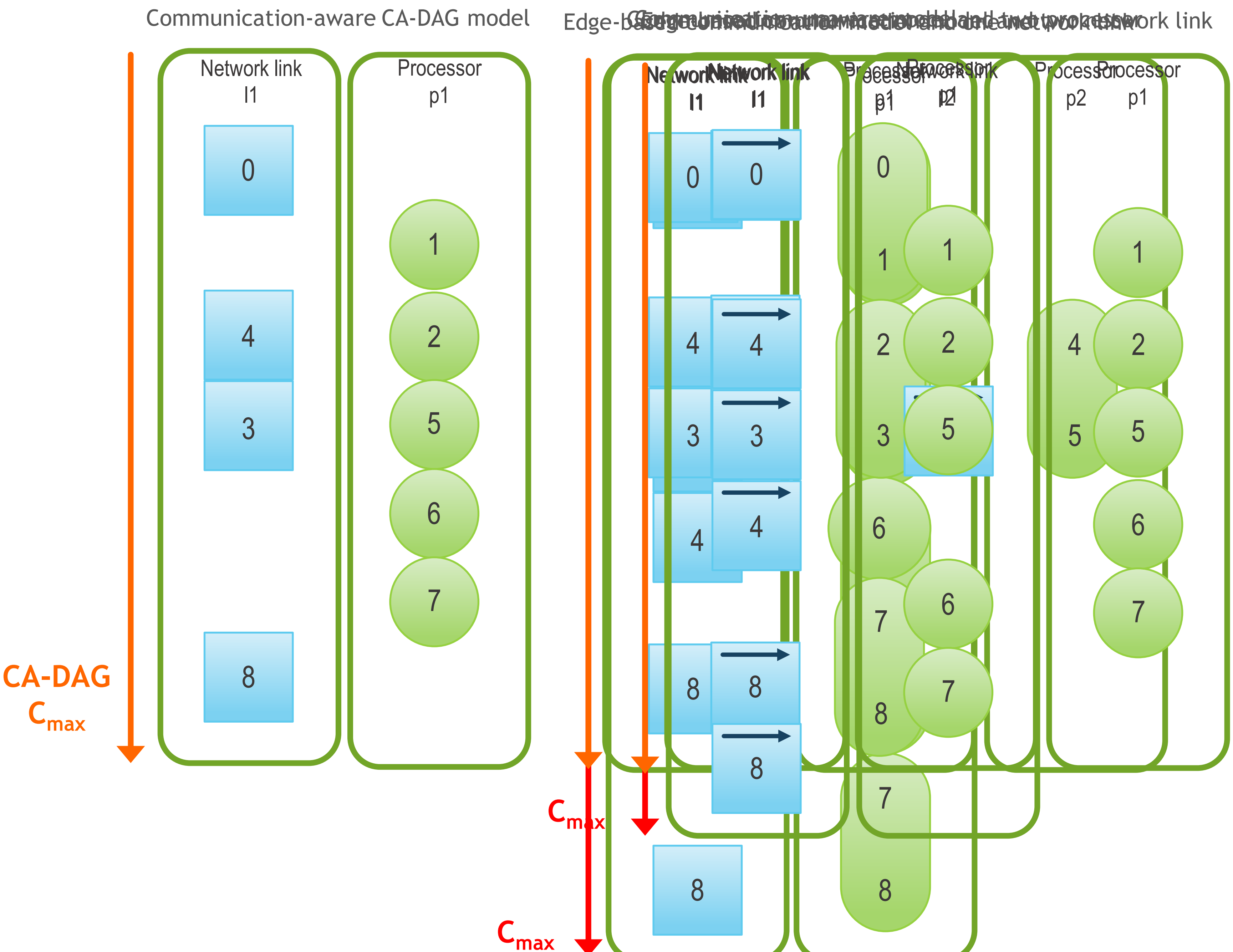
- Step 1: Receive user request and process it
- Step 2: Generate personalized advertisement
- Step 3: Request list of email messages from database

Step 4: Generate HTML pages and send it to the user

Legend:

Schedules

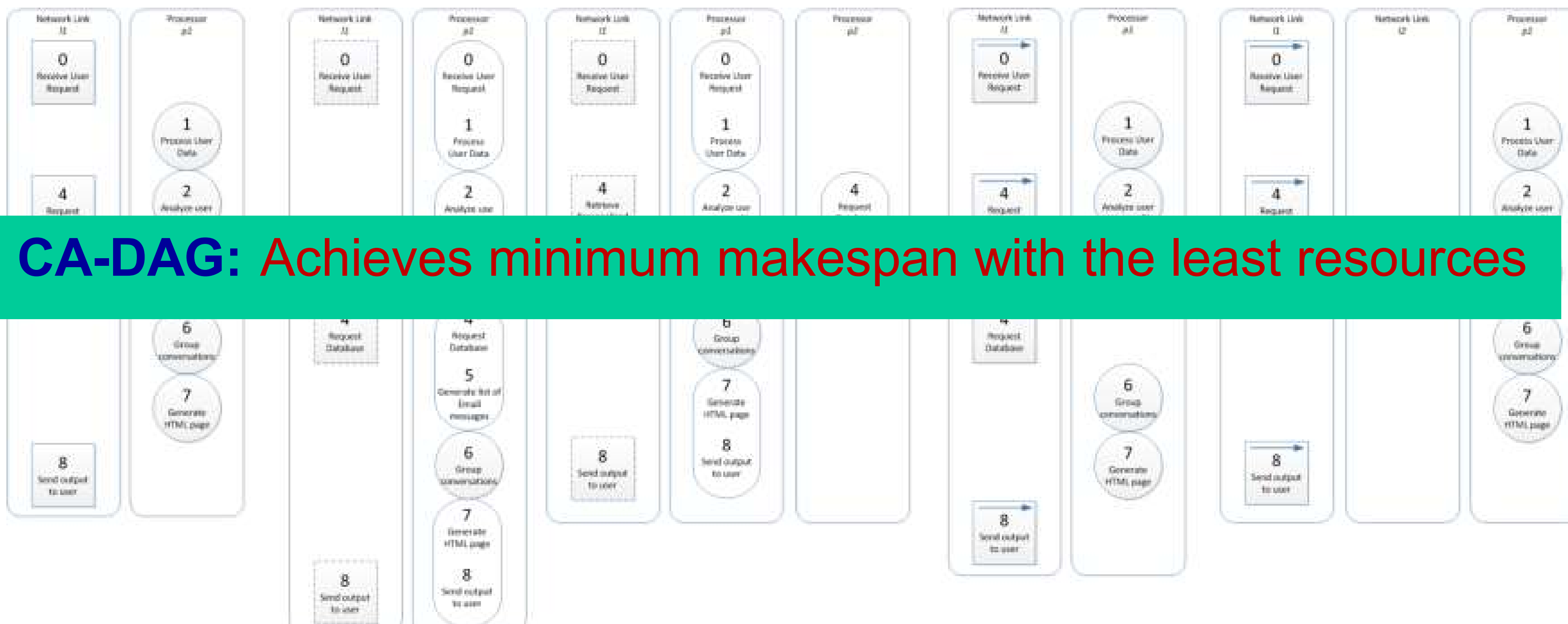




Communication-aware CA-DAG model
Network link
l1
Processor
p1
p2
0
1
2
3
4
5
6
7
8
CA-DAG
Cmax
Cmax
Cmax

Comparison of schedules

CA-DAG model | Communication-unaware model | Edges-based model

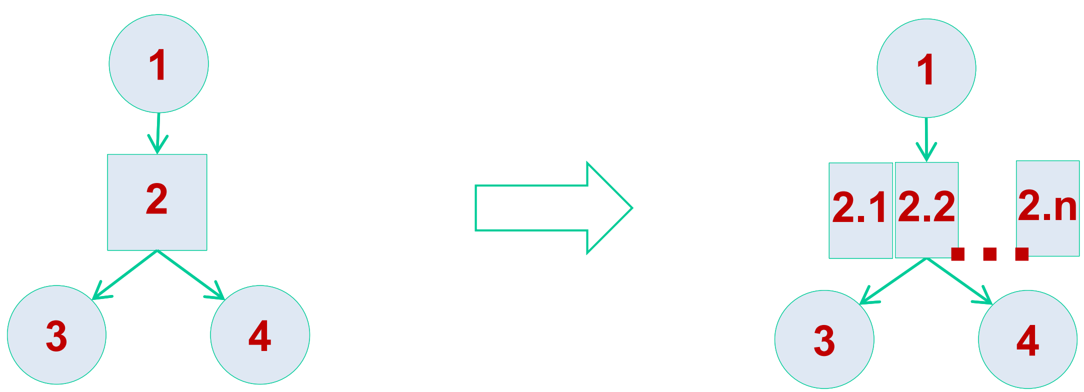


CA-DAG: Achieves minimum makespan with the least resources

# of Processors	# of Network links	Communication-unaware model	Edges-based model	**Proposed CA-DAG model**
1	1	9	8	**7**
1	2	9	7	**7**
2	1	7	8	**7**

Task Parallelization

- Each communication task/vertex can be divided into n different independent communication tasks that can be executed in parallel

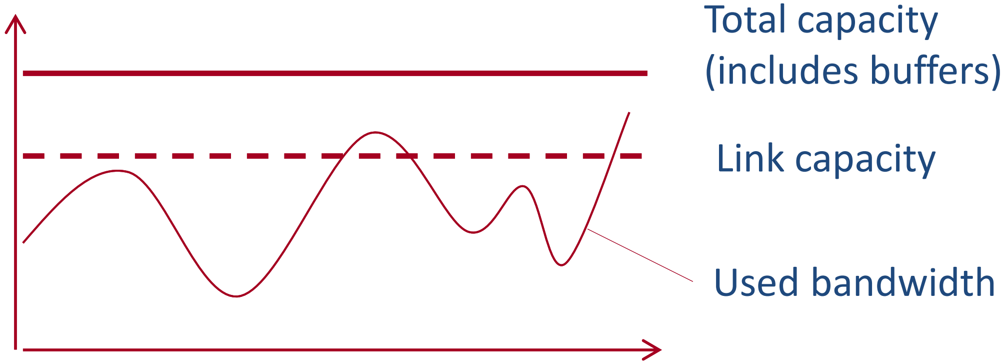


Communication uncertainty

- Static mapping of DAG to communication system with uncertainty is not efficient

CA-DAG can adapt to:

- Communication uncertainty
- Calculation uncertainty
- Available connections and bandwidth
- Parallel transmission



Adaptive energy efficient scheduling in Peer-to-Peer desktop grids

Knowledge Free Scheduling

Andrei Tchernykh — **CICESE Research Center, Mexico**

Aritz Barrondo

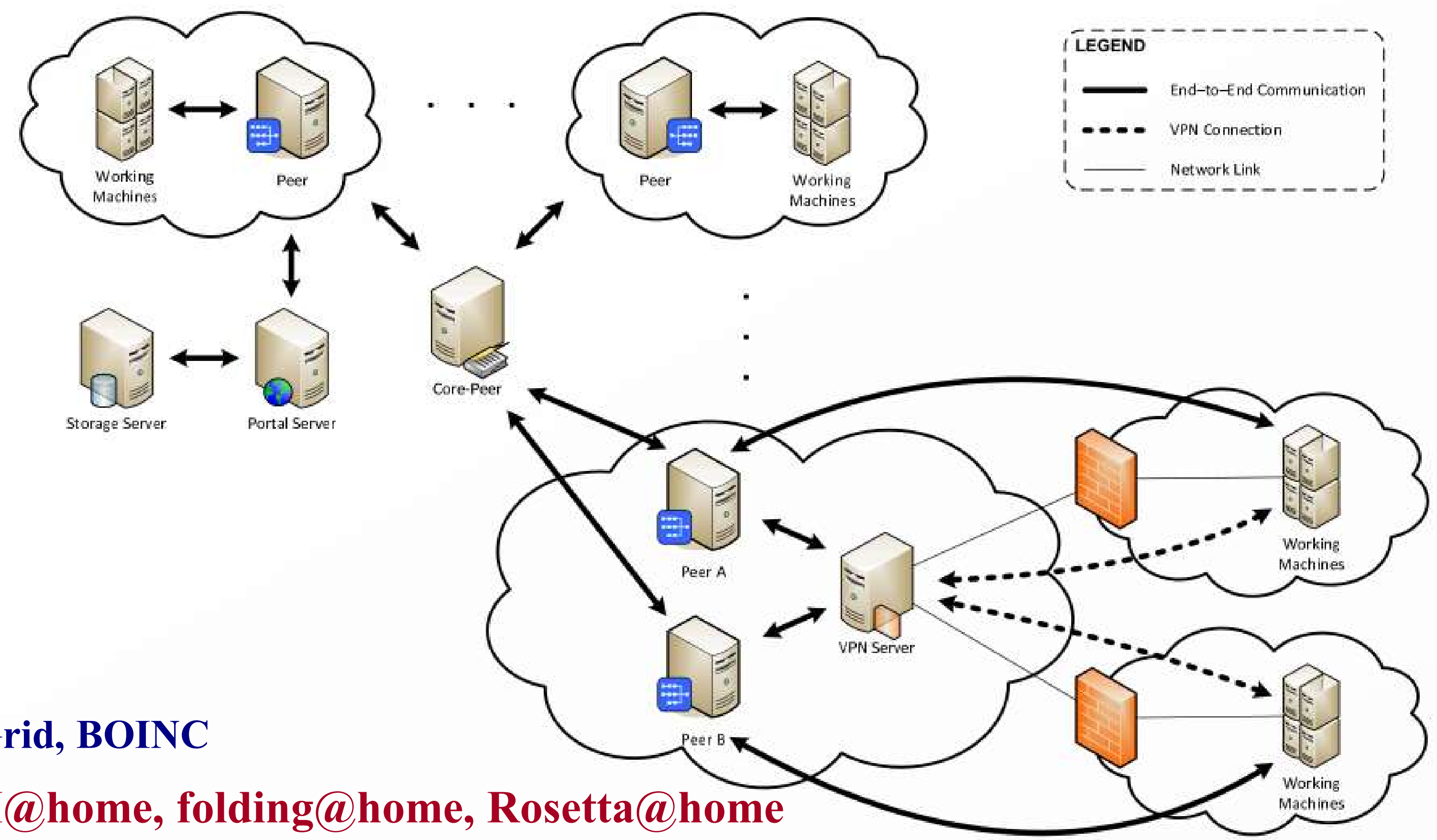
Johnatan E. Pecero — **University of Luxembourg, Luxembourg**

Elisa Schaeffer — **Universidad Autónoma de Nuevo León, Mexico**

Future Generation Computer Systems. 2013

Knowledge Free Scheduling

Knowledge-Free Scheduling



Architecture of ShareGrid.

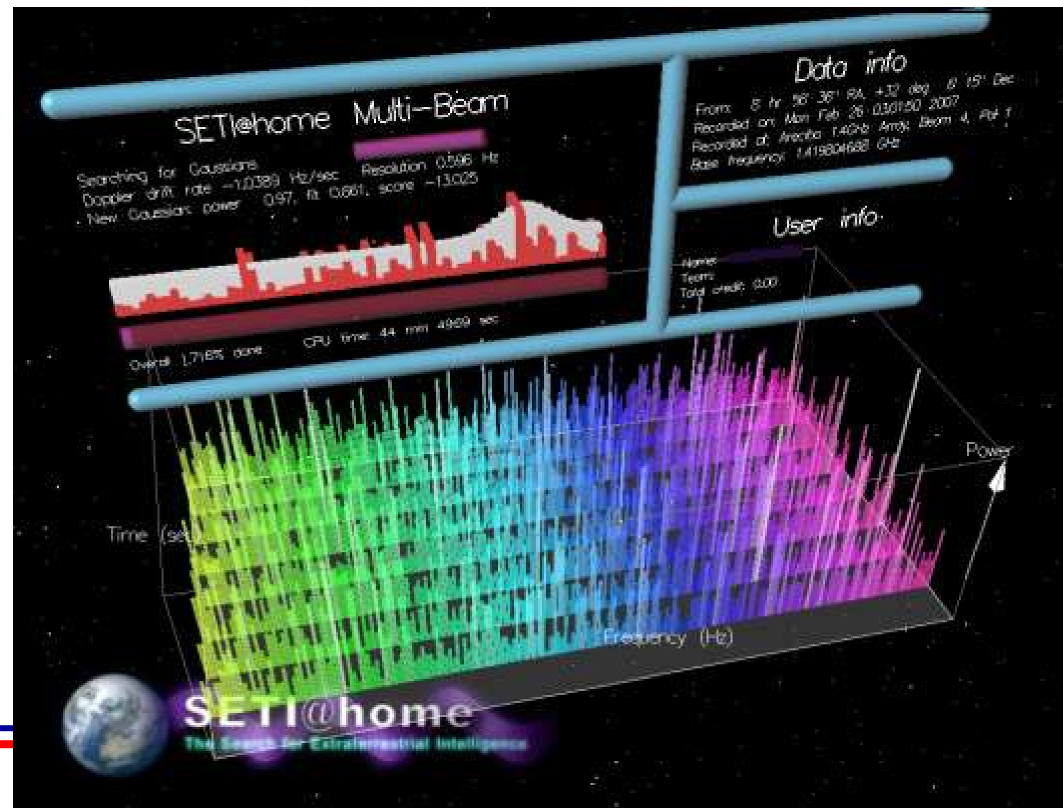
OurGrid, BOINC

SETI@home, folding@home, Rosetta@home
Einstein@home,
+50 projects

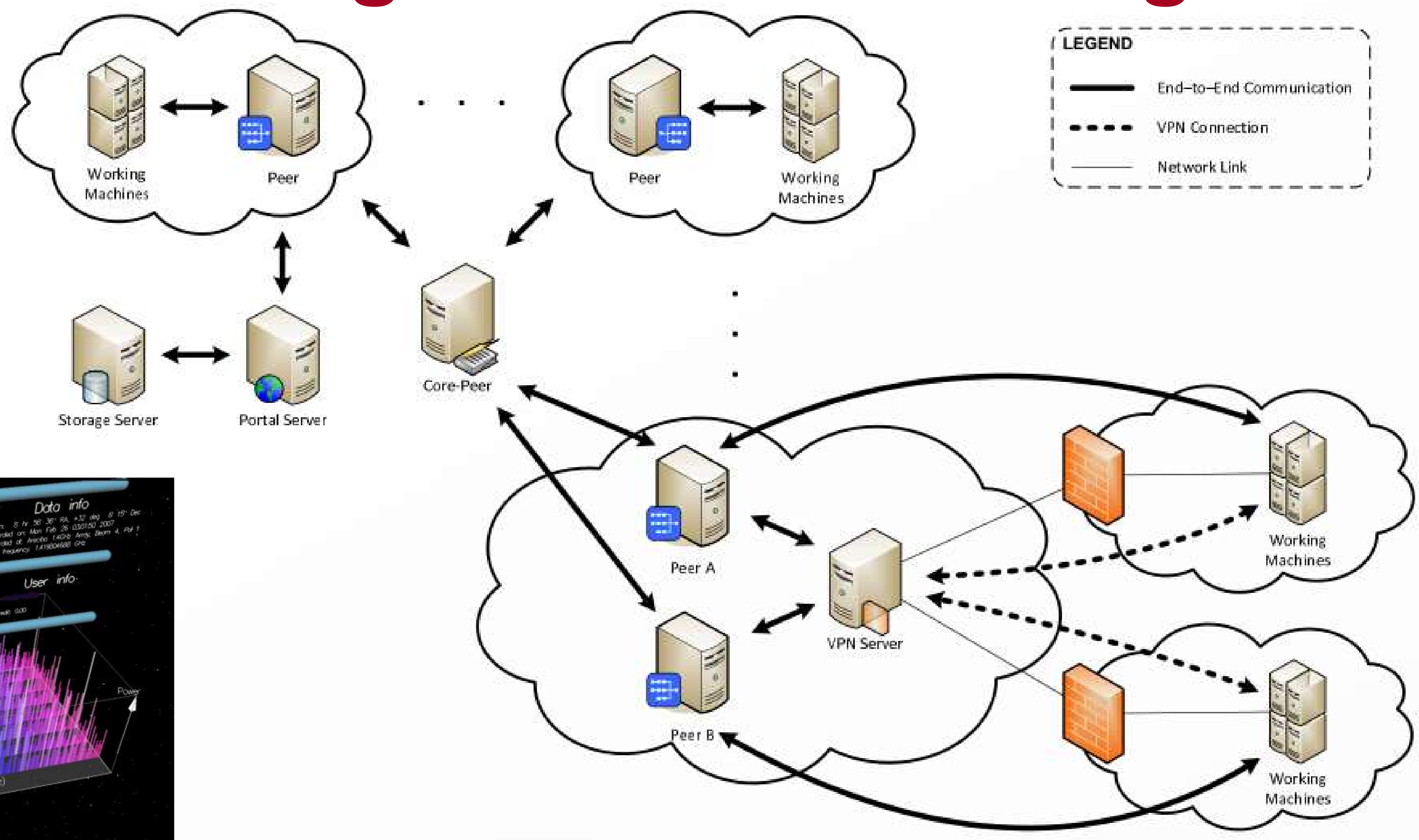
Berkeley Open Infrastructure for Network Computing - BOINC has about 527,880 active computers (hosts) worldwide processing on average 5.428 petaFLOPS as of August 8, 2010

SETI@home

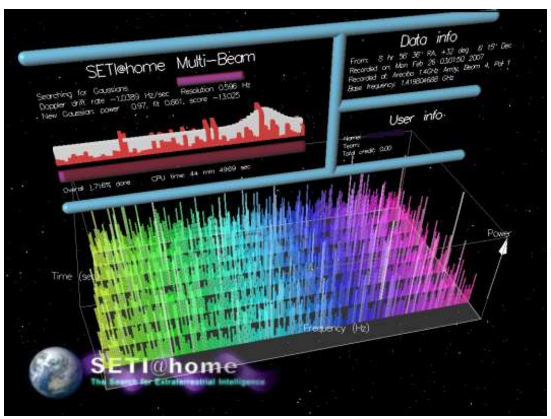
folding@home



Knowledge-Free Scheduling

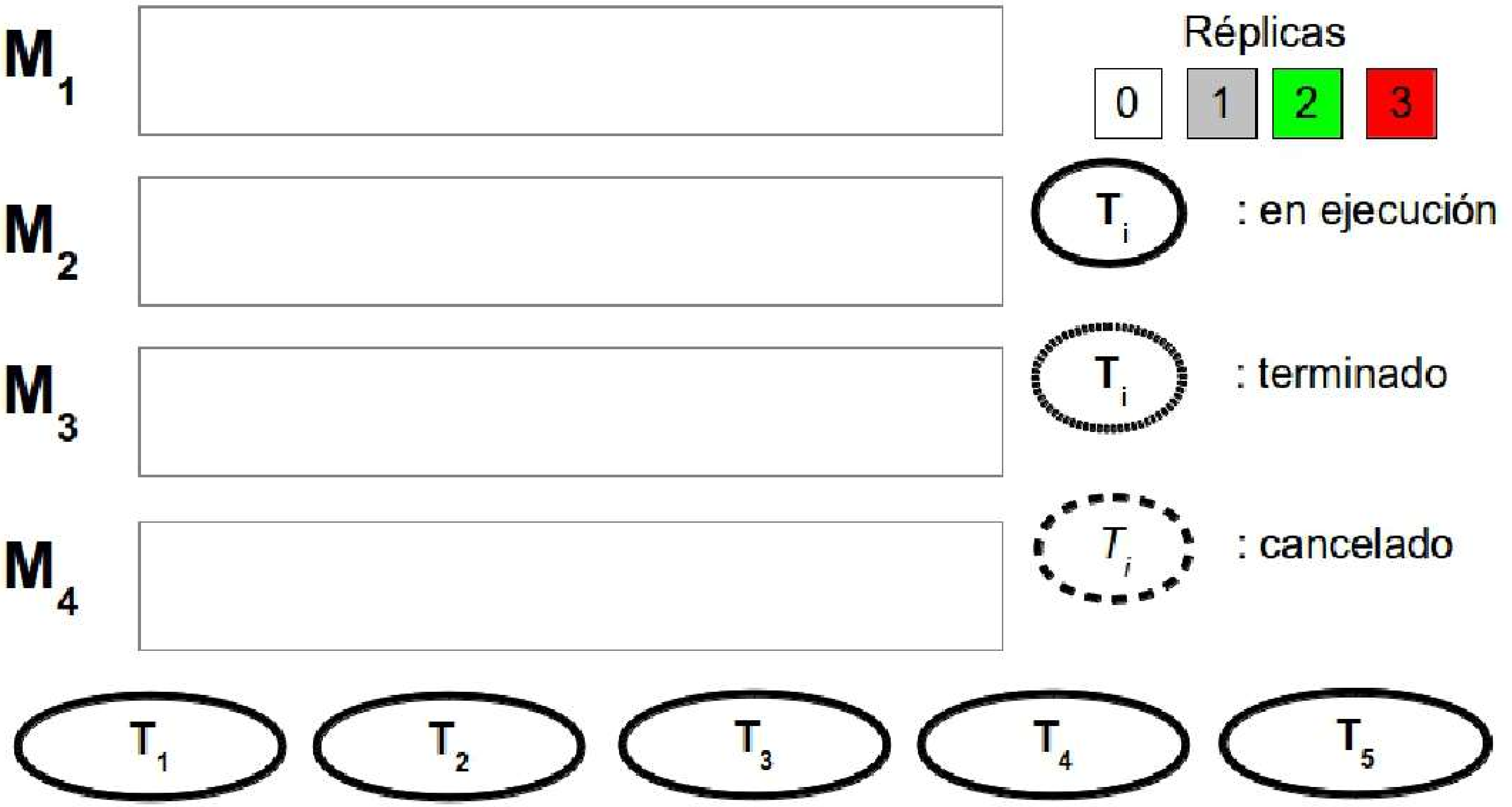


Architecture of ShareGrid.

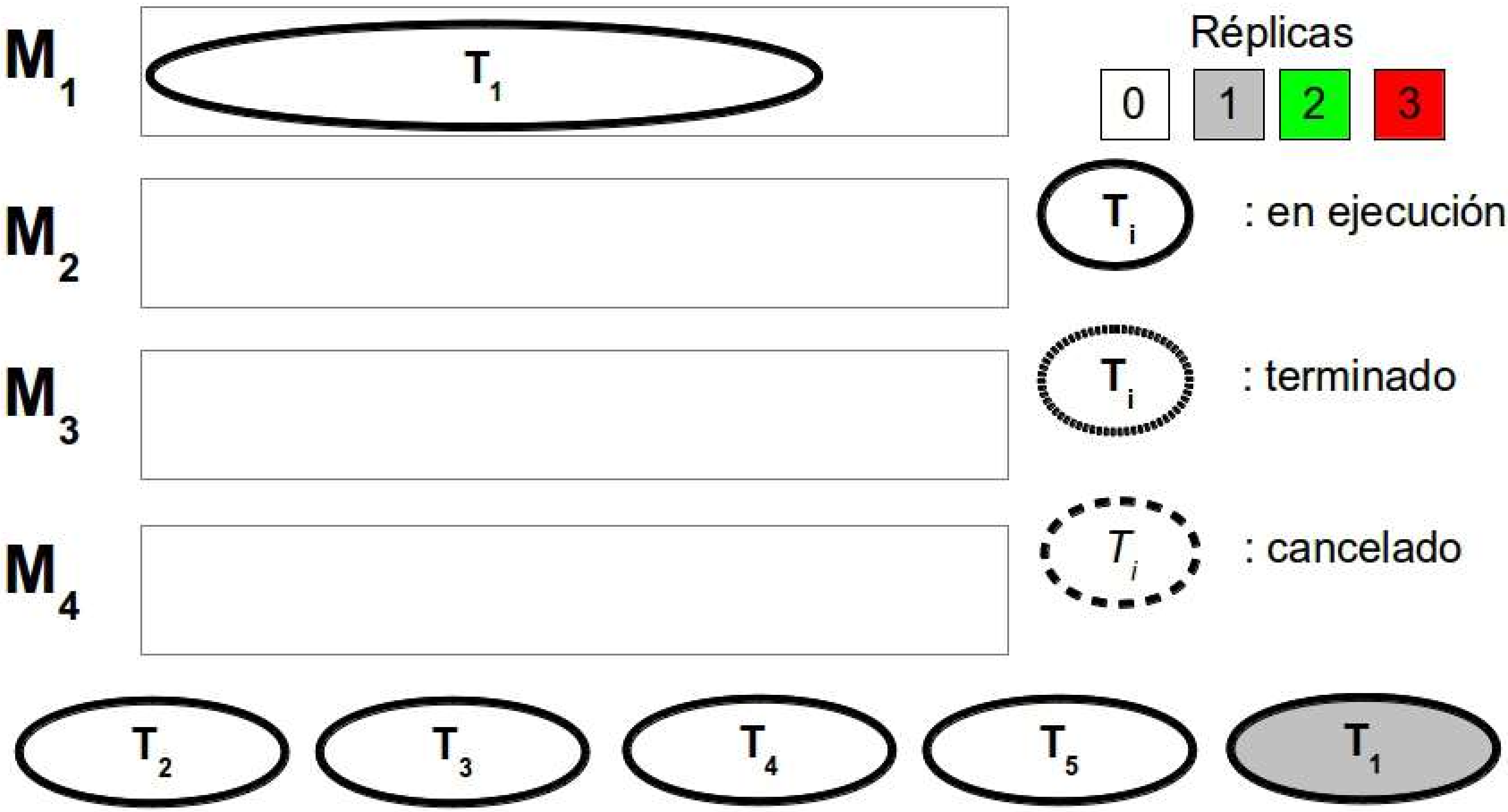


SETI@home

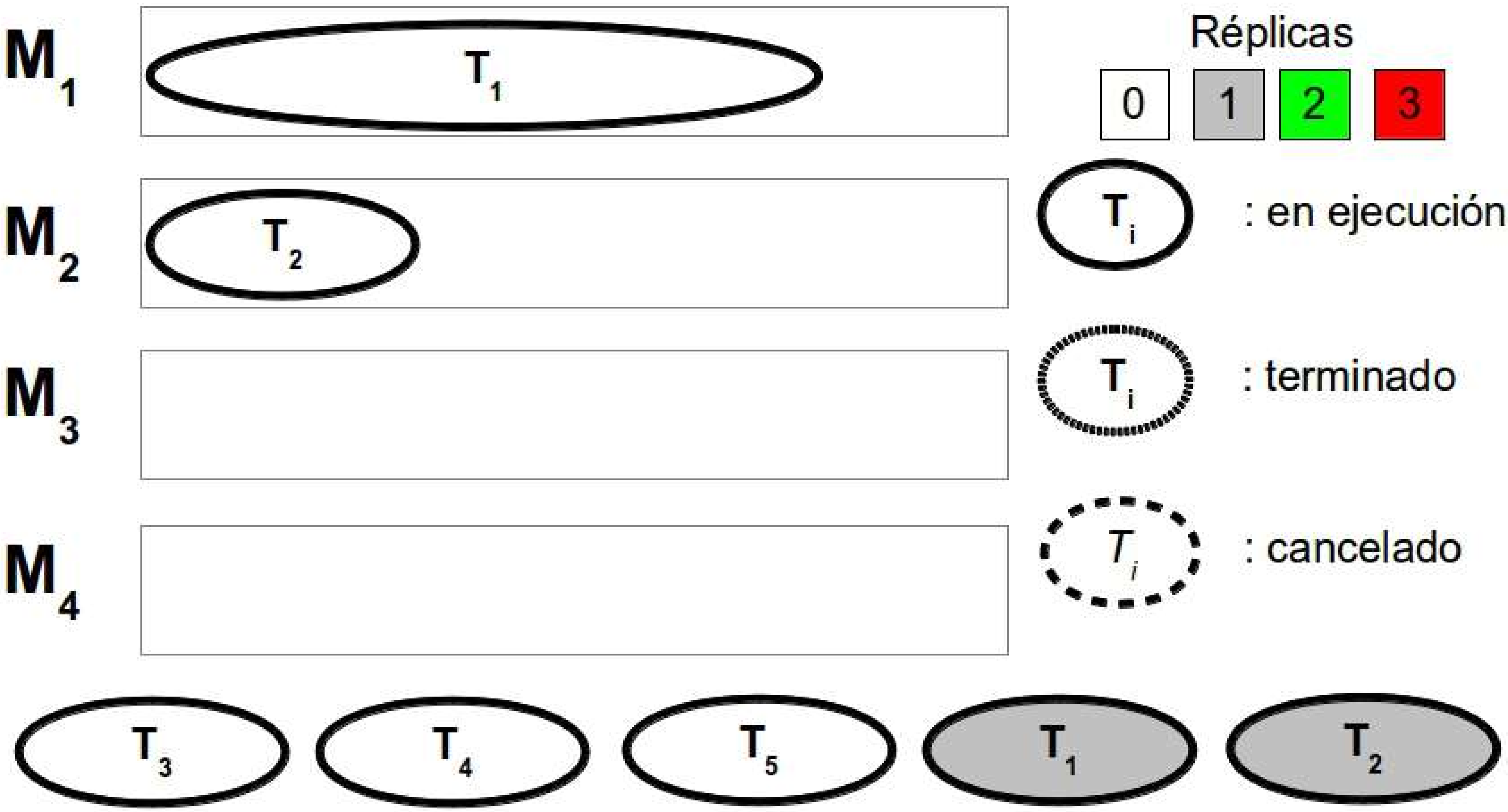
Work Queue with Replication (WQR)



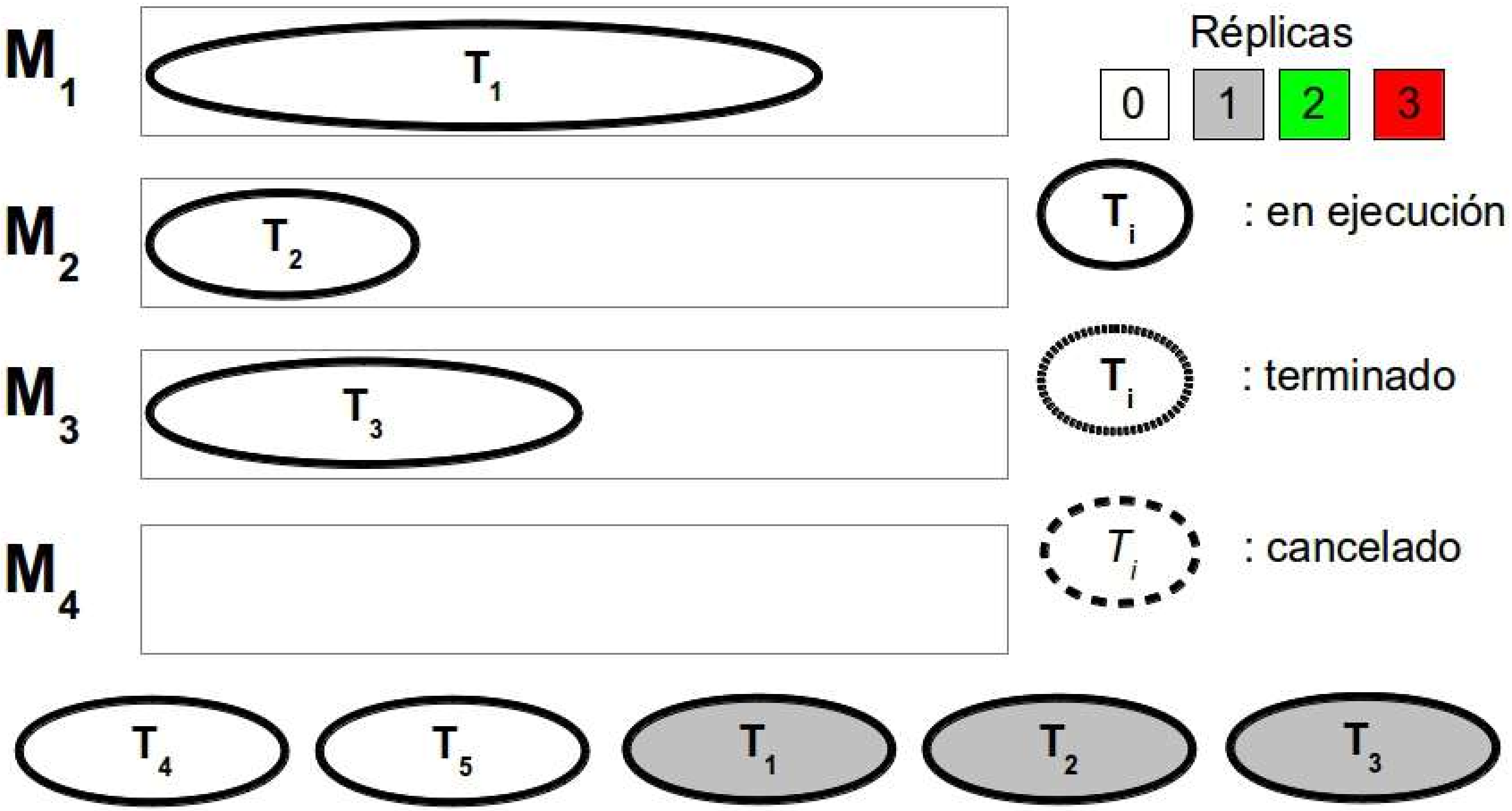
Work Queue with Replication (WQR)



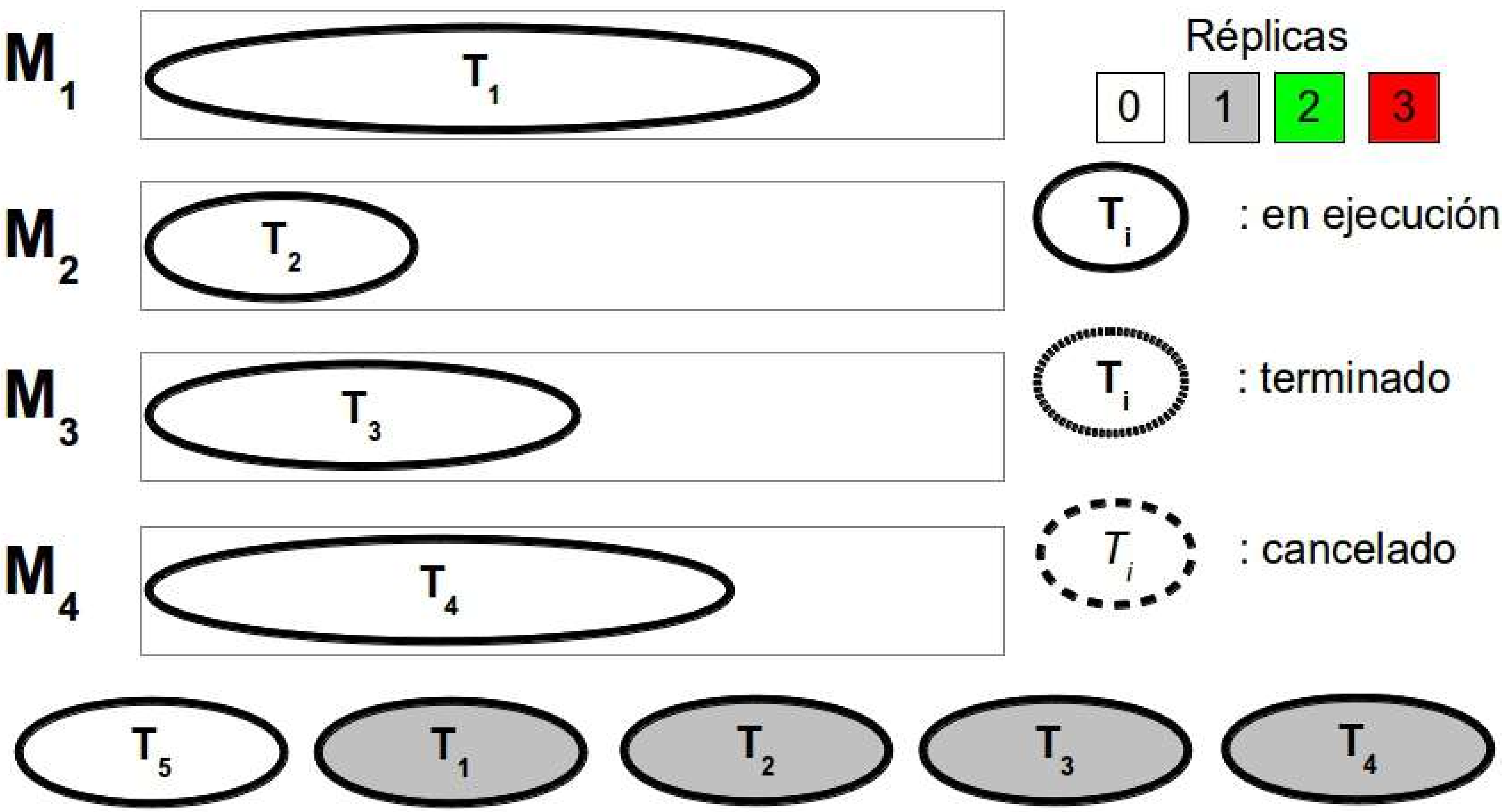
Work Queue with Replication (WQR)



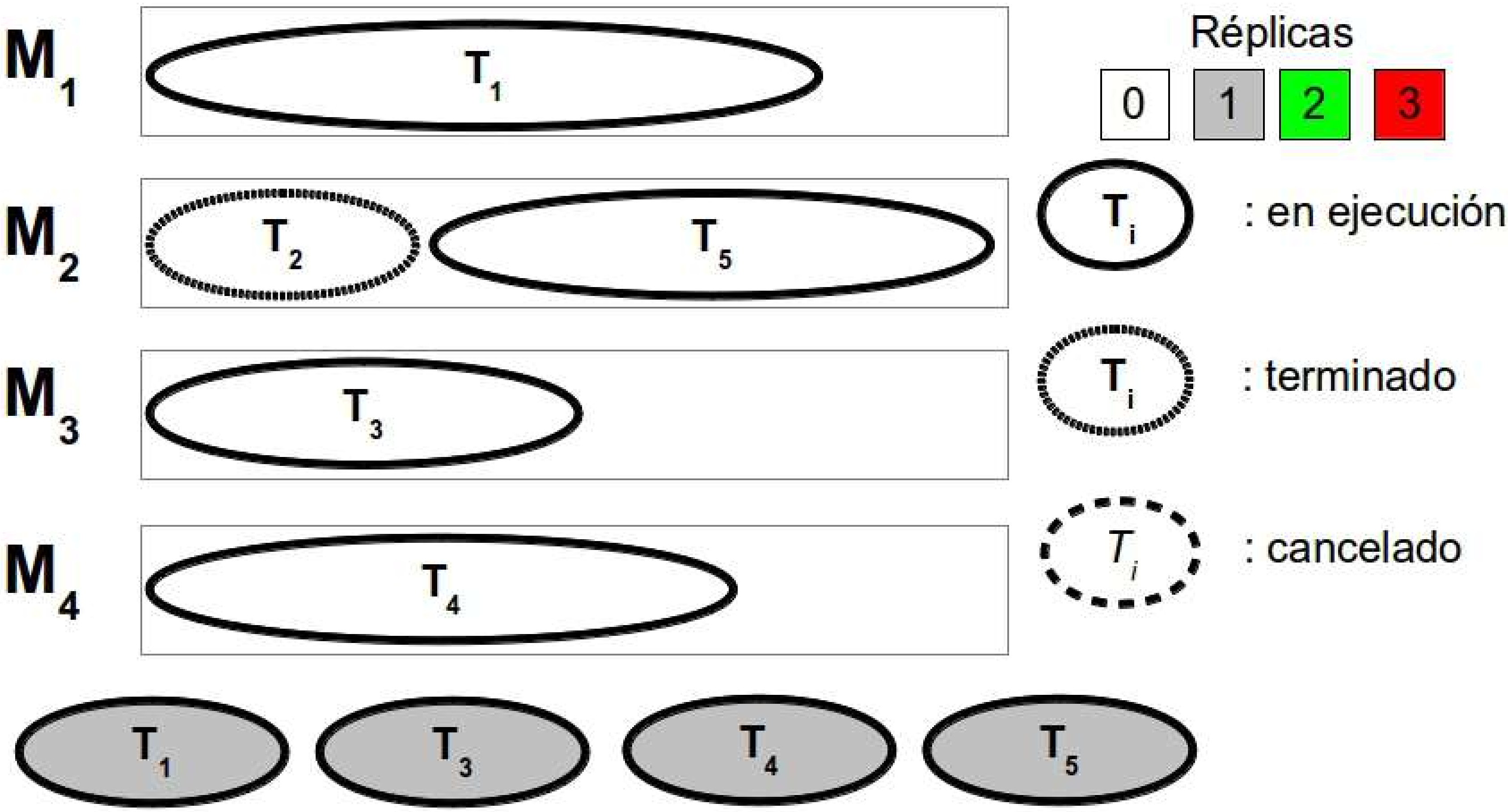
Work Queue with Replication (WQR)



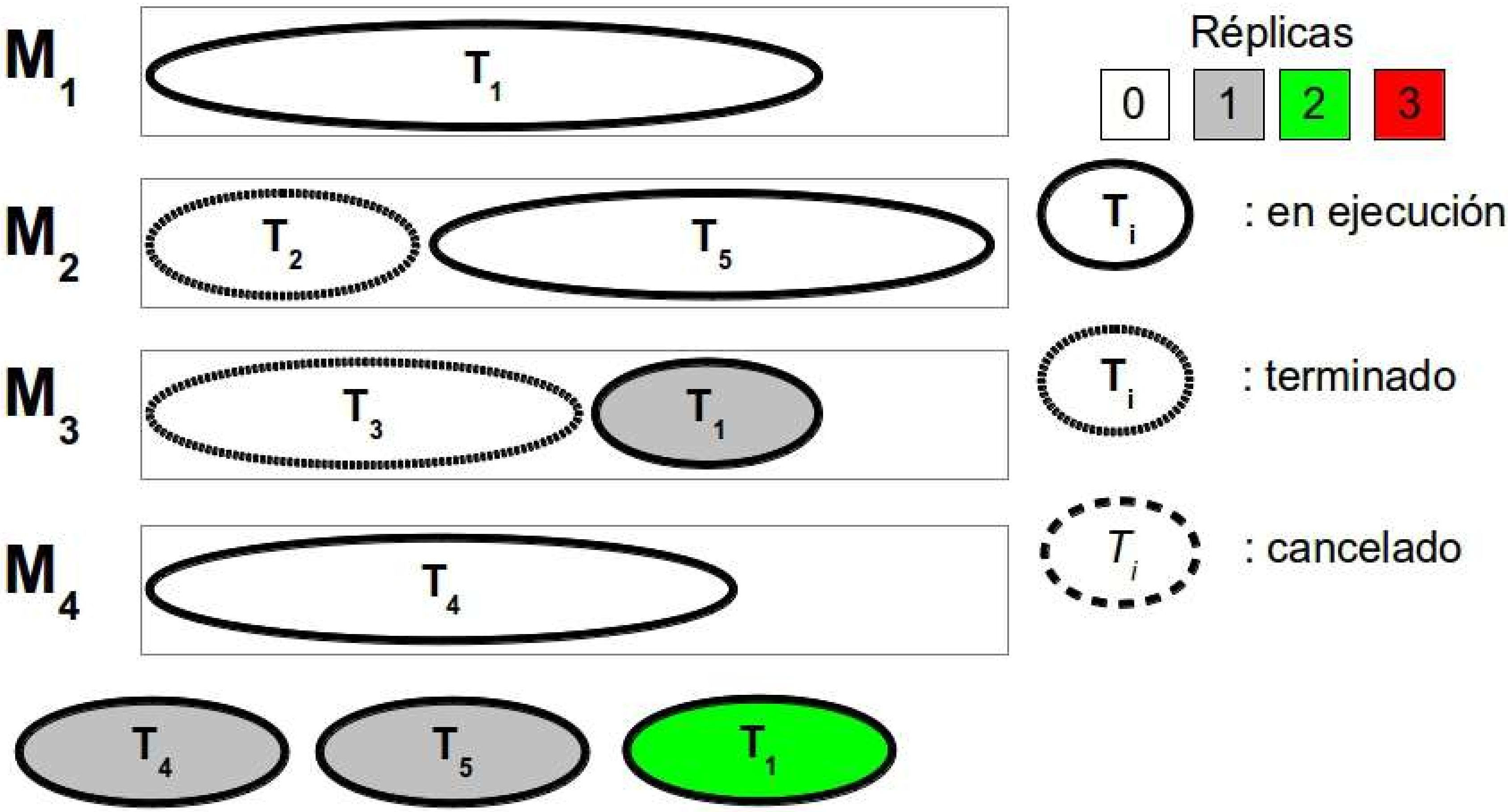
Work Queue with Replication (WQR)



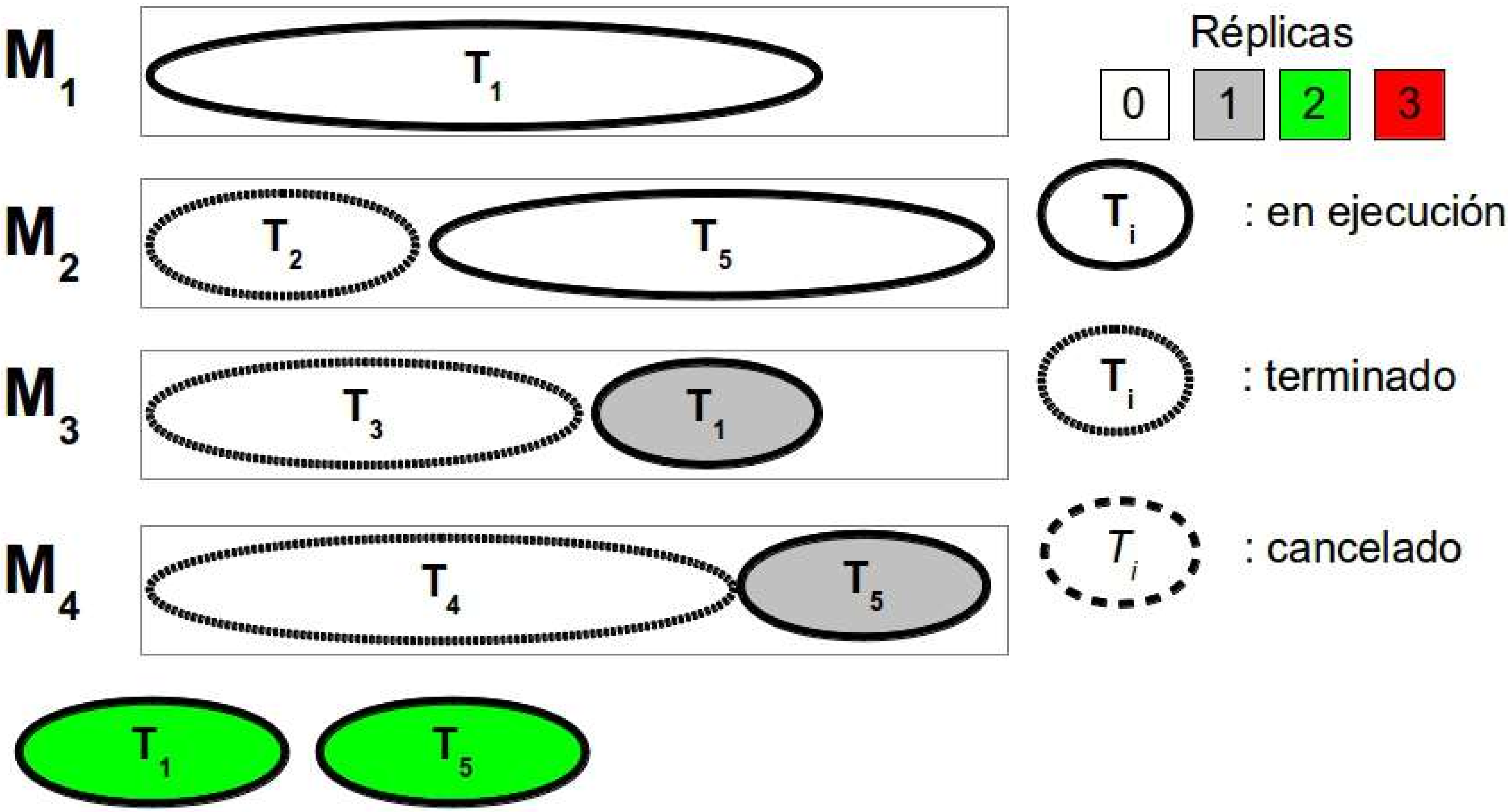
Work Queue with Replication (WQR)



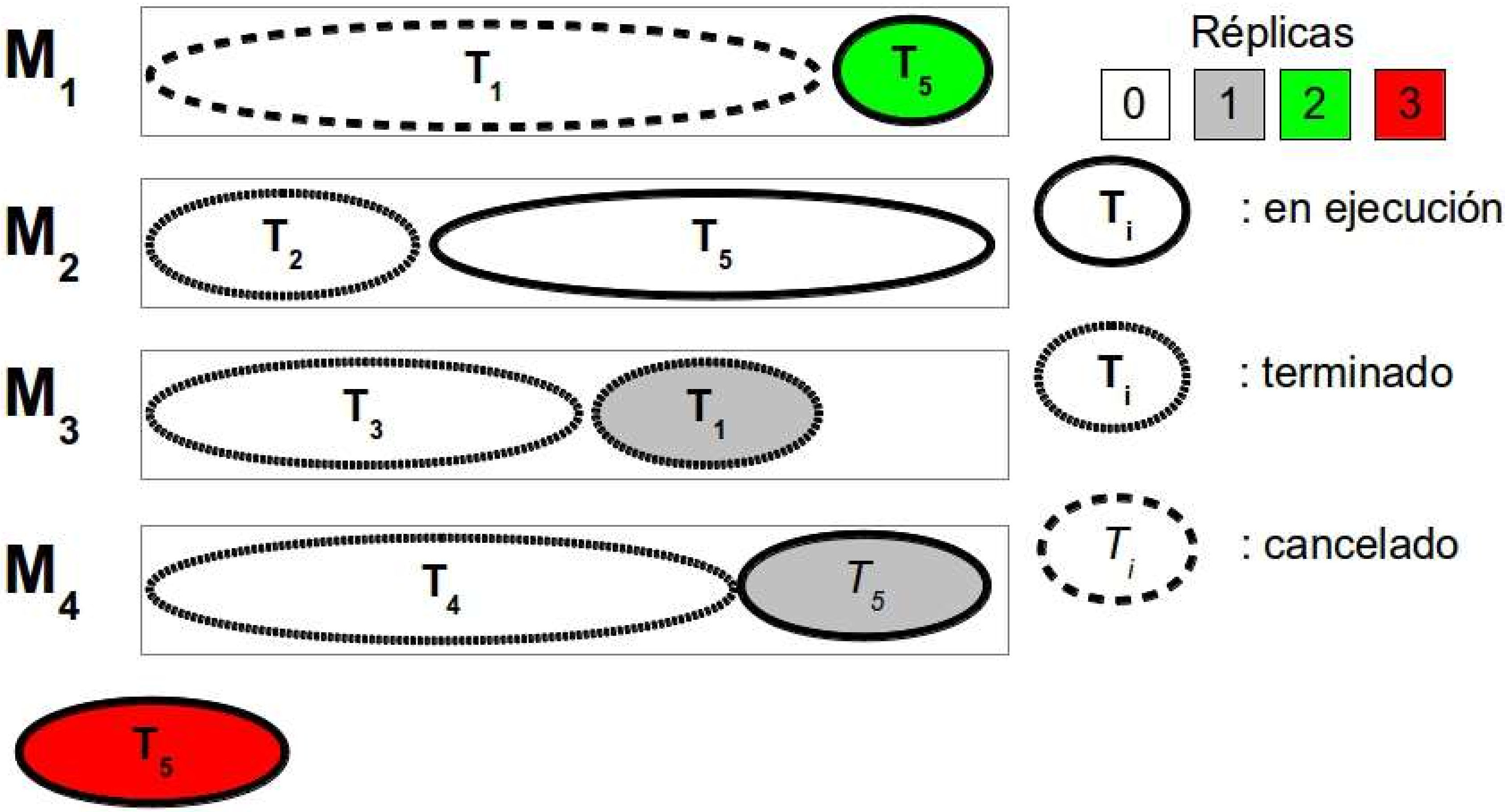
Work Queue with Replication (WQR)



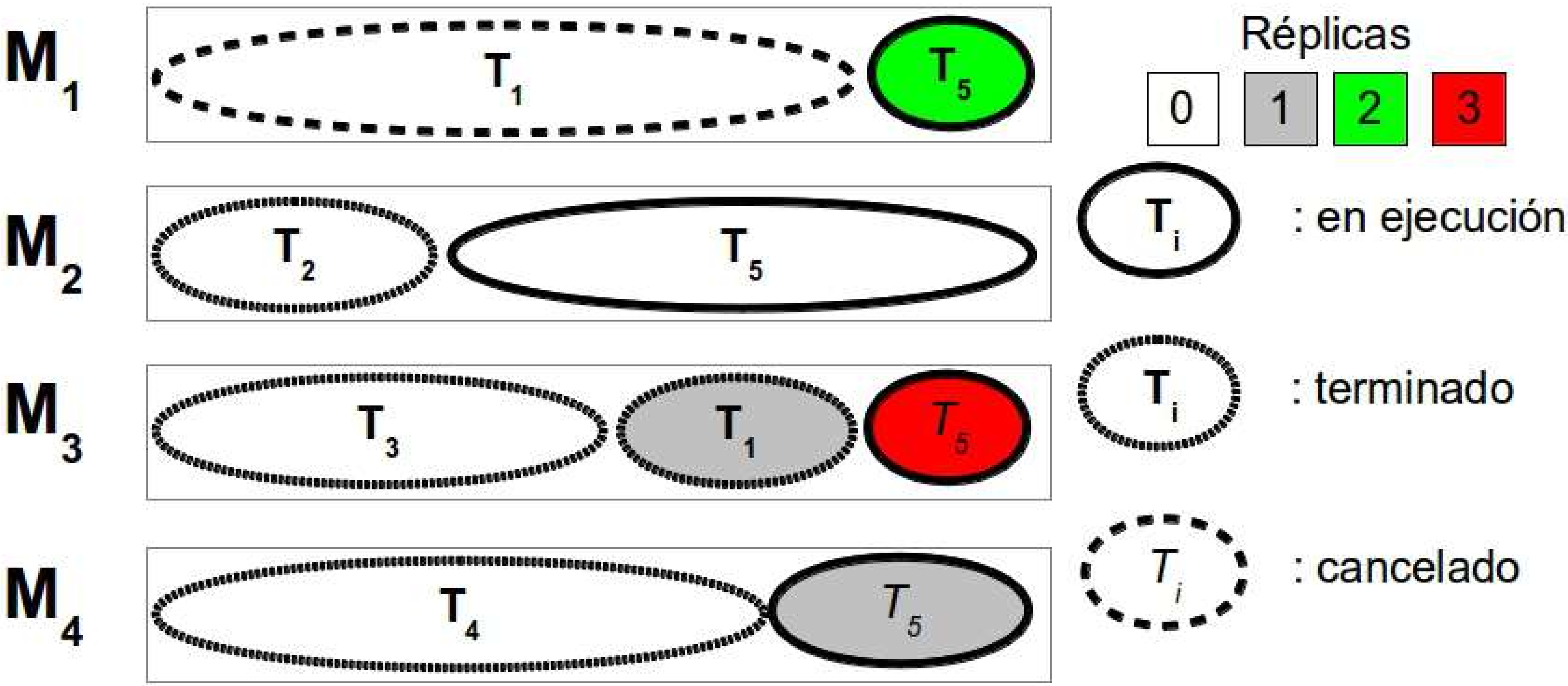
Work Queue with Replication (WQR)



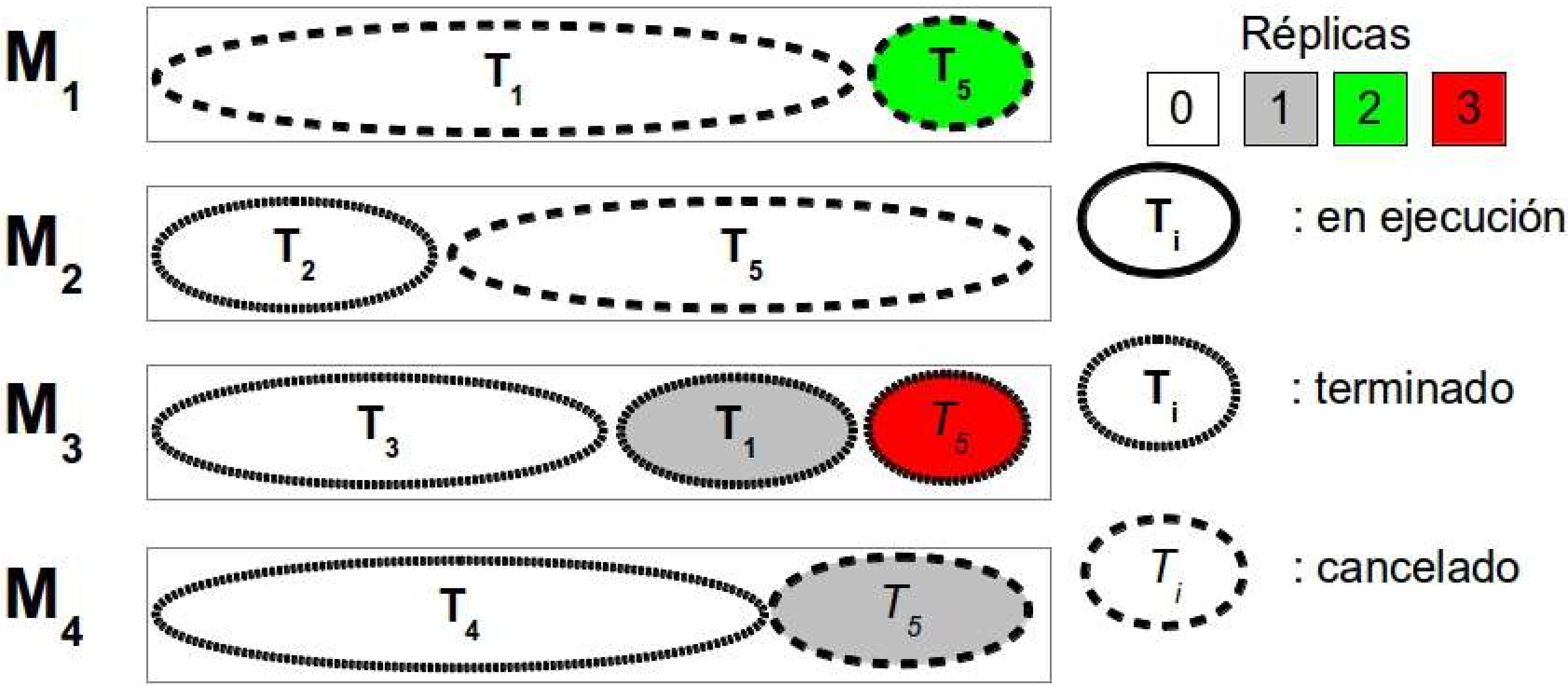
Work Queue with Replication (WQR)



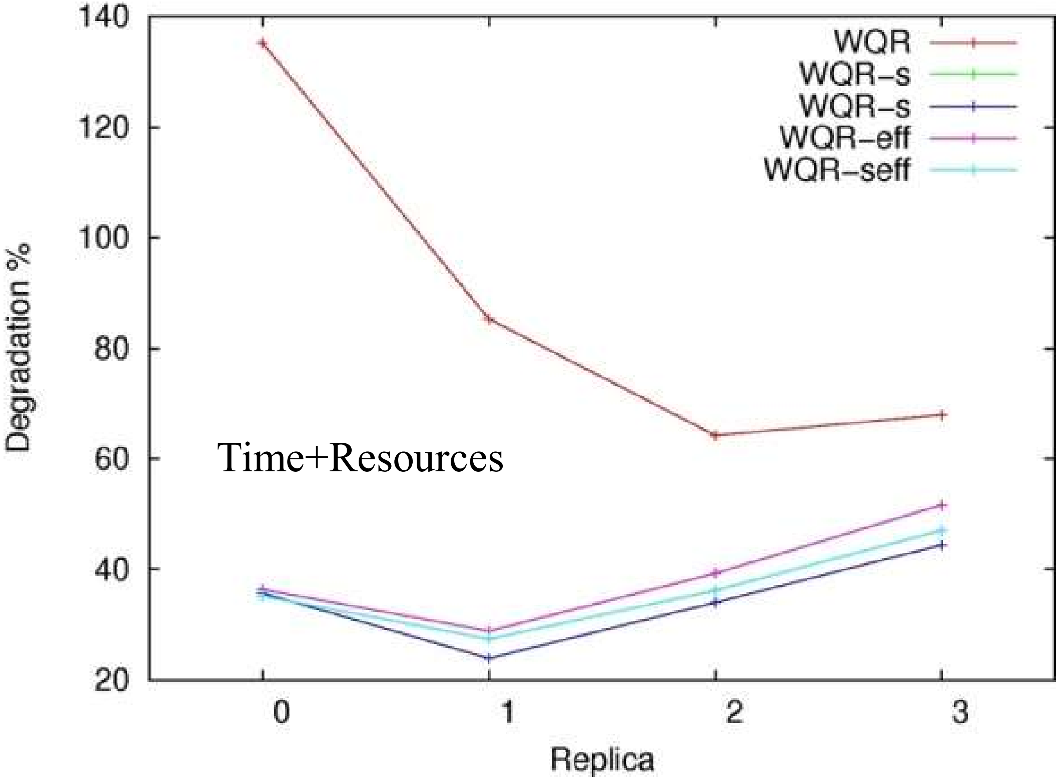
Work Queue with Replication (WQR)

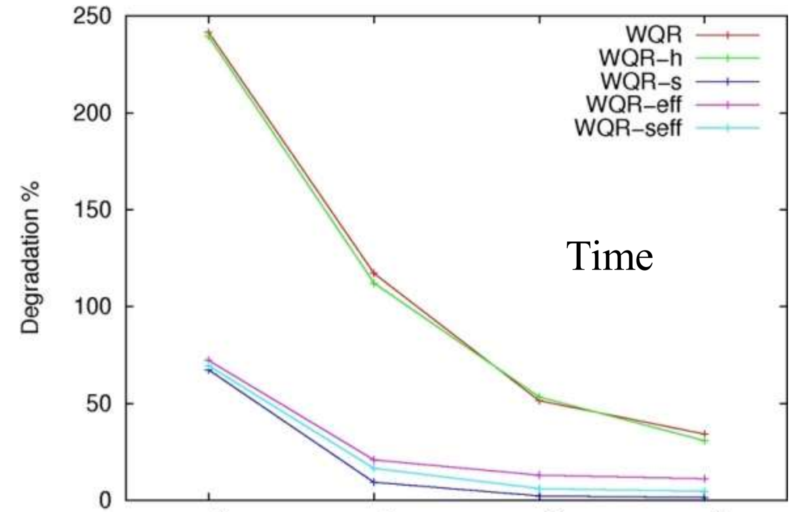


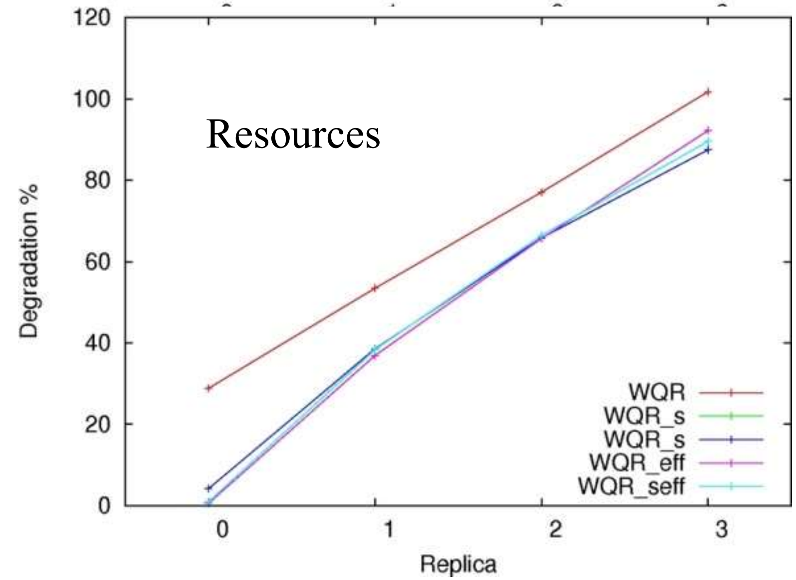
Work Queue with Replication (WQR)



Work Queue with Replication (WQR)







CICESE

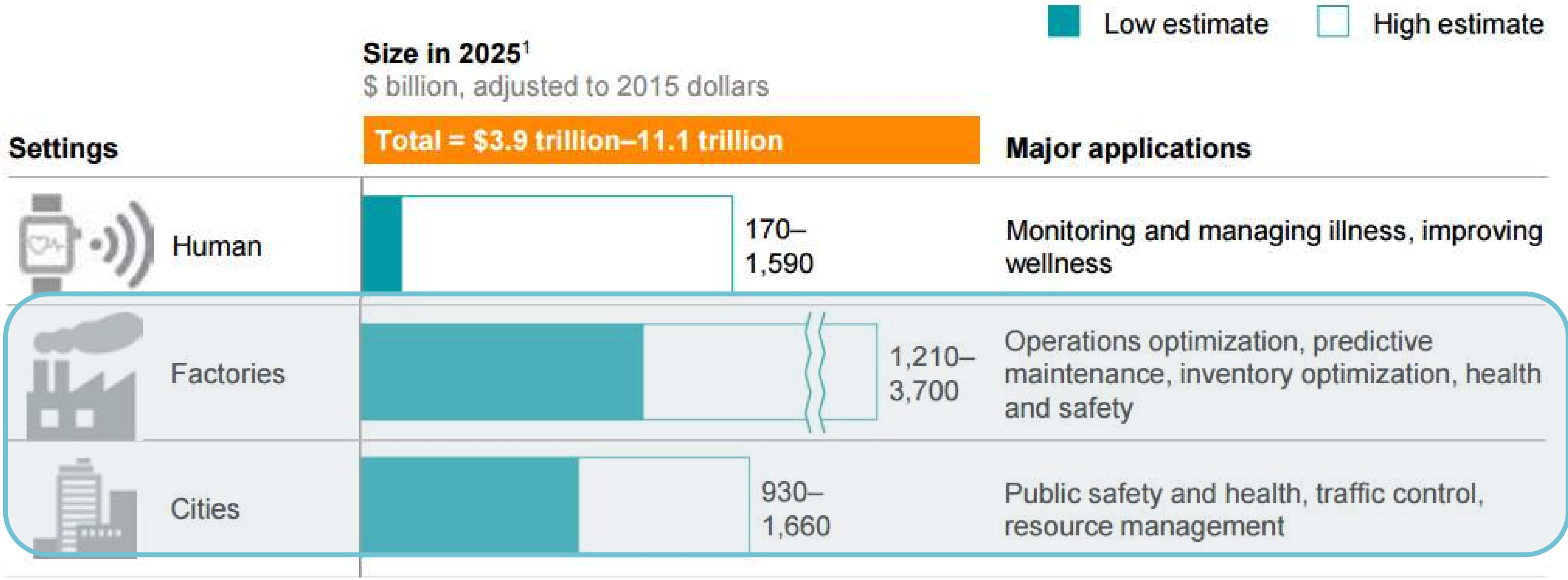
Smart
Anything
Everywhere

Integrating technologies and data
to meet
current challenges and service innovation

- simulation,
- modelling;
- data-analytics;
- advanced smart sensors;
- cyber-physical systems
- Internet of Things (IoT).

Horizon 2020
Programme

Smart Things



- Fundamentally new approaches to digital design based on complete mathematical modeling and optimization technologies;
- Virtual tests, which significantly reduce the amount of expensive field tests;
- Advanced technologies and digital smart production

THE INTERNET OF THINGS: MAPPING THE VALUE BEYOND THE HYPE // McKinsey Global Institute (MGI), 2015.

Smart Everything

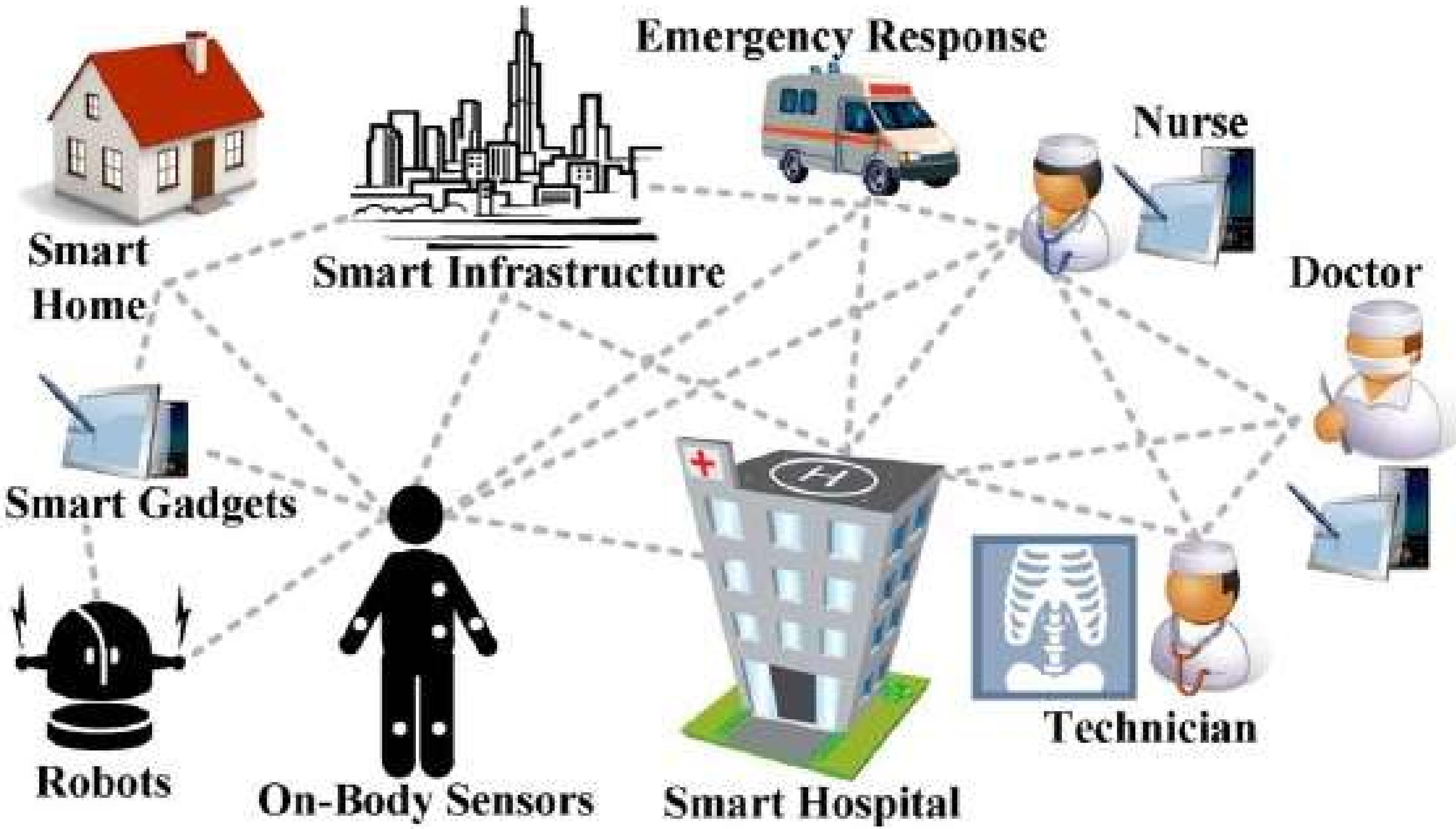
- Smart Industry, Factories of the Future, Industry 4.0
- Smart City
- Smart Home
- Smart Service
- Smart Healthcare
- Smart Economy
- Smart Networking
- Smart Analytics
- Smart Security and Privacy
- Smart autonomous driving
- Smart Oil and Gas Industry
- etc.

Smart Home

Smart Home/Business Gateway Platform

Lowers barrier to convergent smart technical and economic IoT innovation

Smart Healthcare



Smart Industry

Technological evolution

- from embedded systems to cyber-physical systems

Merging of the virtual and physical worlds

- through cyber-physical systems

Fusion of

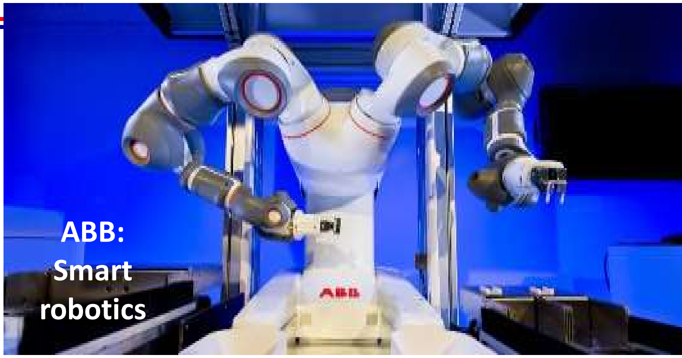
- technical processes and business processes

"Industrial Internet of Things" (IIOT) - driving operational efficiencies through

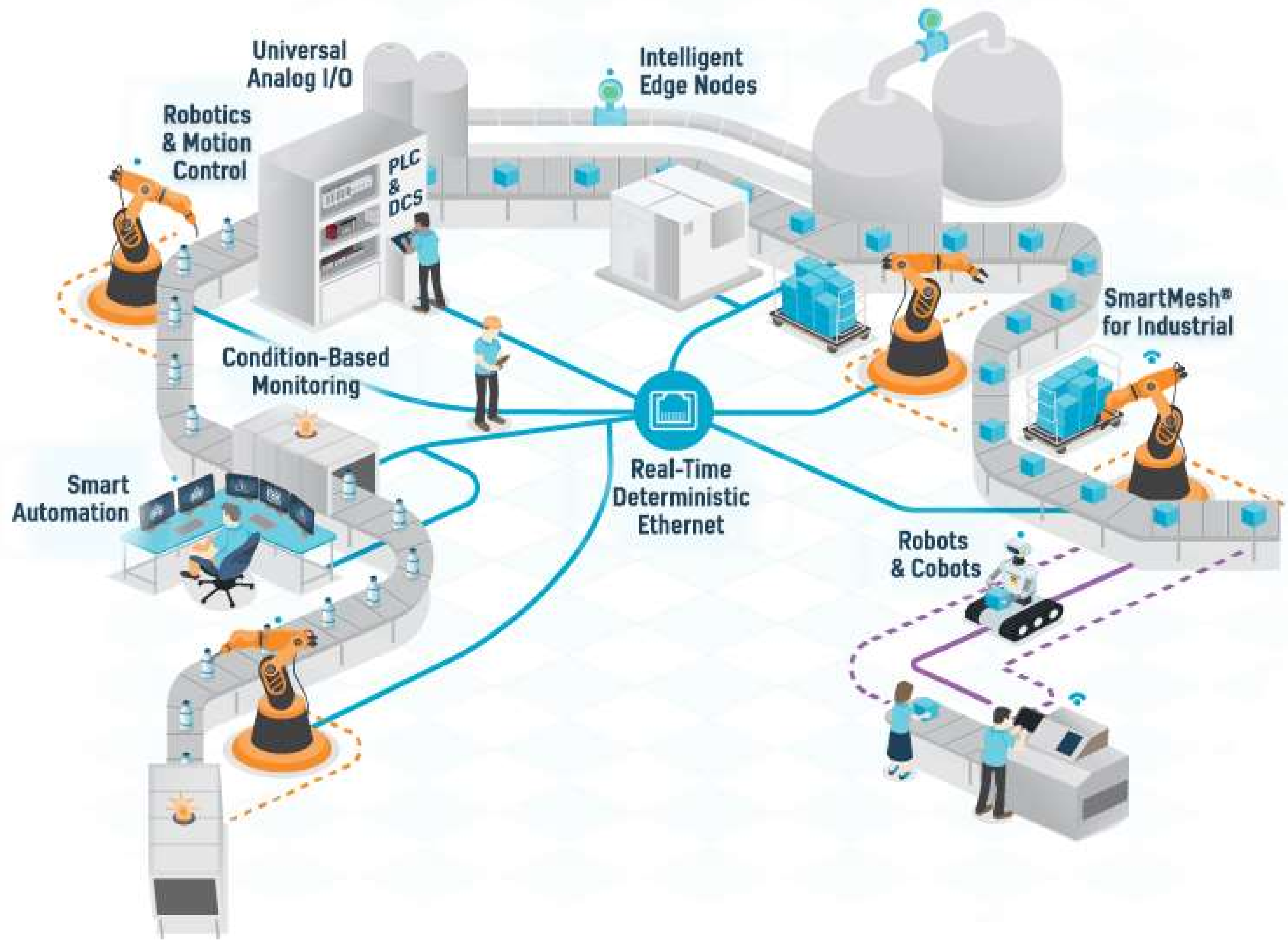
- Automation
- Connectivity
- Analytics

Intellectual sensors → models → Digital twins

operational security – data security

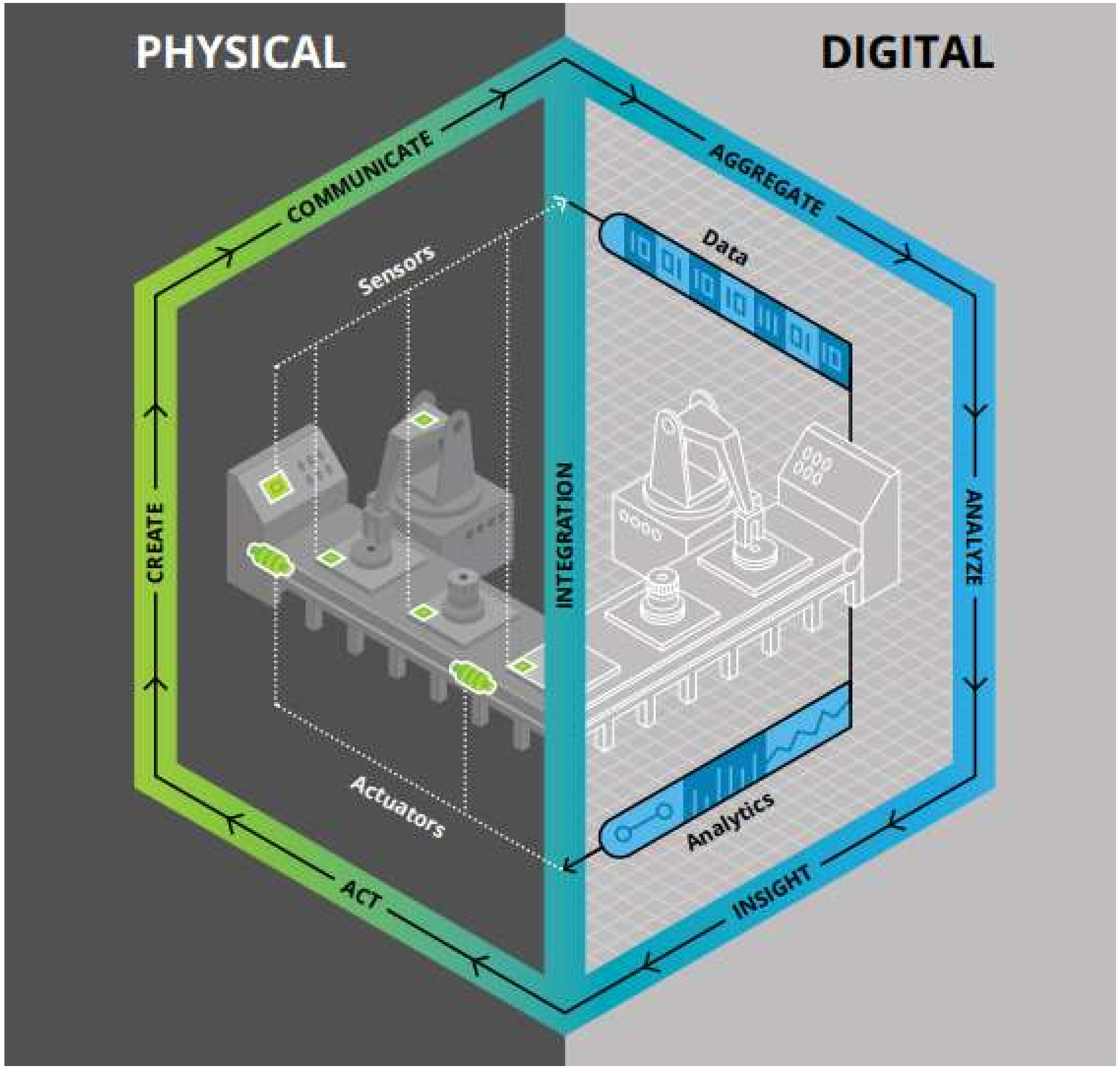


Smart Factory



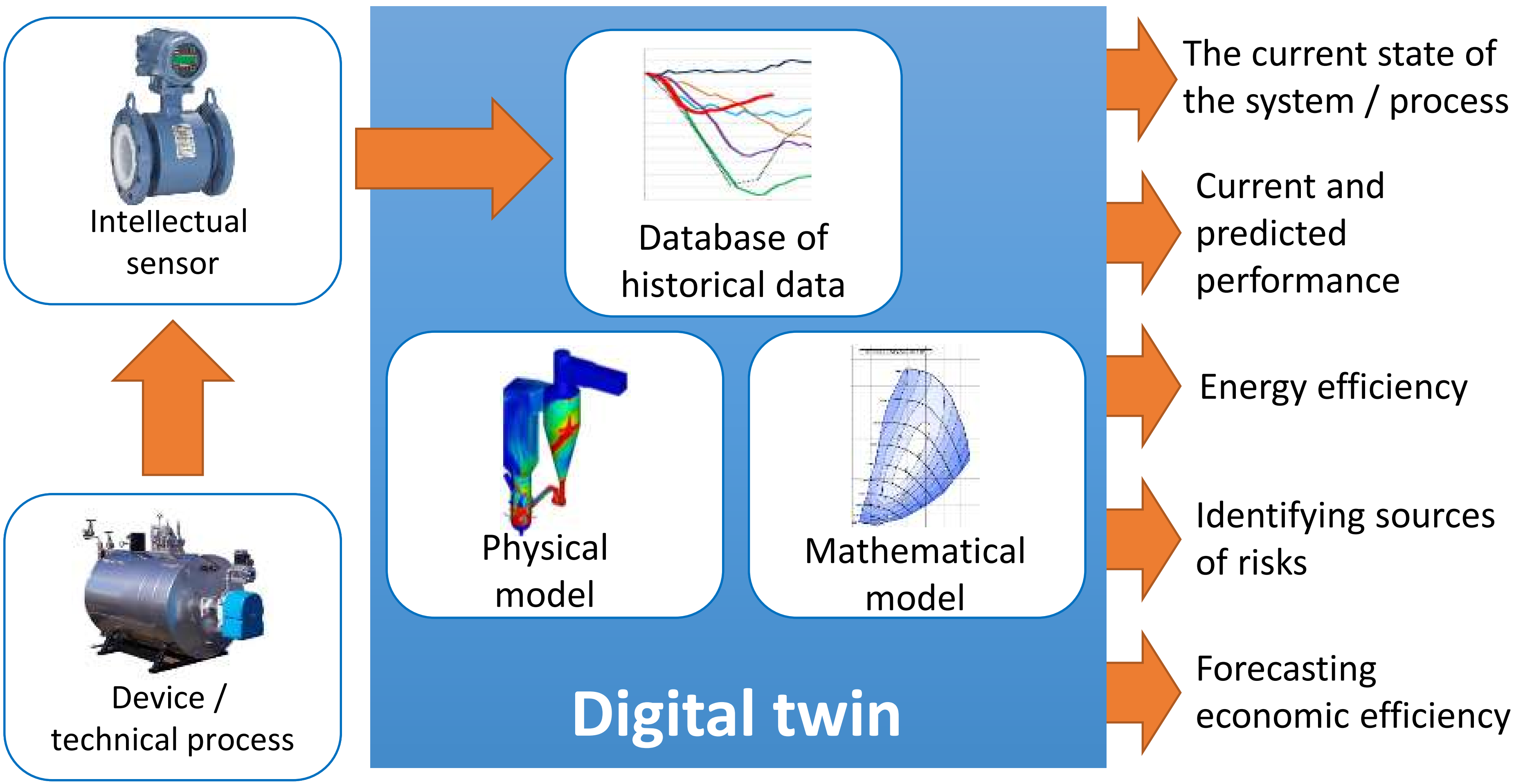
Tomado de https://www.analog.com/en/applications/markets/industrial-automation-technology-pavilion-home/industry-4-pt-0.html

Digital Twin



[Industry 4.0 and the digital twin. Deloitte University Press]

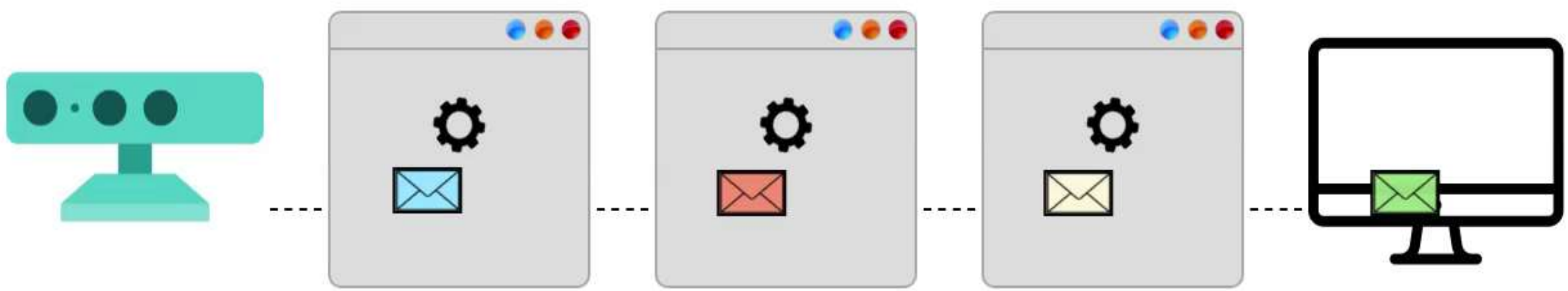
Digital twins



Main objectives

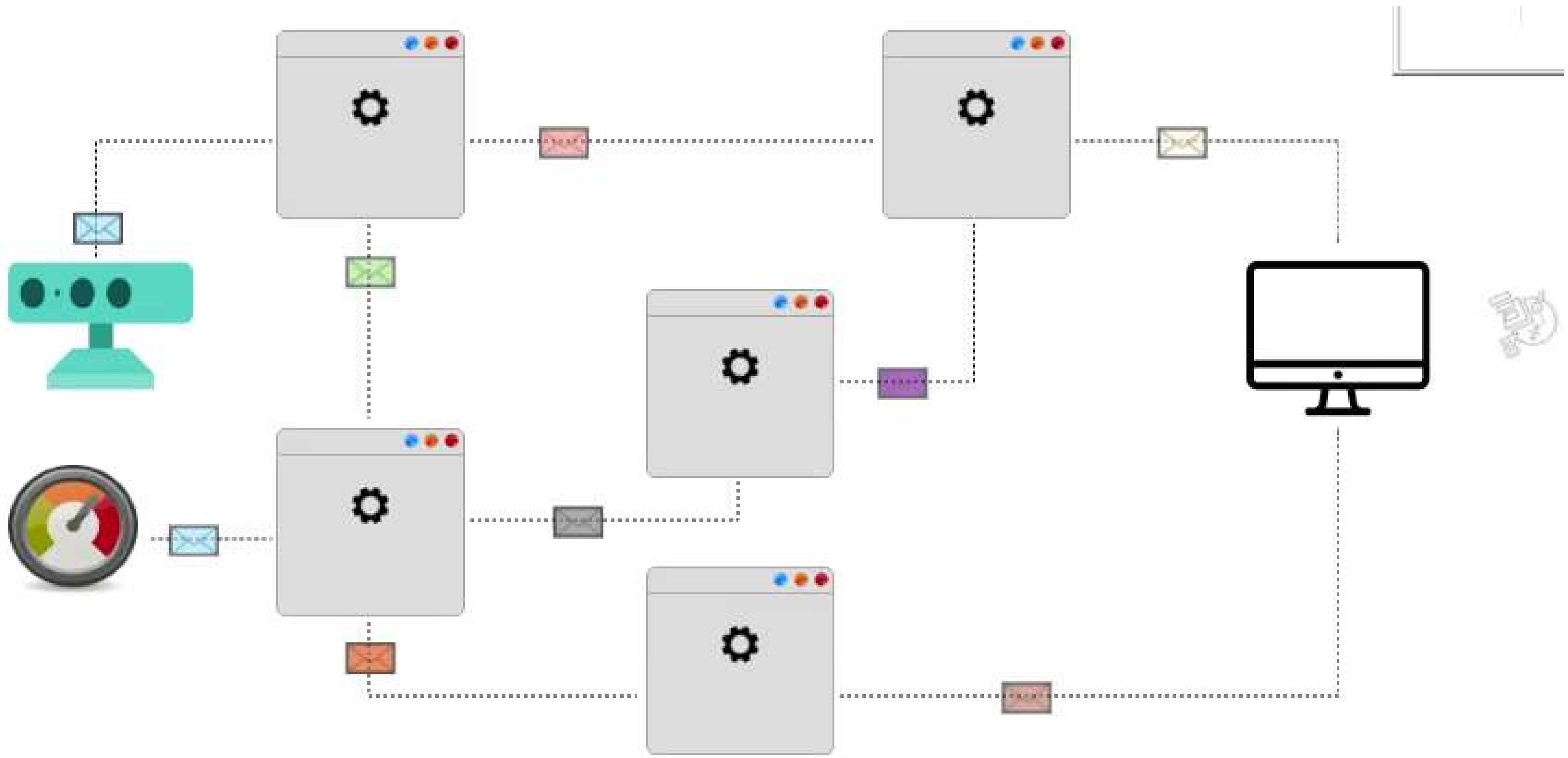
1. **Prediction** and prevention of emergency situations and ensuring economic sustainability
2. **Accumulation** and effective processing of technological knowledge
3. **Transition** from line production to customizable one
4. **Internetization** of manufacturing
5. **Education** on process management using digital simulators and augmented reality

Pipeline



Animation Link

Workflow

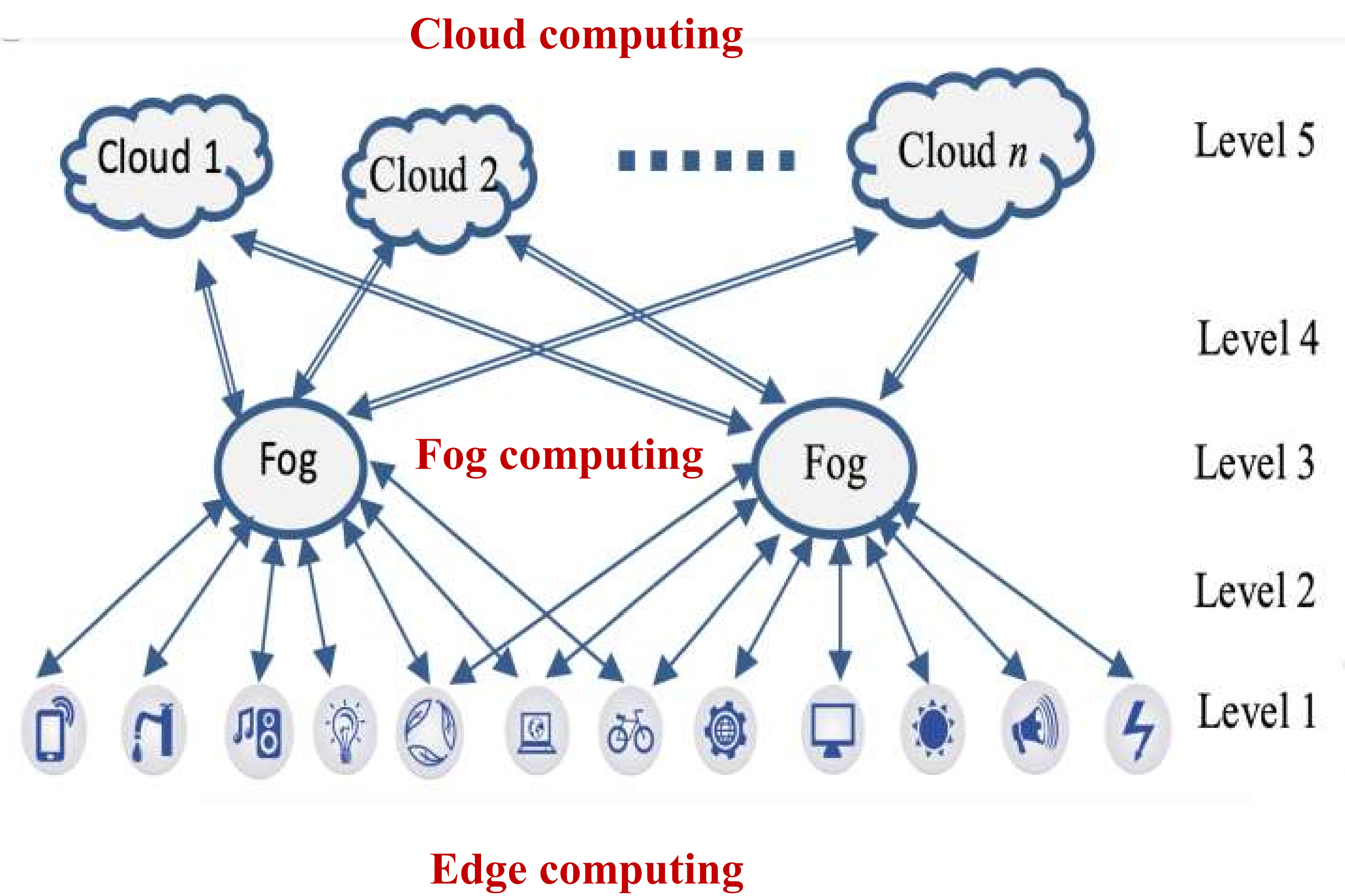


Animation Link

Platform for Connected Smart Objects

Internet of Things

Integrates sensing, communications, and analytics



Analytics

Storage

Computational models

Remote services

Data mining

Remote monitoring

Intelligent sensors

Smart City

More than half the population (54%) are located in urban areas, as oppose to the 30% in 1950. It's expected an estimated increase at 66% of the world population living in cities in 2050 [United Nations 2014].

Cities are 2% of earth surface but 75% of energy consumption

100+ new cities of 1 million+ people in next 10 years

Smart Mobility

- ❑ Improving the personal mobility, comfort, connivance, and safety.
- ❑ Increasing economic productivity for transport service providers.
- ❑ Enhancing efficiency and capacity.
- ❑ Reducing gas consumption and negative environmental impact.

Smart Transport

Three solutions: what to select

Minimize

$$f_1 = \sum_{i=1}^{n} \omega_i,$$

$$f_2 = \sum_{s \in R} LQ_s.$$

subject to:

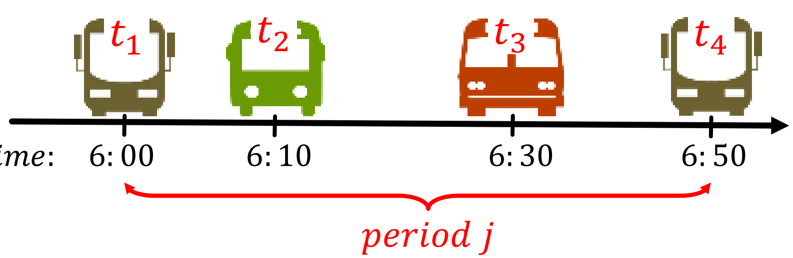
$$c_i = c_i^{bus} + c_i^{gas} + c_i^{driver},$$

$$\omega_i = c_i m_i,$$

$$f_j \geq f_{min} ,$$

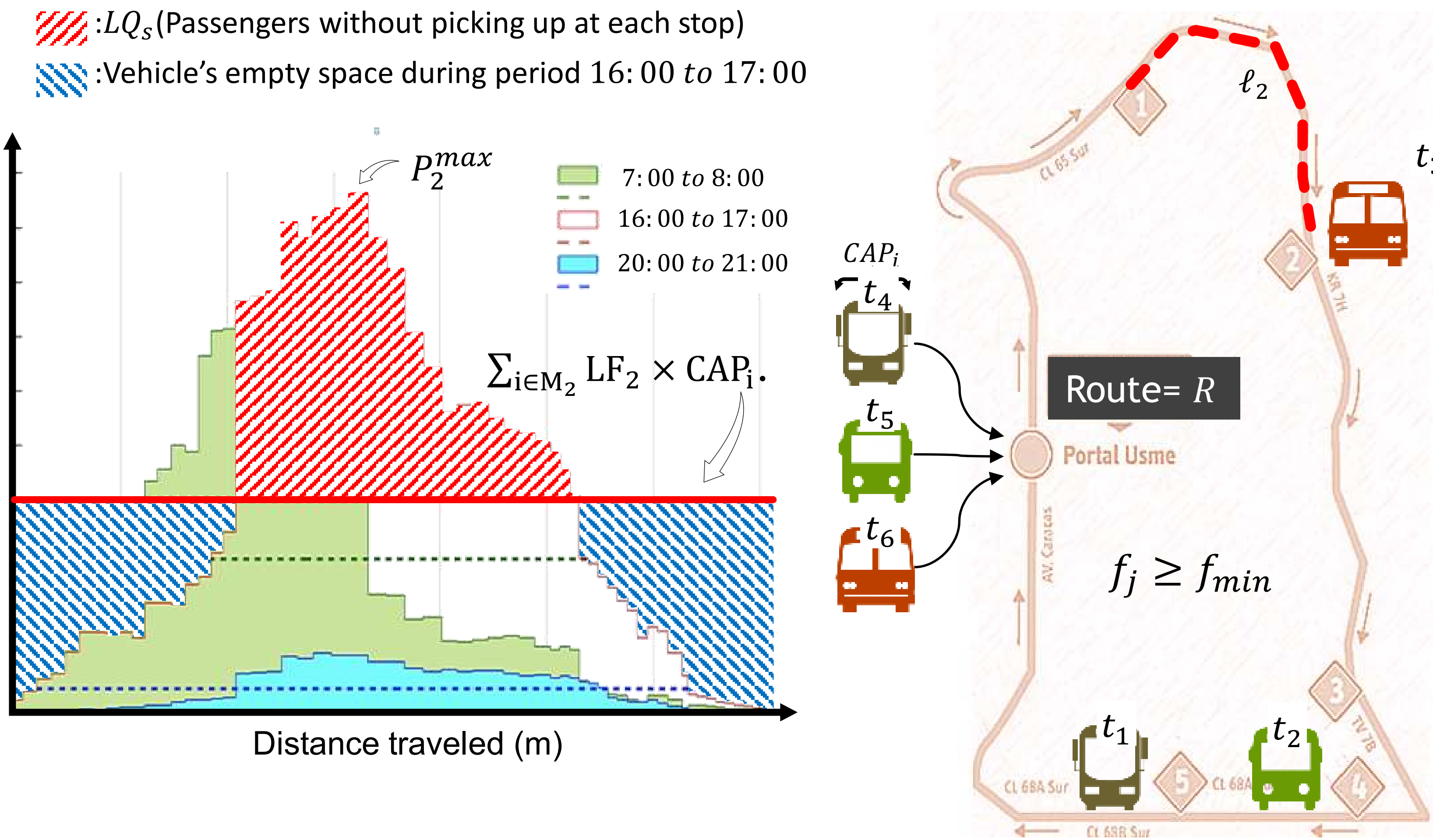
$$\mathrm{LF_j} = \frac{P_j^{max}}{\mathrm{CAP_i} \times \mathrm{f_j}} \leq \mathrm{LF_{max}},$$

$$\mathrm{LQ_s} = \max\left(P_j^s - \sum_{\mathrm{i} \in \mathrm{M_j}} \mathrm{LF_j} \times \mathrm{CAP_i} , 0\right).$$



c_i^{gas}	: Fuel cost for each vehicle.
c_i^{driver}	: Hourly pay cost of the bus driver.
c_i^{bus}	: Vehicle maintenance and operation cost.
c_i	: Cost involved in using a vehicle of type i.
m_i	: Number of vehicles required of type i to service all trips in T.
ω_i	: Total cost involved in using m_i vehicles of type i
P_j^{max}	: Maximum number of passengers at any stop.
P_j^s	: Number of passengers on a s stop during period j.
f_j	: Frequency for period j.
f_{min}	: Minimum required frequency.
LF_j	: Load factor during period j.
LF_{max}	: Maximum load factor.
LQ_s	: Passengers demand at the s stop that exceed the vehicles capacity.
M_j	: Set of vehicles used during the period j
CAP_i	: Capacity of a vehicle of type i.

Unsatisfied demand (f_2)

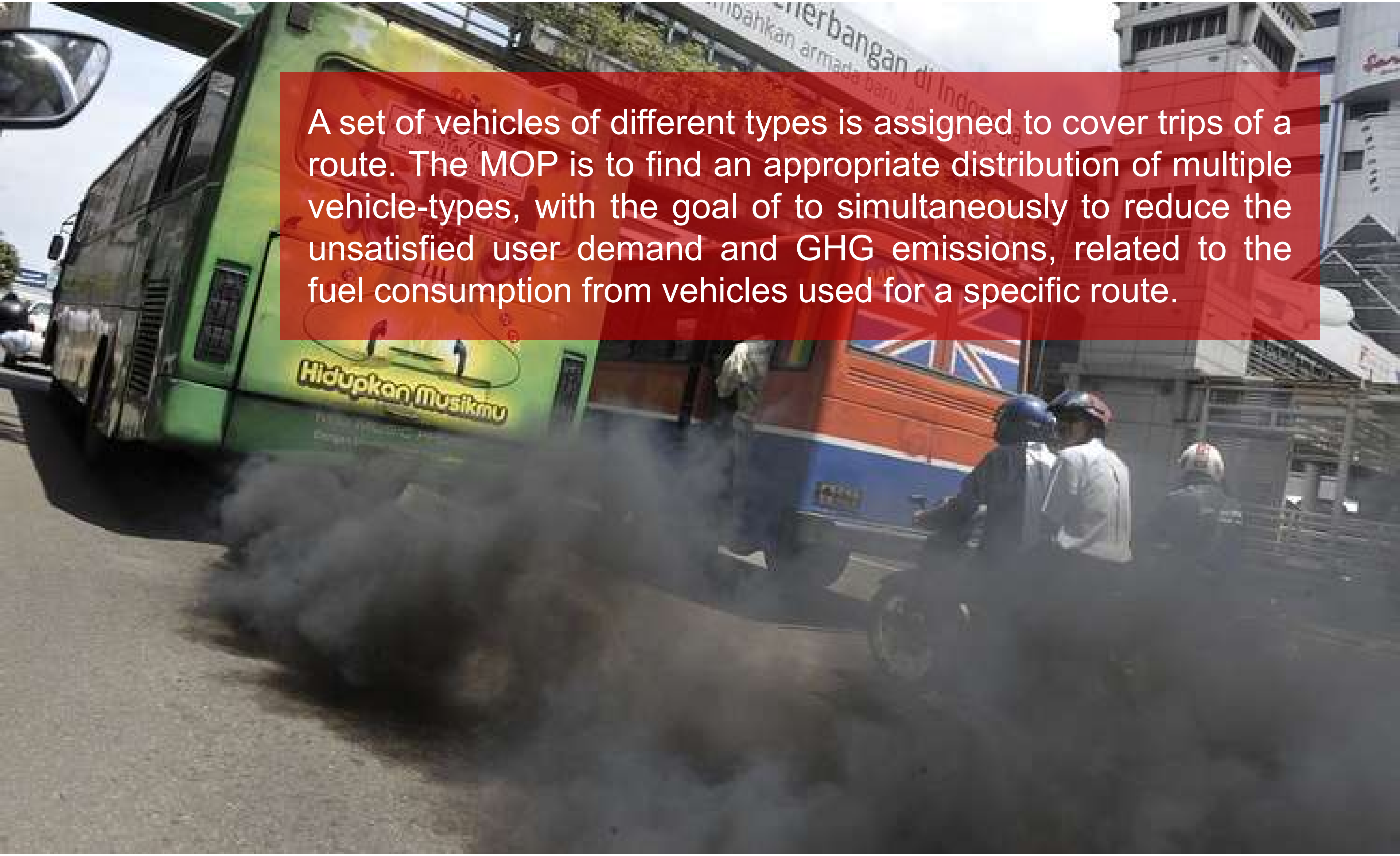


f_2 defines the number of passengers that cannot be moved satisfactorily, which implies more waiting time and overload in the selected vehicles to cover the route in this period.

Uncertainty

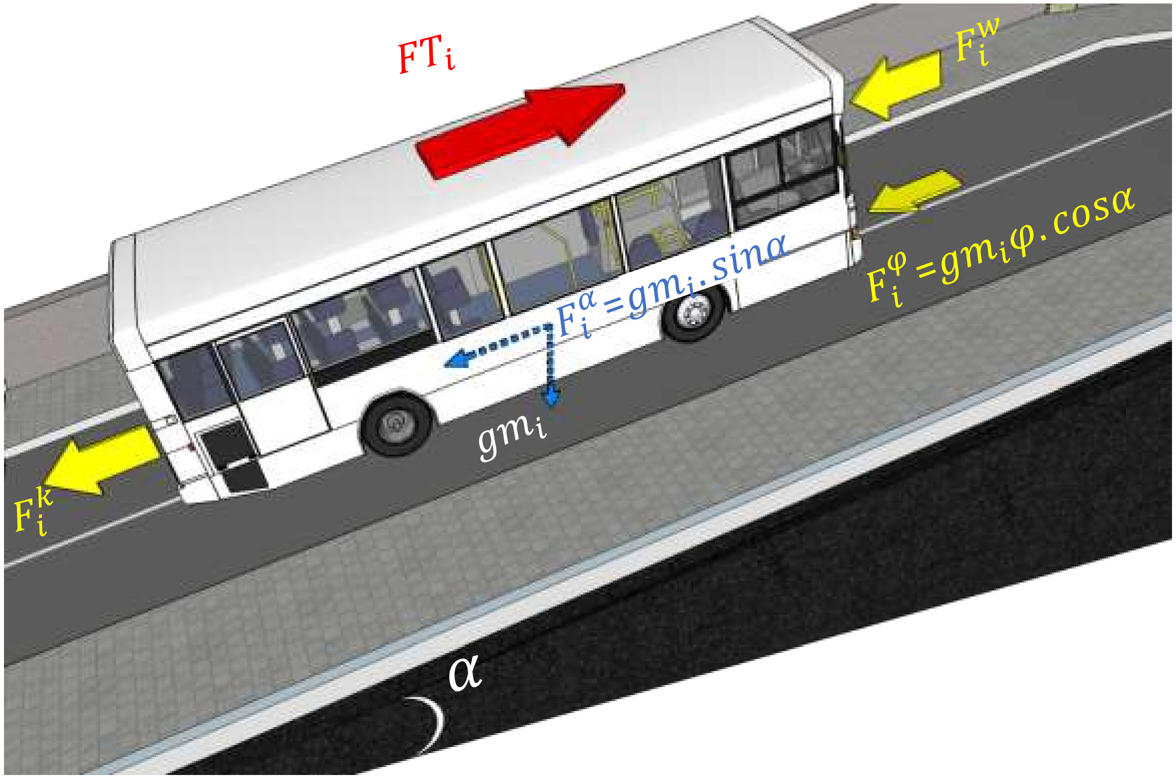
- **Communication failure**
- **Break-down of a vehicle**
- **Failures in the transport network**
- **Passenger demand**
- **Weather changes**
- **Modification of the transportation requests**

Environmental protection

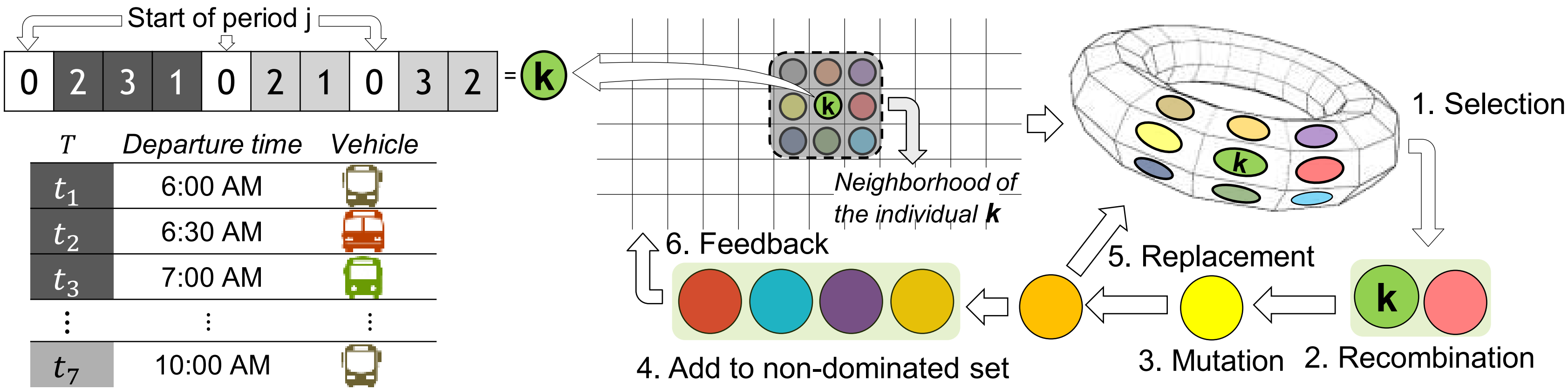


A set of vehicles of different types is assigned to cover trips of a route. The MOP is to find an appropriate distribution of multiple vehicle-types, with the goal of to simultaneously to reduce the unsatisfied user demand and GHG emissions, related to the fuel consumption from vehicles used for a specific route.

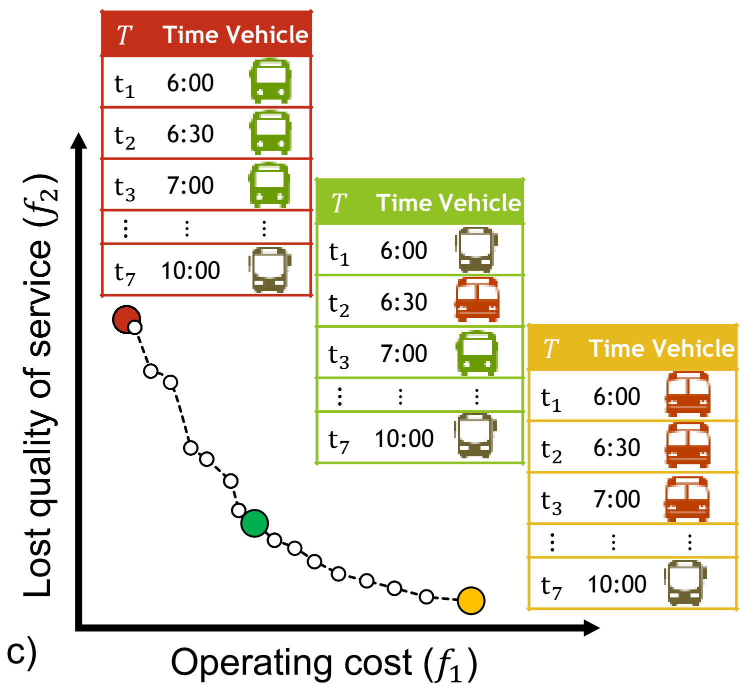
Quality of service, cost, pollution



Multiobjective cGAs (MOcell)



a) b)



c)

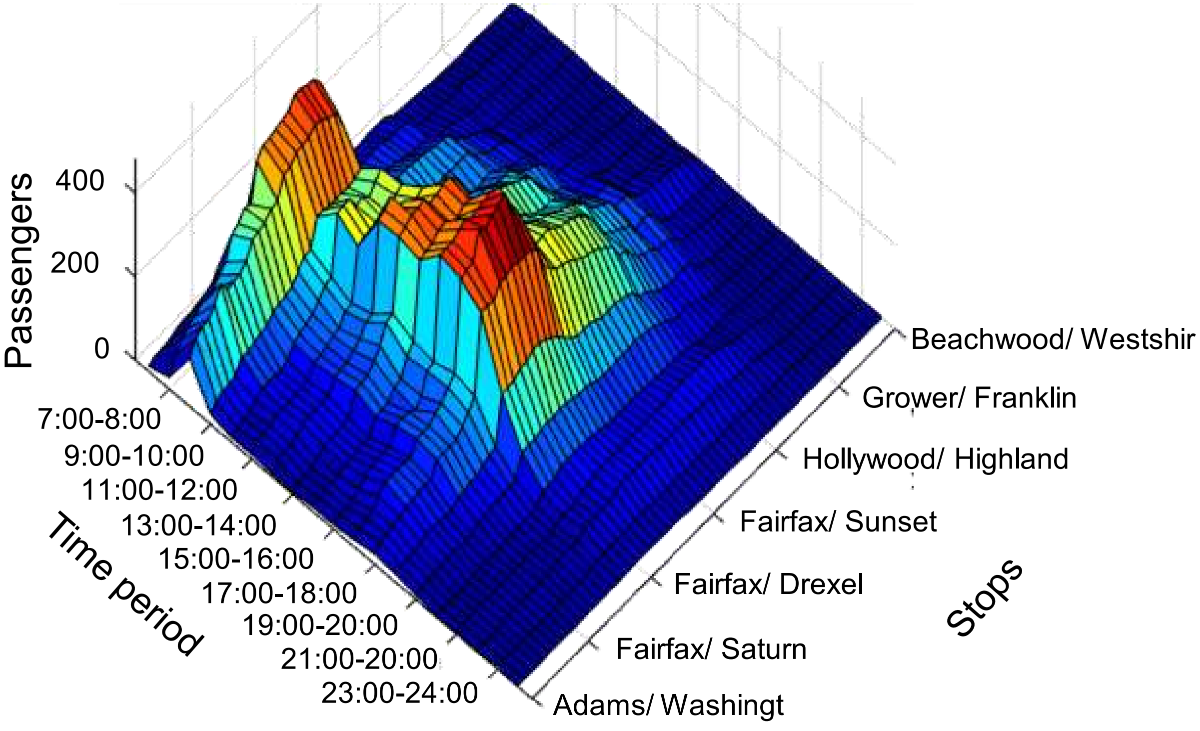
a) Example of solution representation (chromosome).

b) Reproduction steps in asynchronous cGAs.

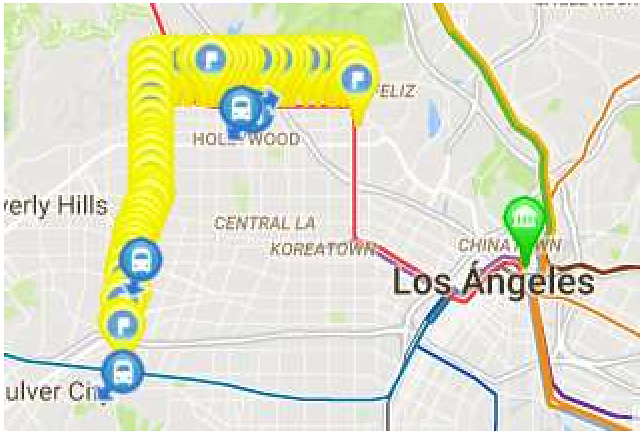
c) Timetables obtained by selecting different solutions of the Pareto front approximation for one execution of the proposed algorithm.

Experiments design

Route 217 Metro Local Line – Los Angeles, California. (a) Passenger demand, ride-check data for 19 time-periods of one hour and 59 stops, maximum load 481 in Fairfax/Rosewood between 17:00 to 18:00 (peak hour). b) Route map with its stops (Rideschedules, 2017)



a)

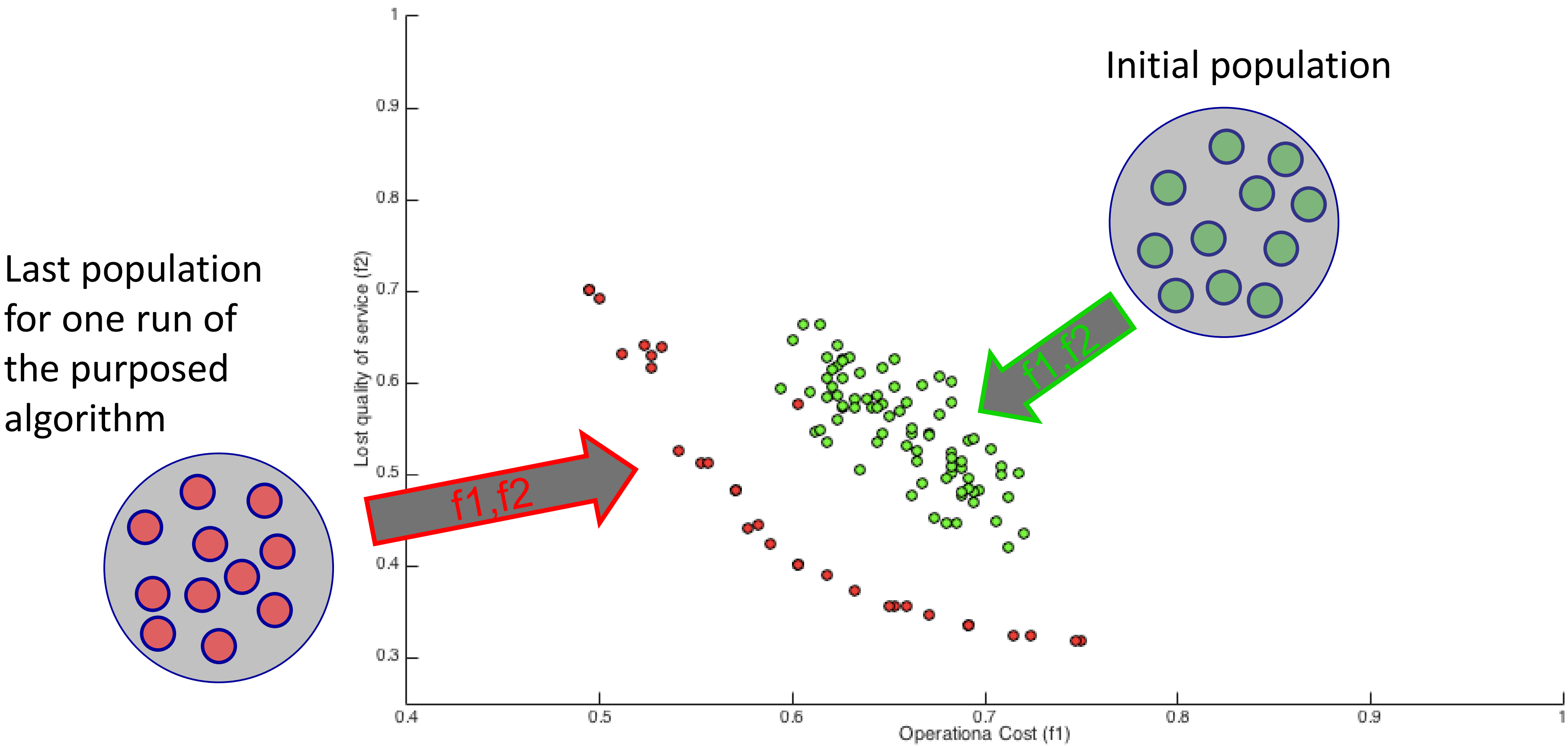


b)

RideSchedules.com, Official LA Metro Bus Data, Updated: May 2, 2017, viewed May 14, 2017. <https://rideschedules.com/schedule.html?23467>.

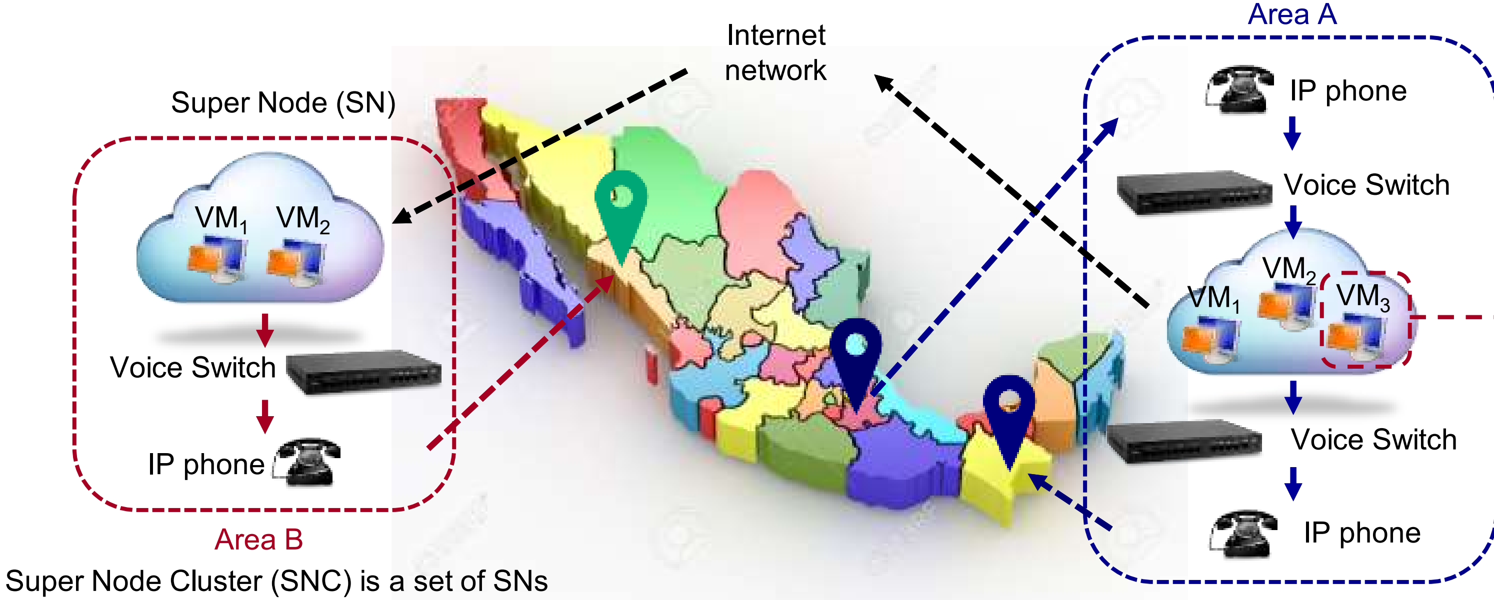
Results

The main objective of multiobjective optimization algorithms is to obtain an approximation of the true Pareto front of a given MOP. In general, MOPs can have a Pareto front composed by a huge (possibly infinite) number of solutions.



A VoIP Service for Cloud Infrastructure

Cloud Voice over IP



Advantages

- Granularity of hardware
- Scalability
- Cost
- Geographic distribution
- Robustness of the solution

Disadvantages

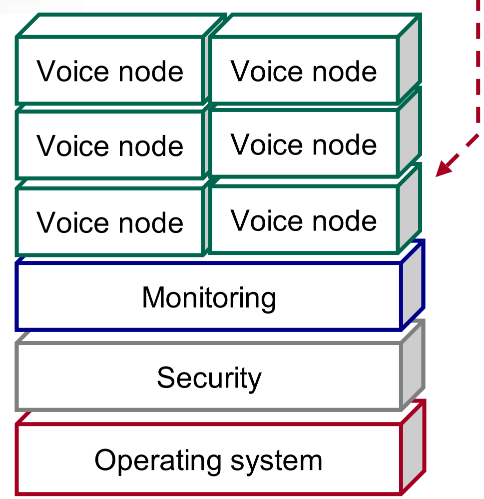
- Call quality reduction
- Load imbalance

Telephone system features: voice mail, call transfer, music on hold, conference function, etc.

Monitor the use of the resources

Access to the server such as SSH, FTP sessions

Execute VoIP software (Asterisk)



Problem

Two objectives:

- Provider cost optimization
- Voice Quality

Bin-packing approach (well-known)

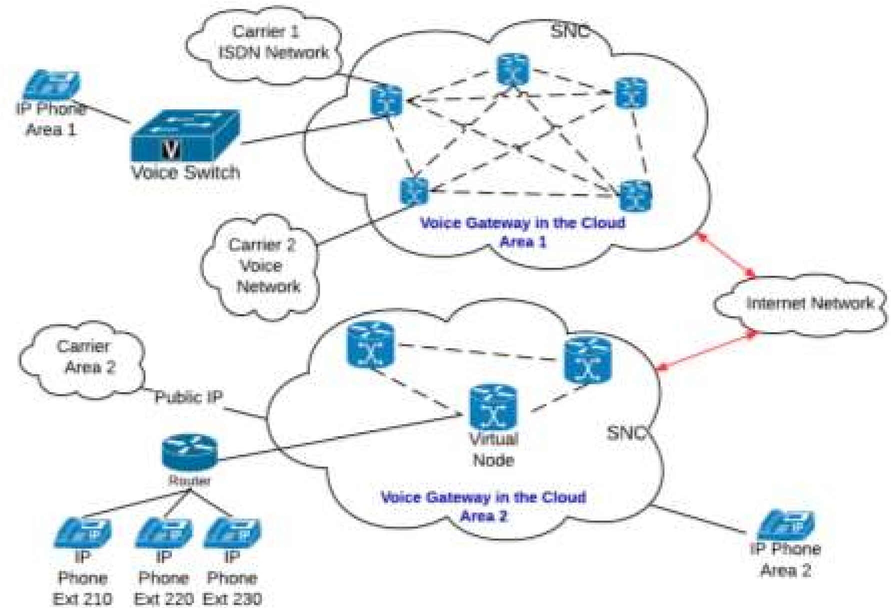
- one-dimensional, on-line
- classic NP-hard optimization problem

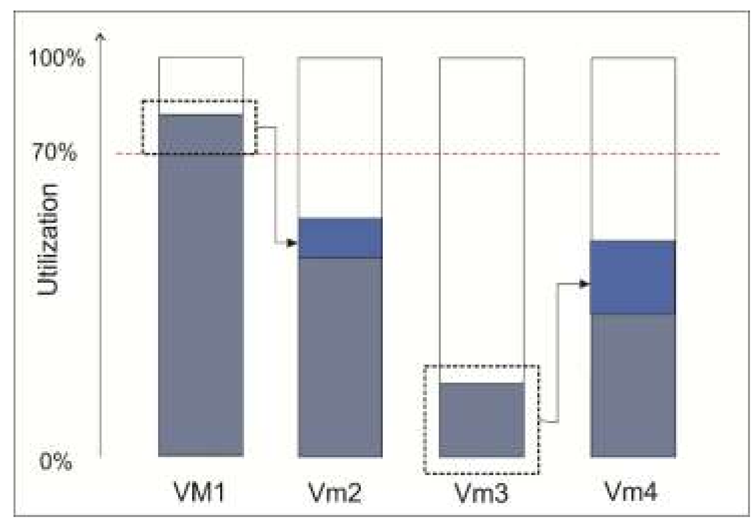
The principal novelty

- state of the bin is determined not only by actions of the decision maker during item allocations,
- but also by item completions after their lifespan.

Unlike in standard formulation,

- bins are always open
- dynamic
- items in bins can be terminated (call termination)
- utilization can be changed





VoIP quality of service

Quality of service (QoS) is a very important factor and its degradation is determined by: call delivery and **call processing**

Call processing → **Quality of voice** → **Codec** → **CPU Utilization**

CPU can not handle the stress when utilization is up to a threshold

A possible generalization of the voice quality is processor utilization:

- Jitters and broken audio appear when CPU utilization is high
- Memory does not influence on the voice quality reduction
- Codec increases the bandwidth but it is less significant [3]

Optimization criteria

Billing hours
($\bar{b}$)

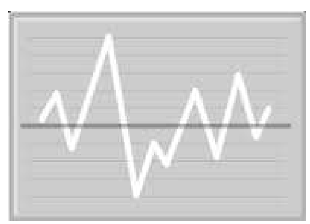
Quality reduction
($\bar{q}$)

Calls to Queue
($\bar{c}$)

Multi-objective optimization problem:
min($\bar{b}$), ***min***($\bar{q}$), ***min***($\bar{c}$)

Evaluation method

Degradation performance



The analysis assumes equal importance of each metric [5]

Problem with startup time delay.

Two objectives:
- Provider cost optimization.
- Voice quality.
- Calls to queue.

Bin-packing approach (well-known)
- one-dimensional, on-line
- classic NP-hard optimization problem

The principal novelty
- Bin startup time delay is determined by instance type, Operation system (Linux, Windows), OS image size, etc.
- It affects time sensitive applications and resource auto-scaling

Unlike in standard formulation,
- Bins are always open
- Dynamic
- Items in bins can be terminated (call termination)
- Utilization can be changed

Average VM startup time delay (stUp).

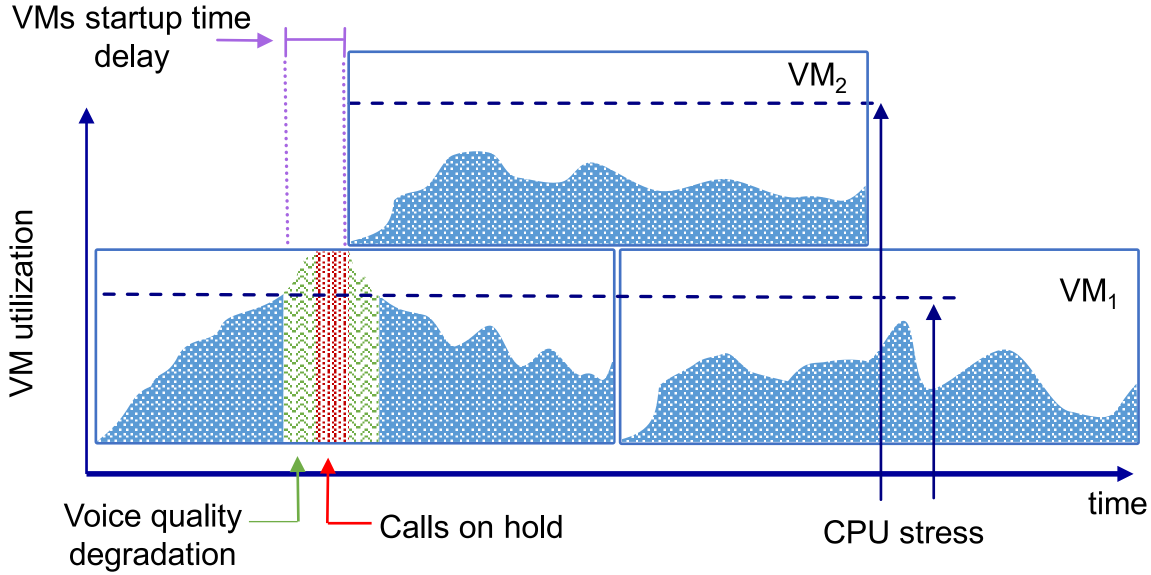
Cloud	OS	stUp (sec.)
EC2	Linux	96.9
	Windows	810.2
Azure	WebRole	374.8
	WorkedRole	406.2
	VMRole	356.6
Rackspace	Linux	44.2
	Windows	429.2

Cloud	stUp (sec.)
Google Cloud	31
AWS	47
Vexxhost	47
Linode	57
DigitalOcean	89
Rackspace	128
Windows	138

Call quality reduction

Call processing is a main issue which determine the quality of calls (QoS) and it focuses on:

- The voice quality influenced by CPU stress
- Calls delayed "on hold" due to the under-provisioning of resources



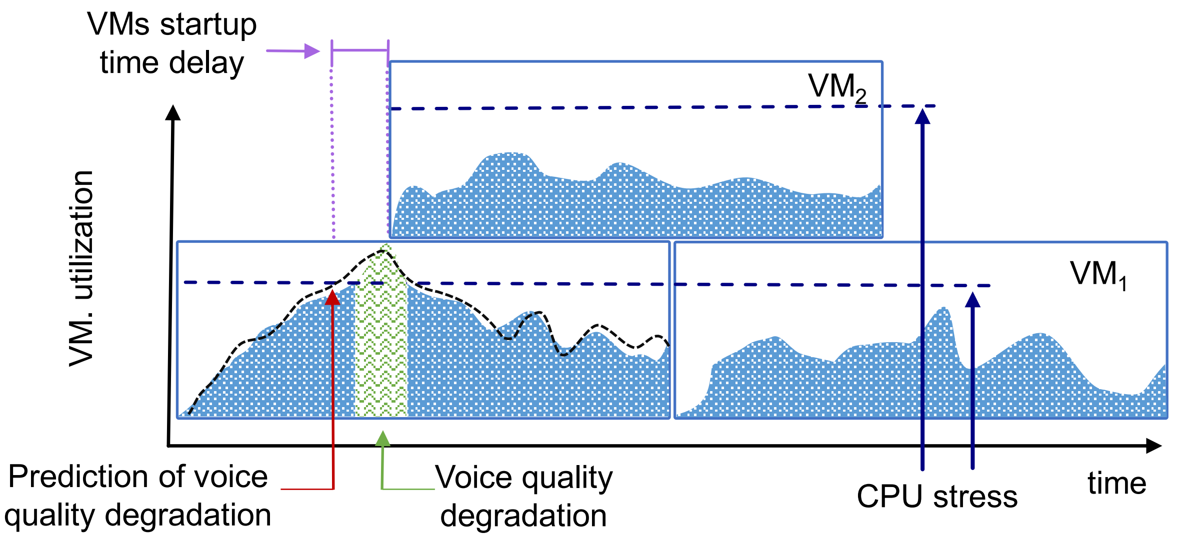
Calls allocation with startup time delay.

During VM startup time delay (StUp):

- VM continues call processing with voice quality degradation
- VM does not have enough resources, the system places calls on hold, waiting for available resources

Call allocation with load prediction

The goals of traffic prediction on cloud computing is to minimize the infrastructure costs and improve the QoS to the end user.



Calls allocation with prediction and startup time delay.

Call allocation and prediction can reduce the billing hours, calls on hold, and quality reduction

Advantages:

- Adequate VM provisioning

Disadvantages

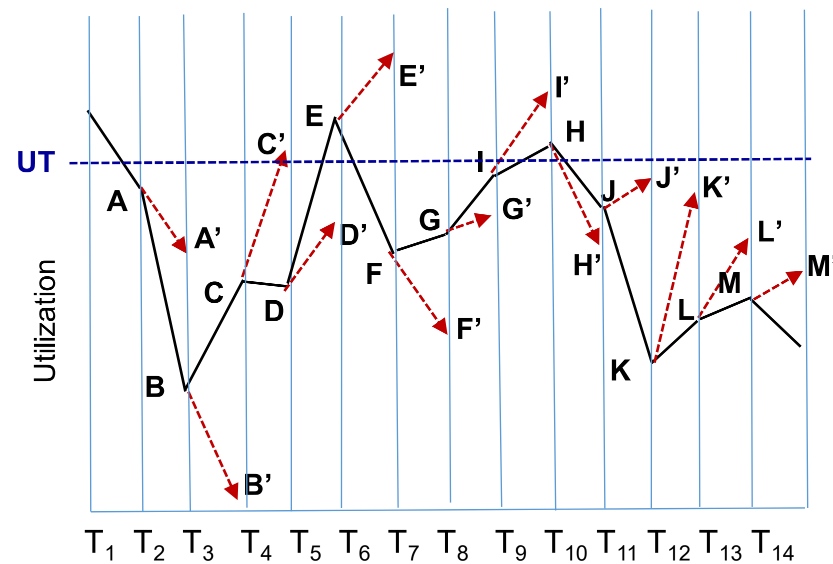
- Incorrect over-provisioning
- Under-provisioning

An accurate prediction model that does not increase the overhead considerately

Rate of change

Rate of Change is a dynamic distributed load balancing algorithm:

- Resources calculate the change in load between two Sample Intervals (*SI*)
- Difference in load (Δ) is an estimation on load for the next *SI*
- Δ is a mechanism to predict requests for new resources (VMs)



Let $u_i(t)$ be the utilization of SNC_i at time t, the rate of load change during SI=$[t - Si, t]$ is defined by:

$$\Delta_i(t) = (u_i(t) - u_i(t - Si))$$

CVoIP system is more vulnerable when the number of VMs is small, so prediction considers the number of VMs running in the system.

$$\Delta_i(t) = (u_i(t) - u_i(t - Si))/k_i(t)$$

Where $k_i(t)$ defines the number of VMs running on SNC_i at time t.

Call allocation strategies

Call allocation strategies.

	Name	Description
KF	Rand	Allocates job j to VM randomly using a uniform distribution.
	RR	Allocates job j to VM using a Round Robin algorithm.
UA	Ffit	Allocates job j to the first VM capable to execute it.
	Bfit	Allocates job j to VM with smallest utilization left.
	WFit	Allocates job j to VM with largest utilization left.
RA	MaxFTFit	Allocates job j to VM with farthest finish time.
	MidFTFit	Allocates job j to VM with shortest time to the half of its rental time.
	MinFTFit	Allocates job j to VM with closest finish time.
KF + TA	Rand_05 Rand_10 Rand_15 RR_05 RR_10 RR_15	Allocates job j to VM that finishes not less than in 5, 10, 15 minutes using the Rand, and RR strategies.
UA + TA	BFit_05 BFit_10 BFit_15 FFit_05 FFit_10 FFit_15 WFit_05 WFit_10 WFit_15	Allocates job j to VM that finishes not less than in 5, 10, and 15 minutes using the Bfit, FFit, and WFit strategies.

Call allocation strategies with prediction.

	Name	Description
LA	Rand_stUp Rand_s10 Rand_s20 Rand_s30 RR_stUp RR_s10 RR_s20 RR_s30	Allocates job j to VM using the Rand, and RR strategies. They use intervals of 10, 20, 30 and stUp seconds to estimate future load
UA + LA	BFit_stUp BFit_s10 BFit_s20 BFit_s30 FFit_stUp FFit_s10 FFit_s20 FFit_s30 WFit_stUp WFit_s10 WFit_s20 WFit_s30	Allocates job j to VM using BFit, FFit, and WFit strategies. They use intervals of 10, 20, 30 and stUp seconds to estimate future load