Supercomputing Consortium of Russian Universities
6th Summer Supercomputing Academy

Supercomputing Co-Design Technology

Prof. Vladimir Voevodin
Deputy Director, Research Computing Center, MSU
Head of Department on Supercomputers and Quantum Informatics, CMC, MSU

voevodin@parallel.ru

June 23rd, 2017,
Moscow – Saint-Petersburg – Arkhangelsk – Ekaterinburg

Supercomputer Centers Today

- Huge potential,
- High demand,
- Technological complexity,
- Difficult support and maintenance,
- High cost,
- Diversity of architectures,
- Complexity of architectures.

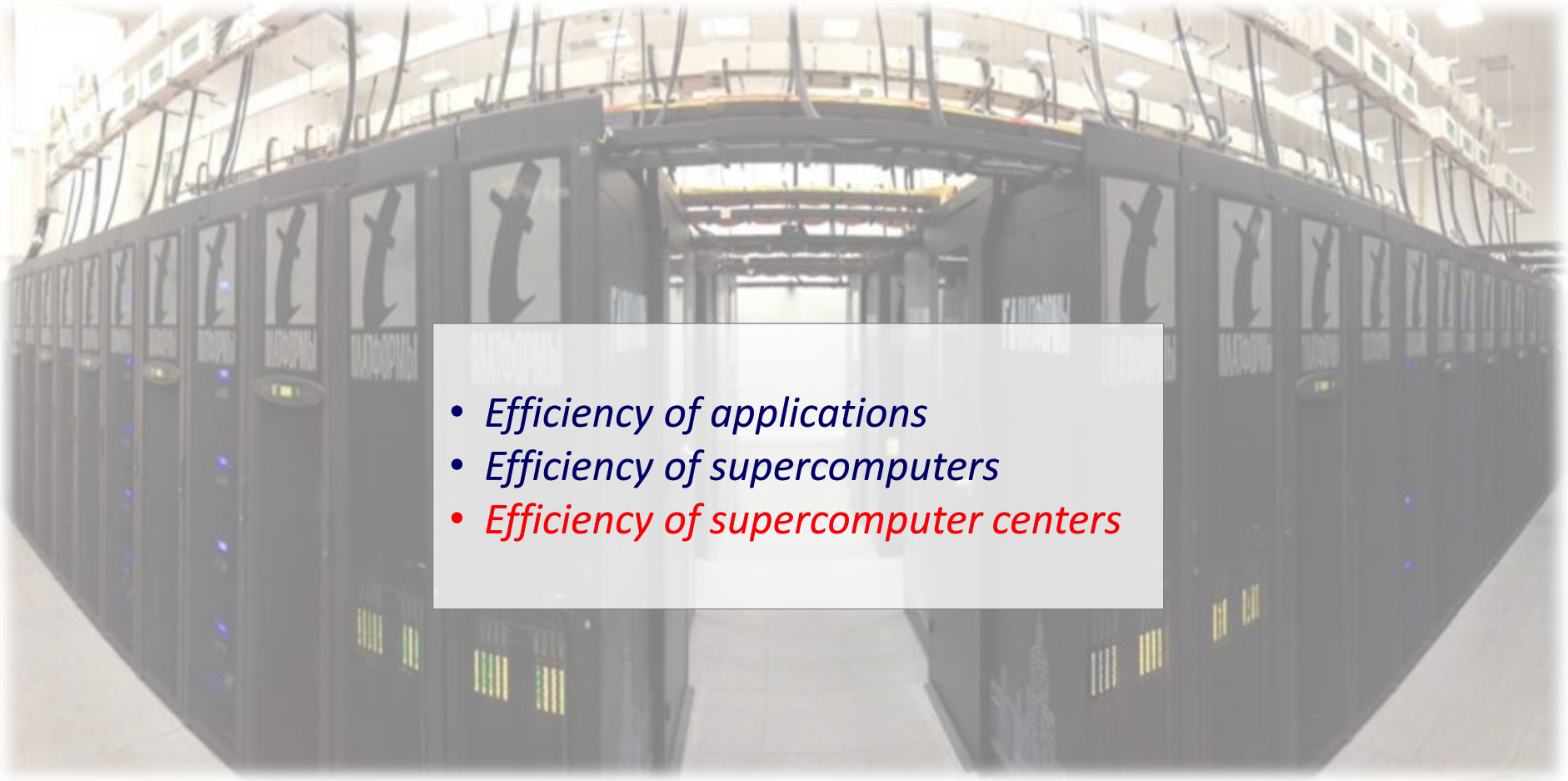
MSU Supercomputers: “Lomonosov” and “Lomonosov-2”

MSU Supercomputing Center today:
Users: 2955
Projects: 880

MSU Faculties / Institutes : 21
Institutes of RAS : 95
Russian Universities: 102

Very serious technological and intellectual potential…

Potential of Supercomputer Centers

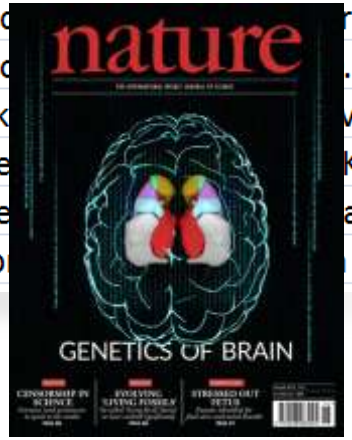


- *Efficiency of applications*
- *Efficiency of supercomputers*
- *Efficiency of supercomputer centers*

Efficiency of Supercomputer Centers

(publications, bibliometric parameters)

Reference	Impact-Factor
A. A. Popov, S. Yang, L. Dunsch. "Endohedral Fullerenes."Chemical Reviews 2013, 113 (8),	41,298
Yolamanova M., Meier C., Shaytan A.K., Vas V., Bertoncini C., Arnold F.,Zirafi O., Usmani S.,	31,170
M. Yolamanova, C. Meier, A. K. Shaytan, V. Vas, C. W. Bertoncini, F. Arnold, O. Zirafi, S. M. U	27,270
Bravaya K.B., Grigorenko B.L., Nemukhin A.V., Krylov, A.I.// Accounts of Chemical Research	20,833
Nikita Gudimchuk, Benjamin Vitre, [...], and Ekaterina L. Grishchuk, Kinetochore kinesin CE	20,800
V.E.Dmitrienko, E.N.Ovchinnikova, S.P.Collins, G.Nisbet, G.Beutier, Y.O.Kvashnin, V.V.Mazu	19,352
I. V. Kuvychko, C. Dubceac, S. H. M. Deng, X. B. Wang, A. A. Granovsky, A. A. Popov, M. A. P	13,734
Kuvychko, Igor V and Dubceac, Cristina and Deng, Shihu HM and Wang, Xue-Bin and Granc	13,734
Kvashnin A.G., Chernozatonskii L.A., Yakobson B.I., Sorokin P.B. Phase diagram of quasi-tw	13,025
Q. Deng, A ral Metallofullerenes: How straine	10,677
Grigorenko .I.; Krylov A.I. // Journal of the Am	10,677
Grigorenko v A // JOURNAL OF AMERICAN CH	10,677
Tang D.M., K., Koskinen P., Ajayan P., Yakobs	10,015
A. L. Svitov ang, M. M. Olmstead, A. L. Balch,	10,015
Davydov II the L12 stalk of bacterial and org	10,015



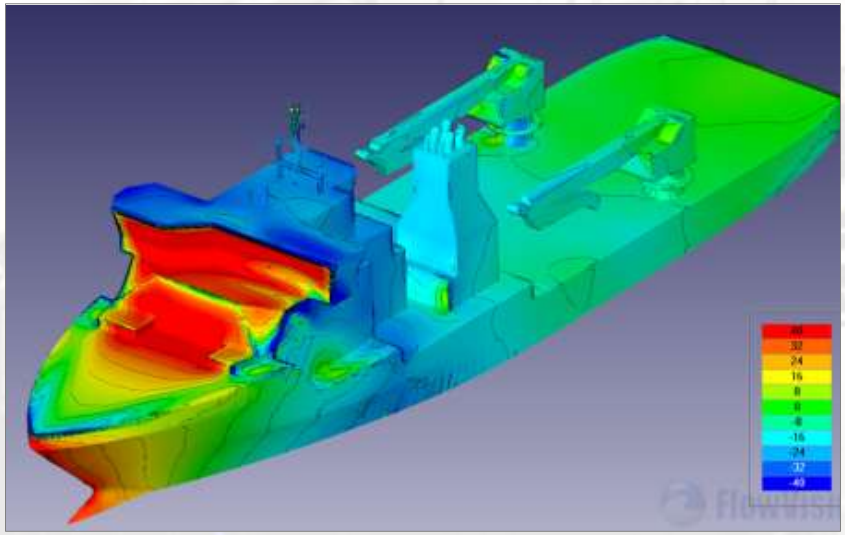
Efficiency of Supercomputer Centers

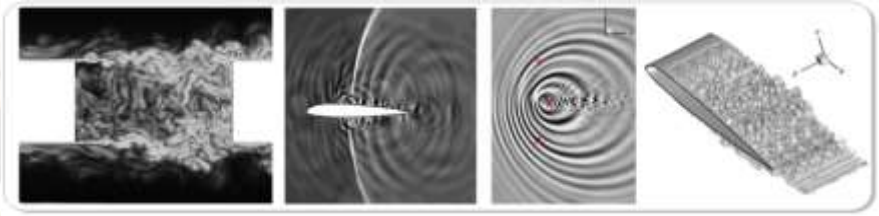
(socially important and/or nation-wide visible projects)

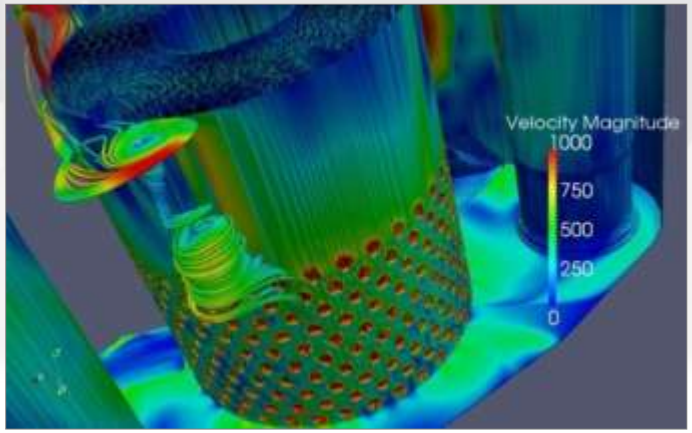
Supercomputer “Lomonosov-2”: visual special effects for “First Time” (“Время первых”).

Efficiency of Supercomputer Centers

(partnership with industry)



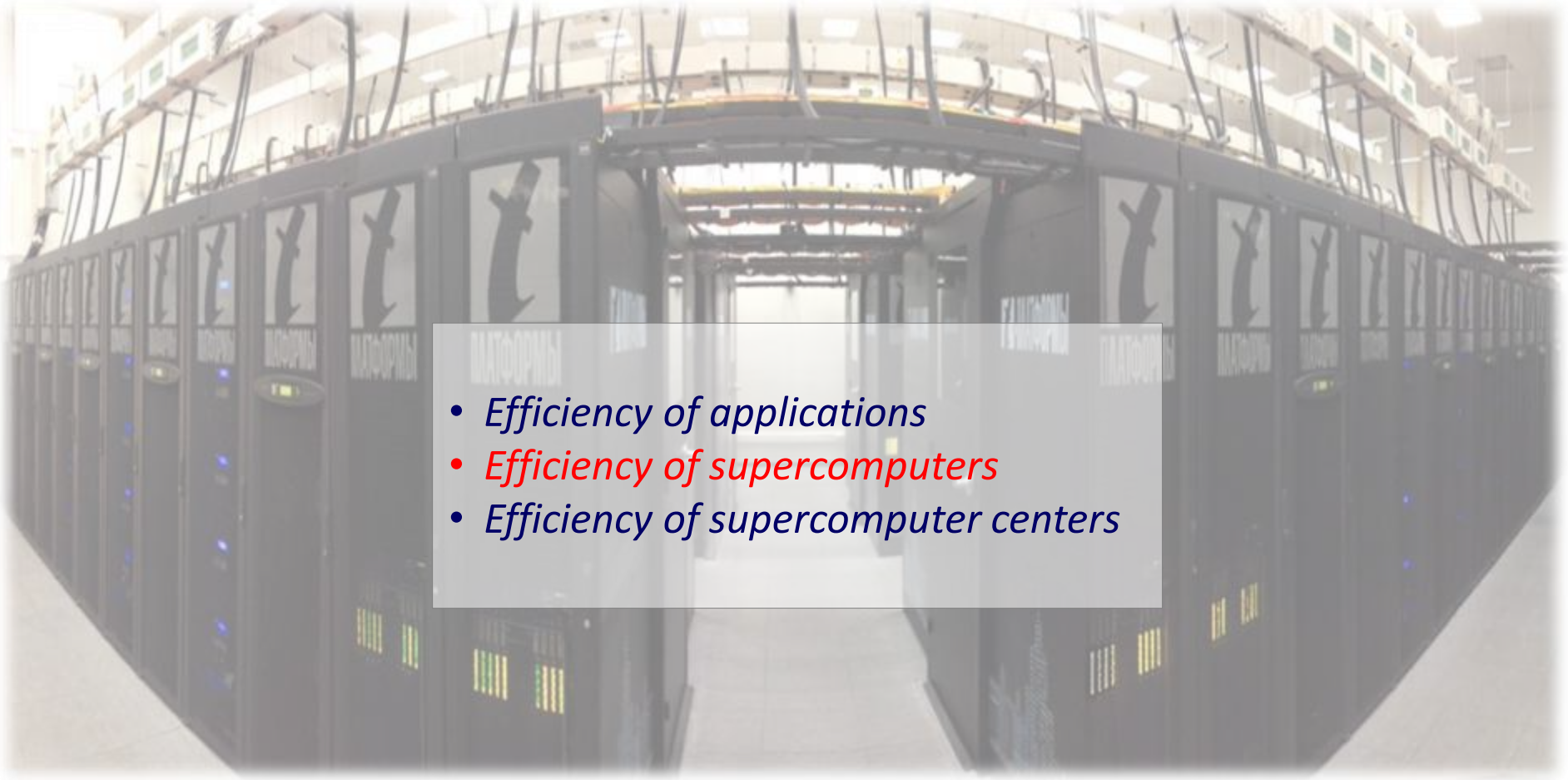




Efficiency of Supercomputer Centers

Bright scientific discoveries …

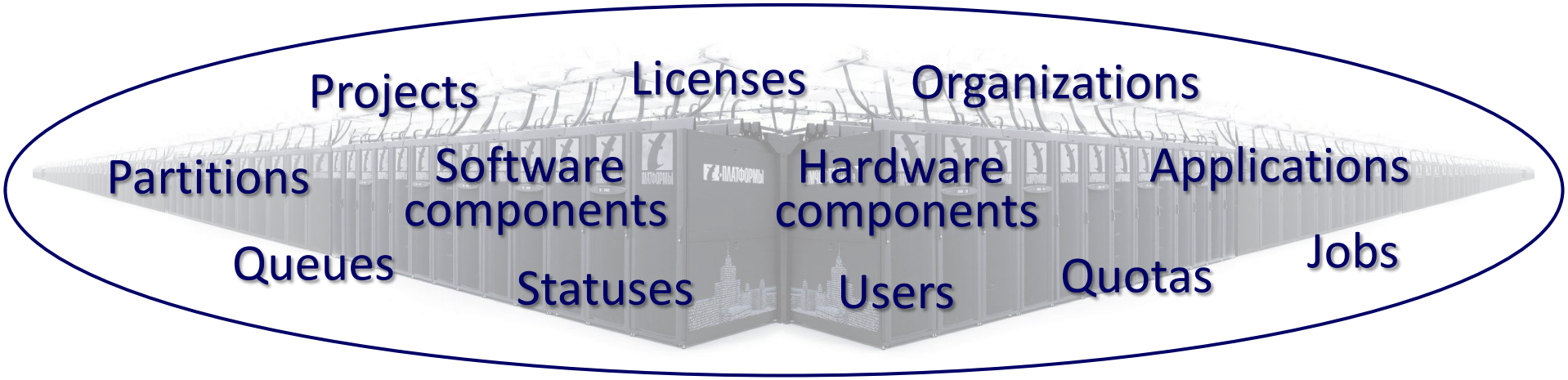
Potential of Supercomputer Centers



- *Efficiency of applications*
- *Efficiency of supercomputers*
- *Efficiency of supercomputer centers*

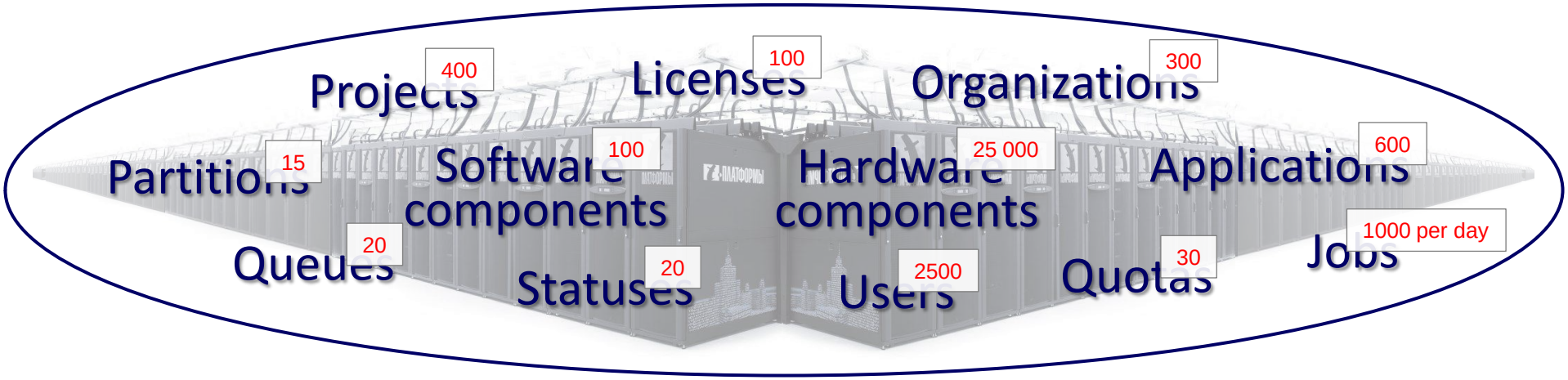
Complexity of Supercomputing Centers

Is it difficult to control few components ? *A few ?..*



A few? Info on MSU "Lomonosov" Supercomputer :

(1.7 Pflops, 6000 computing nodes, 12K CPUs, 2K GPUs…)

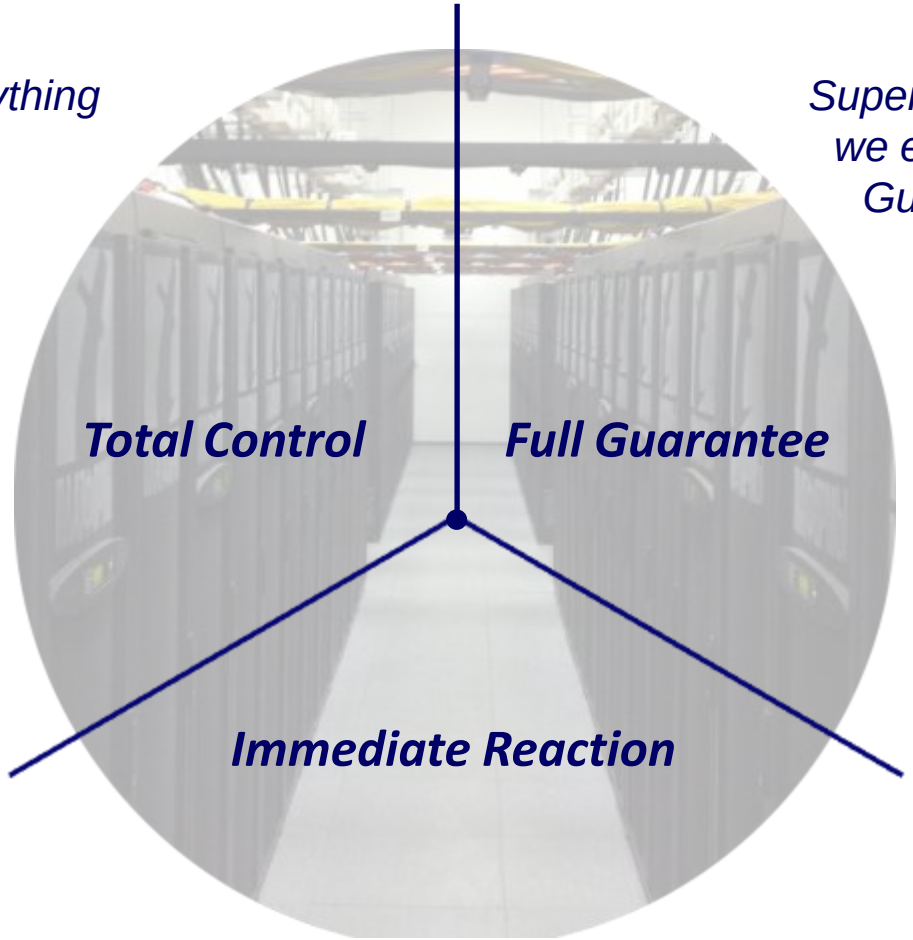


Current trend: all these numbers grow extremely fast !

Efficiency of Supercomputers

We must control everything what is necessary to control efficiency permanently.

Supercomputer behaves like we expect it should behave; Guarantee of coincidence between theory and practice.



We must know about everything and react immediately.

DiMMon: Total Monitoring of Petascale Supercomputers
(Supercomputing Co-Design Technologies and Tools)

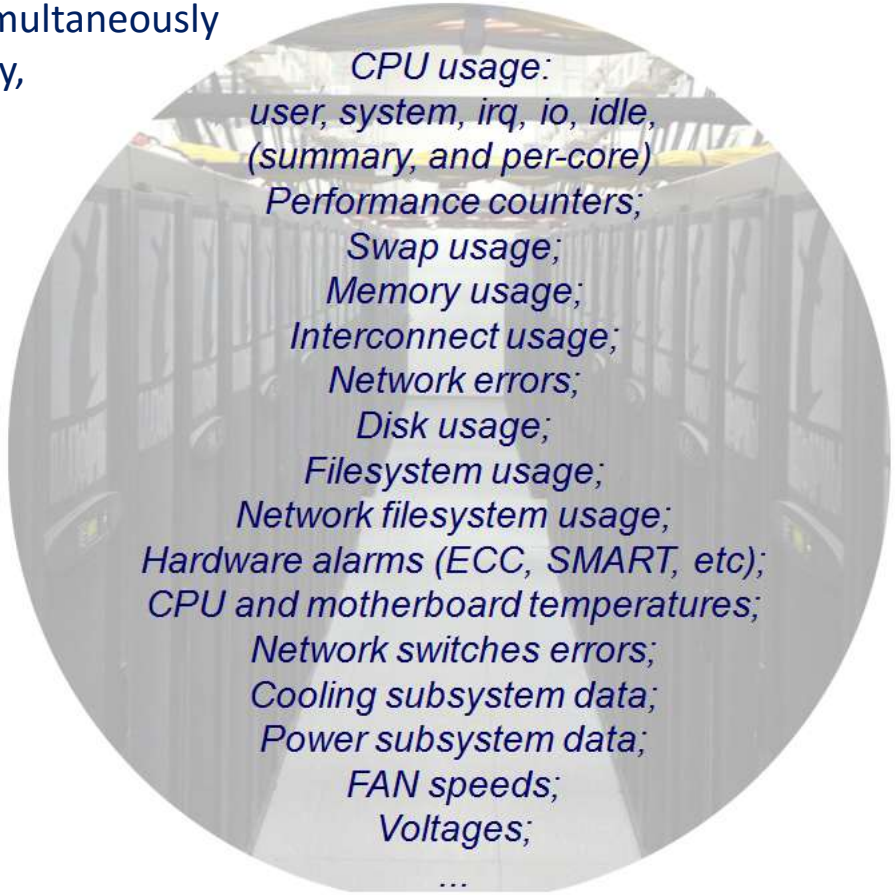
==> We need the total control over HW&SW components of supercomputers, where total monitoring is a key issue. Why is the total monitoring really hard ?

Many different HW&SW components operating simultaneously
Many different applications running simultaneously,
Many different goals of monitoring…

Monitoring system, requirements:

- we need to know: what, where, when.
- scalability: thousands computing nodes,
  dozens sensors per node,
- high frequency: a few seconds and less,
- active and passive modes,
- …

A traditional approach (store all data first, process necessary data later) for the "Lomonosov" supercomputer doesn't work:
initial monitoring data rate – 120 MB/c
3+ Pbytes/year… BigData comes…
A simple analysis of the monitoring data required minutes, hours, days…



Too many possible reasons of performance degradation, but we need to collect and keep all this information

DiMMon: Total Monitoring of Petascale Supercomputers
(Supercomputing Co-Design Technologies and Tools)

Good questions:
Do we really need to keep all the data? Is “store first, process later” the best strategy?
Where is the best point to process data? Static/Dynamic? …

A smart approach to monitoring:
- **on-the-fly analysis**: all relevant information should be extracted from the monitoring data before it’s stored in a database;
- **on-site analysis**: monitoring data must be processed where the data were obtained (process first, move data (if necessary) later);
- **dynamic reconfiguration of monitoring systems**: the monitoring system must be capable to change dynamically its configuration, depending on the current load on the supercomputer and the specific analysis objectives.

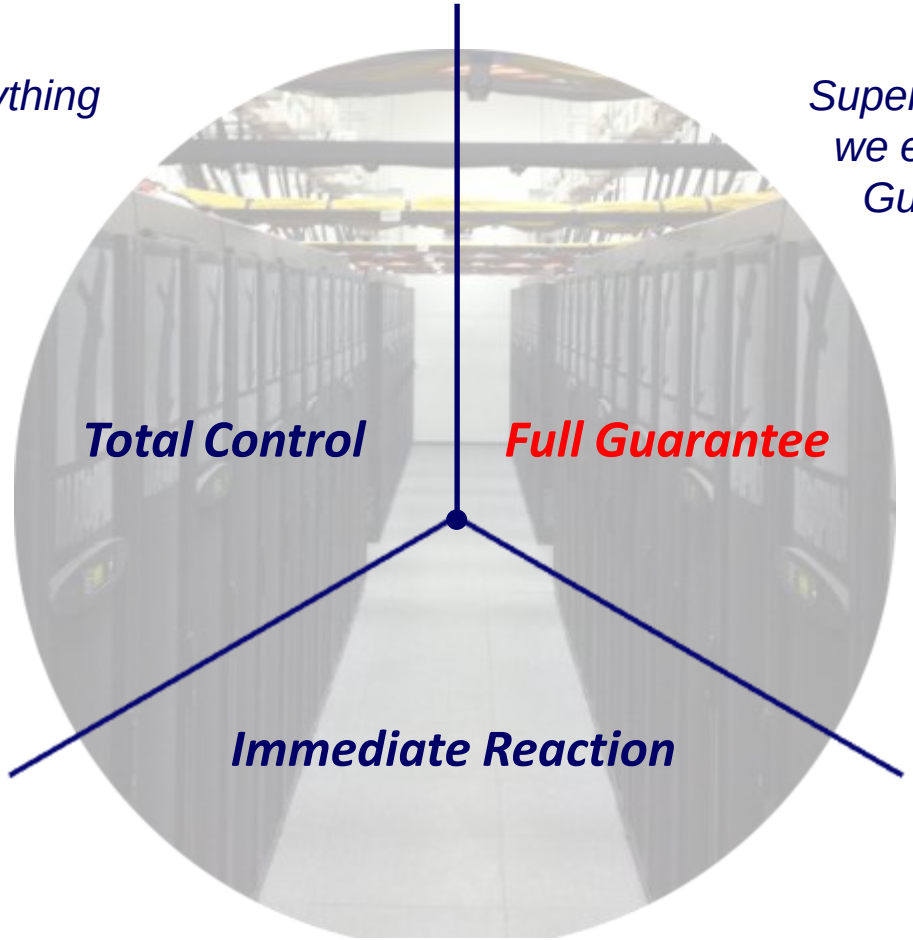
No problem with monitoring of the whole “Lomonosov” supercomputer…
(estimations up to x100-x1000)

Two interesting statements inspired by the practice:
- What is BigData: characteristic of a certain problem or lack of our understanding (nature of data, structure of data, objectives of analysis…) ?
- If you have to deal with BigData, typically you don’t need the most of the data…

Efficiency of Supercomputers

We must control everything what is necessary to control efficiency permanently.

Supercomputer behaves like we expect it should behave; Guarantee of coincidence between theory and practice.



We must know about everything and react immediately.

OctoTron: predictability for supercomputers
(Supercomputing Co-Design Technologies and Tools)

***Large numbers in supercomputers**: cores, processors, accelerators, nodes, HW&SW components, files, indexes, users, projects, processes, threads, running and queued jobs…*

***What is now?** We hope that a component works until we get an evidence that it has failed.*

Our expectations = Reality

***We need a guarantee**: if something goes wrong inside a supercomputer we shall be notified immediately.*

We want a system behaves in a way we expect it should behave.

OctoTron: predictability for supercomputers

(Supercomputing Co-Design Technologies and Tools)

Supercomputers should be autonomous in self-control.
(They become more dynamic, more sophisticated, more and more parallel)

…The larger supercomputers, the more autonomous they should be...

OctoTron: predictability for supercomputers
(Supercomputing Co-Design Technologies and Tools)

A guarantee of "our expectations = reality", how this can be done?

- *a formal model of supercomputers (**model is a graph**),*
- *a set of formal rules,*
- *a set of reactions,*

Autonomous life and control of MSU supercomputers:

*-**"Chebyshev"** supercomputer, 60 Tflops, 625 CPUs:*
10 228 nodes, 24 698 edges, 205 044 attributes, 160 rules, 100 reactions;
*- **"Lomonosov"** supercomputer, 1.7 Pflops, 12 000 CPUs, 2 000 GPU:*
116 000 nodes, 332 000 edges, 2 400 000 attributes,...

Initial deployment, Detection of faults, critical and emergency situations, Turning off minimum amount of hardware, Self diagnostics, Previous accidents, etc. are done according to a model and rules.

Current trend: many decisions about control over HW&SW of supercomputers must be taken automatically.

OctoTron: predictability for supercomputers
(Supercomputing Co-Design Technologies and Tools)

OctoTron model of supercomputers: targets, triggers, rules, reactions (on Python):

```
"sensor" : {
    "ping" : Boolean(timeout),
  },

  "trigger" : {
    "ping_failed" : Match("ping", False)
  },

  "react" : {
    "notify_ping_failed" : Reaction()
      .On("ping_failed", 3, 0)
      .Begin(status("tag", "NETWORK").Msg("loc", "{" + key + "}")
        .Msg("descr", "{type}: ping failed three times")
        .Msg("msg"  , "{type}[{" + key + "}]: ping failed three times"))
      .End(rstatus("tag", "NETWORK").Msg("loc", "{" + key + "}")
        .Msg("descr", "{type}: ping ok")
        .Msg("msg"  , "{type}[{" + key + "}]: ping ok")),
  }
```

```
"const" : {
    "mountpoint" : mountpoint,
    "type" : "mountpoint"
  },
  "static" : {
    "threshold_percent_free" : pct_threshold,
    "threshold_bytes_free" : threshold
  },
  "sensor" : {
    "percent_free" : Long(timeout),
    "bytes_free" : Long(timeout)
  },
  "trigger" : {
    "low_percent_free" : LTArg("percent_free", "threshold_percent_free"),
    "low_bytes_free" : LTArg("bytes_free", "threshold_bytes_free"),
    "low_space" : StrictLogicalOr("low_percent_free", "low_bytes_free")
  },
  "react" : {
    "notify_low_space" : Reaction()
      .On("low_space")
      .Begin(reaction("tag", "DISK").Msg("loc", loc)
        .Msg("descr", loc_s + "free space is low")
        .Msg("msg"  , loc_l + "free space is low: {percent_free}% / {bytes_free}B"))
      .End(GenRStatus(reaction)("tag", "DISK").Msg("loc", loc)
        .Msg("descr", loc_s + "free space is ok")
        .Msg("msg"  , loc_l + "free space is ok: {percent_free}% / {bytes_free}B")),
  }
}
```

OctoTron: predictability for supercomputers
(Supercomputing Co-Design Technologies and Tools)

*** reported events ***

…

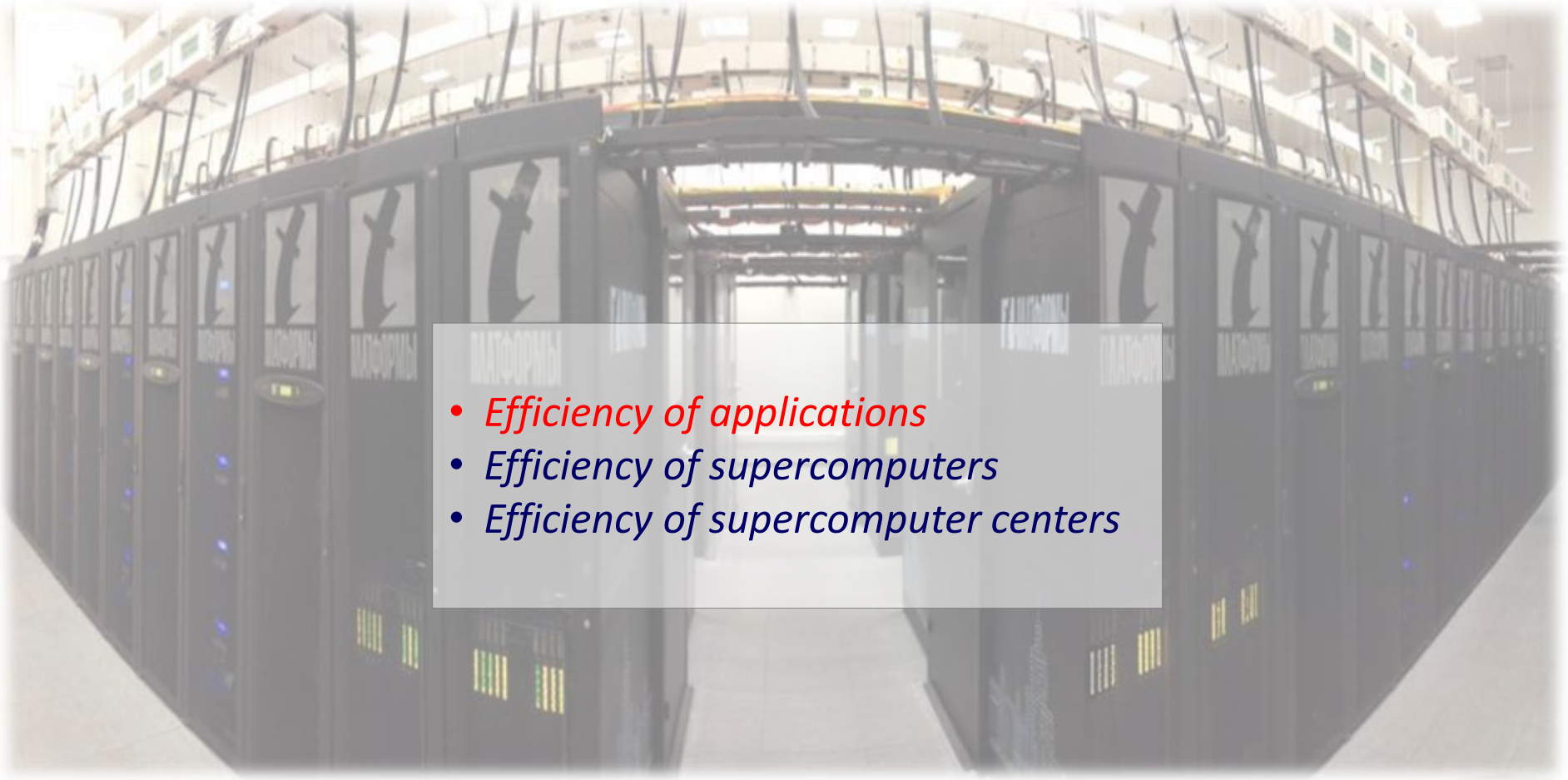
7, CRITICAL, "ups: type 2 errors growing"
7, CRITICAL, "ups: type 1 errors growing"
1, CRITICAL, "critical balance on modem"

5, DANGER, "partition: too many blocked nodes"
3, DANGER, "cold_sensor: very high temperature on cold sensor"
2, DANGER, "emu: ping failed three times"
1, DANGER, "partition: the queue has lost some nodes"
1, DANGER, "octotron on stat1.lom.parallel.ru:4448 failed one check"
1, DANGER, "low balance on modem"

62, WARNING, "big_eth_switch port: is down"
29, WARNING, "rkp: high fluid temperature"
12, WARNING, "cold_sensor: high temperature on cold sensor"
8, WARNING, "rkp: high air temperature"
5, WARNING, "hot_sensor: high temperature on hot sensor"

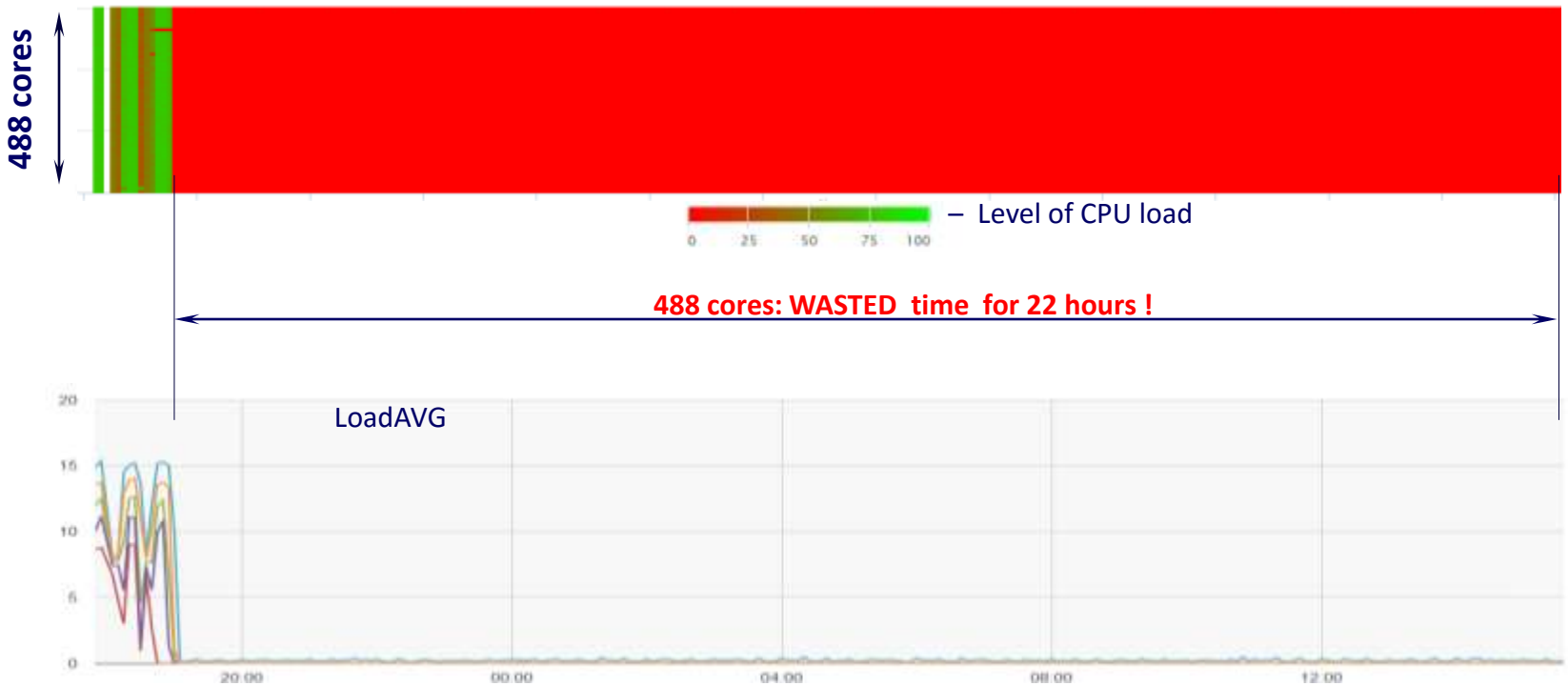
…

Potential of Supercomputer Centers



- *Efficiency of applications*
- *Efficiency of supercomputers*
- *Efficiency of supercomputer centers*

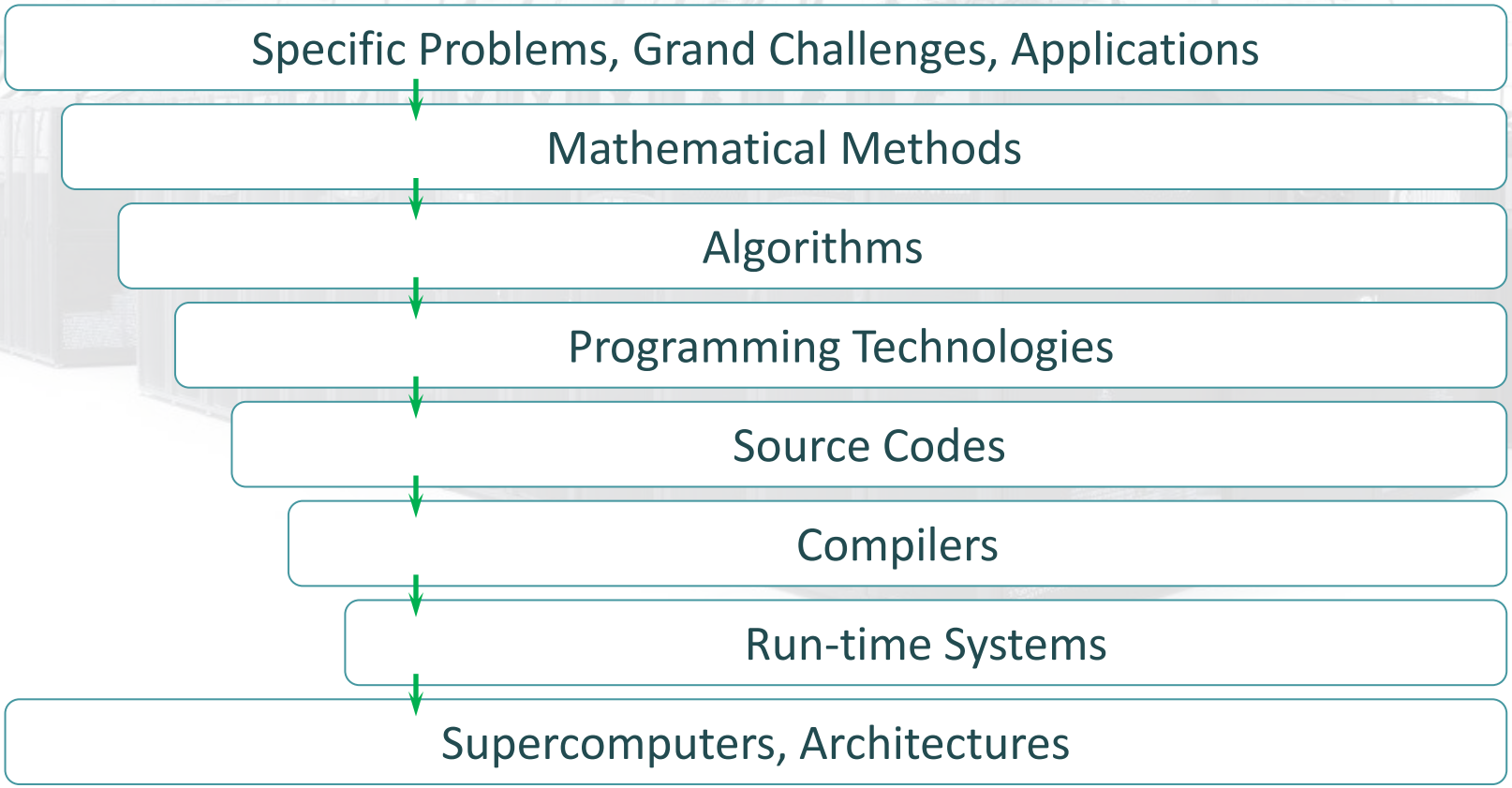
Efficiency of supercomputing applications
(Potential of supercomputer centers)



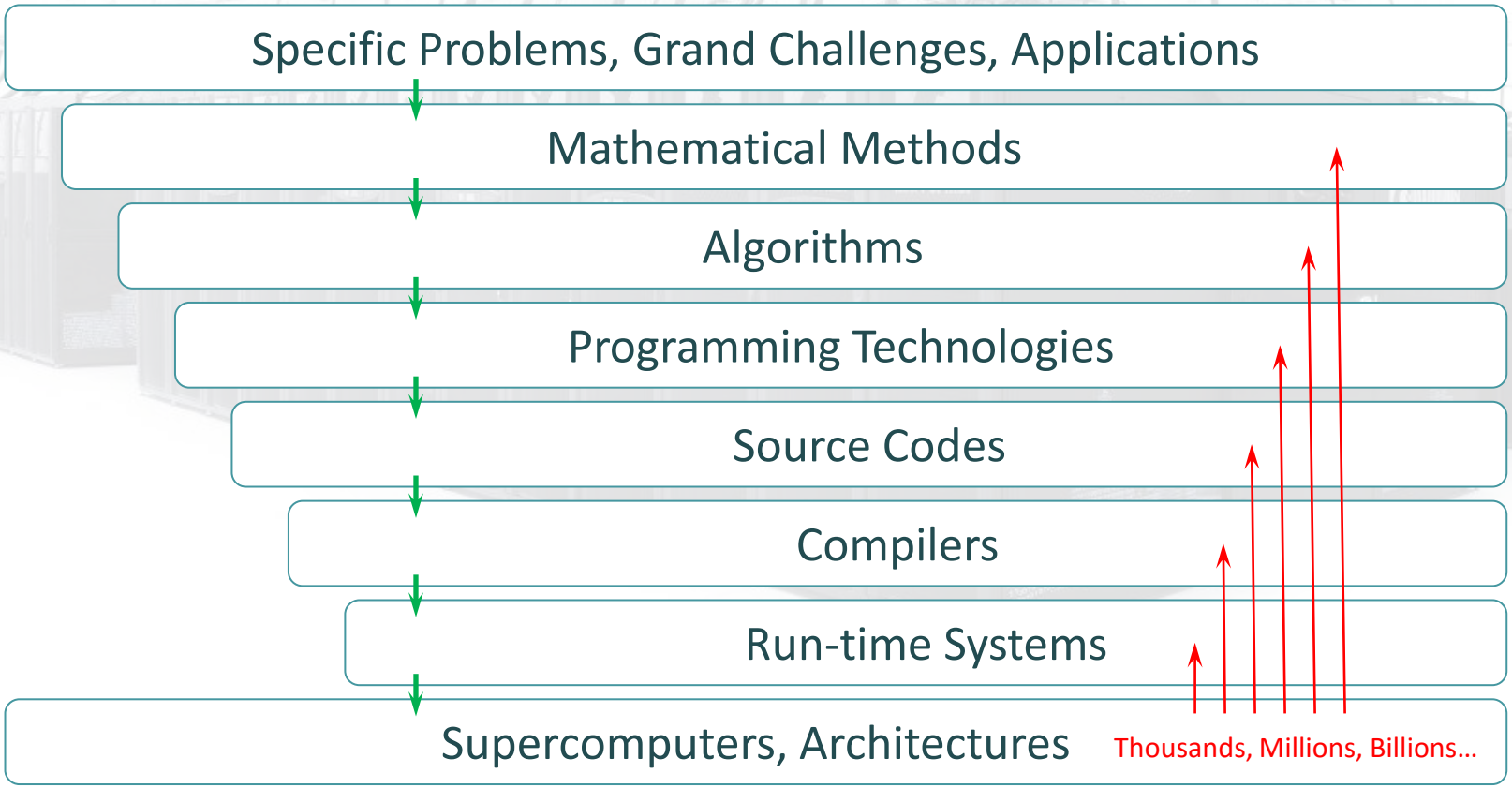
What could be a reason of this situation?

- Hardware failure? Yes, it could be …
- Software failure? Yes, it could be …
- Error in the code? Yes, it could be …
- Algorithmic problem? Yes, it could be …

Supercomputing Co-Design Technologies and Tools

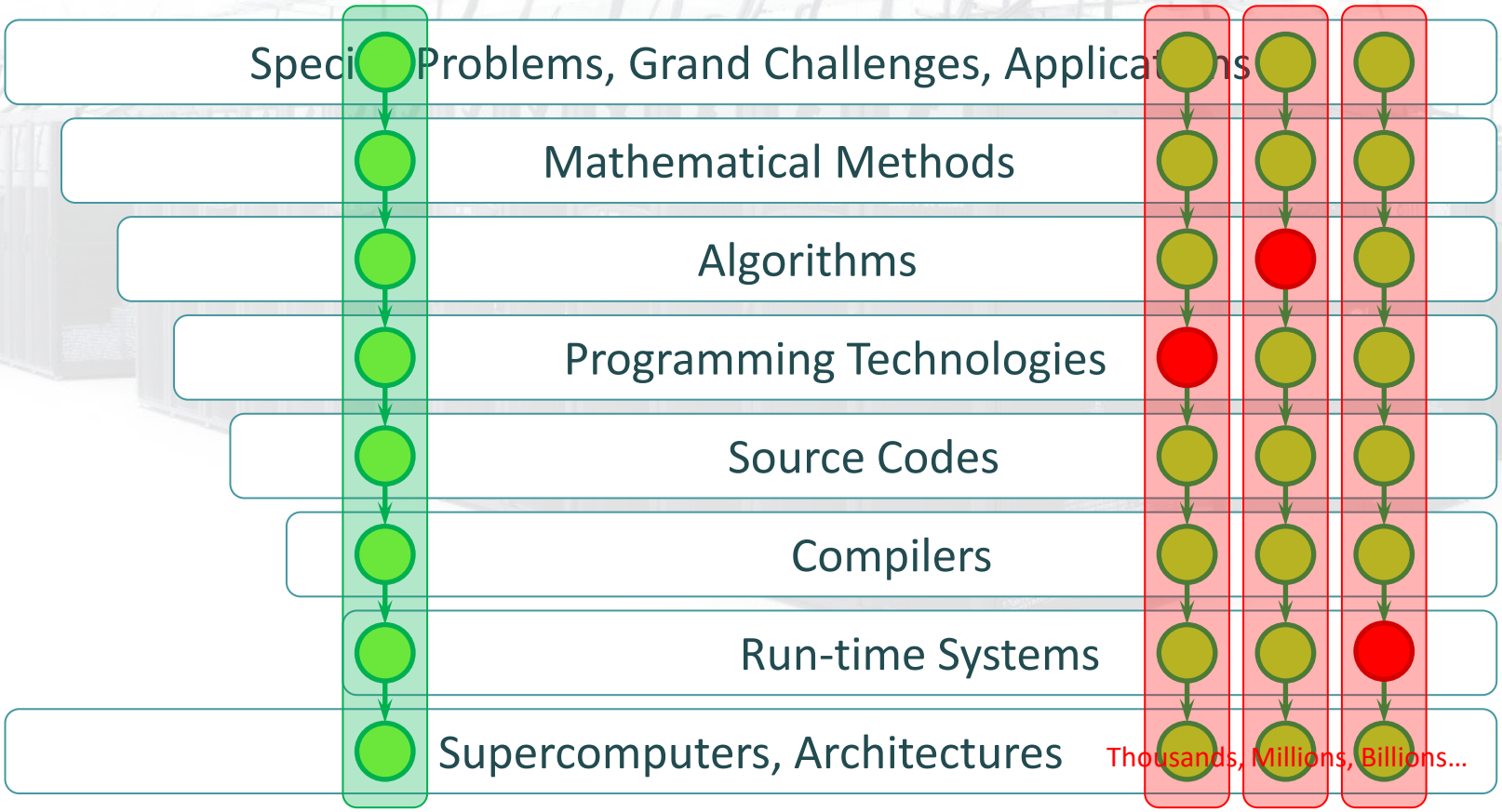


Supercomputing Co-Design Technologies and Tools



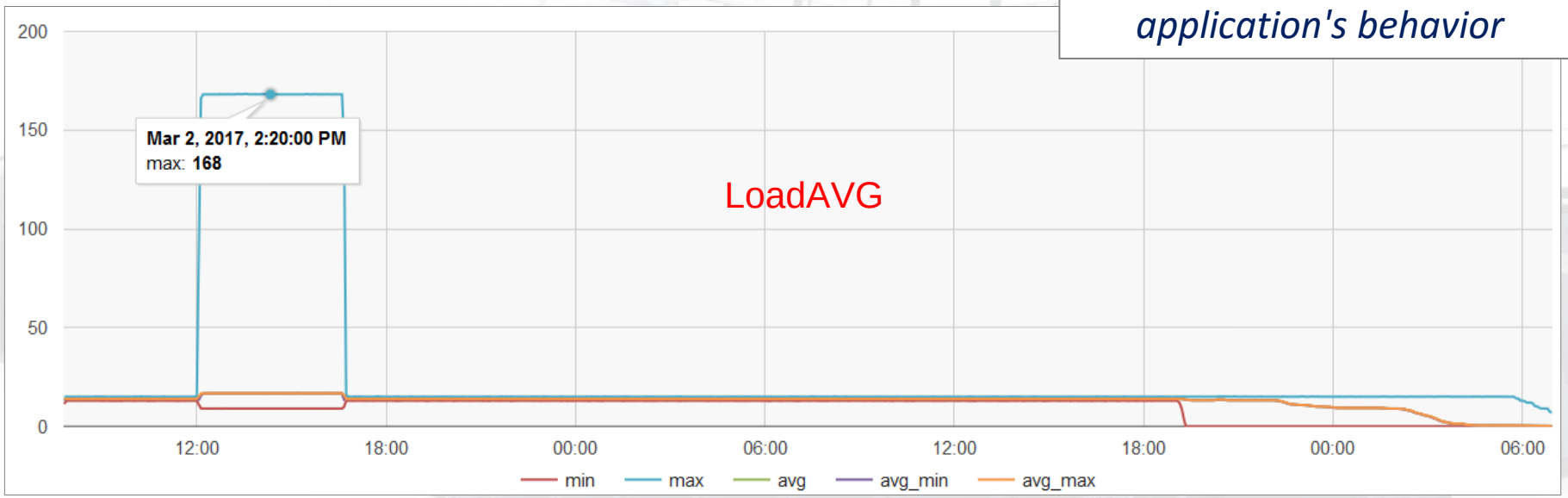
Supercomputing Co-Design Technologies and Tools

How to ensure efficiency of this supercomputing co-design chain ?



JobDigest: efficiency of supercomputer applications
(Supercomputing Co-Design Technologies and Tools)

Fine analysis of application's behavior



JobDigest: efficiency of supercomputer applications

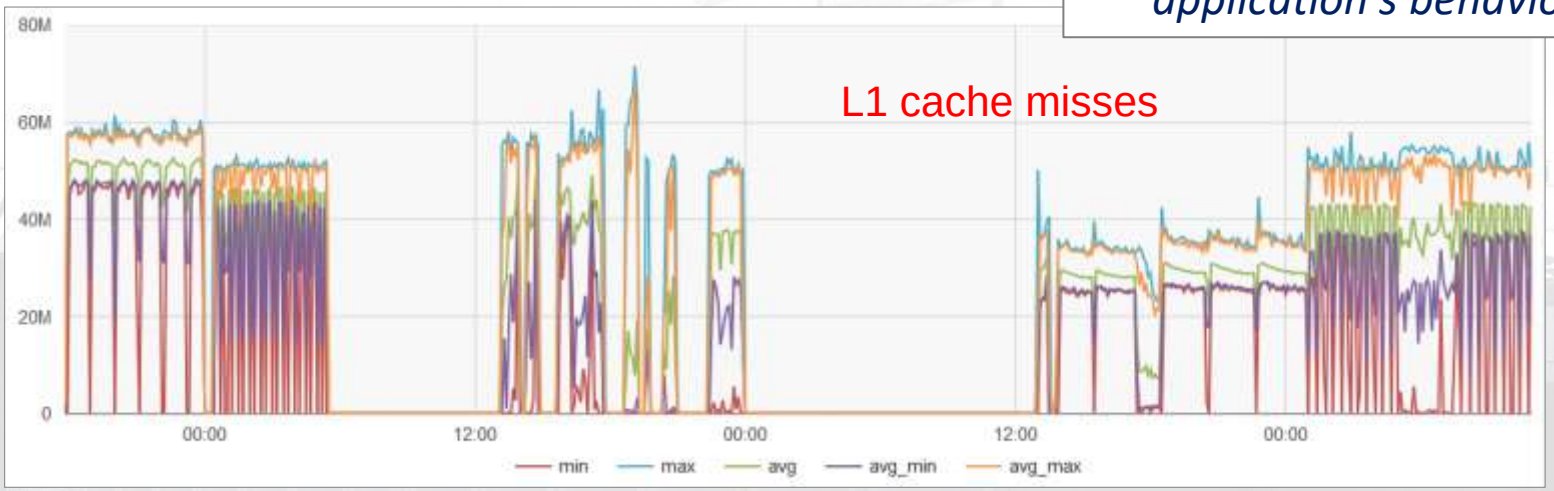
(Supercomputing Co-Design Technologies and Tools)

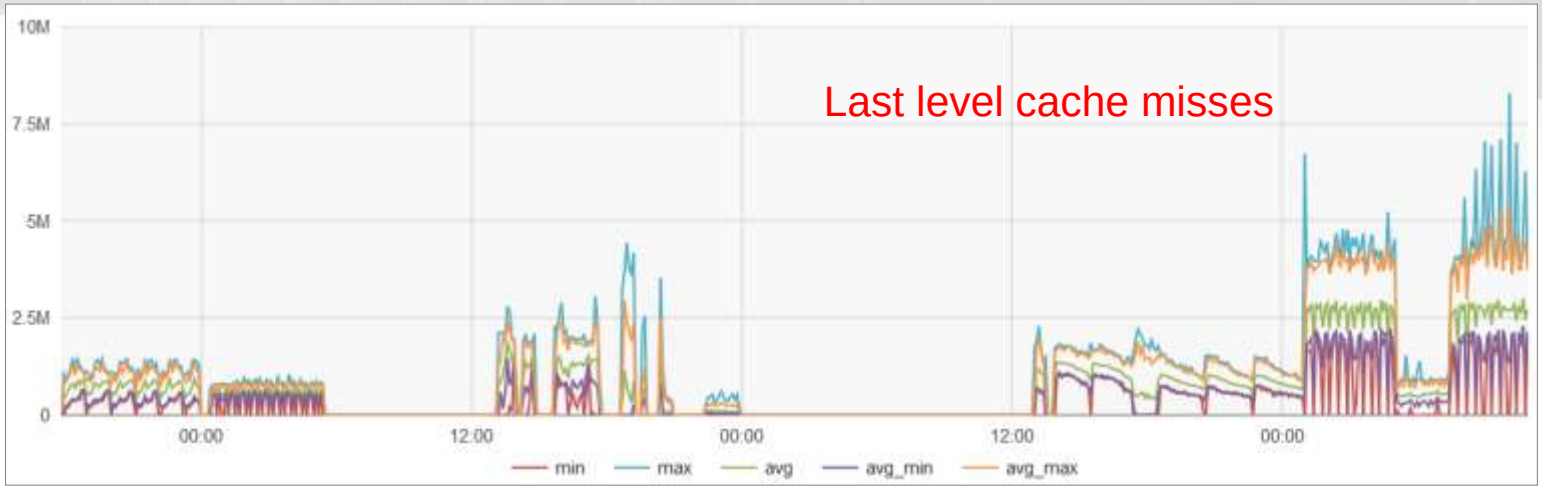
CPU Load

Fine analysis of application's behavior

JobDigest: efficiency of supercomputer applications
(Supercomputing Co-Design Technologies and Tools)

Fine analysis of application's behavior





JobDigest+OctoStat: efficiency of supercomputer applications
(Supercomputing Co-Design Technologies and Tools)

[2017-04-23]: Daily statistics
-------- Supercomputer “Lomonosov-1” -------
Recent task statistics for 2017-04-23:

Daily statistics: http://graphit.parallel.ru:5000/job_stat/preset

Suspicious tasks: everything is fine

http://graphit.parallel.ru:5000/job_table/table/page/0?date_from=1492819201&date_to=1492905601&...

-------- Supercomputer “Lomonosov-2” -------
Recent task statistics for 2017-04-23:

Daily statistics: http://graphit.parallel.ru:5001/job_stat/preset

Suspicious tasks: everything is fine

http://graphit.parallel.ru:5001/job_table/table/page/0?date_from=1492819202&date_to=1492905602&...

JobDigest+OctoStat: efficiency of supercomputer applications
(Supercomputing Co-Design Technologies and Tools)

[2017-04-12]: Daily statistics
-------- Supercomputer “Lomonosov-1” -------
Recent task statistics for 2017-04-12:

Daily statistics: http://graphit.parallel.ru:5000/job_stat/preset

Suspicious tasks: **7**

http://graphit.parallel.ru:5000/job_table/table/page/0?date_from=1491868801&date_to=1491955201&...

-------- Supercomputer “Lomonosov-2” -------
Recent task statistics for 2017-04-12:

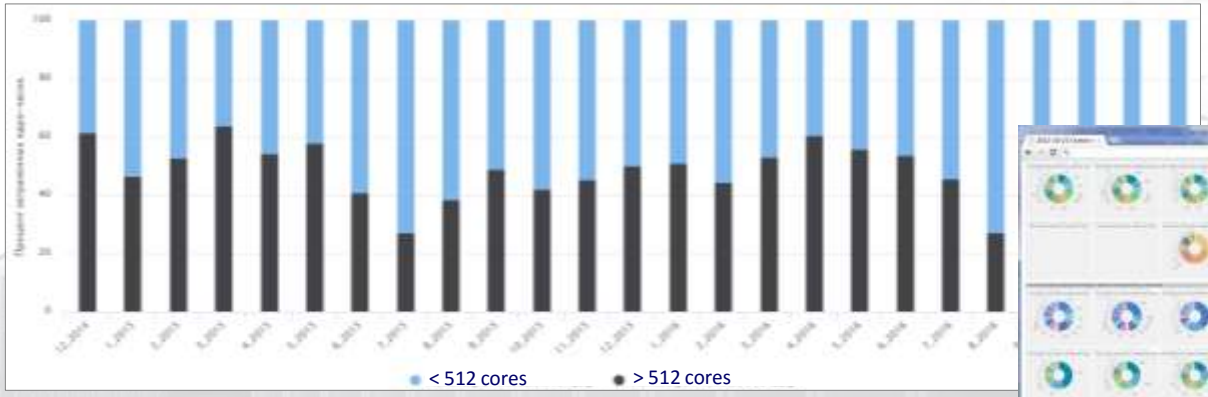
Daily statistics: http://graphit.parallel.ru:5001/job_stat/preset

Suspicious tasks: **4**

http://graphit.parallel.ru:5001/job_table/table/page/0?date_from=1491868802&date_to=1491955202&...

OctoStat: analytics on supercomputing centers

(Supercomputing Co-Design Technologies and Tools)

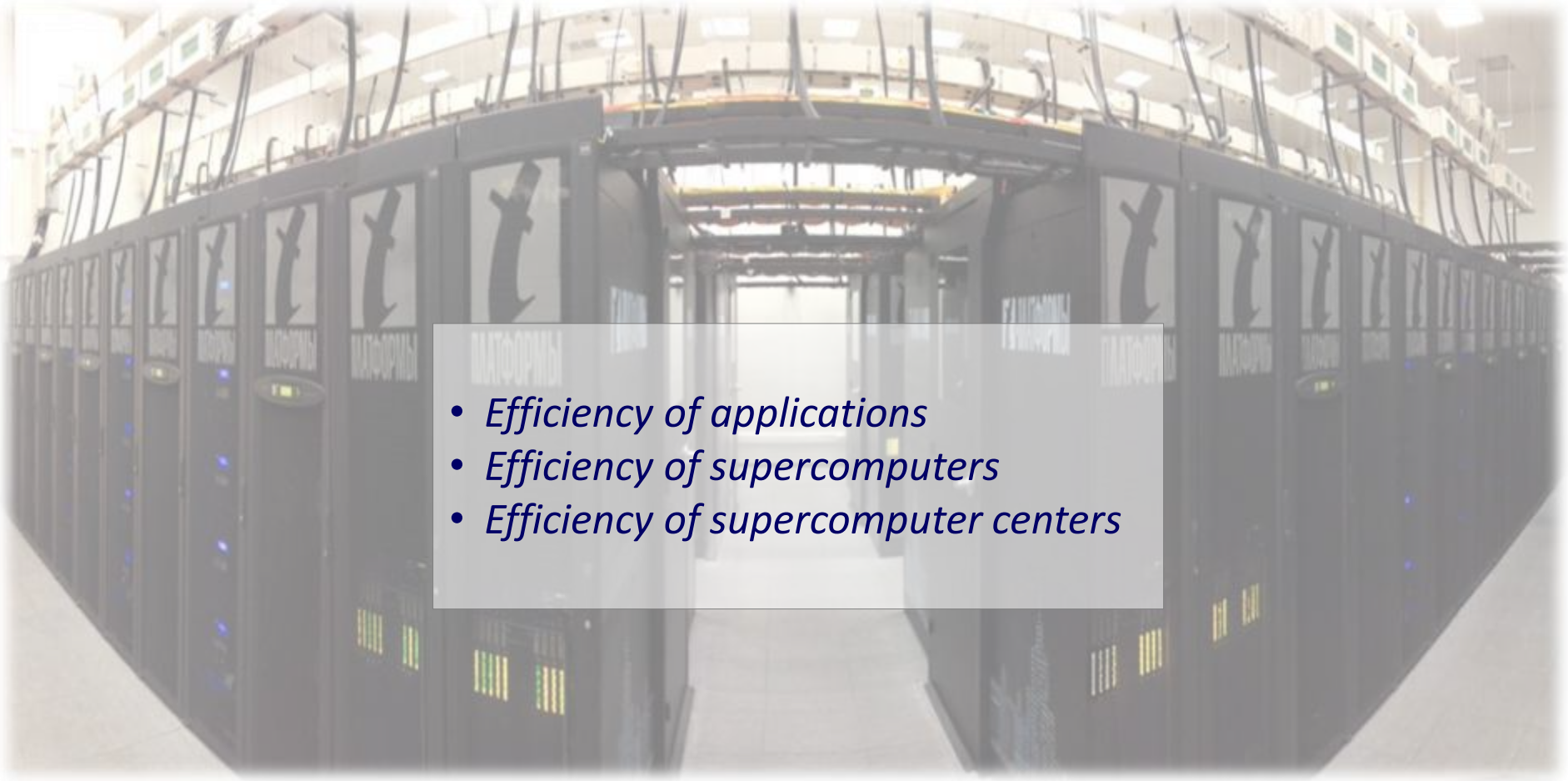


User and project activity, system load,
queue structure, hidden trends and dependencies…

Details and features:

- 1% of tasks consumes 33.2% of core-hours;
- 12% of tasks consumes 90% of core-hours;
- 11.78% of users (87 out of 739) have an average Loadavg < 1 but they consume 0.48% of core-hours;
- An anomaly case: a user has an average Loadavg = 0.07 and consumes 394839 core-hours;
- …

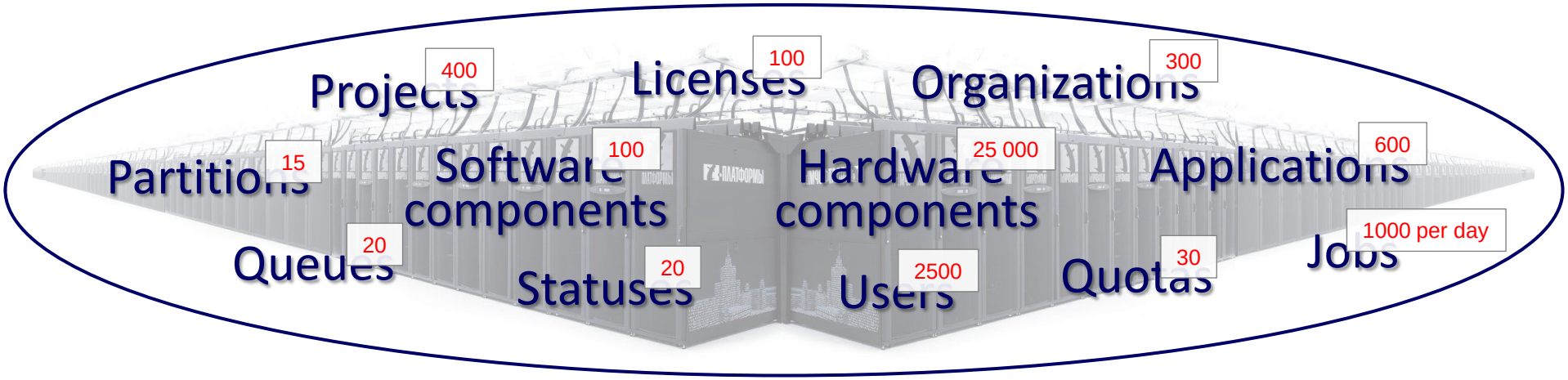
Potential of Supercomputer Centers



- *Efficiency of applications*
- *Efficiency of supercomputers*
- *Efficiency of supercomputer centers*

A few? Info on MSU "Lomonosov" Supercomputer :

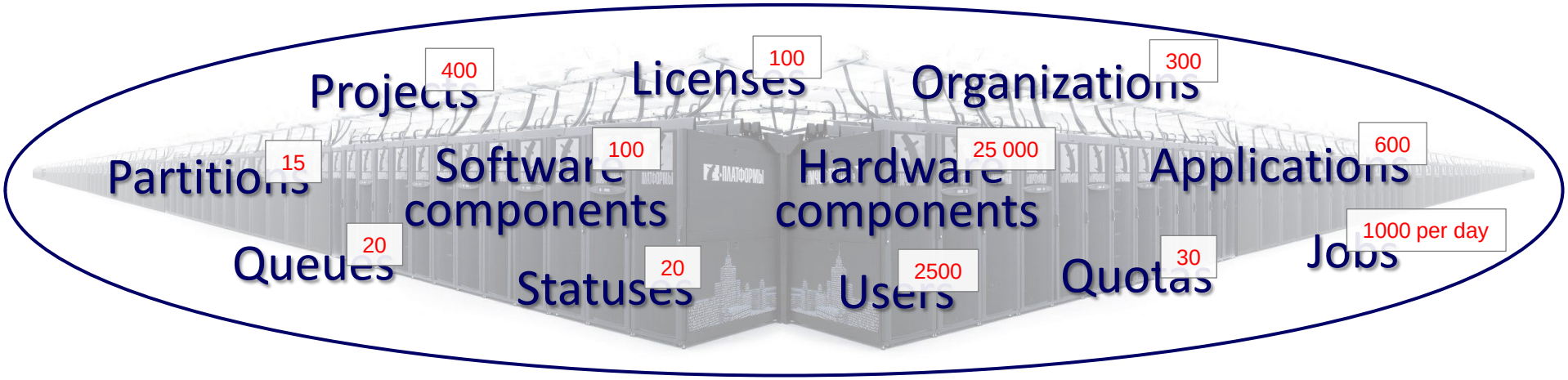
(1.7 Pflops, 6000 computing nodes, 12K CPUs, 2K GPUs…)



Current trend: all these numbers grow extremely fast !

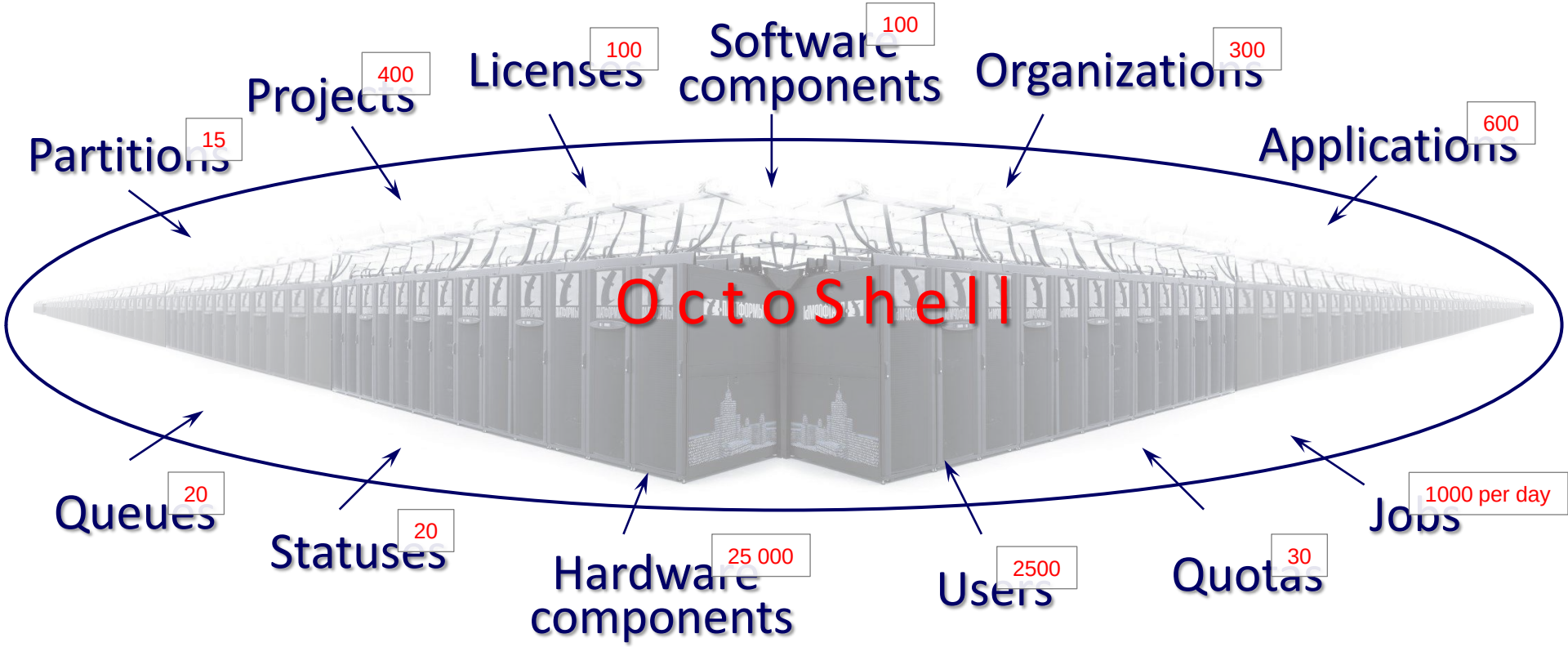
A few? Info on MSU "Lomonosov" Supercomputer :

(1.7 Pflops, 6000 computing nodes, 12K CPUs, 2K GPUs…)



OctoShell: overcoming complexity of supercomputing centers

(Supercomputing Co-Design Technologies and Tools)



OctoShell: overcoming complexity of supercomputing centers
(Supercomputing Co-Design Technologies and Tools)

100

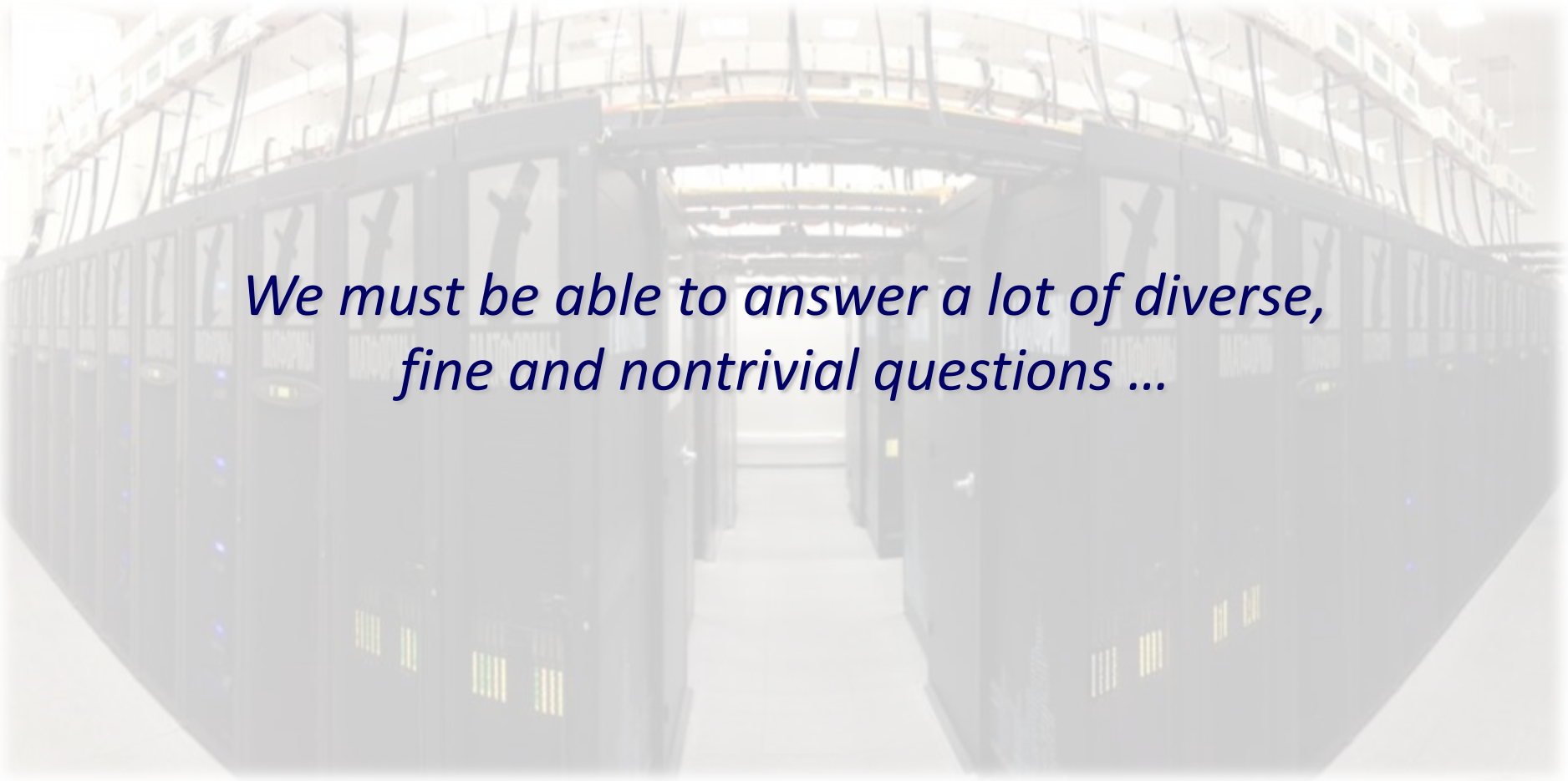
Licenses ? It's so easy…

Software licenses: necessary details on each package, library, tool…:

- title and version;
- license current status;
- contacts on license;
- contacts on technical support;
- license key, license activation code;
- license expiration date;
- support termination date;
- restrictions and limitations of the license;
- license update cost, support update cost;
- path to the package, home directory;
- description of installation and fine tuning procedures, basic parameters in use;
- description of testing and checking procedures after upgrades;
- path to reference guides and users manuals;
- person responsible for installation and upgrades;
- contacts of local experts on the software;
- users and projects who are eligible to use the software.

… For each of the 100 licenses to ensure efficiency of the supercomputing center…

Efficiency of Supercomputer Centers



We must be able to answer a lot of diverse, fine and nontrivial questions …

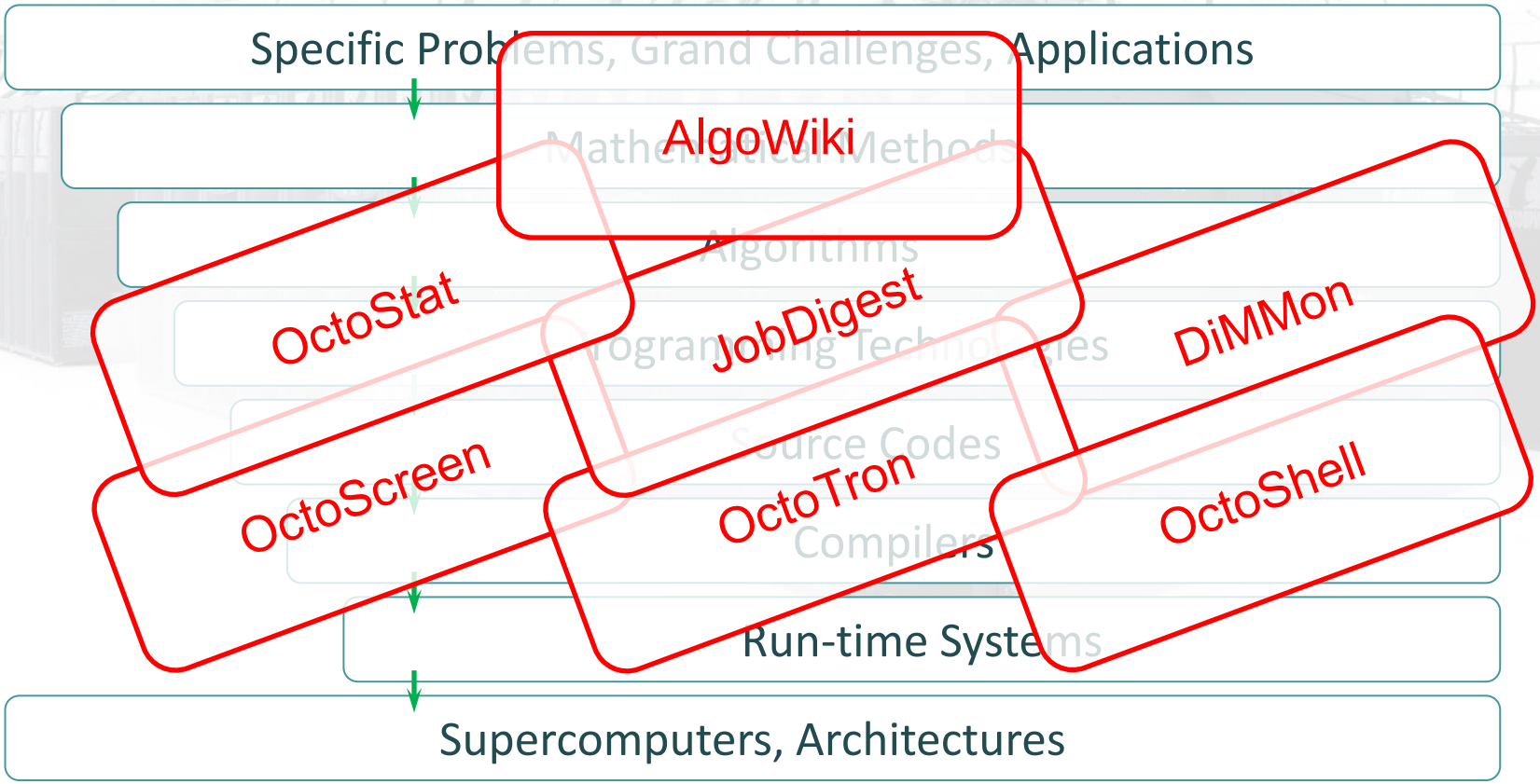
Questions on supercomputer center we should answer

(to have a complete picture on its efficiency all the time)

- *What is a distribution of CPUhours between software packages for the last year? (Should we spend money for the package X ?)*
- *What is average intensity of InfiniBand interconnect usage for different partitions? (How large should be Infiniband island in future configurations ?)*
- *How many nodes/cards/disks/cables fail every month ?*
- *How often has Infiniband resent packages for the last week ?*
- *How often does LoadAVG exceed number of cores on computing nodes ?*
- *What is an min/max/average level of cache misses for applications of a particular user ?*
- *What is a distribution of waiting time in queues ?*
- *How does LoadAVG behave in my application during execution ?*
- *Who are 5% of the most inefficient applications/users ? (regarding CPULoad or LoadAVG or cache misses or …)*
- *…*

Supercomputing Co-Design Technologies and Tools for the Reasonable Parallel Computing World

- Efficiency of applications
- Efficiency of supercomputers
- Efficiency of supercomputer centers



Six Generations of Parallel Computer Architectures

(Six parallel programming paradigms from the 70s up to now
or How often did we have to completely rewrite applications…)

Cray-1

70s - Loop Vectorization (innermost)

Cray Y-MP

80s - Loop Parallelization (outer) + Vectorization (innermost)

Intel Paragon

90s – Message Passing Interface (MPI)

Sun StarFire

mid 90s - OpenMP

"K supercomputer"

mid 2000s - MPI+OpenMP

MSU "Lomonosov"

2010s - CUDA, OpenCL, MPI+OpenMP+accelerators

Six Generations of Parallel Computer Architectures

(Six parallel programming paradigms from the 70s up to now
or How often did we have to completely rewrite applications…)

Parallel programming paradigms (from the 70s up to now):

70s - Loop Vectorization (innermost)
80s - Loop Parallelization (outer) + Vectorization (innermost)
90s - MPI
mid 90s - OpenMP
mid 2000s - MPI+OpenMP
2010s - CUDA, OpenCL, MPI+OpenMP+accelerators (GPU, Xeon Phi)
…

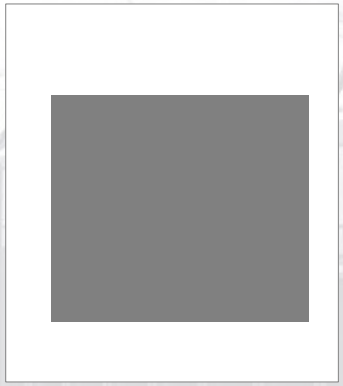
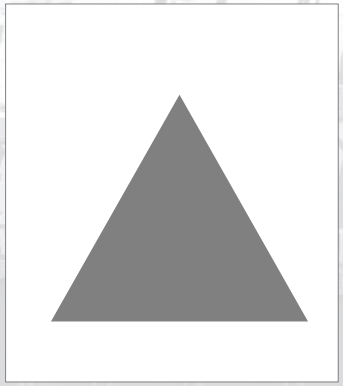
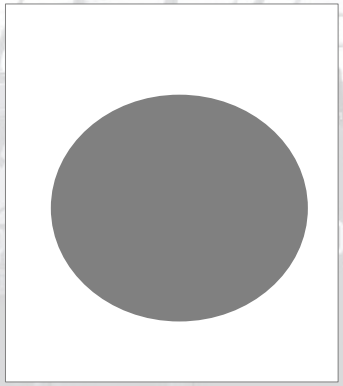
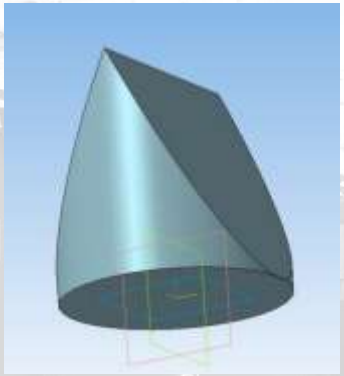
Changes in computer architectures do not change algorithms! But…

For each new generation of computing platforms we have to:
- Analyze algorithms to find the best way to match features of the algorithms to properties of the platform;
- Express the properties of algorithms we found to obtain efficient Implementation for the platform.

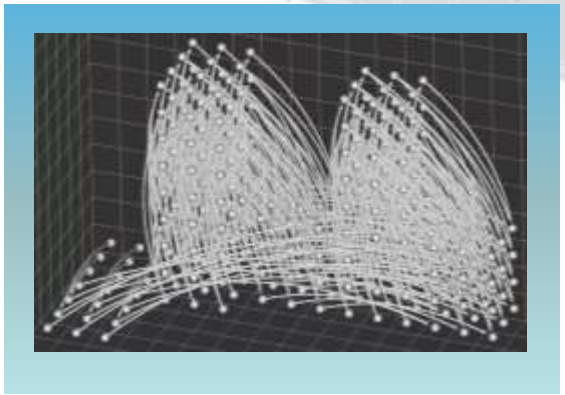
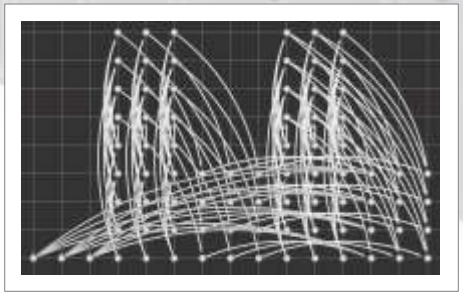
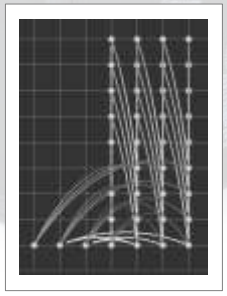
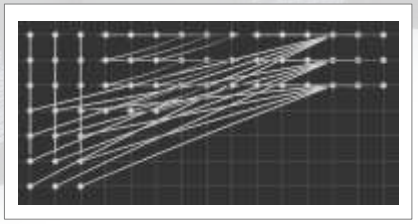
Changes in computer architectures do not change algorithms!

(Algorithms remain the same)

Are these figures different?

What are possible representations of this algorithm?

…

Six Generations of Parallel Computer Architectures

(Six parallel programming paradigms from the 70s up to now
or How often did we have to completely rewrite applications…)

Parallel programming paradigms (from the 70s up to now):

70s - Loop Vectorization (innermost)
80s - Loop Parallelization (outer) + Vectorization (innermost)
90s - MPI
mid 90s - OpenMP
mid 2000s - MPI+OpenMP
2010s - CUDA, OpenCL, MPI+OpenMP+accelerators (GPU, Xeon Phi)
…

Can we analyze algorithms once and for all ?

But…

Changes in computer architectures do not change algorithms!

For each new generation of computing platforms we have to:
- Analyze algorithms to find the best way to match features of the algorithms to properties of the platform;
- Express the properties of algorithms we found to obtain efficient Implementation for the platform.

What are key properties of an algorithm we need to analyze and describe to create an efficient implementation in the future?

What properties are important?

Unified (complete, universal) description of an algorithm: What do we need to take into account ?

Too many “simple” questions…

Properties and Structures of Algorithms…

Yes, we can! **AlgoWiki**

http://AlgoWiki-Project.org/

Can we analyze algorithms ***once and for all*** *?*

- Analyze algorithms to find the best way to match features of the algorithms to properties of the platform;

Description of Algorithms

(What should be included in this description?)

Algorithms: Theoretical Part

(machine-independent properties,

"Once and for all")

Algorithms: Implementation Issues

Information Graph

Determinacy

Computational kernel

Locality of computations

Macrostructure

Data locality

Mathematical description

Properties and Features

Efficiency

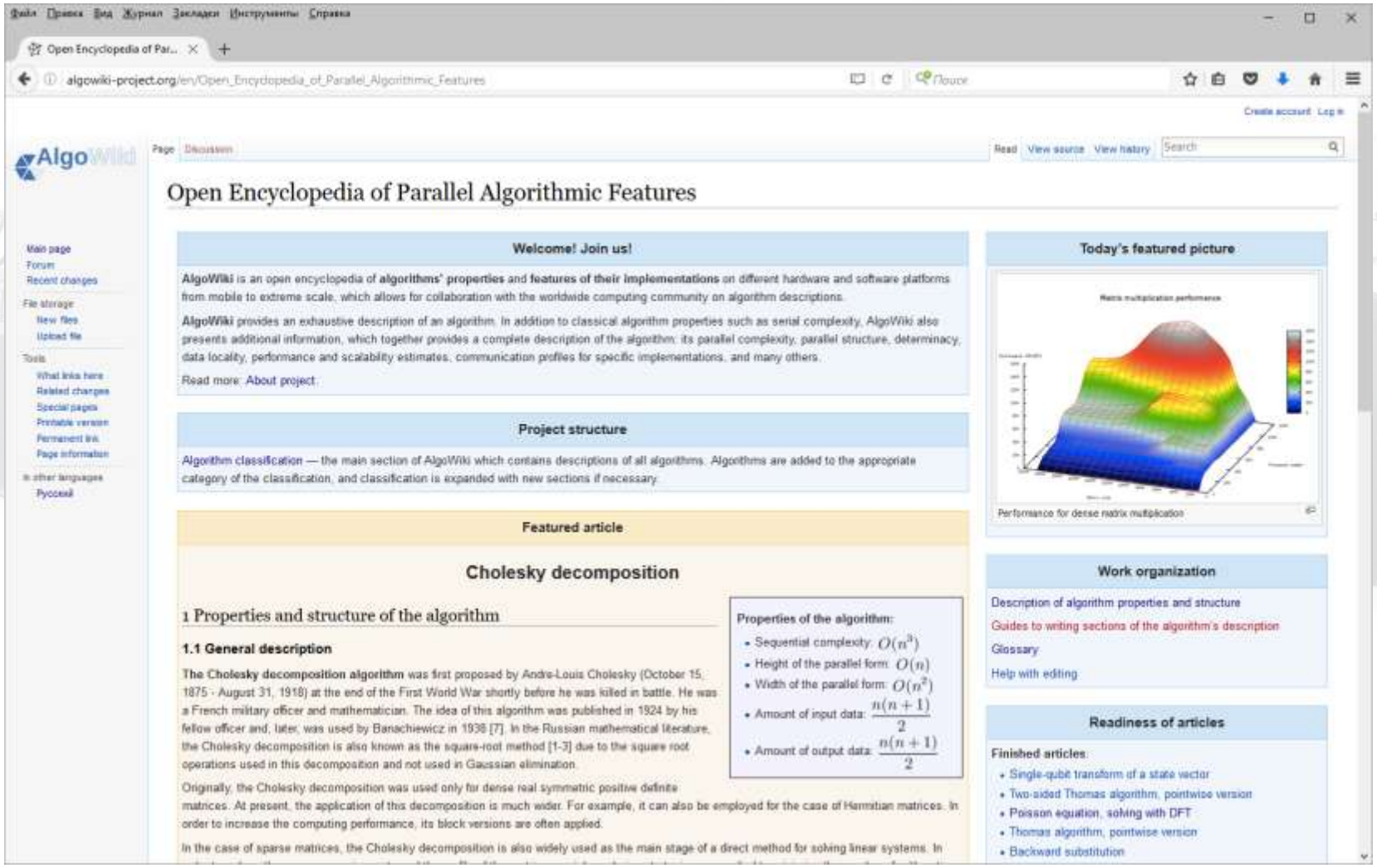
Computational intensity

Serial Complexity

Resource of Parallelism

Input / Output data

Supercomputing Co-Design Technologies and Tools for the Reasonable Parallel Computing World



AlgoWiki *http://AlgoWiki-Project.org*

Supercomputing Consortium of Russian Universities
6th Summer Supercomputing Academy

Supercomputing Co-Design Technology

Prof. Vladimir Voevodin
Deputy Director, Research Computing Center, MSU
Head of Department on Supercomputers and Quantum Informatics, CMC, MSU

voevodin@parallel.ru

June 23rd, 2017,
Moscow – Saint-Petersburg – Arkhangelsk – Ekaterinburg