Towards multi-parameter resource selection for HPC platforms

Master Research Thesis

Dineshkumar RAJAGOPAL
(PDES - MoSIG)

advised by

Yiannis GEORGIOU

Big Data and Security(BDS) lab, BULL-SAS

September 1, 2015

HPC System Software Stack

1. Supercomputer is a **HPC Cluster**
2. **Usage :** Computationally intensive tasks in **Scientific Experiments (Quantum mechanics, Weather Prediction, etc)**

System Software:
Operating System, Runtime System, Resource Management, I/O System, Interfacing to External Environments

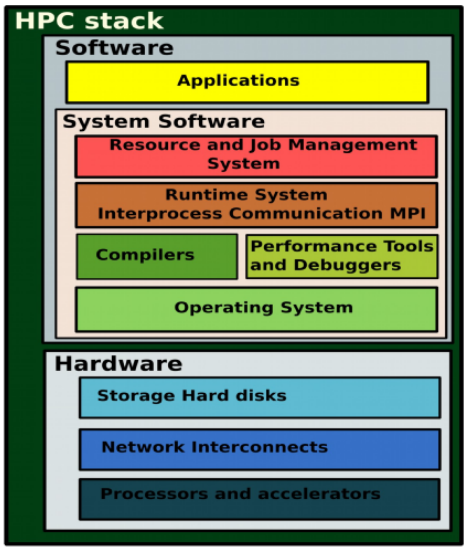


Image credit: Yiannis georgiou,BULL

What is RJMS?

1. The goal of a **Resource and Job Management System (RJMS)** is to satisfy users demands for computation and **assign resources to user jobs** with an efficient manner
2. **RJMS** knows the complete details about the Jobs and Resources of HPC system

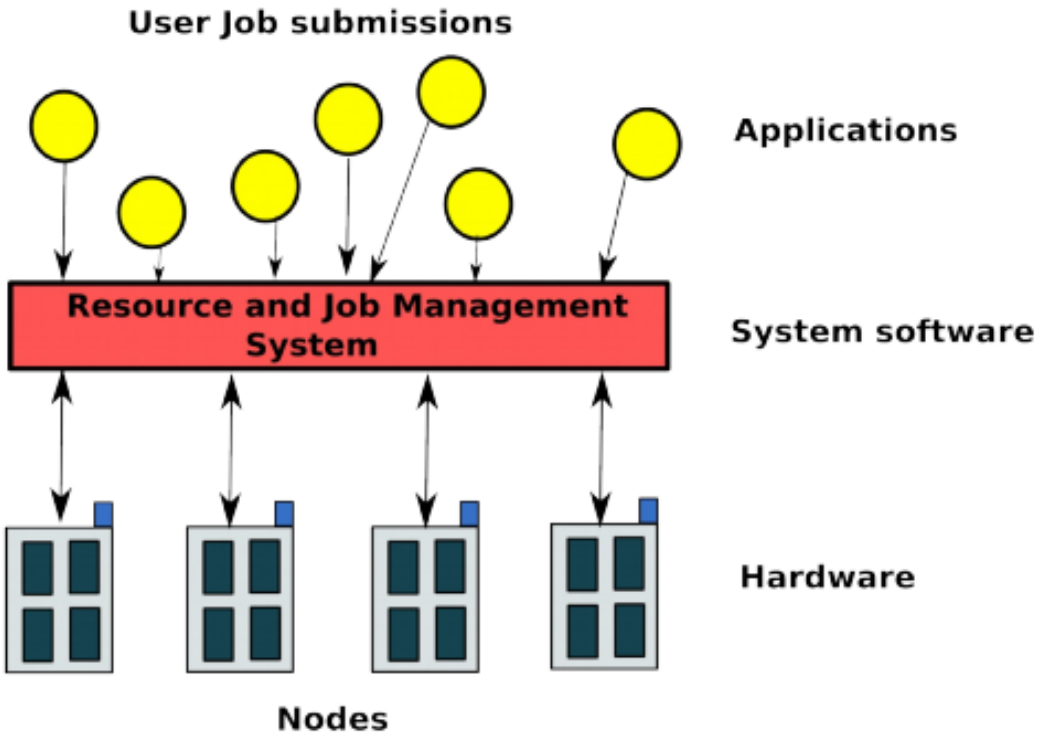


Image credit: Yiannis

Motivation of Resource Selection

1. Resource selection is an internal operation of scheduling
2. Due to the the evolution of HPC platform and internal nodes architecture, Resource Management is dynamic and complex
3. Improper resource management hides resources information to lose global view of resources
4. Power wall problem and the increasing number of nodes in HPC systems, energy efficiency is the important criteria
5. Multi-parameter resource selection to satisfy different criteria to allocate resources perfectly

Related Works

Feature	OAR	SLURM	FLUX
Programming Tools	Perl, MySQL	C, autoconf	C, autoconf
Purpose	Scalable & Flexible	Scalable, Flexible & Performance	Scalable, Flexible & Distributed resource & job management
Resource Management	Flat Hierarchical & managed in the Databse	Linear & Managed in its own data structure(bitmap & list)	Dynamic & Flat hierarchical resource management framework
Resource Selection	Best-fit for Intel cluster architecture, SQL querry to perform selection	Best-fit for different cluster architecture, custom implementation of algorithm	Next generation RJMS, only framework core functionality developed
Topology Awareness	Yes	Yes	Not Yet implemented
Internal Resource Consumption	Yes	Yes	Not Yet implemented

SLURM Architecture

1. SLURM is an **open source RJMS** for Supercomputer

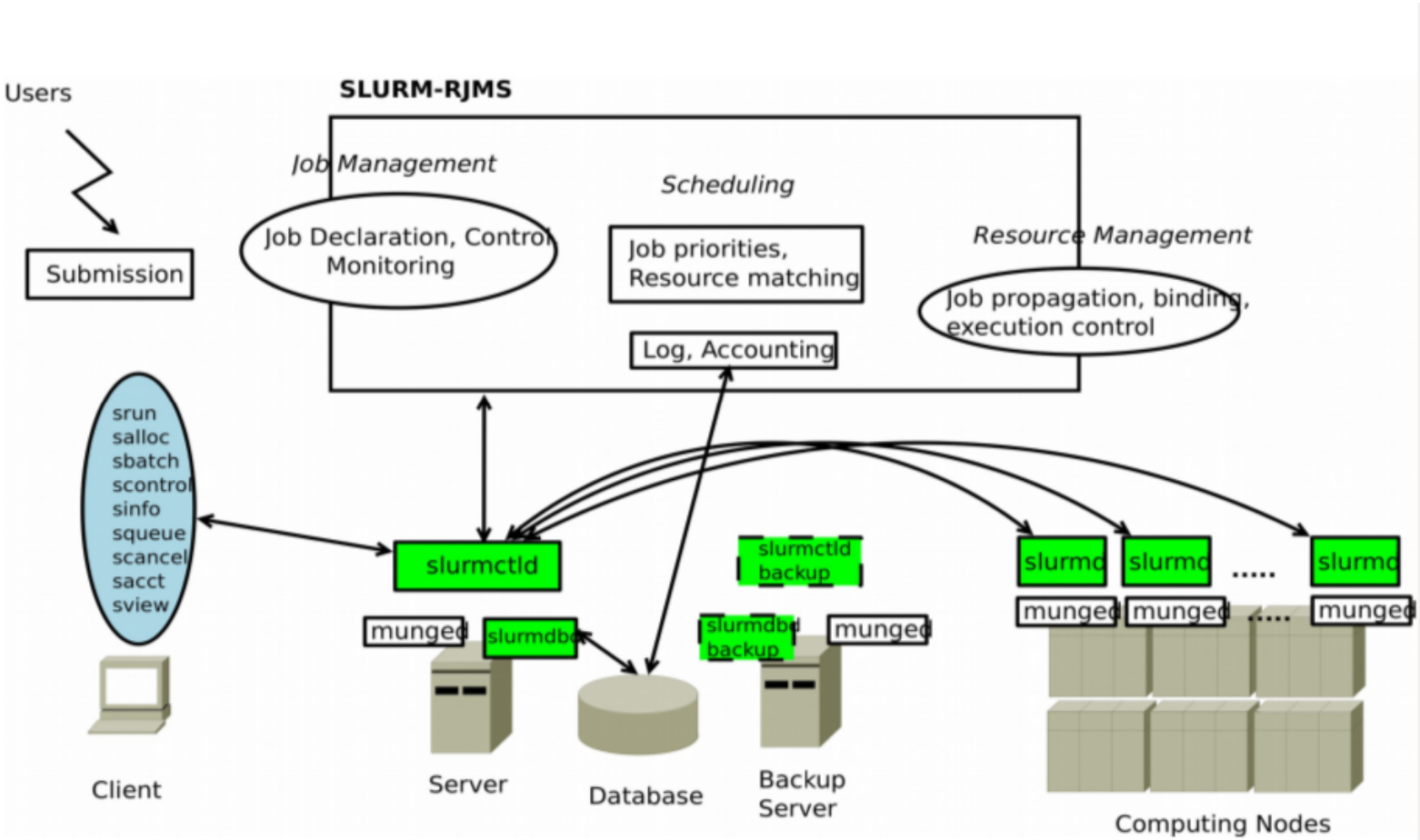
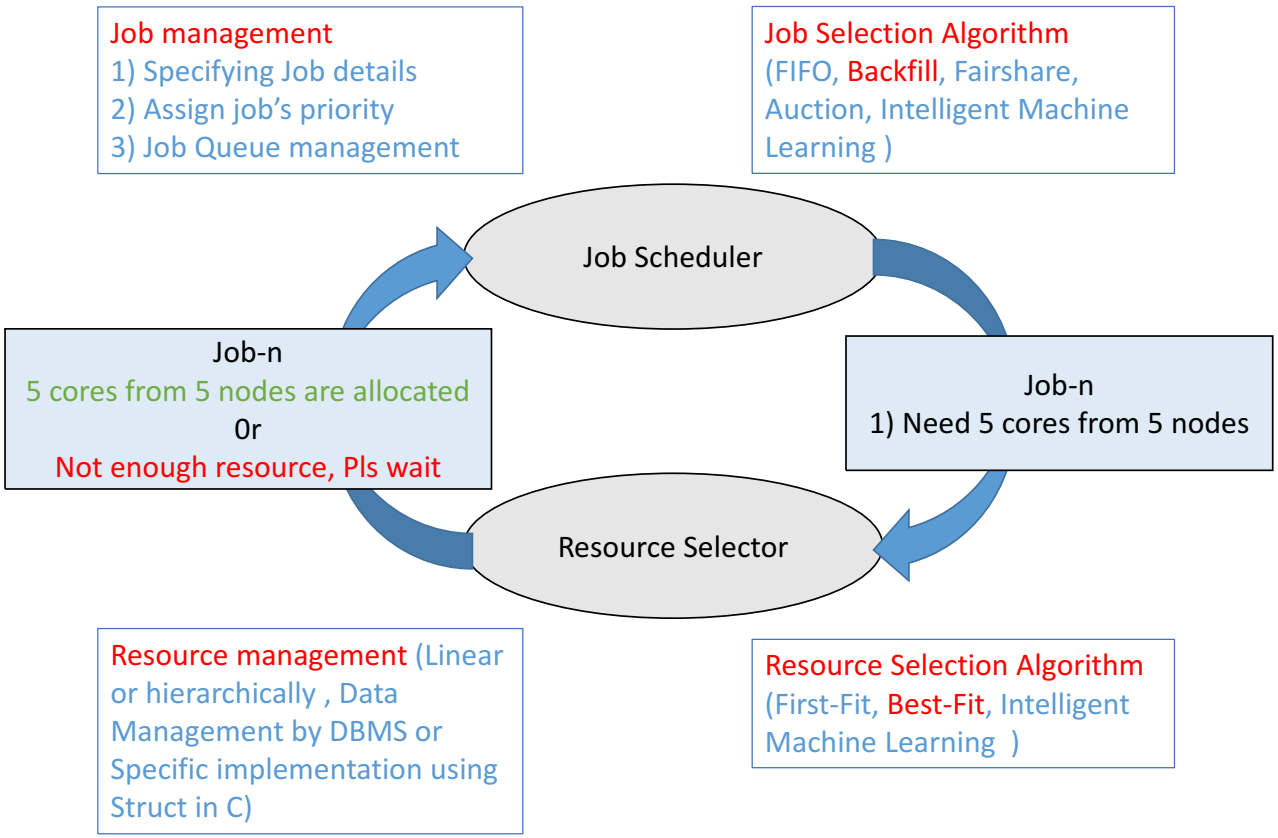


Image credit: Yiannis georgiou,BULL

Batch Scheduling

1. **Scheduling behaviour** depends on the Job scheduler and Resource selector behaviour
2. **Resource Selection** is an internal operation of scheduling



Resource Selection Cycle

1. Select and Topology plugin work together to allocate **topology aware resources**
2. Topology plugin has information of Switch and Node relationship in bitmap, so comparison of two nodes list is very fast and scalable with less memory foot-print

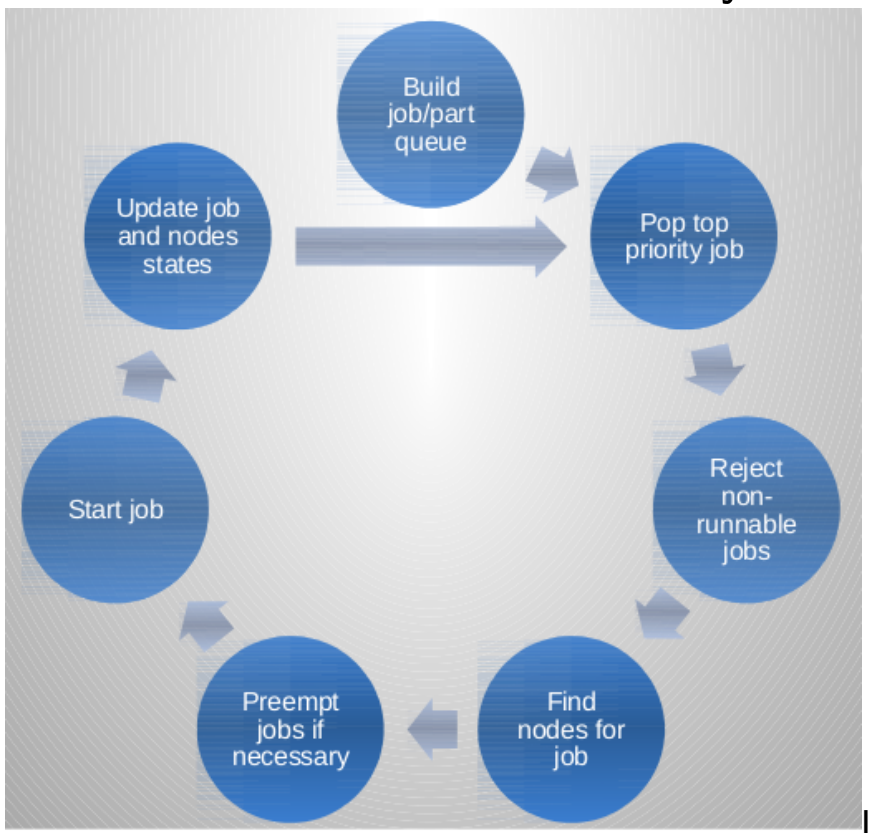


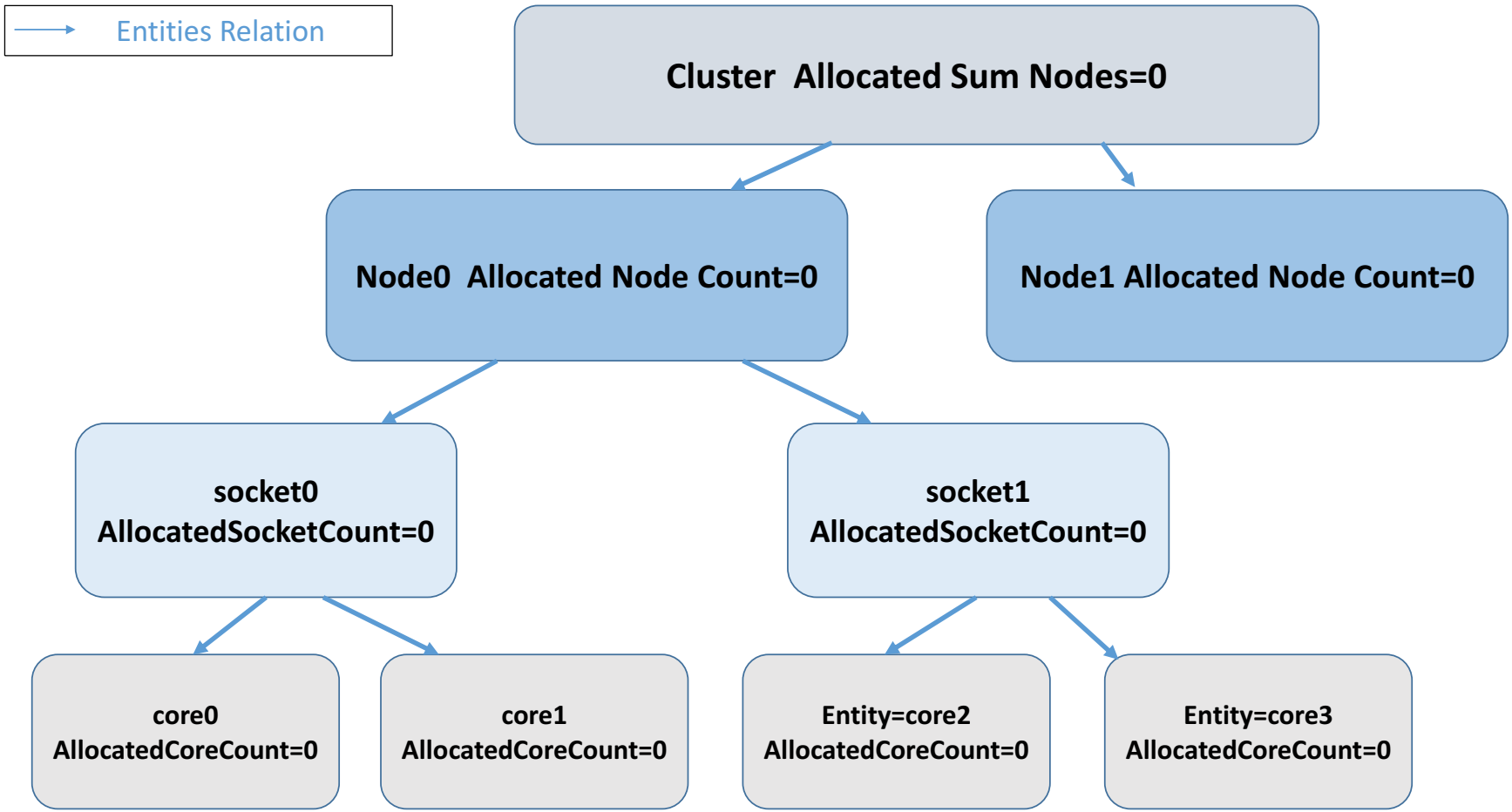
Image credit: Don Lipari, LLNL

LAYOUTS Framework

1. LAYOUTS is the new resource management framework in **SLURM v15.08**
2. Any type of entities can be manageable
3. Entities relation is tree (Inspiration of OAR resource management)
4. Entity attribute is called in LAYOUTS key, stored in the **hash key-value** format
5. Keeping consistency among attributes values across entities based on keys inheritance relations
6. Entities information is available in the different level of resource hierarchy to reveal hidden information of resources

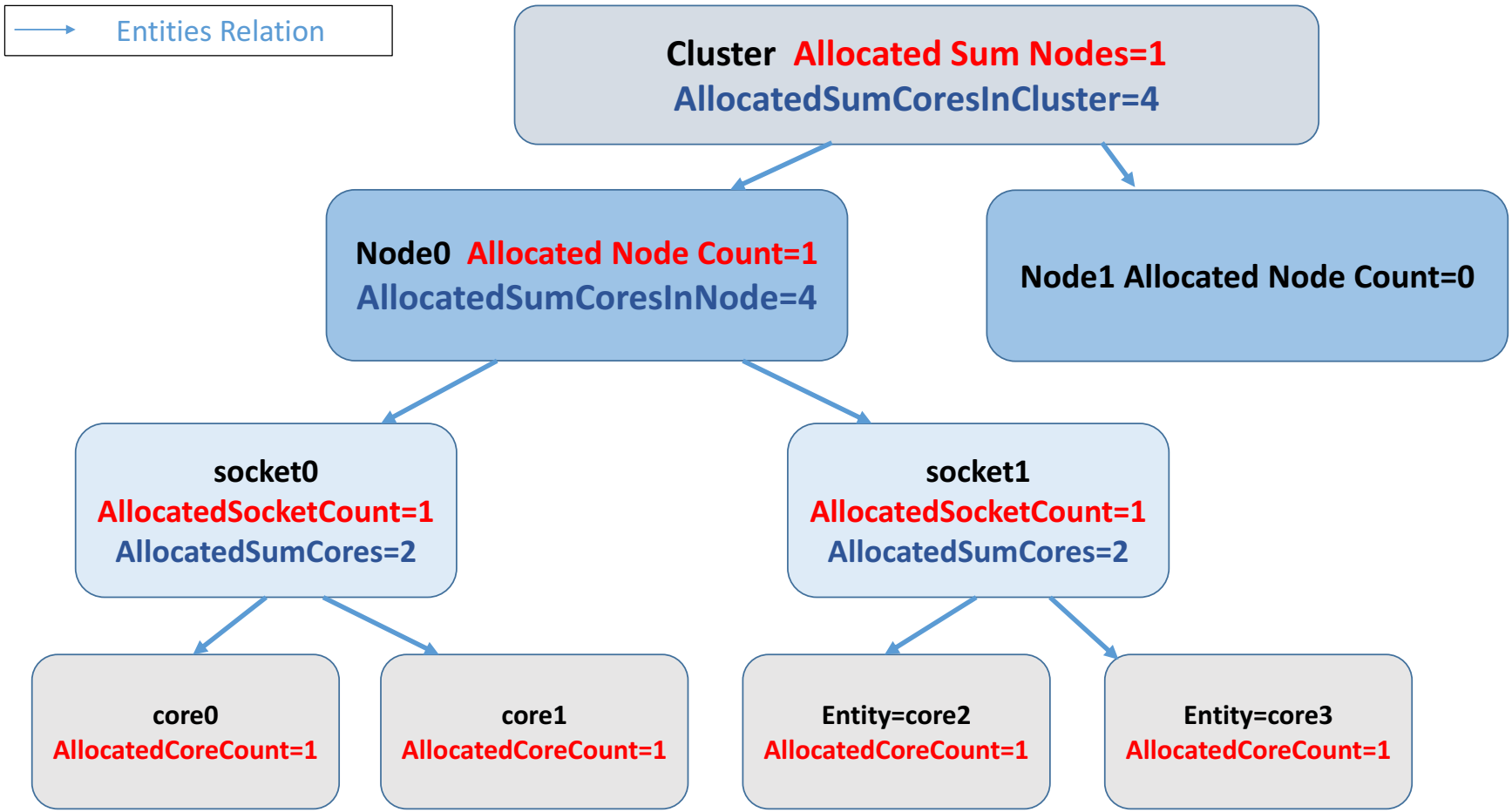
LAYOUTS Entity Keys and Key Relation

1. If all the cores are allocated in the node, then the node is allocated



LAYOUTS Entity Keys and Key Relation(ctd...)

1. If all the cores are allocated in the node, then the node is allocated



LAYOUTS Basic APIs

1. **layouts_entity_get_kv()** - **get** key value
2. **layouts_entity_set_kv()** - **set** key value
3. **layouts_entity_pull_get_kv()** - update key relation value and **get** key value
4. **layouts_entity_set_push_kv()** - **set** key value and update key relation

Cons_res_layout Implementation

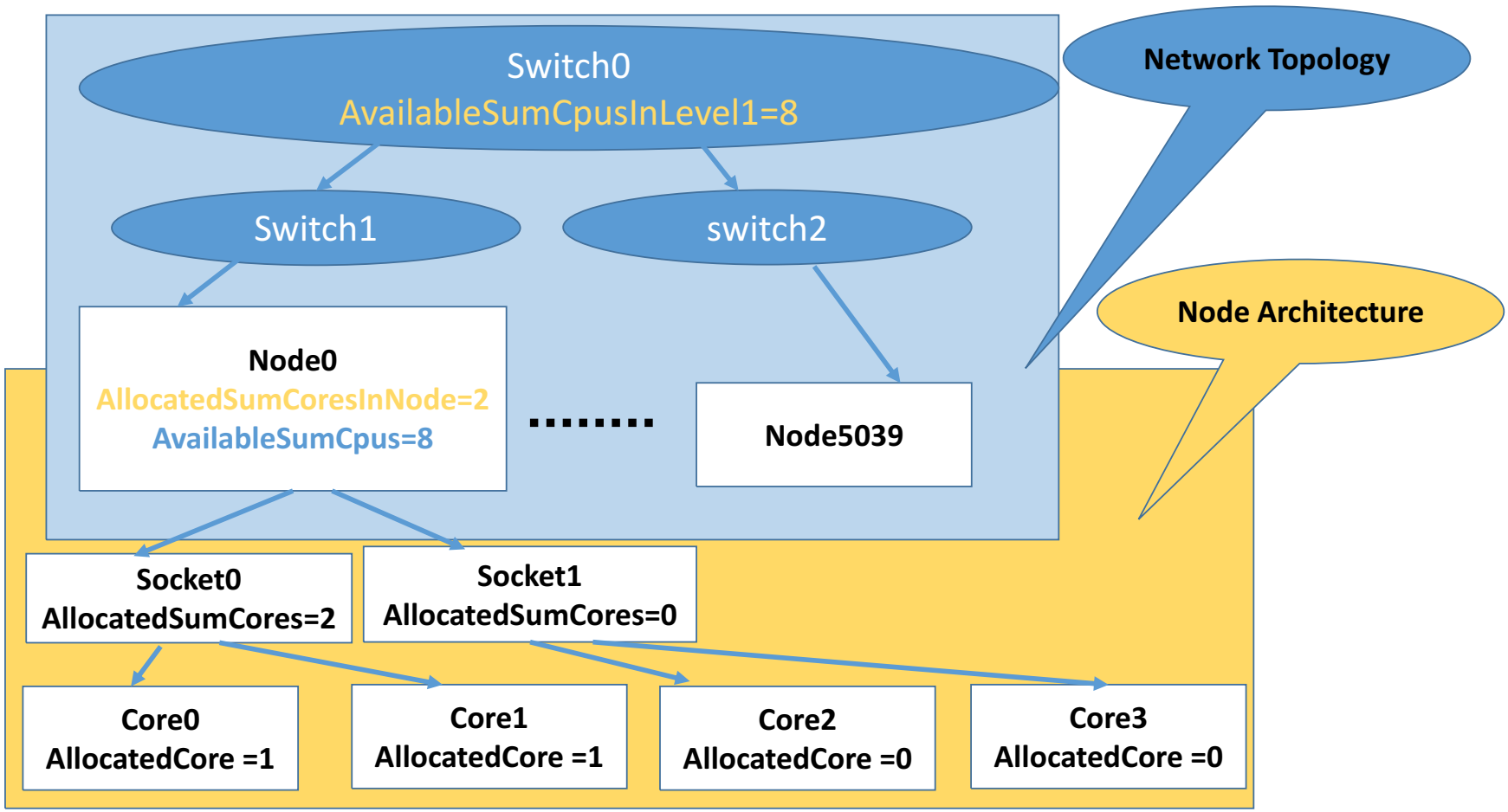
1. Cons_res is the consumable resource selection plugin
   - **Best-fit** to select **minimum satisfiable resources than maximum satisfiable resources**
   - **Topology aware** to increase the user application performance
   - **Cons_res** consumes internal resources of nodes(cores, memory, etc)
   - **Algorithm 1** in the **section 4.1** of the **report** discussed the algorithm step by step
   - **Cons_res** used **list and bitmap** to keep resource information

Cons_res_layout Implementation

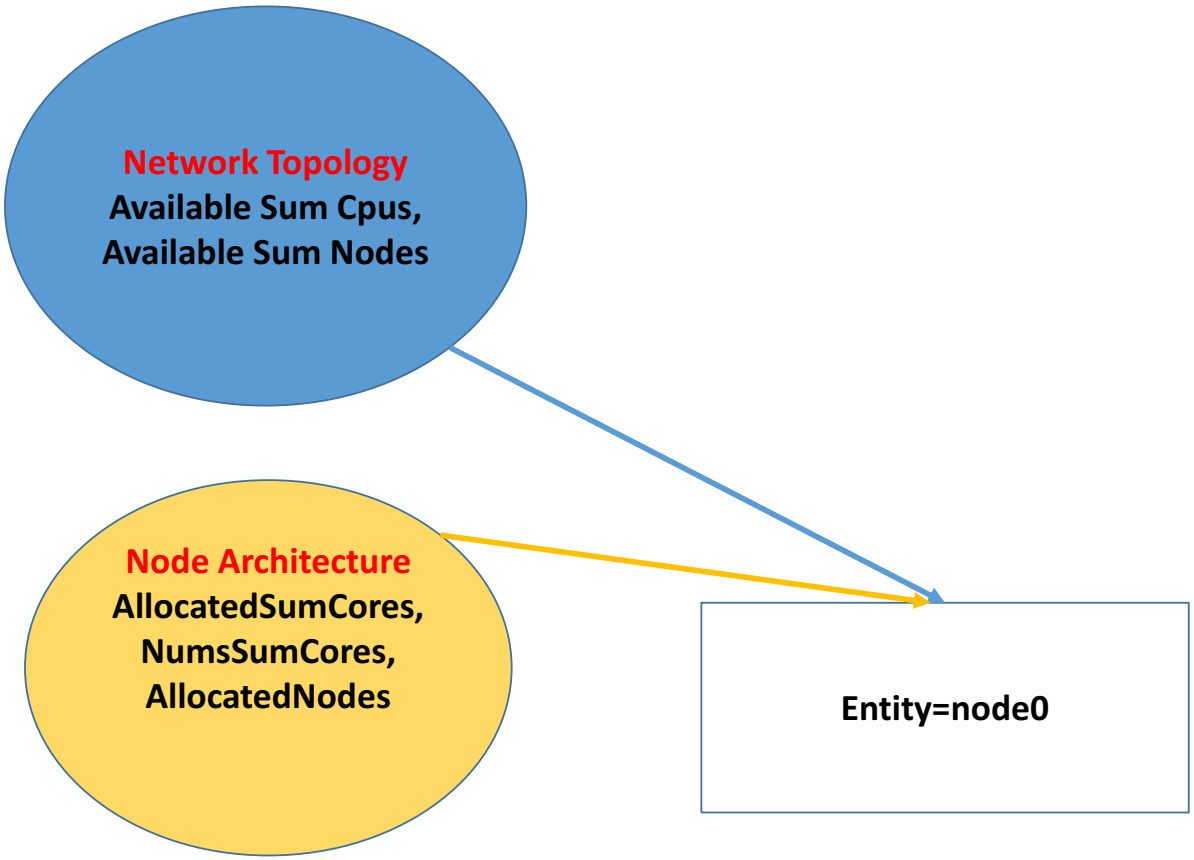
1. Cons_res is the consumable resource selection plugin
   - **Best-fit** to select **minimum satisfiable resources than maximum satisfiable resources**
   - **Topology aware** to increase the user application performance
   - **Cons_res** consumes internal resources of nodes(cores, memory, etc)
   - **Algorithm 1** in the **section 4.1** of the **report** discussed the algorithm step by step
   - **Cons_res** used **list and bitmap** to keep resource information

2. Cons_res_layout is the **new** consumable resource selection plugin based on LAYOUTS
   - Naive implementation of **cons_res_layout** plugin performance was **25 times slower**
   - **Second version** of the cons_res_layout code used a combination of **bitmaps, layouts** to reach the performance of default cons_res code

Layouts Interrelation



Entity Global View

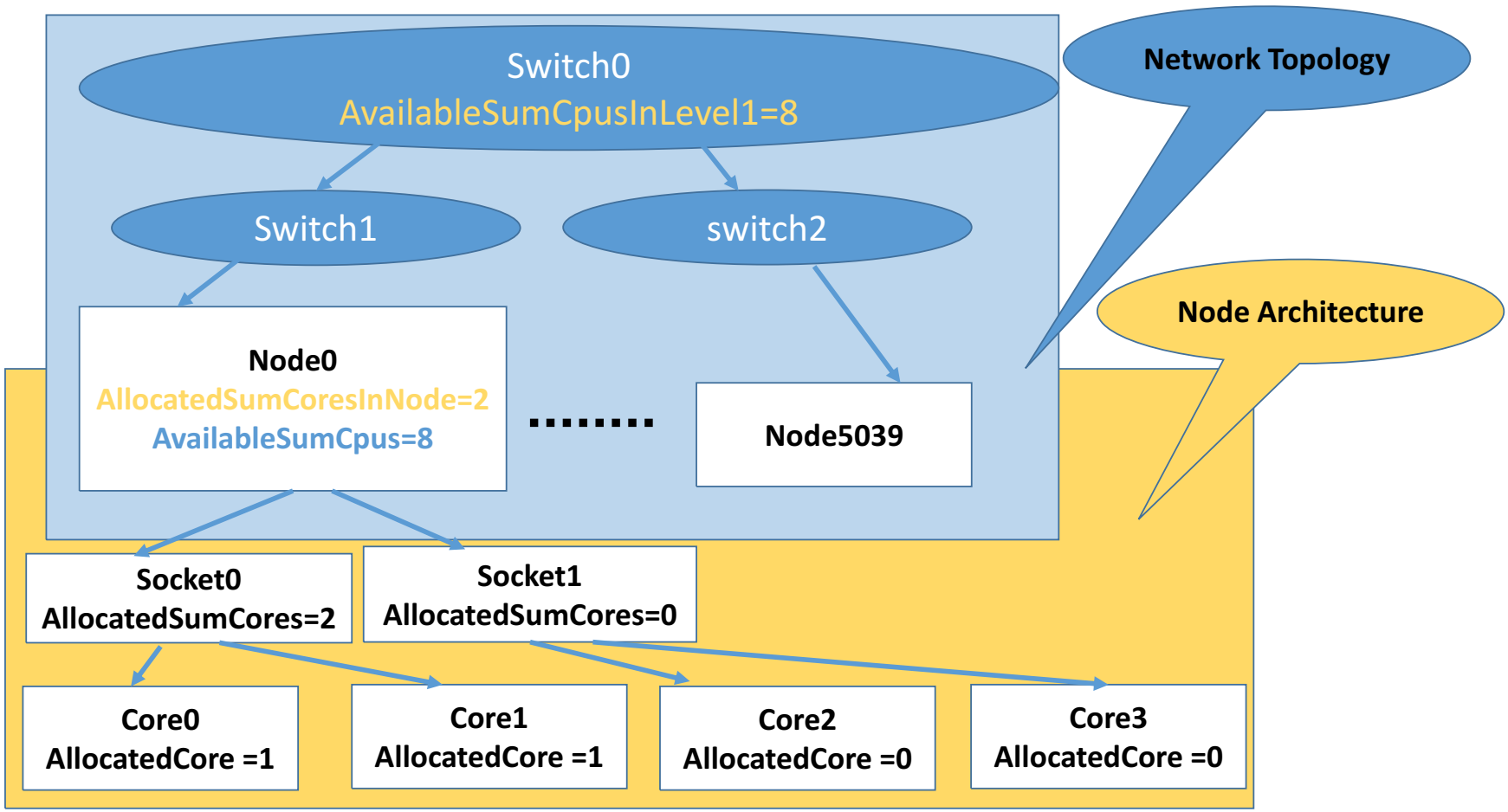


Multi-parameter Resource Selection Algorithm

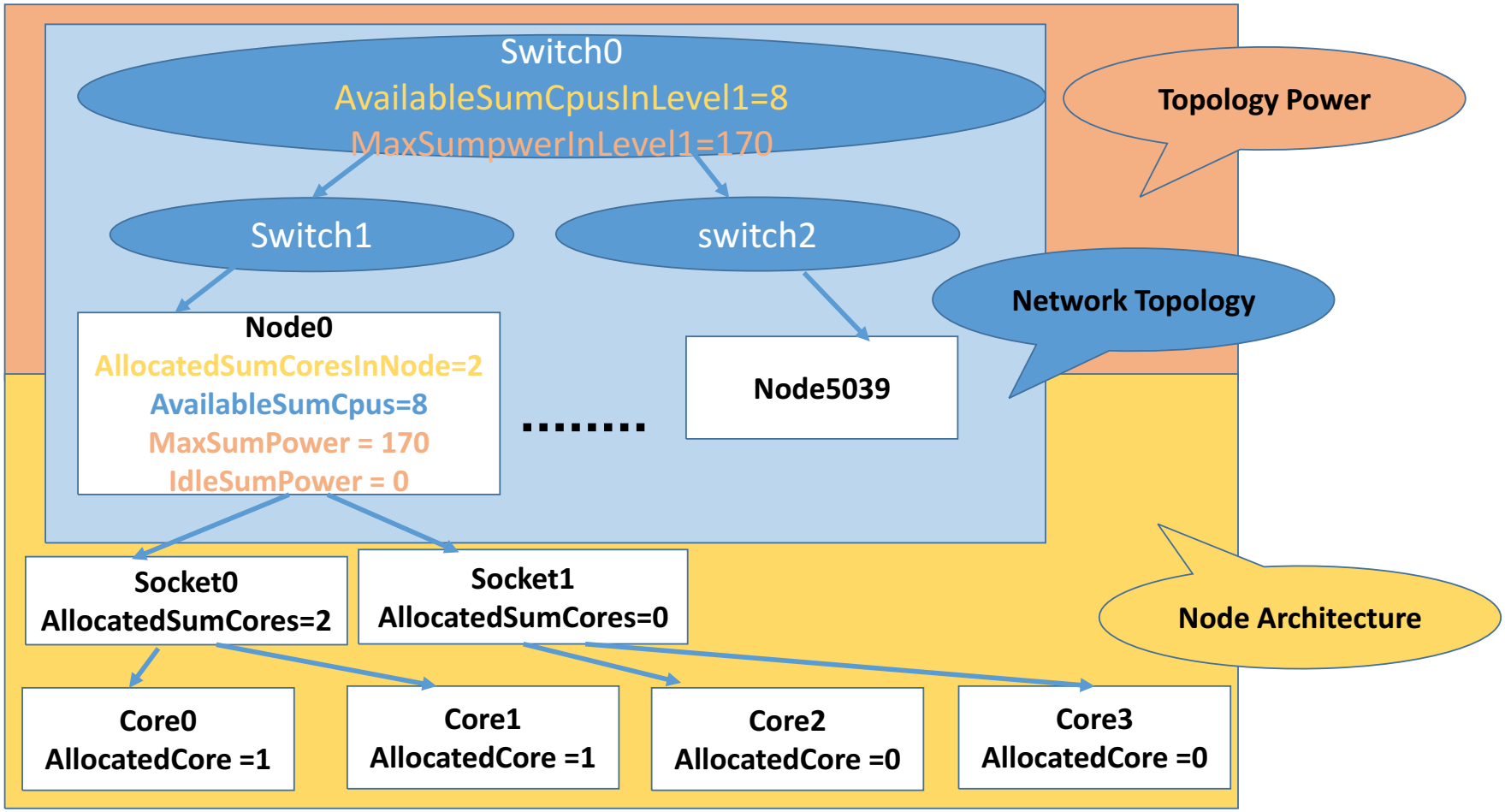
1. **Bestfit energy efficiency** to reduce energy consumption in the **Heterogeneous power consuming cluster**
   - Minimum power consuming allocated nodes
   - Already allocated nodes than idle nodes
   - Minimum power consuming idle nodes
2. **Topology aware** to increase the user application performance
3. **Algorithm 2** in the **section 4.2** of the **report** discussed the algorithm step by step

4. **Advantage:** Multi-parameter(Cons_res_power) resource selection supports **user's performance** and **server's energy** criterias

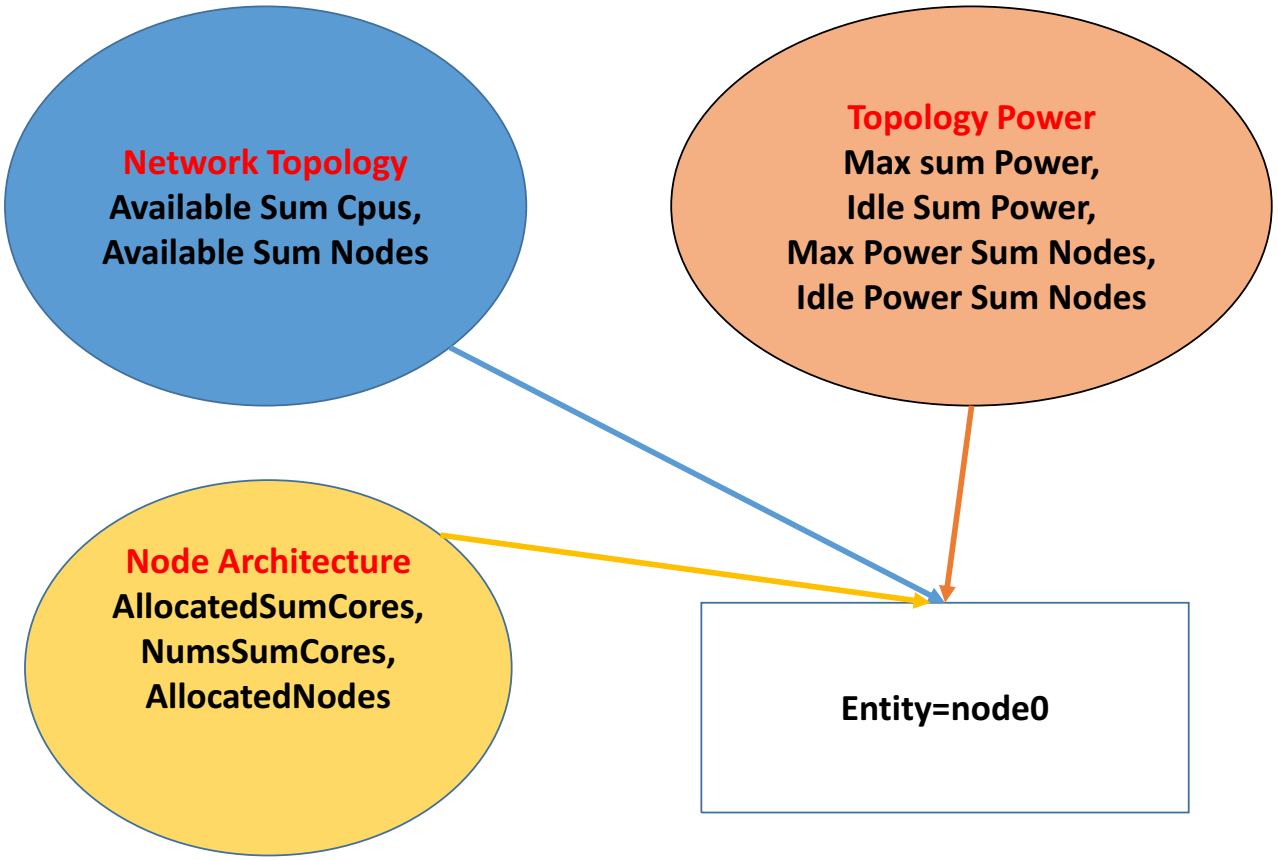
Layouts Interrelation



Layouts Interrelation(Ctd...)



Entity Global View

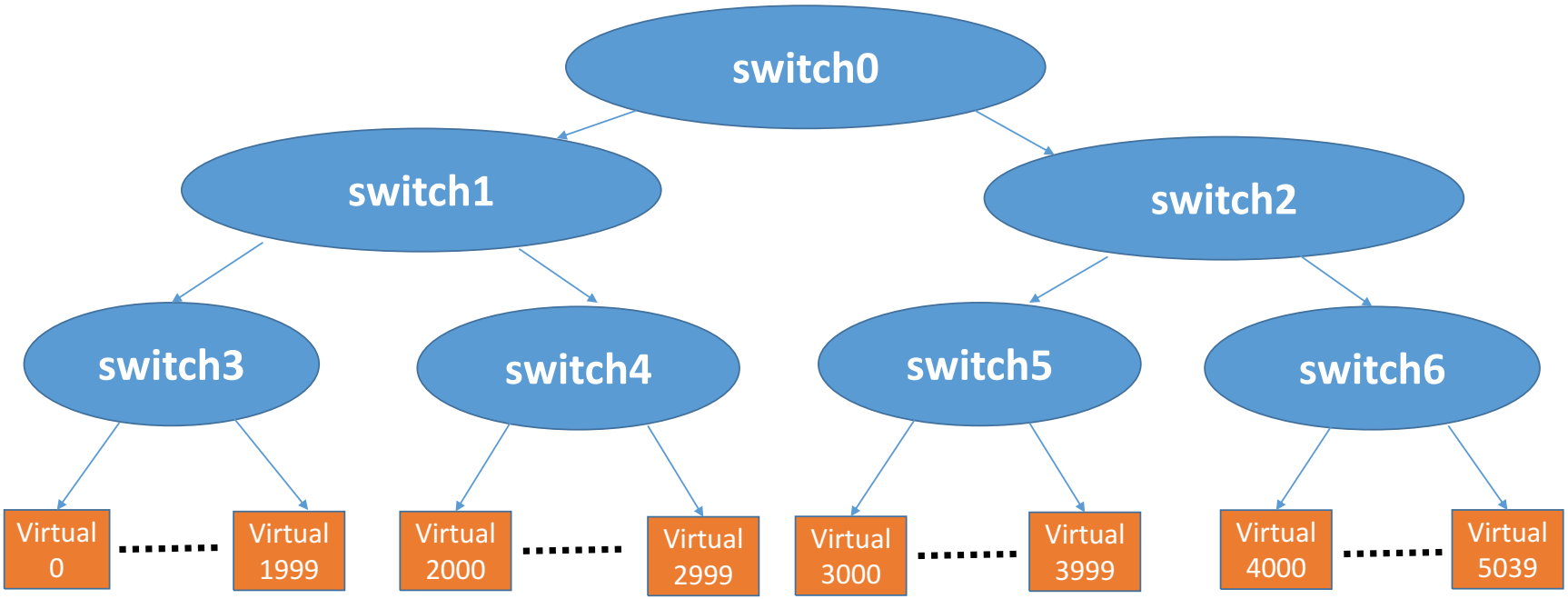


Experiment Environment

1. Emulate real HPC environment using –enable-multiple-slurmd option
2. Synthetic workload of Enhanced System Performance(ESP) Benchmark
3. Use sleep, hostname like simple application
4. Standard Workload Format(SWF) to store workload
5. BULL CUZCO cluster 17 nodes to emulate 5040 nodes HPC cluster
6. Each node configured as 2 sockets, 16 cores and 32GB of memory, more details are in Appendix C.1

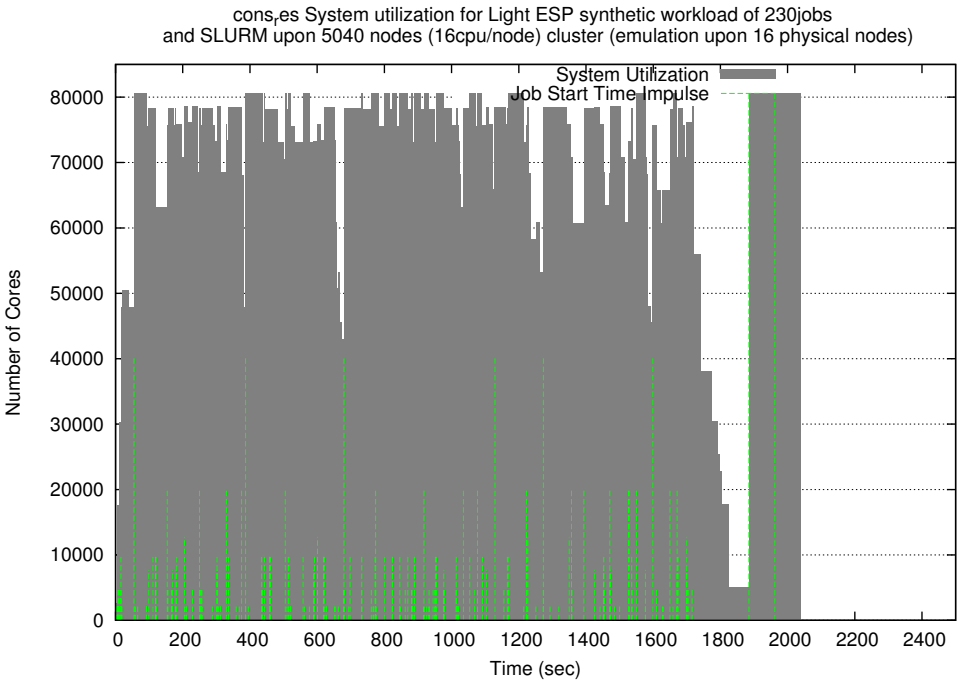
Topology Experiment Environment

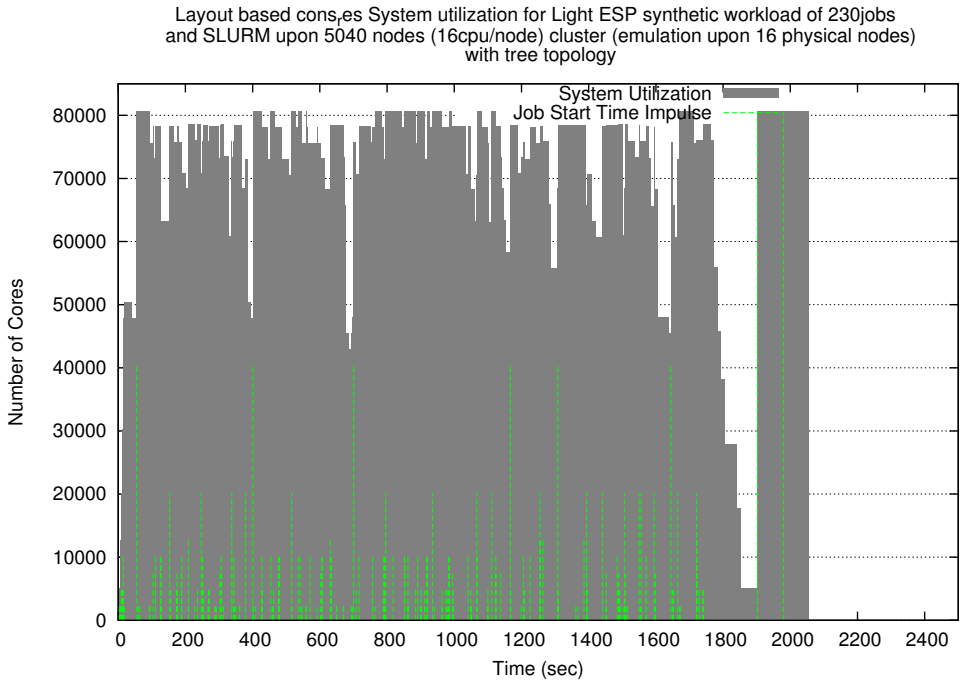
1. Homogeneous cluster environment
2. Simple tree topology to have 4 leaf switches and 3 levels



Cons_res vs Cons_res_layout system utilization

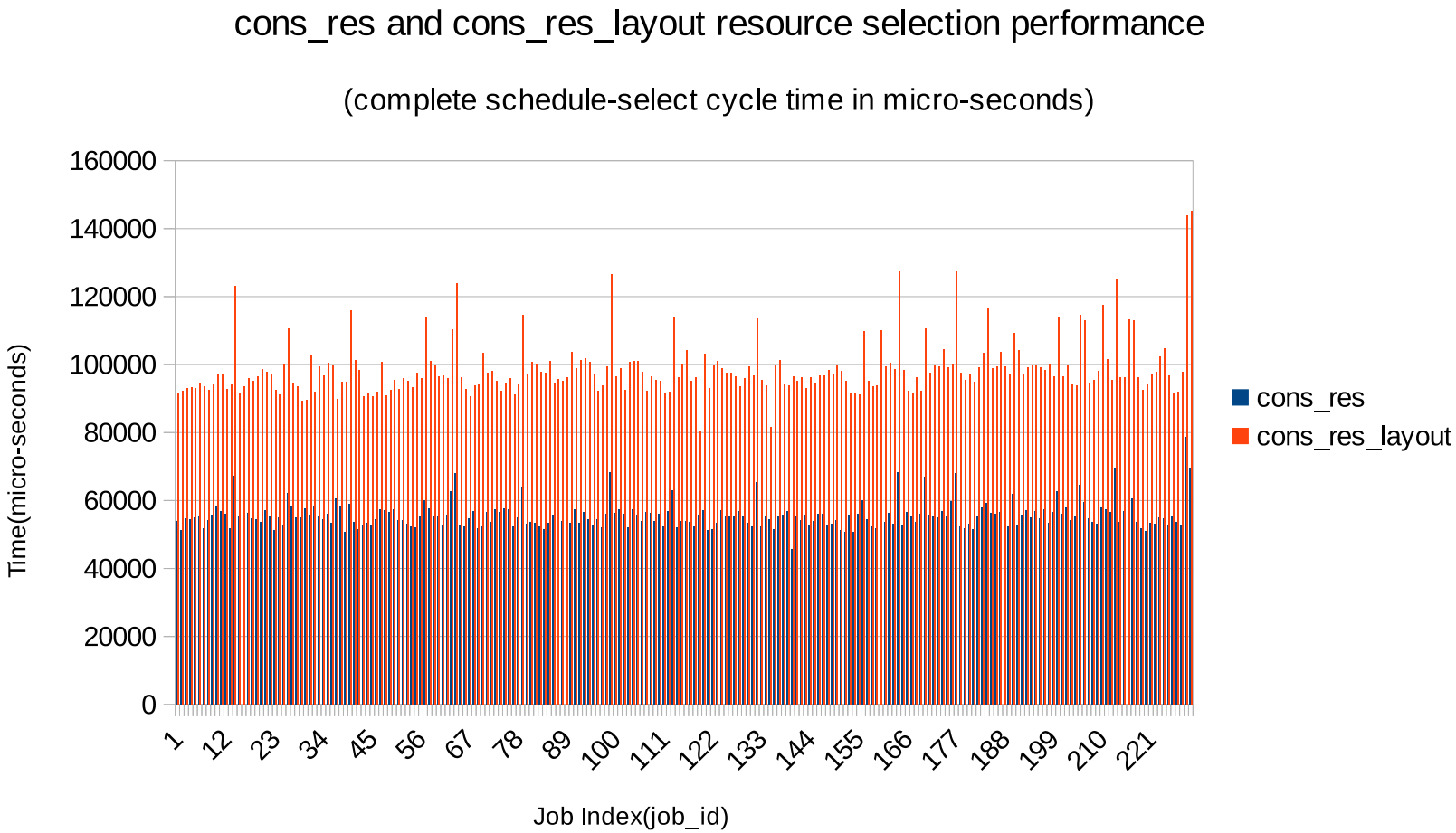
1. System utilization is almost same, because of following same policy



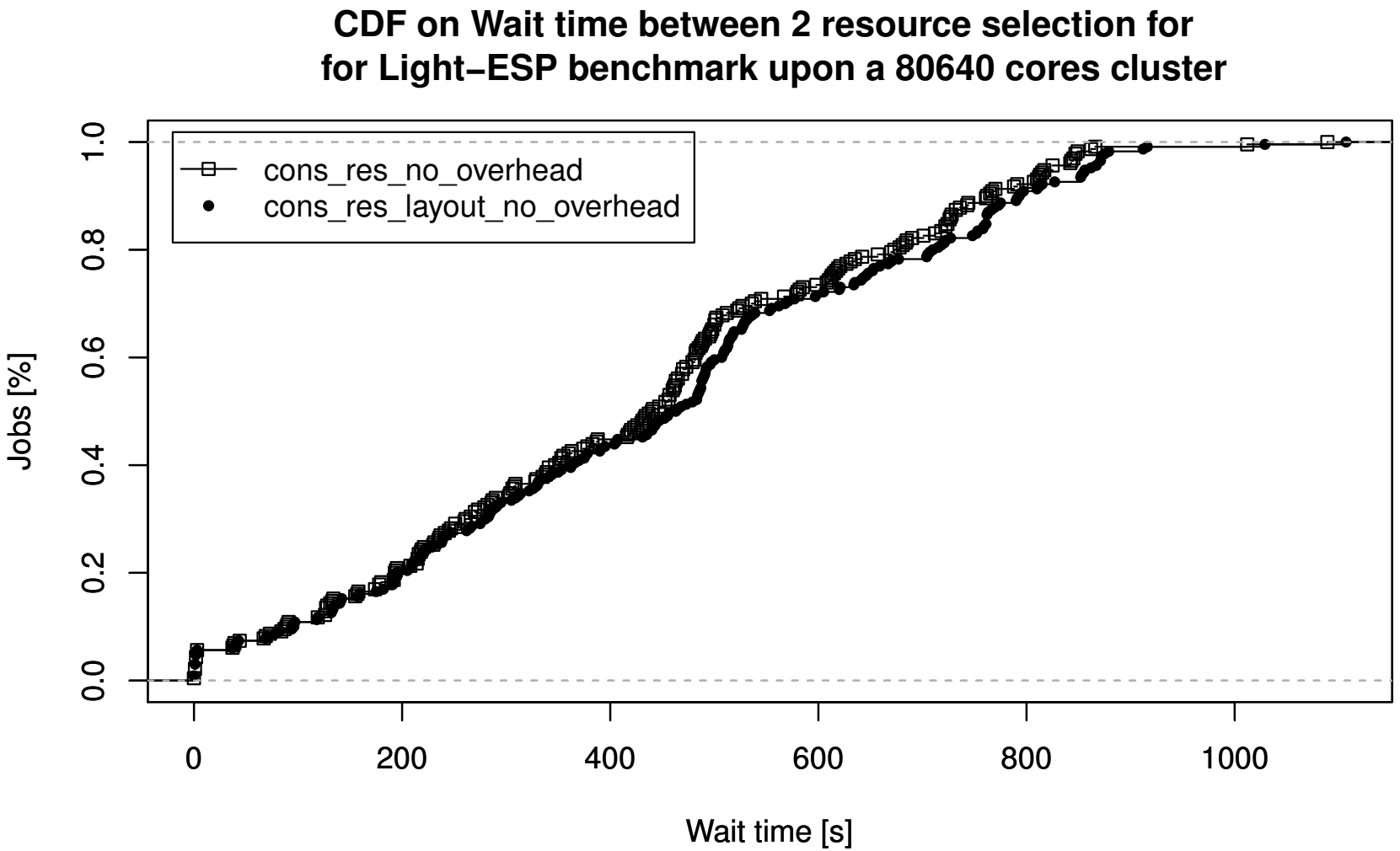


Individual Jobs Performance Comparison

1. Due to large number of entity update in Cons_res_layout plugin, individual resource selection time increased twice

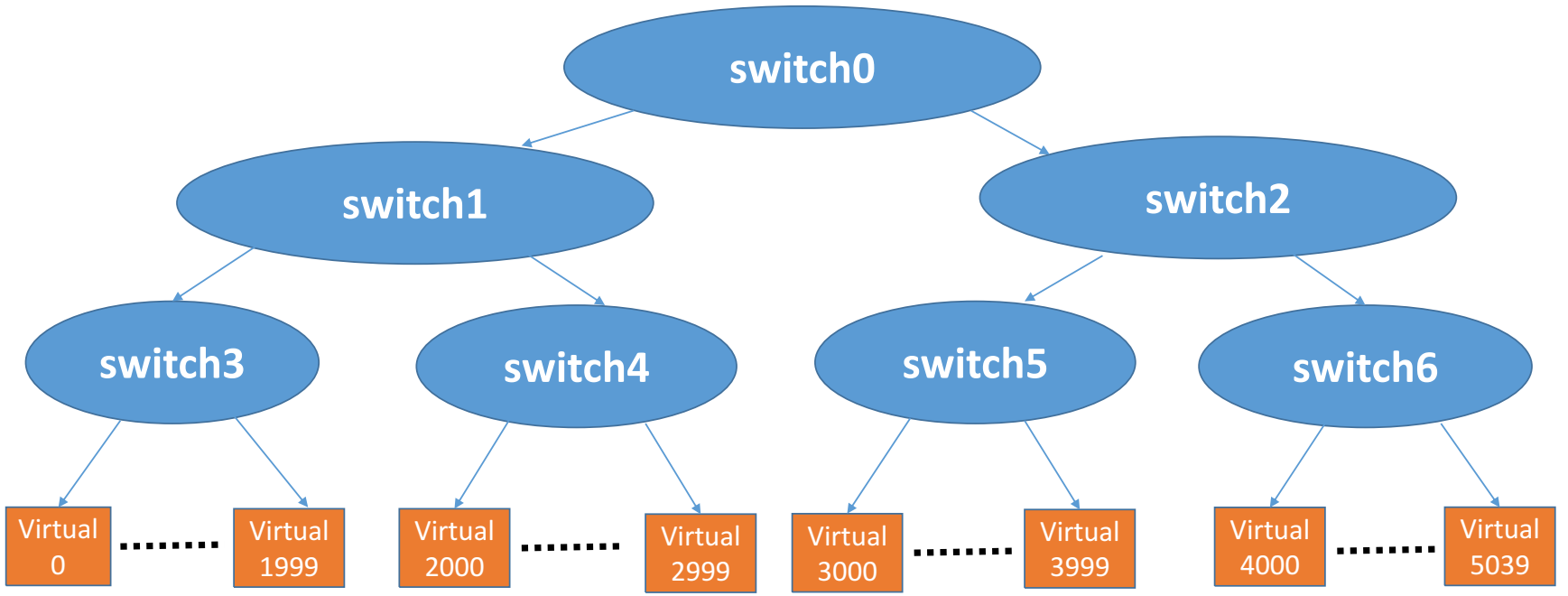


Waiting Time for 2 Plugins

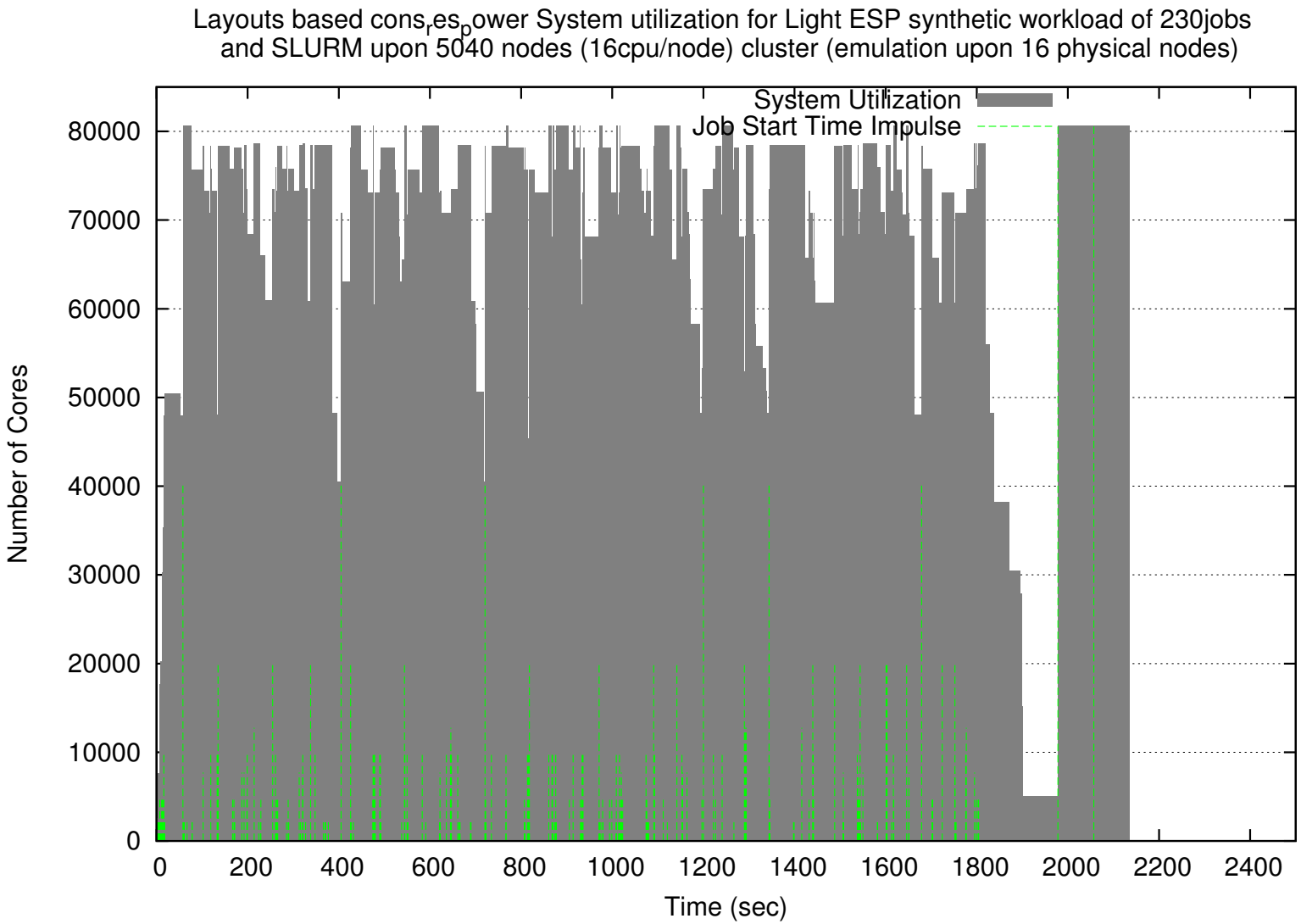


Energy Experiment Environment

1. **Heterogeneous power consuming nodes** cluster environment
2. Simple tree topology to have **4 leaf switches** and **3 levels**
3. **Homogeneous power consuming nodes** within leaf switches

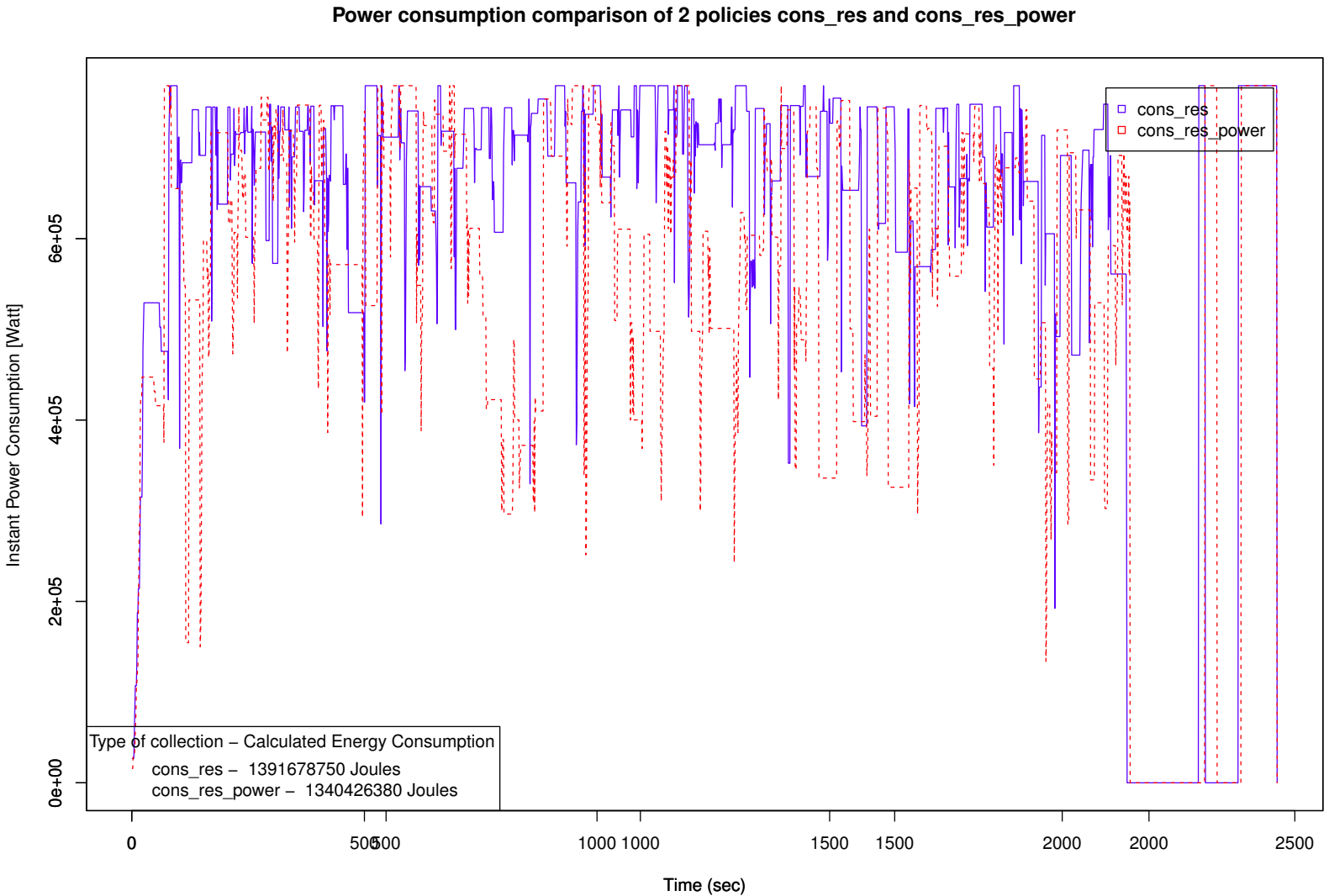


Cons_res_power System Utilization



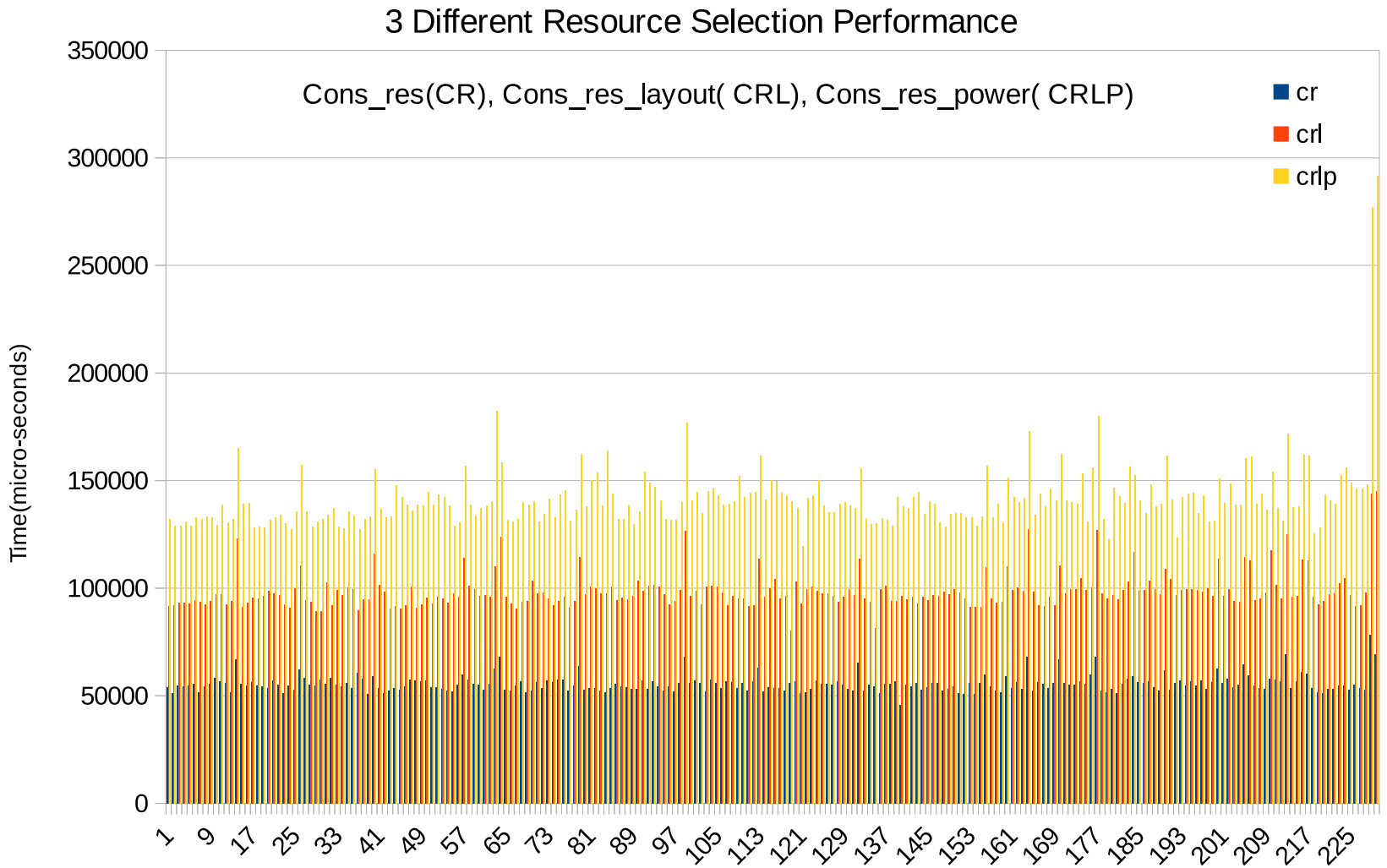
Energy Efficiency

Cons_res_power energy consumption is less than Cons_res by 3.8% (51252370 Joules) for same job workload

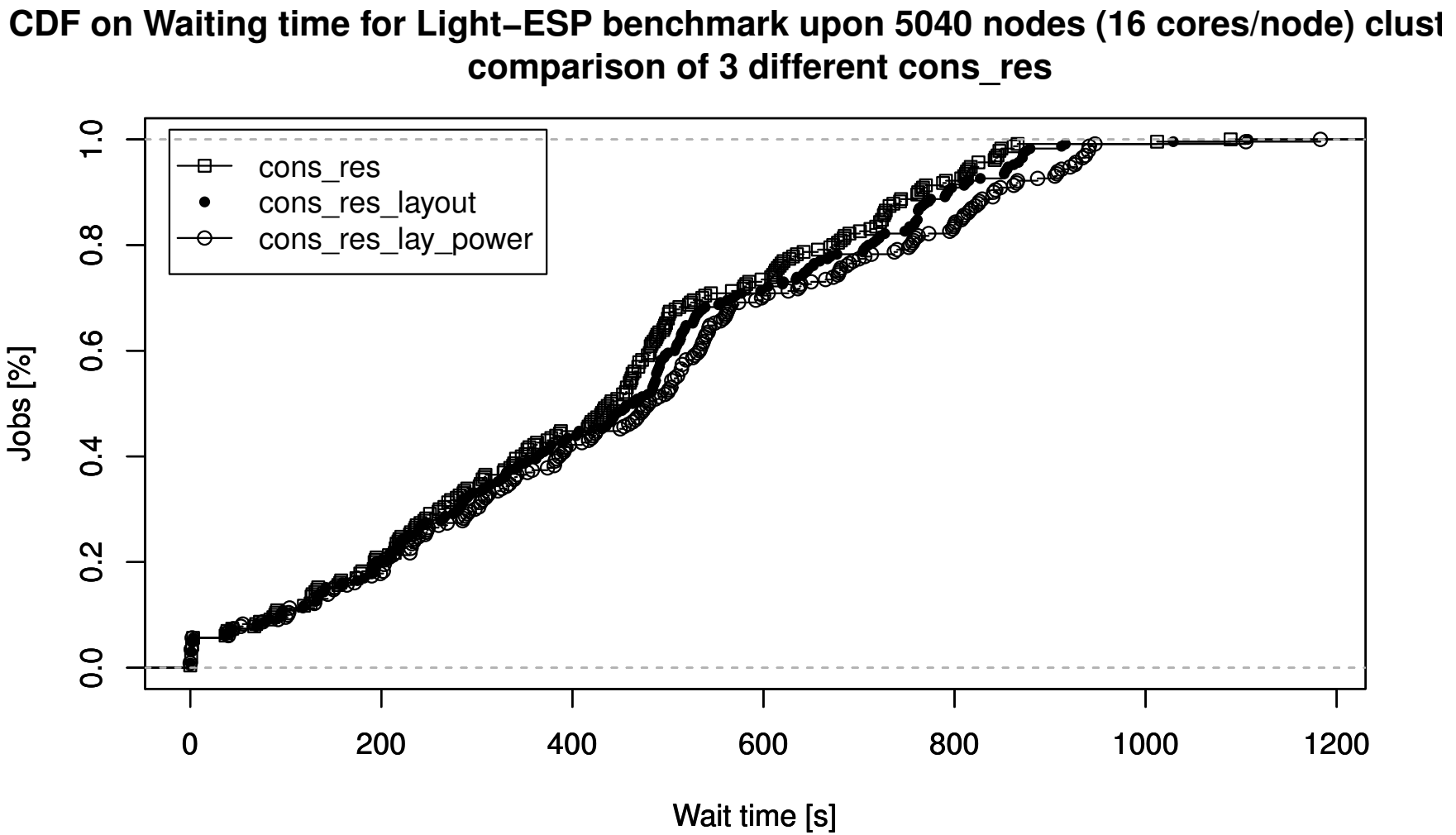


Individual Jobs Performance Comparison

1. **Cons_res_power** individual job resource selection time increased thrice than **cons_res**



Waiting Time for 3 Plugins



Conclusion

1. SLURM plugin enhancement is easier using LAYOUTS resource management framework
2. New Cons_res_layout resource selection plugin developed using LAYOUTS to support hierarchical resource management in SLURM

Conclusion

1. SLURM plugin enhancement is easier using LAYOUTS resource management framework
2. New Cons_res_layout resource selection plugin developed using LAYOUTS to support hierarchical resource management in SLURM
3. Cons_res_layout and cons_res plugin system utilisation and throughput is almost same and minimal increase of resource selection performance overhead

Conclusion

1. SLURM plugin enhancement is easier using LAYOUTS resource management framework
2. New Cons_res_layout resource selection plugin developed using LAYOUTS to support hierarchical resource management in SLURM
3. Cons_res_layout and cons_res plugin system utilisation and throughput is almost same and minimal increase of resource selection performance overhead
4. Multi-parameter resource selection policy adapted in cons_res_layout plugin
5. Energy consumption of multi-parameter resource selection reduced by 3.8 % and minimal increase of resource selection performance overhead

Future Work

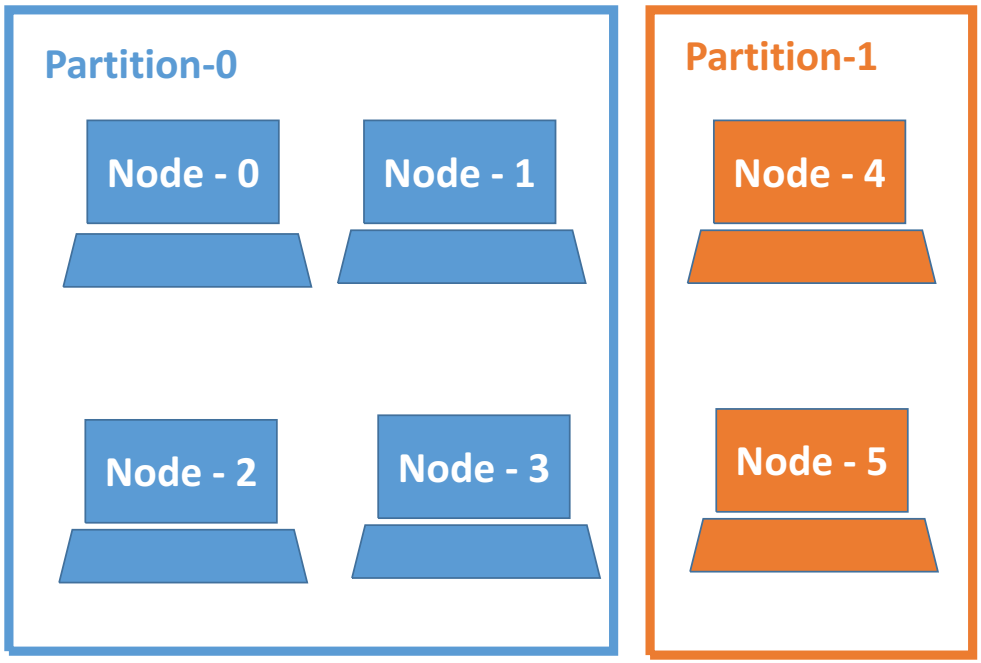
1. Support partition entities and fat-tree topology in the layouts plugins
2. Adapt cons_res_power to support real energy values from RAPL or IPMI technique
3. Include temperature criteria in the resource selection
4. Experiment to measure
   - Perform scalability experiments with large number of nodes
   - Instantaneous job throughput
   - Instantaneous number of job types allocated
   - Cons_res_ power job waiting time is better than powercapping only approach

Thank you ..

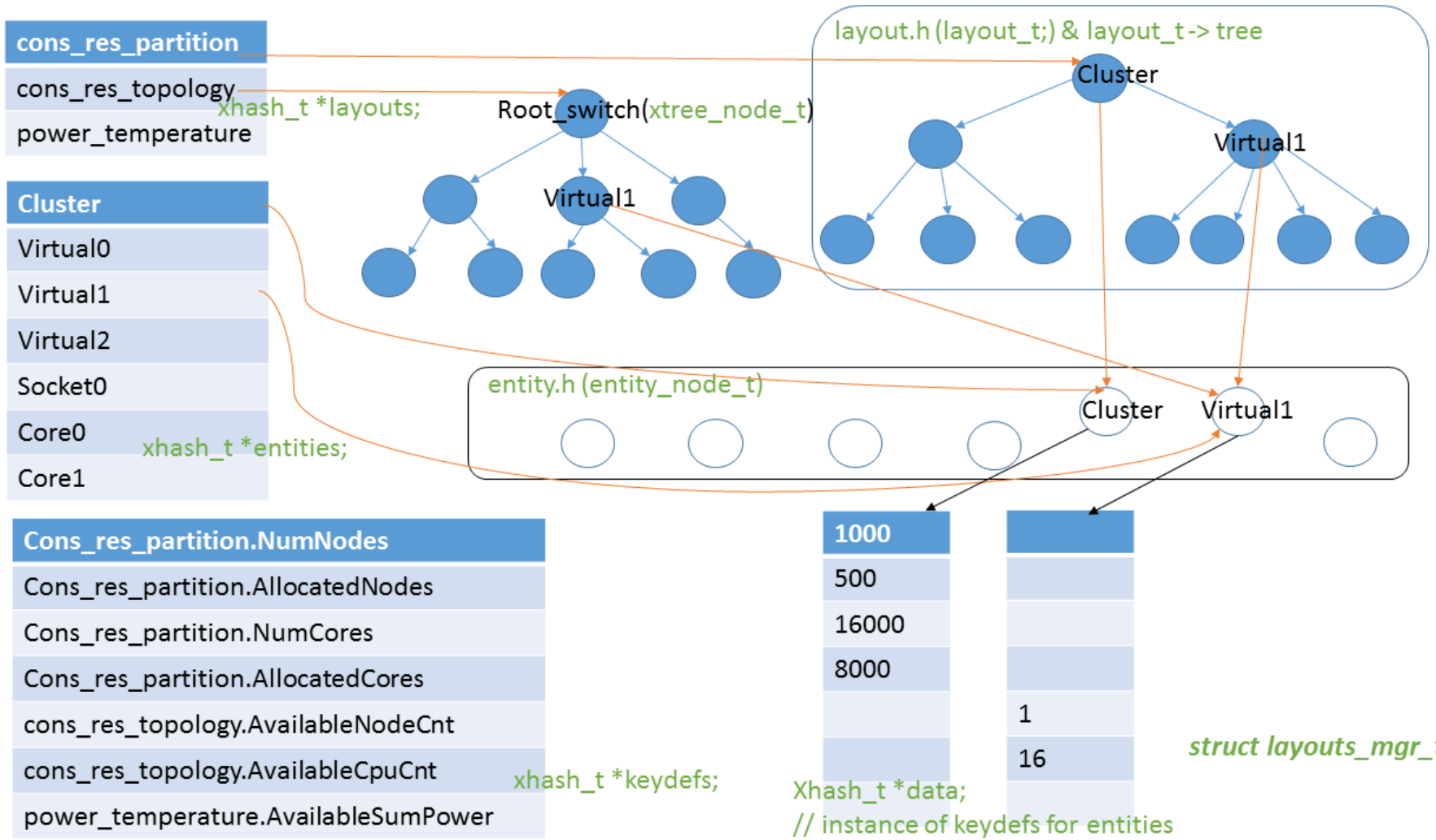
Any questions?

SLURM Entities

1. SLURM **resource management** entities
2. **Job management** entities are not considered



LAYOUTS Internal Architecture



LAYOUTS Aggregate Keys

- Child aggregate functions - Specific operations performed on the children key value and update the calculated value in the parent's key
  1. KEYSPEC_UPDATE_CHILDREN_SUM
  2. KEYSPEC_UPDATE_CHILDREN_AVG
  3. KEYSPEC_UPDATE_CHILDREN_MIN
  4. KEYSPEC_UPDATE_CHILDREN_MAX
  5. KEYSPEC_UPDATE_CHILDREN_COUNT
  6. KEYSPEC_UPDATE_CHILDREN_MASK
- Parent aggregate functions
  1. KEYSPEC_UPDATE_PARENTS_SUM
  2. KEYSPEC_UPDATE_PARENTS_AVG
  3. KEYSPEC_UPDATE_PARENTS_MIN
  4. KEYSPEC_UPDATE_PARENTS_MAX
  5. KEYSPEC_UPDATE_PARENTS_FSHARE
  6. KEYSPEC_UPDATE_PARENTS_MASK

LAYOUTS New APIs

New APIs developed for **new functionality** and **performance** purpose.

1. **layouts_entity_get_parent_name()**
2. **layouts_multi_entity_set_kv()** - set **entity** same key
3. **layouts_multi_entity_get_kv()**
4. **layouts_entity_pull_get_skv()** - update **specific key** and its relations
5. **layouts_entity_set_push_skv()**

Cons_res_layout Performance Improvements

1. Node and Core list is maintained in the **bitmap** data structure to compare and match another node list faster
2. LAYOUTS to access **entities keys** in the different levels

Cons_res_layout Performance Improvements

1. Node and Core list is maintained in the **bitmap** data structure to compare and match another node list faster
2. LAYOUTS to access **entities keys** in the different levels
3. **layouts_multi_entity_set_kv** - transfer information from one layout to another

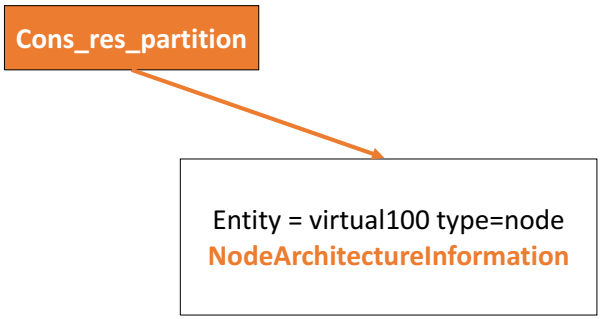
Cons_res_layout Performance Improvements

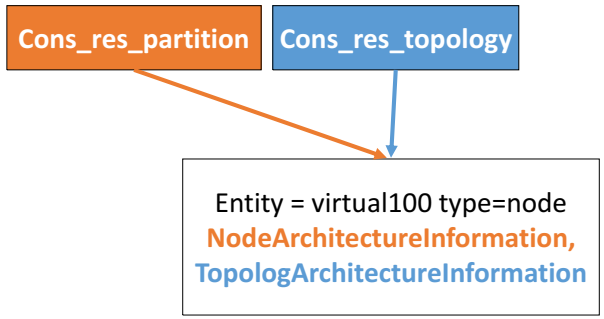
1. Node and Core list is maintained in the **bitmap** data structure to compare and match another node list faster
2. LAYOUTS to access **entities keys** in the different levels
3. **layouts_multi_entity_set_kv** - transfer information from one layout to another
4. **layouts_entity_pull_get_skv** - update only specific key values

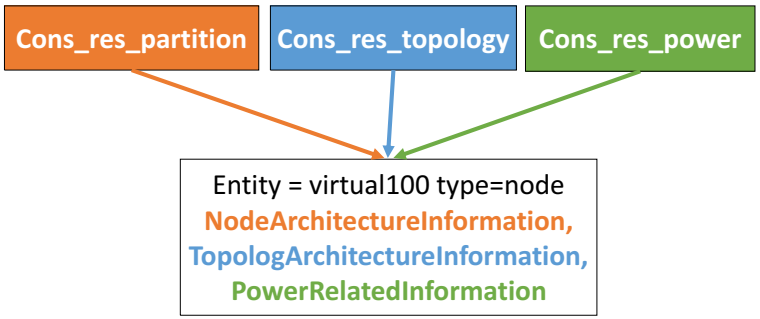
Cons_res_layout Performance Improvements

1. Node and Core list is maintained in the **bitmap** data structure to compare and match another node list faster
2. LAYOUTS to access **entities keys** in the different levels
3. **layouts_multi_entity_set_kv** - transfer information from one layout to another
4. **layouts_entity_pull_get_skv** - update only specific key values
5. Topology aware resource selection algorithm was adapted for **normal tree** topology

LAYOUTS Entity Data Management

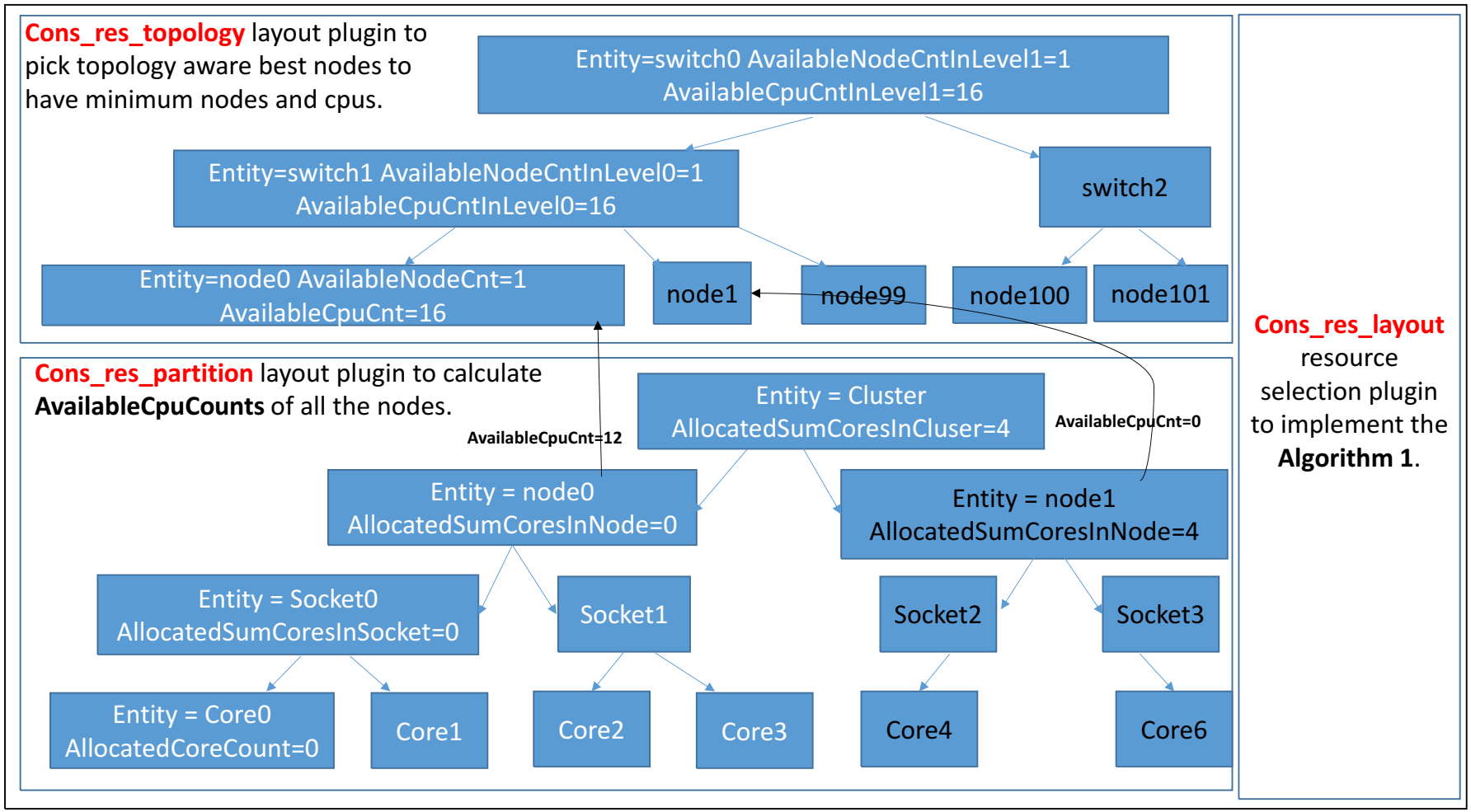




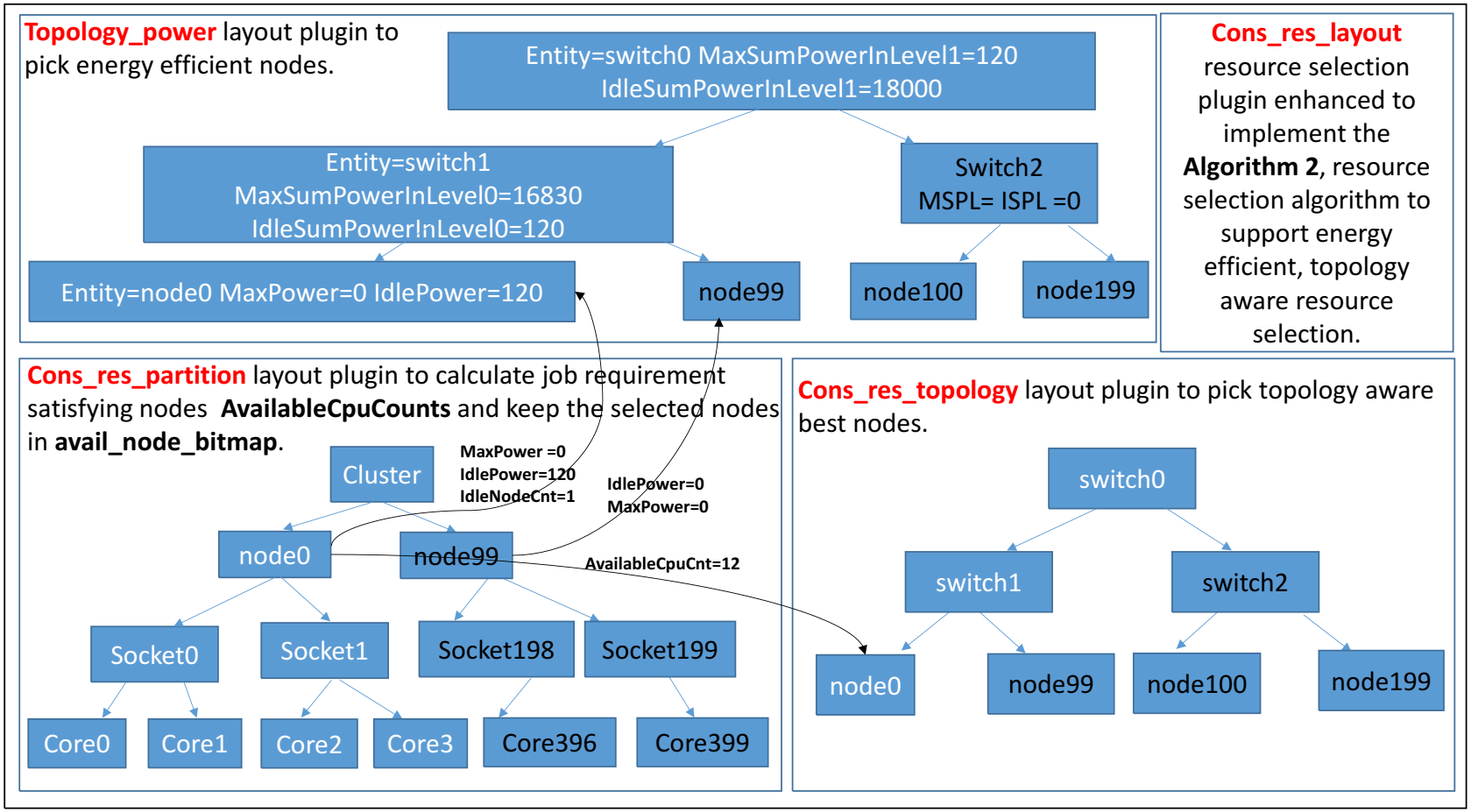


1. Entity and entity keys are in hash data structure and access very fast
2. Entity has different layouts plugin information to view entity different perspective

Cons_res_layout Architecture



Cons_res_power Architecture



Conclusion I

1. SLURM plugin enhancement is easier using LAYOUTS resource management framework

Conclusion I

1. SLURM plugin enhancement is easier using LAYOUTS resource management framework
2. Cons_res_layout resource selection plugin developed using LAYOUTS

Conclusion I

1. SLURM plugin enhancement is easier using LAYOUTS resource management framework
2. Cons_res_layout resource selection plugin developed using LAYOUTS
3. LAYOUTS new APIs developed to increase performance for multiple updates

Conclusion I

1. SLURM plugin enhancement is easier using LAYOUTS resource management framework
2. Cons_res_layout resource selection plugin developed using LAYOUTS
3. LAYOUTS new APIs developed to increase performance for multiple updates
4. Cons_res_layout and cons_res plugin system utilisation and throughput is same

Conclusion I

1. SLURM plugin enhancement is easier using LAYOUTS resource management framework
2. Cons_res_layout resource selection plugin developed using LAYOUTS
3. LAYOUTS new APIs developed to increase performance for multiple updates
4. Cons_res_layout and cons_res plugin system utilisation and throughput is same
5. Cons_res_layout individual resource selection performance overhead was increased twice than cons_res plugin
6. Cons_res_layout job waiting time increased, due to individual performance of resource selection

Conclusion II

1. Multi-parameter resource selection policy implemented in Cons_res_power plugin

Conclusion II

1. Multi-parameter resource selection policy implemented in Cons_res_power plugin
2. Cons_res_power plugin achieved both application performance and server energy efficiency

Conclusion II

1. Multi-parameter resource selection policy implemented in Cons_res_power plugin
2. Cons_res_power plugin achieved both application performance and server energy efficiency
3. Energy consumption of multi-parameter resource selection reduced by 3.8 %

Conclusion II

1. Multi-parameter resource selection policy implemented in Cons_res_power plugin
2. Cons_res_power plugin achieved both application performance and server energy efficiency
3. Energy consumption of multi-parameter resource selection reduced by 3.8 %
4. Cons_res_power individual resource selection performance overhead was increased thrice than cons_res plugin
5. Cons_res_power job waiting time increased, due to individual performance of resource selection