International Conference
Russian Supercomputing Days 2021

Chained Optimization Scheme for Simultaneous Co-allocation of Resources of Different Types

Authors:

Victor Toporkov

Dmitry Yemelyanov

Department of Computing Technologies

National Research University "Moscow Power Engineering Institute"

September
27-28, 2021

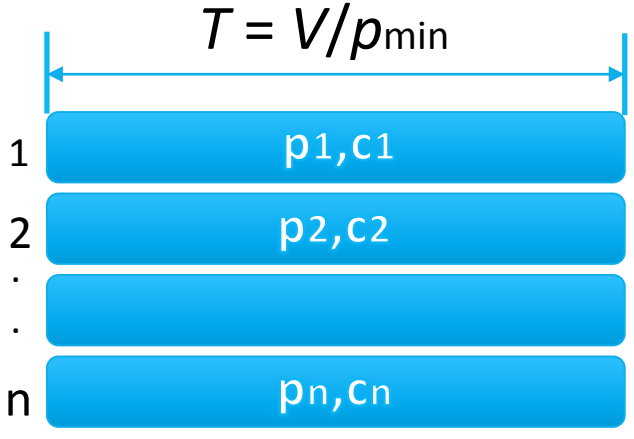
Outline

- Parallel jobs scheduling in high performance and cloud systems requires the use of significant resources of different types (computational nodes, data storages, software packets, network connections, etc.)
- Known scheduling algorithms are designed to allocate homogeneous or heterogeneous resources of the same type
- We propose a solution for the problem of simultaneous co-allocation of resources of different types

Single-type Resource Requests

The resource requirements for a single job execution are arranged into a resource request:

- *n* - number of simultaneously reserved computational nodes
- *p* - minimal performance requirement for each computational node
- *V* - computational volume for a single node task
- *C* - maximum total job execution cost (budget)

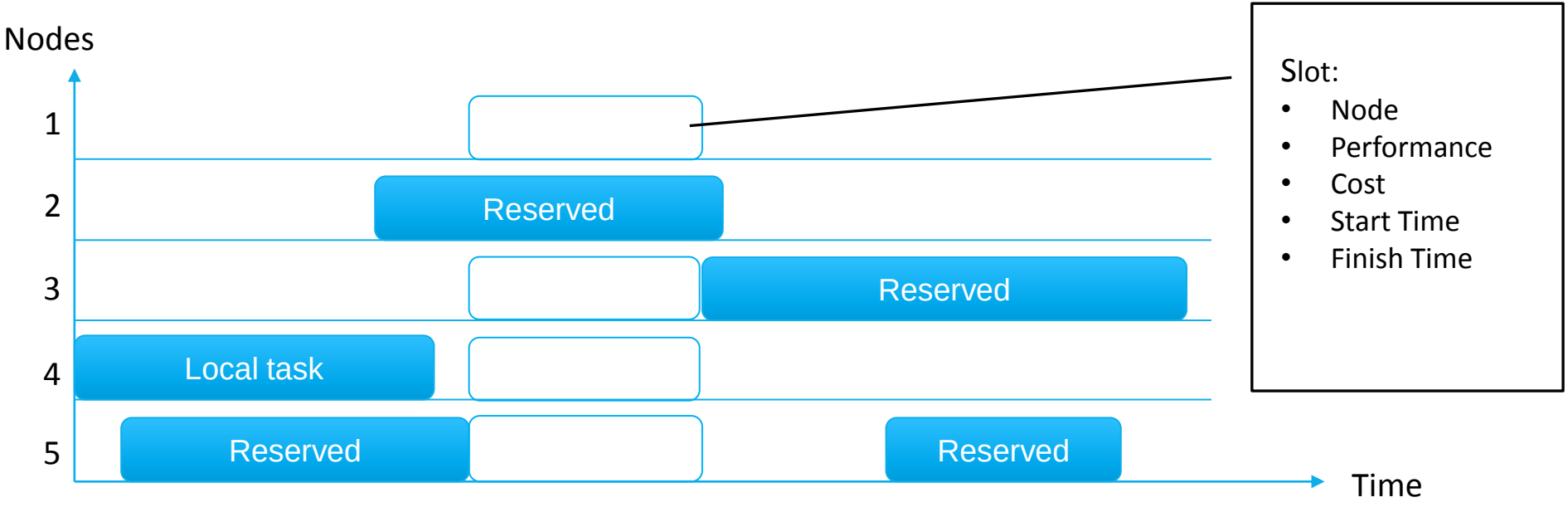


Heterogeneous resources of the same type share the same characteristics:

- performance and cost for computing nodes
- volume and cost for memory modules
- storage capacity and cost for SSD
- etc.

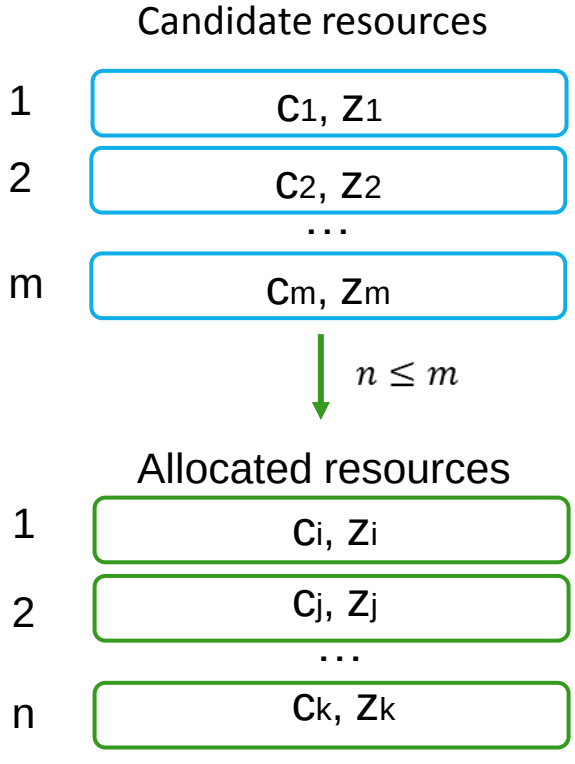
Window Search Problem

Allocate a window of computing four nodes for a time *T*, with requirements on nodes performance and total cost. Minimize window start time:



Chained Optimization Scheme for Simultaneous Co-allocation of Resources of Different Types

0-1 Knapsack Slots Subset Allocation



$$Z = \sum_{i=1}^{m} z_i x_i \rightarrow \max$$

$$\sum_{i=1}^{m} c_i x_i \leq C_j,$$

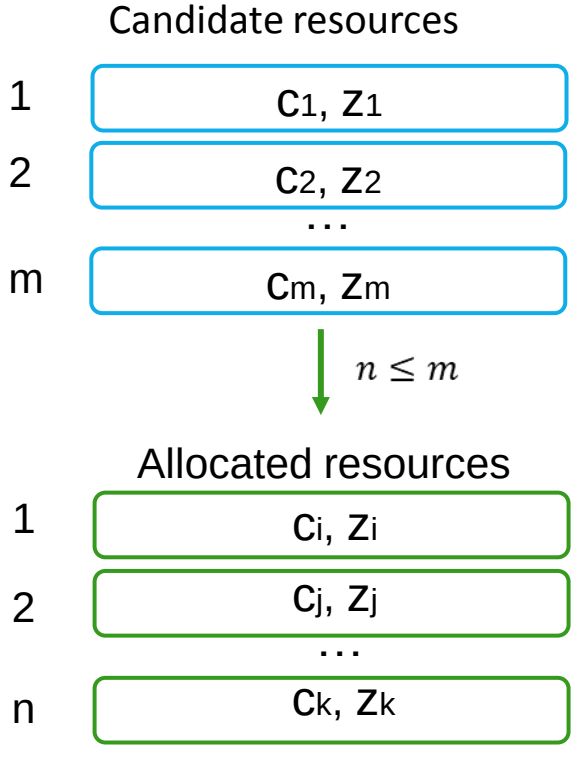
$$x_i \in \{0,1\}, i = 1, \ldots, m$$

Number n of allocated resources is not limited: $n \in [0; m]$

Classic 0-1 knapsack problem

Chained Optimization Scheme for Simultaneous Co-allocation of Resources of Different Types

Limited Size Slots Subset Allocation



$$Z = \sum_{i=1}^{m} z_i x_i \rightarrow \max$$

$$\sum_{i=1}^{m} c_i x_i \le C_j,$$

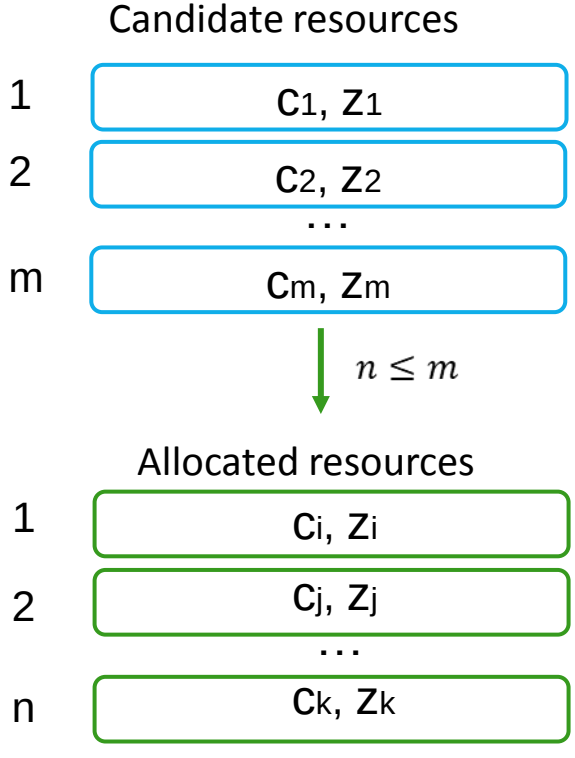
$$\sum_{i=1}^{m} x_i = n,$$

$$x_i \in \{0,1\}, i = 1, \ldots, m$$

Number *n* of simultaneously required resources is predetermined

Chained Optimization Scheme for Simultaneous Co-allocation of Resources of Different Types

Interval-based Slots Subset Allocation



$$Z = \sum_{i=1}^{m} z_i x_i \to \max$$

$$\sum_{i=1}^{m} c_i x_i \le C,$$

$$\sum_{i=1}^{m} x_i \ge n_{\min},$$

$$\sum_{i=1}^{m} x_i \le n_{\max},$$

$$x_i \in \{0,1\}, i = 1, \dots, m$$

Interval of permissible values $[n_{\min}; n_{\max}]$ is defined for *n*

This problem describes a more generic resources allocation scenario

Chained Optimization Scheme for Simultaneous Co-allocation of Resources of Different Types

Recurrent Solution for Interval Problem

$$f_i(c,k) = \max\{f_{i-1}(c,k), f_{i-1}(c-c_i, k-1) + z_i\},$$

$$i = 1,..,m, c = 1,..,C_j, k = 1,..,n_{\max}$$

$$Z_{\max} = \max_n f_m(C,n)$$

i

c

k=1	c=3 z=3	c=2 z=1	c=4 z=6
0	-	-	-
1	-	-	-
2	-	1	1
3	3	3	3
4	3	3	6
5	3	3	6
6	3	3	6

k=2	c=3 z=3	c=2 z=1	c=4 z=6
0	-	-	-
1	-	-	-
2	-	-	-
3	-	-	-
4	-	-	-
5	-	4	4
6	-	4	7

$$O(m * n_{\max} * C)$$

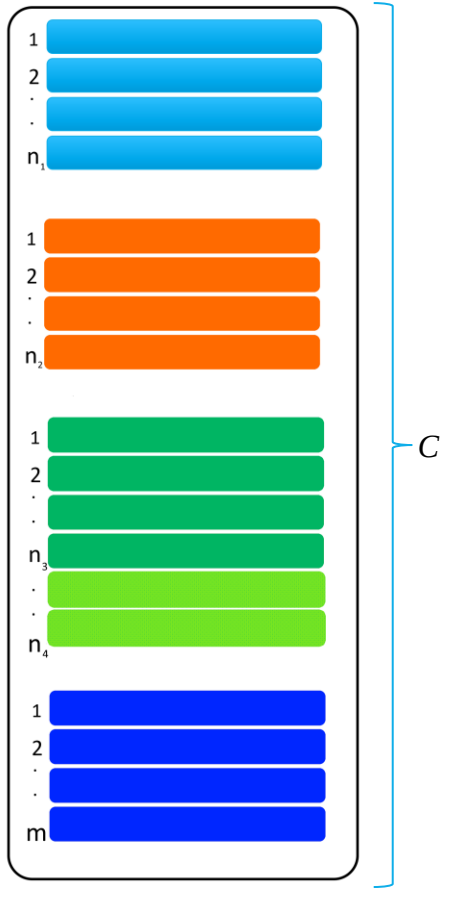
Chained Optimization Scheme for Simultaneous Co-allocation of Resources of Different Types

Composite Scheduling Model

- Different allocation requirements are given for different type of the resources
- There is usually a single common constraint on the total resources' usage price: a budget allocated for the job execution

The united scattered requirements may be compilated in a single composite resource request, for example:

- allocate n_1 computational nodes, the corresponding number n_2 of software packets, $[n_3; n_4]$ storage drives and at least one network connection
- total allocation budget C
- perform optimization by the criterion function Z.



Chained Optimization Scheme for Simultaneous Co-allocation of Resources of Different Types

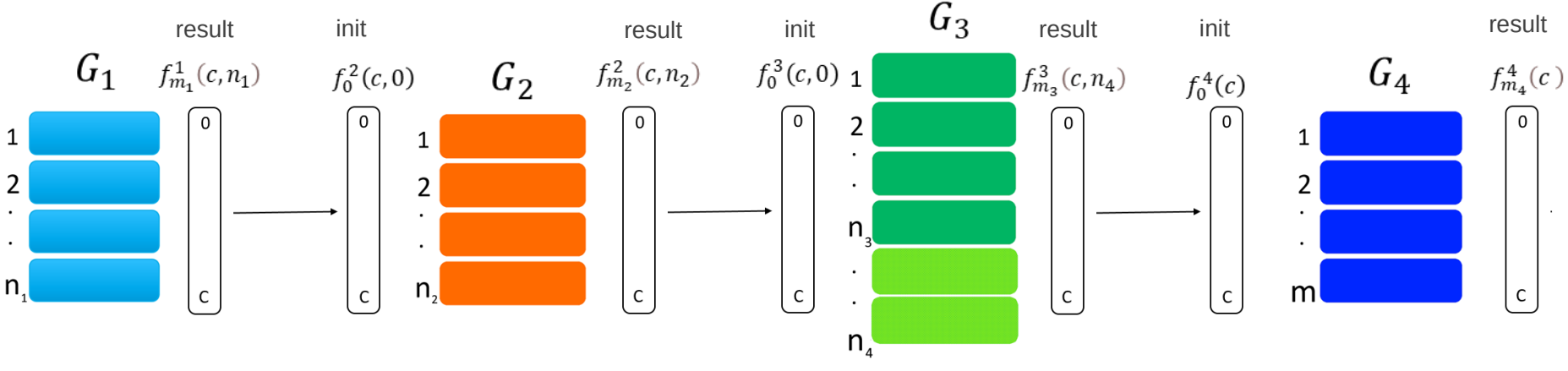
Chain Algorithm

The recurrent calculations for sub-problem G_j is passed as an input initialization data for sub-problem G_{j+1}

Vector $f_0(c,0), c = 0,..,C$ determines the best attainable solution for all the previous sub-problems

Resulting solution when the recurrent scheme for the last *N*th problem is calculated:

$$Z_{\max} = f_{m_N}^N(C, n_N).$$



Chained Optimization Scheme for Simultaneous Co-allocation of Resources of Different Types

Resources Allocation Example

Composite request:

allocate 8-12 VMs, 1-8 storage drives and one OS delivery and support plan (up to 15 workstations) for 350 dollars a month

maximize $\sum_{i=1}^{m} z_i$, where z_i is pre-calculated user utility estimation of each offer

Solution:

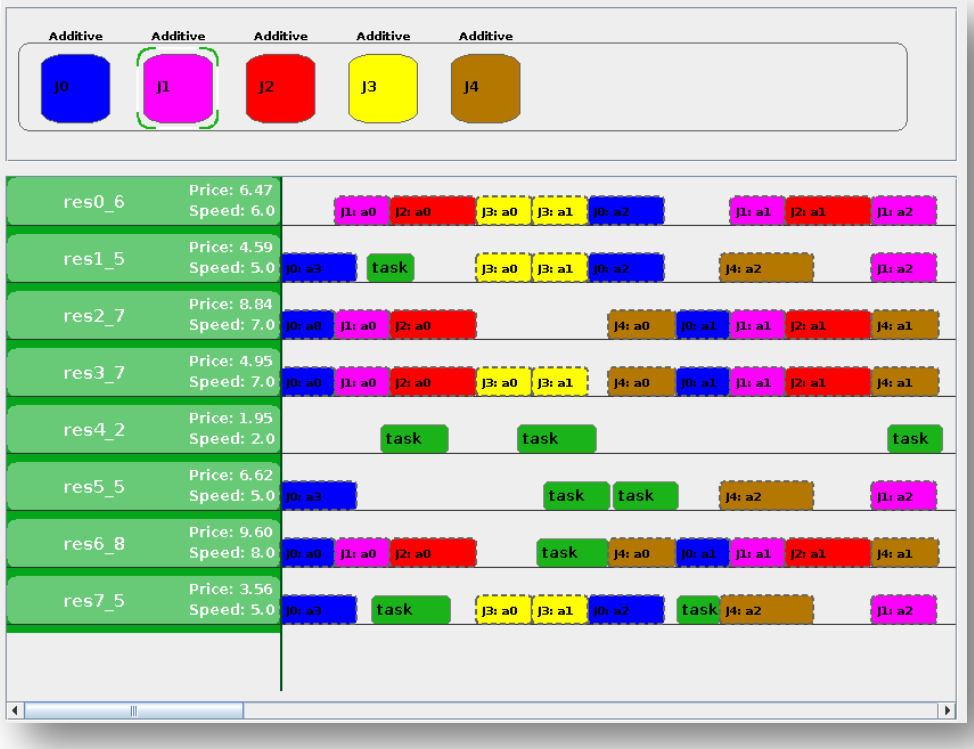
7 VMs with 2GB RAM for 10 dollars, one VM with 1GB RAM for 5 dollars, three additional storage drives of 500Gb each and five drives of 250 Gb each, free OS distribution

Type	Vendor	Characteristic	Price, c_i	Utility, z_i
VM	Vendor 1	1Gb RAM	5	0
VM	Vendor 2	1Gb RAM	8	0
VM	Vendor 1	2Gb RAM	10	2
VM	Vendor 3	4Gb RAM	30	4
VM	Vendor 2	8Gb RAM	62	8
VM	Vendor 3	8Gb RAM	61	8
VM	Vendor 1	16Gb RAM	80	10
DISC	Vendor 1	5Gb	0.5	0
DISC	Vendor 1	10Gb	1	0
DISC	Vendor 1	20Gb	2	0.5
DISC	Vendor 1	40Gb	4	1
DISC	Vendor 1	80Gb	8	2
DISC	Vendor 1	120Gb	12	3.5
DISC	Vendor 1	250Gb	25	8
DISC	Vendor 1	500Gb	50	15
DISC	Vendor 1	1000Gb	100	25
OS	Vendor 4	OS1	0	3
OS	Vendor 4	OS1 Software Support	120	6
OS	Vendor 4	OS1 Enterprise Middle Support	150	8
OS	Vendor 4	OS1 Enterprise 24/7 Support	300	9
OS	Vendor 5	OS1 Network Support	12	5
OS	Vendor 5	OS1 Basic Limited Support	50	6
OS	Vendor 5	OS1 Basic Support	120	9
OS	Vendor 5	OS1 Premier Support	140	14
OS	Vendor 6	OS2 Developer Support	29	7
OS	Vendor 6	OS3 Standard Support	100	12
OS	Vendor 6	OS3 Professional Support	1000	20

Chained Optimization Scheme for Simultaneous Co-allocation of Resources of Different Types

Simulation Study

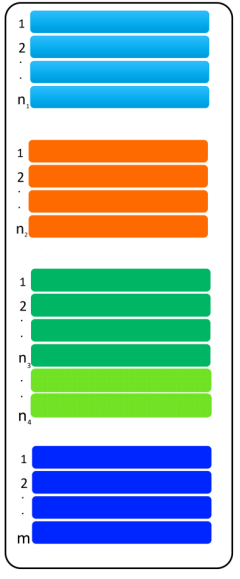
- Heterogeneous environment consists of:
  - VMs,
  - storage drives
  - software products
- Single composite resource request:
  - allocate 8-12 VMs, any number of storage drives and one (common) license for a software product
  - execution budget is 350
- Utility values z_i precalculated randomly for each of 200 different resource instances



Chained Optimization Scheme for Simultaneous Co-allocation of Resources of Different Types

Greedy Solution GX

$$\sum C_i = C$$



1. GX distributes the available budget *C* between the individual sub-problems
2. Each sub-problem is solved independently by the corresponding greedy algorithm
3. The results of the individual solutions are combined into a common composite solution

Budget distribution strategies:

1. *GUniform* distributes budget equally : C/N for each sub-problem
2. *GProportional* allocates budget C according to a heuristic metric proportionally to the sub-problem requirements
3. *GReferenced* accepts the composite problem solution obtained by the *Chain* algorithm as a reference and distributes the available budget in the same proportion (we assume that *Chain* algorithm provides the optimal budget distribution)

Considered Algorithms

For the experiment simulation we consider the following algorithms.

Feasible algorithms:

- *Chain* knapsack-based algorithm
- *GUniform* greedy algorithm
- *GProportional* greedy algorithm

Infeasible algorithms (for additional evaluation):

- *GReferenced* greedy algorithm (requires *Chain* solution as an input)
- *GUniform X2.1* greedy algorithm with available budget 2.1 times increased
- *GProportional X1.35* greedy algorithm with available budget 1.35 times increased
- *GReferenced X1.18* greedy algorithm with available budget 1.18 times increased

Chained Optimization Scheme for Simultaneous Co-allocation of Resources of Different Types

Simulation Results

Algorithm	Z_{max}	Used budget	Working Time, ms
GUniform	622.9	220.2	0.4
GProportional	752	322.5	0.4
GReferenced	776.9	323.2	0.4
GUniform X2.1	825.6	387.4	0.4
GProportional X1.35	829.9	391.8	0.4
GReferenced X1.18	826.6	380.4	0.4
Chain	827.4	349.8	34.3

Results were obtained based on 6000 independent randomized experiment scenarios

Each scenario included resources allocation problem in an environment with 3 types of resources and $C = 350$ \$ budget limit

Simulation Results

- *Chain* knapsack-based algorithm advantage by the target criterion *Z* over greedy analogues ranges from 6% to 25%
- *Greedy* algorithms requires 35% to 110% additional budget to catch up with the *chain* solution
- *Chain* algorithm requires much more time for the calculations (100x times in our test environment)
- *GProportional* algorithm with budget distribution can outperform greedy analogues

Conclusion and Future Work

- *Chain* algorithm is proposed to use knapsack-based allocation procedure for composite scheduling problems with resources of different types (computational nodes, data storages, software packets, network connections, etc.) and a common cost limit
- *Chain* algorithm *automatically* solves the problem of proper budget distribution between the constituent subproblems
- This allows *Chain* algorithm to noticeably outperform greedy analogues and justify its relatively high computational complexity
- The future research directions include design and testing of additional technics for the chain algorithm complexity optimization

References

- Performance Evaluation Models for Distributed Service Networks. Studies in Systems, Decision and Control. Vol. 343. Editors: Grzegorz Bocewicz, Jarosław Pempera, Victor Toporkov. Springer Nature Switzerland AG. 2021. 188 p.
- Victor Toporkov and Dmitry Yemelyanov. Availability-based Resources Allocation Algorithms in Distributed Computing // In: V. Voevodin and S. Sobolev (Eds.): RuSCDays 2020, CCIS 1331, Springer Nature Switzerland AG 2020, pp. 551-562, 2020.
- Victor Toporkov, Dmitry Yemelyanov, and Anna Toporkova. Coordinated Global and Private Job-Flow Scheduling in Grid Virtual Organizations // Simulation Modelling Practice and Theory, Elsevier, 107 (2021) 102228.

Acknowledgments

We would like to thank our colleagues from the Karlsruhe Institute of Technology in Germany and CERN (project CMS) for their valuable comments and suggestions to help and improve our research paper.

Thank you for attention!

International Conference
Russian Supercomputing Days 2021

September 27-28, 2021