Elastic computing self-organizing for artificial intelligence space exploration

A Semenov

Department of Computational Mathematics and Programming, Moscow Aviation Institute (National Research University), 4 Volokolamskoe shosse, Moscow, 125993, Russian Federation

E-mail: SemenovAS@mai.ru

Abstract. The aim of this article is to provide a methodology useful for designing and controlling elastic computing self-organizing for artificial intelligence space exploration. The artificial intelligence application itself should be elastic and distributed in the context of limited information technologies resources in space. The most important use of elastic computing is artificial intelligence's ability to continually learn and adapt to evolving environments and goals. The conceptual framework uses elastic infrastructure model and the terminology of graph dynamical systems to be able to capture a broad variety of processes taking place on self-organizing networks. The methodology's uniqueness lies in the theory of graph dynamical systems used to explain the self-organizing processes life cycle.

1. Introduction

Nowadays, information technology (IT) plays an important role in space exploration by spacecrafts (SCs). Cloud computing is crucial because of the internet that provides fast access and analysis of data through shared servers allows to solve problems in a cost-effective manner. This is achieved through elasticity of the system. The elasticity has become main characteristic of the contemporary information technologies. Run-time factors on demand are increasingly determining the elasticity. One of the fundamental elastic computing properties is autoscaling. It is the process of automatically increasing or decreasing the IT resources delivered to a cloud on demand. IT resources are categorized into hardware, software, and human resources. Hardware resources are processors, memory drives, and various technical devices. Software resources are software packages, applications, databases, and knowledge. Human resources include experience in mission management, the activities of astronauts and the skills they have acquired in training.

Advancement in general artificial intelligence (AI) [1] techniques such as machine learning and adaptation methods has provided relevant growing role for AI space exploration. Artificial intelligence systems are needed to automate the activities of astronauts, depending on emerging needs, to improve the autonomous functions of spacecraft (SC), space robots and equipment. AI poses an increasingly popular approach for implementing SC autonomy. Autonomy is the ability of a SC to achieve goals while operating independently without external control. To operate independently is required system self-sufficiency. To achieve goals is required system self-directedness. High-level autonomy for system and subsystems provides to improve overall performance of human and robotic missions. The autonomy of the SC [2] equipped with AI [3, 4] practically without human intervention is a shift towards the design of self-organizing systems [5] for space exploration that would require the
modernization of industrial robotics [6]. Self-organizing systems have the following advantages: increase efficiency and speed up missions by making decisions on their own, without waiting for instructions from the Earth; learn on the fly; adapt to certain situations and solve problems that were not initially defined for them [7]; explore space and planets using special robotic systems [8, 9]; conduct traditional scientific observations and experiments [10]; increase the efficiency of data processing [11]; commercialize space [12]; automate routine procedures and improve technical systems for servicing the activities of astronauts. Work in this direction is being intensively carried out [13]: autonomous groups and autonomous systems equipped with AI [14], a robot astronaut [15, 16], the Curiosity rover [17], and the commercialization of space [18, 19].

The aim of this article is to provide a methodology useful for designing and controlling elastic computing self-organizing (ECSO) for AI space exploration which should be elastic and distributed in the context of limited IT resources in space. The paper includes the conceptual framework using elastic infrastructure model and the terminology of graph dynamical systems and the methodology having uniqueness lies in the theory of graph dynamical systems used to explain the self-organizing processes life cycle.

2. Conceptual framework for elastic computing self-organizing
A conceptual framework is a set of concepts that should be useful to describe elastic computing self-organizing with AI space exploration context. Types of infrastructures, distributed AI, elasticity model, and graph dynamics are considered in the conceptual framework.

2.1. Types of infrastructures
Cloud computing services form the AI infrastructure. There are different types of infrastructures: cloud infrastructure (a data center), fog infrastructure (a backbone IP network, content delivery networks, and a ground station), edge infrastructure (a cellular core networks, "instant data" that is real-time data generated by sensors, and SC), and hybrid (figure 1).

![Figure 1. Types of infrastructures: cloud, fog, and edge.](image-url)

Cloud computing is a collection of servers that make up a distributed network, fog and edge computing are their extensions. The advantage of cloud computing is that they allow to collect data from multiple sites and devices and make it available anywhere in the world. Data from devices is collected by embedded hardware (edge infrastructure) and sent to the fog computing. The relevant data is then sent to the cloud computing, which is usually located in a different geographic area. The use of
edge computing offers advantages in optimizing the use of IT resources for cloud computing. The probability of a data bottleneck is reduced by reducing network traffic. The security improved by encrypting data near the core of the network. Edge computing, on the other hand, requires a bidirectional method for handling data.

Figure 1 shows the cloud infrastructure of a data center processing data from a SC with centralized elastic computing. Cloud infrastructure has only one autoscaling subsystem and one strategic planning subsystem for provisions IT-resources. And furthermore, IT-resources are centralized.

Due to GPU-enabled chipsets, mobile devices have become extremely powerful. Instead of being centralized, AI can now be moved from cloud data centers to edge devices such like cell phones. Edge computing allows processing to be performed locally at multiple decision points for the purpose of reducing network traffic. For example, Microsoft’s Azure Space combines a set of offerings and products, like cloud computing, artificial intelligence (AI), edge computing and machine learning. Figure 2 shows the shift elastic computing to the distributed edge infrastructure.

Integrating a cloud infrastructure with fog and edge yields superior performance. This allows to effectively solve the following problems: management of interaction and organization of stable communication of SCs; management of dynamically scalable systems, both consisting of several SCs, and consisting of hundreds and thousands of SCs; SC control in conditions of large dynamically moving space debris (traffic in near-earth space is becoming increasingly dense); processing of data received from incrementing number of SCs; commercialization of space missions (to exploit natural resources originating outside Earth and to organize space tourism).

2.2. Distributed artificial intelligence
The traditional AI approach is trained locally machine learning models on a large dataset. The most important use of elastic computing is AI's ability to continually learn and adapt to evolving environments and goals. Continual and adaptive learning focused on increasing amounts of data while maintaining previously acquired skills and updating predictive models. As data keeps coming in, it is possible to scale a distributive cognitive net.

Autonomous SC equipped with AI require new approaches to the use of the elastic computing. Regardless of the chosen AI methods [3], at the stages: training, data processing and decision-making,
different loads on elastic computing are required. For efficient operation of SC with AI, both new methods in processing of distributed data and approaches to organizing computational processes based on elastic computing will play an important role.

2.3. Elasticity model
The distributed elastic computing infrastructure allows to optimize AI systems in conditions of limited resources. It is based on the implementation of a distributed infrastructure with a scalable autonomy for multi-SCs, robots and equipment. The scalable autonomy impacts for distributed planning, scheduling, fault management, data distribution, and task execution. In this context, AI is regarded as a distributed one with high-level of the scalability.

Elasticity is ability of the system to transform its behavior: increase capacity on demand and decrease capacity on demand. Elasticity is the degree of autonomous adaptation over a certain time interval of the service capacity to the workload.

Let us consider workloads change and the system's elasticity over time. Such adaptation must be minimized due to the autonomy of the adaptation process. Workload characterizes the data that must be processed by the service's operations when demand fluctuates, for example, throughout the day.

The increase or decrease of the system capacity can be repeated several times over a period of time until the workload returns to its original value. Server workload and power are at the heart of elastic computing. Figure 1 shows widely used for the analysis of elastic systems pattern named Once-in-a-lifetime Workload [20]. In Figure 1 the transformation of the elastic system at runtime shown by the red curve: workload increases on the left side of the peak loads (vertical green line), as it decreases on the right.

Let us look at example of internet load during the day, where the horizontal axis shows time in hours, the vertical axis shows user requests per seconds. Let the service has enough capacity with a load of 2000 requests/sec (figure 3). Service capacity determines the maximum workload that the system can handle in terms of the number of requests. If the workload has increased to 3000 requests/sec, the capacity of the elastic computer system must dynamically transform.

If the workload increases, the system must dynamically transformed by adding IT resources to the system. If the workload is decreased, for example after 9000 requests/sec, the system capacity must be reduced by releasing IT resources.

Adding and releasing IT resources requires support for late binding of IT resources. This method is successfully used in programming language environments, in operating systems, when modules can be loaded and unloaded at will. Hardware and software binding and replication is eliminated by virtualizing the underlying infrastructure. There are following binding methods: resource-node, client-server, process-resource, geolocation.
To design elastic systems, it is necessary to conduct an analysis in terms of the system autoscaling depending on the workload, planning strategy, provision, binding of IT resources, and system synthesis. In order to summarize these, the following formula has been created:

\[
\text{Elasticity} = \text{Autoscaling} + \text{Strategic Planning} + \text{Binding} + \text{Synthesis}.
\] (1)

Let us give definitions to each term used in the formula (1). Automatic scaling (autoscaling) is a prerequisite for elastic calculations. The autoscaling method is a series of procedures to adapt the system to fluctuating workloads: without adding IT resources, or adding (deploying) IT resources to the system. The system's scalability should be considered during the design stage.

Strategic Planning is an integration of both methods: IT resources allocation and the scheduling. IT resources allocation defines the type and number of functional units, storage components, busses, and other connectivity elements required to generate current hardware infrastructure of the system. There are different models for the resource allocation (RA) problem [21]. Infrastructure may be used to categorize this RA models for cloud, fog, and edge. Further RA models can be categorized by organizing servers with Virtual Machines (VMs) or Containers.

For cloud infrastructure the following RA models were created: to deploy Software as a service (SaaS) applications over cloud computing platforms by taking into account their multi-tenancy [22]; mathematical model based on group technology (GT) to map the VMs to workflows in order to control some costs [23]; request type using autonomous and multi-objective resource sharing model for federated geo-distributed clouds [24]; VM allocation algorithm which optimizes the task execution in a secure cloud environment [25]; the solution of optimized container RA is made by considering objectives like Threshold Distance, Balanced Cluster Use, System Failure, and Total Network Distance, respectively. Finally, the performance of the proposed model is compared over other conventional models and proves its superiority [26]; an online multi-resource neural network is used to create a proactive autoscaling and energy-efficient VM allocation system for cloud data centers [27]; modeling of cloud outages events [28], quality of service model based on comparing the performance metrics (e.g., response time, throughput and resource utilization) [29]; a fair multi-attribute combinatorial double auction model [30]; Hybrid fog/cloud computing RA model based on joint consideration of limited communication resources and user credibility [31]; adaptive RA model allocation based on the billing granularity in edge-cloud infrastructure [32]; a combinatorial double auction RA model [33]; a feedback-based combinatorial fair economical double auction RA model [34]; in a Fog-Cloud environment, dynamic RA and provisioning algorithms based on deadlines [35]; self-adaptive RA model based on iterative quality of service (QoS) prediction [36]; a game theory-based dynamic RA strategy in Geo-distributed Datacenter [37]; a tenant-based RA model for scaling Software-as-a-Service applications over cloud [38].

Consider Strategic Planning for a constellation of autonomous SC (edge infrastructure). It is proposed to use the migrated master node model to request IT resources. Let SC₀ request an IT resource (figure 4, SC₀ node highlighted by red color). This means that a master node will be generated for it. The master node is the main node of the cluster that will be formed from IT resources of the SC constellation. IT resources are allocated in SCs themselves and redistributed among them. An autonomous SC tries to compute all its tasks using its own IT resources and partially the IT resources of others. Let SC₁, SC₂,..., SCₙ, are free resources most suitable for increasing the capacity of the SC₀.

The scheduling method forms software architecture and maps it on hardware architecture. On this step work (for example, W₁, W₂,..., Wₙ) is assigned to resources. The work may be threads, processes or data flows. Computational operations and the order in which each resource will occur are identified. Works are in turn scheduled onto hardware resources.
A binding allows to use services in a variety of ways for access to resources. In this step decisions about the sharing of the functional and storage units of the allocated resources are made. These decisions may have an effect on resource connectivity elements which may be revised.

A synthesis configuration [39] (or, simply, configuration) defines the transformations of the software and hardware infrastructure. Restriction and optimization instructions are used to manage these transformations.

The autoscaling, strategic planning, binding, and synthesis are interdependent and influenced by one another's decisions. Computational cost-effectiveness is closely linked to optimal resource allocation, balancing over-provisioning and under-provisioning. The transformation of the elastic system should be taken into consideration when developing planning strategies.

2.4. Graph dynamics
Swarm intelligence, multi-agent systems, actors, Petri nets, and graph dynamic systems are conceptual frameworks for distributed artificial intelligence. In this article graph based systems are used for distributed problem solving by scaling processing autonomous nodes. The demand for IT-resources in AI systems is growing, particularly if they processing a very large dataset. Distributing the processing to a cluster of autonomous computing nodes speeds up computation. The nodes of ECSO are SCs. Interactions or communication between these nodes are used to make decisions.

Let us consider migrated master node model M and graph dynamics of cluster formation for modeling available IT - resources for elastic computing.

$$M = (\text{mesh}(V', E), F, m)^N,$$

where $N = \{0, 1, \ldots, n\}$ - is a model steps; $\text{mesh}(V, E)$ - is a directed graph, where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of vertices, $(v_i, v_j) \in E = V \times V$ is a set of directed edges, where an edge $(v_i, v_j)$ is considered to be directed from $v_i$ to $v_j$.

$$F : V \times T \times N \times L \rightarrow V$$

is a function of marking vertex, $(v, \tau, n, l) \in V \times T \times N \times L$; $\tau \in T \cap N$ is a duration in model steps when label remains the same; $n \in N$ is a step of the model. $\text{mesh}(V, E) \Rightarrow \text{mesh}(V', E')$ yields in one step, F functions will be carried out; $L = \{\text{free, busy, demand, init, cluster}\}$.
is a set of labels, \( I \in L \). \( m \) is a marker of the master node. The marker moves from the initial vertex labeled by \((v_0, \tau, n, \text{init})\) to the vertex \( v_0 \) labeled by \((v_0, \tau, n, \text{demand})\). Vertex \( v_0 \) demands IT resources.

Let

\[
M = (\text{mesh} (V=\{SC_0,SC_1,SC_2,SC_3\}, E= \{(SC_0,SC_3), (SC_3,SC_2), (SC_2,SC_1)\}), (SC_2, SC_1)}) \),
\[
F=\{f_1(SC_0,10,0,\text{init}), f_2(SC_1,10,0,\text{free}), f_3(SC_2,1,1,\text{busy}), f_4(SC_3,5,0, \text{demand}) \}, m)_{n=3}
\]

Figure 5a shows the initially marked mesh \( M \). Vertexes marking by "free" are depicted green color. Vertex marking by "busy" is depicted red color. Figure 5b shows the model is yielded in one step and the marker \( m \) in the vertex \( SC_3 \). It has become the master node that forms a cluster.

\( \vdash \) is a step of the model firing that transition it to a new configuration: \( \text{mesh}(V,E) \vdash \text{mesh}'(V' = \{SC_0,SC_1,SC_3\}, E' = \{(SC_3, SC_0), (SC_3,SC_1)\}) \), note that at this step \( n=1 \) the vertex \( SC_2 \) became free.

![Figure 5. Migrated master node model of initial mesh (a) and generated cluster (b).](image)

3. Methodology

It should be said that design of the ECSO system is more like supervised learning and simulation when required software subsystems have already been implemented. Because of it, the first stage of methodology is associated with setting the initial values, which can be represented by the following formula (3):

\[
\text{ECSO} = \text{Infrastructure (Cloud, Fog, Edge, Hybrid)} + \text{Elasticity Management Model} \quad (3)
\]

The life cycle of ECSO consists of two stages Stable development and Transition (figure 6). These stages followed one by one, the stages merge with each other. Processes of both stages are considered on a certain time horizon and are characterized by the dynamics of behavior: at the stage Stable development it is ordered, at the Transition stage it is chaotic. Within the life cycle, the sequence of stages is repeated.

Backtracking occurs when the ECSO has to go back to a previous stage for reconsideration before completing the \( j \)-th step iteration. Backtracking within each cycle iteration will help in the design of ECSO system that are able to deal with uncertainty in dynamic problem domains.

ECSO system is involved both in the processing of data received from the outside world and in the SC management. External influences \( F \) affect on the SC, they are converted into datasets and after that they are processed by the ECSO system. When processing data at a Stable development stage imposes constraints on the use of IT resources in the range \([a,b]\). This is especially important for an autonomous constellation of SC. During the transition stage, a new restriction range \([h,d]\) of IT resources is settled. Setting the ranges is necessary to find a suitable alternative with the least amount of available IT resources. In the SC management case, the own state of the ECSO system is taken into account by fitness function.
The Stable development stage includes the processes of adaptation (figure 6, the points of the beginning of the process from A and the end of B) and autoscaling (the points of the beginning of the process from B and the end of C). The adaptation process is a parametric change of the system in order to complete the task. The Transition stage includes the processes of bifurcation (the points of the beginning of the process from C and the end of D) and decision-making (the points of the beginning of the process from D and the end of F). The bifurcation process is associated with the generation of possible alternatives for decision-making, and decision-making is responsible for the further development of the ECSO system.

The change in configuration of the IT resources depends on the criteria of the fitness function, which characterizes the quality of the configuration. For example, on the complexity factor $f(\Omega)$, which either remains constant, or increases or decreases. The amount of change depends on the current configuration of the system and the external influences. The sequence of cycle iterations transfers the system from the current configuration to the next and actually sets the trajectory of the system movement.

4. Conclusion
Strategic Planning for a constellation of autonomous SC with edge infrastructure is considered. For simulation of IT resources cluster forming of ECSO, the migrated master node model are introduced. The model based on the graph dynamics approach. The methodology ECSO provides a guideline to direct search of the solution. The proposed steps are new, but they remain iterative and incremental, as in the already known methodologies. The methodology's uniqueness is that it uses stages of self-organizing processes.

Further research entails the development of a conceptual framework through involvement in it terms of graph-based dynamic analysis [40]. The migrated master node model will be the subject of comprehensive research taking into account cloud, fog, and edge infrastructures. Identification of its properties and development of methods of analysis is intended.
This will serve as a further basis for both methodology development and its application to solve existing problems in space exploration.

References
[1] Boucher P 2020 Artificial intelligence: how does it work, why does it matter, and what can we do about it? Panel for the Future of Science and Technology. Scientific Foresight Unit (STOA) of European Parliament, June 2020, Available at: https://www.europarl.europa.eu/thinktank/en/document.html?reference=EPRS_STU(2020)641547
[2] Wander A, Konstantinidis K, Förstner R and Voigt P 2019 Autonomy and operational concept for self-removal of spacecraft: status detection, removal triggering and passivation. Acta Astronaut. 164 92 doi.org/10.1016/j.actaastro
[3] Balukhto A N and Romanov A A 2019 Artificial intelligence in space technology: state, development prospects. Rocket-space Device Engineering and Information Systems. 6 65 doi.org/10.30894/issn2409-0239.2019.6.1.65.75
[4] Tipaldi M, Feruglio L, Denis P and D’Angelo D 2020 On applying AI-driven flight data analysis for operational spacecraft model-based diagnostics. Annu Rev. Control. 49 197 doi.org/10.1016/j.arcontrol.2020.04.012
[5] Semenov A S 2013 Simulation Self-organised Evolution Processes: Fractal-oriented Approach (Moscow: MAI press) p 160 [in Russian]
[6] Skilton M and Hovsepian F 2018 The Fourth Industrial Revolution: Responding to the Impact of Artificial Intelligence on Business (Palgrave: Macmillan) p 342
[7] Cosmas K K 2020 Utilization of FPGA for onboard inference of landmark localization in CNN-Based spacecraft pose estimation. Aerospace. 7 (11) 159 doi.org/10.3390/aerospace7110159
[8] Schuster M J, et al. 2019 Towards autonomous planetary exploration. J. Intell. Robot. Syst. 93 461 doi.org/10.1007/s10846-017-0680-9
[9] Gao Y and Chien S 2017 Review on space robotics: toward top-level science through space exploration. Science Robotics 2 (7) eaan5074. DOI: 10.1126/scirobots.aan5074
[10] Meunier R 2016 AI and space exploration: the age of adaptability. AI and Space Exploration Available at: https://medium.com/@SAPCAI/ai-and-space-exploration-the-age-of-adaptability-66f7a0d84e49
[11] Abadicio M 2019 Artificial intelligence at NASA – current projects and applications. Emerj Available at: https://emerj.com/ai-sector-overviews/artificial-intelligence-at-nasa-current-projects-and-applications/
[12] Schmelzer R 2020 How is ai helping to commercialize space? Forbes Media LLC. Available at: https://www.forbes.com/sites/cognitiveworld/2020/03/21/how-is-ai-helping-to-commercialize-space/#6ce317587c9f
[13] International Space Activities – Available at: SpacePolicyOnline.com
[14] AIKO: Autonomous Satellite Operations Thanks to Artificial Intelligence 2019. Available at: ESA - AIKO: Autonomous satellite operations thanks to Artificial Intelligence
[15] Liu J, Gao Q, Liu Z and Li Y 2016 Attitude control for astronaut assisted robot in the space station. Int. J. Control Autom. 14(4) 1082 doi.org/10.1007/s12555-014-0568-4
[16] Bo W, Zhihong J, Hui Li, Que D, Wencheng N and Qiang H 2016 Adaptive impedance controller for a robot astronaut to climb stably in a space station. Int. J. Adv. Robot Syst. 13(3) 1 DOI: 10.5772/63544
[17] Beatrice A 2020 Curiosity artificial intelligence: a know-all guide. Analytics Insight, Magazine. Available at: https://www.analyticsinsight.net/curiosity-artificial-intelligence-know-guide/
[18] Pisleag T 2019 International competition in space, political and technological challenges in the 21st century. Acta Universitatis Danubius https://www.ceeol.com/search/article-detail?id=932756
[19] Moltz J C 2019 The changing dynamics of twenty-first-century space power. *Journal of Strategic Security* **12** (1) 15. https://doi.org/10.5038/1944-0472.12.1.1729

[20] Fehling C, Leymann F, Retter R, Schupeck W and Arbitter P 2014 *Cloud Computing Patterns. Fundamentals to Design, Build, and Manage Cloud Applications* (Wien: Springer-Verlag) p 367

[21] Akintoye S B and Bagula A 2017 Optimization of virtual resources allocation in cloud computing environment. *IEEE AFRICON* **2017** 873 doi: 10.1109/AFRICON.2017.8095597.

[22] Espadas J, Molina A, Jiménez G, Molina M, Ramirez R and Concha D 2013 A tenant-based resource allocation model for scaling Software-as-a-Service applications over cloud computing infrastructures. *Future Gener. Comp. Sy.* **29** 273 doi.org/10.1016/j.future.2011.10.013.

[23] Shahdi-Pashaki S, Teymourian E, Kayvanfar V, Komaki M GH and Sajjadi A 2015 Group technology-based model and cuckoo optimization algorithm for resource allocation in cloud computing. *IFAC PapersOnLine* **48** 1140 doi.org/10.1016/j.ifacol.2015.06.237.

[24] Fatemeh E, Seyed M B 2021 Federated geo-distributed clouds: optimizing resource allocation based on request type using autonomous and multi-objective resource sharing model. *Big Data Research* **24** 2214 doi.org/10.1016/j.bdr.2021.100188

[25] Dubey K and Sharma S C 2020 An extended intelligent water drop approach for efficient VM allocation in secure cloud computing framework. *Journal of King Saud University – Computer and Information Sciences* in press. https://doi.org/10.1016/j.jksuci.2020.11.001

[26] Kapil N V and Girish P B 2019 Optimal container resource allocation in cloud architecture: A new hybrid model. *Journal of King Saud University – Computer and Information Sciences* in press https://doi.org/10.1016/j.jksuci.2019.10.009

[27] Saxena D and Singh A K 2021 A proactive autoscaling and energy-efficient VM allocation framework using online multi-resource neural network for cloud data center. *Neurocomputing* **426** 248 doi.org/10.1016/j.neucom.2020.08.076

[28] J Dunne and D Malone 2017 Obscured by the cloud: A resource allocation framework to model cloud outage events. *J. Syst. Software* **131** 218 doi.org/10.1016/j.jss.2017.06.022

[29] Sun Y, White J, Eade S and Schmidt D C 2016 ROAR: A QoS-oriented modeling framework for automated cloud resource allocation and optimization. *J. Syst. Software* **116** 146 doi.org/10.1016/j.jss.2015.08.006

[30] Baranwal G and Vidyarthi D P 2015 A fair multi-attribute combinatorial double auction model for resource allocation in cloud computing. *J. Syst. Software* **108** 60 https://doi.org/10.1016/j.jss.2015.06.025

[31] Chen X, Zhou Y, Yang L and Lv L 2021 Hybrid fog/cloud computing resource allocation: Joint consideration of limited communication resources and user credibility. *Comput. Commun.* **169** 48 doi.org/10.1016/j.comcom.2021.01.026.

[32] Li C, Sun H, Tang H and Luo Y 2019 Adaptive resource allocation based on the billing granularity in edge-cloud architecture. *Comput. Commun.* **145** 29 doi.org/10.1016/j.comcom.2019.05.014

[33] Samimi P, Teimouri Y and Mukhtar M 2016 A combinatorial double auction resource allocation model in cloud computing. *Inform. Sciences.* **357** 201 doi.org/10.1016/j.ins.2014.02.008

[34] Singhal R and Singhal A 2021 A feedback-based combinatorial fair economical double auction resource allocation model for cloud computing. *Future Gener. Compr. Sy.* **115** 780 doi.org/10.1016/j.future.2020.09.022

[35] Naha R K, Garg S, Chan A and Battula S K 2020 Deadline-based dynamic resource allocation and provisioning algorithms in Fog-Cloud environment. *Future Gener. Compr. Sy.* **104** 131 doi.org/10.1016/j.future.2019.10.018

[36] Chen X, Wang H, Ma Y, Zheng X and Guo L 2020 Self-adaptive resource allocation for cloud-based software services based on iterative QoS prediction model. *Future Gener. Compr. Sy.* **105** 287 doi.org/10.1016/j.future.2019.12.005
[37] Yuan X, Min G, Yang L T, Ding Y and Fang Q 2017 A game theory-based dynamic resource allocation strategy in geo-distributed datacenter clouds. *Future Gener. Compr. Sy.* 76 63 doi.org/10.1016/j.future.2017.04.046

[38] Espadas J, Molina A, Jiménez G, Molina M, Ramírez R and Concha D 2013 A tenant-based resource allocation model for scaling Software-as-a-Service applications over cloud computing infrastructures. *Future Gener. Compr. Sy.* 29 273 doi.org/10.1016/j.future.2011.10.013

[39] Ferretti L 2020 *Design Space Exploration in High-Level Synthesis*, PhD thesis, Università della Svizzera Italiana, , Lugano, Switzerland

[40] Semenov A S 2020 Graph-based dynamic analysis of elastic systems. *Proc. 7th Int. Conf. on Control, Decision and Information Technologies.* (Prague: IEEE) p 65 doi: 10.1109/CoDIT49905.2020.9263986