Part A: Method

1. Systematic Review Procedure

We followed the standard guidelines proposed by Kitchenham [1] and used a study protocol for our research.

1. Evaluate the necessity of conducting this systematic review, as done in the previous section;
2. Identify the key review questions for this study;
3. Develop the search strategy:
   - Primary search terms and search database
   - Inclusion and exclusion criteria
   - Quality assessment standard
4. Perform data extraction and synthesis
5. Identify research gaps and opportunities.

2. PICOC Criteria to Formulate Research Questions

We adopt a typical approach - PICOC criteria proposed by Petticrew et al. [2] to frame review questions, which formulates review questions based on 5 attributes: Population, Intervention, Comparison, Outcome, Context. Consequently, in our systematic review, we considered these 5 attributes shown in Table 1 to construct the research questions.

Table 1: Review questions framed by PICOC criteria

Population	Multi-tenancy scheduling approaches in the cloud
Intervention	Methodology/tool/technology for scheduling on multi-tenancy clouds
Comparison	The difference of scheduling approaches, especially showing evolution and improvement on traditional scheduling methods on cloud computing
Outcome	The effectiveness of scheduling approaches and implementation on multi-tenancy clouds
Context	Within the research domain of multi-tenancy clouds' scheduling, especially studies including experimental results

3. Inclusion and Exclusion Criteria

With reference to the context of prior research, papers that contained at least one relevant topic and papers that had the potential to assist in the answering of review questions were included. However, papers with the following features were excluded in this review:

1. Secondary sources, including surveys (investigation and reviews), unrefereed technical reports (theses) or tutorials (proposals) that cannot clearly address the relevant review questions
2. Duplicate or similar papers of the same study
3. Papers with incomplete results
4. Papers not written in English
5. Papers focusing on multi-tenancy but not explicitly concerned with scheduling issue.

4. Quality Assessment

The quality assessment checklist is given in Table 2.

Table 2: Quality assessment checklist of this systematic review

Number	Questions	Answer
QA1	Is the research with clear objectives?	Yes/No/Partially
QA2	Does the research achieve its objectives?	Yes/No/Partially
QA3	Are the findings or results reported?	Yes/No/Partially
QA4	Is the proposed approach of the publication described clearly?	Yes/No/Partially
QA5	Is the experimental environment depicted clearly?	Yes/No/Partially
QA6	Is the experimental result discussed or analysed?	Yes/No/Partially
QA7	Is the background or related work of the method reviewed comprehensively?	Yes/No/Partially
QA8	Is practical or industrial application value of the research presented?	Yes/No/Partially

5. Data Extraction and Synthesis

To actualise the extraction of data from our included papers more explicitly, a data extraction schema was used to collect the relevant data as shown in Table 3. The extraction schema has a set of attributes, which have a direct relationship with the predefined review questions. Each attribute has a corresponding data extraction question [3].

Table 3: Data extraction schema

Number	Data extraction attribute	Data extraction question	Related review question
DA1	Author	Who is the first author?	N/A
DA2	Paper title	What is the title of the article?	N/A
DA3	Publication year	Which year was the paper published?	N/A
DA4	Paper type	What type of the paper? (journals, conference, workshop, symposium, etc.)	N/A
DA5	Venue name	Where is the publication venue name? (Acronym of conference/ journal/workshop name, etc.)	N/A
DA6	Environments	What is the experimental platform in this study?	SRQ2
DA7	Methodologies	What methods are mainly used in this study?	SRQ1-4
DA8	Outcome	What is the outcome in this research?	SRQ1-4
DA9	Weaknesses	How to solve the drawback of current scheduling issues in multi-tenancy cloud platform?	SRQ3
DA10	Interdependencies	How to deploy efficiency scheduling policies on multi-tenancy cloud platform?	SRQ1-4

Part B: Appendix

Appendix A. Quality Evaluation of Included Papers

The references listed below correspond to those prefaced with the letter "P" throughout the paper.

Table A.4: Detailed quality assessment score of included papers

Publication	QA1	QA2	QA3	QA4	QA5	QA6	QA7	QA8	Total
P1	1	1	1	1	0.5	1	1	0.5	7
P2	1	1	1	1	1	0.5	0.5	1	7

Publication	QA1	QA2	QA3	QA4	QA5	QA6	QA7	QA8	Total
P3	1	1	1	1	1	1	1	1	8
P4	1	0.5	1	1	1	0.5	1	0.5	6.5
P5	1	1	1	1	1	1	0.5	1	7.5
P6	1	1	1	1	1	1	1	1	8
P7	1	1	1	0.5	1	1	0.5	0.5	6.5
P8	1	1	1	1	1	0.5	1	0.5	7
P9	0.5	1	1	1	1	1	1	0.5	7
P10	1	1	1	1	1	1	1	1	8
P11	0.5	1	1	1	1	1	1	1	7.5
P12	1	1	1	1	1	1	0.5	1	7.5
P13	1	1	1	0.5	1	1	1	1	7.5
P14	1	1	1	1	1	1	1	1	8
P15	1	1	1	1	0.5	1	1	0.5	7
P16	1	1	1	1	1	1	1	1	8
P17	1	1	1	1	1	1	1	0.5	7.5
P18	1	1	1	1	1	1	1	1	8
P19	1	1	1	1	1	1	1	1	8
P20	1	1	1	1	0.5	1	1	1	7.5
P21	1	1	1	1	1	1	1	1	8
P22	1	0.5	1	1	0.5	1	1	0.5	6.5
P23	1	1	0	1	0	0	1	1	5
P24	1	0.5	1	1	0.5	0.5	1	1	6.5
P25	1	1	1	1	1	1	1	1	8
P26	1	1	1	1	1	1	1	1	8
P27	1	1	1	1	1	1	1	0.5	7.5
P28	1	1	1	1	1	1	1	1	8
P29	1	1	1	1	1	1	1	1	8
P30	1	1	1	1	1	0.5	1	1	7.5
P31	0.5	1	1	1	1	1	1	0.5	7
P32	1	1	1	1	1	1	1	1	8
P33	1	1	1	1	1	1	1	1	8
P34	1	1	1	1	1	1	1	1	8
P35	1	1	1	1	1	1	1	1	8
P36	1	1	1	1	1	1	1	0.5	7.5
P37	1	1	1	0.5	1	0.5	1	0.5	6.5
P38	1	1	1	1	1	1	1	1	8
P39	1	1	1	1	1	1	1	0.5	7.5
P40	1	1	1	1	1	0.5	0.5	0.5	6.5
P41	1	1	1	0.5	1	1	1	0.5	7
P42	1	1	1	1	1	1	1	1	8
P43	1	1	1	1	1	1	1	1	8
P44	1	1	1	1	1	1	1	1	8
P45	0.5	1	1	0.5	1	1	1	0.5	6.5
P46	1	1	1	1	1	1	1	1	8
P47	0.5	1	1	1	0.5	0.5	1	0.5	6
P48	1	1	1	1	1	1	1	0.5	7.5
P49	0.5	0.5	1	1	1	0.5	1	0.5	6
P50	1	1	1	1	1	1	1	1	8
P51	1	1	1	1	1	1	1	1	8
P52	1	1	1	1	1	1	1	1	8

Publication	QA1	QA2	QA3	QA4	QA5	QA6	QA7	QA8	Total
P53	1	1	1	1	1	1	1	1	8
Total	50	51	52	50.5	49	47.5	47.5	43.5	394
Average	0.94	0.96	0.98	0.95	0.92	0.89	0.89	0.82	7.42

Appendix B. Main Contributions of Included Papers

Table B.5: Main contributions for each paper

P1 [5]	This work proposes a hybrid test database design to support SaaS customisation with two-layer database partitioning. The database is extended with a new built-in redundancy with ontology, so SaaS can recover from ontology, data or meta-data failures.
P2 [6]	This work proposes a two-tier SaaS scaling and scheduling architecture that works at both service and application levels to save resources. Several duplication strategies are proposed, including lazy duplication and pro-active duplication to achieve better system performance. Additionally, a resource allocation algorithm is proposed.
P3 [7]	This work proposes a multi-tenant oriented monitoring, detecting and scheduling architecture based on SLA for performance isolation. This architecture monitors the service quality of each tenant, discovers abnormal statuses and dynamically adjusts the use of resources based on quantising SLA parameters to ensure the full realisation of SLA tasks.
P4 [8]	This work presents a combination of a traditional scheduling mechanism and a feedback control loop based controller to ensure performance isolation while efficiently utilising the system and faster reaction to workload changes.
P5 [9]	This work introduces the architecture and prototype of a management system to handle the required resource provisioning and user request routing using distribution strategies for multi-tenants. Proposed optimisation strategies allows for the modeling of resources, multi-tenancy deployment dependencies, and users with specific demands.
P6 [10]	This work establishes formal measurements for under and over provisioning of virtualised resources in cloud infrastructures, specifically for SaaS platform deployments and proposes a resource allocation model to deploy SaaS applications by considering their multi-tenancy, thus creating a cost-effective scalable environment.
P7 [11]	This work introduces the Robust Tenant Placement and Migration Problem (RTP) and makes the case for incremental tenant placement, driven by variations in user load, and thus proposes algorithms that elastically contract and expand a cluster of in-memory databases depending on multi-tenants' behaviour over time.
P8 [12]	This work proposes a framework for the data lifecycle of an SaaS tenant in order to capture the placement determinant factors. According to the framework, a formal placement algorithm is presented based on a formal model of tenant data and a resource estimation method.
P9 [13]	This work proposes an approach which applies resource demand estimation techniques in combination with a request-based admission control for multi-tenants. The resource demand estimation is used to determine resource consumption information for individual requests. The admission control mechanism uses this knowledge to delay requests originated from tenants that exceed their allocated resource share.
P10 [14]	This work proposes a new multi-tenancy load balancing algorithm, "Server Throughput Restriction(STR)", based on the M/G/s/s+r queueing model, in order to guarantee each application's mean response time and also achieve better server throughput.

P11 [15]	This work describes an implementation of the Elastic Application Container (EAC) based lightweight resource management system architecture with the aim of supporting multi-tenant cloud use. It also presents a multi-tenancy scheduling algorithm for EAC resource provisioning.
P12 [16]	This work develops a Least-Busy VM placement scheduler and a load-aware VM placement scheduler with dynamic scaling of infrastructure for service oriented multi-tenancy applications which require rapid responses to meet application quality-of-service requirements.
P13 [17]	This work presents *MengTian*, a Java-based platform for multi tenancy cloud clusters, which is an improved multi tenancy cloud execution environment, extending the Terracotta middleware and Jikes RVM, with mechanisms to measure the application progress and fine-grained resource usage, which can drive a metric of elasticity, inspired by the return-of-investment economic notion.
P14 [18]	This work proposes a dynamic resource allocation framework that periodically re-allocates resources to tenants to maximise resource utilisation while tolerating a low risk of SLA violations, which models the resource allocation problem as a modified unbounded knapsack problem.
P15 [19]	This work presents a framework that takes the tenant workloads, their performance SLOs, and the server hardware as input that is available to the Database as a Service (DaaS) provider. It will then output a cost-effective recipe that specifies how much hardware to provision and how to schedule the tenants on each hardware resource.
P16 [20]	This work proposes an extensible dynamic provisioning framework, starting with defining a Tenancy Requirements Model (TRM) which helps map provisioned resources with tenants. The provisioned and candidate resources are also modeled by the Health Grading Model (HGM) which assists in the continuous monitoring and grading of resources based on health parameters and enables health prediction for future provisioning. Together, TRM and HGM allow for dynamic re-provisioning for existing tenants based on either changing tenancy requirements or health grading predictions.
P17 [21]	This work presents a dynamic and cost-efficient provisioning approach of multi-tenant capable system topologies based on a Monitor-Analyse-Plan-Execute (MAPE) loop concept. For workload estimation and derivation of a capable resource topology, the MAPE loop is executed regularly regarding specified time intervals, which forms a proactive dynamic provisioning approach.
P18 [22]	This work proposes a flexible compensation mechanism including coordination logic and scheduling algorithms which supports customisable and dynamically deployment compensation process for multi-tenants. The mechanism is presented by extending the states transition model of WS-BA.
P19 [23]	This work presents the *strings* scheduler and scheduling policies for GPUs as first-class schedulable entities in high-end cloud services for multi-tenants. Decomposing the scheduling problem into a combination of workload balancing and device-level scheduling, *strings* contributes scheduling policies that explicitly consider data movement to/from accelerators.
P20 [24]	This work proposes a novel combined workload and batch planning approach for multi-tenant business applications offered as service.
P21 [25]	This work proposes a novel scheduler called Symphony that enables efficient, dynamic sharing of a GPU-based heterogeneous cluster across multiple concurrently-executing client-server applications, each with arbitrary load spikes.
P22 [26]	This work proposes a GPU multi-tenancy system, named "Rain", for GPU-based servers used in cloud computing, which efficiently utilises GPUs without compromising fairness among multiple tenant applications. "Rain" uses a multi-level GPU scheduler that decomposes the scheduling problem into a combination of load balancing and per-device scheduling.
P23 [27]	This work proposes a methodological framework to manage the degree of tenancy for a microservice based multi-tenant cloud application.
P24 [28]	This work proposes a cloud, named "Edu" to support multi-tenants and make better utilisation of resources. This framework concentrates on management system and security level. Each service is provisioned to the user with a particular scheduling algorithm.

P25 [29]	This work proposes a host optimisation process while enforcing constraints, which optimises the number of hosts necessary for scheduling the VMs in a conflict-free manner, for achieving an energy-efficient datacentre cost optimisation.
P26 [30]	This work presents a prediction-driven elastic resource scaling system for multi-tenant cloud computing, named "CloudScale", which automates fine-grained elastic resource scaling for multi-tenant cloud computing infrastructures. CloudScale employs online resource demand prediction model and predicts error handling to achieve adaptive resource allocation without assuming any prior knowledge about the applications running inside the cloud.
P27 [31]	This works proposes a method for converting realistic task resource utilisation patterns directly into boxes (termed resource boxing) making them directly exploitable for theoretical scheduling and proposes four resource conversion algorithms capable of accurately representing real task utilisation patterns in the form of scheduling boxes.
P28 [32]	This work proposes Two-Dimensional Fair Queueing (2DFQ), which spreads requests of different costs across different threads and minimises the impact of tenants with unpredictable requests. This request scheduling algorithm –"2DFQ" produces fair and smooth schedules in systems that can process multiple requests concurrently.
P29 [33]	This work proposes an integrated QoS-aware resource provisioning platform based on virtualisation technology for computing, storage and network resources. Coarse-grained CPU mapping and fine-grained CPU scheduling mechanisms are proposed to enable adjustable computing power.
P30 [34]	This work proposes the use of suspend-resume mechanisms to mitigate the overhead of preemption in cluster scheduling. Instead of killing preempted jobs or tasks, this work uses a system level, application-transparent checkpointing mechanism to save the progress of jobs for resumption at a later time when resources are available.
P31 [35]	This work designs "Wisp", a framework for building SOAs that transparently adapts rate limiters and request schedulers system-wide according to operator policies to satisfy end-to-end goals while responding to changing system conditions.
P32 [36]	This work presents "HAVEN" - a system for holistic load balancing and auto scaling in a multi-tenant cloud environment that is naturally distributed and hence scalable. "HAVEN" supports multi-tenancy and takes into account the utilisation levels of different resources in the cloud as part of its load balancing and auto scaling algorithms.
P33 [37]	This work proposes "Argus", a workload-aware resource reservation framework that targets multiple resource reservations and aims to prevent performance interference, in terms of fair throughput violation, in NoSQL stores.
P34 [38]	This work proposes a novel multi-tenancy Hadoop supporting multi-tenancy features for Apache Hadoop, a large scale distributed system commonly used for processing big data. It also proposes a multi-tenant scheduler, a new development, which is necessary to provide multi-resource allocation by users and jobs.
P35 [39]	This work proposes an improved Dominant Resource Fairness (DRF) algorithm with 3-dimensional demand vector ¡CPU, memory, vdisk¿ to support disk resources as the third dominant shared resource, enhancing fairer resource sharing.
P36 [40]	This work proposes a proactive admission controller and disk scheduling framework PCOS for I/O intensive applications. By foreseeing the resource utilisation patterns of the applications while scheduling new requests on a server, PCOS enables the selection of suitable workload combination for servers to optimise disk bandwidth utilisation.
P37 [41]	This work proposes an efficient bi-criteria approach based on tenant migrations number and cost optimisation, solving iteratively for each number of migrations a repacking step for the existing resources, followed by a variable cost and size bin packing, and by a step of consolidation. This approach enables business processes to be insured with elasticity and multi-tenancy mechanism while adjusting the available resources to the dynamic load distribution.

P38 [42]	This work proposes a virtualisation framework that takes advantage of the flourishing application of distributed virtual switch (DVS), and leverages the blooming adoption of OpenFlow protocols. This work also designs an elaborately link establishment algorithm to achieve load balancing.
P39 [43]	This work proposes "DockerCap", a software-level power capping orchestrator for Docker containers that follows an Observe-Decide-Act loop structure: this allows to quickly react to changes that impact on the power consumption by managing resources of each container at run-time, to ensure the desired power cap.
P40 [44]	This work presents an autoscaling cloud computing multi-tenancy architecture performing the resource management distribution through a collection of fuzzy-based load-balancing systems. The proposed approach requires the systematic extraction of process-related information from cloud computing systems and the composition of distributed data into event logs.
P41 [45]	This work proposes the algorithm called Multi-tenant Load Distribution Algorithm for Fog Environments (MtLDF) to optimise the load balancing in Fogs environments considering specific multi-tenancy requirements (delay and priority).
P42 [46]	This work proposes a novel Sliding-Scheduled Tenant request model which enables tenants to specify the required duration of their application within a certain window, in addition to its resource requirement graph.
P43 [47]	This work proposes an SLA negotiation framework, in which the provider and the tenant define the performance objective together in a fair way. This work formally defines the cost-effective query optimisation problem, including the economic cost and the benefit.
P44 [48]	This work presents "OPTiC", a multi-tenant scheduler intended for distributed graph processing frameworks. OPTiC proposes opportunistic scheduling, whereby queued jobs can be pre-scheduled at cluster nodes when the cluster is fully busy with running jobs.
P45 [49]	This work builds "Hubbub-Scale", an elasticity controller that is reliable in the presence of performance interference and achieves predictable performance in the face of resource contention without any significant overhead.
P46 [50]	This work presents "PriDyn", a novel scheduling framework which is designed to consider I/O performance metrics of applications such as acceptable latency and convert them to an appropriate priority value for disk access based on the current system state. This framework aims to provide differentiated I/O service to various applications and ensures predictable performance for critical applications in multi-tenant cloud environment.
P47 [51]	This work proposes an SDN-empowered task scheduling system (ASETS) for HPC as a service on the cloud (HPCaaS) in the cloud as well as a novel task scheduling algorithm (SETSA [52]) that utilises Software-Defined Networking (SDN) APIs to monitor network properties in the cloud for better scheduling, aiming to address the problem of multi-tenancy within the network.
P48 [53]	This work proposes a novel cloud-based workflow scheduling (CWSA) policy for compute-intensive workflow applications in multi-tenant cloud computing environments, which helps minimise the overall workflow completion time, tardiness, cost of execution of the workflows, and utilise idle resources of cloud effectively.
P49 [54]	This paper presents an augmented Shuffled Frog Leaping Algorithm (ASFLA) based technique for resource provisioning and workflow scheduling in the Infrastructure as a service (IaaS) cloud environment.
P50 [55]	This work proposes a resource provisioning and scheduling strategy designed specifically for Workflow as a Service (WaaS) environments. The algorithm is scalable and dynamic to adapt to changes in the environment and workload.
P51 [56]	This work proposes model-driven techniques for both mapping and allocation that rely on low-overhead a priori performance modeling of tasks. Proposed scheduling algorithms are able to offer predictable and low resource needs that is suitable for elastic pay-as-you-go cloud resources, support a high input rate through high VM utilisation, and can be combined with other mapping approaches as well.

P52 [57]	This work presents a novel approach and algorithm with mathematical formula for obtaining the exact optimal number of task resources for any workload running on Hadoop MapReduce. This algorithm for optimal resource provisioning allows users to identify the best trade-off point between performance and energy efficiency on the runtime elbow curve fitted from sampled executions on the target cluster for subsequent behavioural replication.
P53 [58]	This work designs a workload sensitive server scheduling algorithm (WSSS) and checkpoint optimisation algorithm (TCC) to tolerate and eliminate the Byzantine faults before it makes any impact. The WSSS algorithm keeps track of server performance which is part of virtual clusters to help allocate best performing server to mission critical application. The TCC algorithm works to generalise the possible Byzantine error prone region through monitoring delay variation to start new virtual nodes (VNs) with previous checkpointing.

References

[1] B. Kitchenham, Procedures for performing systematic reviews, Keele, UK, Keele University 33 (2004) 2004.

[2] M. Petticrew, H. Roberts, Systematic reviews in the social sciences: A practical guide, Wiley. com, 2008.

[3] Z. Li, H. Zhang, L. O'Brien, R. Cai, S. Flint, On evaluating commercial cloud services: A systematic review, Journal of Systems and Software 86 (9) (2013) 2371–2393.

[4] A. B. Soomro, N. Salleh, E. Mendes, J. Grundy, G. Burch, A. Nordin, The effect of software engineers' personality traits on team climate and performance: A systematic literature review, Information and Software Technology 73 (2016) 52–65.

[5] W.-T. Tsai, Q. Shao, Y. Huang, X. Bai, Towards a scalable and robust multi-tenancy SaaS, in: Proceedings of 2nd ACM Asia-Pacific Symposium on Internetware, ACM, 2010, p. 8.

[6] W.-T. Tsai, X. Sun, Q. Shao, G. Qi, Two-tier multi-tenancy scaling and load balancing, in: Proceedings of 7th IEEE International Conference on e-Business Engineering (ICEBE), 2010, pp. 484–489.

[7] X. Cheng, Y. Shi, Q. Li, A multi-tenant oriented performance monitoring, detecting and scheduling architecture based on SLA, in: Proceedings of IEEE Joint Conference on Pervasive Computing (JCPC), IEEE, 2009, pp. 599–604.

[8] R. Krebs, A. Mehta, A feedback controlled scheduler for performance isolation in multi-tenant applications, in: Proceedings of 3rd IEEE International Conference on Cloud and Green Computing (CGC), IEEE, 2013, pp. 195–196.

[9] C. Fehling, F. Leymann, R. Mietzner, A framework for optimized distribution of tenants in cloud applications, in: Proceedings of 3rd IEEE International Conference on Cloud Computing (CLOUD), IEEE, 2010, pp. 252–259.

[10] J. Espadas, A. Molina, G. Jiménez, M. Molina, R. Ramírez, D. Concha, A tenant-based resource allocation model for scaling Software-as-a-Service applications over cloud computing infrastructures, Future Generation Computer Systems 29 (1) (2013) 273–286.

[11] J. Schaffner, T. Januschowski, M. Kercher, T. Kraska, H. Plattner, M. Franklin, D. Jacobs, RTP: robust tenant placement for elastic in-memory database clusters, in: Proceedings of ACM International Conference on Management of Data, ACM, 2013, pp. 773–784.

[12] K. Tang, Z. B. Jiang, W. Sun, X. Zhang, W. S. Dong, Research on tenant placement based on business relations, in: Proceedings of 7th IEEE International Conference on e-Business Engineering (ICEBE), IEEE, 2010, pp. 479–483.

[13] R. Krebs, S. Spinner, N. Ahmed, S. Kounev, Resource usage control in multi-tenant applications, in: Proceedings of 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), IEEE, 2014, pp. 122–131.

[14] H. Sun, T. Zhao, Y. Tang, X. Liu, A QoS-aware load balancing policy in multi-tenancy environment, in: Proceedings of 8th IEEE International Symposium on Service Oriented System Engineering (SOSE), IEEE, 2014, pp. 140–147.

[15] S. He, L. Guo, Y. Guo, C. Wu, M. Ghanem, R. Han, Elastic application container: A lightweight approach for cloud resource provisioning, in: Proceedings of 26th IEEE International Conference on Advanced information networking and applications (AINA), IEEE, 2012, pp. 15–22.

[16] W. Lloyd, S. Pallickara, O. David, M. Arabi, K. Rojas, Dynamic scaling for service oriented applications: Implications of virtual machine placement on IaaS clouds, in: Proceedings of IEEE International Conference on Cloud Engineering (IC2E), IEEE, 2014, pp. 271–276.

[17] J. Simao, N. Rameshan, L. Veiga, Resource-aware scaling of multi-threaded java applications in multi-tenancy scenarios, in: Proceedings of 5th IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Vol. 1, 2013, pp. 445–451.

[18] J. Zhu, B. Gao, Z. Wang, B. Reinwald, C. Guo, X. Li, W. Sun, A dynamic resource allocation algorithm for Database-as-a-Service, in: Proceedings of IEEE International Conference on Web Services (ICWS), IEEE, 2011, pp. 564–571.

[19] W. Lang, S. Shankar, J. M. Patel, A. Kalhan, Towards multi-tenant performance SLOs, in: Proceedings of 28th IEEE International Conference on Data engineering, IEEE, 2012, pp. 702–713.

[20] A. Gohad, K. Ponnalagu, N. C. Narendra, Model driven provisioning in multi-tenant clouds, in: Proceedings of IEEE SRII Global Conference (SRII), IEEE, 2012, pp. 11–20.

[21] T. Ritter, B. Mitschang, C. Mega, Dynamic provisioning of system topologies in the cloud, in: Enterprise Interoperability V, Springer, 2012, pp. 391–401.
[22] X. Ding, R. Luo, M. Hui, A multi-tenant oriented customizable compensation mechanism for workflows, in: Proceedings of 3rd IEEE International Conference on Broadband Network and Multimedia Technology (IC-BNMT), IEEE, 2010, pp. 951–955.
[23] D. Sengupta, A. Goswami, K. Schwan, K. Pallavi, Scheduling multi-tenant cloud workloads on accelerator-based systems, in: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, IEEE Press, 2014, pp. 513–524.
[24] C. Momm, W. Theilmann, A combined workload planning approach for multi-tenant business applications, in: Proceedings of 35th IEEE International Conference Workshops on Computer Software and Applications Conference Workshops (COMPSACW), IEEE, 2011, pp. 255–260.
[25] M. Rafique, S. Cadambi, K. Rao, A. R. Butt, S. Chakradhar, Symphony: A scheduler for client-server applications on coprocessor-based heterogeneous clusters, in: Proceedings of IEEE International Conference on Cluster Computing (CLUSTER), IEEE, 2011, pp. 353–362.
[26] D. Sengupta, R. Belapure, K. Schwan, Multi-tenancy on gpgpu-based servers, in: Proceedings of 7th ACM International Workshop on Virtualization Technologies in Distributed Computing, ACM, 2013, pp. 3–10.
[27] S. Kalra, Prabhakar, Towards dynamic tenant management for microservice based multi-tenant SaaS applications, in: Proceedings of the 11th ACM International Conference on Innovations in Software Engineering, ACM, 2018, p. 12.
[28] Rangavittala, Sanjay, S. Salvi, Enhanced multi-tenant architecture for DaaS, PaaS, IaaS and SaaS in Edu-Cloud: Simplifying the service provisioning in Edu-Cloud by multi-tenant architecture, in: Proceedings of 6th ACM International Conference on Computer and Communication Technology, ACM, 2015, pp. 51–56.
[29] K. Bijon, R. Krishnan, R. Sandhu, Mitigating multi-tenancy risks in IaaS cloud through constraints-driven virtual resource scheduling, in: Proceedings of 20th ACM Symposium on Access Control Models and Technologies, ACM, 2015, pp. 63–74.
[30] Z. Shen, S. Subbiah, X. Gu, J. Wilkes, Cloudscale: elastic resource scaling for multi-tenant cloud systems, in: Proceedings of 2nd ACM Symposium on Cloud Computing, ACM, 2011, p. 5.
[31] B. Primas, P. Garraghan, K. Djemame, N. Shakhlevich, Resource boxing: converting realistic cloud task utilization patterns for theoretical scheduling, in: Proceedings of 9th IEEE/ACM 9th International Conference on Utility and Cloud Computing (UCC), IEEE, 2016, pp. 138–147.
[32] J. Mace, P. Bodik, M. Musuvathi, R. Fonseca, K. Varadarajan, 2DFQ: Two-dimensional fair queuing for multi-tenant cloud services, in: Proceedings of ACM International Conference on Special Interest Group on Data Communication (SIGCOMM), ACM, 2016, pp. 144–159.
[33] Z. Li, L. Wang, Y. Zhang, T. Truong-Huu, E. S. Lim, P. M. Mohan, S. Chen, S. Ren, M. Gurusamy, Z. Qin, Integrated QoS-aware resource provisioning for parallel and distributed applications, in: Proceedings of 19th IEEE International Symposium on Distributed Simulation and Real Time Applications, IEEE Press, 2015, pp. 171–178.
[34] J. Li, C. Pu, Y. Chen, V. Talwar, D. Milojicic, Improving preemptive scheduling with application-transparent checkpointing in shared clusters, in: Proceedings of 16th ACM Middleware Conference, ACM, 2015, pp. 222–234.
[35] L. Suresh, P. Bodik, I. Menache, M. Canini, F. Ciucu, Distributed resource management across process boundaries, in: Proceedings of ACM International Symposium on Cloud Computing, ACM, 2017, pp. 611–623.
[36] R. Poddar, A. Vishnoi, V. Mann, HAVEN: Holistic load balancing and auto scaling in the cloud, in: Proceedings of 7th IEEE International Conference on Communication Systems and Networks (COMSNETS), IEEE, 2015, pp. 1–8.
[37] J. Zeng, B. Plale, Workload-aware resource reservation for multi-tenant NoSQL, in: Proceedings of IEEE International Conference on Cluster Computing (CLUSTER), IEEE, 2015, pp. 32–41.
[38] H. Won, M. C. Nguyen, M.-S. Gil, Y.-S. Moon, Advanced resource management with access control for multitenant Hadoop, Journal of Communications and Networks 17 (6) (2015) 592–601.
[39] R. Jia, J. Grundy, Y. Yang, J. Keung, H. Li, Providing fairer resource allocation for multi-tenant cloud-based systems, in: Proceedings of 7th IEEE International Conference on Cloud Computing Technology and Science (CloudCom), IEEE, 2015, pp. 306–313.
[40] N. Jain, Lakshmi, PCOS: Prescient cloud I/O scheduler for workload consolidation and performance, in: Proceedings of International Conference on Cloud Computing and Big Data (CCBD), IEEE, 2015, pp. 145–152.
[41] G. Rosinosky, S. Youcef, F. Charoy, An efficient approach for multi-tenant elastic business processes management in cloud computing environment, in: Proceedings of 9th IEEE International Conference on Cloud Computing (CLOUD), IEEE, 2016, pp. 311–318.
[42] J. Duan, Y. Yang, A load balancing and multi-tenancy oriented data center virtualization framework, IEEE Transactions on Parallel and Distributed Systems 28 (8) (2017) 2131–2144.
[43] A. Asnaghi, M. Ferroni, M. Santambrogio, DockerCap: A software-level power capping orchestrator for docker containers, in: Proceedings of 15th IEEE International Symposium on Distributed Computing and Applications for Business Engineering (DCABES), IEEE, 2016, pp. 90–97.
[44] G. Acampora, M. L. Bernardi, M. Cimitile, G. Tortora, A. Vitiello, A fuzzy-based autoscaling approach for process centered cloud systems, in: Proceedings of IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), IEEE, 2017, pp. 1–8.
[45] E. P. Neto, G. Callou, F. Aires, An algorithm to optimise the load distribution of fog environments, in: Proceedings of IEEE International Conference on Systems, Man, and Cybernetics (SMC), IEEE, 2017, pp. 1292–1297.
[46] A. Dalvandi, M. Gurusamy, K. C. Chua, Application scheduling, placement, and routing for power efficiency in cloud data centers, IEEE Transactions on Parallel and Distributed Systems 28 (4) (2017) 947–960.
[47] S. Yin, A. Hameurlain, F. Morvan, Sla definition for multi-tenant dbms and its impact on query optimization, IEEE

Transactions on Knowledge and Data Engineering.
[48] M. R. Rahman, I. Gupta, A. Kapoor, H. Ding, OPTiC: Opportunistic graph processing in multi-tenant clusters, in: Proceedings of IEEE International Conference on Cloud Engineering (IC2E), IEEE, 2018, pp. 113–123.
[49] N. Rameshan, Y. Liu, L. Navarro, V. Vlassov, Hubbub-scale: Towards reliable elastic scaling under multi-tenancy, in: Proceedings of 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), IEEE, 2016, pp. 233–244.
[50] N. Jain, Lakshmi, Pridyn: enabling differentiated i/o services in cloud using dynamic priorities, IEEE Transactions on Services Computing 8 (2) (2015) 212–224.
[51] S. Jamalian, H. Rajaei, Asets: A sdn empowered task scheduling system for hpcaas on the cloud, in: Proceedings of IEEE International Conference on Cloud Engineering (IC2E), IEEE, 2015, pp. 329–334.
[52] S. Jamalian, H. Rajaei, Data-intensive hpc tasks scheduling with sdn to enable hpc-as-a-service, in: Proceedings of 8th IEEE International Conference on Cloud Computing, IEEE, 2015, pp. 596–603.
[53] B. P. Rimal, M. Maier, Workflow scheduling in multi-tenant cloud computing environments, IEEE Transactions on Parallel and Distributed Systems 28 (1) (2017) 290–304.
[54] P. Kaur, S. Mehta, Resource provisioning and work flow scheduling in clouds using augmented shuffled frog leaping algorithm, Journal of Parallel and Distributed Computing 101 (2017) 41–50.
[55] M. Rodriguez, R. Buyya, Scheduling dynamic workloads in multi-tenant scientific workflow as a service platforms, Future Generation Computer Systems 79 (2018) 739 – 750.
[56] A. Shukla, Y. Simmhan, Model-driven scheduling for distributed stream processing systems, Journal of Parallel and Distributed Computing 117 (2018) 98 – 114.
[57] P. Nghiem, S. Figueira, Towards efficient resource provisioning in MapReduce, Journal of Parallel and Distributed Computing 95 (2016) 29 – 41.
[58] S. Chinnathambi, A. Santhanam, J. Rajarathinam, Senthilkumar, Scheduling and checkpointing optimization algorithm for byzantine fault tolerance in cloud clusters, Cluster Computing (CoRR).