

Serverless Computing: New Trends and Research Directions



Rajalakshmi Krishnamurthi , Adarsh Kumar , Sukhpal Singh Gill ,
and Rajkumar Buyya 

Abstract Serverless computing is an innovative method for the production and distribution of software since it does not rely on a centralised server management infrastructure. As a result of this, serverless computing is becoming more widespread. Instead, the cloud service provider must ensure that the code will execute as intended in the cloud environment. Because everything is taken care of automatically, developers are free to concentrate on creating code rather than establishing and maintaining the infrastructure that is necessary for their programmes to execute. This frees up more time for the developers to work on their projects. This chapter introduces serverless computing and associated technologies. Further, this work summarizes the work done in this book, recent developments and presents future directions.

Keywords Application development · Serverless computing · Serverless functions · Services · Security

R. Krishnamurthi

Department of Computer Science and Engineering, Jaypee Institute of Information Technology, Noida, India

e-mail: k.rajalakshmi@jiit.ac.in

A. Kumar

School of Computer Science, University of Petroleum and Energy Studies, Dehradun, Uttarakhand, India

e-mail: adarsh.kumar@ddn.upes.ac.in

S. S. Gill (✉)

School of Electronic Engineering and Computer Science, Queen Mary University of London, London, UK

e-mail: s.s.gill@qmul.ac.uk

R. Buyya

Cloud Computing and Distributed Systems (CLOUDS) Laboratory, School of Computing and Information Systems, The University of Melbourne, Melbourne, Australia

e-mail: rbuyya@unimelb.edu.au

1 Introduction

Serverless computing eliminates the requirement for a dedicated server farm, making it possible to manage enormous dispersed workloads. Large, geographically scattered workloads may be managed using this kind of computing, eliminating the need for a centralised data centre [1]. This computing method eliminates the need to set up a specialised network of computers to coordinate the efforts of many workers in different places. By using this technique, a cluster of computers is not needed to handle globally distributed computations. In the computer world, eliminating servers has allowed us to manage massive workloads that are distributed across several locations. It wasn't possible until now to do this. Lambda, provided by Amazon Web Services (AWS), is one of the most well-known examples of serverless computing offered by a major cloud provider. This is the case, for example, with cloud systems like Microsoft Azure and Google Cloud. The Google Cloud Platform and the Microsoft Azure cloud storage service are two examples of where this kind of technology is put to use. In this technology, large-scale computer systems provide a substantial challenge that must be controlled when it comes to the process of moving data from one function to another. This problem need to be addressed. The creation of a possible solution is required for this matter.

The major advantages of serverless technology include [1–3]:

- (i) This kind of computing is gaining traction in the corporate world because it may relieve programmers of their server maintenance obligations. So, developers may build and expand their apps without worrying about exceeding the server's resources. This paves the way for the development of innovative app features.
- (ii) When it comes to assuring the continuous good company's information technology infrastructure, business executives often run across challenges. Through the use of serverless computing, it is no longer necessary for programmers to manage the machines on which their programmes are executed. This involves keeping an eye on the server, ensuring that the operating system is up to date, and generating user accounts for all of the different user groups who will be using the server.
- (iii) The advent of cloud computing has made it possible to share software without the need for a single centralised server. It frees up money that may be used toward other endeavours, such as the development of a product that has a greater capacity for usefulness and distinction.
- (iv) Serverless cloud computing frees users from the obligation of managing their own servers, enabling them to focus their attention where it will be of the greatest benefit: on the development and improvement of valuable applications.
- (v) As serverless apps make use of an architecture external to your company, you can take advantage of only benefiting from the functions that you need to develop it. It adapts to your budgets, since its functions scale according to the number of requests that are made.
- (vi) One company that provides this kind of technology is Amazon Web Services, and one of the services it provides is called Lambda. With Lambda, you have

the tools at your disposal to turn your ideas into applications that people will find intuitive and easy to use. This is likely made possible by the presence of in-app purchases, geolocation features, user-friendly file and photo sharing, and maybe many more features. Now that serverless computing is a thing, cloud-based IT service concepts that were previously unimaginable are within reach.

- (vii) To speed up the process of product distribution, you should attempt to spend less time and effort on administrative responsibilities. Serverless computing provides this opportunity.

1.1 Motivation

The serverless paradigm makes it possible to make software in many different ways, which means it can be used in a wide range of high-performance, and always-available apps. The growth of the Internet of Things (IoT) devices, online and mobile apps, Big Data, and machine learning are just a few examples of the many different domains where serverless computing is finding expanding usage. There are many more sectors as well [3]. This is because serverless apps have their own specific needs, each of these spheres will have its own set of nuances that set it apart from the others, improved resource management practises that take into account this reality are necessary as quickly as possible. A serverless architecture may be useful for workloads that often expand and contract but still need a significant amount of computing power. The concept that will drive the next generation of cloud computing services, known as serverless computing, is garnering an increasing amount of attention. Even when their functionality and popularity increase, it is essential for serverless systems to keep the important qualities and characteristics that set them apart in the first place.

1.2 Traditional Versus Serverless Computing

An increasing number of people are looking towards serverless computing as a viable option to the conventional server and cloud-based designs. This trend may be attributable to the fact that serverless computing has become more popular since it does away with the requirement for traditional server infrastructure. Taking this kind of action is counter to accepted procedures in the world of web design. There is no longer any need for developers to set up and manage backend servers because of serverless architectures. This eliminates the need for programmers to do these mundane chores. Less money will be spent on creating and maintaining the product. The term “serverless computing” is gaining popularity because it appeals to programmers who would rather not have to worry about the care of server infrastructure. This is one of the reasons why people are using the term more often. As a direct consequence of this advancement, programmers are no longer restricted by the capabilities

of the servers they make use of in their work. As a result, investing in DevOps is a decision that will prove to be profitable, and using this strategy has the potential to save costs associated with the investment.

This chapter is organized as follows. Section 1 introduces the background of serverless computing, compares the features of serverless computing with traditional computing and discussed the need for serverless computing in present and future. Section 2 introduces the important serverless functions, architectures and computing feasibilities. Section 3 introduces the integration of serverless computing with advanced technologies. Section 3 presents resource management in serverless computing environments. Section 4 presents the integration of serverless computing with other advanced technologies. Section 5 presents the open challenges. Section 6 discusses the future directions. Finally, the conclusion is presented in Sect. 7.

2 Serverless Functions, Architectures and Computing

FaaS provides capability to execute the applications without dependent on any infrastructure and effortless managing of services to the customers. The key characteristics of the FaaS includes (i) support for wide programming languages, (ii) no overhead of administration, (iii) scalability and availability, (iv) APIs and Cloud services based triggers for execution of codes. The conglomeration of “Functions as a Service” together with the “Backend as a Service” leads to emerging of the serverless computing. The leading Serverless solutions include Amazon Web Services (AWS), Azure and Google Cloud [4].

Amazon Web Services: AWS is the top leading marketer of serverless products and cloud space provider [5]. AWS cloud services provide developers to build and execute their applications independent of infrastructure, and computing resources. The basic functionalities such as traverse, deploy and publish of serverless applications within fractions of time. AWS allows developers to (i) incorporate multiple serverless services and applications, (ii) customize the computing resources as per the user requirements and (iii) integrate variety of serverless applications [6]. In addition, visual based workflow creation, coordination, inconnection between Lambdas can be incorporated using AWS step functions.

Amazon provides several real time data processing solutions such as AWS Lambda, Amazon S3, Amazon Kinesis and Dynamo DB. Amazon kinesis provides real time streaming of data and data analytics. The Amazon databases supports NoSQL database functionalities for two formats of data storage namely (i) key value model and (ii) document type data. Further, AWS supports messaging services namely (i) Amazon SNS for Publish/Subscriber and (ii) Amazon SQS for Message Queuing.

Figure 1 depicts the serverless computing based basic web applications and its associated backends. Here, S3 can be utilized for web hosting, Lambda functions can be used to data processing and Amazon API gateways for configure the environment and Dynamo DB can be used for retrieving data.

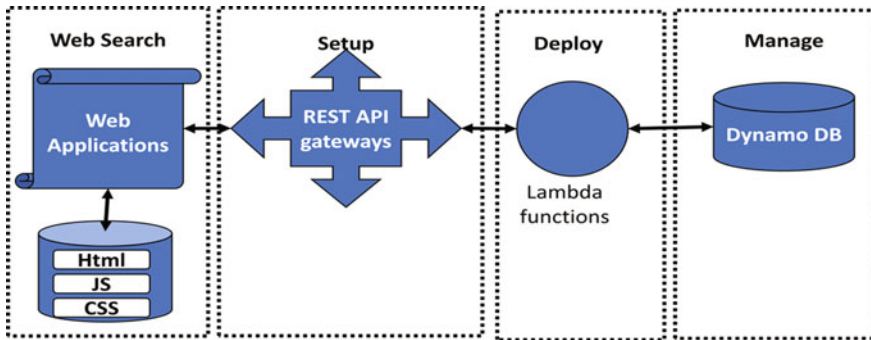


Fig. 1 Basic Amazon Web Services in serverless computing for web application

In terms of intelligent functionalities, Amazon machine learning services enables the real time prediction of hidden pattern and knowledge discovery from the processed data [7]. Similarly Amazon SageMaker provides facility to manage and deploy large scale machine learning models. Other services include Amazon Recognition for digital video and audio processing, Amazon Lex for semantic analytics and Amazon Polly for text to speech conversions [8]. Several tools are available for developers, such as AWS serverless Application Model (SAM), AWS CodeBuild and Code Deploy, CodeStar and Code Pipeline. In AWS, the Integrated Development Environment (IDE) supports several enhanced editing and project management platforms such as Cloud9IDE, Visual Studio. It also supports software development tool kits such as Python, Go, Scala, Node.js, .NET, Java programming [9].

Microsoft Azure: Second most popular Serverless computing platform is the Microsoft Azure and it provides large scale serverless computing space [10]. Azure functions are based on event driven FaaS. Particularly, Azure IoT Edge enables to excite program within the IoT edge devices even in unrealisable network connectivity. Azure storage space supports voluminous data storage, high scalability, availability and efficient durability [11].

Azure Active Directory provides strong security mechanism and access control methods for remote cloud systems. In distributed environments, the interconnectivity between public and private cloud platforms can be performed using Azure Service Bus Messaging functionalities, and Azure Event Grid can be utilized for the event based routing services. In terms of API management, the Azure Logic enables the integration of data with developer applications, and provides transparent code integration between systems without complex coding by developers. In addition, Azure Functions Proxies are capable of providing microservices through monolith APIs using single API interface.

Microsoft Azure provides intelligence support system by means of Azure Bot services for various platform namely Twitter, Microsoft teams, Slack, Skype etc. Several cognitive functionalities such as audio, video, image, speech, text based interpretation and processing are handled by Azure Intelligence services [12, 13].

From developers point of view, Microsoft Azure provides several serverless framework plugins. For example, Visual studio IDE framework permits developers to incorporate various functions and logical applications according the user specifications and requirements. In addition, Azure SDKs support almost all major software development platforms and programming languages.

Google Cloud Platform: GCP sets as the world's third top player of serverless computing [14]. The cloud function of GCP incorporates the event drive computation on serverless platform. The object based storage is involved on the GCP cloud storage units. The cloud datastores includes services such as NoSQL Database as a Service (DBaaS). For real time storage of data from IoT edge devices, the GCP provides Firebase Real time Database platform. In terms of security and privacy, Google Firebase platform supports wide variety of authentication APIs and allows user to customize their mobile applications. GCP provides visualization and management of workflow through GCP FantasM platform. Google Cloud Endpoints and APIgee API management allows developers to design, scale applications, predictive analytics and deploy securely on unreliable multiregional distributed environments [15]. GCP enables to utilize intelligence through serverless machine learning using Cloud ML Engine, vision processing through Cloud Vision API, Speech processing through Cloud Speech API, Text processing through Cloud Translational APIs. Google Cloud Function provides developers with coding event triggered applications, deploying, management and operational infrastructure.

3 Resource Management in Serverless Computing Environments

A serverless environment's resource management is the process of balancing the needs of an application's workload with those of the system, with the customer's participation kept to a minimum [16]. Given the autonomous character of the anticipated resource management process in such contexts, careful attention to every stage of this procedure is necessary to improve application and system efficiency [17]. We single out three key areas of resource management that must be addressed in a way that is appropriate for the serverless computing paradigm.

1. To characterise and anticipate workload performance, programmers would want as little effort as possible when utilising a serverless deployment paradigm. If an application's deployment requires the specification of a resource configuration and other features, the process may be laborious. Because of this, it is preferable for a serverless platform to be able to forecast performance by inferring features of applications and workloads using simulation and benchmarks methodologies [18]. Users' quality of service needs may be met with the use of an efficient strategy for establishing this comprehension, which in turn leads to improved scheduling and scalability decisions for available resources.

2. Resource scheduling is a major difficulty for both developers and cloud providers or system owners since it involves effectively allocating workloads to host nodes with the appropriate resources [16]. When the need for resources is greater than the supply, scheduling also entails deciding which programmes will run first. Even if the developer requires certain quality of service guarantees, the supplier must prioritise resource efficiency [17].
3. The serverless architecture dynamically spins up environments and distributes their resources to apps in response to incoming demands. This guarantees increased efficiency and adaptability in the use of available resources [19]. Scaling at such a fine granularity necessitates the use of sophisticated and dynamic resource scaling approaches to preserve the expected level of application performance.

4 Serverless Computing and Advanced Technologies

Using serverless computing has now become extremely prevalent for developing cloud-native apps. When it comes to cloud computing, the serverless paradigm is all about removing the burden of managing servers. Serverless computing is expected to grow at a far faster rate than traditional cloud services since developers no more need to be concerned about keeping up with infrastructure [3]. With serverless computing, cloud service providers might have an easier time handling infrastructure management and automatic provisioning. The amount of work and materials needed to maintain the infrastructure are cut down as well [20]. The goal of serverless computing is to leverage the most cutting-edge serverless technology while minimising costs and maximising returns.

AI is the potential of technology, so it's no surprise that platforms are starting to include it. Due to these AI-driven platforms, we've been able to make more accurate, timelier decisions [21]. Their impact may be seen in the altered methods of doing company, communicating with consumers, and analysing financial information. Software engineers' output and effectiveness are drastically impacted by complex machine learning algorithms. However, most of the problems that programmers face may be solved by switching to a serverless architecture. By employing a serverless architecture, both the machine learning models and their associated resources may be controlled with more efficiency and precision. Thanks to this architecture, programmers may be able to devote more time and energy to training AI models and less to maintaining servers.

Building machine learning systems is often necessary when confronting difficult problems. They perform tasks such as data analysis and pre-processing, model training, and AI model tuning [22]. Therefore, APIs should function without a hitch. The usage of serverless computing and artificial intelligence can guarantee the constant storage and transmission of data and messages. Since serverless architecture provides a number of benefits and advantages, it may be a good fit for machine learning models. Almost no management is needed to support the operation of any

kind of application or back-end service [23]. The supplier of the underlying infrastructure efficiently distributes its own CPU execution power in response to incoming requests of varying traffic loads [24]. The advantages of serverless architecture [25–28] are as follows:

1. Serverless architecture enables usage-based pricing, so you'll only ever have to fork out cash for the services you actually need. Because of this, the pricing structure is more adaptable, and the overall price is decreased.
2. Because of serverless computing, software developers may work independently and quickly. Because of this, models are treated separately from other functions. Activating this feature at any moment is completely safe and will not affect the rest of the system in any way.
3. With the advent of the "pay-per-use" model, clients are charged solely for the resources they actually employ. In serverless computing, you pay for the services you employ instead of the number of servers you utilize.
4. Serverless computing eliminates the requirement for consumers to monitor and maintain servers by making available back-end services on demand. Users of a serverless service don't have to worry about setting up or maintaining the underlying infrastructure. When using serverless architecture, service providers may easily scale up or down their bandwidth needs without making any changes to their current infrastructure.
5. Serverless programmes have gained traction as a result of their inherent reliability and fault tolerance. Because of this, you won't have to build any services to provide these capabilities to your application.

5 Open Challenges of Resource Management in Serverless Computing

We've discovered that serverless architectures provide unique difficulties in terms of resource management [16–19, 29]. The following are important challenges of resource management in serverless.

1. **Cold Start Latency:** Auto-scaling systems introduce delay since resources must be created dynamically, delaying the start of a function's execution by a significant amount of time [30]. Especially for routines with relatively brief execution periods, this early delay might have a considerable impact on application performance. Nevertheless, in order to solve this problem, several providers keep reserves of available resources.
2. **Resource Utilization:** Serverless platforms are efficient in terms of resource usage since they only pay for what is actually used by an application through its execution, as opposed to a more generic cloud computing pricing approach [17]. Nevertheless, the providers could be keeping the underlying infrastructure operational for longer. Since this is the case, it's crucial to pay extra attention to developing techniques for optimal resource utilisation by the host nodes. Customers often

overbook resources for function executions in an effort to prevent their applications from performing poorly [18]. Regularly underutilizing these resources may cause the user to receive bad price for their money and lose faith in the reliability of these services.

3. **Heterogenous Workloads:** Controlling a wide variety of workloads with little input from the user is made possible by serverless architectures. Thus, in order to provide a desirable result, such systems must learn about the specifics of the application and workload on their own [16, 29]. This is complicated by the wide variety of serverless apps in use today. It's possible that customer discontent will come from delays in resource installation time, increased resource interference impacts, and other similar issues due to a lack of knowledge of the application's requirements and features.
4. **QoS:** There are no assurances of QoS because the serverless architecture hides most internal workings from the customers [30]. While most consumers would appreciate this, platforms without appropriate performance assurances may be useless for high-precision, latency-sensitive workloads [31]. Providing a consistent level of service to all of the users in a distributed system is a hard and time-consuming undertaking for the provider.

6 Future Directions

The following are promising future directions in the area of serverless computing:

1. **Delay:** Response time in a serverless architecture is the sum of request queuing, resource setup, and function execution [31]. Although most independent functions have execution duration that are less than a second, or of just few seconds, the capability to keep low latency for function executions is a crucial challenge in serverless deployments [32]. Since the time it takes to build up resources from scratch during a serverless environment's "cold start" is typically far longer than the time it takes to actually run an application, this is the primary reason for excessive delay.
2. **Reliability:** When anything goes wrong on a serverless platform, the platform will often repeat the operation. In the event that a platform's function execution fails, it will, for instance, resubmit the request immediately [16]. It has been determined that, especially when state management is employed via external storage services, a retry-based approach to fault tolerance might still not yield right output. They stress that precision may be compromised if a partially performed failure attempt of the same execution is viewed by a parallel execution of the function [19].
3. **Sustainability:** Since serverless computing facilitates the on-demand provisioning and release of resources utilised in the execution of functions, it has been heralded in the sustainability literature as a key technology for advancing green computing [33]. As a plus, the approach of invoicing per execution time encourages programmers to reduce resource use and boost code execution speed. Nevertheless, decomposing an app into functions and the practise of configuring

resources on demand are believed to result in extra delay and an execution cost, both of which impact energy usage [34].

4. **Utilization of Resources:** Because of the granularity of the serverless billing model, users are only paid for the resources their applications really utilise [17, 18]. However, the provider is still responsible for the whole infrastructure, therefore it is in the provider's long term interest to move as many serverless apps as feasible onto a single host. Unfortunately, performance suffers when there are too many requests competing for a limited resource [31]. This is indicative of the usual tension between the goals of suppliers and customers, who each want to minimise costs while simultaneously maximising benefits [19, 30]. As a result, it's crucial to arrive at a mutually agreeable resource consolidation plan.
5. **Security:** The use of serverless computing has improved the safety of a variety of different infrastructures and computer systems. For example, Bila et al. [35] provide in-depth information about one method that may be used to secure Linux containers. There are security solutions that can detect intrusions using serverless services [35]. There are more solutions required than existing to secure sensitive information that has been saved in the cloud. This is because advancements have been made in serverless architectures. For example, advancements in authentication and authorization schemes, attacks against common execution environments, resource exhaustion attacks, and privacy concerns are some of the challenges that need to be addressed in future [1].
6. **Lack of Tools and Paradigms:** There aren't enough tools available right now to make serverless apps. This is a big challenge. Further, the use of insufficient modelling paradigms, which in turn leads to the creation of incoherent methodology, directly contributes to a drop in the overall quality of the code that has been written. Pérez et al. [2] made a new way to write code and middleware for use with serverless apps. This method could help apps that don't need a server. Benchmark suites [1] are important tools that help people figure out how likely it is that a new idea or concept will work. Thus, there is a need to focus on new tools and paradigms for serverless computing and application development.
7. **Price Standardization:** The market for serverless computing services is now controlled by several significant technology companies, each of which offers its unique price tiers and feature sets. It is expected in future that there will be an expansion not just in the number of companies offering serverless services but also in the range of price choices that are available for such services. Both of these trends are expected to occur shortly. The estimate provided by each company is one-of-a-kind because it takes into consideration a variety of criteria, such as the sort of platform it utilises, the history of client association, transparency in service and the time of day when the request is made. This is because it may be difficult to devise a pricing model for service providers, there is a need to continue looking into the matter.
8. **Data Sharing and Infrastructure Requirements:** Serverless software is composed of several distinct functions, all of which work together to provide the required functionality. It will be necessary for the functions to have some means of interacting with one another and transferring either their data or the condition they

are now in for this objective to be realised. Thus, the first challenge that has to be conquered is function addressing, and the second challenge that needs to be conquered is intercommunication for functions. Both of these challenges need to be solved. Additionally, data sharing and infrastructure requirements are related to each other because of auto-scaling, short-lived functions, and exponential growth in the usage of serverless services with an increase in the number of function copies.

9. **Other Challenges:** Serverless describes a scenario in which an application may scale automatically without adding additional servers. This goal may be achieved by completing and delivering copies of the service offerings to the appropriate customers. Given that there are no established rules for determining the places where the real copies of functions are saved, there is no way to offer an answer to this request as it is impossible to do so. Additionally, data caching, service provider management, distributed processing with different modes of execution and customized service scheduling are some of the other concerns that need to be focused upon in detail.

7 Conclusion

This work looks at how serverless computing opens up new perspectives, terminologies, architectures, service options and opportunities. Later in the work, the possibilities, different points of view and recent developments are discussed briefly. Initially, this work outline how the wide use of serverless computing technology has opened up a world of possibilities. Next, this work discussed the open issues and challenges that keep serverless services from being as good as they could be. After the current problems are fixed, serverless computing is likely to become the most popular way to do computing, overtaking cloud computing shortly.

References

1. Shafiei H, Khonsari A, Mousavi P (2022) Serverless computing: a survey of opportunities, challenges, and applications. *ACM Comput Surv (CSUR)* 54(11):1–32
2. Pérez A, Moltó G, Caballer M, Calatrava A (2019) A programming model and middleware for high throughput serverless computing applications. In: *Proceedings of the 34th ACM/SIGAPP symposium on applied computing*, Apr 2019, pp 106–113
3. Mampage A, Karunasekera S, Buyya R (2022) A holistic view on resource management in serverless computing environments: taxonomy and future directions. *ACM Comput Surv (CSUR)* 54(11s):1–36
4. Pierleoni P, Concetti R, Belli A, Palma L (2020) Amazon, Google and Microsoft Solutions for IoT: architectures and a performance comparison. *IEEE Access* 8:5455–5470. <https://doi.org/10.1109/ACCESS.2019.2961511>
5. Mathew S, Varia J (2014) Overview of Amazon Web Services. *Amazon Whitepap* 105:1–22
6. Wittig M, Wittig A (2018) *Amazon Web Services in action*. Simon and Schuster

7. Newcombe C, Rath T, Zhang F, Munteanu B, Brooker M, Deardeuff M (2015) How Amazon Web Services uses formal methods. *Commun ACM* 58(4):66–73
8. Chong N, Cook B, Eidelman J, Kallas K, Khazem K, Monteiro FR, Schwartz-Narbonne D, Tasiran S, Tautschnig M, Tuttle MR (2021) Code-level model checking in the software development workflow at Amazon Web Services. *Softw Pract Exp* 51(4):772–797
9. Jackson KR, Ramakrishnan L, Muriki K, Canon S, Cholia S, Shalf J, Wasserman HJ, Wright NJ (2010) Performance analysis of high performance computing applications on the Amazon Web Services cloud. In: 2010 IEEE second international conference on cloud computing technology and science. IEEE, pp 159–168
10. Bisong E (2019) An overview of Google Cloud Platform services. In: *Building machine learning and deep learning models on Google Cloud Platform*, pp 7–10
11. Wankhede P, Talati M, Chinchamalature R (2020) Comparative study of cloud platforms—Microsoft Azure, Google Cloud Platform and Amazon EC2. *J Res Eng Appl Sci* 5(02):60–64
12. McGlade J, Wallace L, Hally B, White A, Reinke K, Jones S (2020) An early exploration of the use of the Microsoft Azure Kinect for estimation of urban tree diameter at breast height. *Remote Sens Lett* 11(11):963–972
13. Kamal MA, Raza HW, Alam MM, Su'ud MM (2020) Highlight the features of AWS, GCP and Microsoft Azure that have an impact when choosing a cloud service provider. *Int J Recent Technol Eng* 8(5):4124–4232
14. Bataineh AS, Bentahar J, Mizouni R, Wahab OA, Rjoub G, Barachi ME (2022) Cloud computing as a platform for monetizing data services: a two-sided game business model. *IEEE Trans Netw Serv Manage* 19(2):1336–1350. <https://doi.org/10.1109/TNSM.2021.3128160>
15. Ariza J, Jimeno M, Villanueva-Polanco R, Capacho J (2021) Provisioning computational resources for cloud-based e-learning platforms using deep learning techniques. *IEEE Access* 9:89798–89811. <https://doi.org/10.1109/ACCESS.2021.3090366>
16. Li Z, Guo L, Cheng J, Chen Q, He B, Guo M (2022) The serverless computing survey: a technical primer for design architecture. *ACM Comput Surv (CSUR)* 54(10s):1–34
17. Suresh A, Somashekar G, Varadarajan A, Kakarla VR, Upadhyay H, Gandhi A (2020) Ensure: efficient scheduling and autonomous resource management in serverless environments. In: 2020 IEEE international conference on autonomic computing and self-organizing systems (ACSOS). IEEE, pp 1–10
18. Großmann M, Ioannidis C, Le DT (2019) Applicability of serverless computing in fog computing environments for IoT scenarios. In: *Proceedings of the 12th IEEE/ACM international conference on utility and cloud computing companion*, Dec 2019, pp 29–34
19. Cicconetti C, Conti M, Passarella A, Sabella D (2020) Toward distributed computing environments with serverless solutions in edge systems. *IEEE Commun Mag* 58(3):40–46
20. Mampage A, Karunasekera S, Buyya R (2021) Deadline-aware dynamic resource management in serverless computing environments. In: 2021 IEEE/ACM 21st international symposium on cluster, cloud and internet computing (CCGrid). IEEE, pp 483–492
21. Gill SS, Xu M, Ottaviani C, Patros P, Bahsoon R, Shaghaghi A, Golec M et al (2022) AI for next generation computing: emerging trends and future directions. *Internet Things* 19:100514
22. Agarwal S, Rodriguez MA, Buyya R (2021) A reinforcement learning approach to reduce serverless function cold start frequency. In: 2021 IEEE/ACM 21st international symposium on cluster, cloud and internet computing (CCGrid). IEEE, pp 797–803
23. Jonas E, Schleier-Smith J, Sreekanti V, Tsai C-C, Khandelwal A, Pu Q, Shankar V et al (2019) Cloud programming simplified: a Berkeley view on serverless computing. *arXiv preprint arXiv:1902.03383*
24. Golec M, Ozturac R, Pooranian Z, Gill SS, Buyya R (2021) iFaaSBus: a security- and privacy-based lightweight framework for serverless computing using IoT and machine learning. *IEEE Trans Ind Inform* 18(5):3522–3529
25. Castro P, Ishakian V, Muthusamy V, Slominski A (2019) The rise of serverless computing. *Commun ACM* 62(12):44–54
26. Zafeiropoulos A, Fotopoulou E, Filinis N, Papavassiliou S (2022) Reinforcement learning-assisted autoscaling mechanisms for serverless computing platforms. *Simul Model Pract Theory* 116:102461

27. Du D, Liu Q, Jiang X, Xia Y, Zang B, Chen H (2022) Serverless computing on heterogeneous computers. In: Proceedings of the 27th ACM international conference on architectural support for programming languages and operating systems, pp 797–813
28. Aslanpour MS et al (2021) Serverless edge computing: vision and challenges. In: 2021 Australasian computer science week multiconference, pp 1–10
29. Xie R, Tang Q, Qiao S, Zhu H, Yu FR, Huang T (2021) When serverless computing meets edge computing: architecture, challenges, and open issues. *IEEE Wireless Commun* 28(5):126–133
30. Djemame K (2021) Energy efficiency in edge environments: a serverless computing approach. In: International conference on the economics of grids, clouds, systems, and services. Springer, Cham, pp 181–184
31. Gill SS (2021) Quantum and blockchain based serverless edge computing: a vision, model, new trends and future directions. *Internet Technol Lett* e275
32. Baldini I, Castro P, Chang K, Cheng P, Fink S, Ishakian V, Mitchell N et al (2017) Serverless computing: current trends and open problems. In: Research advances in cloud computing. Springer, Singapore, pp 1–20
33. McGrath G, Brenner PR (2017) Serverless computing: design, implementation, and performance. In: 2017 IEEE 37th international conference on distributed computing systems workshops (ICDCSW). IEEE, pp 405–410
34. Hassan HB, Barakat SA, Sarhan QI (2021) Survey on serverless computing. *J Cloud Comput* 10(1):1–29
35. Bila N, Dettori P, Kanso A, Watanabe Y, Youssef A (2017) Leveraging the serverless architecture for securing Linux containers. In: 2017 IEEE 37th international conference on distributed computing systems workshops (ICDCSW), pp 401–404. <https://doi.org/10.1109/ICDCSW.2017.66>