Master research Internship

Bibliographic report

Function placement for FaaS applications in the fog

Domain: Distributed, Parallel, and Cluster Computing – Fog: Cloud after the edge

Author:
Volodia Parol-Guarino

Supervisor:
Nikolaos Parlavantzas
MYRIADS

INSA

Abstract: write your abstract here

Contents

1 Introduction

- Differences btw Cloud, Edge & Fog
- emphasize of the relative novelty of fog and so the confusion between fog and edge
- define Function-as-a-Service (FaaS)
- List of open questions from Kjorveziroski, Filiposka, and Trajkovik [1]
  1. Scheduling
  2. deployement
  3. performance
  4. cold start
  5. vendor lock-in
  6. security & isolation
  7. improvement to function chaining / combination (preferrably even)
  8. support for hardware acceleration (GPU, AI)
- Another approach is contributed by Xie, Tang, Qiao, *et al.* [2]. They explore the challenges from the perspective of networking, as this is the main layer that enables the Fog to exist in the first place
  1. service deployement
  2. resource awareness and service discovery
  3. service scheduling
  4. incentive mechanism
  5. exceptions and failure recovery
- Usually papers about IoT because these kind of devices are expected to take over
- Research on the metrics to optimize as well as the way to optimize them
- Serverless is still improving, especially on the biggest problems (eg. coldstart) where even the base technologies are starting to be questioned and concurrenced by faster and more precise technologies Hykes [3]
- This issue is a problem because strategies needs to precisely account for this such as Kaffes, Yadwadkar, and Kozyrakis's [4] work that emphasizes about predictabability of performances. To do so they defend a centralized approaches instead of decentralization. Main causes: burstiness of the load + cold start that need to be managed and orchestrated at a higher level
- Need to introduce to the "big question". Where to place FaaS functions in the Fog, in a place that spans from the emitter to the cloud itself, while travelling a number of heterogen nodes.
- Explain in what it difficult to solve (heterogeneity, dynamism, ownership, geographical distributions, etc.)

2 Background

- Distinguish between Fog & Edge again. Figure 7, 12 of IEEE Standards Association's [5] openfog standard
- Edge is a layer in the established model [6]
- Basically, if an equation can be made: Fog computing = cloud + edge + end-user devices as a single platform. (From Guillaume Pierre)
- Two terms are confusing, because many articles contributes to the Edge and do not consider other layers.
- Can Fog be owned by differents landlords while still collaborating (abolish vendor lock-in, not evenly distributed geopraphically or "networkally")

2.1 Context

2.2 Applications

Put it after Platforms' section

- Vehicules
- Artificial Intelligence (AI)

3 Platforms

- Several choices
- Main question is to either use already developed code to adapt to the Fog, or to create a new interoperable platform
- The focus is on adaptability, as Fog encompass and extends Cloud
- The technical problem is then to adapt the technology for a new, more demanding platform.
- a distinction can be made between commercial extension and Open-Source Software (OSS) extension. The later one can mitigate the vendor lock-in problem described by Kjorveziroski, Filiposka, and Trajkovik [1], and if not, is the only category compatible with open/reproducible research.

3.1 Commercial focus

- Existing offers from cloud giants, such as Amazon greengrass [7], Microsoft Azure IoT edge [8] and Google Iot core [9]. Elgamal's [10] work emphasize on cost optimization of serverless computations spanning from the cloud to the very edge where Greengrass can be executed on a node by a user. Though the authors use what they called an edge node, it actually corresponds to the characteritics of a fog node placed at the edge of their network.

3.2 Open-Source Software (OSS)

- Of course more transparent platforms are available, this is especially true for OSS projects such as Apache OpenWhisk [11]. It has been opened by IBM. This kind of platform is preferred in the era of open-research for obvious reasons. Adaptations have been made to retain the benefit of the source code while running in the Fog. The project is named Lean OpenWhisk [12].

- a Lot of kubernetes platforms

  1. Fission
  2. Kubeless
  3. Knative
  4. OpenFaaS
  5. Nuclio

- Other platforms are edge-native, such as the work of Pfandzelter and Bermbach [13]. This particular example shows better performances than Lean OpenWhisk. While the work is impressive, could it ever reach the industrial development level of platforms such as OpenWhisk?

- [14] is another mighty small platform placed at the edge. Thought not compared, it would extend the reach of Fog to Clusters of IoT devices by rendering devices such as Espressif's ESP8266 soc [15] able to run FaaS platforms. At the state of a proof of concept for now.

4 Placement

Feature / Article	1	2	3	4	5	6	7	8	9
	Centralization	Control level	Isolation mechanism	Placement optimization goals	Allocation mechanisms	Scheduling metrics	SLA or SLO support	Ownership	Security
Akkus, Chen, Rimac, *et al.* [16]									

4.1 Literature review

4.2 Charateristics

Introduce to the different hightlighted differences between all the methods

- centralized vs decentralized
- coarse grained vs fine grained
- isolation mechanism (VM, container, WASM, etc.)
- Placement aim (latency, trhoughtput, cost reduction, redundancy, etc.). What does it optimize?
- Allocation mechanism (fair, always a wninner, utility optimization, auctions, etc.)
- Used metrics for scheduling
- SLA support, or any kind of agreements
- Multi landlord environement?
- Security, cf Figure 16 of IEEE Standards Association's [5] openfog standard
- Usually more popupar in the litterature
- Simplifies the problem by considering the infrastructure owned by the same actor
- Simplifies the layered model: Cloud, then Edge often as a horizontal layer
- Was developed and advertised before the Fog

Usually these approach rely on a scheduler mastering a subset or the whole node network. This is the approach that is the most optimal as the master knows the whole state of the system

- matrix chain ordering problem Elgamal, Sandur, Nguyen, *et al.* [17]
- Palade, Mukhopadhyay, Kazmi, *et al.*'s [18] defines Multi-acccess Edge Computing. A server acts as an intermediary between the cloud and other sub-nodes. This is a sort of master of the underlying network. The scheduler is cloud based.
- Wang, Ali-Eldin, and Shenoy's [19] Developped a platform that uses fair share allocation mechanisms. This guarantees a minimum allocated resources to each function, in case of an overload of the node. Their platform is recommended by the authors for highly dynamic predictible workloads as it utilizes aggressive approaches to reclaim resources at the scale of hundreds of milliseconds. Sadly, this approach is tested for the edge, with more computing power than in fog. Another critic is that it only focuses on a single node.
- Not all papers are about managing the behaviours of the Fog, some focus on optimization of the execution of the functions themselves, such approches are explored by Shen, Yang, Su, *et al.* [20] that utilizes mining pattern to extract execution dependencies, thus optimizing along the way.

- Mutichiro, Tran, and Kim's [21] tested an approched focuses solely on QoS optimization
- Lee, Jeong, Masood, *et al.* [22] utilizes VCG auctions to allocate computing resources. The transactions are secured using blockchain contracts. The paper is only about a simulation as a proof of feasability and lacks proper load verification.
- Approach of blockchain, w/ replacement of the Proof-of-Work with something lighter, as highlighted by Xie, Tang, Qiao, *et al.* [2] in their survey of applied auctions an blockchained.
    1. Zavodovski, Bayhan, Mohan, *et al.* [23] created a distributed auction platform
    2. Debe, Salah, Rehman, *et al.* [24] explores reverse bidding
    3. Yu, Chen, Fu, *et al.* [25] makes sure to find a winner in the edge nodes for every requests submitted
    4. There are also mentions about deep learning-based auctions, that are optimal
    5. Mutichiro, Tran, and Kim's [21] tested an approched focuses solely on QoS optimization

4.3 Centralized vs Decentralized

4.3.1 Centralized

- Cheng, Fuerst, Solmaz, *et al.* [26] defines *Fog functions* as a FaaS platform. Nodes are edge-base or cloud-based and running the same software. Discovery and orchestration is centralized to a particular node. The orchestrator is made aware of the data flowing to its nodes. It then decides where to execute the function. Data is then sent to that node so processing can take place.

4.3.2 Decentralized

- Baresi and Filgueira Mendonça [27] theorised an edge-only platform that would collaborate with others. Their goal is to optimize latency and trhoughtputs of the applications they execute. Their platform would both be able to orchestrate the swarm vertically and horizontally, the later one being prioritized to maximize robustness and availability. Baresi, Mendonça, and Quattrocchi's [28] Partitioning, Allocation, Placement, and Scal- ing (PAPS) automatic platform is an application of their principles, at a superiror scales from their perspective. They can execute 100 functions on an intense workload. The platform divides itself into 3 layers. 1. A supervisor node that knows the global topology. Its responsability is to form communities 2. A community is an aggregation of nodes with a reduced communication delay (under a defined threshold). It has been assembled by the Supervisor node and is in charge of making sure Service Level Objectivess (SLOs) are reached and Service Level Agreements (SLAs) not violated. The functions processed by the community are allocated by a leader. This centralized approach avoids the need of a more complex protocol while using well known optimal centralized techniques. The authors note the utilization of containers in PAPS make reactivity challenging. 3. The community leader passes the responsabilty to the local node, alongside the allocated resources. Then it is up to that local node to make sure SLO is not broken. So each node actually micro-manage its allocated functions and their fluctuations the best they can, while still respectinf the leader's decision and global view.

4.4 Control level

4.5 Isolation mechanism

- [26] uses Docker to isolate each functions.

4.6 Placement optimization goal

- [26] tries to respect SLO at runtime

4.7 Allocation mechanism

4.8 Scheduling underlaying metrics

4.9 Service Level Agreement (SLA) support

A Service Level Agreement (SLA) is an agreement between the cloud provider and the client specifying uptimes, responsiveness, performance-level, issue resolution timeframe [29]. In short what the client expects from what the provider advertised. This cloud native contract would provide a well-known framework to define what to expect from a Fog network. Because Fog is decentralized, one would however need to leverage the correct toolset to make the contract known by all. Such a mechanism can be contributed by smart contracts [30]–[32].

4.10 Ownership: living in a multi landlord ecosystem

4.11 Security

4.11.1 Physical attacks

Compromised hardware can be a possibility, however harming the provider.

4.11.2 Anything else

Other risks are known, from exploitation of resource contrained policies to more problematic auxilary attacks. To not over-extend this research domain, Fog computing would only worsen issues such as coverting attacks as desmonstrated by Maurice, Weber, Schwarz, *et al.* [33]. If before the cloud provider was in charge of the virtual machine distribution to a certain extent, Fog introduces geographic constraints. Those would theoretically enable geographically-located attacks. This is a critical aspect as IEEE Standards Association [5] also points out.

5 Curated list of unresolved problems

- What did we just learn?
- Focus on some of the issues yet to solve
- Where are we going? (no papers that go there, yet...)

6 Conclusion

References

[1] V. Kjorveziroski, S. Filiposka, and V. Trajkovik, "IoT serverless computing at the edge: A systematic mapping review," *Computers*, vol. 10, no. 10, p. 130, Oct. 2021, Number: 10 Publisher: Multidisciplinary Digital Publishing Institute. DOI: 10.3390/computers10100130. [Online]. Available: https://www.mdpi.com/2073-431X/10/10/130 (visited on 11/30/2021).

[2] R. Xie, Q. Tang, S. Qiao, H. Zhu, F. R. Yu, and T. Huang, "When serverless computing meets edge computing: Architecture, challenges, and open issues," *IEEE Wireless Communications*, vol. 28, no. 5, pp. 126–133, Oct. 2021, Conference Name: IEEE Wireless Communications, ISSN: 1558-0687. DOI: 10.1109/MWC.001.2000466.

[3] S. Hykes. "Solomon Hykes sur Twitter," Twitter. (2019), [Online]. Available: https://twitter.com/solomonstre/status/1111004913222324225 (visited on 12/06/2021).

[4] K. Kaffes, N. J. Yadwadkar, and C. Kozyrakis, "Centralized core-granular scheduling for serverless functions," in *Proceedings of the ACM Symposium on Cloud Computing*, ser. SoCC '19, New York, NY, USA: Association for Computing Machinery, Nov. 20, 2019, pp. 158–164, ISBN: 978-1-4503-6973-2. DOI: 10.1145/3357223.3362709. [Online]. Available: https://doi.org/10.1145/3357223.3362709 (visited on 12/01/2021).

[5] IEEE Standards Association, "IEEE standard for adoption of OpenFog reference architecture for fog computing," *IEEE Std 1934-2018*, pp. 1–176, Aug. 2018, Conference Name: IEEE Std 1934-2018. DOI: 10.1109/IEEESTD.2018.8423800.

[6] Wikipedia, *Edge computing*, in *Wikipedia*, Page Version ID: 1057908860, Nov. 30, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Edge_computing&oldid=1057908860 (visited on 12/12/2021).

[7] "AWS IoT Greengrass - Amazon Web Services," Amazon Web Services, Inc. (), [Online]. Available: https://aws.amazon.com/fr/greengrass/ (visited on 12/06/2021).

[8] "IoT edge | cloud intelligence | microsoft azure." (), [Online]. Available: https://azure.microsoft.com/en-us/services/iot-edge/ (visited on 12/06/2021).

[9] "Cloud IoT core," Google Cloud. (), [Online]. Available: https://cloud.google.com/iot-core (visited on 12/06/2021).

[10] T. Elgamal, "Costless: Optimizing cost of serverless computing through function fusion and placement," in *2018 IEEE/ACM Symposium on Edge Computing (SEC)*, Oct. 2018, pp. 300–312. DOI: 10.1109/SEC.2018.00029.

[11] "Apache OpenWhisk is a serverless, open source cloud platform." (), [Online]. Available: https://openwhisk.apache.org/ (visited on 12/06/2021).

[12] D. Breitgand. "Lean OpenWhisk: Open source FaaS for edge computing," Apache OpenWhisk. (Jul. 25, 2018), [Online]. Available: https://medium.com/openwhisk/lean-openwhisk-open-source-faas-for-edge-computing-fb823c6bbb9b (visited on 11/15/2021).

[13] T. Pfandzelter and D. Bermbach, "tinyFaaS: A lightweight FaaS platform for edge environments," in *2020 IEEE International Conference on Fog Computing (ICFC)*, Apr. 2020, pp. 17–24. DOI: 10.1109/ICFC49376.2020.00011.

[14] G. George, F. Bakir, R. Wolski, and C. Krintz, "NanoLambda: Implementing functions as a service at all resource scales for the internet of things.," in *2020 IEEE/ACM Symposium on Edge Computing (SEC)*, San Jose, CA, USA: IEEE, Nov. 2020, pp. 220–231, ISBN: 978-1-72815-943-0. DOI: 10.1109/SEC50012.2020.00035. [Online]. Available: https://ieeexplore.ieee.org/document/9355717/ (visited on 11/15/2021).

[15] "ESP8266 wi-fi MCU i espressif systems." (), [Online]. Available: https://www.espressif.com/en/products/socs/esp8266 (visited on 12/06/2021).

[16] I. E. Akkus, R. Chen, I. Rimac, M. Stein, K. Satzke, A. Beck, P. Aditya, and V. Hilt, "{SAND}: Towards high-performance serverless computing," presented at the 2018 {USENIX} Annual Technical Conference ({USENIX} {ATC} 18), 2018, pp. 923–935, ISBN: 978-1-939133-01-4. [Online]. Available: https://www.usenix.org/conference/atc18/presentation/akkus (visited on 11/15/2021).

[17] T. Elgamal, A. Sandur, P. Nguyen, K. Nahrstedt, and G. Agha, "DROPLET: Distributed operator placement for IoT applications spanning edge and cloud resources," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, San Francisco, CA, USA: IEEE, Jul. 2018, pp. 1–8, ISBN: 978-1-5386-7235-8. DOI: 10.1109/CLOUD.2018.00008. [Online]. Available: https://ieeexplore.ieee.org/document/8457776/ (visited on 11/29/2021).

[18] A. Palade, A. Mukhopadhyay, A. Kazmi, C. Cabrera, E. Nomayo, G. Iosifidis, M. Ruffini, and S. Clarke, "A swarm-based approach for function placement in federated edges," in *2020 IEEE International Conference on Services Computing (SCC)*, ISSN: 2474-2473, Nov. 2020, pp. 48–50. DOI: 10.1109/SCC49832.2020.00013.

[19] B. Wang, A. Ali-Eldin, and P. Shenoy, "LaSS: Running latency sensitive serverless computations at the edge," *Proceedings of the 30th International Symposium on High-Performance Parallel and Distributed Computing*, pp. 239–251, Jun. 21, 2021. DOI: 10.1145/3431379.3460646. arXiv: 2104.14087. [Online]. Available: http://arxiv.org/abs/2104.14087 (visited on 11/15/2021).

[20] J. Shen, T. Yang, Y. Su, Y. Zhou, and M. R. Lyu, "Defuse: A dependency-guided function scheduler to mitigate cold starts on FaaS platforms," in *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*, ISSN: 2575-8411, Jul. 2021, pp. 194–204. DOI: 10.1109/ICDCS51616.2021.00027.

[21] B. Mutichiro, M.-N. Tran, and Y.-H. Kim, "QoS-based service-time scheduling in the IoT-edge cloud," *Sensors*, vol. 21, no. 17, p. 5797, Jan. 2021, Number: 17 Publisher: Multidisciplinary Digital Publishing Institute. DOI: 10.3390/s21175797. [Online]. Available: https://www.mdpi.com/1424-8220/21/17/5797 (visited on 12/01/2021).

[22] Y. Lee, S. Jeong, A. Masood, L. Park, N.-N. Dao, and S. Cho, "Trustful resource management for service allocation in fog-enabled intelligent transportation systems," *IEEE Access*, vol. 8, pp. 147 313–147 322, 2020, Conference Name: IEEE Access, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3015550.

[23] A. Zavodovski, S. Bayhan, N. Mohan, P. Zhou, W. Wong, and J. Kangasharju, "DeCloud: Truthful decentralized double auction for edge clouds," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, ISSN: 2575-8411, Jul. 2019, pp. 2157–2167. DOI: 10.1109/ICDCS.2019.00212.

[24] M. Debe, K. Salah, M. H. U. Rehman, and D. Svetinovic, "Blockchain-based decentralized reverse bidding in fog computing," *IEEE Access*, vol. 8, pp. 81 686–81 697, 2020, Conference Name: IEEE Access, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.2991261.

[25] B. Yu, Y. Chen, S. Fu, W. Yu, and X. Guo, "Building trustful crowdsensing service on the edge," in Jun. 21, 2019, pp. 445–457, ISBN: 978-3-030-23596-3. DOI: 10.1007/978-3-030-23597-0_36.

[26] B. Cheng, J. Fuerst, G. Solmaz, and T. Sanada, "Fog function: Serverless fog computing for data intensive IoT services," in *2019 IEEE International Conference on Services Computing (SCC)*, ISSN: 2474-2473, Jul. 2019, pp. 28–35. DOI: 10.1109/SCC.2019.00018.

[27] L. Baresi and D. Filgueira Mendonça, "Towards a serverless platform for edge computing," in *2019 IEEE International Conference on Fog Computing (ICFC)*, Jun. 2019, pp. 1–10. DOI: 10.1109/ICFC.2019.00008.

[28] L. Baresi, D. Mendonça, and G. Quattrocchi, "PAPS: A framework for decentralized self-management at the edge," in Oct. 22, 2019, pp. 508–522, ISBN: 978-3-030-33701-8. DOI: 10.1007/978-3-030-33702-5_39.

[29] Wikipedia, *Service-level agreement*, in *Wikipedia*, Page Version ID: 1045280242, Sep. 19, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Service-level_agreement&oldid=1045280242 (visited on 12/13/2021).

[30] L. Hang and D.-H. Kim, "SLA-based sharing economy service with smart contract for resource integrity in the internet of things," *Applied Sciences*, vol. 9, no. 17, p. 3602, Jan. 2019, Number: 17 Publisher: Multidisciplinary Digital Publishing Institute. DOI: 10.3390/app9173602. [Online]. Available: https://www.mdpi.com/2076-3417/9/17/3602 (visited on 12/13/2021).

[31] E. Di Pascale, J. McMenamy, I. Macaluso, and L. Doyle, "Smart contract SLAs for dense small-cell-as-a-service," *arXiv:1703.04502 [cs]*, Mar. 13, 2017. arXiv: 1703.04502. [Online]. Available: http://arxiv.org/abs/1703.04502 (visited on 12/13/2021).

[32] H. Zhou, C. de Laat, and Z. Zhao, "Trustworthy cloud service level agreement enforcement with blockchain based smart contract," in *2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, ISSN: 2330-2186, Dec. 2018, pp. 255–260. DOI: 10.1109/CloudCom2018.2018.00057.

[33] C. Maurice, M. Weber, M. Schwarz, L. Giner, D. Gruss, C. A. Boano, S. Mangard, K. Roemer, and S. Mangard, "Hello from the other side: SSH over robust cache covert channels in the cloud," in *Proceedings 2017 Network and Distributed System Security Symposium*, San Diego, CA: Internet Society, 2017, ISBN: 978-1-891562-46-4. DOI: 10.14722/ndss.2017.23294. [Online]. Available: https://www.ndss-symposium.org/ndss2017/ndss-2017-programme/hello-other-side-ssh-over-robust-cache-covert-channels-cloud/ (visited on 12/13/2021).