# Some Placement Techniques of Test Components Inspired by Fog Computing Approaches

Moez Krichen, Wilfried Yves Hamilton Adoni, Tarik Nahhal

# Some Placement Techniques of Test Components Inspired by Fog Computing Approaches

Moez Krichen[1], Wilfried Yves Hamilton Adoni[2], and Tarik Nahhal[2]

[1] FCSIT, Albaha University, Albaha, Saudi Arabia
ReDCAD Laboratory, University of Sfax, Tunisia
moez.krichen@redcad.org
[2] Hassan II University of Casablanca, Morocco
adoniwilfried@gmail.com, t.nahhal@fsac.ac.ma

**Abstract.** In this work we are interested in placing test components for Internet of Things (IoT) and Smart Cities. Our work is inspired by similar works aiming the placement of application components in Fog computational nodes. First we give an overview about the decision variables to consider. Then, we define several types of constraints that may be included in the placement problem. Moreover, We list a set of possible Objectives Functions to maximize or minimize. Finally, we propose some algorithms and techniques to solve the considered Test Component Placement Problem (TCPP) taken from the literature.

**Keywords:** Internet of Things, Smart Cities, Test Components, Placement, Optimisation, Constraints, Objective Functions, Algorithms, Fog.

## 1 Introduction

The city of the future is more sustainable, cleaner, safer and smarter than ever. Full networks and smart cities offer versatile solutions to current challenges such as environmental protection, growing traffic volumes and increasing urbanization. In particular, the Internet of Things (IoT) plays a key role here, as it links devices, programs and users together. In Smart Cities cars communicate with houses, houses with digital devices, and these in turn communicate with the city's inhabitants. Most errors and problems of IoT applications only become apparent during actual use by the customer. This is why a classic integration test is usually not enough. In addition, it is hardly possible in the laboratory to reproduce the variety of devices and operating systems combinations. Moving testings to the real world [22, 20, 16, 18] also helps counteract operational blindness. Because some usability problems are not noticed by the developers, because they deal with the product every day. Often, real users interact with the application differently than thought, resulting in unexpected bugs. That is why it is important to test under real conditions with potential end users.

Test Components can be placed either locally on the same computational node under test or remotely on the cloud. An other half-way alternative consists

in using Fog computing techniques which allow to distribute application components -test components in our case- on the different available computational nodes in the Fog. In some of previous works [13, 11, 12] we proposed a formal work for testing security aspects of IoT and Smartc Cities. In [25, 21, 15], we were interested in placement problems of test components for distributed and dynamic distributed systems in general. In the present work we concentrate on test component placement problem (TCPP) for IoT and Smart Cities. We are mainly inspired by the work of Brogi et al. [4]. The remaining part of the paper is organised as follows. Section 2 provides some preliminaries about IoT, Testing, Fog computing and placement problem. In section 3, we deal with decision variables and different types of constraints used in the placement problem. Section 4 lists a set of possible Objective Functions. Section 5 proposes a set of algorithms and techniques to solve TCPP. Finally Section 6 concludes the paper.

## 2 Preliminaries

### 2.1 Fog Computing

After Cloud Computing [19, 17] now Fog Computing is to come. The goal of Fog Computing is to bring the analysis, processing and storage functions from the cloud back to the edge of the network. The more networked our world becomes, the more data is processed. The communication path into a cloud computing center and then back to the terminal leads to high latencies - for example for autonomously driving cars whose information processing must be guaranteed in real time.

Some advantages of Fog Computing are as follows:

- Faster data processing due to reduced network traffic.
- Networked devices in the IoT also work when the Internet goes down or the cloud connection causes delays.
- Sensitive company and customer data would not have to be transferred to the cloud and can remain on the spot.

### 2.2 Test Component Placement Problem (TCPP)

Let $TC$ be the set of test components we aim to distribute. The set of requirements to satisfy by the placement strategy is called $Req$ and the distributed architecture inside which the test components will be placed is called $Infr$. Possible Solutions to the *Test Component Placement Problem* (TCPP) are mappings from the set of test components $TC$ to the set of computational nodes $Infr$, satisfying the requirements fixed by $Req$ and maximizing/minimizing a set of objective functions $Obj$ used to measure their quality. The adopted Solutions are many-to-many. That is a test component can be assigned to one or many nodes and a computational node can execute one or many test components.

### 2.3 Decision Variables

The values of these variables need to be calculated during the optimisation procedure. They also define the requirements $Req$ to satisfy. Values of the variables which do not satisfy the constraints are rejected and can not be considered as acceptable solutions. For TCPP, they may be binary decision variables which indicate if a test component of $TC$ is allocated to a computational node of $Infr$.

## 3 Different Constraints

### 3.1 Energy Constraints

The authors of [2] characterised the fog nodes with their energy capacities. Devices with batteries are considered in order to guarantee lifetime constraints. The authors of [30] introduced the concept of energy cells to calculate the energy consumed by the computational devices. The optimisation procedure considers the number of available energy cells. The goal is to minimise energy consumption in constrained nodes.

### 3.2 Network Constraints

Network latency, bandwidth, link reliability and topology are examples of network constraints. For instance on the one hand, the authors of [26] considered only latency constraint. On the other hand in [8], the authors considered both latency and bandwidth. Bandwidth was also taken into account in [32, 1] along with hardware requirements.

### 3.3 Node Constraints

This type of constraints correspond to the resource capacity of the computational nodes. Resources are usually modelled as a vectors of elements like CPU, RAM, storage, etc. For instance the authors of [8] considered only CPU and the authors of [38] considered only storage. In [1], the authors considered CPU and storage. In [6], RAM, CPU and storage were all taken into account.

### 3.4 Application Constraints

Three types of application constraints have been considered for the classification of the papers in the survey: constraints related to the dependencies between the modules or services of the applications; the workload generated over the applications; and if the users are able to define any kind of preference in the deployment of the services.

# 4 Objective Functions

## 4.1 Network Delay and Execution Time

For instance this objective function is considered in [39]. Similarly the authors of [37] minimised the response time to increase the number of requests which are served before a fixed deadline. Moreover, the authors of [31] diminished the computation time of the service by calculating the consumed time slots.

## 4.2 Energy

The optimisation of energy was considered from different aspects. For instance, the authors of [2] proposed a linear function of the cost of energy and they optimised it. Similarly in [9], the authors were interested in reducing the cost of communication energy. Their approach was mainly based on placing interrelated applications and services on the same device. This choice allows to diminish communications between computational nodes.

## 4.3 Cost

In Fog computing the consideration of cost aspects is at an initial phase. In [1] the authors aimed to minimize the total cost of applications deployment by assigning them to the computational nodes with have optimal costs. Similarly the authors of [35] considered cost minimisation of each device and link.

## 4.4 Migrations

This objective function consists in minimizing migrations number and/or the effects of migrations on the system. In [27], the authors optimised the number of migrations by minimising the network use without influencing the network latency. Similarly, the authors of [38] optimised the number of migrations along with resource usage and latency. Moreover, a technique was presented in [5] to reduce the number of migrated applications between computational nodes.

## 4.5 QoS-assurance

In [26], the objective of the optimisation procedure was to diminish the number of active Fog nodes. The QoS requirement considered in this work consisted in obtaining execution times which are shorter than the deadlines of the application. In addition, the authors of [34] considered QoS aspects realted to the frequency of executed requests and transactions.

# 5 Algorithms

## 5.1 Mathematical Programming

Mathematical programming is usually used to find solutions for optimisation problems by exploring the domain of the objective function. Many approaches [38, 6] tackled the placement problem using this mathematical procedure by exploiting ILP (Integer Linear Programming), MILP (Mixed-Integer Linear Programming) or MINLP (Mixed-Integer Non-Linear Programming).

## 5.2 Dynamic Programming

The authors of [30] modelled the placement problem as a multidimensional knapsack problem with the goal objective of optimising a particular objective function. Similarly in [28], the authors modelled the placement problem as a knapsack instance, by considering the assignment of application services to running computational nodes in a Fog infrastructure.

## 5.3 Search-based Algorithms

The authors of [8] proposed a search algorithm to compute a placement solution of IoT applications to a tree-structured infrastructure. Further in [7] proposed a distributed search strategy to compute the best placement solution in the Fog, which corresponds to the minimal distance between clients and requested services. In addition the authors of [3] proposed a greedy and exhaustive backtracking algorithm to solve the placement problem.

## 5.4 Genetic Algorithms

Genetic algorithms use meta-heuristics to deal with search and optimisation problems. In [36], the authors provided a description of the use of genetic algorithms and parallel genetic algorithms to solve to placement problems. Similarly, the authors of [29] also proposed a genetic algorithm approach solution implemented in iFogSim.

## 5.5 Deep Learning

The authors of [33] used recent learning techniques to solve placement problems. After defining a Markov Decision Process to optimize power consumption, communication delay and migration costs, a deep learning algorithm is provided to support migration of applications hosted in computational nodes. The proposed approach takes into consideration the mobility of the user.

### 5.6 Game Theory

The authors of [40] modelled the palcement problem as a set of games.The first game is defined to find the number of computing blocks that users have to buy. The second game is defined for providers to set prices so to maximise financial profits. A matching game is proposed to map providers to computational nodes. Finally, matching between subscribers and computational nodes is refined.

## 6 Conclusion

Although the smart city is still considered as a future scenario, it will soon become a reality for future generations. How people move in cities in the future and how they live there will change significantly as a result of increasing networking.

In this work we were interested in the test component placement problem for these modern technologies. We identified existing techniques in the literature used for the placement of application components in Fog domain. In the future we need to adapt the found methods to the case of TCPP for IoT and Smart cities and to implement them. We may also use optimization techniques like in [10] and combine functional and load tests as proposed in [14, 23, 24].

## References

1. Arkian, H.R., Diyanat, A., Pourkhalili, A.: Mist: Fog-based data analytics scheme with cost-efficient resource provisioning for iot crowdsensing applications. Journal of Network and Computer Applications **82** (2017) 152 – 165
2. Barcelo, M., Correa, A., Llorca, J., Tulino, A.M., Vicario, J.L., Morell, A.: Iot-cloud service optimization in next generation smart environments. IEEE Journal on Selected Areas in Communications **34**(12) (Dec 2016) 4077–4090
3. Brogi, A., Forti, S.: Qos-aware deployment of iot applications through the fog. IEEE Internet of Things Journal **4**(5) (Oct 2017) 1185–1192
4. Brogi, A., Forti, S., Guerrero, C., Lera, I.: How to place your apps in the fog - state of the art and open challenges. CoRR **abs/1901.05717** (2019)
5. Filiposka, S., Mishev, A., Gilly, K.: Community-based allocation and migration strategies for fog computing. In: 2018 IEEE Wireless Communications and Networking Conference (WCNC). (April 2018) 1–6
6. Gu, L., Zeng, D., Guo, S., Barnawi, A., Xiang, Y.: Cost efficient resource management in fog computing supported medical cyber-physical system. IEEE Transactions on Emerging Topics in Computing **5**(1) (Jan 2017) 108–119
7. Guerrero, C., Lera, I., Juiz, C.: A lightweight decentralized service placement policy for performance optimization in fog computing. Journal of Ambient Intelligence and Humanized Computing **10**(6) (Jun 2019) 2435–2452
8. Gupta, H., Dastjerdi, A.V., Ghosh, S.K., Buyya, R.: ifogsim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments. Softw., Pract. Exper. **47**(9) (2017) 1275–1296
9. Huang, Z., Lin, K.J., Yu, S.Y., jen Hsu, J.Y.: Co-locating services in iot systems to minimize the communication energy cost. Journal of Innovation in Digital Ecosystems **1**(1) (2014) 47 – 57

10. Krichen, M.: Improving formal verification and testing techniques for internet of things and smart cities. Mobile Networks and Applications (Sep 2019)
11. Krichen, M., Alroobaea, R.: A new model-based framework for testing security of iot systems in smart cities using attack trees and price timed automata. In: Proceedings of the 14th International Conference on Evaluation of Novel Approaches to Software Engineering, ENASE 2019, Heraklion, Crete, Greece, May 4-5, 2019. (2019) 570–577
12. Krichen, M., Cheikhrouhou, O., Lahami, M., Alroobaea, R., Jmal Maâlej, A.: Towards a model-based testing framework for the security of internet of things for smart city applications. In: Smart Societies, Infrastructure, Technologies and Applications, Cham, Springer International Publishing (2018) 360–365
13. Krichen, M., Lahami, M., Cheikhrouhou, O., Alroobaea, R., Maâlej, A.J. In: Security Testing of Internet of Things for Smart City Applications: A Formal Approach. Springer International Publishing, Cham (2020) 629–653
14. Krichen, M., Maâlej, A.J., Lahami, M.: A model-based approach to combine conformance and load tests: an ehealth case study. IJCCBS **8**(3/4) (2018) 282–310
15. Krichen, M., Maâlej, A.J., Lahami, M., Jmaiel, M.: A resource-aware model-based framework for load testing of ws-bpel compositions. In Hammoudi, S., Śmiałek, M., Camp, O., Filipe, J., eds.: Enterprise Information Systems, Cham, Springer International Publishing (2019) 130–157
16. Lahami, M., Krichen, M.: Test isolation policy for safe runtime validation of evolvable software systems. In: 2013 Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises. (June 2013) 377–382
17. Lahami, M., Krichen, M., Alroobaea, R.: Towards a test execution platform as-a-service: Application in the e-health domain. In: 2018 International Conference on Control, Automation and Diagnosis (ICCAD). (March 2018) 1–6
18. Lahami, M., Fakhfakh, F., Krichen, M., Jmaiel, M.: Towards a ttcn-3 test system for runtime testing of adaptable and distributed systems. In Nielsen, B., Weise, C., eds.: Testing Software and Systems, Berlin, Heidelberg, Springer Berlin Heidelberg (2012) 71–86
19. Lahami, M., Krichen, M., Alroobaea, R.: Tepaas: test execution platform as-a-service applied in the context of e-health. IJAACS **12**(3) (2019) 264–283
20. Lahami, M., Krichen, M., Barhoumi, H., Jmaiel, M.: Selective test generation approach for testing dynamic behavioral adaptations. In: Testing Software and Systems, Cham, Springer International Publishing (2015) 224–239
21. Lahami, M., Krichen, M., Bouchakwa, M., Jmaiel, M.: Using knapsack problem model to design a resource aware test architecture for adaptable and distributed systems. In: ICTSS. Volume 7641 of Lecture Notes in Computer Science., Springer (2012) 103–118
22. Lahami, M., Krichen, M., Jmaiel, M.: Safe and efficient runtime testing framework applied in dynamic and distributed systems. Science of Computer Programming **122** (2016) 1 – 28
23. Maâlej, A.J., Krichen, M.: A model based approach to combine load and functional tests for service oriented architectures. In: Proceedings of the 10th Workshop on Verification and Evaluation of Computer and Communication System, VECoS 2016, Tunis, Tunisia, October 6-7, 2016. (2016) 123–140
24. Maâlej, A.J., Krichen, M., Jmaiel, M.: Model-based conformance testing of WS-BPEL compositions. In: 36th Annual IEEE Computer Software and Applications Conference Workshops, COMPSAC 2012, Izmir, Turkey, July 16-20, 2012. (2012) 452–457

25. Maâlej, A.J., Lahami, M., Krichen, M., Jmaïel, M.: Distributed and resource-aware load testing of WS-BPEL compositions. In: ICEIS (2), SciTePress (2018) 29–38
26. Mahmud, R., Ramamohanarao, K., Buyya, R.: Latency-aware application module management for fog computing environments. ACM Trans. Internet Technol. **19**(1) (November 2018) 9:1–9:21
27. Ottenwälder, B., Koldehofe, B., Rothermel, K., Ramachandran, U.: Migcep: Operator migration for mobility driven distributed complex event processing. In: Proceedings of the 7th ACM International Conference on Distributed Event-based Systems. DEBS '13, New York, NY, USA, ACM (2013) 183–194
28. Rahbari, D., Nickray, M.: Scheduling of fog networks with optimized knapsack by symbiotic organisms search. In: 2017 21st Conference of Open Innovations Association (FRUCT). (Nov 2017) 278–283
29. Skarlat, O., Nardelli, M., Schulte, S., Borkowski, M., Leitner, P.: Optimized iot service placement in the fog. Service Oriented Computing and Applications **11**(4) (Dec 2017) 427–443
30. Souza, V.B., Masip-Bruin, X., Marin-Tordera, E., Ramirez, W., Sanchez, S.: Towards distributed service allocation in fog-to-cloud (f2c) scenarios. In: 2016 IEEE Global Communications Conference (GLOBECOM). (Dec 2016) 1–6
31. Souza, V.B.C., Ramírez, W., Masip-Bruin, X., Marín-Tordera, E., Ren, G., Tashakor, G.: Handling service allocation in combined fog-cloud scenarios. In: 2016 IEEE International Conference on Communications (ICC). (May 2016) 1–5
32. Taneja, M., Davy, A.: Resource aware placement of iot application modules in fog-cloud computing paradigm. In: 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM). (May 2017) 1222–1228
33. Tang, Z., Zhou, X., Zhang, F., Jia, W., Zhao, W.: Migration modeling and learning algorithms for containers in fog computing. IEEE Transactions on Services Computing (2018) 1–1
34. Venticinque, S., Amato, A.: A methodology for deployment of iot application in fog. Journal of Ambient Intelligence and Humanized Computing **10**(5) (May 2019) 1955–1976
35. Wang, S., Zafer, M., Leung, K.K.: Online placement of multi-component applications in edge computing environments. IEEE Access **5** (2017) 2514–2533
36. Wen, Z., Yang, R., Garraghan, P., Lin, T., Xu, J., Rovatsos, M.: Fog orchestration for internet of things services. IEEE Internet Computing **21**(2) (Mar 2017) 16–24
37. Xia, Y., Etchevers, X., Letondeur, L., Coupaye, T., Desprez, F.: Combining hardware nodes and software components ordering-based heuristics for optimizing the placement of distributed iot applications in the fog. In: Proceedings of the 33rd Annual ACM Symposium on Applied Computing. SAC '18, New York, NY, USA, ACM (2018) 751–760
38. Yang, L., Cao, J., Liang, G., Han, X.: Cost aware service placement and load dispatching in mobile cloud systems. IEEE Transactions on Computers **65**(5) (May 2016) 1440–1452
39. Zeng, D., Gu, L., Guo, S., Cheng, Z., Yu, S.: Joint optimization of task scheduling and image placement in fog computing supported software-defined embedded system. IEEE Transactions on Computers **65**(12) (Dec 2016) 3702–3712
40. Zhang, H., Xiao, Y., Bu, S., Niyato, D., Yu, F.R., Han, Z.: Computing resource allocation in three-tier iot fog networks: A joint optimization approach combining stackelberg game and matching. IEEE Internet of Things Journal **4**(5) (Oct 2017) 1204–1215