A Profitable and Energy-Efficient Cooperative Fog Solution for IoT Services

Ismaeel Al Ridhawi[1], Moayad Aloqaily[2], Yehia Kotb[3], Yaser Jaraweh[4], and Thar Baker[5]
[1]University of Ottawa, School of Electrical Engineering and Computer Science, Ottawa, Ontario, Canada, K1N6N5, Ismaeel.AlRidhawi@uottawa.ca
[2]Gnowit Inc., 7 Bayview Road, Ottawa, ON, Canada, K1Y2C5, Moayad@gnowit.com
[3]Department of Computer Science, University of Western Ontario, London, Ontario, Canada, YKotb@alumni.uwo.ca
[4]Jordan University of Science and Technology, Irbid, Jordan, YiJararweh@just.edu.jo
[5]Liverpool John Moores University, United Kingdom, T.Baker@ljmu.ac.uk

***Abstract*—Fog-to-Fog (F2F) communication has been introduced to deliver services to clients with minimal reliance on the cloud through resource and capability sharing of cooperative fogs. Current solutions assume full cooperation among the fogs to deliver simple and composite services. Realistically, each fog might belong to a different network operator or service provider and thus will not participate in any form of collaboration unless self-monetary profit is incurred. In this paper, we introduce a fog collaboration approach for simple and complex multimedia service delivery to cloud subscribers while achieving shared profit gains for the cooperating fogs. The proposed work dynamically creates short-term service-level-agreements (SLA) offered to cloud subscribers for service delivery while maximizing user satisfaction and fog profit gains. The solution provides a learning mechanism that relies on online and offline simulation results to build guaranteed workflows for new service requests. The configuration parameters of the short-term SLAs are obtained using a modified tabu-based search mechanism that uses previous solutions when selecting new optimal choices. Performance evaluation results demonstrate significant gains in terms of service delivery success rate, service quality, reduced power consumption for fog and cloud datacenters, and increased fog profits.**

Keywords —IoT, Cloud, Fog, Workflow, Tabu-Search, SLA.

I. Introduction

Cloud computing continues to evolve as a domain by providing enhanced services for subscribers through the traditional cloud layer and resource-rich edge devices. Resource-constraint mobile devices requesting cloud services for time-sensitive applications make use of the fog computing paradigm for reliable service delivery [1]. To coordinate and manage the usage of cloud and fog resources, a fog-to-cloud (F2C) architecture has been proposed in [2], which considers the identification of existing clouds and fogs in the environment. The solution also considers a taxonomy for facilitating the mapping between the requested and available resources, a resource discovery and allocation technique, and a service execution scheduling mechanism. Although the proposed work did not provide an effective solution for the considered issues, the work provided an initiative to the research community to enhance and provide a framework for the F2C paradigm. The F2C architecture was further extended and enhanced in [3] to provide a fog-to-fog (F2F) solution. In such a scenario, fogs are not limited to task execution at the fog or cloud, but also can communicate with other fogs to process job requests. This, in essence, provides reduced end-to-end latency and helps the F2C paradigm reach its maximum potential. The solution is limited to processing tasks at a single fog, such that if one of the fogs is unable to process the service request, an alternative fog is chosen to complete the task. In case none of the fogs are able to complete the task, then the traditional F2C solution is adopted.

In this paper we introduce a cooperative fog solution built on the F2F and F2C frameworks for the purpose of delivering simple and complex cloud multimedia services to achieve increased profits for fog service providers, enhanced service quality for customers, fog load balancing, and ultimately, reduced overall energy usage on both cloud and fog datacenters. Since user-defined services such as composite media files involves a one-time requester/provider interaction, short-term service level agreements (SLA) are negotiated between the service requester and provider. Such short-term SLAs are dynamically configured according to cloud and fog resource availability to meet service demands for cloud subscribers. The proposed SLA configuration process relies on a real-time simulator to continuously find optimal or near-optimal configurations through a modified tabu-based search mechanism.

Real-time simulations which run as a background process on the cloud test different scenarios which encompass distinctive network policy configurations. Simulation results are used to determine the optimal configurations that would lead to satisfactory network behavior in terms of user service quality satisfaction and service provider profit enhancements. Moreover, simulation tests determine the optimal workflow choice resembling the service composition process for customer requests of complex cloud multimedia services. The workflow selection process is dependent on a reinforced learning approach which takes into consideration previous successful workflow adaptations. Real-time simulators are used to test the selected workflow choice to determine its impact in terms of the network behavior.

The rest of the paper is organized as follows: Section II outlines some related work in the literature. Section III discusses the proposed architecture and provides a solution overview. Section IV focuses on the problem model and the optimization problem. Section V discusses the solution adopted to solve the optimization problem. Section VI focuses on the service composition workflow selection process. Simulation results are discussed in Section VII. Finally, Section VIII concludes the paper.

II. Related Work

The concept of fog cooperation is considered a recent contemporary issue in the telecommunication and engineering fields, where research in this area is limited. Naranjo et. al [4] introduced a fog computing network architecture that defines different types of communications between smart city devices to allow for computation offloading to the fog. Moreover, device collaboration is achieved through a peer-to-peer type network. The solution aims at enabling low energy usage in smart cities.

The authors in [5] considered the problem of offloading cloud datacenter workload to a set of fog nodes. The work investigates two performance metrics, namely, user QoE and fog node power efficiency. A cooperation strategy between fog nodes is introduced, in which fog nodes can forward part of its unprocessed workload to neighboring fog nodes rather than the cloud datacenter to improve user perceived QoE. The proposed solution uses a distributed optimization problem that is based on the alternating direction method of multipliers (ADMM) to achieve optimality in terms of user QoE maximization subject to the given power efficiency. The optimization problem is divided into a number of sub-problems, one for each fog node, solved using the available information on the fog. The results of each optimization sub-problem are forwarded to the cloud for the final decision on fog workload distribution. Numerical results show that the cooperative approach outperforms the traditional F2C solution in terms of user QoE improvement.

The authors in [6] proposed a cooperative fog architecture for the Internet of Vehicles (IoV) which considers intra- and inter-fog communication. The solution incorporates a centralized regional coordinator server at the fog layer to manage and assign tasks to different fog servers. Local fog servers dynamically update their performance and status of the assigned tasks with the coordinator server. This will allow the coordinator to continuously schedule the dataflow and adjust operating conditions of the fog servers to guarantee the QoS for users or optimize the performance of fog servers through load balancing. The article outlines different scenarios for fog cooperation. For instance, two fog servers can cooperate to handover service requests from different vehicles, such that if the vehicle is leaving the domain of one of the fogs, another fog is selected to resume the service for the vehicle once it arrives at the domain of the other fog. Another scenario involves the fog coordinator server requesting from different local fogs, traffic updates, such that each fog communicates back information to the coordinator according to its service capabilities (e.g. video surveillance service, GPS tracking, etc.). The authors also focus on different scenarios that involve load balancing and energy efficiency. Simulation results show that the cooperative fog approach reduces energy consumption and the packet dropping rate when compared to the traditional non-cooperative fog approach.

Huang et al. [7] considered a study on the competition involved between cloud and network service providers. The authors present a new economic model characterizing the competition involved, in addition to findings from conducted theoretical analyses and numerical experiments. The work leverages the game theory approach to model competitions among different cloud service providers. Results of the conducted tests show that service providers with a small replacement coefficient have a competitive edge and high profit. Additionally, higher connectivity rates from customers would require providers to improve service quality by increasing costs, and thus leading to reduced profit. Moreover, larger service coefficients would lead to service provider profit that depends more on its service rate. Finally, the Nash equilibrium point increases as the service coefficient, connectivity rate, or the number of service requests decrease.

The work proposed in this article is distinguished from other proposals in the literature, where we mainly focus on identifying optimal solutions for fog service providers' profit maximization subject to other conditions, using a meta-heuristic search algorithm. The search technique identifies the network configurations required to achieve such maximized provider profits while guaranteeing user's QoS requirements.

III. Proposed Architecture

We consider a cloud environment comprised of multiple fogs acting as service providers (SPs), where each client may be served by one or more fogs (i.e. SPs) at a given time as shown in Figure 2.

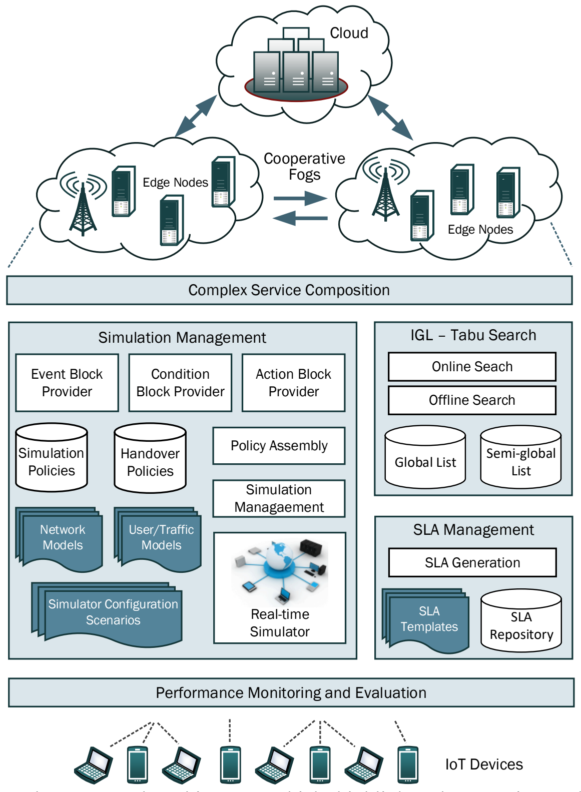


Fig.1. The proposed architecture which highlights three main modules: Simulation Management, IGL-TS search module, and SLA Management.

A. *Service Request*

A mobile client (u_j) can request a service (either simple or complex) with service quality offerings described by the class of service (CoS) from the service provider (SP_i). A CoS quality offering is described by the vector $\boldsymbol{q_{ij}} = \left(\underline{\boldsymbol{q}}_{\boldsymbol{ij}}, \overline{\boldsymbol{q}}_{\boldsymbol{ij}}\right)$, where $\underline{\boldsymbol{q}}_{\boldsymbol{ij}} = \left(\underline{q}_{ij}^{1}, \underline{q}_{ij}^{2}, \dots, \underline{q}_{ij}^{|K|}\right)$, and $\overline{\boldsymbol{q}}_{\boldsymbol{ij}} = \left(\overline{q}_{ij}^{1}, \overline{q}_{ij}^{2}, \dots, \overline{q}_{ij}^{|K|}\right)$ are the upper and lower bound range of the offered service from the service feature set K. The values $\underline{q}_{ij}^{k}$ and $\overline{q}_{ij}^{k}$ are the upper and lower bounds of service feature $k \in K$. Those upper and lower bound QoS values are prone to change frequently according to network performance. For instance, if network resources are limited, then the offered QoS value range for a particular CoS would be reduced to accommodate for the available resources (both at the serving fog and other cooperating fogs). This introduces the concept of short-term SLAs, where users would receive different QoS configurations belonging to the same CoS due to the availability of network resources. In addition to the service quality offerings integrated in the short-term SLA between client u_j and a provider SP_i, a set of penalty values $\boldsymbol{p_{ij}} = (p_i^R, p_i^L, p_i^E, p_i^T)$ are accumulated on the SP if the agreed upon QoS guarantees are not met. p_i^R is a penalty for service rate downgrade, p_i^L is a penalty for service disruption, p_i^E is a penalty for error rate violations, and p_i^T is a penalty for forced fog handovers that result in service quality degradation [8].

B. *Challenges*

A challenge that arises would be how to continuously provide optimal QoS configurations to be offered to clients through short-term SLAs within a limited time manner. Those configurations should both guarantee the QoS levels requested by the clients through fog cooperation and sustain monetary profits for cooperative fogs. To achieve such guarantees, we model the problem as a set of policies. The events, conditions, and actions are denoted in policy form to enable the automation of resource management. Such an approach allows for the separation of policies from the underlying managed system needed to attain system flexibility and scalability. For instance, consider a service request for a multimedia file with certain quality preferences is received at one of the serving fogs. Assuming that the serving fog is incapable of fulfilling the request due to limitation in resources and high energy consumption, new policy configurations (i.e. SLA offered QoS configurations) are considered to achieve the service request. Those configurations are attained by having a set of fogs collaborate in order to accomplish the requested multimedia service. Such collaboration might result in either client handoff from one fog to another, or sharing of resources between fogs.

C. *Solution Overview*

A real-time simulator incorporated within the *Simulation Management Module* of the architecture is used to determine the optimal scenario (e.g. handover user from fog 1 to fog 2) that would result in better network behavior $B(\boldsymbol{x}(t))$, where $\boldsymbol{x}(t) = (x_1, x_2, \dots, x_M)$ is a vector of policy configurations. Each parameter x_m, $m \in 1,2,\dots,M$, represents a variable in a policy event, condition or action. A domain $D(x_m)$ is associated with the parameters which contains all permissible values that x_m can take. The objective is to determine the set of optimal policy configurations $\boldsymbol{x} \in X$ such that when applied, the result is a set of observed QoS readings $\widetilde{\boldsymbol{q}}_{\boldsymbol{ij}}(t)$ which meets the user's request (i.e. short-term SLA). $\widetilde{\boldsymbol{q}}_{\boldsymbol{ij}}(t) = \left(\tilde{q}_{ij}^{1}(t), \tilde{q}_{ij}^{2}(t), \dots, \tilde{q}_{ij}^{|K|}(t)\right)$, where $\tilde{q}_{ij}^{k}(t)$ is the observed QoS reading for service feature $k \in K$ at time point t. The result, which is then depicted into the SLA, is beneficial for both service requesters and providers. Requestors are able to attain the requested service in limited resource environments. On the contrary, providers are able to attain profit through cooperation with limited resource and energy usage.

We incorporate a novel heuristic search technique called the *Iterated Global and Local - Tabu Search* (IGL-TS) that enables both online and offline search for policy configurations. Results from previous simulation runs are used by the *IGL–Tabu Search Module* to adapt the tested policy configurations online for faster service delivery in environments with similar network conditions. A more optimal solution is considered using run-time simulators for service requests with less-stringent time-constraints. Based on the simulation results, the *IGL–Tabu Search Module* submits the obtained configurations to the *SLA management module* to generate new short-term SLAs.

Moreover, user requests which entail *composite* services require another set of fog cooperation and scenario testing to ensure that composite services are delivered to users according to their quality preferences. For instance, consider a service request for a multimedia file with specific enhancements is received at one of the serving fogs. Assuming that the serving fog is incapable of fulfilling the request due to the unavailability of multimedia enhancement capabilities, fogs must collaborate and perform task planning in order to accomplish the requested composite service. The *Complex Service Composition Module* creates a set of composition workflows that resemble the resource components and fog capabilities needed from the cooperating fogs to compose and deliver the composite service. Workflows are an extension to petri-nets, and are comprised of places, transitions, tokens, and arcs. Details in regards to petri-net and workflow modeling are out of the scope of this article and can be found in [9]. The workflow composition process is dependent on a reinforcement learning approach that takes into consideration previous successful workflow adaptations. Workflows are then tested using a real-time simulator to determine the optimal scenario (i.e. workflow thread) that would result in a better network behavior $B(\boldsymbol{x}(t))$.

IV. Profit Model and Optimization Problem

We define the profit function $\mathcal{P}\left(\boldsymbol{q_{ij}}, \widetilde{\boldsymbol{q}}_{\boldsymbol{ij}}(\boldsymbol{t})\right)$ as the profit gained by adapting new policy configurations through a scenario S, having observed the QoS readings $\widetilde{\boldsymbol{q}}_{\boldsymbol{ij}}(t)$ for each service feature $k \in K$. $\widetilde{\boldsymbol{q}}_{\boldsymbol{ij}}(t)$ considers the experienced service rate $\tilde{r}_{ij}(t)$, error rate $\tilde{e}_{ij}(t)$, and used capacity $\tilde{c}_{ij}(t)$ by u_j at time t. Additionally, $\widetilde{\boldsymbol{q}}_{\boldsymbol{ij}}(t)$ considers the experienced service disruption $\tilde{l}_{ij}(t)$. Profit is defined as the net revenue gained from each client u_j minus the costs for the service provider SP_i [8]. Such calculations in terms of revenue and cost are set according to the agreed upon SLA.

Although each fog SP_i can control the offered service quality $\boldsymbol{q_{ij}}$, it cannot predict the observed quality of the service readings $\boldsymbol{\tilde{q}_{ij}}(t)$ before adaptation. QoS depends on different factors such as the number of users serviced by the fog and the tasks assigned by each user. The optimal configurations needed to provide the optimal performance $\boldsymbol{\tilde{q}_{ij}}(t)$ are adapted using the proposed IGL-TS method. The new configurations also aim at maximizing the SP monetary profit $\mathcal{P}(\boldsymbol{q_{ij}}, \boldsymbol{\tilde{q}_{ij}}(t))$. Hence, the objective function for the cloud considered here is set according to (1), subject to user satisfaction (i.e. according to obligations provided by the contracted SLA).

$$\boldsymbol{P}: \left(\text{maximize} \sum_{i=1}^{N} \sum_{j=1}^{U(t)} \mathcal{P}(\boldsymbol{q_{ij}}, \boldsymbol{\tilde{q}_{ij}}(t)) \right) \tag{1}$$

Additionally, the solution to the optimization problem must satisfy fog resource constraints:

$$\tilde{r}_{ij}(t) \leq \boldsymbol{R}_{\mathrm{i}} \qquad for\ i = 1, \dots, N, j = 1, \dots, U(t). \tag{2}$$

where the sum of the service rate allocated to all the users serviced by a fog must not exceed the fog's total resources $\boldsymbol{R}_{\mathrm{i}}$.

The search for new configurations $\boldsymbol{\tilde{q}_{ij}}(t)$ leading to a better network behavior is performed through a scenario search problem. Scenarios target testing one or more new configurations through a simulator. The process of identifying and searching for those configurations are discussed in the following section.

V. Scenario Search Problem

A. *Simulation Scenario*

To manage fog resources optimally, it is highly important to test the network decision before the adaptation of the new configurations. A set of simulation scenarios that incorporate a set of network configurations are tested using a run-time simulator to determine whether user-requested service qualities can be met while increasing service provider profit. Specific details regarding the adopted simulator are out of the scope of this article and have been provided in an earlier work [10]. The simulation process is used to provide a measurement for the effect of mapping new policy configurations $\boldsymbol{x}$ to the cloud network environment. Information in regards to the underlying physical network, users and their generated traffic is represented by the models M^{NW} and M^{Users}. Scenarios are maintained with the aid of the simulation management module to be used by the simulator. A scenario, S, is characterized as a tuple $(M^{NW}, M^{Users}, \boldsymbol{x})$. The system targets testing one or more scenarios with new configurations obtained using the IGL-TS method. Such tests include tasks such as: fog SP resource reallocation, user handoff scheme reconfiguration, and composite service composition workflows depending on the type of the performance degradation.

Simulation results indicate whether to perform further simulations, adapt the configurations to the network, or communicate the new configurations to the SLA manager for new service requests. Figure 2 provides a visual representation of the scenario mapping process through the simulator. The figure is divided into three domains, namely: Network/User models, policy configurations, and network behavior. The simulator begins by testing the effects of adapting the initial scenario S_0 (i.e. a set of policy configurations adapted on the current network and user models) and then observing the network behavior $\boldsymbol{\tilde{q}_{ij}}(t)$. When adapting another scenario S, which adheres to new policy configurations found using the proposed search algorithm, the network performance is tested and compared against the initial scenario. Positive results (i.e. higher network behavior score $B(\boldsymbol{x}(t))$) entail adaptation of the new configurations.

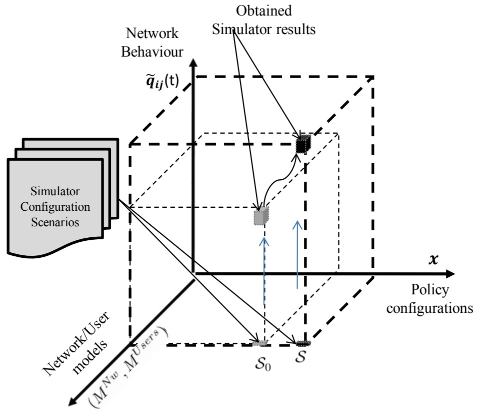


Fig.2. Graphical representation of the mapping process.

B. *Iterated Global and Local Tabu Search*

The IGL-TS method is used to automate the process of finding new policy configurations. To determine whether the new configurations result in a better network behavior, a behavior function (3) is incorporated to provide a measure for the observed QoS performance $\boldsymbol{\tilde{q}_{ij}}(t)$. The behavior function $B(\boldsymbol{x}(t))$ is associated to a set of policies identified by the policy vector $\boldsymbol{x}$ at time t.

$$B(\boldsymbol{x}(t)) = \sum_{i=1}^{N} \sum_{j=1}^{U(t)} \sum_{k=1}^{|K|} \frac{score\left(\tilde{q}_{ij}^{k}(t)\right)}{N \times U(t) \times |K|} \tag{3}$$

where $score\left(\tilde{q}_{ij}^{k}(t)\right)$ is a value ranging from 0 to 1, used to indicate the effect of adapting the vector $\boldsymbol{x}$ in terms of QoS aspect K.

Two sets of lists are used to store the best previous configurations found most recently using IGL-TS. The *Global Candidate List* $X_G = (\boldsymbol{x}_G^1, \boldsymbol{x}_G^2, \dots, |\boldsymbol{x}_G|)$ and the semi-*Global Candidate List* $X_{sG} = (\boldsymbol{x}_{sG}^1, \boldsymbol{x}_{sG}^2, \dots, |\boldsymbol{x}_{sG}|)$ are used to ensure network configurations are updated for current clients without continuous reliance on simulation results. The rank for each solution $\boldsymbol{x}_G^z \in X_G$ is determined according to (4). A solution is prone to being moved up or down, where the one with the greatest number of acceptances is stored at the top of the list.

$$Rank(\boldsymbol{x}_G^z) = \frac{\#\ of\ times\ \boldsymbol{x}_G^z\ is\ chosen}{\#\ of\ times\ \boldsymbol{x}_G^z\ is\ visited}, \quad \forall \boldsymbol{x}_G^z \subset X_G \tag{4}$$

The IGL-TS method first evaluates the current policy configurations $\boldsymbol{x_0}$ for QoS performance using the behavior function $B(\boldsymbol{x}(\boldsymbol{t}))$. Then, a fast, online search procedure is used to find better network performance. An offline and more in-depth search technique is then used to find configurations leading to optimal performance. Solutions from both the online and offline search are available for other fogs to be used in similar network scenarios.

The online search is first performed on X_G, where each solution in the list is tested and evaluated by the simulator using (3). Any solution found that outperforms the behavior of $\boldsymbol{x_0}$ is adapted. In case no solution is found, a list of neighbors for $\boldsymbol{x_0}$ is generated according to Algorithm 1.

Algorithm 1: Find $\boldsymbol{x}' \in \boldsymbol{N}(\boldsymbol{x}(\boldsymbol{t}))$ that results in better $\tilde{\boldsymbol{q}}_{ij}(\boldsymbol{t})$

```
construct $N(\boldsymbol{x}(t))$ with $\theta$ and $\beta$
Repeat
  remove $\boldsymbol{x}'$ from $N(\boldsymbol{x}(t))$
  simulate $\boldsymbol{x}'$ and compute $B(\boldsymbol{x}')$
until $(B(\boldsymbol{x}') > B(\boldsymbol{x}(t))$ or $N(\boldsymbol{x}(t))$ is empty$)$
```

The size of the neighborhood $N(\boldsymbol{x})$ for a vector $\boldsymbol{x}$ is determined according to (5). It is controlled by two parameters, θ and β, where θ is the size of the area considered for search, and β is the distance between the two nearest configurations in the search space.

$$N(\boldsymbol{x}) = \{\boldsymbol{x} \in X: x_{i-1}^k \le x_i^k \le \theta x_{i-1}^k, k = 1, \dots, |k|\} \quad (5)$$

$$\theta = \frac{B(\boldsymbol{x_{i-1}}(t)) - B(\boldsymbol{x_{i-1}}(t + \Delta t))}{B(\boldsymbol{x_{i-1}}(t))} \quad (6)$$

A more exhaustive search using the space $X_G \cup X_{sG} \cup N(\boldsymbol{x}(t))$ is then performed using the general tabu-search procedure [11]. An element from the $X_{candidate}$ list (i.e. X_G and X_{SG}) is exchanged with that of an element of the new search space according to Algorithm 2.

Algorithm 2: Find x' that results in optimal $\tilde{q}_{ij}(t)$ and update X_G and X_{sG}

```
construct $N(\boldsymbol{x}(t))$ with a smaller step size $\beta$
$\forall\, \boldsymbol{x}' \in X_G \cup X_{sG} \cup N(\boldsymbol{x}(t))$
  $tabu(\boldsymbol{x}") = 0$;
  $X_{candidate} = \emptyset$;
  Repeat
    select $\boldsymbol{x}' \in X_G \cup X_{sG} \cup N(\boldsymbol{x}(t))$ when $\boldsymbol{x}'$ is not tabu
    $tabu(\boldsymbol{x}') = tabu(\boldsymbol{x}') + 1$;
    if $(TS_{tenure} - tabu(\boldsymbol{x}') \ge 0)$ then
      simulate and calculate $B(\boldsymbol{x}')$
          if $size(X_{candidate} \le MaxSize)$
            add $\boldsymbol{x}'$ to $X_{candidate}$
          else $(\exists\, \boldsymbol{x} \in X_{candidate}$ s.t. $B(\boldsymbol{x}) < B(\boldsymbol{x}'))$ then
            replace $\boldsymbol{x}$ with $\boldsymbol{x}'$ in $X_{candidate}$
            calculate $Rank(\boldsymbol{x}')$;
            if $Rank(\boldsymbol{x}') > Rank(|\boldsymbol{x}_G|)$ then
              $X_{sG} = X_{sG} \cup |x_G| - |x_{sG}|$;
              $X_G = X_G \cup \boldsymbol{x}' - |x_G|$;
  else
    $X_{candidate} = X_{candidate} - \boldsymbol{x}'$;
    mark $\boldsymbol{x}'$ as tabu
  until duration $= T_{IGL}$
```

VI. Complex Service Requests

Simple and complex services are both a result of service composition. However, when service elements (referred to in this article as '*service units*') are available within a fog node, cooperation is not mandatory to provide the composed service. Such a composition process is referred to as simple service composition. On the contrary, for complex services, user requests require the composition of multiple service elements belonging to different fog nodes. Such a composite service can only be fulfilled through the cooperation of those nodes. For instance, if a user is requesting a multimedia service that requires the addition of visual and audio enhancements, then, we assume that the service units in this case are: the original video stream, the visual enhancement (assuming a single visual effect is added), and the audio enhancement (assuming a single audio effect is added). Each service unit belongs to a different fog node. Moreover, multiple cooperating fog nodes may have the capability of providing the same service unit. Therefore, a selection mechanism is required for service units leading to an optimal network behavior in terms of SP profit and QoS enhancements.

The selection process begins with a request for a composite service arriving at the serving fog node. To determine the set of service units needed to compose the requested service, a decomposition module is used to decompose the service into a set of sub-services (i.e. service elements). This decomposition process is repeated until a sub-service is eventually characterized as a service unit, in which, the decomposed units are non-decomposable, and if decomposed any less, it will not add value to the decomposition. Details in regards to the decomposition process is out of the scope of this article and can be found in [12]. Those units are then checked against capability and cost matrices outlining each fog node's service capabilities in terms of service units with a fuzzified aggregate score for each applicable service unit and the cost of performing that service. Hence, a fuzzification module is integrated within the system. Workflow patterns representing the cooperative set of fog nodes needed to compose the requested service, in addition to each node's fuzzified scores are created. Those workflows are then applied to the re-enforced learning module which uses results from previous similar solutions to determine the best set of workflows which are applicable to the current network configuration. Moreover, an offline technique is then adopted, similar to the method used previously in Section V, where the set of workflows are tested using a real-time simulator to determine the accuracy of the re-enforced learning algorithm. Those offline results are used for future requests and are used to enhance the learning process. Figure 3 outlines the modules involved in the composition process.

A. QoS and Cost Fuzzification Process

A service s belongs to the set of all services S defined in the cloud environment, such that $s \in S \mid S = \{s_1, s_2, \dots, s_m\}$, where m is the number of all defined services. A service s is composed of service units μ_n, such that $\forall\, s \in S,\ s = \prod_{n=1}^{N} \mu_n$, where $N = |S|$ and $\mu_n \in U$. A complex service request R is decomposed into service units as described earlier, and is only maintainable through the cooperation C of

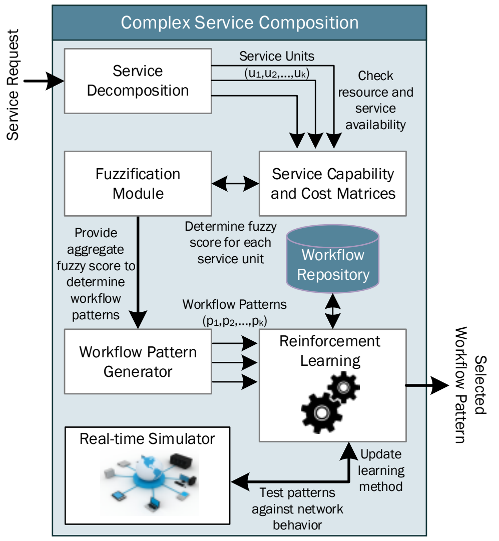

Complex Service Composition
Service Request
Service Decomposition
Service Units (u1,u2,...,uk)
Check resource and service availability
Fuzzification Module
Service Capability and Cost Matrices
Provide aggregate fuzzy score to determine workflow patterns
Determine fuzzy score for each service unit
Workflow Repository
Workflow Patterns (p1,p2,...,pk)
Workflow Pattern Generator
Reinforcement Learning
Selected Workflow Pattern
Real-time Simulator
Update learning method
Test patterns against network behavior

Fig.3. Complex service composition module integrated within the cloud datacenter used to compose and select optimal workflow patterns.

different fog nodes SP. Hence, $\exists R, \forall SP_i \vee SP_{i+1} \mid \exists\, C[SP_i, SP_{i+1}] \leftrightarrow \forall \mu_n \in R, \mu_n \in SP_i \vee SP_{i+1}$. Cooperative fogs are selected using the capability and cost matrices. A decomposed service request is first checked against the service capability matrix, which outlines the cooperating fogs' service capabilities (i.e. μ_n) and their fuzzified aggregate QoS scores $z_{QoS}(SP_i(\mu_n))$, where z_{QoS} is the QoS fuzzification function associated with service unit μ_n and provided by fog SP_i. Below is an example of the capability matrix:

$$
s_{Capability} = \begin{array}{c} \\ SP_1 \\ SP_2 \\ SP_3 \\ SP_4 \\ SP_5 \\ SP_6 \end{array}
\begin{array}{c} \begin{array}{ccc} \mu_1 & \mu_2 & \mu_3 \end{array} \\
\begin{bmatrix}
z_{QoS}(SP_1(\mu_1)) & z_{QoS}(SP_1(\mu_2)) & z_{QoS}(SP_1(\mu_3)) \\
z_{QoS}(SP_2(\mu_1)) & z_{QoS}(SP_2(\mu_2)) & z_{QoS}(SP_2(\mu_3)) \\
z_{QoS}(SP_3(\mu_1)) & z_{QoS}(SP_3(\mu_2)) & z_{QoS}(SP_3(\mu_3)) \\
z_{QoS}(SP_4(\mu_1)) & z_{QoS}(SP_4(\mu_2)) & z_{QoS}(SP_4(\mu_3)) \\
z_{QoS}(SP_5(\mu_1)) & z_{QoS}(SP_5(\mu_2)) & z_{QoS}(SP_5(\mu_3)) \\
z_{QoS}(SP_6(\mu_1)) & z_{QoS}(SP_6(\mu_2)) & z_{QoS}(SP_6(\mu_3))
\end{bmatrix} \end{array}
$$

where the fuzzified score values $z_{QoS}(SP_i(\mu_n)) = [0,1,2,3,4]$ represent the fuzzified regions [Not Acceptable (NA), Somewhat Acceptable (SA), Acceptable (A), Very Acceptable (VA), and Extremely Acceptable (EA)], respectively.

Similarly, the request is then checked against the service cost matrix, which outlines the fuzzified costs of performing a service unit $z_{Cost}(SP_i(\mu_n))$. Therefore, s is defined as the combination of two fuzzified service unit workflow pattern tuples, namely, $z_{QoS}(SP_i(U))$ and $z_{Cost}(SP_i(U))$, such that $s = z_{QoS}(SP_i(U)) \wedge z_{cost}(SP_i(U))$, where $z_{QoS}\big(SP_i(U)\big) = [z_{QoS}\big(SP_i(\mu_1)\big),\ z_{QoS}\big(SP_i(\mu_2)\big),\ \dots,\ z_{QoS}\big(SP_i(\mu_n)\big)]$ and $z_{Cost}\big(SP_i(U)\big) = [z_{Cost}\big(SP_i(\mu_1)\big),\ z_{Cost}\big(SP_i(\mu_2)\big),\ \dots,\ z_{Cost}\big(SP_i(\mu_n)\big)]$.

The fuzzification function presented in this article uses the generalized bell-shaped membership function $f(x; a, b, c)$ defined in [13], suitable for both QoS properties and cost. A set of fuzzified workflow pattern tuples, s, are formed, which identify the set of collaborative fogs needed to compose the service. Clearly, multiple tuples might be discovered, which are then pushed to the reinforcement learning module to identify the most suitable pattern according to the current network configuration.

B. *Workflow Pattern Selection Using Reinforcement Learning*

A gain matrix $\mathbb{M}_s$ is derived for the service request R as shown in the below example, in which each element in the matrix indicates the current learned gain for executing service unit μ_n by fog service provider SP_i, where $\mathbb{M}_s\big(SP_i(\mu_n)\big) = z_{QoS}\big(SP_i(\mu_n)\big) + z_{Cost}\big(SP_i(\mu_n)\big)$.

$$
\mathbb{M}_s = \begin{array}{c} \\ SP_1 \\ SP_2 \\ SP_3 \\ SP_4 \\ SP_5 \\ SP_6 \end{array}
\begin{array}{c} \begin{array}{ccc} \mu_1 & \mu_2 & \mu_3 \end{array} \\
\begin{bmatrix}
\mathbb{M}_s(SP_1(\mu_1)) & \mathbb{M}_s(SP_1(\mu_2)) & \mathbb{M}_s(SP_1(\mu_3)) \\
\mathbb{M}_s(SP_2(\mu_1)) & \mathbb{M}_s(SP_2(\mu_2)) & \mathbb{M}_s(SP_2(\mu_3)) \\
\mathbb{M}_s(SP_3(\mu_1)) & \mathbb{M}_s(SP_3(\mu_2)) & \mathbb{M}_s(SP_3(\mu_3)) \\
\mathbb{M}_s(SP_4(\mu_1)) & \mathbb{M}_s(SP_4(\mu_2)) & \mathbb{M}_s(SP_4(\mu_3)) \\
\mathbb{M}_s(SP_5(\mu_1)) & \mathbb{M}_s(SP_5(\mu_2)) & \mathbb{M}_s(SP_5(\mu_3)) \\
\mathbb{M}_s(SP_6(\mu_1)) & \mathbb{M}_s(SP_6(\mu_2)) & \mathbb{M}_s(SP_6(\mu_3))
\end{bmatrix} \end{array}
$$

For fog SP_i to execute service unit μ_n, a history record of executions $\mathbb{H}_{SP_i(\mu_n)} = \{\mathbb{M}_s(0), \mathbb{M}_s(1), \mathbb{M}_s(2), \dots, \mathbb{M}_s(T)\}$ is considered, where T is the index of the last time point when service unit μ_n was used by SP_i. The mean square error is calculated as follows:

$$E = \int_{h=1}^{T} \frac{(\mathbb{M}_{eh} - \mathbb{M}_{ah})^2}{2}\,\partial h \tag{7}$$

where T is the history length, $\mathbb{M}_{eh}$ is the expected gain at history record h, and $\mathbb{M}_{ah}$ is the actual gain after the execution of history record h. Moreover, the average error is calculated as follows:

$$\bar{E} = \frac{E}{t} \tag{8}$$

The reward range according to the calculated average error would be $\mathbb{M}_s^{min}\big(SP_i(\mu_n)\big) \leq \mathbb{M}_s^{a}\big(SP_i(\mu_n)\big) \leq \mathbb{M}_s^{max}\big(SP_i(\mu_n)\big)$, where $\mathbb{M}_s^{a}\big(SP_i(\mu_n)\big)$ is the actual gain for performing service unit μ_n using fog SP_i, $\mathbb{M}_s^{min}\big(SP_i(\mu_n)\big) = \mathbb{M}_s\big(SP_i(\mu_n)\big) - \bar{E}$, and $\mathbb{M}_s^{max}\big(SP_i(\mu_n)\big) = \mathbb{M}_s\big(SP_i(\mu_n)\big) + \bar{E}$. Assuming that the gain is uniformly distributed between $\mathbb{M}_s^{min}\big(SP_i(\mu_n)\big)$ and $\mathbb{M}_s^{max}\big(SP_i(\mu_n)\big)$, the probability of getting the maximum gain is as follows:

$$P\left(\mathbb{M}_s^{max}\big(SP_i(\mu_n)\big)\right) = \frac{1}{\mathbb{M}_s^{max}\big(SP_i(\mu_n)\big) - \mathbb{M}_s^{min}\big(SP_i(\mu_n)\big)} \tag{9}$$

A reward value $\mathbb{R}_s\big(SP_i(\mu_n)\big)$ is derived for each gain and is calculated as follows:

$$\mathbb{R}_s\big(SP_i(\mu_n)\big) = P\left(\mathbb{M}_s^{max}\big(SP_i(\mu_n)\big)\right) \times \mathbb{M}_s^{max}\big(SP_i(\mu_n)\big) \tag{10}$$

The value function $V_{SP_i(\mu_n)}^{s(t)}$ for performing service unit μ_n by fog SP_i is then derived as follows:

$$V_{SP_i(\mu_n)}^{s(t)} = V_{SP_i(\mu_n)}^{s(t-1)} + \alpha\left(V_{SP_i(\mu_n)} - V_{SP_i(\mu_n)}^{s(t-1)}\right) \tag{11}$$

where $V_{SP_i(\mu_n)}^{s(t-1)}$ is the previously accumulated value function, $V_{SP_i(\mu_n)} = \mathbb{R}_s\big(SP_i(\mu_n)\big)$ and α is the learning rate with range $0 < \alpha < 1$. The value functions for each service unit and for all cooperating fogs are derived and are presented in a matrix pattern which we refer to as the value function matrix.

Workflow patterns resembling the chosen service units are selected to compose the service using the value function matrix that results in higher reward values. The selected workflow pattern is modeled as a plan of execution $\mathbb{P}$ as shown in equation (12).

$$\mathbb{P} = \bigcup V_{SP_i(\mu_n)} \| V_{SP_i(\mu_n)} = max\left(V_{SP_i(\mu_n)}\right), 1 \leq i \leq N \; for \; some \; n \quad (12)$$

The total value function for the plan $\mathbb{P}$ is calculated as shown in equation (13).

$$V^{s(t)} = \int_{n=1}^{m} max\left(V_{SP_i(\mu_n)}\right) dn, 1 \leq n \leq m \quad (13)$$

VII. Simulation Results

Simulations were conducted using the NS-3 simulator. The simulation topology followed the network scenario shown in Figure 4, which consists of mobile clients that can acquire services (i.e. data retrieval) by connecting to one of the SPs available. Thus, a client can connect to an LTE network through enhanced NodeB (eNB) base station BS_1 or through eNB BS_2, owned by two different service providers, namely, SP_2 and SP_3 which belong to BS_1 and BS_2, respectively. Moreover, clients are able to retrieve data from five WLANs managed by five different SPs, namely, SP_4, SP_5, SP_6, SP_7 and SP_8. Each SP resembles a fog which consists of an access point, storage site and a gateway, with the exception of SP_1, in which it resembles a cloud.

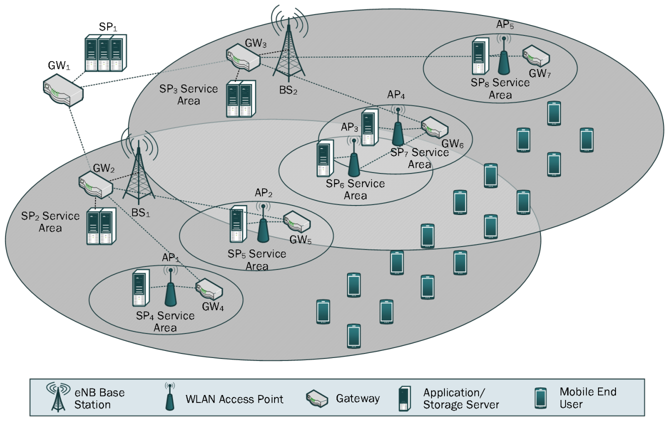


Fig.4. Adopted simulation scenario to closely resemble the considered problem.

The LTE eNB network coverage area is 7 km with a transmission power of 30 dBm. Each of the WLAN access points AP1-AP5 has a coverage area of 100 m, with a data rate of 54Mbps. Different simulation scenario configurations have been adapted with up to 100 mobile nodes. Nodes' movement was set at the speeds between 1-2 m/s according to the Gauss Markov mobility model with the aid of BonnMotion [14]. Each mobile node is equipped with both WLAN and LTE interfaces. The SPs receive and send packets from the mobile nodes using the base stations and access points. SPs communicate together through gateways. The IGL-TS algorithm was implemented using Java and the OpenTS library [15].

To simplify the problem, we decreased the number of policy configurations to three, such that each configuration is affected by three parameters, namely, the received signal strength, delay, and bandwidth offered to a mobile node. Moreover, for online search, the value of β is set to be bigger than 0.45, which will reduce the size of the list, hence, providing a faster solution. On the contrary, for an offline search, we set $\beta \leq 0.45$, which will result in a more in-depth performance. We considered four different tests, namely: SP profit, QoS adherence, energy efficiency, and service retrieval success rate. Simulation tests indicated that for the IGL-TS to achieve optimal solutions from an empty set, 17 iterations are needed with an accumulated time of 10 seconds. On the contrary, the non-modified TS (i.e. without use of global and semi-global lists), achieves optimality in 5800 seconds. Figure 5 depicts the normalized QoS performance for both the TS and the proposed IGL-TS techniques against the number of mobile clients. Results show that the average performance for all nodes using IGL-TS outperforms TS by 32% in terms of closeness to the optimal solution. IGL-TS shows a capability of achieving a closeness of 0.3%, while TS is only capable of achieving a closeness of 32.3% to the optimal solution.

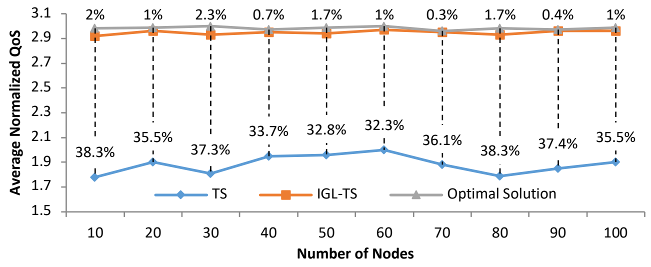


Fig.5. Normalized QoS performance, $B(\boldsymbol{x}(t))$, for TS and IGL-TS.

Service retrieval success rate as the number of nodes requesting services increase was considered in the tests. Cooperative and non-cooperative techniques were considered, namely, the proposed cooperative IGL-TS solution (CP-IGLTS), the cooperative non-modified TS solution (CP-TS), and a normal non-cooperative solution (NC), where SPs provide services to clients whenever resources are available without reliance on other SPs. Results depicted in Figure 6 indicate that as the number of nodes increase in the environment, the success rate gap between the cooperative and non-cooperative solutions increase significantly. For instance, with 100 mobile nodes in the environment, the service hit ratio gap between NC and CP-TS is 44%, while the gap between NC and CP-IGLTS is 50%. Moreover, when the proposed cooperative solution is compared against the non-modified TS

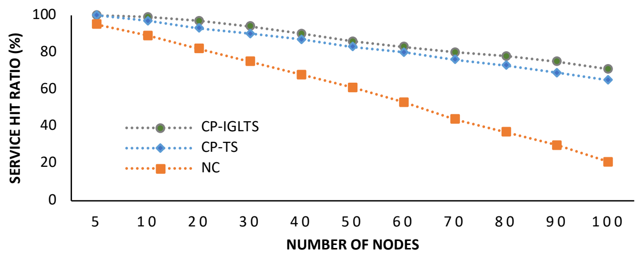


Fig.6. Service retrieval success rate for cooperative (IGLTS and TS) and non-cooperative solutions.

cooperative solution, we see that the gap in service hit ratio is 6% in favor of CP-IGLTS, hence, showing superiority.

Profit analysis was conducted on the proposed IGL-TS solution against the non-modified TS and non-cooperative solution. It is evident from Figure 7, that both cooperative solutions (CP-IGLTS and CP-TS) outperform the noncooperative solution in terms of accumulated profit, such that on average, for both the cooperative solutions, an increase of 56.8% in profit is seen over the non-cooperative solution. Moreover, the IGL-TS solution outperforms CP-TS by 9.3%. The inability to meet the requirements of mobile clients by not meeting the SLA guarantees is a major contributor to the decreased values in profit for the non-cooperative solution.

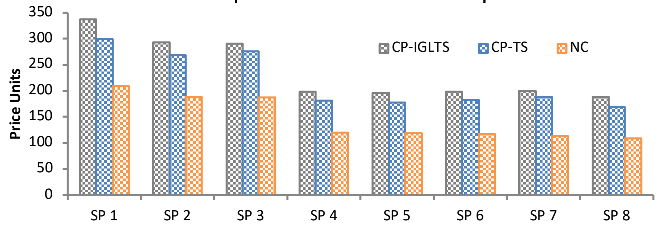


Fig.7. Accumulated profit for cooperative (IGLTS and TS) and non-cooperative solutions.

Simulation tests were also considered for the evaluation of energy consumption. It was assumed that one unit of energy is consumed at the service providers for each instance of service retrieval by the client. We assumed that a service is composed of one or more service units, conducted using the proposed complex service composition technique for the cooperative solutions. For the non-cooperative solution, we assumed that all services are available by a single service provider, but resources might not be available at the time of request, hence, increasing delay and the number of repeated requests, in essence leading to higher energy consumption. As depicted in Figure 8, when comparing the proposed cooperative IGL-TS approach against the non-cooperative approach, results show that the overall energy consumption is nearly reduced by threefold (i.e. a reduction of 66.4% in the overall energy consumption). When comparing the cooperative IGL-TS against the cooperative TS, results indicate that CalralrP-IGLTS outperforms CP-TS, with a reduction of 18.4% to the overall energy consumption. Other results have indicated that the overall energy consumption is balanced among the SPs. The energy consumption on SP1 (cloud) was reduced due to having tasks being offloaded to other SPs (fogs). At the same time, profit for SP1-SP8 have increased due to reduction in the number of occurrences in SLA violations, leading to less penalties being incurred by the SPs.

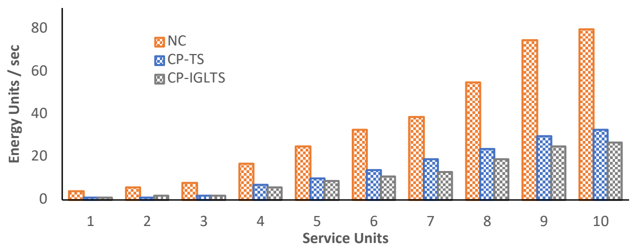


Fig.8. Overall accumulated power consumption at all service providers using cooperative (IGLTS and TS) and non-cooperative solutions.

VIII. Conclusion

This article proposed a solution to maximize fog service provider profit subject to maintaining user requested QoS according to agreed-upon SLAs. The solution uses a meta-heuristic search algorithm that identifies network configurations needed to achieve maximized provider profits. We show that complex services require fog node cooperation to achieve the requested tasks. A reinforcement learning algorithm is used to compose services through the use of workflows. Real-time simulators are used to test different network configurations and workflows to determine the optimal configurations needed to achieve maximized profits, reduced energy usage, and guaranteed service retrievals.

References

[1] M. Aloqaily, I. A. Ridhawi, H. B. Salameh and Y. Jararweh, "Data and Service Management in Densely Crowded Environments: Challenges, Opportunities, and Recent Developments," in IEEE Communications Magazine, vol. 57, no. 4, pp. 81-87, April 2019.

[2] X. Masip-Bruin, E. Marín-Tordera, G. Tashakor, A. Jukan and G. J. Ren, "Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems," in IEEE Wireless Communications, vol. 23, no. 5, pp. 120-128, October 2016.

[3] W. Masri, I. Al Ridhawi, N. Mostafa and P. Pourghomi, "Minimizing delay in IoT systems through collaborative fog-to-fog (F2F) communication," 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN), Milan, 2017, pp. 1005-1010.

[4] P. G. V. Naranjo, Z. Pooranian, M. Shojafar, M. Conti, R. Buyya, "FOCAN: A Fog-supported smart city network architecture for management of applications in the Internet of Everything environments," in Journal of Parallel and Distributed Computing, 2018.

[5] Y. Xiao and M. Krunz, "QoE and power efficiency tradeoff for fog computing networks with fog node cooperation," IEEE INFOCOM 2017 - IEEE Conference on Computer Communications, Atlanta, GA, 2017, pp. 1-9.

[6] W. Zhang, Z. Zhang and H. Chao, "Cooperative Fog Computing for Dealing with Big Data in the Internet of Vehicles: Architecture and Hierarchical Resource Management," in IEEE Communications Magazine, vol. 55, no. 12, pp. 60-67, December 2017.

[7] J. Huang, J. Zou and C. Xing, "Competitions Among Service Providers in Cloud Computing: A New Economic Model," in IEEE Transactions on Network and Service Management, vol. 15, no. 2, pp. 866-877, June 2018.

[8] I. Al Ridhawi, N. Samaan and A. Karmouch, "Simulator-Assisted Joint Service-Level-Agreement and Vertical-Handover Adaptation for Profit Maximization," 2012 IEEE/IPSJ 12th International Symposium on Applications and the Internet, Izmir, 2012, pp. 74-82.

[9] I. Al Ridhawi, Y. Kotb and Y. Al Ridhawi, "Workflow-Net Based Service Composition Using Mobile Edge Nodes," in IEEE Access, vol. 5, pp. 23719-23735, 2017.

[10] I. Al Ridhawi, N. Samaan and A. Karmouch, "A Policy-Based Simulator for Assisted Adaptive Vertical Handover," 2011 IEEE International Symposium on Policies for Distributed Systems and Networks, Pisa, 2011, pp. 41-48.

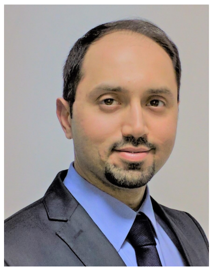
[11] I. Al Ridhawi, "Simulation-Assisted QoS-Aware VHO in Wireless Heterogeneous Networks," University of Ottawa, 2014.

[12] I. Al Ridhawi, M. Aloqaily, Y. Kotb, Y. Al Ridhawi, and Y. Jararweh, "A collaborative mobile edge computing and user solution for service composition in 5G systems," Transactions on Emerging Telecommunications Technologies.

[13] Y. Al Ridhawi, "Dynamic Composition of Service Specific Overlay Networks," University of Ottawa, 2013.

[14] "BonnMotion: A mobility scenario generation and analysis tool," [Online]. http://net.cs.uni-bonn.de/wg/cs/applications/bonnmotion/.

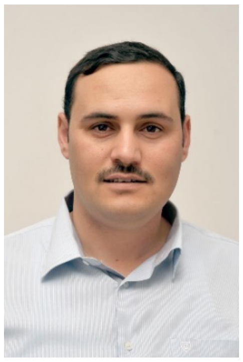
[15] "OpenTS – Java Tabu Search, http://www.coin-or.org/Ots/index.html".



Ismaeel Al Ridhawi received his BASc, MASc, and Ph.D degrees in Electrical and Computer Engineering from the University of Ottawa, Canada, in 2007, 2009, and 2014 respectively. He is a researcher in the field of wireless communications and has worked at the American University of the Middle East in Kuwait as an Assistant Professor of Computer Engineering from 2014 to 2019. He's a Senior IEEE member with many peer-reviewed publications in highly ranked magazines, journals and conference proceedings. He is an associate and guest editor in many journals. He organized a number of IEEE conferences over the years and served as session chair for a number of symposiums. He also served as a reviewer and was part of the technical program committee for numerous journals and conferences. His current research interests include service delivery and provisioning in fog and cloud computing, quality of service monitoring for wireless networks, MEC network management, and service-specific overlay networks.

Moayad Aloqaily received the M.Sc. degree in Electrical and Computer Engineering from Concordia University, Montreal, QC, Canada, in 2012, and the Ph.D degree in Electrical and Computer Engineering from the University of Ottawa in 2016. He is an instructor in the Systems and Computer Engineering Department at Carleton University, Ottawa, Canada. He is also working with Gnowit Inc. as a Senior Researcher and Data Scientist since 2016. He is a Professional Engineer Ontario (P.Eng.). His current research interests include AI and ML, Connected Vehicles, Blockchain Solutions, and Data Management. He is and IEEE member and actively working on different IEEE events. He has chaired and co-chaired many IEEE workshop and events such as ICCN INFOCOM Workshop and AICSSA'19. He is a guest editor in many journals including Cluster Computer, Internet Technology Letters, Transaction on Telecommunications Technologies, Security and Privacy. He is also an Associate Editor with IEEE Access.

Yehia Kotb received his PhD from the Faculty of Computer Science, University of Western Ontario, Canada in 2011. He is currently an Assistant Professor of computer engineering at the College of Engineering and Technology, American University of the Middle East (AUM). Prior to joining AUM, Dr. Kotb was a senior software developer at Akira Systems in London, ON, Canada. His current research interests include probabilistic process learning and multi-agent cooperation in distributed systems.



Yaser Jararweh received his Ph.D. in Computer Engineering from University of Arizona in 2010. He is currently an associate professor of Computer Science at Jordan University of Science and Technology, Jordan. He has co-authored several technical papers in established journals and conferences in fields related to cloud computing, edge computing, SDN and Big Data. He is a steering committee member and co-chair for CCSNA 2018 with Infocom. He is the General Co-Chair in IEEE International conference on Software Defined Systems SDS-2016 and SDS 2017. He is also chairing many IEEE events such as ICICS, SNAMS, BDSN, IoTSMS and many others. Dr. Jararweh served as a guest editor for many special issues in different established journals. Also, he is the steering committee chair of the IBM Cloud Academy Conference. He is associate editor in the Cluster Computing Journal (Springer), Information Processing & Management (Elsevier) and others.

Thar Baker is Senior Lecturer in Distributed Systems Engineering and Head of Applied Computing Research Group (ACRG) in the Faculty of Engineering and Technology at Liverpool John Moores University (LJMU, UK). He received his PhD in Autonomic Cloud Applications from LJMU in 2010, and became a Senior Fellow of Higher Education Academy (SFHEA) in 2018. Dr Baker has published numerous refereed research papers in multidisciplinary research areas including: Big Data, Algorithm Design, Green and Sustainable Computing, and Energy Routing Protocols. Dr Baker has been actively involved as member of editorial board and review committee for a number peer reviewed international journals, and is on programme committee for a number of international conferences. For example, he is Associate Editor of Future Generation Computer System. Dr. Baker is Expert Evaluator of EU H2020, ICTFund, and British Council.