Incentivising Resource Sharing in Edge Computing Applications

Ioan Petri[1(✉)], Omer F. Rana[2], Joseph Bignell[2], Surya Nepal[3], and Nitin Auluck[4]

[1] School of Engineering, Cardiff University, Cardiff, UK
petrii@cardiff.ac.uk
[2] School of Computer Science and Informatics, Cardiff University, Cardiff, UK
[3] CSIRO, Canberra, Australia
[4] Indian Institute of Technology Ropar, Rupnagar, India

Abstract. There is increasing realisation that edge devices, which are closer to a user, can play an important part in supporting latency and privacy sensitive applications. Such devices have also continued to increase in capability over recent years, ranging in complexity from embedded resources (e.g. Raspberry Pi, Arduino boards) placed alongside data capture devices to more complex "micro data centres". Using such resources, a user is able to carry out task execution and data storage in proximity to their location, often making use of computing resources that can have varying ownership and access rights. Increasing performance requirements for stream processing applications (for instance), which incur delays between the client and the cloud have led to newer models of computation, which requires an application workflow to be split across data centre and edge resource capabilities. With recent emergence of edge/fog computing it has become possible to migrate services to micro-data centres and to address the performance limitations of traditional (centralised data centre) cloud based applications. Such migration can be represented as a cost function that involves incentives for micro-data centres to host services with associated quality of services and experience. Business models need to be developed for creating an open edge cloud environment where micro-data centres have the right incentives to support service hosting, and for large scale data centre operators to outsource service execution to such micro data centres. We describe potential revenue models for micro-data centers to support service migration and serve incoming requests for edge based applications. We present several cost models which involve combined use of edge devices and centralised data centres.

Keywords: Edge computing · Micro-data centres · Resource sharing · Cost · Business models

1 Introduction and Motivation

With the increasing number of devices that are now generating data, it is necessary to understand how this data should be stored, processed and archived.

C. Pham et al. (Eds.): GECON 2017, LNCS 10537, pp. 204–215, 2017.
DOI: 10.1007/978-3-319-68066-8_16

Various projections have been made about the number of Internet of Things (IoT) devices we are likely to see over the next few years (often around 2020) – from Cisco, Gartner, etc. The numbers vary, but there is general agreement that this is likely to be in the order of billions, and 40% of generated data will come from sensors [5] by this projected time period. Currently, most data generated in this way is transmitted to a cloud-based data centre, processed and returned back to the user. Although this has become the dominant mode of operation, this introduces significant limitations for applications that have latency constraints, or which require response times to be within a particular threshold. Supporting application requirements through a cloud-based data centre is constrained by the last-mile network connecting the user to the network. Whereas the network around the data centre is often high speed and of high capacity, the network from the user to the first hop network component can have varying properties (especially true for mobile users).

To overcome these constraints, the fog and edge computing paradigm has been proposed, to enable processing and data storage between the user and the data centre. Fog and edge computing (FEC) resources can have significant heterogeneity (performance, data formats, energy use, type, security capability, etc.), and may be offered by a variety of different vendors, e.g. coffee shops, University campuses, Point of Presence from mobile network providers, etc [11]. There is also differing terminology associated with such FEC resources, for instance, some also refer to these as “cloudlets” [9] “micro data centres” (MDC) [1] that exist at the network edge, and peer with cloud-based data centres, etc. Many also consider a Peer-2-Peer approach for aggregating edge capacity, by enabling such cloudlets to interact with each other directly. A variety of applications have been suggested to benefit from FEC infrastructure, such as supporting storage and caching, partial processing of video feeds, monitoring physical assets (e.g. in retail or supply chain applications), on-line and interactive gaming etc. There is also emerging literature on how potential *interference* caused by co-located workloads in a cloud environment can be migrated to FEC devices, focusing on just-in-time migration of services, e.g. INDICIES [10], Caglar et al. [3].

There is now increasing literature on how such FEC infrastructure can be realised in practice [6–8]. Approaches range from the use of Raspberry Pi/Arduino-based resources that can host a Web Server (such as Flask), to more specialist micro data centres that can implemented using a computing cluster. There is however limited coverage on business and revenue models that would incentivise resource providers to offer FEC resources for third party use. We investigate this aspect in here using two application scenarios. An architecture is proposed for supporting these business models, which can be generalised to a variety of applications. We use simulations using iFogSim [4] to demonstrate the benefit of using FEC resources using our prototype applications. Section 2 provides an architecture to give context to the discussion in this paper. The reminder of the paper is as follows: In Sect. 2 we present our micro-data centre architecture and description. In Sect. 3 we describe the application scenario and overall methodology followed by the performance evaluation in Subsect. 3.3). In Sect. 4 we present business models for fog computing and conclude our work in Sect. 5.

2 FEC Architecture

Figure 1 illustrates a conceptual architecture for FEC application orchestration. A mobile device (D) has the ability to generate/ingest data and submit tasks for processing. The tasks are submitted to a micro data centre (MDC), that is "closer" to the user device (geographically or based on access latency). MDCs are capable of holding data and executing tasks with a very low latency (compared to a cloud data centre). For simplicity, we assume that each device is connected to one MDC (their "home MDC"). However, it is possible for user devices to connect to multiple MDCs. Similarly, each MDC is connected to its Cloud Data Center (CDC), i.e., home CDC; it is possible for one MDC to be connected to multiple CDCs. Existence of an MDC therefore data transfer from a user to this MDC in the first instance. The MDC can also act as a data cache to to subsequently transfer this data to a CDC if needed. To ensure support for data privacy and security (a major constraint in use of MDC at present), we assume that a device D can trust its home MDC. The home CDC is classified as semi-trusted, i.e., it can do the task as requested, but cannot guarantee the privacy of data and tasks. Here, the data and tasks are exposed to potential attacks such as an insider attack. Other MDCs and CDCs are untrusted.

We consider a decentralised distributed orchestration model, in which each computing device in the network has an orchestration agent. Agents work collaboratively towards achieving the goal, i.e. complete a user task within a budget and a given deadline, subject to security constraints.

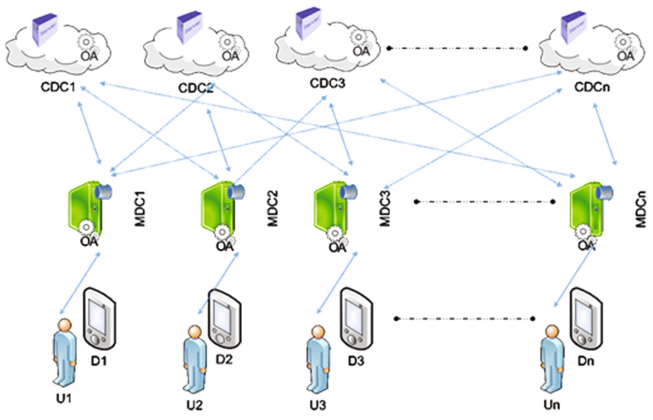

Fig. 1. Conceptual architecture for FEC application orchestration.

2.1 Best Effort Orchestration Protocol

Orchestration Agents (OAs) at both MDC and CDC work together to schedule user submitted tasks. Consider three task execution scenarios, at: (a) home MDC, (b) home CDC, and (c) remote MDC, as illustrated in Fig. 2. The diagram demonstrates a best effort orchestration protocol as an example to illustrate the concept.

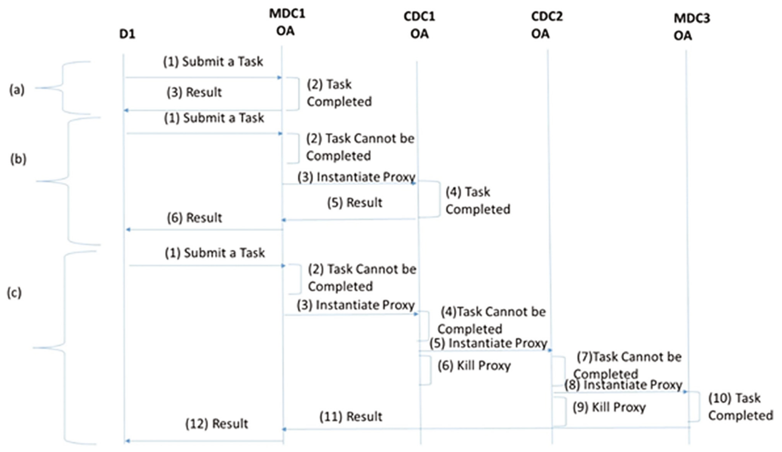

Fig. 2. Orchestration Protocol between MDC and CDC.

In scenario 1, device (D) submits a task to its home MDC (MDC1), which can meet the cost, deadline and security requirements. The task is then executed at the local MDC and result returned to D. As local MDCs are trusted, this case uses the highest level of security. In scenario 2, the task submitted by D cannot be executed at the local MDC, being unable to meet task deadline due to existing workload. The local OA forwards to CDC (CDC1) OA to create a proxy agent. The proxy agent then takes over the responsibility for task completion. The task is then completed by CDC1 and the result is returned to D via MDC1. As CDC1 is considered to be semi-trusted, this scenario represents a semi-trusted task execution at CDC (meeting the other two constraints of cost and deadline). In scenario 3, the task cannot be completed by the home CDC (CDC1), requiring other CDCs or MDCs that can complete the task. In our example, the CDC1 first contacts CDC2, but it cannot meet the latency requirement as the result of the task is much bigger in size than the original task. This means the task has to be computed closer to the device to meet the latency requirement. The CDC2 finds another MDC closer to D that can meet the latency requirement. It instantiates a new proxy agent at the remote MDC (MDC3) and passes the task to it – and the local agent is terminated. MDC3 completes the task and returns the results directly to MDC1, which then passes this to D. All proxy agent instances for a specific task are killed once the task is completed. Note that we have described normal execution scenarios using a best effort orchestration protocol. Our multi-agent based distributed orchestration can handle variations in the execution of the protocol due to failures, including failures to meet the specified requirements (cost, security and deadline).

Data transfer between OAs remains an important challenge in these scenarios, which may involve two aspects: (i) a user device submits data to the "local" MDC, and from this point onwards requires the OA at the MDC to manage and coordinate data management. This could involve migrating the data to a CDC or another MDC, based on the level of "trust" that has been identified by the user. Alternatively, the OA can also encrypt this data prior to migrating this to a CDC, depending on the sophistication and computational capacity of an MDC; (ii) a user device aims to find a suitable location to execute a task, but does not undertake any data submission before it has found a valid location for task execution (i.e. local MDC, CDC). Once a suitable location has been confirmed, and an OA has been deployed, data is transferred directly from the user device to the device hosting the OA. In case (ii), data only needs to be transferred once and the user device takes control of undertaking this transfer.

3 Application Use Cases

We consider two application scenarios to motivate the use of FEC resources, a Vehicle-2-Vehicle (V2V) and Vehicle-to-Infastructure (V2I) scenario (Sect. 3.1) and a healthcare data processing scenario (Sect. 3.2).

3.1 Vehicle-2-Infrastructure Interaction

This scenario involves determining congestion within a particular area by using: (i) localised information and alerting; (ii) global processing of the information at a CDC. The scenario is realised using three agents, represented as *controllers* in Fig. 3. A camera monitors a given traffic area, and based on observed motion, alerts an "Area Traffic Controller" (acting as MDC) – which takes into account location of vehicles (and co-position to each other), and sends updates to a "Car Controller" (alongside the road, another MDC) and a "Global Traffic Controller" (running in a CDC) – the last of these can aggregate forecast received from other Area Traffic Controllers across multiple regions. Each of these controllers have different resource capacities and latencies from the Car Controller. The Car Controller measures speed of each passing vehicle in its vicinity, and sends this data for aggregation to the other controllers. We simulate this scenario using iFogSim, giving different capacities to each of these controllers (including an energy consumption profile), as illustrated in Table 1.

In this scenario both the MDC and CDC are considered to be "trusted", i.e. data can be exchanged between any MDC and the CDC. Trust in this case can also be related to potential availability of an MDC, i.e. if the network connecting the MDC to the CDC is likely to fail, then multiple MDCs may co-exist at the same location to provide greater resilience. Trust in this instance measures the

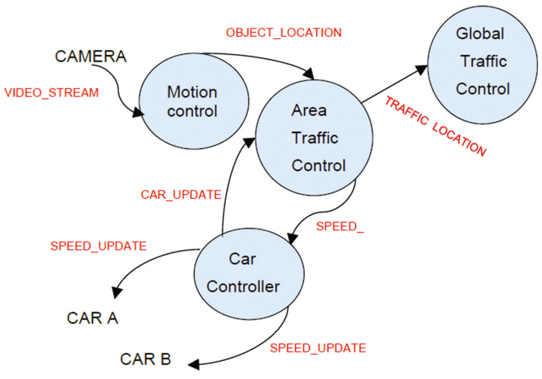

Fig. 3. Controllers for V2V/V2I scenario.

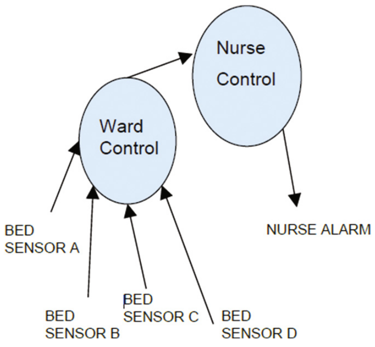

Fig. 4. Controllers for healthcare scenario.

Table 1. V2V configuration for iFogSim.

Src	Dest.	CPU (MIPS)	Network (pckts)	Data type	Direc.
Camera	Motion Detec.	1000	20000	Stream	Up
Motion Detec.	Area Traffic Contr.	3000	2000	Obj. Loc.	Up
Area Traffic Contr.	Glob. Traf. Contr.	5000	2000	Traf. Loc.	Up
Car Contr.	Area Traffic Contr.	500	2000	Car Update	Up
Car Contr.	Speed	100	100	Speed Upd.	Down

likelihood that an MDC will return a suitable reading to a CDC within a given time interval.

3.2 Healthcare Data Processing

In this scenario, all beds within a hospital ward have sensors that monitor vital signs of a patient (such as heart rate, movement pattern, ECG data, etc.). The "Ward Controller" takes this raw data and performs analysis of the data locally, and acts as an MDC in this case. Data from each patient can be analysed to identify any particular triggers or anomalies that should be identified to a clinician – referred to as the "allNurse" alarm tuple in Fig. 4. In this instance, the data remains within the ward (i.e. the context of the local MDC), and does not need to be exported to any external system. The "Nurse Control" involves interaction with local nurses, and if an alarm is received a "nurseAlarm" actuator is triggered.

Where an anomaly has been found and further analysis needs to be carried out, the data is exported to an extended data centre (CDC). This analysis could involve: (i) integrating this data with other sources available about the same individual; (ii) carrying out a multi-patient population study to investigate patients with a similar profile. In both case, the data can be presented to a clinician for further analysis. The objective in this application scenario is that in the majority of cases the data will remain within a Ward Controller (MDC), and will not need to be exported to a CDC. In this instance, the MDC is trusted as it collects data directly from a patient within a ward. Migration of the data to a CDC would imply that confidential data needs to be exported to a remote location for analysis, requiring security credentials of the CDC to be validated before this is undertaken (Table 2).

Table 2. Healthcare scenario configuration for iFogSim.

Src	Dest.	CPU (MIPS)	Network (pckts)	Data type	Direc.
Bed sensor	Heart Rate Detec.	1000	200	Raw data	Up
Heart Rate Detec.	Nurse Contrl.	2000	1000	Call Nurse	Up
Nurse Contrl.	Alarm	100	100	Alarm	Down

3.3 Evaluation: Performance

Both scenarios have been simulated via iFogSim changing a number of parameters associated with each. For the V2V scenario, the number of areas and cars/area were modified, to investigate the impact of executing car updates at local MDC (Area Controller) vs. at the CDC (Global Controller). As illustrated in Fig. 5, the benefit of using an MDC alongside a CDC is illustrated, where the processing time on a CDC which takes 11000 simulation cycles is reduced to 4000 cycles using a combination of CDC+MDC for 100 cars per area. In his simulation a total of 10 areas were considered, with this graph representing the average obtained from the simulation.

Figure 6 illustrates how data exchanged between the MDC and CDC varies, due to network congestion. The aim is to demonstrate that due to network congestion between the MDC and CDC, it does not make sense to transmit local data to the CDC for processing, necessitating se of local MDC. This could

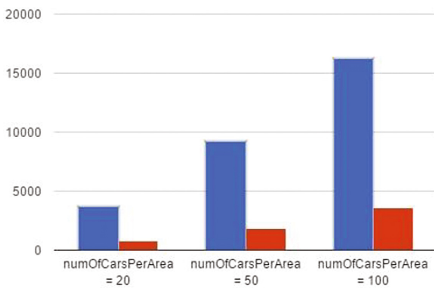

Fig. 5. MDC+CDC – Sim. Time on y-axis.

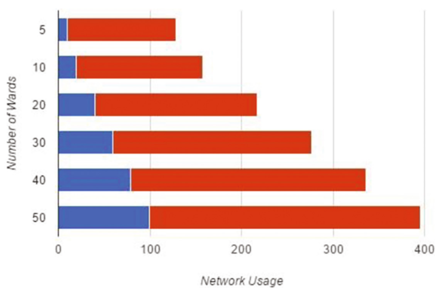

Fig. 6. MDC vs. CDC – Net. Traffic.

occur, for instance, if more wards or more beds per ward are introduced into the simulation. For this particular simulation, we maintain the number of beds/ward at 10 and increase the number of wards to 50. The y-axis shows total network usage when the simulation was run at CDC (the red bars) being significantly larger than the total network usage with MDC (blue bars). Utilizing local MDC reduces network traffic, as smaller amounts of data need to be shipped outside the local region/area of the MDC, thereby leading to lower network congestion. Within a decentralised hospital we would expect to have thousands of sensors constantly sending data so network congestion would be a real problem, looking at the results from this scenario we can see that a decentralised hospital would not be possible without use of FEC devices.

3.4 Evaluation: Cost Analysis

Based on the V2V scenario in Sect. 3.1, we consider that we have a set of MDCs – $N = \{n_1, ..., n_m\}$, responsible for managing data from Car Controllers $C = \{c_1, ..., c_n\}$. Each c_i broadcasts data to an MDC based on an analysis of congestion within a given area. Therefore, we have a number of computational jobs that need to be processed to calculate this congestion profile, with a preference for execution at a particular n_j. We consider that a job has an associated cost c, calculated as:

$$c = exec._{time} \times cost_{excution} + net._{tranfer} \times cost_{transfer} + storage_{time} \times cost_{storage}$$

where $cost_{execution}$ is the cost per CPU, $cost_{transfer}$ represents data size transferred and $cost_{storage}$ represents cost for storage of data. The costs reported in the experiments are calculated based on Amazon EC2 small instances in dollars($). We present only the MCDs cost perspective and we are not taking into consideration aspects related to delays in execution and time-to-execute constraints.

Increasing number of cars: In this scenario we investigate the impact on cost when increasing the number of vehicles for a fixed number of areas. As illustrated in Fig. 7, on the x-axis the parameters [5, 10] refer to 5 areas and 10 cars per area.

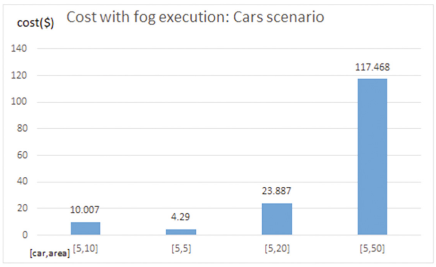

Fig. 7. Transaction costs – Cars.

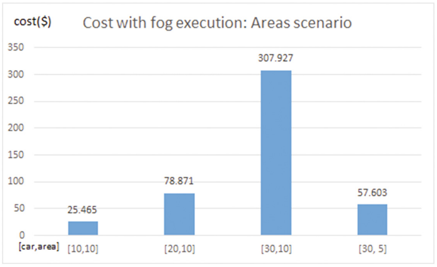

Fig. 8. Transaction costs – Areas.

We observe that the cost fluctuates with the number of cars. When increasing the number of cars per area, due to an increase in the number of devices submitting jobs, the network communication increases requiring greater congestion on FEC devices. We observe that when using 5 areas and 50 cars [5, 50] the cost reaches the highest level.

Increasing number of areas: We measure the impact of changing the number of areas with associated FEC devices monitoring a number of vehicles in their proximity. An area can host an MDC that can be used to carry out analysis on data from the vehicles. In this experiment the number of vehicles is fixed while the number of areas is modified.

In Fig. 8 we observe that changing the number of areas has a more significant impact than number of vehicles. Areas have a specific number of Car Controllers that can execute jobs by estimating car speeds and locations. The maximum cost is recorded when using 30 areas with 10 cars per area, expressed as [30, 10]. We also consider how the system reacts to different loading scenarios (i.e. changes in number of jobs that need to be processed) when both cars and areas change. Figure 9 shows how execution time changes with increasing number of cars and areas, and subsequent impact on execution time due to increase in network traffic and number of tasks.

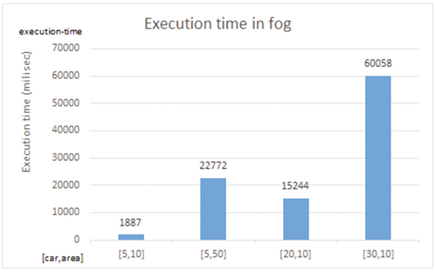

Fig. 9. Execution time: increasing cars & areas.

These simulations demonstrate how FEC devices (represented as Car and Area Controllers) can be used alongside a cloud system (represented as a Global Controller), and the associated cost of using such resources.

4 Business and Coordination Models

Given the two scenarios in Sect. 3 and the general interaction protocol outlined in Sect. 2, we generalise business models for making use of FEC resources. In this discussion we are not using FEC resources on their own, but alongside the availability of CDC(s). Several business models may become relevant when considering migration of services between MDC and CDC. As illustrated in the V2I scenario, multiple MDCs may exist between the user device (D) and the

CDC, and the potential revenue generation for each of these MDC layers should be taken on board. Similar to current broad availability of WiFi access points, we envisage three general ways of funding MDCs: (i) by cloud providers; (ii) by local businesses; (iii) by public funding. We expand on each of these aspects below. In a previous analysis [2] we investigate similar approaches for Virtual Machine migration in Fog systems.

Our business models are centered on the use of an Orchestration Agent (OA), as described in Sect. 2. An OA is responsible for executing one or more tasks on the behalf of a user application, and can launch proxy OAs on remote resources (e.g. other MDC or CDC) to achieve this. An OA also interrogates remote resources to determine potential costs of execution, and ensure that security credentials of the remote resource are valid. Our business models are centered on the use of the OA-based abstraction:

– **Dynamic MDC discovery:** In this model, an OA would be able to choose an MDC provider on-the-go, according to the MDC availability profile, security credentials, or type. The use of a service-based approach enables loose coupling, enabling an eco-system of providers to co-exist. However, there is no guarantee that integrating externally provisioned services will lead to the fulfilment of the OA objectives. An OA can therefore record "preferred" MDCs and cache this information locally. We envision cloud providers maintaining and operating MDCs in order to extend their revenue beyond resource provisioning in CDCs. Dynamic MDC discovery equates to finding an MDC in the vicinity of a user device.
– **Pre-agreed MDC contracts:** In this model, the OA would rely on informative/detailed contracts that adequately capture the circumstances and criteria that influence the performance of the externally provisioned services that are subject of the contract. An OA would therefore have pre-agreed contracts with specific MDC operators, and would interact with them preferentially. This also reduces the potential risks incurred by the OA. In performance-based contracts, an MDC would need to provide a minimum level of performance (e.g. availability) to the OA which is reflected in the associated price. This could be achieved by interaction between MDCs being managed by the same operator, or by MDC outsourcing some of their tasks to a CDC.

  Consider the following scenario to illustrate the pre-agreed MDC contracts business model: a coffee chain offers contracts for use of MDCs operated by this coffee chain across a city or country. A user wishing to make use of MDCs owned and operated by this coffee chain would need to agree to a: (i) security certificate provided by this coffee chain; (ii) have a pre-agreed subscription for use of resources provided at MDCs operated by this coffee chain. With an increasing number of branches/locations of this coffee chain, a user would have a greater choice of locations available for use of an MDC. This is equivalent to accessing wireless networks (Wifi) at locations offered by a particular provider which can have presence at multiple locations. Such a coffee shop chain may also decide to enter into preferential agreements with cloud data centre operators (e.g. public cloud providers) to integrate their regional

MDCs with CDCs operated by the data centre provider. This aligns closely with expansion of Amazon Web Services (AWS) from an infrastructure owned and operated by an ecommerce provider, to a more globally accessible public cloud infrastructure. In the same way, a coffee shop chain which operates local infrastructure to offer Wifi services to customers, could also operated micro data centres that can offer additional services to customers.
- **MDC federation:** In this model multiple MDC operators can collaborate to share workload within a particular area, and have preferred costs for exchange of such workload. This is equivalent to alliances established between airline operators to serve particular routes. To support such federation, security credentials between MDCs must be pre-agreed. This is equivalent to an extension of the *pre-agreed MDC contracts* business model, where MDCs across multiple coffee shop chains can be federated, offering greater potential choice for a user.
- **MDC – CDC exchange:** In this model an OA would contact a CDC in the first instance, which could then outsource computation to an MDC if it unable to meet the required Quality of Service targets (e.g. latency). A CDC could use any of the three approaches outlined above – i.e. dynamic MDC discovery, preferred MDCs, or choice of an MDC within a particular group. A CDC operator needs to consider whether outsourcing could still be profitable given the type of workload a user device is generating.

5 Conclusion

In this paper we present how micro-data centres can be used to support service migration and serve incoming requests from applications. We consider that micro-data centres have a cost function that involves incentives for micro-data centres for hosting various services and an associated quality of services.

We describe, using two scenarios, how FEC resources can enhance the capability of a cloud-based data centre. Revenue models for supporting the combined use of such resources are outlined, demonstrating how latency sensitive applications can be supported across such infrastructure. We use a V2V application to demonstrate how FEC resources, closer to the data generation source, can reduce processing times as the data sources are scaled.

We demonstrate that in our applications cost is directly related to the number of fog devices, network architecture and application specificities. Such factors should be considered when developing corresponding business models.

References

1. Bahl, V.: Micro datacenter middleware for mobile computing (keynote). In: ACM Middleware, Vancouver, Canada, 7–11 December 2015. http://2015.middleware-conference.org/keynote-talk-victor-bahl/. Accessed June 2017
2. Bittencourt, L., Lopes, M.M., Petri, I., Rana, O.F.: Towards virtual machine migration in fog computing. In: 10th International 3PGCIC Conference 2015, Poland, pp. 1–8, November 2015

3. Caglar, F., Shekhar, S., Gokhale, A., Koutsoukos, X.: An intelligent, performance interference-aware resource management scheme for IoT cloud backends. In: Proceedings of the 1st IEEE International Conference on Internet-of-Things: Design and Implementation, Berlin, Germany, pp. 95–105, April 2016
4. Gupta, H., Dastjerdi, A.V., Ghosh, S.K., Buyya, R.: iFogSim: a toolkit for modeling and simulation of resource management techniques in Internet of Things, edge and fog computing environments. https://arxiv.org/abs/1606.02007. Accessed June 2017
5. Noronha, A., Moriarty, R., OConnell, K., Villa, N.: Attaining IoT value: how to move from connecting things to capturing insights: gain an edge by taking analytics to the edge. Cisco White Paper (2014). http://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/iot-data-analytics-white-paper.PDF. Accessed June 2017
6. Aazam, M., Huh, E.-N.: Fog computing and smart gateway based communication for cloud of things. In: International Conference on Future Internet of Things and Cloud (FiCloud), pp. 464–470. IEEE (2014)
7. Yannuzzi, M., Milito, R., Serral-Gracia, R., Montero, D., Nemirovsky, M.: Key ingredients in an IoT recipe: fog computing, cloud computing, and more fog computing. In: IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), pp. 325–329. IEEE (2014)
8. Vaquero, L.M., Rodero-Merino, L.: Finding your way in the fog: towards a comprehensive definition of fog computing. ACM SIGCOMM Comput. Commun. Rev. **44**(5), 27–32 (2014)
9. Satyanarayanan, M., Bahl, P., Caceres, R., Davies, N.: The case for VM-based cloudlets in mobile computing. IEEE Pervasive Comput. Mag. **8**(4), 14–23 (2009)
10. Shekhar, S., Chhokra, A., Bhattacharjee, A., Aupy, G., Gokhale, A.: INDICES: exploiting edge resources for performance-aware cloud-hosted services. In: 1st IEEE International Conference on Fog and Edge Computing (ICFEC), Madrid, Spain (2017)
11. Yi, S., Li, C., Li, Q.: A survey of fog computing: concepts, applications and issues. In: Proceedings of Workshop on Mobile Big Data, Mobidata 2015, Hangzhou, China, pp. 37–42. ACM Press (2015)