Robust Resource Allocation Using Edge Computing for Smart Oil Fields

Razin Farhan Hussain[1], **Mohsen Amini Salehi**[1], **Anna Kovalenko**[1], **Saeed Salehi**[2], **and Omid Semiari**[3]
[1]High Performance Cloud Computing (HPCC) Laboratory, School of Computing and Informatics, University of Louisiana at Lafayette, Lafayette, LA, USA
[2] Petroleum & Geological Engineering Department, University of Oklahoma, Norman, OK, USA
[3]Department of Electrical and Computer Engineering, Georgia Southern University, Statesboro, GA, USA

Abstract— *Efficient and safe petroleum extraction in remote offshore oil-fields is a challenging and hazardous operation. To address this challenge smart oil-fields have been proposed and deployed for remote offshore oil-fields. A smart oil-field includes a wide range of sensors (e.g., pipeline pressure, gas density, temperature) that collectively generate terabytes of data per day which needs to be analyzed in real-time. Existing smart oil-field solutions utilize satellite communication and distant Cloud Datacenters which are unviable due to significant transfer time delay. In this paper, a robust resource allocation model using edge computing is proposed which considers connectivity, limited computational capacity, and resource intensiveness of applications in the oil-fields. To achieve robustness, the proposed model efficiently allocates tasks to an appropriate edge or cloud resource to satisfy the real-time constraint of applications. Evaluation results show that proposed model can significantly improve the performance of the system in comparison to conventional cloud-based architectures.*

Keywords: Cloud computing, Edge computing, Smart oil field, Robustness.

1. Introduction

For nearly two centuries, petroleum has been a main natural resource used to produce many industrial products such as gasoline, diesel, oil, gas, asphalt, and plastic. Over the years, high demand for petroleum products led to a scarcity of natural resources in easy-access areas and forced companies to reallocate their oil and gas (O&G) extraction sites to remote offshore (*e.g.,* sea) reservoirs [1]. Meanwhile, operations at remote sites are costly, hazardous, and constrained with limited crew and equipment resources. Moreover, petroleum extraction is a fault-intolerant process that requires ultra-high reliability, specifically in the face of a disaster (*e.g.,* oil spill [2], gas leakage [3]). Therefore, achieving efficient and safe petroleum extraction —especially when coupled with the location constraints of remote offshore reservoirs is a challenging task for oil and gas (O&G) companies.

In order to address these challenges, smart oil-fields have been proposed and deployed in over the past decade. Smart oil-fields include a diverse set of sensors (*e.g.,* Gas density, Pipeline pressure, Temperature sensors, Fire / Gas / H2S alarms, Flow monitoring & Tank levels) and computational facilities. Real-time monitoring of the site, including rigs' structure, wells, and distribution lines is performed to avoid oil and gas leakage, identify corrosion level of the infrastructure, and predict potential future incidents. It maximizes production efficiency and minimizes negative environmental impacts. Thus, a large amount of raw sensor data (between one to two terabytes) is generated in a single day of operation in a smart oil-field [4]. With an absence of a full management crew at the site, real-time decision making depends on the ability to quickly process and analyze large amounts of data. The situation can become extremely hazardous due to uncertainties in O&G extraction process (stemming from stochastic gas pressure in the reservoir and leakage of hazardous gases such as H_2S) especially in presence of on-site human workers. This imposes a major challenge —data management and analytics. According to the Cisco Public white paper on New Realities in Oil and Gas[5], 48% of all respondents involved in O&G industry admitted that proper data management and analysis is the major challenge for acquiring the best efficiency from the smart oil-fields.

Most of the generated data, such as those pertaining to drilling-platform safety, are time-sensitive and must be processed in real-time. For instance, data obtained from sensors to monitor the release of toxic gases (*e.g.,* Hydrogen Sulphide (H_2S) which is common in remote oil-fields) need to be processed in less than five seconds to preserve workers' safety [6]. Processing such volume of data requires high-end communication and computation facilities that are not available in remote (offshore) oil-fields. Satellite connection is the common vehicle of transmitting data from oil-fields to onshore Data Centers. However, the bandwidth of such connections ranges from 64 Kbps to 2 Mbps, making it 12 days to transmit one day's worth of oilfield data to an onshore Data Center [4]. As such, the major challenges for the remote offshore oil-fields can be specified as follows:

- Real-time processing of resource-intensive emergency applications.
- Constant decision-making during the extraction process.

- Real-time monitoring of the site.

To address these challenges, there is a need for a system to collect data from oil-fields, process them instantly, and make necessary decisions for a seamless and reliable O&G site. Despite recent technological advancements, to date, no comprehensive solution exists that can support bandwidth and computationally intensive operations expected in smart oil-fields. Provided the difficulties in achieving a real-time response required for time-sensitive applications in remote smart oil-fields, empowering smartness for remote oil-fields remains an open challenge. Edge computing systems, if deployed cleverly, has the potential to obviate these difficulties and enable them to take advantage of services offered by smart oil-fields.

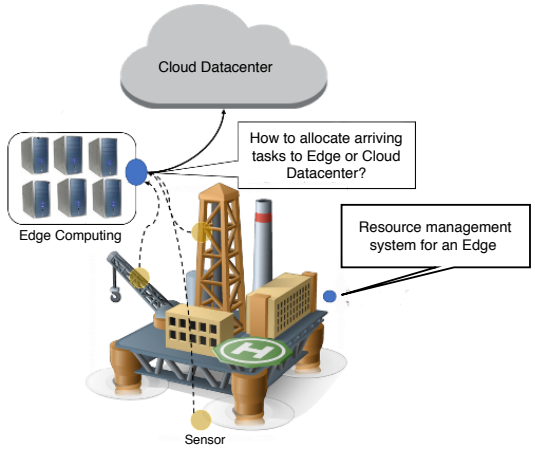


Fig. 1: A smart oil-field equipped with sensors and the edge node.

As depicted in Figure 1, our approach to the problems of smart oil-fields in remote areas is to use an Edge Computing system that is aware of the Quality of Service (QoS) demands of different applications types. It accounts for performance characteristics of the computational resource and their limited availability in the edge environment. It also considers low and unreliable connectivity to the onshore Data Centers. The question is how to efficiently process resource-intensive and time-sensitive applications in the presence of a weak and unreliable connection to onshore Data Centers? Efficiency here refers to solutions that are aware of the connectivity, limited computational capacity, and resource intensiveness of emergency management applications in remote oil-fields. Therefore, the specific problem definition in this research can be stated as *how to allocate arriving tasks to an Edge resource or in Cloud Datacenter in a robust manner (i.e., in a manner that the number of tasks missing their individual deadline is minimized)?*

The contributions of this paper are as follows:

- Proposing a resource allocation model to efficiently use limited computational resources of Edge resources and minimizes reliance on onshore resources.
- Developing a coordinator heuristic that provides robust task processing for real-time monitoring and decision making.
- Analyzing the performance of the proposed heuristic under various workload conditions.

The rest of the paper is organized as follows. Section 2 introduces the system model with formal problem statement, assumptions and scenario overview. Section 3 discusses on-time completion certainty of a task on a computing unit. Sections 4 and 5 present heuristics and performance evaluation respectively where resource allocation heuristics, experimental setup and experiments with the results are particularly described. Section 6 presents the related work. Finally, section 7 concludes the paper.

2. System Model

2.1 Formal Problem Statement

In proposed system model, a set of tasks is generated to process the sensor data and sent to a computing unit (Edge or Cloud). Every task has its own deadline within which it has to be completed. A resource allocation method, in this system, aims at maximizing the number of tasks meeting their deadlines. The set of arriving tasks can be defined as T, where $T : \{t_1, t_2, t_3, t_4 \ldots, t_n\}$ and the set of computing units S, where $S : \{s_1, s_2, s_3, s_4 \ldots, s_m\}$. The set of tasks that meet their deadlines is denoted as T_s, and we have $T_s \subseteq T$. It is assumed that a task t_i is allocated to the computing unit s_j when the task t_i can meet its deadline δ_i in that specific computing unit.

2.2 Assumptions

In our system model, an Edge machine is a stationary device with memory storage, constrained computational capacity and wireless communication capability [7]. The Edge machine is located at the offshore (remote) oil rig on the platform above the water surface. Owing to the hardware limitations, Edge machines are generally not suitable for computationally intensive tasks. Nevertheless, Cloud Datacenter has a massive computational capacity that makes it appropriate for intensive computations [8]. Different sensors (*e.g.,* capture pipeline pressure, cathodic protection, flow monitoring, air pollution, and gas density) produce various types of data (*e.g.,* numeric, image, and video) which are utilized by several applications (*e.g.,* disaster management and remote monitoring system) [9]. As such, we define *task type* to represent the variety of applications that exist in the system. For instance, task type A can be image processing for a disaster management application and task type B can be related to scheduled maintenance application [10].

Depending on the nature of different task types, some tasks are urgent (*i.e.,* delay-intolerant), and some are not, meaning that they are delay-tolerant. For example, cost-efficient drilling strategy, compressing and archiving captured videos ([11], [12], [13]) are delay-tolerant tasks,

whereas, pipeline pressure monitoring, oil spill monitoring, and temperature monitoring tasks are delay-sensitive.

We assume the arrival rate of tasks to the system is not known in advance. However, we consider scenarios in which the receiving computing unit is oversubscribed. That is, it receives the number of tasks beyond its capacity to execute them within their deadlines. Therefore, some tasks are projected to miss their deadlines. If such tasks are delay-sensitive, then there is no value in executing them and they are dropped from the system ([14], [15], [16]). For instance, in the live video stream for monitoring activities, if a streaming task misses its deadline then there is no value to process that and it should be dropped [10].

2.3 Scenario Overview

As an overview of the edge computing system, we consider for oil-field, is shown in Figure 2. In this system, upon arrival of a task to a resource allocator, it is immediately assigned to a computing unit (which can be an Edge system or Cloud Datacenter). Then, the task enters the batch queue of the computing unit to be assigned (*i.e.,* mapped) to a machine utilizing the scheduler. In this system, we define the time passed from the arrival of the task until its processing completion as the *computational delay* (denoted as d_P).

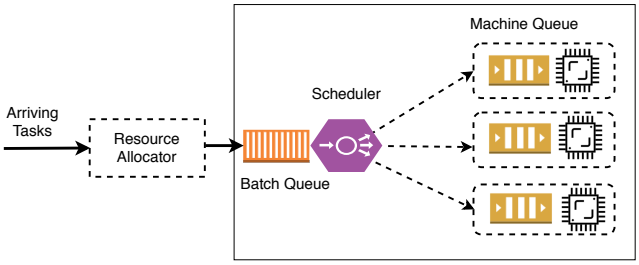


Fig. 2: Architecture of the Edge Computing Unit in Smart Oil-Field.

3. On-Time Completion Certainty of a Task on a Computing Unit

3.1 Overview

An overview of the proposed resource allocation model for the arriving tasks of a remote oil-field is demonstrated in Figure 3. The objective of the proposed model is to make the oil-field robust against uncertainties in task arrival as well as in communication delay that particularly occurs amid a disaster (*e.g.,* oil spill). To achieve the robustness goal, the proposed model aims at maximizing the number of tasks that can meet their deadlines.

Upon arrival of a task request, the task is received by a Coordinator. The Coordinator then allocates the task to the appropriate computing units. The appropriateness is characterized based on the computing unit that maximizes the likelihood of the task meeting its deadline. Within the rest of this section, we calculate this likelihood by defining the certainty of completing the task on-time. Once the resource allocation decision of the task is determined by the Coordinator, it is transferred to that particular computing unit.

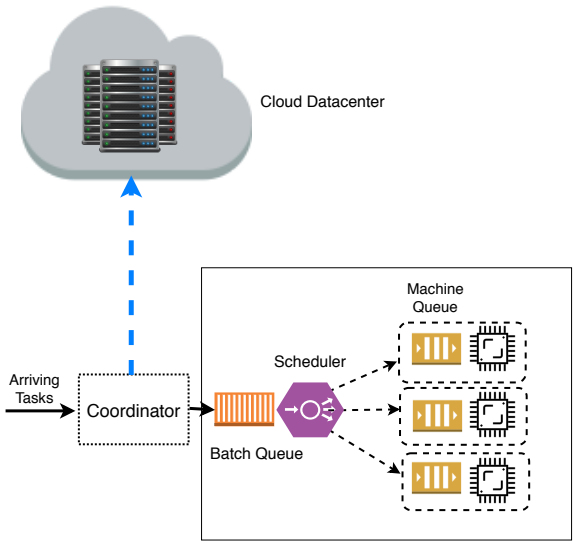


Fig. 3: A proposed model where the coordinator allocates arriving tasks to appropriate computing unit.

3.2 Calculating End-to-End Delay to Edge and Cloud Computing Systems

When the task arrives at a computing unit, it is assigned an individual deadline. An individual deadline comprises the task arrival time and the end-to-end delay the task can endure [17]. In a real-time scenario, a disaster (*e.g.,* oil spill, gas leakage) management application can submit requests that require high data rates, such as the thermal-maps or satellite images[18]. In any case, the number of applications that manage a disaster situation can be large and each application can create several tasks that potentially strain the capacity of the wireless network between sensors and the computing unit. Subsequently, communication delay, which is composed of up-link and down-link delay, can have a significant impact on the end-to-end delay of tasks. Accordingly, we consider the communication delay in deadline calculation. For an arriving task t_i, deadline δ_i is defined as: $\delta_i = arr_i + E_i + \epsilon + d_C$, where arr_i is the arrival time of the task, E_i is the average task completion time, ϵ is a constant value defined by the processing device (slack time), and d_C is the average communication delay.

For Edge computing, a communication delay (d_C) can be further broken down into the average up-link (d_U) delay and the average down-link (d_D) delay. For task t_i from an

application i to the Edge system m, the up-link delay is defined as:

$$d_U = \frac{L_i}{R_i^m} \tag{1}$$

where L_i is the task packet size, R_i^m is the transmission data rate for the link from i to m (*i.e.,* up-link bandwidth). Similarly, the down-link delay is defined as Equation 2.

$$d_D = \frac{L_i}{R_m^i} \tag{2}$$

where R_m^i is the transmission data rate from m to i (*i.e.,* down-link bandwidth).

In the case of cloud computing system, since the Datacenters are located far from the offshore oil-field, the propagation delay of tasks becomes remarkable. Therefore, we also need to consider propagation delay (denoted as d_R) when calculating the communication delay. Satellite communication is commonly utilized for transferring the task in remote oil-fields to the Cloud Datacenters. Hence, the propagation delay is defined based on Equation 3.

$$d_R = \frac{Distance}{Speed} \cdot 2 \tag{3}$$

In the equation 3, we consider a round trip time which is twice the propagation delay. Therefore, communication delay for the Cloud Datacenter consists of a transmission delay along with a propagation delay. As such, we define communication delay for the cloud computing system based on Equation 4.

$$d_C = \frac{L_i}{R_i^m} + \frac{Distance}{Speed} \cdot 2 + \frac{L_i}{R_m^i} \tag{4}$$

3.3 Completion Time Certainty

To maximize the likelihood for an arriving task meeting its deadline, we propose that Coordinator makes resource allocation decisions based on the computing unit (*i.e.,* Edge or Cloud Datacenter) the task receives the highest certainty to complete on-time (*i.e.,* before its deadline). For an arriving task t_i on a computing unit j, let E_{ij} be the mean completion time of task t_i on j. The value of E_{ij} can be obtained by analyzing the historic completion time information of task t_i on machine j. Then, we define *on-time completion certainty*, denoted as $C_j(t_i)$, based on Equation 5.

$$C_j(t_i) = \delta_i - E_{ij} \tag{5}$$

For an arriving task, as shown in Figure 3, the Coordinator calculates the certainty of completing the task both on the Cloud Datacenter and on the Edge system. After that, the task is allocated on the computing unit that provides a higher certainty of task completion. More specifically, once the Coordinator receives a task t_i, initially it calculates the deadline (δ_i) for this task. Then, it calculates the certainty of completing the task t_i, denoted as $C(t_i)$, by deducting the completion time of that particular task from its deadline. The value of certainty indicates the time remaining before the deadline occurs. The higher the difference, the more chances the task can meet its deadline. Therefore, the Coordinator checks the certainty of an arriving task both for the Edge system and the Cloud Datacenter. Finally, Coordinator assigns the task to the computing unit that offers the highest certainty.

4. Resource Allocation Heuristics

4.1 Coordinator

The Coordinator makes an efficient resource allocation decision for processing each task within its individual deadline. The arriving task is instantly allocated to an appropriate computing unit determined by the Coordinator. The Coordinator includes a buffer that handles multiple tasks arriving simultaneously. Once the task is allocated, it cannot be reallocated due to data transfer overhead. In this section, we introduce resource allocation heuristics that can be utilized by the Coordinator.

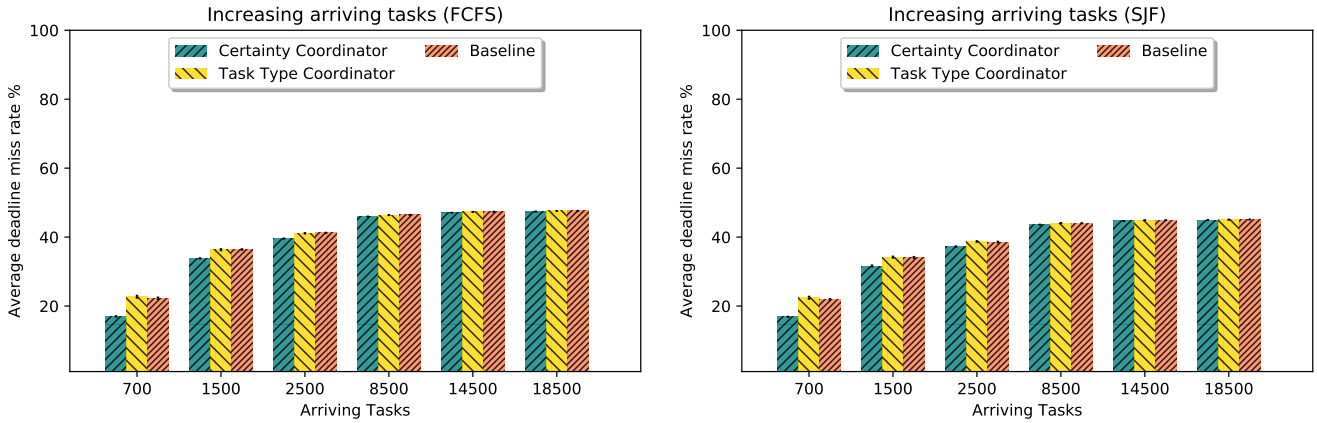
After the resource (task) allocation, the scheduler of every Edge Node allocates the task to the VMs. It is realized that this scheduling policy impacts the robustness of the system, in terms of a number of tasks meeting their deadlines. Consequently, in our research, we evaluate the impact of using two different scheduling policies on the edge as follows:

First Come First Serve (FCFS): FCFS is one of the popular baselines for the task scheduling policy. According to this policy, the tasks that arrive earlier get scheduled first. The scheduler allocates tasks from the head of the queue to the machines local queue to load its data and execute it. We determine that machines have a limited local queue. Accordingly, the edge internal scheduler is invoked whenever a free spot appears in local queue of a machine.

Shortest Job First (SJF): SJF schedules the tasks that have the shortest execution time. Initially, the heuristic organizes the tasks in the batch queue according to their execution time in ascending order. Therefore, the tasks with a shorter execution time stay in the head of the queue and are scheduled immediately whenever a free spot appears in machines local queue. The execution time of a task can be obtained from historic execution time information.

4.2 Baseline Heuristic

Existing remote oil-fields only use onshore Cloud Datacenters and satellite communication for processing the arriving tasks. As such, for the baseline method, the first heuristic naïvely allocates every task to the onshore Cloud Datacenter for processing.



(a) FCFS scheduling

(b) SJF scheduling

Fig. 4: Deadline missing rate is measured using three heuristics (Maximum Certainty, Task Type and Baseline). Two different scheduling policies (FCFS and SJF) are used for evaluation of the system with respect to deadline miss rate.

4.3 Maximum Certainty (MC) Heuristic

The objective of this heuristic is to maximize the robustness of the system through maximizing individual task's certainty of meeting its deadline. It allocates an arriving task to the computing unit that maximizes certainty of the task. Upon arrival of a task to the Coordinator, it calculates the certainty of that task for the Edge and the Cloud Datacenter and allocates it to the computing unit that has the highest task certainty.

4.4 Task Type Based (TT) Heuristic

This heuristic uses the task type as the decisive factor for resource allocation. Recall that urgent tasks (*i.e.,* those related to managing oil spill) have shorter deadlines. This heuristic harnesses this knowledge and allocates the arriving tasks that are urgent to the Edge Node. Other tasks that are of non-urgent type are dispatched to onshore Cloud Datacenters for processing. We should note that this heuristic does not account for communication delay in its decision making.

5. Performance Evaluation

5.1 Experimental Setup

CloudSim simulation [19] is used to evaluate our system's performance . CloudSim is a discrete event simulator that provides Cloud and Edge Computing models. In our simulation Edge Nodes are Datacenters which are devices with limited computational capacity (8 cores). Each core is utilized by one Virtual Machine (VM). All VMs in an Edge Node are homogeneous, i.e. they have the same computational power (MIPS). We consider a large Cloud Datacenter including 16 cores, is allocated to manage tasks of the oil-field under study. We implement the bandwidth for the Cloud Datacenter and simulate the propagation delay to calculate the communication latency in our study. All these parameters are scalable to larger magnitude.

To implement the tasks of smart oil-field we have used CloudSim's prototype structure defined as *cloudlets*. An Edge Node and a Coordinator component have been developed for allocating the tasks according to our proposed heuristic.

5.2 Generated Workload

In our simulation, execution times for tasks are generated using Gaussian distribution[20]. We use results of the Extreme Scale System Center (ESSC) at Oak Ridge National Laboratory (ORNL) ([15], [16]) to implement the arrival time for each task. Nevertheless, the arrival time from this workbench is very sparse which is not applicable to our work. In order to make it relevant to our work, we tune the workbench and set the number of tasks to 24,000 to generate a dense arrival time.

Initially, every Edge and Cloud are assigned initial workload that represents their current workload. In addition to the initial workload, Coordinator receives the generated workload (testing workload). This workload is the one we consider for our result's evaluation. By manipulating the number of tasks in the initial workloads, we are able to control the system's oversubscription level. For the sake of accuracy and to remove any uncertainty in the results, we conduct each experiment 10 times and report the mean and 95% confidence interval of the results.

5.3 Experiment Results

To study the performance of our model thoroughly, we evaluate the system under various number of arriving tasks. We analyze the impact of the increasing number of tasks within testing workload on the oversubscribed system. We

consider the minimum testing workload to be a batch of 700 tasks and the maximum to be a batch of 18,500 tasks. In each experiment, we count the number of tasks that miss their deadlines. Each workload is evaluated against the three resource allocation heuristics mentioned in the previous section.

Figure 4 demonstrates the results obtained from the experiments. Sub-figure (a) and (b) of Figure 4 represents the results obtained using the FCFS and SJF scheduling policy respectively. As we can see, Coordinator results lead to a higher performance when it uses the certainty heuristic rather than Baseline and Task Type heuristics. The certainty heuristic accounts for the deadline of the individual task and tends to support real-time tasks processing in Edge. The experiment indicates that even a limited edge node deployed on the edge of the network can improve the robustness of the whole system. That is, it reduces the deadline miss rate of the system in face of oversubscription.

We observe in Figure 4 that only 17% of all tasks miss their deadlines using the certainty heuristic, whereas more than 22.2% of the tasks miss their deadline with the baseline heuristic. Although the deadline miss rate grows with the increase of the number of tasks within workload for all three heuristics, the certainty heuristic, outperforms in the beginning to the midrange of the tasks increase. From the midrange to the largest tasks workload, as the edge machines are saturated with the assigned tasks, the certainty-based heuristic performs nearly the same as the other two heuristics. We assume that the performance of the certainty heuristic depends on the level of the initial oversubscription of the system, but we leave finding the proof of our assumption for the future research.

We also notice that certainty-based heuristic performs slightly better using the SJF scheduling rather than FCFS. The reason for that can be the efficient scheduling of short tasks first. Hence, we can conclude that certainty-based heuristic outperforms the Task Type and Baseline heuristics and can provide a higher robustness for the whole system in face of oversubscription.

6. Related work

Edge Computing concepts have been previously proposed in the literature for delay tolerant networks. Lorenzo *et al.,* in [21] proposed resource allocation methods for Edge Computing environments that considers unreliable network connectivity. However, in similar kind of research works ([22], [23], [24]) authors neither consider the case of emergency management applications nor the heterogeneity of the Edge resources, while performing resource allocation.

The more specific problem of resource provisioning for real-time disaster management applications in Edge Computing with low-connectivity to the back-end Data Centers has not been explored in the context of remote smart oil-fields and there is a limited research on these issues in other contexts. Efforts towards smart oil-fields have been predominantly on analyzing the big data extracted from oil wells by Cameron *et al.,* in [25] or applying machine learning methods to reduce exploration or drilling costs by Parapuram *et al.,* in [26]. Warning systems for early prediction of disasters were analyzed by Xu *et al.,* in [27]. These solutions are all reliant on onshore Data Centers [9] which are not viable for remote and offshore oil-fields.

To date, limited work exists from academia and industry to design wireless communication networks for remote smart oil-fields ([1], [4], [29], [30], [31], [32]). Particularly, prior art cannot address the key challenges of remote operations, due to the following reasons. *First*, the works in [33] rely on satellite communications between the oil rigs and offshore management centers. However, satellite communication is not suitable for real-time decision making during oil extraction process, as the delay can be substantially large. *Second*, the works in [34] assume the existence of a macro cell Base Station at a nearby onshore location that provides wireless support for oil rigs. Nonetheless, remote reservoirs can be very far away from the shore. *Third*, most of existing networks [35] operate at sub-6 GHz frequency bands with limited capacity that cannot manage large data rates and URLLC requirements of smart oil-fields. *Fourth*, ad-hoc communications protocol with random channel access cannot satisfy the ultra-reliability requirements in smart oil-fields with a dense number of wireless devices.

7. Conclusions

In this paper, we presented a robust resource allocation model using Edge Computing that copes with the uncertainties that exists both in communication and computation for offshore smart oil-fields. We leveraged the task deadline and historical data for completion time to devise a coordinator that operates near the smart oil-field (*i.e.,* at the Edge level) and distribute arriving tasks to the most certain computing unit (Edge or Cloud) in the face of oversubscription. The method considers the nature of the task type (*e.g.,* real-time and delay intolerant). Experimental results express that our proposed model can improve the robustness of the system compared to distant Cloud Datacenter working similarly in case of disaster or emergency-related tasks. From simulation results, it is observed that for various amounts of arriving task, our proposed heuristic outperforms the baseline and the Task Type heuristics. Approximately 17% of all tasks miss their deadline using the certainty-based coordinator, where more than 22.26% tasks miss their deadline with baseline and task type heuristics. In future, we plan to extend our proposed model to work more efficiently by considering heterogeneity within each computing unit (Edge and Cloud). We also plan to study the impact of approximate computing on the performance of the system.

References

[1] WoodMackenzie. Why are some deepwater plays still attractive? White paper, September 2017.

[2] Merv Fingas and Carl Brown, "Review of oil spill remote sensing," *Journal of Marine Pollution Bulletin*, vol. 83(1), pp. 9–23, Jun 2014.

[3] 10 deadliest accidents in oil and gas industry - whatwhenwhy. [Online]. Available: http://whatwhenwhy.net/deadliest-accidents-in-oil-and-gas-industry/

[4] Cisco. A New Reality for Oil & Gas: Data Management and Analytics. White paper, April 2015.

[5] New realities in oil and gas: Data management and analytics. https://www.cisco.com/c/dam/en_us/solutions/industries/energy/docs/OilGasDigitalTransformationWhitePaper.pdf, accessed 2017.

[6] Sudhir Kumar Pandey, Ki-Hyun Kim, and Kea-Tiong Tang. A review of sensor-based methods for monitoring hydrogen sulfide. *TrAC Trends in Analytical Chemistry*, 32:87 – 99, 2012.

[7] Kai Liu and Victor CS Lee, "Rsu-based real-time data access in dynamic vehicular networks," in *Proc. of the 13th International Conference on Intelligent Transportation Systems (ITSC)*, Sept 2010, p. 1051–1056.

[8] Mohsen Amini Salehi and Rajkumar Buyya, "Adapting Market-Oriented Scheduling Policies for Cloud Computing," in *Proc. of Algorithms and Architectures for Parallel Processing ICA3PP'10*, 2010, paper 6081, p. 351–362.

[9] Wan Jun and Wang Hongyuan. Application of internet of things technology in digital oilfield. *Automation in Petro-Chemical Industry*, 1:001, 2015.

[10] Matin Hosseini, Mohsen Amini Salehi, and Raju Gottumukkala, "Enabling interactive video streaming for public safety monitoring through batch scheduling," in *Proc. of the 19th International Conference on High Performance Computing and Communications*, Dec. 2017.

[11] Xiangbo Li, Mohsen Amini Salehi, Magdy Bayoumi, and Rajkumar Buyya, "CVSS: A Cost-Efficient and QoS-Aware Video Streaming Using Cloud Services," in *Proc. of the 16th IEEE/ACM International Conference on Cluster Cloud and Grid Computing CCGrid'16*, May 2016.

[12] X. Li, M. A. Salehi, M. Bayoumi, N. Tzeng, and R. Buyya, "Cost-efficient and robust on-demand video transcoding using heterogeneous cloud services," *Journal of Transactions on Parallel and Distributed Systems (TPDS)*, vol. 29, pp. 556–571, Oct. 2017.

[13] Xiangbo Li, Mohsen Amini Salehi, and Magdy Bayoumi, "VLSC: Video Live Streaming Based On Cloud Services," in *Proc. of Big Data & Cloud Applications Workshop, as part of the 6th IEEE International Conference on Big Data and Cloud Computing BDCloud '16*, Oct 2016.

[14] Bhavesh Khemka, Ryan Friese, Sudeep Pasricha, Anthony A Maciejewski, Howard Jay Siegel, Gregory A Koenig, Sarah Powers, Marcia Hilton, Rajendra Rambharos, and Steve Poole, "Utility driven dynamic resource management in an oversubscribed energy-constrained heterogeneous system," in *Proc. of Parallel & Distributed Processing Symposium Workshops IPDPSW*, May 2014, p. 58–67.

[15] Bhavesh Khemka, Ryan Friese, Sudeep Pasricha, Anthony A. Maciejewski, Howard Jay Siegel, Gregory A. Koenig, Sarah Powers, Marcia Hilton, Rajendra Rambharos, and Steve Poole, "Utility maximizing dynamic resource management in an oversubscribed energy-constrained heterogeneous computing system," *Journal of Sustainable Computing: Informatics and Systems*, vol. 5, pp. 14– 30, Mar 2015.

[16] Bhavesh Khemka, Ryan Friese, Luis D Briceno, Howard Jay Siegel, Anthony A Maciejewski, Gregory A Koenig, Chris Groer, Gene Okonski, Marcia M Hilton, Rajendra Rambharos, et al, "Utility functions and resource management in an oversubscribed heterogeneous computing environment," *Journal of Transactions on Computers*, vol. 64(8), pp. 2394–2407, Aug 2015.

[17] Mohsen Amini Salehi, Jay Smith, Anthony A. Maciejewski, Howard Jay Siegel, Edwin K. P. Chong, Jonathan Apodaca, Luis D. Briceno, Timothy Renner, Vladimir Shestak, Joshua Ladd, Andrew Sutton, David Janovy, Sudha Govindasamy, Amin Alqudah, Rinku Dewri, and Puneet Prakash, "Stochastic-based robust dynamic resource allocation for independent tasks in a heterogeneous computing system," *Journal of Parallel and Distributed Computing (JPDC)*, vol. 97(C), Nov. 2016.

[18] Camilla Brekke and Anne H.S. Solberg, "Oil spill detection by satellite remote sensing," *Journal of Remote Sensing of Environment*, vol. 95(1), pp. 1–13, Mar. 2005.

[19] Rodrigo N Calheiros, Rajiv Ranjan, Anton Beloglazov, César AF De Rose, and Rajkumar Buyya, "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience SPE*, vol. 41(1), p. 23–50, Aug. 2011.

[20] Jun Li, Carlos Natalino, Dung Pham Van, Lena Wosinska, and Jiajia Chen, "Resource management in fog-enhanced radio access network to support real-time vehicular services," in *Proc. of 1st IEEE International Conference on Fog and Edge Computing (ICFEC)*, May 2017, p. 68–74.

[21] Beatriz Lorenzo, Juan Garcia-Rois, Xuanheng Li, Javier Gonzalez-Castano, and Yuguang Fang, "A robust dynamic edge network architecture for the internet-of-things," *Journal of arXiv preprint arXiv:1710.04861*, Oct. 2017.

[22] S. Wunderlich, J. A. Cabrera, F. H. P. Fitzek, and M. Reisslein, "Network coding in heterogeneous multicore iot nodes with dag scheduling of parallel matrix block operations," *Journal of Internet of Things*, vol. 4(4), pp. 917–933, Aug 2017.

[23] Y. Wang, X. Lin, and M. Pedram, "A nested two stage game-based optimization framework in mobile cloud computing system," in *Proc. of the 7th IEEE International Symposium on Service-Oriented System Engineering*, Mar 2013, p. 494–502.

[24] Z. Hu, Y. Wei, X. Wang, and M. Song, "Green relay station assisted cell zooming scheme for cellular networks," in *Proc. of the 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery*, Aug 2016, p. 2030–2035.

[25] David Cameron et al, "Big data in exploration and production: Silicon snake-oil, magic bullet, or useful tool?," in *Proc. of SPE Intelligent Energy Conference & Exhibition*, Apr 2014.

[26] George K Parapuram, Mehdi Mokhtari, Jalel Ben Hmida, et al, "Prediction and analysis of geomechanical properties of the upper bakken shale utilizing artificial intelligence and data mining," in *Proc. of Conference on Society of Petroleum Engineers/American Association of Petroleum Geologists/Society of Exploration Geophysicists Unconventional Resources Technology*, July 2017.

[27] B. Xu, W. Wang, Y. Wu, Y. Shi, and C. Lu, "Internet of things and big data analytics for smart oil field malfunction diagnosis," in *Proc. of the 2nd IEEE International Conference on Big Data Analysis ICBDA'17*, Mar. 2017, p. 178–181.

[28] Zehua Zhang and Xuejie Zhang, "A load balancing mechanism based on ant colony and complex network theory in open cloud computing federation," in *Proc. of 2nd International Conference on Industrial Mechatronics and Automation (ICIMA)*, May 2010, paper 2, p. 240–243.

[29] S.R.B. Prabhu, E. Gajendran, and N. Balakumar, "Smart oil field management using wireless communication techniques," *Journal of Inventions in Engineering & Science Technology*, vol. (2), pp. 2454–9584, Jan 2016.

[30] Gartner. Hype cycle for upstream oil and gas technologies. White paper, July 2014.

[31] Gartner. Top 10 technology trends impacting the oil and gas industry in 2015. White paper, March 2015.

[32] Gartner. Hype cycle for upstream oil and gas technologies, 2014. White paper, July 2014.

[33] PM Bogaert, W Yang, HC Meijers, CM van Dongen, M Konopczynski, et al, "Improving oil production using smart fields technology in the sf30 satellite oil development offshore malaysia," in *Proc. of Offshore Technology*, May 2004.

[34] Boselin Prabhu, E Gajendran, and N Balakumar. Smart oil field management using wireless communication techniques. 2017.

[35] Mohammad reza Akhondi, Alex Talevski, Simon Carlsen, and Stig Petersen, "Applications of wireless sensor networks in the oil, gas and resources industries," in *Proc. of 24th Conference on Advanced Information Networking and Applications AINA*, 2010, p. 941–948.