# R-MDP: A Game Theory Approach for Fault-Tolerant Data and Service Management in Crude Oil Pipelines Monitoring Systems

Safuriyawu Ahmed(✉), Frédéric Le Mouël, Nicolas Stouls, and Jilles S. Dibangoye

Univ Lyon, INSA Lyon, Inria, CITI, EA3720, 69621 Villeurbanne, France
safuriyawu.ahmed@insa-lyon.fr

**Abstract.** Failures in pipeline transportation of crude oil have numerous adverse effects, such as ecological degradation, environmental pollution and a decrease in revenue for the operators, to mention a few. Efficient data and service management can predict and prevent these failures, reducing the downtime of the pipeline infrastructure, among other benefits. Thus, we propose a two-stage approach to data and service management in Leakage Detection and Monitoring Systems (LDMS) for crude oil pipelines. It aims to maximise the accuracy of leakage detection and localisation in a fault-tolerant and energy-efficient manner. The problem is modelled as a Markov Decision Process (MDP) based on the historical incident data from the Nigerian National Petroleum Corporation (NNPC) pipeline networks. Results obtained guarantee detection in at least two deployed nodes with a minimum localisation accuracy of 90%. Additionally, we achieved approximately 77% and 26% reduction in energy consumption compared to a pessimistic strategy and a globalised heuristic approach, respectively.

**Keywords:** pipeline monitoring · iot · wsn · game theory · crude oil · data analytics

## 1 Introduction

The operations and processes across the oil and gas industry's three sectors (upstream, midstream, and downstream) present vast data. Although there are several efforts to gain insights into the data in the upstream and downstream sectors through big data analytics [1,2], the data in the midstream (transportation) sector is largely left unexploited. In this sector alone, every 150,000 miles of pipeline produces up to ten terabytes of data [3] through monitoring systems. Such data enable the timely detection and localisation of leakages in the pipeline and allow the deployment of services like predictive analysis, emergency service and others. The monitoring systems utilised include legacy Leak Detection


S. Longfei and P. Bodhi (Eds.): MobiQuitous 2022, LNICST 492, pp. 44–64, 2023.
https://doi.org/10.1007/978-3-031-34776-4_3

Monitoring Systems (LDMS) such as Supervisory Control And Data Acquisition (SCADA) and, more recently, Wireless Sensor Networks (WSN) and Internet of Things (IoT)-based systems.

Nowadays, data and service management has become paramount in the midstream sector due to ageing infrastructure, outdated technology, and incessant node vandalisation [4,5]. Its benefits include the reduction of the annual downtime of LDMS by 70% and the associated cost by 22% through timely failure detection and predictive maintenance [4]. Aliguliyev *et al.* and Hajirahimova, in their work [2,6], also highlight other benefits of data management and analytics in this context.

However, data and service management must be efficiently done as it affects various performance metrics in WSN and IoT-based LDMS. For example, despite enabling data processing through fog computing in these systems, their responsiveness, the energy consumption [7] and real-time management [8] in the form of network fluctuation, latency, communication failure, and node failures are nonetheless a challenge.

Therefore, this paper presents our work on efficient and robust data and service management in WSN and IoT-based LDMS. The objectives are:

1. To analyse the historical data related to pipeline failures in Nigerian National Petroleum Network (NNPC) pipelines.
2. To propose a Regionalised Markov Decision Process (R-MDP) for ensuring a similar level of performance across the NNPC pipeline network.
3. To determine strategies using the MDP on the pipeline network's defined regions for minimal total energy consumption.

We aim to achieve this by exploiting the fog computing paradigm for distributed data and service management through simultaneous data and service *placement, replication and migration*. This strategy is modelled as a Markov Decision Process management based on the historical incident data from the NNPC pipeline networks. Our technique is further divided into two stages: first, we determine the performance measure for each predefined pipeline region; then, we find the optimal value that minimises the energy consumption in the region considered.

The remainder of this work is structured as follows: In Sect. 2, we discuss related work, followed by our contribution in Sect. 3. In Sect. 4, we conclude the paper and present our future work.

## 2 Related Work

Cloud computing enabled advanced data analytics, optimisation and decision-making in IoT applications. However, the increased load for time-sensitive applications and real-time monitoring can significantly affect the system's performance. Fog computation and edge analytics addressed such limitations of centralised cloud computing [9] by its integration into the network design. This design significantly increases the scalability of the network by reducing latency

and computational overhead at the cloud servers. In addition, maintenance or enhancement of system performance such as improved *fault tolerance* in real-time operations is realised. Other approaches have also been adopted to improve the overall system's performance.

Therefore, this section discusses several approaches to improved efficiency in WSN/IoT-based applications, real-time applications and fog-based infrastructures. In particular, we examine placement, communication strategies and game theories for optimisation in the following subsections.

### 2.1 Placements and Communication Strategies in Fog Architectures

Although fog-based infrastructures can reduce latency and computational overhead for real-time operations, misplacing data in the fog nodes can have a detrimental effect. Some of the applications of fog-based systems were demonstrated in [10], where it extends the data analytic capability of cloud computing in the context of smart cities using smart pipeline monitoring prototypes. Giordano *et al.* [11] also showed its application operating a platform that incorporates smart agents.

Paramount, however, are placement strategies on fog-enabled infrastructures aimed to reduce the overall network *latency* and increase the *fault tolerance* of the system. Naas *et al.* [12] proposed a runtime data placement algorithm in a fog-based architecture focused on the nature of the data, the holder node's behaviour and location. The results show that overall latency was reduced by 86% compared to cloud solutions and 60% to simple fog solutions. Eral *et al.* [13] worked on a replica placement algorithm using the size, location and priced storage as the constraints for reducing latency in edge networks. Their work yielded a 14% to 26% reduction -per tradeoffs- in latency compared to replicas' absence. Shao *et al.* in [14] also worked on placing data replicas for IoT workflows in both fog and cloud environments. Their work is based on an intelligent swam optimisation algorithm based on user groups, data reliability, and workflows. Results show improvement in comparison to other research.

In tandem with data placement, service placement plays a crucial role in the overall efficiency of a fog-enabled environment, i.e. generic or wrong placements could result in latency increment [15]. Hence, Velasquez *et al.* [15] defines an IoT service placement architecture fusing cloud and fog computing and constrained by the system's operating condition and latency requirements. They used a three-module (service repository, information collection, and service orchestrator) generic and scenario-agnostic placement algorithm. Services conforming to the state of the network and the user and server's location are placed in the fog nodes. Wang *et al.* [8] proposed an optimal data scheduling policy operating multiple channels based on a four-layer fog computing architecture. It comprises the device, data scheduler, Jstorm, and cloud layers. The Jstorm layer integrates geographically distributed fog nodes into several clusters. As such, generated big data is split into several blocks and transferred to different Jstorm clusters for processing. Simulation shows a 15% gain over other data scheduling policies.

Additionally, Elsayed *et al.* [16] worked on distributed fault tolerance of sensor node hardware WSNs. Experimental results proved their method performed better than the compared scheme by tolerating approximately 67% of the encountered failures. Yuvaraja *et al.* [17] also worked on fault tolerance in WSN by detecting and recovering node failures using a least disruptive topology repair mechanism.

Other considerations for efficient systems are the mechanisms of data transmission or flow between data producers (clients) and data consumers (agents). Commonly used are the publish-subscribe (PB) messaging paradigms between clients and agents [18]. Ioana *et al.* in their work [19] demonstrated the applicability of the PB systems in several complex scenarios, for example, the Open Platform Communication Unified Architecture protocol for Industry 4.0. Primarily, they exhibited how a multi-channel User Datagram Protocol (UDP) communication strategy for PB systems enables transmitting high-volume data like images in a time frame fitted for the industry. Aslam *et al.* [20] worked on adaptive methods to handle unknown subscriptions in a low-latency PB model for processing multimedia events. Their system achieved between 79% and 84% accuracy. Jafarpour *et al.* [21] focused on computing and transmission cost minimisation through content-subscribers-based requested formats.

### 2.2 System Optimisation Using Game Theory

Game theory is increasingly being used to improve application efficiency and optimise system performance. Garg *et al.*, in their work [22], evaluated three dynamic placements (greedy approximation, integer programming optimisation and learning-based algorithms) for maximal user equipment availability employing minimal infrastructure. A drone swarm application experiment indicates that all tested techniques met the latency requirement. Nevertheless, the learning-based algorithm performed better in terms of the minimal variation in solution providing a more stable deployment and thereby guaranteeing a reduction in infrastructural cost. Also, on placement optimisation, Ting *et al.* [23] presented an optimal provision of edge services such as storage, communication and computational resources. Using a trace-driven simulation, they carried out an analysis of the results obtained on optimal request scheduling (ORS as the baseline), greedy service placement with optimal request scheduling (GSP ORS) and greedy service placement with greedy request scheduling (GSP GRS). For joint service placement and resource scheduling, both GSP ORS and GSP GRS, including their linear programming relaxation, achieved optimal or near-optimal solutions.

Cai *et al.*, in their work [24], introduced a Reinforcement Learning Heuristic Optimisation (RLHO) framework aimed at the provision of better initial values for the algorithm. They conducted a comparative analysis between RLHO, simulated annealing and proximal policy optimisation. An experiment on a bin packing problem validated that RLHO outperformed pure reinforcement learning. Likewise, the following research based on game theory aims to improve pipeline monitoring systems.

Islam *et al.* [25] and Rezazdeh *et al.* [26] worked on third-party interference such as terrorist attacks on pipeline infrastructures. The former proposed a Stackelberg competition-based attacker-defender model to find the equilibrium between pipeline security and possible attacks. They proved that in an equilibrium state, the monitoring system achieves the best result by maintaining its strategy, assuming both the defender and attacker act rationally. On the other hand, the latter proposed two-player non-zero-sum modelling of the problem. They also assumed that both players acted rationally based on the chosen indices. Two methods were proposed to solve this problem: a local optimisation for comprehensive analysis of the effect of countermeasures on attacks and a global optimisation enabling the security personnel (defender) to provide a solution from the attacker's perspective. Rezazadeh *et al.* [27] worked on modelling a monitoring system for pipeline security using the Bayesian Stackelberg game. They proposed a scheduling policy based on time and distance discretisation. The proposed framework enables the ranking of security risks, allowing different patrol paths to be utilised.

While these works considered the efficient placement of data and services to reduce latency in fog-enabled architectures, they have not considered the robustness of their approach to failures. In addition, to the best of our knowledge, no work has considered a data-driven MDP for efficiently managing failures in crude oil pipelines. Thus, in our work, we take cognisance of failures, their nature and history to guarantee the monitoring system's availability and performance by optimising the placement strategies across a multi-layered architecture.

## 3 Data-Driven Resilient and Regionalised MDP (R-MDP)

This research implements fault-tolerant data and service management through distributed data and service *placement, replication and migration*. As a prerequisite, we propose a three-layer fog-based architecture shown in Fig. 1 that allows data and service placements closer to the users (sensors and gateways). The three layers comprise sensors at layer one, the gateways (fog nodes) at layer two and the cloud servers at layer three. The proposed architecture allows data sharing amongst predefined neighbourhood sensors at layer one, enabling the implementation of services such as detection and localisation of leakages at that layer as in our earlier works [28,29]. We also extend data placement to the fog layer to facilitate data processing, prioritisation, aggregation, replication, and migration services. Lastly, the cloud layer stores historical data for long-term services like predictive maintenance.

The sensor nodes produce various types of data made available through publication to which services can subscribe. As shown in Fig. 1, the data and services are divided into two sub-layers. The publish-subscribe paradigm is used as the sub-layer interaction model to improve efficiency further. The communication aspect is the same as in our previous works [28,29].

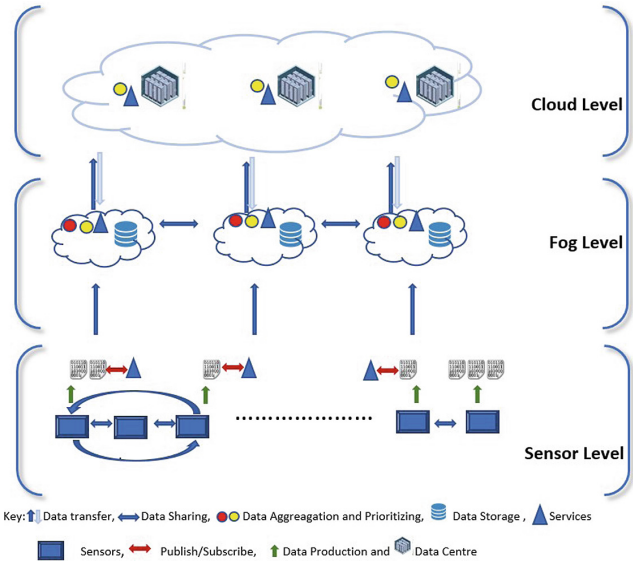


**Fig. 1.** Data and service placement

The following subsections elaborate on the problem definition and its modelling.

### 3.1 Background and Problem Definition

Failures in pipeline transportation of crude oil include erosion, corrosion, equipment failure, vandalisation, network failure, and others. Several pipeline monitoring systems focus on accurately detecting the leakages in the pipelines. However, the monitoring systems are also susceptible to third-party interference, which is one of the leading causes of failure in this mode of transporting crude oil. Consequently, this work aims to circumvent this problem through *efficient* data and service management for the *continuous detection* of leakages in the presence of failures in pipelines and deployed monitoring systems.

Our work is focused on a crude oil pipeline in Nigeria, specifically the NNPC pipeline network shown in Fig. 2. This pipeline network spans several thousand kilometres and is divided into five areas, i.e. PortHarcourt, Kaduna, Warri, Gombe and Mosimi, based on their geographical locations. Different kinds of failures characterise each area, each differing from another significantly. The factors contributing to this vast difference range from weather conditions to the pipeline area's proximity to the border; for instance, the Mosimi and PortHarcourt areas located close to the border experience the highest failure rates. A snippet of historical data, as shown in Fig. 3, presents the incident rate depicting the disparity in the number of incidents in each area.

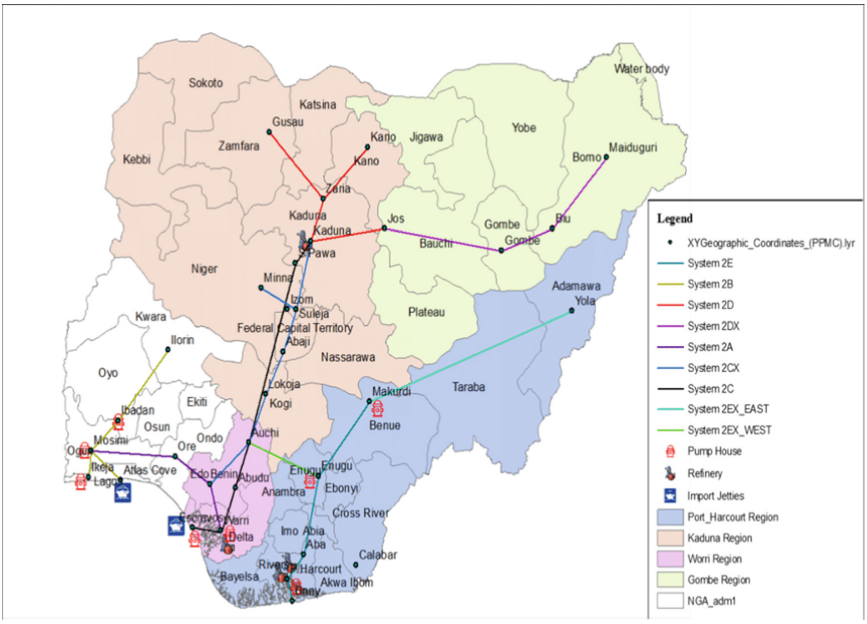


**Fig. 2.** NNPC pipeline network [30]

Thus, to model this problem, let us consider the two principal components we have discussed. On one hand, we have various *failure-causing elements* in pipeline transportation of crude oil. On the other hand, we have several *monitoring techniques* proposed as solutions to this problem.

Considering this information, we could model the problem as a non-cooperative two-player game with player 1 representing the failure-causing components and player 2, the monitoring system. Since our goal is to provide a fault-tolerant monitoring system, we could also apply the maximin strategy for player 2 (the monitoring system). The maximin strategy of a player is that strategy that maximises the player's worst-case payoff, i.e. the minimum amount of payoff guaranteed or the security level of the player [32]. For example, let us denote a player's security level as $\underline{Z_i}$; then, for player 2, the security level is defined in the following equation:

$$\underline{Z_2} = \max_{A_2} \min_{A_1} r_2(A_2, A_1) \tag{1}$$

Equation 1 above is used to find the policy that maximises player 2's security level through action(s)-$A_2$- and minimises the effect of action(s)-$A_1$- taken by player 1, i.e. the saddle point of the two players. With this approach, although pessimistic, we will guarantee leakage detection, whatever the failure is (i.e. the other player will not change its strategy based on the theory of Nash equilibrium).

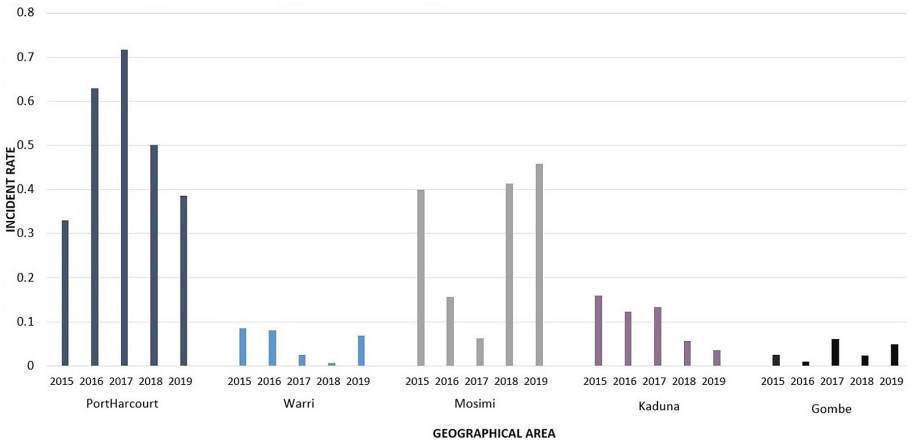


**Fig. 3.** 5 year pipeline incidents [31]

However, to consider our problem a two-player game, we must define the actions set $A$ and utilities $r$ of at least one player. While we can easily quantify the utility in the case of player 2, for example, as the number of leaks detected in ratio to the total number of leaks- over a predefined period, it does not translate to the same utility in the case of the other player. For player 1, most failure-causing components like corrosion and erosion result from natural occurrences. Thus, their effects cannot be considered utility for the player. Moreover, we are more interested in the impact of player 1's action(s) on player 2 than its gain for itself, if any. Thus, making it difficult to be modelled as a 2-players game.

However, we have an environment (NNPC pipeline network presented in Fig. 2) with data that gives us information on the properties and the behaviour of player 1. Hence, we model this problem as a game against nature, i.e. a one-player game. In the subsequent subsection, we discuss, in more detail, this modelling.

### 3.2 Environment Setting and Model Definition

Modelling the problem as a game against nature allows us to optimise the utilities defined for our player (the monitoring system) solely in response to the failures in our environment. While this can be applied as a global solution, the data presented in Fig. 3 shows a large diversity in incident rates from one area to the other. Thus, we propose a tailored solution for each area by broadly categorising them into different *logical regions* defined as $R = \{r0, r1, r2\}$. The aim is to optimise the performance of each region without the cost of a globalised solution.

Using an empirical study, we define a failure-rate-based threshold on which the region of the area is determined. The area(s) with an incidents rate (IR) of 0–5% is categorised as region $r0$ and represents the area(s) with the least number of incidents. The area(s) with an average number of incidents from 6%

to 20% is in region $r1$. The area(s) with IR of 21% and above is placed in the critical region $r2$. Putting the areas into small, average and high ($r0, r1$, and $r2$ respectively) incident rates logical regions allows for practical implementation of data-driven strategies tailored to each region.

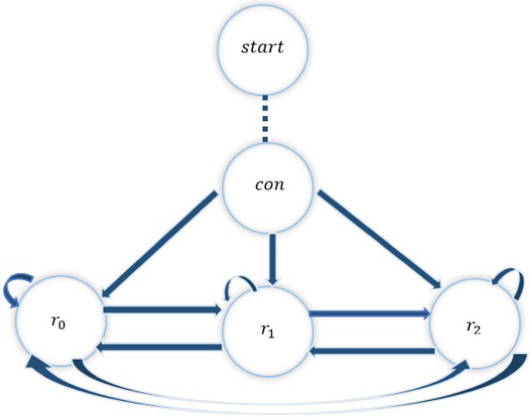


**Fig. 4.** State transition diagram

In addition, each area has a high probability of remaining in the same region for an extended period. For example, the PortHarcourt area is consistently in $r2$ while the Gombe and Kaduna areas are in the region $r0$ and $r1$ 80% of the time, only changing in the year 2017 and 2019, respectively. Mosimi and Warri represent areas with the most likely changes of approximately 40% of the timeline. Accordingly, the probability of each area transitioning from one region to another is low. Therefore, we depict such transition in a broad representation of each area's states as shown in Fig. 4. Each state will be discussed in more detail in the next subsection.

Following the definition of the regions, we aim to provide efficient and fault-tolerant data and service management using a *two-stage* approach. The first stage is to find the best detection policies (a learning agent behaviour) in terms of convergence and optimal detection accuracy using a data-based Markov Decision Process (MDP). We build the second stage on the results of the first stage to find the policy that minimises the processes' energy consumption. Thus, we propose our objective function summarised as follows:

$$\min_{(\pi^*, V^*)} V_e(\pi^*, V^*) \tag{2}$$

where $\pi^*$ is the optimal policy, $V^*$ is a value function following the policy $\pi^*$.

The value function obtained in each region allows us to measure and compare how good (reward and energy consumption) an applied policy is in that region. The performance measure is constrained by the number of nodes (at least two

for fault tolerance in the monitoring system) that provide the accuracy within a bound, to be defined later. In the following subsections, we present the details of the two stages of the objective function.

### 3.3 The First Stage: Accuracy Optimisation

As discussed in Subsect. 3.2, we model our work as an MDP- a reinforcement learning formalisation in machine learning. Unlike other machine learning techniques, i.e. supervised learning (training a set of labelled data) and unsupervised learning (discovering structure in hidden collections of unlabelled data), reinforcement learning is used to learn dynamically through continuous *feedback* from its *interaction* in an environment to arrive at a *goal* [33].

Formalising the decision process in a stochastic environment using MDP is done by the tuple $< S, A, p, r >$, where $S$ represents the sets of states, $A$ the available actions, $p$ the probability transition function, and $r$ the reward function [32]. Another essential variable is the discount factor $\beta$ for emphasising the importance of future rewards. Given these variables, an optimal policy that maximises a model's expected discounted cumulative reward can be determined using the well-known Bellman optimality equation. This cumulative reward helps us measure the goodness of a state, i.e. the maximum attainable reward in a state.

Generally, the Bellman equation following *an arbitrary policy* $\pi$ is defined as follows:

$$\forall s : V^{\pi}(s) = \sum_{s'} p(s'|\pi(s), s)[(r(s, \pi(s)) + \beta V^{\pi}(s'))] \tag{3}$$

where $V^{\pi}(s)$ is the value in state $s$ based on a policy $\pi$, $p(s'|\pi(s), s)$ is the probability of going from state $s$ to $s'$ with action following the policy, $(r(s, \pi(s))$ equals the immediate reward in state $s$, $\beta$ is the discount factor for future rewards and $V^{\pi}(s')$ is the value in state $s'$.

To maximise the accumulated reward, an agent should follow *an optimal policy* -i.e. the policy that takes action with the maximum reward. This process is represented by the Bellman optimality equation as follows:

$$\forall s : V^{*}(s) = \max_{a} \sum_{s'} p(s'|s, a)[(r(s, a) + \beta V^{*}(s')] \tag{4}$$

Note how the Bellman optimality equation (Eq. 4) differs from the Bellman equation (Eq. 3). In Eq. 4, the action $a$ taken always returns the optimal reward, unlike the former with an arbitrary policy. The first part of Eq. 4 represented by $r(s, a)$ is the immediate reward, while the future reward for the second part is embedded in the iterative part $V^{*}(s')$. The hyperparameter $\beta$ is a vital part of the equation for avoiding infinite cycles and allowing solutions to converge eventually. It also underscores the importance of future rewards, i.e. higher $\beta$ value implies a more critical long-term reward and vice versa.

In the first part of our work, we use Bellman's optimality equation to evaluate different policies and to determine the policies that maximise the reward in terms of accurate leakage detection. Thus, we map the equation to the NNPC environment as follows:

**The States $S$:** We define a state as the conditions based on which the detection and localisation process is applied. So, $S$ is represented as a set $S = \{start, con, r_0, r_1, r_2\}$, where *start* and *con* are the initial states i.e., the state before any failure and the initialisation state, $r_0, r_1, r_2$ denotes the states during failure depending on the failure rate of the geographical area.

**The Actions $A$:** Actions are the mechanism of transitioning from one state to another. In our work, we define the set of actions as $A = \{s_t, c, s_a, r_{ds}, m_{ds}, rm_{ds}\}$. Each element in the set denotes process start, connection initialisation, service activation, replication of data and services, migration of data and services and replication and migration of data and services, respectively.

$s_t$ (start-action): $s_t$, as the name implies, is a switch-on action which symbolises the beginning of the decision process.

$c$ (connection initialisation): this action is used to realise the connectivity between nodes as defined in our previous work [29].

$s_a$ (service activation): the $s_a$ action denotes the activation of services in the nodes. Our work has two possible service deployment modes: service predeployment and optimised/dynamic service deployment. In the case of the former, the $s_a$ action is used to reduce energy consumption by intermittently activating services as they are needed. It is, thus, applicable alongside other actions.

$r_{ds}$ (data and service replication): $r_{ds}$ denotes the replication of data and services. It represents the creation of copies of data as they are being produced or services in nearby nodes. In addition, it incorporates replication to the fog nodes instead of the limitation to sensor nodes in our earlier work. This extension to the fog nodes allows the implementation of other services, such as alarms, data prioritisation and filtration in the monitoring system. The amount of replication might change over time relative to changes in the experienced failure in the region.

$m_{ds}$ (data and service migration): involves the migration of data and services. Although it is quite similar to $r_{ds}$, whereas $r_{ds}$ is taken to counter failures, $m_{ds}$ in addition to that is also used for specific needs. For example, we use this action to reassign services to other nodes in case of full storage. Increased latency between a service and its required data is another reason we migrate a service.

Both $r_{ds}$ and $m_{ds}$ have several, but different, communication requirements, as discussed in the following subsection.

$rm_{ds}$ (data and service and replication/migration): this action allows the combination of replication and migration in a region where the failure rate is exceptionally high.

**The States Transition Function $p$:** The state transition function $p$ utilises the information from the data presented in Fig. 3. It shows that some geographical areas are likely to transition from one region to another. Based on that, we determine the transition probability of each geographical area within the logical

regions. In addition, Fig. 3 allows us to make a trend analysis on the failure pattern from one year to another and to specify the value and confidence level for future rewards using the hyperparameter $\beta$.

**The Reward Function** $r$**:** We define the reward function as $r \in [0, 100]$ per node for detection and localisation accuracy of leakages. For every action taken, the reward attainable is equivalent to the accuracy of leakage localisation. We also consider the number of nodes that falls within the allowance threshold for an acceptable accuracy level to represent the aspect of fault tolerance, as we will see later in the results section.

### 3.4 The Second Stage: Optimising the Energy Consumption

In the second stage of our work, the policy that minimises energy consumption for each region is selected based on the results obtained in the first stage. To determine the energy consumption of the policies, we first represent the environment described in Subsect. 3.1 as follows.

Let us define a set of nodes $N = \{n_1, n_2, ...., n_u\}$, where $u$ equals the total number of sensor nodes and gateways. Each node $n_i \in N$ produces a set of data it also stores and is used by the services deployed in the system. If $G$ is a matrix for the use of data by each service, then $g_{ij} = 1$ represents service in node $n_i$ requiring data produced by node $n_j$. Hence, we propose $h_{ij}$ to equate the number of hops between these nodes.

The modes of communication between sensor nodes and gateways is represented by $l_{ij} \in \{0, 1, 2\}$ where $l_{ij} = 0$ means no communication, $l_{ij} = 1$ a connection via LoRaWAN and $l_{ij} = 2$ a connection via 3G/4G communication networks. Each communication link varies in its capacity and will influence the kind of actions (or communication) between the nodes.

The service activation is a direct action depending only on the predefined neighbourhood connection, i.e. the implementation of state *con*.

To replicate, the communication between originating and destination nodes must be $l = 1$ for both data and services.

Migration of services necessitates higher bandwidth than data migration; therefore, their communication requirement is set as $l \in \{1, 2\}$ for data and service, respectively. For each action, the energy consumption is calculated using the following relation:

$$cost_{ij} = lc * h_{ij} \tag{5}$$

where $lc$ is the cost per packet, and per link of the service or data, $h_{ij}$ is the number of hops between the origin and destination nodes. Note that service deployment mode results in changes to cost, i.e., the cost of service migrations is not included in the case of pre-deployed services. In such a scenario, the service activation action is used in place of the service migration action.

A good service placement is on nodes at most one hop away from the leakage point. Therefore, we can minimise energy consumption by reducing the distance

(in the number of hops) between services and the needed data to run the service. We define the objective of the second stage as follows:

$$\min_{(\pi^*,V^*)} V_e(\pi^*, V^*) = \min_{(\pi^*,V^*)} [\sum_{i=1}^{u}\sum_{j=1}^{u} cost_{ij}. \sum_{s'} \phi^{\pi}(s') \sum_{s} g_{ij}(s'|\pi(s), s).V^*(s')] \quad (6)$$

where $\phi^{\pi}(s') = p(s'|s, \pi)$ is the steady state defined by the probability of moving to the next state following a policy ($\pi$) in the current state, $cost_{ij}$ is the energy consumed defined in Eq. 5, $g_{ij}(s'|\pi(s), s)$ represents if data/service is resident in different nodes when we follow a policy from one state to another. This equation presents our objective function, i.e. to find and ensure convergence to the optimal reward with minimum energy consumption through the various placement strategies in the network nodes.

In the following section, we discuss the implementation of the objective function and the results.

## 4 Implementation and Results

The implementation of the model was carried out using the Gym library, an open-source toolkit for developing and approximating reinforcement learning algorithms. Furthermore, we simulated the pipeline environment shown in Fig. 2 using the NS3 network simulator. However, the realisation of the communication between OpenAI Gym and NS3 is based on the work conducted in [34] named ns3-gym. The ns3-gym enables seamless interaction between the NS3 network simulator and OpenAI Gym framework using an environment proxy in the Gym environment and an instantiated gateway in the network.

Figure 5 illustrates our implementation of the interface between the simulated pipeline network and OpenAI Gym in NS3. For comparative analysis, we used multiple agents on the gym side. Each agent interacts with a gateway through the dedicated proxy to implement the region placement strategies while carrying out the reinforcement learning using Algorithm 1.

We present the simulation parameters in Table 1. Other information exchanged between an agent and a gateway in each episode are (*i*) observation space, (*ii*) the action space, (*iii*) the reward and (*iv*) the game over conditions, defined in our work as follows:

**The Observation Space:** Failures, i.e. leakages, network or communication failures.

**The Action Space:** Actions include *start*, *connection*, *service activation*, *data and service replication* and *data and service migration.*

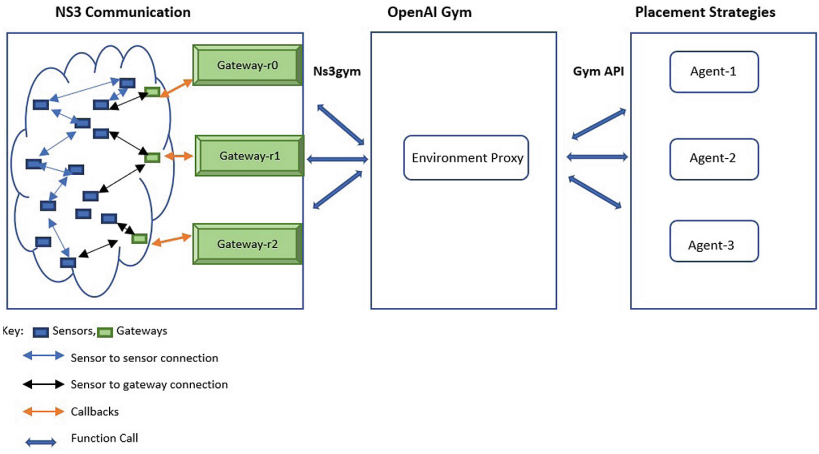


**Fig. 5.** Simulated pipeline network and OpenAI Gym interaction

**Reward:** The reward is based on the localisation accuracy of the leakage and the number of nodes that localises the leakages.

**Game Over:** A game is considered over at the end of the episodes.

**Table 1.** Simulation Parameters

| | |
|---|---|
| Timestep | 1000 |
| Areas | 5 |
| Region | 3 |
| Proxy | 1 |
| Alpha | 1 |
| Beta | 0.99 |
| Number of episodes | 1000 |
| Reward per node | [0,100] |
| Error Type | Rate error model, list error model |
| epsilon | [0, 1] |

The Q-learning algorithm is a well-known reinforcement learning algorithm for solving the Bellman Optimality Equation for MDPs by estimating the action-value function. It enables an early convergence to the optimal action-value function. Given the nature of the problem we are solving, we used the *epsilon*-greedy version of the Q-learning algorithm shown in Algorithm 1 and extracted from [33] to ensure a well-balanced exploration and exploitative steps.

Whereas the convergence to a global optimal value function is assured, the incident rate across the pipeline environment differs significantly, increasing the

**Algorithm 1.** $\epsilon-$greedy Q-learning

Reset Environment
$\epsilon \in (0, 1]$, $\alpha = 1$, $\beta = 0.99$
Initialise Q(s,a), E(s,a) for all s $\in S+$, a $\in$ A(s) arbitrarily except Q(terminal,.)=0

**for each timestep in each episode do**
  Initialise s
  **for each region do**
    Choose a from S using policy ($\epsilon-$greedy) derived from Q
    Perform action a
    Observe r, s'
    $Q(s,a) \leftarrow Q(s,a) + \alpha[R + \beta max_a Q(s',a) - Q(s,a)]$
    s $\leftarrow$ s'
    Get corresponding energy consumption using 5 {Update $E$}
  **end for**
  Until s is terminal
**end for**
Reset Environment
For each region, choose the policy with the least energy consumption and optimal reward

overall cost, i.e. the energy consumption. Thus, we regionalised the environment to solve the MDP and determine the policy that minimises global energy consumption without affecting the optimal reward. In Algorithm 1, we begin by defining the region of each area in the NNPC pipeline network. We set the corresponding failure rate for each region for both failure types (rate and list error types). The state-action values are stored using a matrix - $Q$; then, we find the best policy that minimises the energy consumption by keeping track of the corresponding energy consumption using another matrix - $E$. We chose the optimal policy for each region at the end of all episodes.

In the subsequent subsections, we discuss the result obtained from the simulations.

### 4.1 Accuracy in Detection and Fault Tolerance

The *fault tolerance*, *accuracy* in each region (r0, r1 and r2) and the corresponding *obtainable reward* using the detection and localisation algorithm from our previous work [29] is evaluated in this subsection. To achieve this, we used two different types of communication failures, i.e. *Rate error type* and *List error type* in NS3. The Rate error type involves random packet drop, out-of-order delivery and delayed packet. In contrast, the list error type involves a pre-selected list of packet drops. We made a uniformly distributed selection for the list error type from all the packets shared among the nodes. Also, the simulation was carried out across several randomly selected leakage points. Obtained results are presented in Figs. 6, 7, 8, 9, 10 and 11. We limit this evaluation to error rates between the range of 0% to 20% as either detection or localisation beyond this range is impossible.

In addition, we present each evaluation -i.e. the fault tolerance, accuracy, and obtainable reward- with two figures each: the first figure represents all possible observations. In contrast, the second figure represents observations from High Performing Nodes (HPN), i.e. nodes with results greater than or equal to the target accuracy level of 90%. The set threshold defines the benchmark based on which we can ensure standardised performance. We choose this level of accuracy as the minimum level based on the requirements of the oil and gas operators for high localisation accuracy [35]. It also ensures that all regions have a standardised performance level.

**Fault Tolerance.** One of our detection and localising algorithm's strengths is removing single point of failure associated with centralised systems. To maintain this property, we ensure that there are at least two HPNs across all regions. Figures 6 and 7 show the number of nodes that can detect and localise leakages.

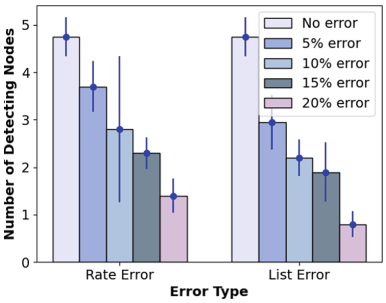


**Fig. 6.** Total number of nodes detecting leakage

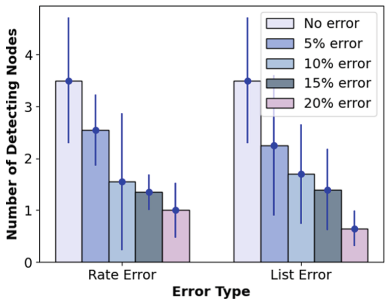


**Fig. 7.** Total number of HPN

In Fig. 6, we capture the total number of nodes detecting and localising leakage without considering the accuracy level, while Fig. 7 shows only the HPN, i.e. with accuracy over 90%. Following the benchmark, we observe (from Fig. 7) that areas with 5% failure rate for both rate and list error types maintain the required number (at least two) of nodes for fault tolerance differing mostly in the variance. This variance is observed from nodes located at the extremities of the pipelines in the area(s), which can be addressed with increased node density in such location(s). However, other areas, i.e. with a failure rate above 5%, do not meet this requirement.

**Accuracy.** We examine how the accuracy level changes based on the failure rate. Results are presented in Figs. 8 and 9.

Figure 8 shows that the list error type provides a higher accuracy level than the rate error type, with both presenting similar trends in error. Nonetheless, when we consider results from HPN shown in Fig. 9, the rate error type presents


**Fig. 8.** Accuracy of leakage detection in all nodes


**Fig. 9.** Accuracy of leakage detection in HPN

much variance in changes in the accuracy of localisation compared to the list error type. Hence, we can conclude that while the lower error rates favour a higher number of nodes in leakage detection, it does not have such an impact on the accuracy of localisation.

**Rewards.** The reward function is defined based on the minimal accuracy requirement and fault-tolerance results from the two previous subsections. The total reward is the sum of the level of accuracy obtained by each detecting node in the defined region. This is represented in Figs. 10, and 11 also separated as all nodes and high-performance nodes. In both cases, we observe a similar pattern of decrements with increasing failure rate.


**Fig. 10.** Total obtainable reward across the error rates


**Fig. 11.** Total obtainable reward with HPN

Following the benchmarks discussed in Subsect. 4.1, we present the expected reward across all regions in Fig. 11. We expect to have two HPNs at all times for all failure rates, thus putting the target at the level shown in the figure.

### 4.2 Optimised Accuracy and Energy Consumption

This subsection discusses the rewards obtained based on our implemented algorithms (learning-based, heuristic and pessimistic).

We used a uniform-random policy and our proposed regionalised $\epsilon$-greedy Q-learning (R-MDP) for the learning-based algorithms. The uniform-random policy has a uniform distribution over the space from which actions are taken. Given the objective function, the $\epsilon$-greedy Q-learning algorithm lets the agent exploit (dependent on what the agent has learnt) and explore (take a random action with a probability of $\epsilon$), thus evading local optimum in each region.

Also implemented is a heuristic approach based on weighted set cover service placement (WSC) from the work [36] to provide a baseline for comparison. The goal of the WSC in our context is also to maximise detection accuracy with minimum cost (i.e. energy consumption) in a globalised manner. The pessimistic approach deals with the worst-case scenario from the environmental data. Following is the performance evaluation of each algorithm.

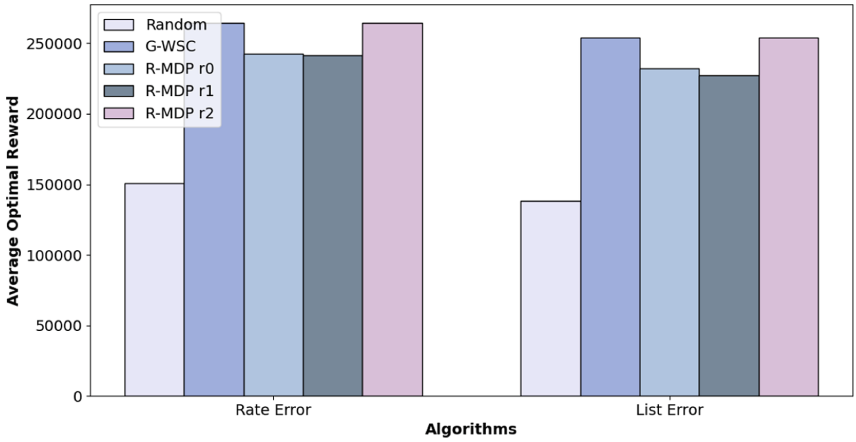


**Fig. 12.** Total reward by algorithm

Figure 12 reveals the optimal rewards following the three implemented algorithms. As expected, the random policy has the lowest total reward. Results from the $\epsilon$-greedy algorithm for each region satisfy the optimality conditions defined in the objective function (Eq. 6). The WSC-based algorithm shows about a 6% increase in the obtained total value attributed to the dynamics of the environment.

To study the exploration effect, i.e. the value of $\epsilon$ on the results, we present the total energy consumption in Fig. 13. Results show less energy consumption for the average epsilon values less than 0.4 and much higher consumption as $\epsilon$ approaches the maximum average value of 1. We observe that the exploration's effect differs considerably for each region. For example, regions r0 and r2 perform much better with a near absolutely greedy policy. While the reward of r0 slightly varies as the exploration increases, the reward of r2 is inversely proportional to the epsilon value. However, region r1 presents optimal results with an epsilon

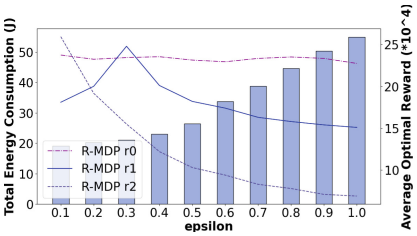


**Fig. 13.** Regional reward vs energy consumption

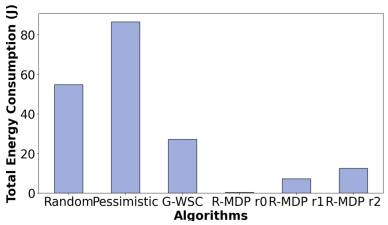


**Fig. 14.** Total energy consumption by algorithm

value of approximately 0.3. Thus, partitioning the environment into distinct regions ensures an optimal solution for each region with minimised total global energy consumption.

Presented in Fig. 14 is the energy consumption by algorithms. For this analysis, we considered the total energy consumed in three different cases, i.e. a regionalised learning approach (R-MDP), a globalised heuristic approach (G-WSC) and a pessimistic approach. From the result, the pessimistic approach has the highest energy consumption, as expected, given that we considered the worst-case scenario. Compared to the total energy (sum of energy consumption in all the regions) by R-MDP, we achieved about a 77% reduction in energy consumption. However, when we correlate R-MDP to G-WSC, the reduction in the energy consumption is about 26%, much lesser than the worst-case scenario, nevertheless, a significant improvement in the energy consumed by the globalised solution.

## 5 Conclusion and Future Work

This paper presented our work on fault-tolerant and energy-efficient data and service management modelled as an MDP. Our work is based on the NNPC pipeline network and the associated failures. We implemented a regionalised MDP that ensures optimal reward with a detection accuracy threshold above 90% in an energy-efficient manner. Results show that the total energy consumption is minimised by a significant reduction of nearly 77% compared to the pessimistic method and approximately 26% compared to a globalised heuristic technique. In future works, we will consider other forms of failures and constraints, such as latency issues, and dynamic topologies, such as mobile sensors/gateways - e.g. UAVs - for dynamic communication coverage.

**Acknowledgment.** This work was supported by the Petroleum Technology Development Fund (PTDF).

## References

1. Mohammadpoor, M., Torabi, F.: Big data analytics in oil and gas industry: an emerging trend. Petroleum **6**(4), 321–328 (2018)

2. Aliguliyev, R., Imamverdiyev, Y.: Conceptual big data architecture for the oil and gas industry. Probl. Inf. Technol. **08**, 3–13 (2017)
3. Slaughter, A., Bean, G., Mittal, A.: Connected barrels: transforming oil and gas strategies with the Internet of Things. Deloitte Center for Energy Solutions, Technical report (2015)
4. Mittal, A., Slaughter, A., Zonneveld, P.: Bringing the digital revolution to midstream oil and gas. Deloitte Center for Energy Solutions, Technical report (2018)
5. Zonneveld, P., Slaughter, A., Mittal, A.: Protecting the connected barrels cybersecurity for upstream oil and gas. Deloitte Center for Energy Solutions, Technical report (2017)
6. Hajirahimova, M.: Opportunities and challenges big data in oil and gas industry. In: National Supercomputer Forum (NSKF-2015) (2015)
7. Song, J., He, H., Wang, Z., Yu, G., Pierson, J.-M.: Modulo based data placement algorithm for energy consumption optimization of mapreduce system. J. Grid Comput. **1**, 1–16 (2016)
8. Wang, W., Wu, G., Guo, Z., Qian, L., Ding, L., Yang, F.: Data scheduling and resource optimization for fog computing architecture in industrial IoT. In: Fahrnberger, G., Gopinathan, S., Parida, L. (eds.) ICDCIT 2019. LNCS, vol. 11319, pp. 141–149. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-05366-6_11
9. Patel, P., Intizar Ali, M., Sheth, A.: On using the intelligent edge for IoT analytics. IEEE Intell. Syst. **32**(5), 64–69 (2017)
10. Tang, B., Chen, Z., Hefferman, G., Wei, T., He, H., Yang, Q.: A hierarchical distributed fog computing architecture for big data analysis in smart cities. In: Proceedings of ASE International Conference on Big Data (2015)
11. Giordano, A., Spezzano, G., Vinci, A.: Smart agents and fog computing for smart city applications. In: Alba, E., Chicano, F., Luque, G. (eds.) Smart-CT 2016. LNCS, vol. 9704, pp. 137–146. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39595-1_14
12. Naas, M.I., Parvedy, P.R., Boukhobza, J., Lemarchand, L.: iFogStor: an IoT data placement strategy for fog infrastructure. In: IEEE 1st International Conference on Fog and Edge Computing (ICFEC), pp. 97–104 (2017)
13. Aral, A., Ovatman, T.: A decentralized replica placement algorithm for edge computing. IEEE Trans. Netw. Serv. Manage. **15**(2), 516–529 (2018)
14. Shao, Y., Li, C., Tang, H.: A data replica placement strategy for IoT workflows in collaborative edge and cloud environments. Comput. Netw. **148**, 11 (2018)
15. Velasquez, K., Abreu, D.P., Curado, M., Monteiro, E.: Service placement for latency reduction in the internet of things. Ann. Telecommun. **72**, 105–115 (2017)
16. Elsayed, W.M., Sabbeh, S.F., Riad, A.M.: A distributed fault tolerance mechanism for self-maintenance of clusters in wireless sensor networks. Arab. J. Sci. Eng. **43**, 6891–6907 (2017)
17. Yuvaraja, M., Sabrigiriraj, M.: Fault detection and recovery scheme for routing and lifetime enhancement in WSN. Wireless Netw. **23**, 267–277 (2015)
18. Qian, S., Mao, W., Cao, J., Le Mouël, F., Li, M.: Adjusting matching algorithm to adapt to workload fluctuations in content-based publish/subscribe systems. In: Proceedings of the IEEE Conference on Computer Communications (INFOCOM 2019), Paris, France, pp. 1936–1944 (2019)
19. Ioana, A., Burlacu, C., Korodi, A.: Approaching OPC UA publish-subscribe in the context of UDP-based multi-channel communication and image transmission. Sensors **21**(4), 1296 (2021)

20. Aslam, A., Curry, E.: Investigating response time and accuracy in online classifier learning for multimedia publish-subscribe systems. Multimedia Tools Appl. **80**, 13021–13057 (2021)
21. Jafarpour, H., Hore, B., Mehrotra, S., Venkatasubramanian, N.: CCD: a distributed publish/subscribe framework for rich content formats. IEEE Trans. Parallel Distrib. Syst. **23**(5), 844–852 (2012)
22. Garg, D., Narendra, N.C., Tesfatsion, S.: Heuristic and reinforcement learning algorithms for dynamic service placement on mobile edge cloud. CoRR (2021)
23. He, T., Khamfroush, H., Wang, S., La Porta, T., Stein, S.: It's hard to share: joint service placement and request scheduling in edge clouds with sharable and non-sharable resources. In: 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS), pp. 365–375 (2018)
24. Cai, Q., Hang, W., Mirhoseini, A., Tucker, G., Wang, J., Wei, W.: Reinforcement learning driven heuristic optimization. CoRR (2019)
25. Islam, M.S., Nix, R., Kantarcioglu, M.: A game theoretic approach for adversarial pipeline monitoring using wireless sensor networks. In: 2012 IEEE 13th International Conference on Information Reuse Integration (IRI), pp. 37–44 (2012)
26. Rezazadeh, A., Talarico, L., Reniers, G., Cozzani, V., Zhang, L.: Applying game theory for securing oil and gas pipelines against terrorism. Reliab. Eng. Syst. Saf. **191**, 106140 (2019)
27. Rezazadeh, A., Zhang, L., Reniers, G., Khakzad, N., Cozzani, V.: Optimal patrol scheduling of hazardous pipelines using game theory. Process Saf. Environ. Prot. **109**, 242–256 (2017)
28. Ahmed, S., Le Mouël, F., Stouls, N.: Resilient IoT-based monitoring system for crude oil pipelines. In: Proceedings of the 7th International Conference on Internet of Things: Systems, Management and Security (IOTSMS). IEEE (2020)
29. Ahmed, S., Le Mouël, F., Stouls, N., Lipeme Kouyi, G.: HyDiLLEch: a WSN-based distributed leak detection and localisation in crude oil pipelines. In: Barolli, L., Woungang, I., Enokido, T. (eds.) AINA 2021. LNNS, vol. 225, pp. 626–637. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-75100-5_54
30. Ambituuni, A., Ochieng, E., Amezaga, J.M.: Optimizing the integrity of safety critical petroleum assets: a project conceptualization approach. IEEE Trans. Eng. Manage. **66**(2), 208–223 (2019)
31. NNPC: 2019 annual statistical bulletin. Nigerian National Petroleum Corporation, Technical report (2019)
32. Shoham, Y., Leyton-Brown, K.: Multiagent Systems Algorithmic, Game-Theoretic, and Logical Foundations. Cambridge University Press, Cambridge (2008)
33. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, 2nd edn, F. Bach, Ed. The MIT Press, Cambridge (2020)
34. Gawlowicz, P., Zubow, A.: ns-3 meets OpenAI Gym: the playground for machine learning in networking research. In: ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM) (2019)
35. Ostapkowicz, P.: Leak detection in liquid transmission pipelines using simplified pressure analysis techniques employing a minimum of standard and non-standard measuring devices. Eng. Struct. **113**, 194–205 (2016)
36. Garg, D., Narendra, N.C., Tesfatsion, S.: Heuristic and reinforcement learning algorithms for dynamic service placement on mobile edge cloud (2021)