Adaptive Storage Optimization Scheme for Blockchain-IIoT Applications Using Deep Reinforcement Learning

Nana Kwadwo Akrasi-Mensah [*] et al.

[*]Distributed IoT Platforms, Privacy and Edge-Intelligence Research (DIPPER) Lab, Faculty of Electrical and Computer Engineering, KNUST.

Background

Blockchain-IIoT integration into industrial processes promises greater security, transparency, and traceability, particularly in collaborative production systems such as food supply chains. In the blockchain, a network of peers maintain a decentralized ledger as each peer stores a full copy of the ledger. The ledger can only be appended to and is updated through the consensus of the peers in the network. Data generation may exceed the peers' capacity to store blocks in the high transaction environment of IIoT networks, which would limit the involvement of devices with limited resources, such as low-power IoT devices. Literature has proposed utilizing cloud storage systems to mitigate the storage pressure on peers in IoT systems. However, this increases latency and impacts the blockchain's performance [1].

Block Selection Problem

The block selection problem is an optimization problem in which the conflicting objectives of query probability of blocks in the cloud, cloud storage cost, and local space occupancy should be minimized while selecting *M*, the ideal set of blocks to transfer to cloud storage, out of *N*, the total number of blocks on the blockchain.

$$\begin{aligned} \min \quad & (O_M, P_M, C_M), \\ \text{s.t.} \quad & 1 \le M \le N. \end{aligned} \tag{1}$$

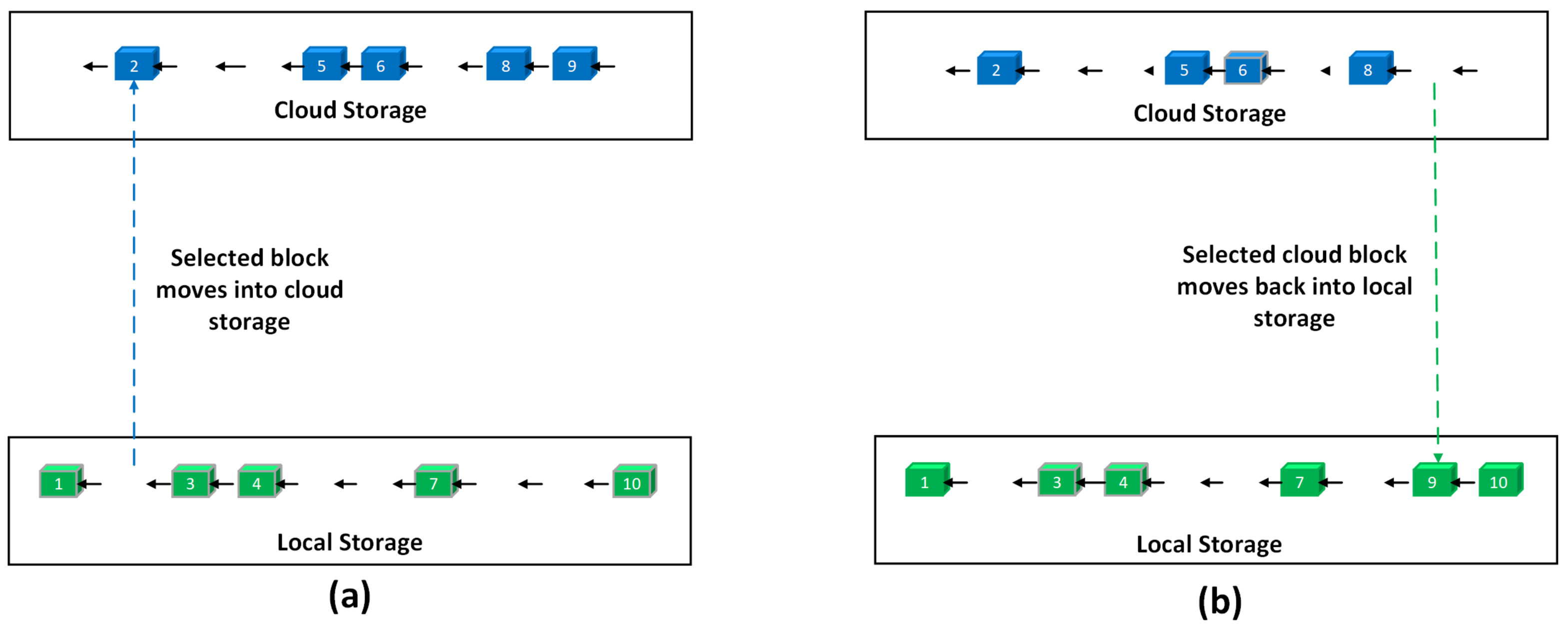

Figure 1. Bi-directional Block Selection: (a) The agent selects a locally stored block and moves it to the cloud. (b) The agent selects a block stored in the cloud and moves it to local storage.

Proposed Approach Using DRL

In this work, DRL is proposed as an alternative to the evolutionary algorithms introduced in [2] and [3] for solving the multi-objective optimization problem of block selection for storage optimization in blockchain-IIoT systems [1]. This approach may offer benefits over evolutionary algorithms and custom heuristics such as **a shorter runtime**, **strong generalization ability** and **scalability** [4].

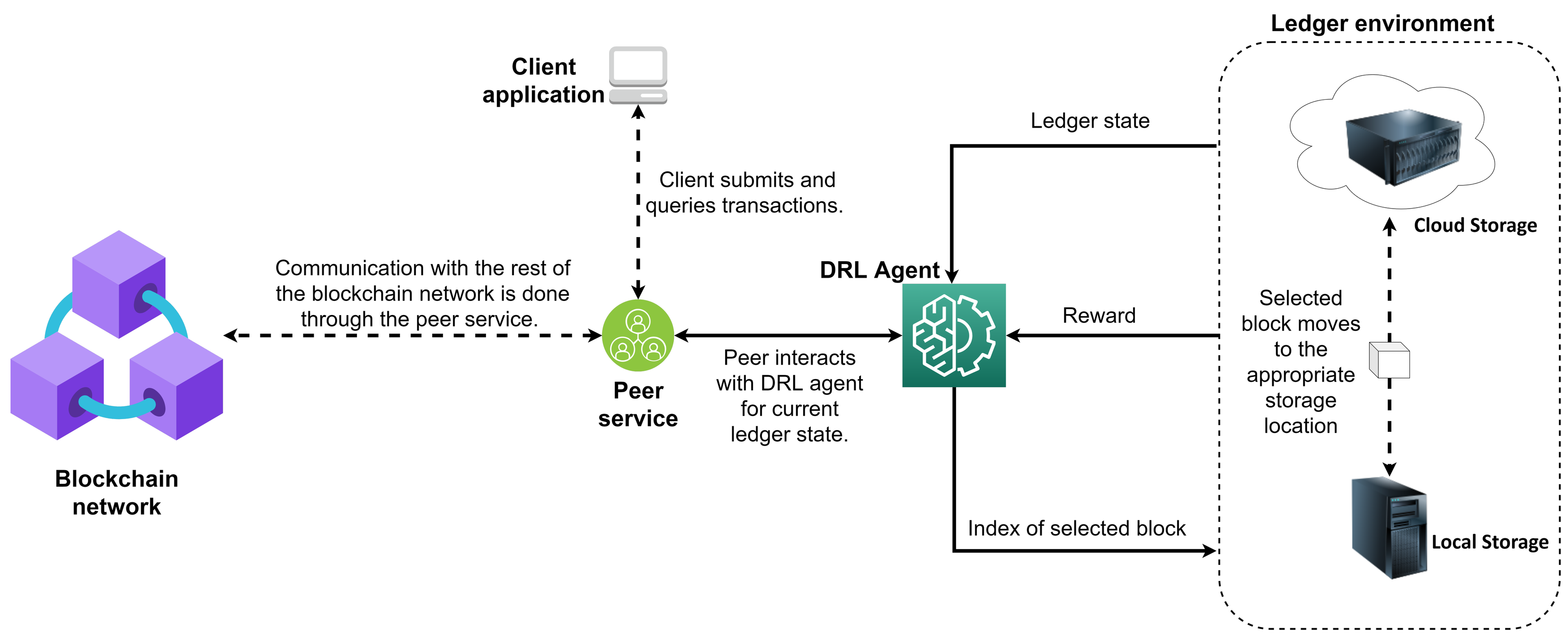

Figure 2. DRL-based Storage Optimization Scheme

MDP Formulation

An MDP can be defined by the 4-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$, which corresponds to the state space, action space, state transition function and reward function.

The system state s, at any given time step t, can be denoted by:

$$s_t = [F_t, s_t, B_t], \tag{2}$$

where $F_t = \{F_{1_t}, \ldots, F_{N_t}\}$, $s_t = \{s_{1_t}, \ldots, s_{N_t}\}$ and $B_t = \{B_{1_t}, \ldots, B_{N_t}\}$. F_t represents the initial query frequency of a block, s_t represents the size of the block and B_t represents the block status.

The agent needs to select an action at every time step. The actions available to the agent is given by the discrete set of block numbers. The action is given by the number of the block to be selected. If no block is selected, the action is represented by zero.

$$a_t = \begin{cases} k, & 1 \le k \le N, \\ 0, & \text{no block selected.} \end{cases} \tag{3}$$

The reward function is given by:

$$R = \alpha(1 - P_M) + \beta(1 - \frac{C_M}{C_{max}}) + \gamma(1 - O_M). \tag{4}$$

α, β and γ represent the objective function weights for denoting the importance of each objective relative to the other objectives and are such that $\alpha + \beta + \gamma = 1$.

$$r_t = \begin{cases} R, & D_U < D_L \quad \textit{and} \quad C_M \le C_L, \\ -R, & D_U \ge D_L \quad \textit{and} \quad C_s = 0, \\ (\frac{D_L - D_U}{D_L}) \times R, & D_U \ge D_L \quad \textit{and} \quad C_s > 0, \\ 0, & C_M > C_L. \end{cases} \tag{5}$$

Results and Conclusion

The simulated environment for training was implemented in Python using OpenAI's Gym. The agents based on the proposed DRL algorithms were also implemented in Python using Stable Baselines3 which is based on PyTorch. The DRL agents used in our approach were first trained over 50,000 time steps before being evaluated.

Parameter	Value
N	200 blocks
N_s	300 blocks
D	300MB
s_{avg}	1MB
M_b	$0.1766 per 100MB
θ	0.1
D_L	150MB
C_L	$0.1766
α	0.5
β	0.2
γ	0.3
F_0	0.95
s_{b_i}	$1\text{KB} < s_{b_i} \le 2\text{MB}$
B_i	(0,1)

Table 1. Optimization Parameters

Algorithm	Run-time
PPO	46.7s
A2C	53.6s
AT-MOPSO	384.2s
NSGA3	474.1s

Table 2. Run-time of DRL Algorithms and Evolutionary Algorithm

On average, the A2C agent received higher rewards than the PPO agent. However the PPO agent showed improvement over the course of the test and received a higher reward at the end of the test compared to the A2C agent.

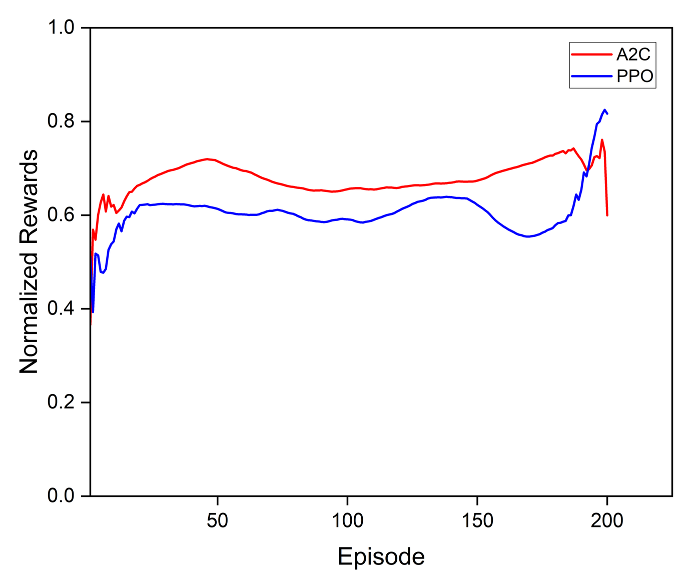

Figure 3. Rewards obtained by trained agents over 200 episodes

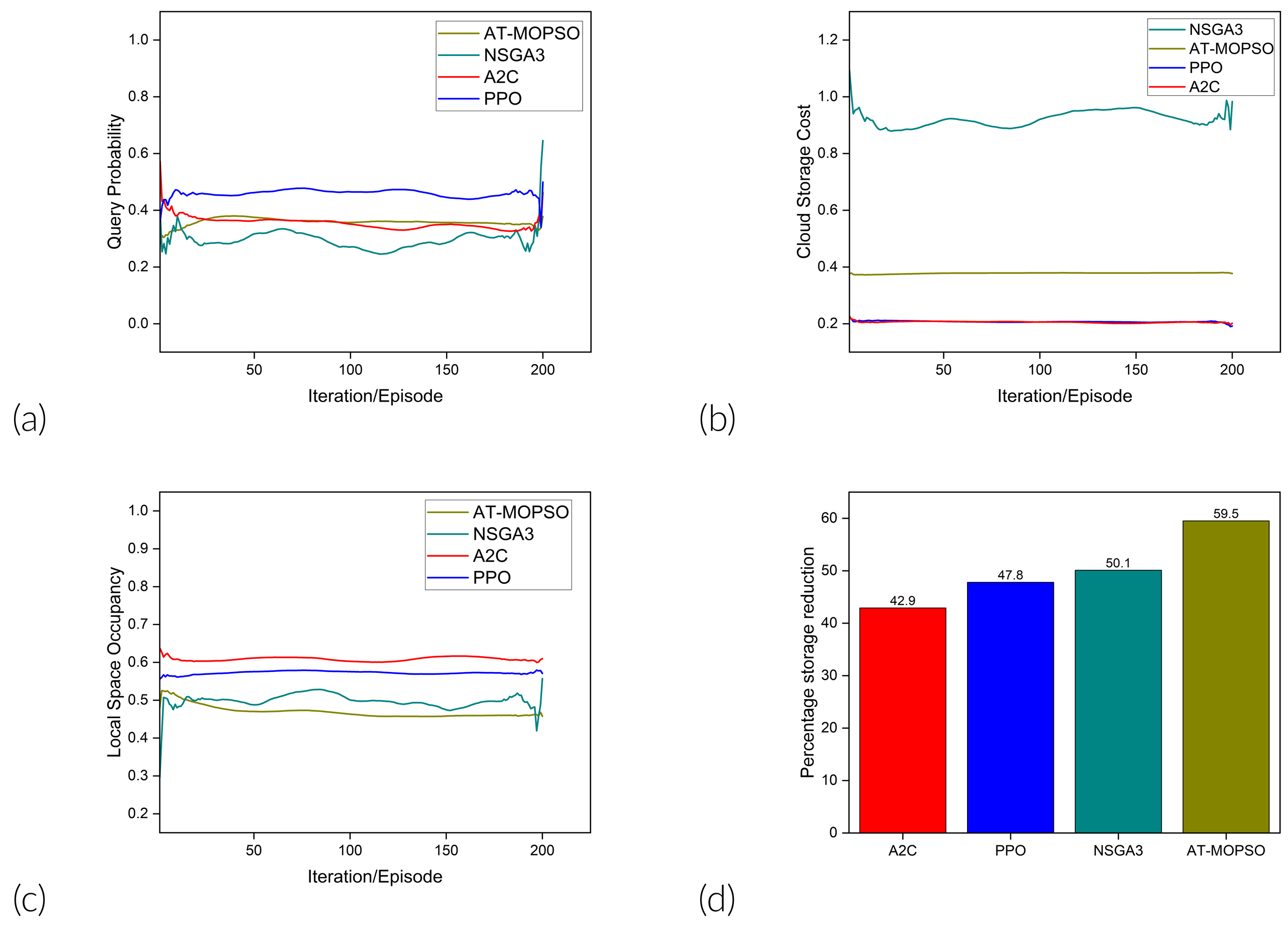

Figure 4. Comparison of algorithms on objective functions and storage reduction

The significant storage reduction achieved and the fast execution time of the trained model makes this approach a better solution for the block selection problem in blockchain-IIoT environments where speed and efficiency are critical. The short execution time also means less energy is consumed in computation.

References

[1] N. K. Akrasi-Mensah, A. S. Agbemenu, H. Nunoo-Mensah, E. T. Tchao, A.-R. Ahmed, E. Keelson, A. Sikora, D. Welte, and J. J. Kponyo, "Adaptive storage optimization scheme for blockchain-iiot applications using deep reinforcement learning," *IEEE Access*, vol. 11, pp. 1372–1385, 2023.

[2] M. Xu, G. Feng, Y. Ren, and X. Zhang, "On Cloud Storage Optimization of Blockchain with a Clustering-Based Genetic Algorithm," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8547–8558, 2020.

[3] C. Nartey, E. T. Tchao, J. D. Gadze, B. Yeboah-Akowuah, H. Nunoo-Mensah, D. Welte, and A. Sikora, "Blockchain-IoT peer device storage optimization using an advanced time-variant multi-objective particle swarm optimization algorithm," *EURASIP Journal on Wireless Communications and Networking*, vol. 2022, no. 1, 2022. [Online]. Available: https://doi.org/10.1186/s13638-021-02074-3

[4] K. Li, T. Zhang, and R. Wang, "Deep Reinforcement Learning for Multiobjective Optimization," *IEEE Transactions on Cybernetics*, vol. 51, no. 6, pp. 3103–3114, 2021.