深層強化学習を使用したエッジコンピューティングにおける動的オフロードメカニズムの改善

鶴丸涼太　　アプドゥハン・ベーナディ
九州産業大学 理工学部 情報科学科

概要

エッジコンピューティングには、1 台のエッジサーバに負荷が集中することで処理の遅延が発生するという欠点が挙げられており、その問題を解決するために計算オフロード技術が提案されている。本研究では、深層強化学習アルゴリズム DDPG を使用したオフロードメカニズムによるエッジサーバの負荷分散とリアルタイム処理を行う計算オフロードの実現を目指した。予備実験ではオフロード先エッジサーバの位置情報と負荷率から、本実験ではオフロード先エッジサーバの位置情報、負荷率、CPU 等の使用率を考慮したオフロード決定を行った。予備実験と本実験共に報酬が増加したことを確認し、最適なオフロードポリシーを学習できたと言える。

Abstract

Edge computing has been cited as having the drawback of processing delays caused by concentrated load on a single edge server, and computational offloading techniques have been proposed to solve this problem. In this study, we aimed to realize computational offloading that distributes the load of edge servers and performs real-time processing through an offloading mechanism using the deep reinforcement learning algorithm DDPG. In the preliminary experiment, the offload decision was based on the location and load ratio of the offload destination edge server, and in the main experiment, the offload decision was based on the location, load ratio, and utilization of CPU and other resources of the offload destination edge server. We confirmed that the rewards increased in both the preliminary and main experiments, and we can say that the optimal offload policy was learned.

1. はじめに

1.1 研究背景

近年、パソコンやスマートフォンなどのインターネット端末に加え、家電や産業ロボット、工場のライン整備など、様々な業界のモノがインターネットに繋がる技術 IoT(Internet of Thing)が活用されている。その影響により、今後さらにこのようなインターネットに繋がるモノ(IoT デバイス)が増加していくと予想されている[1]。

現在、IoT デバイスが発する大量のデータを処理する技術としてクラウドコンピューティングが利用されている。クラウドコンピューティングとは、ユーザがソフトウェアやアプリケーションを必要なときに必要な分だけネットワークで利用することができるサービスである。しかし、IoT デバイスの増加に伴ってデータ通信容量が増大し、通信帯域が圧迫されることで発生する通信の遅延やデータセンターまでの距離が遠いことでリアルタイムな要求への対応が難しいなどの問題がある。

そこで、そのような問題に対処するために注目されているのがエッジコンピューティングという技術である。エッジコンピューティングとは、デバイス付近にエッジサーバを設置し、デバイスが発するデータやタスクをエッジサーバが代わりに処理する技術である。エッジコンピューティングはデバイス付近で処理を行うことができるため、インターネットを経由するクラウドコンピューティングよりもリアルタイムな処理を行うことができ、デバイスとクラウドサーバ間の通信遅延を抑えることができる。また、エッジサーバのみで処理を行うことができるため、ネットワークに負荷をかけることなく、処理結果のみをクラウドサーバへ送信することで通信量を削減することも可能である。しかし、1 台のエッジサーバに負荷が集中することで処理の遅延が発生するという問題が挙げられている。その問題を解決するために、計算オフロードという技術が考えられた。計算オフロードとは、処理の一部を外部のシステムへ転送することで、本体システムの負荷を軽減する技術である。エッジコンピューティングの問題に対して、1 台のエッジサーバの処理を別のエッジサーバへ転送し、負荷を軽減することで問題を解決しようと研究が進められている。

1.2 研究目的

本研究の目的は、エッジサーバの処理遅延解決に向けて、深層強化学習を使用したオフロードメカニズムによるエッジサーバの負荷分散とタスクのリアルタイム処理を行う計算オフロードの実現を目指すことである。深層強化学習を使用したオフロードメカニズムが、最適なオフロード決定を学習し、報酬を増加させるような行動を取るか動作の検証を行う。

Improvement of a Dynamic Offloading Mechanism for Edge Computing using Deep Reinforcement Learning
Ryota Tsurumaru, Bernady O. Apduhan
Faculty of Science and Engineering, Kyushu Sangyo University, Japan

2. 関連技術

2.1 強化学習

強化学習とは、データを用意せずにエージェント(システム本体)が試行錯誤しながら精度を高めていく学習方法である。強化学習のモデルは一般的にエージェントと呼ばれる。エージェントが、環境から与えられる報酬をもとに行動を最適化し、報酬を最大化する一連の意思決定を行う。人間が正解となるデータを定義せず、達成すべきゴールのみを指示として与えることでエージェントが試行錯誤し、最適な行動を探し出す。そのため、人間が介入することや明示的なプログラムを入力することを行う必要がない。エージェントは、環境から得た状態と報酬をもとに方策を更新し、行動を選択する。その一連の流れを繰り返し、環境から得られる報酬をより多く収集するために学習を続ける[2]。そのため、ある行動に対する評価の定義が難しい問題を取り扱う際に有効な手法であると言える。

2.1.1 強化学習モデル

強化学習は、実際に学習を行うエージェントと制御対象となる環境の主に 2 つの要素から成り立っている。強化学習の基本的なモデルを図 1 に示す。強化学習のモデル下において、エージェントが何らかの行動を行うと、その環境下におけるエージェントを取り巻く状態が変化し、その結果が報酬としてエージェントへ与えられる。その報酬を最大化させるためにエージェントの行動を最適化していくことが強化学習の基本的なモデルとなる。また、強化学習では様々なパラメータの調節を通じて学習が進められていく。調節するパラメータは表 1 の通りである。

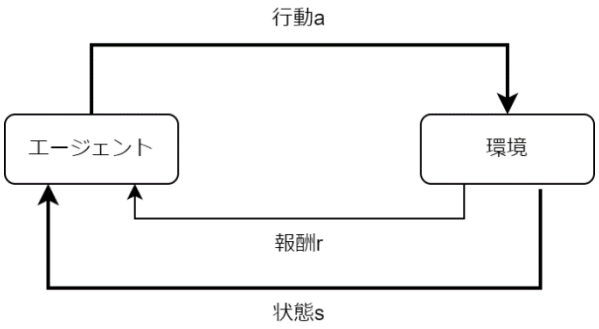

図 1：強化学習の基本的なモデル

表 1：強化学習の基本的なパラメータ

	記号	備考
状態	s	状況を表すパラメータ
行動	a	実際の行動を表すパラメータ
報酬	r	行動の結果としてエージェントに与えられる報酬を表すパラメータ
方策	π	エージェントが行動を選択する際のルール

2.1.2 価値

強化学習では、環境における価値を最大化するようにエージェントを学習させる。価値とは、学習時に得られる即時報酬ではなく、将来的に得られる報酬の推定値を指す。報酬の合計値を最大化するためには、即時報酬が多く得られる近視眼的な行動ではなく、将来的に報酬が多くなるような行動を選択する必要がある。長期的な評価を行うことで、ある行動が将来的に報酬を多く獲得できる価値を持っているのか判断し、報酬の合計値を最大化させることができる。

2.1.3 方策関数と価値関数

方策関数とは、状態 s を入力として行動 a を出力する関数である。一意的な行動を出力する方策を決定的方策、行動の確率分布を出力する方策を確率的方策と言う。

価値関数とは、状態や行動の価値を推定する関数である。価値関数には、状態価値関数と行動価値関数の 2 種類が存在する。状態価値関数は、現在の状態から方策に従って行動を決定した場合に得られる累計報酬の期待値を意味する。行動価値関数は、現在の状態から方策に従って行動を取った場合に得られる累計報酬の期待値を意味する。

価値関数が既にわかっている場合、方策関数 π を求めることが可能である。この手法は方策ベースの学習と呼ばれ、直接最適方策を学習することで期待値を最大化する。一方、価値関数を学習する手法は価値ベースの学習と呼ばれ、最適価値関数を学習し、そこから適当な方策に従って行動を選択する[3]。

2.1.4 Actor-Critic

Actor-Critic とは、TD 誤差学習の手法の一つであり、価値関数と方策関数の両方を学習する特徴を持つ。行動を決める Actor と行動を評価する Critic で構成される。まず、Actor が方策をもとに行動を選択し、実行する。その行動によって得られた状態と報酬を Critic が環境から観測する。観測した状態と報酬をもとに Critic が Actor の行動を評価し、Critic の評価をもとに Actor が方策を更新する。この流れを繰り返していくことで精度を高めていく学習手法である[4]。Actor-Critic には、行動選択に最小限の計算量しか必要ないことと、確率的な行動選択の学習が可能であることの 2 つの利点が存在する。

2.1.5 Q 学習

Q 学習とは、行動価値関数を求める強化学習アルゴリズムの一つである。各状態における行動の行動価値関数を保有する Q テーブルを保持しており、得られた報酬や割引率などを用いて得られる TD 誤差(期待値と見込みの差分)を最小化するように学習し、テーブルの行動価値関数を更新する[5]。また、Q 学習では ε -greedy 法を使用した探索が行われる。

ε -greedy 法とは、ε の確率でランダムな行動を選択し、それ以外の確率で行動価値関数が最大となる行動を選択する手法である。

2.1.6 方策勾配法

方策勾配法とは、方策をあるパラメータで表された関数とし、そのパラメータを学習することで方策を学習する手法である[6]。2.1.5 で述べた Q 学習は、最適な方策を見つけ出すことが難しいため価値関数を最適化する考え方をしているが、方策勾配法では直接最適な方策を見つけ出す考え方をする。方策勾配法は、特に行動の選択肢が多い場合に用いられる。

2.1.7 決定論的方策勾配法（Deterministic Policy Gradient）

決定論的方策勾配法とは、連続行動空間を制御するために考案された、方策勾配を決定論的に更新する強化学習アルゴリズムである。従来の方策勾配法で行われていた確率論的な更新よりも環境を適切に探索することができる。Actor-Critic 手法を用いて行動価値関数と方策関数を学習するが、方策勾配法を使用せずに学習する手法となっている。決定論的方策勾配法も 2.7.6 で述べた Q 学習と同様に、行動価値関数を最大化する手法となっている[7]。

2.2 深層学習

深層学習とは、人間の神経細胞の仕組みを再現したニューラルネットワークを用いた機械学習手法の一つである。多層構造のニューラルネットワークを用いる特徴があり、現在では画像認識や音声認識、翻訳など様々な分野で大きな成果を生み出している[8]。深層学習を行うには教師データを使用する必要がある。手書きの数字を人工知能によって認識させる場合、手書きの数字の画像と正しい数字を記した教師データを用意し、多層構造のニューラルネットワークに入力することで自動的に学習し、手書き数字を認識することができるという仕組みになっている。

2.2.1 深層強化学習

深層強化学習とは、強化学習と深層学習を組み合わせた手法である。強化学習では「ある状態で、ある行動を取ると、どのような報酬が得られるか」を学習することで、最適な行動を選択する。深層強化学習も学習対象は強化学習と同じであるが、行動の選択にニューラルネットワークを用いるため、連続値を扱うことができる。従来の強化学習では、Q テーブルを用いて状態や行動、報酬を管理していたが、テーブルには限界があるため、状態が連続する場合に離散化する必要があった。しかし、離散化は一定幅の状態が同じ扱いとなるため、表現力や精度が落ちてしまう。深層強化学習は連続値を扱うことができるため、表現力や精度を落とすことなく情報を扱えるようになった[9]。また、特徴量の設計をニューラルネットワークが行うため、人手を必要とすることがなくなったなどの利点もある。

2.2.2 Deep Deterministic Policy Gradient（DDPG）

Deep Deterministic Policy Gradient(以下 DDPG)とは、連続行動空間の扱いに対応した深層強化学習アルゴリズムである。決定論的方策勾配法に深層強化学習を適用した手法であるため、決定論的方策勾配法がベースとなっている。一般的な方策を学習する手法では、獲得する報酬の期待値を最大化するのに対し、DDPG では報酬の合計推定値を最大化することが特徴である。行動価値関数が最大の行動を選択するのではなく、行動価値関数が最大となる行動をアクションポリシーが選択し、学習する。DDPG の学習モデルは図 2 の通りである。Actor と Critic は別モデルとなっている。まず、Actor が状態から行動を出力し、Critic が状態と Actor の行動を入力に、Actor の行動を評価して行動の価値を出力する。そして、Critic からの評価をもとに Actor は方策を更新し、行動を選択する[10]。これらの流れを繰り返すことで、より多くの報酬を得ることができる最適な行動を学習する。

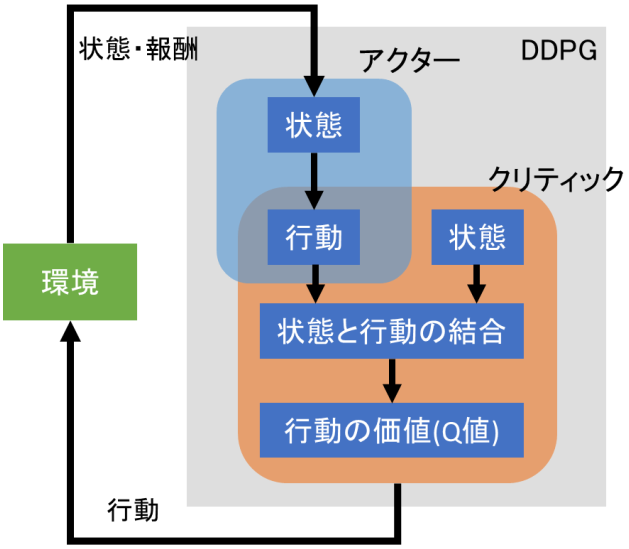

図 2：DDPG の学習モデル

2.2.3 DDPG に用いられる 3 つの手法

DDPG には、Experience Replay や Target-Network、ε -greedy 法といった学習を安定させるための 3 つの手法が用いられている。

(1) Experience Replay

Experience Replay とは、エージェントの学習データを Replay Buffer というメモリに蓄積し、学習する際に蓄積した過去のデータからランダムに選択し、学習する手法。強化学習は、報酬の期待値を更新する際、データ間の相関が大きくなり、学習が進まない問題が発生する場合がある。そのため、Experience Replay を使い、ランダムに選択された過去のデータから様々なエピソードデータを学習する

ことで汎用的な学習を行うことができる。

(2) Target-Network

Target-Network とは、TD 誤差を計算する際に利用されるネットワークである。強化学習では、行動価値関数を直接学習した場合、環境によっては行動価値関数が発散し、学習が不安定になることがある。Target-Network を用いることで、推定する行動価値関数の学習を遅らせ、行動価値関数の発散を低減することができる。Target-Network には、定期的に Q Network の重みを同期する Hard-Target と、Q Network を更新する度に少しずつ重みを近づける Soft-Target の 2 種類存在する。

(3) ε -greedy 法（貪欲法）

強化学習における行動選択の際にランダム性を導入した手法。強化学習は、報酬を多く獲得するために最適な行動を取り続けるが、近似的な行動に偏り、様々な状況に対応できない可能性がある。 ε -greedy 法は、一定の確率でランダムな行動を取ることで、新しい知識の更新を行うことができ、行動の偏りを低減させることができる。

3. 関連研究

井手の参考論文[11]は、エッジコンピューティングにおける負荷分散とリアルタイム処理の両立を実現するため、深層強化学習アルゴリズム DDPG を使用した高性能エッジコンピューティングシステムの開発を目指した。井手の研究では、スマートフォンなどのモバイルと各エッジサーバの位置情報をもとに、モバイルから各エッジサーバへ最適なオフロード先を決定することを想定した研究となっている。本研究では位置情報に加え、CPU・メモリ・ハードディスクの使用率をもとに、エッジサーバから別のエッジサーバへ最適なオフロード先を決定することを試みる。

Yunzhao Li らの参考論文[12]は、エッジサーバの負荷が不均衡となる計算オフロードの問題に対して、深層強化学習アルゴリズム DDPG を使用したシステム開発を行い、最適化された計算オフロード決定を目指した研究である。DDPG アルゴリズムを使用したシステムを Deep Q Network(DQN)アルゴリズムや Asynchronous Advantage Actor-Critic(A3C)アルゴリズムと比較し、DDPG アルゴリズムを使用したシステムの優位性を示すことができた。Yuzhao Li らの研究は、スマートフォンや家電など、移動するデバイスと移動しないデバイスなど様々な IoT デバイスからエッジサーバへタスクをオフロードすることを想定した研究である。本研究では、監視カメラやセンサなどの移動しないデバイスからエッジサーバへオフロードされたタスクを、別のエッジサーバへオフロードすることを想定している。

4. システムモデルと提案手法

4.1 システムモデル

本研究では、図 3 に示すように、エッジサーバ 0 が所持しているタスクを外部にある複数のエッジサーバへオフロードすることを想定する。エッジサーバ 0 が所持しているタスクは、監視カメラやセンサなどの固定デバイスからオフロードされたタスクである。タスクを所持したエッジサーバ 0 と外部にある複数のエッジサーバは別のエッジサーバと考え、エッジサーバ 0 は所持するタスクを外部にある複数のエッジサーバへ計算オフロードすることだけ行い、別のエッジサーバからタスクが送信されてくることはない。外部にある複数のエッジサーバは、エッジサーバ 0 から送られるタスクを受け取ることだけ行い、別のエッジサーバへタスクを送信することはない。

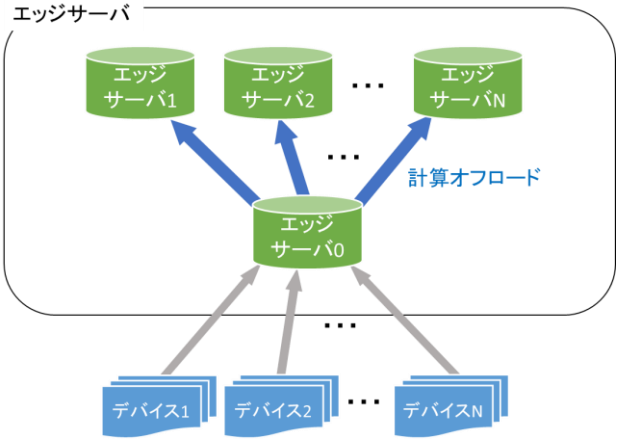

図 3：エッジサーバから別のエッジサーバへの計算オフロードのイメージ

4.2 提案手法

本研究では、タスクを所持したエッジサーバ 0 から外部にある複数のエッジサーバへの効率的な計算オフロードを行うため、Broker Server をエージェントとしたシングルエージェントシナリオを想定した連続的な行動決定を行うタスクオフロードメカニズムを提案する。エッジサーバ 0 は、Broker Server へ最適なオフロード先を要求し、Broker Server から出力された行動に従って外部にある複数のエッジサーバへ計算オフロードを行う。Broker Server と環境のやり取りを図 4 に示す。

エージェントの複雑な意思決定を実現させるため、Actor-Critic モデルに基づく深層強化学習アルゴリズム Deep Deterministic Policy Gradient(DDPG)を採用する。一般的な強化学習アルゴリズムでは、連続値を扱うことができず、常に離散的な行動決定しか行うことができないためである。DDPG アルゴリズムは、連続的な行動空間に対応できるという特徴を持っているため、動的な負荷分散ポリシーを学習することができる。

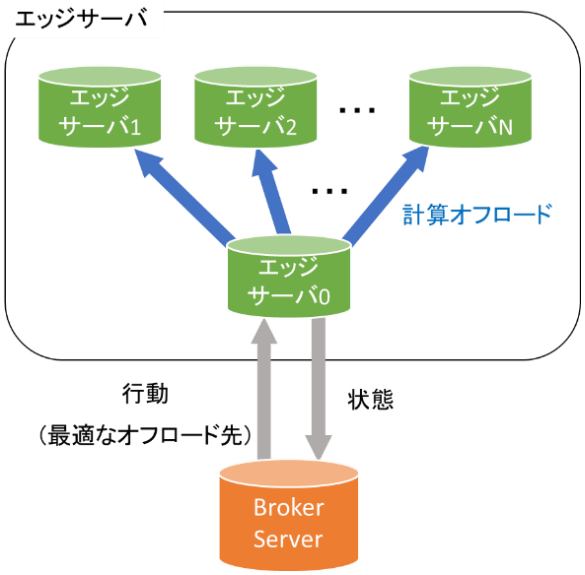

図 4：Broker Server と環境のやり取り

4.2.1 学習の流れ

本手法による学習の流れは図 5 の通りである。2.2.3 で述べた ε -greedy 法と Experience Replay を使用した。 ε -greedy 法によって行動決定にランダム性を持たせ、 ε の確率でランダムに行動することで近似的な行動の偏りを低減する工夫を施した。また、Experience Replay によって Replay Buffer に蓄積されたデータからランダムに学習を行い、汎用的な学習を行うようになっている。キャパシティの数値が一定の値に達した場合、まとめて学習を行い、新たなエピソード学習を始める。

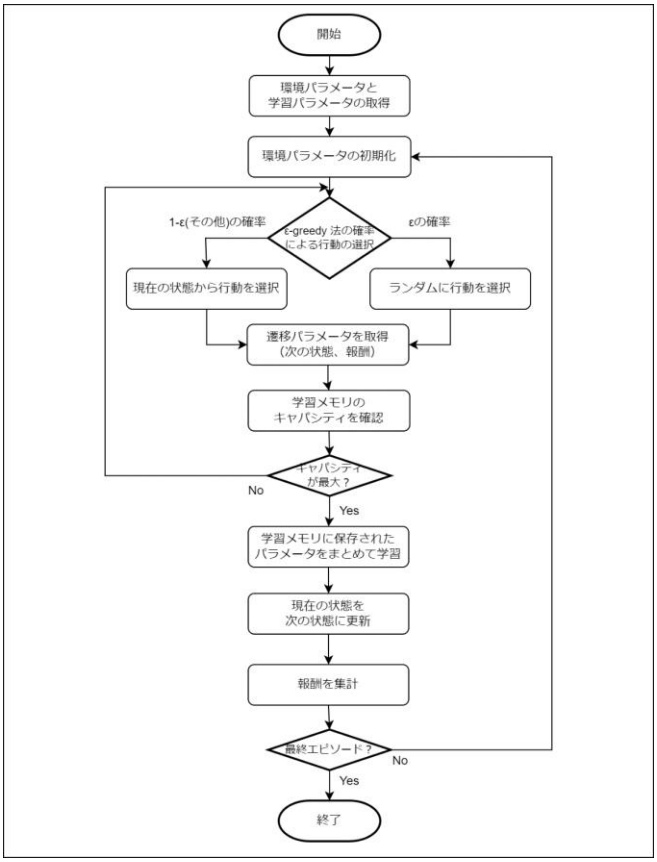

図 5：エージェントによる学習のフローチャート

5. 実験環境

予備実験と本実験に使用したマシン性能を表 2、ソフトウェアについて表 3 に示す。

表 2：使用したマシン性能

OS	Windows10 Home
CPU	Intel(R) Core (TM) i7-9700K CPU@3.60GHz
GPU	NVIDIA GeForce RTX2070 SUPER
メモリ	16.00GB
ストレージ	464GB

表 3：実験に使用したソフトウェア

パッケージ	Anaconda
開発言語	Python(ver3.7.1)
統合開発環境(IDE)	PyCharm Community
ライブラリ	TensorFlow(ver2.3)

表 4：シミュレーションパラメータ

Parameter	Value
学習ステップ	3000steps
データサイズ	50MB
マイグレーション帯域幅	1GB/s
タスク数	90 台
エッジサーバ数	10 台
エッジサーバの最大同時処理数	10 台

表 5：モデルパラメータ

Parameter	Value
レイヤ数	5
活性化関数	ReLu 関数
Actor 学習率	1.0×10^{-4}
Critic 学習率	2.0×10^{-4}
割引率	0.9
Soft Update 更新率	1.0×10^{-2}
Experience Replay バッファサイズ	1.0×10^{4}
バッチサイズ	32

6. 予備実験

6.1 目的

各エッジサーバの位置情報のみを用いて、タスクを所持したエッジサーバからオフロード先のエッジサーバへのオフロード決定を行い、システムが最適なオフロード先決定を行えるか動作を確認する。

6.2 方法

タスクサイズや各エッジサーバの位置情報がランダムに生成された数値データをエージェントに入力し、各エッジサーバの位置情報と負荷率をもとに最適なオフロード先を決定する。また、予備実験では独自に構築した仮想的なエッジコンピューティング環境におけるシミュレーション形式で実施することとする。システム管理者は、環境にデプロイするエッジサーバや生成されるタスクサイズなどを事前にパラメータとして設定することができる。

6.3 結果と考察

予備実験では、各エッジサーバに位置情報を設定し、オフロード先エッジサーバの位置情報と負荷率から最適なオフロード先決定を行えるか、システムの動作確認を行った。オフロードするエッジサーバは、監視カメラやセンサなどの固定デバイスから発生されるタスクを所持し、それを外部にある複数のエッジサーバへオフロードすることを想定した実験となっている。

報酬は step ごとに処理されたタスクが多いほど高くなるように定義した。予備実験の学習によって得られたエピソードごとの報酬を図 6 に示した。横が報酬を多く獲得するために学習したエピソード数、縦が選択した行動によって得られた報酬を表す。エピソードが進むごとに、環境から得られる報酬の値が増加していることが確認できる。このことから、適切なオフロードポリシーを学習できていると考えられる。また、報酬はタスク処理数によって増減するため、報酬が増加している様子からタスクを多く処理できていることがわかる。そのため、エピソードが進むごとに各エッジサーバへタスクを適切に分散できていると考察できる。

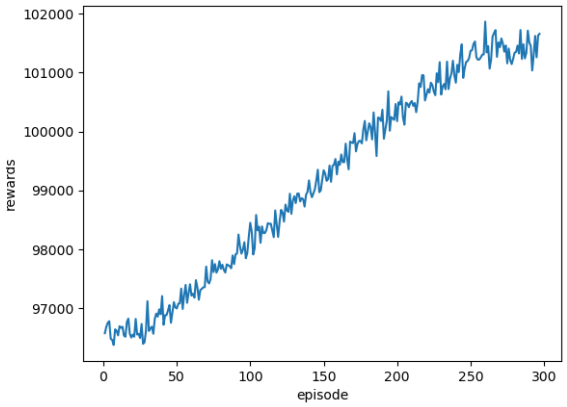

図 6：予備実験におけるエピソードごとの報酬の値

7. 本実験

7.1 目的

本実験では、現実の状況により近い環境を想定し、タスクを所持したエッジサーバからオフロード先のエッジサーバへのオフロード決定を行い、システムが最適なオフロード先決定を行えるか動作を確認する。

7.2 方法

予備実験で使用したタスクサイズやランダムに生成された各エッジサーバの位置情報の数値データに加え、ランダムに生成されたオフロード先エッジサーバの CPU 使用率、メモリ使用率、ハードディスク使用率の数値データをエージェントへ入力し、オフロード先のエッジサーバの CPU 使用率、メモリ使用率、ハードディスク使用率、負荷率、各エッジサーバの位置情報をもとに最適なオフロード先を決定する。予備実験と同様に、独自に構築した仮想的なエッジコンピューティング環境におけるシミュレーション形式で実施し、システム管理者は環境にデプロイするエッジサーバや生成されるタスクサイズなどを事前にパラメータとして設定することとする。

7.3 結果と考察

各エッジサーバの位置情報と負荷率に加え、オフロード先エッジサーバの CPU 使用率、メモリ使用率、ハードディスク使用率を考慮した最適なオフロード先決定を行えるか、システムの動作確認を行った。タスクは、予備実験と同様に監視カメラやセンサなどの固定デバイスから発生されたものを想定している。また、予備実験ではタスク処理数によって報酬の値が増減するように定義していたが、本実験ではそれに加え、CPU 使用率、メモリ使用率、ハードディスク使用率が最小かつ 50%以下であるエッジサーバへオフロードを決定できた場合に追加報酬を与えるように定義した。

本実験の学習によって得られたエピソードごとの報酬の値を図 7 に示す。エピソードが進むごとに報酬が増加していることが確認できる。報酬が約 202,000 の辺りで学習を終了しているが、これは安定して高い報酬を得ることができるようになったとシステムが判断し、学習が完了したからである。また、予備実験では報酬が約 97,000 から約 102,000 で約 5,000 増加していたが、本実験では約 192,000 から約 202,000 で約 10,000 増加しており、予備実験に比べて本実験の報酬の増加率が約 2 倍になっていることが確認できる。このことから、最適なオフロード先を決定できたときの追加報酬が正しく加わっており、オフロード先エッジサーバの CPU 使用率、メモリ使用率、ハードディスク使用率を考慮した学習を正常に行うことができていると考えられる。

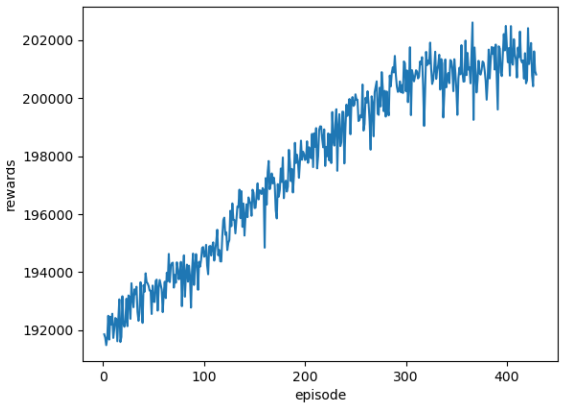

図 7：本実験におけるエピソードごとの報酬の値

8. おわりに

8.1 まとめ

本研究は、深層強化学習アルゴリズム DDPG を使用したオフロードメカニズムによる計算オフロードを行うことでエッジサーバの負荷分散とタスクのリアルタイム処理を実現し、エッジコンピューティングの問題解決を図った。

予備実験では、独自に構築した仮想的なエッジコンピューティング環境にて、各エッジサーバの位置情報をもとにタスクを所持したエッジサーバから別のエッジサーバへのオフロード決定を行い、システムが適切なオフロードポリシーを学習できていることを確認できた。

本実験では、予備実験と同様に独自に構築した仮想的なエッジコンピューティング環境にて、各エッジサーバの位置情報に加え、オフロード先エッジサーバの CPU 使用率、メモリ使用率、ハードディスク使用率をもとにタスクを所持したエッジサーバから別のエッジサーバへのオフロード決定を行った。その結果、システムがオフロード先エッジサーバの CPU 使用率等を考慮した適切なオフロード決定を学習できていることを確認することができた。

8.2 今後の課題

本研究の実験における環境では、タスクサイズが全て同じ数値に設定されている。しかし、実際の環境では様々なタスクが存在し、タスクサイズもそれぞれ異なることが想定される。より実用的なシステムを実現するため、タスクサイズにランダム性を与えた環境での実験を行う必要があると考える。また、本研究の実験で使用した環境はあくまで仮想的なエッジコンピューティング環境であり、実際のエッジサーバなどは使用していない。そのため、実際の環境で本研究のオフロードメカニズムを使用した実験を行い、本研究のオフロードメカニズムが実際の環境においても有効であるか検証する必要があると考える。

謝辞

実験にご協力いただいた井手慎太郎氏に感謝いたします。

参考文献

[1] 鈴木一哉, 森本昌治, 岩井孝法. IoT 技術の最新動向. 電子情報通信学会 通信ソサイエティマガジン. 2018, vol.12, no.1, p. 12-20.

[2] 毬山利貞, 太田佳. 強化学習の基礎と実用化に向けた課題. 映像情報メディア学会誌. 2019, vol.73, no.2, p. 265-270.

[3] 小林泰介. 《第 7 回》機械学習と制御：連続行動空間における強化学習. 計測と制御. 2019, vol.58, no.10, p. 806-810.

[4] 久保晃, 成川航祐, 清水雅樹. 流れの最適制御のための強化学習. 計測と制御. 2020, vol.59, no.8, p. 565-570.

[5] 宮島洋文, 重井徳貴, 宮島廣美, 白鳥則郎. 簡易秘密計算法による安全な連続値 Q-learning の実現. バイオメディカル・ファジィ・システム学会誌. 2018, vol.20, no.2, p. 1-7.

[6] 妹尾卓磨, 今井倫太. Policy Gradients with Memory-Augmented Critic. 人工知能学会論文誌. 2021, vol.36. no.1, p. B-K71_1-8.

[7] 梅本晴弥, 豊田哲也, 大原剛三. 学習の安定化のために方策の埋め込みを利用する強化学習手法の検討. 人工知能学会研究会資料 知能ベースシステム研究会. 2019, vol.118, p. 3.

[8] 神田寛行. Deep learning（深層強化学習）の基礎知識と、医療応用に向けた課題. 視覚の科学. 2018, vol.39, no.3, p. 75-79.
[9] 小林一郎. 深層強化学習 （Deep Reinforcement Learning）. 知能と情報. 2018, vol.30, no.5, p. 252.
[10] 池本隼也, 潮俊光. 深層ニューラルネットワークを利用した強化学習の制御への応用. 日本神経回路学会誌. 2019, vol.26, no.4, p. 135-144.
[11] 井手慎太郎. 深層強化学習を使用したエッジコンピューティングのための動的オフロードメカニズムの開発, 修士論文, 九州産業大学大学院・情報科学研究科, 2022 年 1 月.
[12] Yunzhao Li, Feng Qi, Zhili Wang, Xiuming Yu, Sujie Shao. Distributed Edge Computing Offloading Algorithm Based on Deep Reinforcement Learning, IEEE Access, 2020, vol.8, pp. 85204-85215.