社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

リソース分離型マイクロデータセンターのネットワーク評価のための資源間通信シミュレータの設計

生駒　昭繁† 　大下　裕一† 　村田　正幸†

† 大阪大学 大学院情報科学研究科　〒565–0871 大阪府吹田市山田丘 1–5
E-mail: †{a-ikoma,y-ohsita,murata}@ist.osaka-u.ac.jp

あらまし　近年、エッジサービスの広がりにより、ユーザにより近いエッジに小規模のデータセンターであるマイクロデータセンターを配置し、サービスを提供することが提案されている。しかし、マイクロデータセンターは、大規模なデータセンターと比べて保有しているリソースの量は限られているため、無駄のない資源の利用が重要となる。そこで、資源を独立させ、ネットワークによって資源間を相互に接続することで構成するリソース分離型マイクロデータセンター（μDDC）が提案されている。μDDC において、資源同士はネットワークによって接続されるため、アプリケーション実行時に資源間の通信遅延が発生し、アプリケーションの実行性能が低下するという問題がある。そのため、資源やその間の経路の割り当て方の検討やそれに基づいた μDDC に向けたネットワークの構成が必要である。これらの検討のためには、μDDC の各資源間の通信が実行アプリケーションに対してどれだけ影響を持つのかを正確に把握する必要があり、その計測のための、資源間の通信シミュレータが必要となる。また、μDDC において、資源間の通信は CPU のような計算資源がメモリからデータを取得する際に発生し、その発生タイミングや頻度は実行処理や資源に依存するため、各資源の動作を考慮したシミュレータでなければならない。本稿において、μDDC ネットワークの評価のための計算資源の動作も考慮したシミュレータの設計について示し、その動作を実証する。
キーワード　マイクロデータセンター、リソース分離、シミュレータ

Simulator of communication between resources for evaluation of network structure in a disaggregated micro data center

Akishige IKOMA†, Yuichi OHSITA†, and Masayuki MURATA†

† Graduate School of Information Science and Technology, Osaka University
Yamadaoka 1–5, Suita, Osaka, 565–0871 Japan
E-mail: †{a-ikoma,y-ohsita,murata}@ist.osaka-u.ac.jp

Abstract　In recent years, with the spread of edge services, it has been proposed to deploy micro data centers, which are small data centers, at the edge closer to the user to provide services. However, micro data centers have a limited amount of resources compared to large-scale data centers. Therefore, it is important to efficiently use resources. Disaggregated micro data center (μDDC) has been proposed, which consists of independent resources and interconnects the resources by a network. In a μDDC, because resources are connected to each other by a network, application execution performance degrade due to communication delays. It is necessary to consider how to allocate resources and routes between them, and to configure the network for a μDDC. Therefore, it is necessary to accurately understand how much the communication between resources in the μDDC affects the execution of applications. For this evaluation, a communication simulator between resources that occur in the network is needed. In μDDC, communication between resources occurs when computing resources such as CPUs acquire data from memory, and the timing and frequency of such communication depends on the execution process and resources, so a simulator that takes into account the behavior of each resource is required. In this research, we present the design of a simulator that also considers the behavior of computing resources for the evaluation of μDDC networks, and demonstrate its operation.
Key words　Micro data center, Resource Disaggregation, Simulator

1. はじめに

近年、多種多様な情報サービスがクラウドを用いて提供されている。しかしながら、多くのクラウドサービスは、遠方のデータセンターとユーザが相互に通信しながら処理を行う必要があるため、自動運転のようなリアルタイム性が求められる処理には不向きである。この問題の解決のために、ユーザにより近いエッジに小規模のデータセンターであるマイクロデータセンター (μDC) を配置し、サービスを提供する、エッジサービスの提供が検討されている。ただし、μDC は、大規模なデータセンターと比較すると、CPU や GPU、メモリのような計算資源が少ない。そのため、μDC において多種多様なサービスを提供するためには、効率的な資源の利用が必要となる。

資源の効率的な利用を実現する方法の一つとして、CPU やメモリ等の資源を独立させ、それらをネットワークでつなぐことによってデータセンターを構成する、リソース分離があげられる。リソース分離において各資源は独立しているため、サービスに対して必要な資源だけを割り当てていくことが可能であり、無駄のない資源利用を実現できる [1]。そこで、私たちは、図 3 に示すような、リソース分離を行った μDC である、リソース分離型 μDC(以降、μDDC) を検討している。

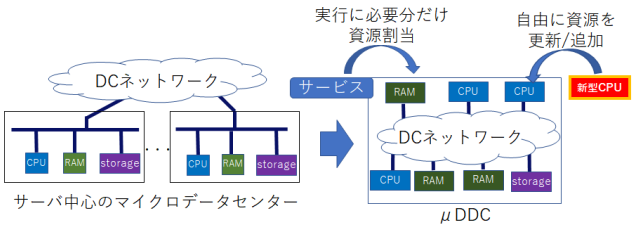

図 1: リソース分離型マイクロデータセンターの概要

μDDC では、処理の実行要求時に、実行資源とその間の通信経路が割り当てられたのち、割当資源同士が必要なデータの読み込みや書き込みのために通信を行いながら処理を実行していく。そのため、実行時にネットワーク遅延が発生し、実行処理の性能が低下してしまうという問題がある。つまり、μDDC において多種多様なサービスを提供するためには、処理の性能要件を満たすのに十分な遅延で資源同士が通信できる必要がある。そして、資源間の通信遅延は、実行資源とその経路の割当や、ネットワーク性能に依存しており、これらを複合的に考慮した μDDC に向けたネットワーク構成の検討が必要である。

現在、私たちは、ネットワークがアプリケーションの処理性能に及ぼす影響を考慮した資源割り当て手法 [2] を提案しており、シミュレーションによってその有効性を確認している。ただし、文献 [2] では、通信されるデータがポアソン分布によって到達すると仮定し、各資源間の遅延を待ち行列モデルを用いて導出することで資源間の通信が性能に及ぼす影響を考慮しているのみであり、パケット単位での詳細な動作を考慮することができていない。そのため、実際の運用に耐える手法ではない可能性もあり、各ネットワーク機器の動作を考慮した、詳細なシミュレータが必要となる。

本研究では、μDDC に向けたネットワーク構成の評価のための、資源間通信シミュレータの設計を示す。シミュレータでは、CPU、メモリ、スイッチの各動作をコンテナとして管理し、キューによる待ち時間として通信遅延時間を模擬する。本稿では、設計したシミュレータを用いて、以前提案した資源割当手法 [2] と従来の資源割当手法で資源割当がなされたときの、資源間の通信遅延時間を計測し、効率的な資源割当の重要性や、資源間の遅延時間がアプリケーションの実行に及ぼす影響について示す。

本稿の構成は以下の通りである。2 章では、リソース分離型のアーキテクチャの実行に必要な要素について説明する。3 章で、μDDC 内の資源間の通信模擬するシミュレータの設計について示す。4 章において、提案されている資源割当手法によって、実行資源とその間の経路を割り当て、シミュレータによって資源間の通信遅延と資源割当の関係について計測する。最後に、5 章でまとめと今後の課題について述べる。

2. リソース分離型アーキテクチャ

リソース分離を行ったデータセンター (以降、DDC) は、CPU、GPU、メモリなどの資源がネットワークで接続されて構築される。リソースを分割を行うことによって、資源の利用率やスケーリング性が向上する [3]。DDC でアプリケーションを実行する際、アプリケーションを実行するために利用する資源とその間の経路が割り当てられ、割り当てられた資源間の通信を通じてアプリケーションの各タスクが処理される。

DDC の構成のためには、(1) 資源間をどのように接続するか、(2) 資源やその経路をどのように割り当てるか、(3)DDC 内でタスクをどのように処理するか、という点を考慮しなければならない。本章において、DDC の構成に必要な要素について示す。

2.1 資源の接続

DDC は、従来のデータセンターと異なり、CPU やメモリが通信するだけで遅延が発生してしまうため、実行されるアプリケーションの性能は低下する [4]。そのため、DDD では、性能低下を防ぐために、高帯域、低遅延で通信可能なネットワークの構築が必要となる。

高帯域、低遅延で通信可能な光スイッチを用いた DDC ネットワークが提案されている [3] [5]。文献 [3] では、光回線スイッチを用いたネットワークアーキテクチャが提案されている。光回線スイッチによって構成することで、資源間の遅延が削減され、従来のサーバ中心のデータセンターと比較して、アプリケーションの実行要求のブロッキング率を下げることに成功している。しかし、光回線スイッチでは、光パスの設定のために資源間の各経路が占有されてしまうため、アプリケーションの実行要求の増加によって、経路が枯渇し、柔軟な資源割り当てが難しくなる可能性がある。文献 [5] では、光パケットスイッチを用いたネットワークの構成が提案されており、光通信による低遅延通信を実現しつつ、パケットスイッチによる柔軟な資源割当を可能としている。一方で、パケットスイッチは光回線スイッチと比較すると、スイッチング処理にかかる遅延が大き

く、資源間の通信遅延が性能に大きく影響する DDC においては、その影響を考慮したネットワークの構成が必要となる。

光回線スイッチやパケットスイッチそれぞれに利点と欠点があり、状況に応じて使い分けていくことが重要となる。とくに、それぞれのスイッチの利用による通信遅延の違いは重要であり、DDC を構成していくためにも、ネットワークの動作を正確に模擬したシミュレータが必要となる。

2.2 資源割当

DDC において、アプリケーションを実行するためには、アプリケーションを実行するためにどの資源を利用し、それらの資源間の通信経路としてどのリンクを使用するかを決定する必要がある。なお、サーバ中心のデータセンターにおける資源割当手法は多く提案されているが、従来のデータセンターは CPU やメモリが同一のマザーボード上に取り付けられており、資源の割り当てによって、計算資源の処理性能には影響しない。DDC においては、資源割当によって、CPU がメモリからデータを読み込む時間そのものが決定づけられるため、資源割当によって資源の性能が決定されるといえる。この点において、DDC に向けた資源割当手法が求められる。

DDC に向けたいくつかの資源割当手法が提案されている [3], [6], [7]。文献 [6] では、資源利用率、経路の帯域幅、資源間の遅延を考慮した資源割当手法が提案されている。各項目の重みを任意に設定することで、管理者の資源管理ポリシーに即した資源割当が可能である。また、文献 [3] では、資源間の帯域幅と経路長を考慮した資源割当手法が提案されている。この手法は、利用可能な帯域幅が大きく、より短いホップで通信可能な資源間にパスを割り当てることで、性能の低下を最小限に抑えている。文献 [7] では、ネットワーク帯域とタスクの完了時間に基づくコストベースの割当手法が提案されている。また、私たちは、ネットワークがアプリケーションの実行性能に与える影響を考慮した資源割当手法を提案しており、資源割当とアプリケーションの実行性能との関係をモデル化し、それをもとに、将来的に必要となる可能性の高い資源やその経路の割り当てを避けることで、より多くの実行要求の割当を実現している。

このように、いくつかの資源割当手法が提案されているが、資源割当が性能に与える影響を考慮している点は共通している。そのため、実際の運用時に資源間の通信がどれだけかかるのかを計測することは、より効率的な資源割当手法の提案や評価のために非常に重要である。

2.3 実行システム

DDC は従来のデータセンターと異なり、資源が分散している。そのため、分散した資源を管理し、アプリケーションを実行するためのシステムが必要となる。そこで、リソース分離型のアーキテクチャに向けた OS である、LegoOS が提案されている [8]。このシステムは、分散した各資源の機能に応じて OS の機能を分割し、それらを分散管理する。さらに、著者らは、Linux との互換性とアプリケーションの展開可能性を文献内で実証している。このシステムを用いることによって、既存のアプリケーションを実行することが可能であり、リソースを分離したアーキテクチャも実現可能であることが示されている。

3. シミュレータの設計

本シミュレータでは、リソース分離型アーキテクチャにおいて、性能低下の直接的な原因となる、CPU とメモリ資源間の通信をシミュレーションする。

図 3 に、シミュレータの操作画面を示す。シミュレータは、全体の動作を管理するマスター、シミュレーション時の動作を指定するクライアント、CPU、メモリ、スイッチで構成される。これらは、それぞれがコンテナとして動作しており、マスターによる管理のもと、CPU が通信相手のメモリからデータを読み込む、またはデータを書き込む際の各機器の動作と遅延をクライアントによって設定されたタスクをもとに模擬していくことで、各資源間で通信が発生する際にかかる遅延時間について計測する。このとき、通信時に各機器の処理にかかる時間はシミュレータ全体で同期されるマスターのグローバルクロックによって管理され、構成機器がデータの送受信待ちや処理にかかるクロック分処理を待機していくことで、資源間の通信時にかかる遅延を再現する。また、シミュレータの使用者は、処理にかかるクロックや各ノード間の通信時にかかるクロックを個別に設定していくことで、各ネットワーク機器の処理時間の違いを再現していくことも可能である。

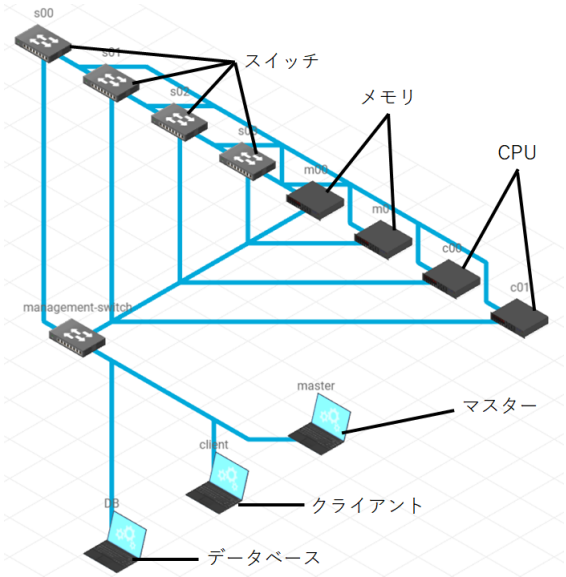

図 2: シミュレータ操作画面

3.1 設定項目

シミュレータで設定できる項目について、表 3 に示す。

3.2 資源間の通信

図 3 に CPU がメモリと通信する際の概要図を示す。

シミュレータでは、あらかじめ使用者が設定する CPU、メモリ、スイッチに対応する各ノードのネットワークトポロジをもとに、クライアントが資源間の経路を指定することで、論理的なネットワークを構築し、資源間の通信をシミュレーションしていく。このとき、各ノードの NIC にはそれぞれ複数のキューを用意されており、各キューでの待機にかかるクロックをもと

表 1: シミュレーション時に設定できるパラメータ項目

パラメータ	説明
スイッチ	
スイッチ処理遅延	通信時のパケット転送処理にかかる遅延
キューの数	スイッチが保有するキューの個数
CPU	
FLOPS	各 CPU の演算速度
クロック周波数	各 CPU のクロック周波数
キュー	CPU が保有するキューの個数
CPU 資源数	各 CPU が保有する資源数
メモリ	
キューの数	メモリが保有するキューの個数
メモリ資源数	各メモリが保有する資源数
リンク	
帯域幅	ノードをつなぐリンクの帯域幅

に、通信時のデータ転送にかかる処理時間を模擬する。また、CPU とメモリにはコアとして複数の領域に分けておき、同一の CPU において、複数の通信の模擬を可能としている。各ノード間の通信について、各スイッチで指定される処理時間をもとにして、一定クロックごとに次ホップへデータを転送していく。前述したように、これらはグローバルクロックによって管理され、資源間の通信時間の違いを再現していくことができる。

3.2.1 処 理 手 順

資源間通信時のシミュレータの処理手順について示す。

（1） 各スイッチや資源間のつながりを表すトポロジ情報をもとに物理的なノードのつながりをシミュレータ上に構築

（2） マスターへ CPU やメモリ、スイッチに対応する各ノードのノード名、アドレス、通信ポート、種別を送信

（3） マスターからデータベースでノード情報を集約

（4） クライアントが資源間の通信時の経由ノードと、メモリから CPU への書き込みまたはメモリから CPU への読み込み命令をマスターへ送信

（5） マスターから対応する各ノードへ命令と経由ノードの情報を送信し、資源間の通信をシミュレーション

上に示すように、利用者は、物理的なネットワークトポロジと各機器の遅延時間を指定し、クライアントを通して通信する資源間の経路と、その経路で CPU、メモリとの書き込み/読み込み命令を指定することでシミュレーションを行うことができる。指定できる命令としては、CPU からメモリへのデータの書き込み、メモリから CPU へのデータの読み出しの 2 つである。処理が割り当てられている複数の資源間で、並列的にデータの読み出しや書き込みを行うことで、μDDC の運用時の、資源間の同時通信の際に遅延がどのようになるかを計測できる。

4. シミュレーション

μDDC での実行が想定されるアプリケーションの実行資源と資源間の経路を以前提案した資源割当手法を用いて割り当てた場合について、シミュレータによって各資源間の遅延を計測することで、資源割当手法の有効性を確認するとともに、実際に CPU やメモリ、スイッチの処理が行われた際の資源間の遅延について考察する。

4.1 計 測 環 境

計測のために設定したネットワークトポロジ、シミュレータのパラメータ設定、割当アプリケーション、実行した資源割当について示す。

4.1.1 ネットワークトポロジ

資源間の通信遅延を計測するために、ネットワークトポロジの典型例として 2D トーラストポロジで計測を行う。本稿では、シミュレータの動作確認としての計測をしており、小規模な環境での計測を行う。そこで、9 個のスイッチで構成された 3 × 3 の 2D トーラストポロジで計測を行う。図 4 に 2D トーラストポロジにおける各資源の接続を示す。資源間の通信遅延を抑制するために、CPU とメモリは隣接するように配置されており、2D トーラスを構成する全てのスイッチに資源が接続している。なお、同種の資源は資源プールとして複数個がまとめられており、スイッチには 1 つの資源プールが接続する。各メモリプールには 4 つメモリが含まれ、CPU プールには 2 個の計算資源が含まれる。

4.1.2 実行アプリケーション

μDDC はエッジに配置されるデータセンターであるため、エッジで実行される典型例として、ResNet [9] を用いた画像分類アプリケーションを想定する。想定する画像分類アプリケーションは、3 つのプロセスによって構成される。プロセス 1 では、タスクの実行命令に対する、実行資源への命令処理、プロセス 2 では、実行資源がメモリから必要なデータを読み込む処理、プロセス 3 では、タスクのメインプロセスである画像分類を実行する。ここで、図 5 に想定するアプリケーションのプロセスと、実行資源の関係について示す。

4.1.3 資 源 割 当

資源割当による通信遅延の影響について計測するために、4.1.3 章にしめすアプリケーションに対して、以前私たちが提案した資源割当手法 [2] と、従来提案されていた、その時点で最も低遅延で通信可能な資源を割り当てる手法の 2 手法によって資源割当を行う。この 2 手法を、それぞれ RA-CNP、NP と呼ぶ。そして、資源割当のために、ネットワーク上の存在する全ての計算資源が割り当てられるまで、アプリケーションの実行要求を生成する。各手法は生成された実行要求のために、に示す計算資源とメモリ資源、その間の経路を割り当てていく。

また、実環境では様々な時間制約を持つ実行要求がなされると考えられる。そこで、資源間で許容できるホップ数が 3 種類の実行要求を生成し、そのホップ数の制約を満たすように割り当てる。各要求とホップ数の制約を表 2 に示す。なお、生成する要求の総数は 6 個であり、各実行要求は 2 つずつなされる。なお、資源割当方法によっては、ホップ数制約を満たす割当が不可能である場合がある。その場合は、可能な限りホップ数が小さくなるように要求を割り当てる。

4.1.4 シミュレータのパラメータ設定

シミュレーションのために設定したパラメータについて、以下に示す。

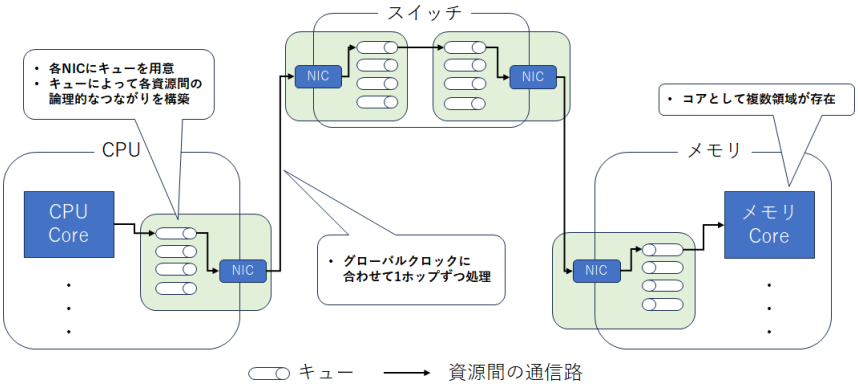

図 3: シミュレータ操作画面

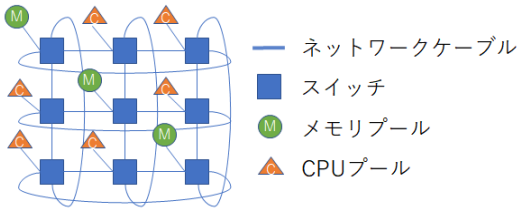

図 4: 2D トーラストポロジ

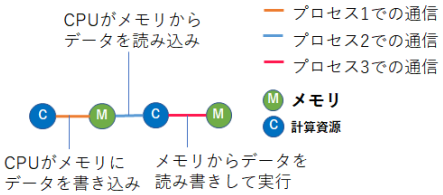

図 5: 実行資源の関係

表 2: 各要求のホップ数制約と生成数

	制約 1	制約 2	制約 3
ホップ数制約	6	4	3
生成数	2	2	2

表 3: シミュレーションのためのパラメータ設定

パラメータ	値
スイッチ	
スイッチ処理遅延	3μs
キューの数	5
CPU	
FLOPS	76.8
クロック周波数	2.9GHz
キュー	5
CPU 資源数	2
メモリ	
キューの数	5
メモリ資源数	4
リンク	
帯域幅	10Gbps

4.2 計測結果

図 6 に、生成された 6 個の要求に対して、各資源割当手法によって資源割当を行ったときの資源間の通信遅延時間の合計を示している。資源間の通信遅延時間の合計は図 5 に示す割当資源の関係をもとに、プロセス 1 の実行計算資源がメモリ資源へデータを書き込む時間と、プロセス 2 において計算資源がメモリ資源からデータを読み込む時間、プロセス 3 において計算資源がメモ委資源からデータを読み込む時間の合計として求めている。また、各要求の制約は表 2 の表記と対応している。

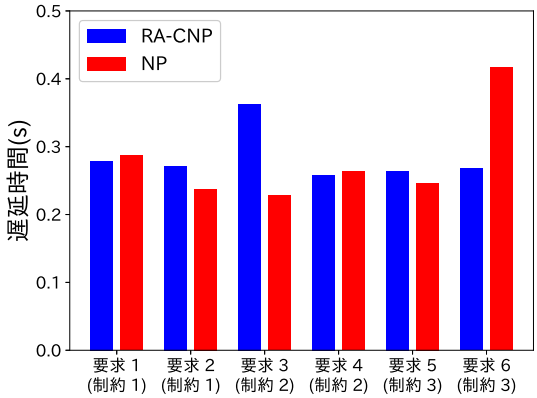

図 6: 各割当手法による生成要求ごとの通信遅延時間

図 6 から、各資源割当手法において、ほとんどの要求において 0.3 秒弱程度の遅延時間内に収まっていることがわかる。一方で、手法 RA-CNP では、要求 3 の遅延時間が大きく、手法 NP では、要求 6 の遅延時間が大きくなっている。表 2 から、要求 3 の方が要求 6 よりもホップ数制約は緩く、手法 RA-CNP が、低遅延で通信可能な経路を温存することで、将来のより厳しい要求のための低遅延で通信可能な経路を残すことができたといえる。遅延時間を見ても、要求 6 の各手法の遅延時間において、0.1 秒程度の差があり、タイムセンシティブなサービスの提供も考えられる μDDC においては無視できない差となっている。μDDC において、資源割当は非常に重要な要素であるといえ、将来のアプリケーションを考慮する資源割当手法が有効であるといえる。

また、ホップ数制約が 3 であり、本ネットワークトポロジ上では最短ホップでの通信が必須である要求 5 や 6 でも、遅延時間は最小で 0.25 秒程度かかっている。実際にサービスを実行する際には、資源間の通信時間に加え、CPU での処理時間や、

ユーザと μDDC 間の遅延も発生する。より高速通信が可能なネットワーク機器の利用や、トポロジの検討、アプリケーションの通信タイミングの効率的な制御方法が必要になる。

5. おわりに

本稿において、μDDC ネットワークの評価のための計算資源の動作も考慮した資源間通信シミュレータの設計について示した。本シミュレータでは、CPU、メモリ、スイッチの動作を模擬し、割り当てられた資源間で通信が発生する際の、処理遅延をクロック単位で制御することにより、資源間で発生する通信時間やその違いを再現することができる。本シミュレータによって、以前私たちが提案した資源割当手法による資源割当時の、資源間の通信遅延時間を計測し、従来手法と比較して、割り当て要求の制約を考慮した資源割り当てが可能であることを資源間の通信遅延の面から実証した。さらに、資源間の通信遅延が性能に対して非常に大きな影響を持つことも確認し、より低遅延での通信が可能なネットワーク構成の必要性についても確認した。

今後の予定としては、シミュレータにおいて、小規模のネットワークでの計測にとどまっており、より多くの資源間で通信が発生するような場合に、資源間の通信遅延はどのように変化するかの計測が必要である。また、スイッチの処理遅延や CPU の性能が通信に及ぼす影響について、複数のパラメータを設定し、その影響を評価していくことで、μDDC に求められる機器の性能要件について検討していく予定である。

謝　辞

本研究の一部は NICT 委託研究・JPJ012368C00101「多種多様なサービスに対応可能な高機能エッジクラウド情報処理基盤の研究開発」によるものである。また、本シミュレータは、株式会社あくしゅによって実装されたものである。ここに記して謝意を表す。

文　献

[1] S. Han, N. Egi, A. Panda, S. Ratnasamy, G. Shi, and S. Shenker, “Network support for resource disaggregation in next-generation datacenters,” in *Proceedings of the Twelfth ACM Workshop on Hot Topics in Networks*, pp. 1–7, Nov. 2013.

[2] A. Ikoma, Y. Ohsita, and M. Murata, “Disaggregated micro data center: Resource allocation considering impact of network on performance,” in *Proceedings of 2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*, pp. 360–365, 2023.

[3] G. Zervas, H. Yuan, A. Saljoghei, Q. Chen, and V. Mishra, “Optically disaggregated data centers with minimal remote memory latency: Technologies, architectures, and resource allocation [invited],” *Journal of Optical Communications and Networking*, vol. 10, no. 2, pp. A270–A285, 2018.

[4] P. X. Gao, A. Narayan, S. Karandikar, J. Carreira, S. Han, R. Agarwal, S. Ratnasamy, and S. Shenker, “Network requirements for resource disaggregation,” in *Proceedings of 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, (Savannah, GA), pp. 249–264, USENIX Association, Nov. 2016.

[5] N. Terzenidis, M. Moralis-Pegios, G. Mourgias-Alexandris, T. Alexoudi, K. Vyrsokinos, and N. Pleros, “High-port and low-latency optical switches for disaggregated data centers: The hipo 了 aos switch architecture,” *Journal of Optical Communications and Networking*, vol. 10, no. 7, pp. 102–116, 2018.

[6] A. D. Papaioannou, R. Nejabati, and D. Simeonidou, “The benefits of a disaggregated data centre: A resource allocation approach,” in *Proceedings of 2016 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7, Dec. 2016.

[7] M. Amaral, J. Polo, D. Carrera, N. Gonzalez, C.-C. Yang, A. Morari, B. D’Amora, A. Youssef, and M. Steinder, “Dr-maestro: orchestrating disaggregated resources on virtualized data-centers,” *Journal of Cloud Computing*, vol. 10, pp. 1–20, mar 2021.

[8] Y. Shan, Y. Huang, Y. Chen, and Y. Zhang, “LegoOS: A disseminated, distributed OS for hardware resource disaggregation,” in *Proceedings of 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, (Carlsbad, CA), pp. 69–87, USENIX Association, Oct. 2018.

[9] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.