エッジクラウド環境におけるマルチクラスタ向けコンテナ配置手法の提案

2020SE007 林朱里人
指導教員 宮澤元

1 はじめに

IoT(Internet of Things) デバイスやコネクティッドカーの普及に伴い，それらのデータ処理に用いられるエッジクラウドへの注目が高まっている．エッジクラウドとは，クラウドコンピューティングの仮想化技術をエッジコンピューティングに適用し，両者の区別なく状況に応じて適切な計算リソースを利用できるようにする技術である．

エッジクラウド環境は地理的に分散した複数のクラスタから構成されるので，これらを一元管理して 1 つのクラスタとして用いるマルチクラスタ技術が注目されている．一元管理されたどのクラスタにもコンテナを配置できるのでクラスタを跨いだ負荷分散や障害対策に利用できる．

一方，従来のマルチクラスタコンテナオーケストレータでは，クライアントのアクセス頻度やクラスタ間通信遅延を収集し，コンテナ配置に動的に反映する仕組みを備えていない．宣言的に行った各クラスタの重み付けに基づくコンテナ配置は可能だが，状況が変化した場合，利用者がそれを検知して重み付けを自身で変更する必要がある．このような問題に対し，クラスタ間の通信遅延に基づいてサービス応答時間を最適化するコンテナ配置を行う先行研究はある [1] が，複数クライアントがマルチクラスタにアクセスする場合のクラスタ間の通信遅延と負荷分散の両方を考慮したコンテナ配置手法に関する先行研究は少ない．

本研究の目的は，マルチクラスタ環境において複数クライアントからのアクセスがある場合に特定のクラスタへの処理負荷の集中を抑えるとともにサービス処理時間を低減することである．特定のクラスタへの処理の集中を抑えることで，そのクラスタの計算リソース不足を抑えられる．また，アクセス時の通信遅延が小さいクラスタにサービスを配置することでサービスの処理時間を改善できる．

本稿ではクラスタ間通信遅延を考慮しつつ，クライアントからのリクエスト数とクラスタの計算リソースに応じてコンテナを配置するマルチクラスタ向けコンテナ配置手法について述べる．クライアントはもっとも近いクラスタにリクエストを出すが，クラスタのリソース使用状況と他のクラスタとの通信遅延を考慮して，リクエストを処理するコンテナ配置を決定する．そのためのクラスタのリソース使用状況とクラスタ間通信遅延の具体的な条件は実験により決定する．研究課題は以下の 2 点である．

- マルチクラスタのクラスタ間通信遅延とクライアントのリクエスト数を考慮したコンテナ配置手法の提案
- 実験によるクラスタのリソース使用状況とクラスタ間通信遅延の条件の決定

2 研究の背景

本節ではエッジクラウド環境におけるコンテナスケジューリングの先行研究と KubeFed について述べる．

2.1 エッジクラウド環境のコンテナスケジューリング

古澤らは遅延値に基づく車両向けコンテナアプリケーションのオフローディング先を，車載器，エッジサーバ，クラウドから応答時間を基準に動的に切り替えるフレームワークを提案している [1]．この研究では，複数クライアントが同時にリクエストを送信することは想定していない．

上野はエッジコンピューティング環境でデバイスと計算ノード間の通信遅延を基にコンテナの再配置，配置されているポッドの削除を行うデスケジューラを提案している [2]．この研究ではマルチクラスタ環境でのクラスタ間スケジューリングは考慮されていない．

2.2 KubeFed

Kubernetes Cluster Federation(KubeFed) は複数の Kubernetes クラスタの構成を調整・管理するマルチクラスタオーケストレータである [3]．KubeFed は Federation 内の複数クラスタを管理するホストクラスタ 1 つと 1 つ以上のメンバークラスタで構成される．

3 提案手法

複数コンテナのクラスタ間配置を最適化し，複数クライアントからのアクセス要求に対するサービス処理時間の低減を目的とし，マルチクラスタ向けコンテナ配置手法を提案する．クライアントが最も近くのクラスタにサービスを要求すればサービス処理時間を低減できるので，各クラスタにサービスを要求するクライアント数に応じてサービスコンテナを割り当てる．しかし，クラスタの計算リソースに対して過剰なコンテナが割り当てられると，サービス応答時間が増大する．そこでこのような場合，計算リソースに余裕があるクラスタへのコンテナの割り当てを増やし負荷分散をはかる．ただし，クライアントに対するサービス処理時間の観点からは，割り当てるコンテナ数の具体的な条件は，クラスタのリソース使用状況だけではなく，クラスタ間通信遅延も考慮して決定する必要がある．

4 実験

提案手法におけるコンテナの割り当てについての具体的な条件を決定するために実験を行った．

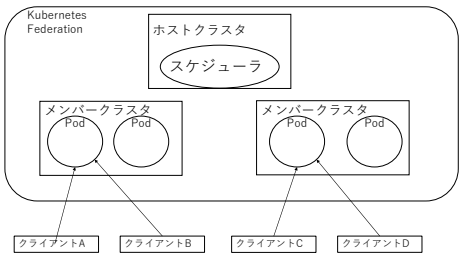

図 1 実験におけるマルチクラスタの構成

4.1 実験環境

PC を 2 台と Raspberry Pi 4 Model B(以下 RPi) を 1 台用いた．PC のうち 1 台に構築したクラスタをホスト，他のクラスタをメンバとして KubeFed を用いてマルチクラスタを構成した．図 1 に実験環境を示す．

4.2 実験内容

各メンバクラスタで，nginx Web サーバの Pod に対し同一クラスタで動作する Apache Benchmark から HTML ファイルにアクセスする．このときのクライアントに対するサービス処理時間と各クラスタにおける CPU とメモリの使用率を計測する．アクセスするファイルサイズは 1KB, 10KB, 1MB, 100MB の 4 通り，同時接続数は 10, 50 の 2 通り，接続ごとのリクエスト数は 10，50 の 2 通りに変化させた．各クラスタに配置する Pod 数は 20 とした．

4.3 実験結果

PC 上，RPi 上のメンバクラスタに対する実験結果をそれぞれ表 1，2 に示す．N/A は接続エラーによる未計測を示す．1KB と 10KB の場合， PC 上も RPi 上もリクエスト応答時間に大きな差がないのに対し，ファイルサイズが大きいとファイル転送時間が直接影響し，ファイルサイズが 1MB の場合 PC で 1 ミリ秒程度，RPi で 10 ミリ秒程度が要求の処理にかかる．また，CPU 使用率はファイルサイズが大きく総リクエスト数が多いほど高い傾向で，PC で最大 18% 程度，RPi で最大 40% 程度の負荷だった．メモリ使用量は PC，RPi ともに大きな変化はなかった．

表 1 PC 上クラスタの平均サービス処理時間 (msec.)

ファイル	接続数 × リクエスト数			
サイズ	10 × 10	10 × 50	50 × 10	50 × 50
1KB	0.141	0.123	0.128	0.121
10KB	0.138	0.126	0.133	0.126
1MB	1.038	1.042	1.015	N/A
100MB	37.493	37.472	N/A	N/A

表 2 RPi 上クラスタの平均サービス処理時間 (msec.)

ファイル	接続数 × リクエスト数			
サイズ	10 × 10	10 × 50	50 × 10	50 × 50
1KB	0.843	0.779	0.859	0.791
10KB	1.041	0.822	0.840	0.831
1MB	10.243	N/A	10.957	10.895
100MB	1149.786	1156.505	1227.589	N/A

4.4 考察

本実験では PC 上のクラスタに対して負荷を十分かけられなかった．一方，RPi 上のクラスタでは 1MB 程度のファイル転送でもある程度の負荷がかかる．エッジデバイスのような処理性能の低いデバイスを利用する場合，高負荷時にクラスタ間で負荷分散を行うことで，リクエスト応答時間を保ちつつエッジデバイスの負荷を低減できる可能性がある．本実験はサービス側の計算負荷が比較的小さい HTML ファイル転送の実験であり，サーバ側からクライアント側に情報を受け渡す単純なファイル共有サービスやビデオオンデマンドのようなアプリケーションを想定している．クライアントからサービスへデータをアップロードするような IoT アプリケーションを想定したものにはなっておらず，エッジクラウド環境で想定されるコンテナ配置の具体的条件を示すのに十分とは言えない．また，現状では KubeFed にクラスタ間で負荷分散を行う機能がなく，クラスタ間通信遅延や帯域の変化の影響を調べる実験を行うことができなかった．今後，さまざまなワークロードや環境を想定した実験をさらに行う必要がある．

5 おわりに

新たなマルチクラスタ環境向けコンテナ配置手法の検討を行った．現在のマルチクラスタ環境におけるコンテナスケジューリングの課題を指摘し，新たなコンテナ配置手法を提案するための実験を行った．実験の結果，エッジデバイスでは単純なファイル転送でも中程度の負荷がかかることがわかり，エッジクラウド環境における負荷分散を考慮したマルチクラスタ向けコンテナ配置手法の必要性を示せた．今後はエッジクラウド環境で想定される様々なワークロードに対してクラスタ間通信遅延や帯域の変化の影響を調べる実験を行い，具体的なマルチクラスタ向けコンテナ配置手法を提案する．

参考文献

[1] 古澤 徹 他: “遅延値に基づく車両向けコンテナ型アプリケーションの動的エッジオフローディングの実装と評価,” 情報処理学会論文誌デジタルプラクティス (TDP), Vol.3, No.3, pp.32–43, 2022.

[2] 上野 空: “エッジコンピューティングプラットフォームにおける通信遅延を考慮したコンテナデスケジューラ,” 2022 年度南山大学理工学部卒業論文, 2023.

[3] Kubernetes Cluster Federation, `https://github.com/kubernetes-retired/kubefed`, (2024/1/27 access).