5A-05

エッジコンピューティング環境における コンテナスケジューリングに関する検討

宮澤 元 †
南山大学 理工学部 電子情報工学科 †

1 はじめに

今日のクラウドコンピューティング環境 (クラウド環境) においては，Google 社で開発された Kubernetes[1] をはじめとして，コンテナ化されたアプリケーションの管理を自動化するコンテナプラットフォームが広く利用されている．コンテナプラットフォームによって，クライアントからのアクセス状況に応じてアプリケーションコンテナをスケールしたり，計算ノードの構成の変化に柔軟に対応したりすることができる．

一方，Internet of Thins (IoT) の普及を背景に，ネットワークエッジの計算リソースを活用するエッジコンピューティングが注目されている [2]．エッジコンテナプラットフォームとは，コンテナプラットフォームをクラウド環境だけでなくエッジコンピューティングをも含めた環境 (エッジ環境) で動作するように拡張するもので，様々なシステムが提案されている [3, 4]．

エッジコンテナプラットフォームを効率的に運用するためには，エッジ環境における計算ノードやネットワーク接続性の多様性を考慮する必要がある．クラウド環境では多くの計算リソースを持つ計算ノードが広帯域ネットワークで接続されているのに対し，エッジ環境では IoT 機器のような比較的計算リソースに乏しい計算ノードが狭帯域ネットワークで接続されていたり，クライアントと計算ノード間の通信レイテンシが大きく変動したりすることもある．

本稿ではエッジコンテナプラットフォームにおける計算ノードへのコンテナの配置手法について検討する．コンテナを計算ノードへ配置するにあたり，計算ノードの計算リソースやその利用状況だけではなく，クライアントから計算ノードへの通信レイテンシを考慮するコンテナスケジューリングシステムを提案する．

2 エッジコンテナプラットフォーム

エッジ環境の普及に伴い，クラウド環境におけるデータセンタなどの計算ノードとエッジ環境の計算ノードの両方を計算リソースとして利用するエッジコンテナプラットフォームが提案されている [3, 4]．図 1 にエッジコンテナプラットフォームの概要を示す．クラウドノードだけでなくエッジノードや，場合によってはクライアントにもコンテナを配置して実行することで，データローカリティを生かし通信レイテンシを低減することができる．

エッジコンテナプラットフォームを効率的に運用するには，以下のような条件を考慮してコンテナを適切な計算ノードに配置して動作させる必要がある．

- 計算ノードにおける計算リソース利用状況
- 計算ノードの CPU 性能
- コンテナによる計算ノードに対する優先度
- クライアントとコンテナを実行する計算ノード間の通信レイテンシ

A Discussion on Container Scheduling in Edge Computing Environment

† Hajime Miyazawa, Department of Electronics and Communication Technology, Faculty of Science and Engineering, Nanzan University

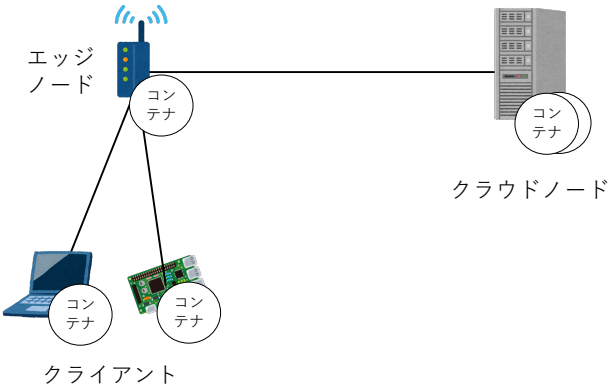

図1　エッジコンテナプラットフォームの概要

- クライアントによるコンテナを実行する計算ノードに対する優先度

既存のコンテナプラットフォームでもこれらの条件の一部を考慮してコンテナを配置している．しかし，エッジ環境における計算ノード間の計算リソースや CPU 性能の差異に配慮したものは少ない．また，コンテナを利用するクライアントとの関係性に着目したものはほとんどない．

3　コンテナスケジューリングの実現

本研究ではエッジコンテナプラットフォームを実現するために，2 節で示したような条件をコンテナ配置に利用するコンテナスケジューリングシステムを提案する．計算ノードごとに動作するエージェントを用いて計算ノードの計算リソースの状況を把握するとともに，他の計算ノード上，特にクライアント側エージェントと通信することによって通信レイテンシを計測する．クライアント側エージェントはサービスを利用する際のエンドポイントとしても利用され，クライアントからの通信を適切な計算ノードへ転送する．

提案したコンテナスケジューリングシステムを Kubernetes を拡張して実装することを検討している．具体的には以下のエージェントを実装する．

レイテンシサーバ　クライアントと計算ノードで動作し，これらの間の通信レイテンシを計測する．

スケジューラ　Kubernetes の Scheduling Framework を用いて実装する．

デスケジューラ　一度配置されたコンテナ (Pod) を再配置するための Descheduler に通信レイテンシに関するポリシを追加する．

プロキシ　kube-proxy を拡張し，iptables を制御する．必要に応じてスケジューラ，デスケジューラにコンテナの配置/再配置を促す．

4　まとめ

本稿ではエッジコンテナプラットフォームにおける計算ノードへのコンテナの配置手法について検討した．コンテナを計算ノードへ配置するにあたり，計算ノードの計算リソースやその利用状況だけではなく，クライアントから計算ノードへの通信レイテンシを考慮するコンテナスケジューリングシステムを提案する．今後は提案したシステムを実装し，エッジコンテナプラットフォームの実現に関してさらに検討を進める．

参考文献

[1] Kubernetes; http://kubernetes.io/. (accessed Jun. 7, 2022.)

[2] F. Bonomi, et al., “Fog computing and its role in the internet of things,” in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing - MCC ’12*, pp.13–16, 2012. DOI: 10.1145/2342509.2342513

[3] Ying Xiong, et al., “Extend Cloud to Edge with KubeEdge,” in *2018 IEEE/ACM Symposium on Edge Computing (SEC)*, pp.373–377, 2018.

[4] Corentin Dupont, et al., “Edge computing in IoT context: Horizontal and vertical Linux container migration,” in *2017 Global Internet of Things Summit (GIoTS)*, pp.1–4, 2017.