

IaaSクラウド向け軽量実行環境におけるアプリケーション

実行性能に関する考察

宮澤 元[†]南山大学 理工学部 ソフトウェア工学科[†]

1. はじめに

IaaS(Infrastructure-as-a-Service) 環境を提供するクラウドコンピューティング(クラウド)のアプリケーション実行基盤としてコンテナ仮想化技術が普及している。マルチテナントの IaaS 環境では複数の実行環境を適切に隔離することが不可欠だが、従来広く利用されてきた仮想マシンは仮想マシンモニタによる物理計算機の仮想化を利用してこれを実現したものである。一方コンテナ仮想化では、オペレーティングシステム(OS)カーネルが用意する名前空間の機能を用いて提供されるコンテナと呼ばれる実行環境を用いてアプリケーションプロセスの粒度で隔離を行う。コンテナはそれぞれにゲストOSを用意する必要がなく、仮想マシンモニタでの仮想CPUやメモリの管理が仮想マシン上のゲストOSでのプロセスや仮想メモリの管理と重複してしまうといった問題が存在しないので低オーバーヘッドで実現することができる。

一方、Unikernels[1]をはじめ、従来の仮想マシンのオーバーヘッドを低減するための研究も行われている。Unikernelsとは、ユーザコードにライブラリOSをリンクして仮想マシン上で直接動作できるようにした単一のアプリケーションバイナリのことである。Unikernelsを利用すると、仮想マシンのゲストOSとして汎用OSを利用する場合と比べ、ゲストOSに起因するオーバーヘッドを低減でき、性能向上が見込まれる。

しかし、コンテナとUnikernelsのどちらがアプリケーションの実行により適しているかは状況により異なる。従来型の仮想マシンも含めたアプリケーションの実行環境の性能評価に関する研究も行われているが、使用する仮想マシンモニタの種類やアプリケーションの特性などによっても結果は異なる[3, 4, 5]。例えばUnikernels環境では、アプリケーションはライブラリOSと一体化した単一のバイナリとして動作するので、汎用のホストOS上でプロセスとして動作するコンテナ仮想化環境よりコンテキストスイッチが少ない分、性能的に有利だと考えられる。その反面、Unikernelsにおいても仮想化の

オーバーヘッドは依然として存在する。特に入出力仮想化のオーバーヘッドは一般に大きく、ネットワーク通信などの入出力を多く行うアプリケーションではコンテナ仮想化が有利となってもおかしくない。

我々は、コンテナやUnikernelsをクラウドにおけるアプリケーションの軽量実行環境と捉え、アプリケーションの動作状況に合わせて適切な軽量実行環境を与えられるようなIaaS実行基盤の実現を目指している。そのためにアプリケーションの動作状況に対して、既存の軽量実行環境がどのように振る舞うかを具体的に把握する必要がある。本稿では、軽量実行環境におけるアプリケーションの実行性能について考察する。単純な数値計算アプリケーションを複数の軽量実行環境上で動作させて性能測定実験を行い、比較を行う。

2. 軽量実行環境の性能比較

コンテナ仮想化環境やUnikernel環境など、軽量実行環境についての性能比較は数多くなされている。起動時間やメモリフットプリントといった軽量実行環境そのものの性能比較や、いくつかの軽量実行環境におけるアプリケーションの性能比較については様々な報告がある。

一方、複数のアプリケーションを動作させた状態でのプラットフォーム全体での性能比較はほとんどなされていない。マルチテナントのIaaS環境では、同一の物理マシン上で複数の仮想マシンやコンテナが動作しており、このような状況における軽量仮想実行環境ごとのアプリケーションの性能特性を明らかにすることは重要だと思われる。

3. 実験

軽量実行環境として代表的なUnikernels環境の一つであるOSv(version 0.54.0)と、広く利用されているコンテナ仮想化環境のDocker(version 19.03.5)を利用して、単純な数値計算アプリケーションを動作させ、実行時間を計測する実験を行った。OSvについては、QEMU(version 2.11.1)を用いて動作させた。

3.1 実験環境

実験には表1に示す仕様のPCを用いた。また軽量実行環境にはそれぞれ2Gバイトのメモリを割り当てた。

A Study on Performance of Applications on Lightweight Execution Environments for IaaS Cloud

[†] Department of Software Engineering, Faculty of Science and Engineering, Nanzan University

表 1: 実験に利用した PC の仕様

CPU	Intel Core i7-8700K 3.7GHz
コア数	6 コア 12 スレッド
メモリ	DDR4-2666 DRAM 32G バイト
ストレージ	Plextor SSD PX512M9PEG
OS	Ubuntu Server 18.04.3 Kernel 4.15.0

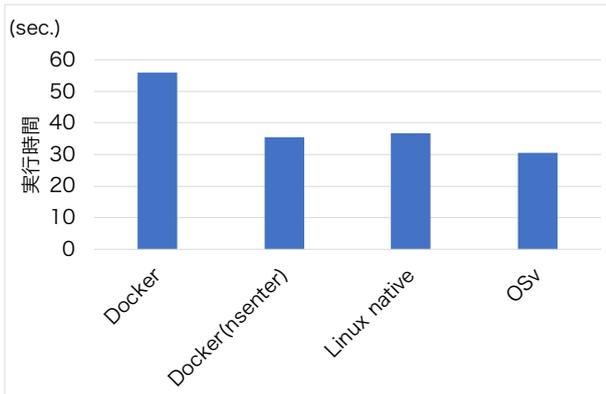


図 1: 計算実験の結果

3.2 計算実験

モンテカルロ法を用いて円周率の近似値を計算するアプリケーションを作成し、このアプリケーションの実行時間を計測する実験を行った。このアプリケーションではサンプルサイズとして定めた 10 億個の点の座標を単位格子内にランダムに生成する。これらの点のうち、単位円内に生成された点の数を集計することにより 1/4 円の面積を計算して円周率の近似値を求める。プログラムは C 言語で作成した。

Docker 環境と OSv 環境の他、コンテナを利用しない Linux 環境 (Linux native) で行った実験結果を図 1 に示す。Docker 環境については、アプリケーションを直接コンテナ内で動作させたものの他、nsenter コマンドを用いてコンテナの名前空間内でアプリケーションを動作させた後にコンテナの cgroup にアプリケーションを追加する方法 (Docker(nsenter)) でも実験を行った。各環境で実験を 10 回ずつ行った平均値を結果とした。

ほぼ CPU バウンドの単純なアプリケーションであるにも関わらず、Docker 環境での実行時間がそれ以外の環境での実行時間と比べて 1.5 倍程度遅かった*。この実験結果は、田尻による Apache Spark を用いた類似の実験の結果 [6] と整合する。Docker(nsenter) 環境も含め、Docker 環境以外ではほぼ同等の実行時間であり、この

*筆者の以前の報告 [7] では Docker 環境での実行時間が他の軽量実行環境より 2 倍近く速い結果となっていたが、これは乱数発生に用いた random 関数の実装が Docker 環境と OSv 環境とで異なっていたことが原因であり、今回のアプリケーションでは修正済みである。

アプリケーションの実行時間には CPU の処理性能がそのまま反映されたと考えられる。一方、Docker 環境での性能低下がスケジューリングのオーバーヘッドや cgroup によるリソース制限に起因するものとは考えにくく、原因を調査中である。なお、OSv 環境での実行時間は他の環境に比べてやや短く、OS オーバヘッドが低減される効果が現れていると考えられる。

4. まとめ

いくつかの軽量実行環境を用いて単純な数値計算アプリケーションを実行させ、実行時間を計測する実験を行った。実験を行ったいずれの環境でも同等の実行時間を要するが、Unikernels 環境での実行時間がやや短い結果となった。しかしコンテナ環境では標準的な操作で動作させた場合の結果が他の環境に比べて大幅に遅いことが分かり、原因を調査する必要がある。

今後は、さまざまな状況において、アプリケーションの実行にどのような軽量実行環境が適しているかを調べる。特に I/O バウンドアプリケーションについて軽量実行環境を比較する実験を行う他、より現実的なワークロードにおけるアプリケーションの振る舞いについても調べる。

参考文献

- [1] Anil Madhavapeddy, et al., “Unikernels: Library Operating Systems for the Cloud,” in *Proceedings of the 18th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS’13)*, pp.461–472, 2013.
- [2] Filipe Manco, et al., “My VM is Lighter (and Safer) than your Container,” in *Proceedings of the 26th Symposium of Operating Systems Principles*, pp.218–233, 2017.
- [3] M. G. Xavier, et al., “Performance Evaluation of Container-Based Virtualization for High Performance Computing Environments,” in *Proceedings of the 21st Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP 2013)*, pp.233–240, 2013.
- [4] S. Gupta and D. Gera, “A Comparison of LXD, Docker and Virtual Machine,” in *International Journal of Scientific & Engineering Research*, Volume. 7, Issue 9, pp.1414–1417, 2016.
- [5] B. Xavier, et al., “Time Provisioning Evaluation of KVM, Docker and Unikernels in a Cloud Platform,” in *Proceedings of the 16th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*, pp.277–280, 2016.
- [6] 田尻翔太, “IaaS 環境におけるマルチプロセスアプリケーションを考慮した仮想化基盤の動的構成管理に関する研究,” 南山大学大学院理工学研究科修士論文, 2017.
- [7] 宮澤元, “軽量実行環境におけるアプリケーションの実行性能比較,” 情報処理学会第 81 回全国大会講演論文集 (1), pp.29-30, 2019 年.