1L-07

分散環境におけるコンテナオーケストレーションシステムのためのライブマイグレーションの実現

北出　紘章†　　塩川　浩昭‡　　北川　博之‡

筑波大学　情報学群　情報科学類†　　筑波大学　計算科学研究センター‡

１　はじめに

近年，クラウドコンピューティングや分散データ処理の分野において，コンテナ技術を活用した分散システムが増えている．また，コンピュータクラスタを利用してスケーラブルでマルチテナントなコンテナ実行環境を構成し，アプリケーションのデプロイ，リソースのスケジューリング，および障害復旧処理を自動化するため，コンテナオーケストレーションシステムに需要がある．

Apache Hadoop YARN [1][2] は分散データ処理に使われるクラスタのリソースとコンテナの管理システムである．YARN はプログラミングモデルを問わず多種多様な分散処理フレームワークの実行基盤となっているほか，現在ではバッチ処理だけでなく Long-running サービスや Docker にも対応しており [3]，コンテナオーケストレーションシステムと同等の機能を有する．

ところが，Hadoop YARN やその他の公開されているコンテナオーケストレーションツール ([4] など) には，実行中のコンテナを終了・再起動せずに他のホストに移動するライブ マイグレーションを実装しているものがない．オーケストレーション環境下におけるライブ マイグレーションが実現することで，1) クラスタのメンテナンスなどによりノードを切り離す場合でもダウンタイムおよび状態の揮発なしにコンテナを移動できる，2) Locality awareness の機会が増えることでデータ転送にかかるオーバーヘッドの削減が期待できる，および 3) [5] を適用することによりクラスタの性能を向上させることが可能になる，といった利点が挙げられる．

本研究は，Hadoop YARN を対象としてコンテナのライブマイグレーション機能を実装し，コンテナオーケストレーションシステムにおけるライブ マイグレーションが実現可能であることを示すことを目的とする．

Container live migration technique for the orchestration system in a distributed environment

† Hiroaki KITADE, College of Information Science, School of Informatics, University of Tsukuba

‡ Hiroaki SHIOKAWA, Hiroyuki KITAGAWA, Center for Computational Sciences, University of Tsukuba

２　対象システム

Hadoop YARN におけるコンテナの実体は，計算リソース（メモリと仮想 CPU コアの組）が隔離されて実行されているプロセスであり，環境変数やコマンドなどを設定して起動する．プロセスのライブマイグレーションは，チェックポイント／リストア技術 (C/R) を利用することで実現できる．

2.1　Hadoop YARN

YARN はマスター・スレーブ型のクラスタで構成される（図１）．代表の 1 ノードにはクラスタのリソースとコンテナの管理・スケジューリングを行う ResourceManager (RM) が，その他のノードには自身のノードのリソースとコンテナを管理する NodeManager (NM) が稼働する．また YARN 上で実行されるジョブは複数のコンテナで構成されるが，このうち最初に起動するコンテナは ApplicationMaster (AM) となり，他のワーカコンテナを統制する役割をもつ．

クライアントが RM にジョブを投入すると，AM コンテナの実行に必要なリソースがスケジューリングされる．AM が配置され起動すると，AM は RM にワーカコンテナを要求し，同様にリソースがスケジューリングされる．AM はこのワーカを用いることで処理を分散する．

2.1　チェックポイント／リストア (C/R)

C/R は，ある実行中のプロセスツリーの状態をイメージとして直列化し，後でそのイメージ

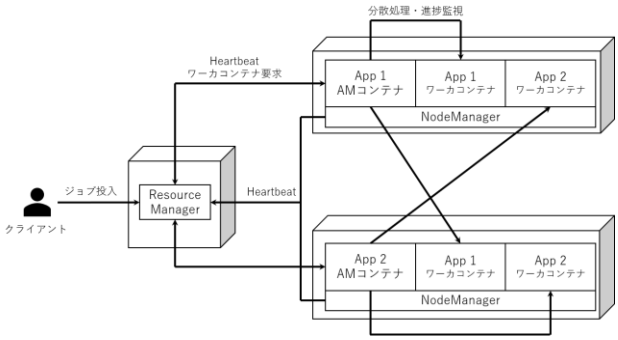

図１．YARN クラスタのアーキテクチャ

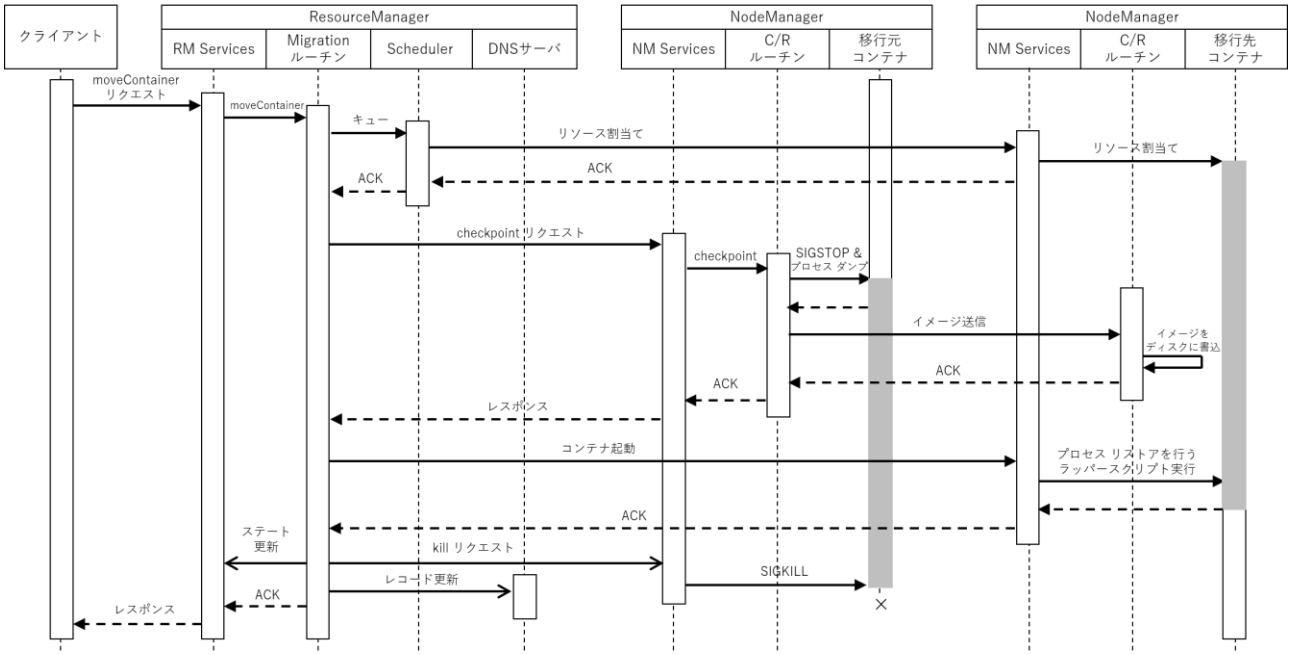

図２．YARN クラスタにおけるコンテナ ライブ マイグレーションのシーケンス

を元にプロセスツリーを再構築し実行を再開することを可能にする技術である．C/R の実装例には様々なものがあるが，今回は実装のシンプルな CRIU [6] を利用する．

３　提案アーキテクチャ

Hadoop YARN においてライブマイグレーション処理を行うシーケンスは図２のようになる(コンテナの実行仕様が網掛けになっているところは，リソースが確保されているがプロセスは実行中でないことを表す)．大まかな流れは以下のようになる．

1. クライアントは RM に対象のコンテナと移行先のノードを与える．
2. RM は対象コンテナと同等のスペックを持つ新しいコンテナを移行先ノードに確保するようスケジューリングする．
3. 対象コンテナのチェックポイント処理を行う．移行元ノードの NM は，対象のコンテナを凍結し，CRIU によりコンテナ内のプロセスをダンプし，生成されたイメージを移行先ノードに転送する．
4. 対象コンテナを移行先ノードで再開する．RM は（通常の AM / ワーカ コンテナを起動するときと同じように）移行されたコンテナを起動するが，そのコマンドに CRIU のリストアを設定する．このとき，Kernel namespace の機能を利用することにより実行コンテキストの衝突を回避する．
5. 各種ステートを更新する．

移行先のコンテナの生成はスケジューラを介して行われる．これは，クラスタリソース管理の一貫性を保つために重要である．

また，この機能は透過的である：つまり，対象コンテナに特別な要件を必要とせず，アプリケーションのコードを変更しなくてよい．

４　まとめと今後の課題

Hadoop version 3.1.0 のコードをベースに，上述のアーキテクチャに基づいてライブマイグレーション機能を実装している．

本稿執筆時点では，ディスク I/O を含む単純な処理を行うコンテナのライブマイグレーションを可能といている．引き続き，より本格的なアプリケーションに適用し実装・評価を行う．

参考文献

[1] Vinod Kumar Vavilapalli, et al. 2013. Apache Hadoop YARN: yet another resource negotiator. In Proceedings of the 4th annual Symposium on Cloud Computing (SOCC '13)
[2] Apache Hadoop, https://hadoop.apache.org/
[3] Karanasos K., et al. 2018. Advancements in YARN Resource Manager. In Encyclopedia of Big Data Technologies. Springer
[4] Kubernetes, https://kubernetes.io/
[5] P. U-Chupala, et al. 2017. Container Rebalancing: Towards Proactive Linux Containers Placement Optimization in a Data Center. IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)
[6] CRIU, https://criu.org/