1A-06

コンテナ型仮想化環境向き負荷予測システム「Tetris」の開発

叶　如絵†　　田胡　和哉‡

東京工科大学大学院バイオ・情報メディア研究科コンピュータサイエンス学部†

東京工科大学コンピュータサイエンス学部‡

1 背景

近年，サービスやデータをインターネット経由で利用するクラウドコンピューティングに注目が集まっている．様々な企業・団体がクラウドサービスを提供し，大学向けハイブリッドクラウドにも導入されている[1].

本稿では，コンテナ型仮想化環境における Kubernetes のスケジューラについて考察し，コンテナの最適な実行を実現する予測・判断システムについて提案する.

2 コンテナ型仮想化技術

2.1 Docker

Docker は，Docker 社が開発しているオープンソースのコンテナ型仮想化ソフトウェアである[2]．Docker では，ハイパーバイザ型仮想化と比べてハードウェア資源の消費や性能劣化が小さいだけではなく，コンテナ起動時には OS はすでに起動しており，プロセスの起動のみを行うため，コンテナの起動・終了が非常に高速という特徴がある．また，構築環境をパッケージングして，他の環境ですぐに使えるという高いポータビリティを実現している.

2.2 Kubernetes

Kubernetes は，Docker コンテナによるクラスタリングを行うオープンソースプロジェクトである[3]．Docker をベースとした PaaS 環境を実現している．Docker イメージを利用することで，実行環境のカスタマイズが容易になり，さらに完成したイメージを PaaS 環境から持ちだして，他の Docker ノードでも利用できるようになるという特徴がある．ノードやネットワークなど全体を管理する Kubernetes Master によって，管理されるコンテナをまとめた minion というノードがある．minion 上にグループ化したコンテナを Pod という単位で構築・管理している．どのノードのどの Pod にコンテナの配置するのか決定することはスケジューラが行う.

3負荷予測システム - Tetris

3.1 アプローチ

Kubernetes のスケジューラはプラグイン方式で構成される．標準的な実装は，コンテナを順番に振り分ける単純なラウンドロビン方式である．しかし，現実には，各ノードのスペックが異なり，コンテナの負荷もそれぞれ異なる．この現状をそのまま放置すると，特定ノードに負荷が集中してしまう問題が生ずる.

ここでは，ノードのリソースを考慮すると共に，コンテナの負荷パラメータと事前に計測した Docker イメージのプロパティを加えて，コンテナが最適な実行先を自動的に判断できる方法を考える．これで，ノード間の負荷を最適なバランスで分散することが実現できる．これらの機構を持った負荷予測システムを“Tetris”と呼称する.

3.2 アーキテクチャ

Tetris のアーキテクチャを図 1 に示す．本システムは，コンテナイメージをモデル化する“Molder”，リアルタイムのサーバリソース状況を取得する“Monitor”とコンテナをコントロールする“Container Controller”3 つコンポーネントから構成される.

Tetris は，主に次に示す機能によって構成される.

- モデリング機能
  事前にコンテナイメージのプロパティを計測して，各自特性を抽出しモデル化する
- モニタリング機能
  稼働中ノードのリソース状況とコンテナの負荷パラメータを監視する
  テストノードのパフォーマンスを計測する
- コンテナコントロール機能
  コンテナの配置・グルーピングなどを行う

Tetris: A Container Based Load Prediction System for Virtualization Environments

† Ruhui YE
Bionics, Computer and media science, Entrepreneurship program, Tokyo University of Technology Graduate School.
‡ Kazuya TAGO
School of Computer Science,Tokyo University of Technology.

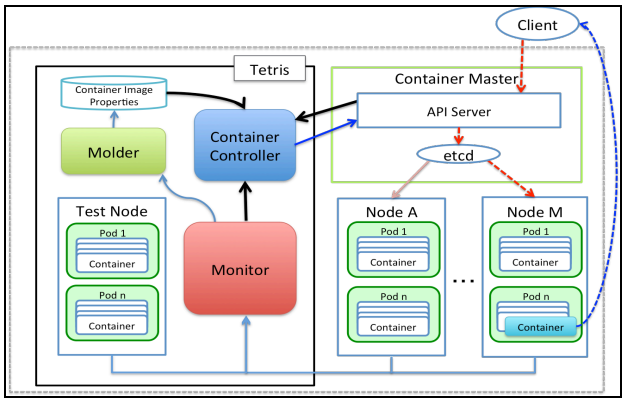

図 1 Tetris のアーキテクチャ

負荷パラメータがそれぞれ異なる，コンテナイメージの特性を把握するため，コンテナイメージを追加する前にモデル化する必要がある．Test Node はコンテナの負荷テストを行うサーバである．Test Node に測定するコンテナを配置し，負荷テストを行う．Monitor は Test Node のパフォーマンスとコンテナの負荷パラメータを計測して，モデリングするために分析データを提供する．Molder はこの分析データを基づいて，コンテナイメージの特性を抽出しモデル化する．作成したモデルは Container Image Properties に格納し，Container Controller の解析情報として提供する．

一方，稼働中ノードのリソース状況とコンテナの負荷パラメータを監視するにも Monitor の役割がある．主に“CPU”，“メモリ”，“ネットワーク”，“ディスク I/O”４つの負荷状況を監視する．

API Server は，クライアントからのコンテナ作成リクエストを受け取ると，Container Controller を呼び出し，コンテナの配置を依頼する．Container Controller はコンテナの配置要求を受け，Container Image Properties から当該コンテナのモデルを取り出し，負荷パラメータを確定する．更に，Molder からノートのリアルタイムリソース状況を加えて，コンテナを実行する最適な Pod を決定できる．API Server はコンテナの起動要求を受け，コンテナの配置・起動を行い，コンテナへの接続情報をクライアントに通知する．

4 実験と評価

4.1実験環境

実験環境では，負荷テストサーバ 1 台，Tetris 実装サーバ 1 台と Docker ホスト 2 台で設置した．負荷テストサーバは Docker ホストと同じく環境を構築した．Container Master Server と Tetris のコンポーネントはすべて Tetris 実装サーバに統合してある．Monitor は 2 つの部分から構成される．1 つは，Zabbix で Docker ホストのリソース状況を監視する部分であり，もう 1 つは，cAdvisor でコンテナの負荷パラメータを取得する部分となる．

今回の実験は，“Web アプリケーションサーバ”と“データベースサーバ”2 種類のコンテナで行った．サーバの負荷を飽和状態になる時に，ノードのパフォーマンスの変化を観察し，コンテナの負荷パラメータの特性を把握することを試みた．

4.2評価

Web アプリケーションサーバは，リクエスト数を増加とともに，メモリ使用率，ハードウェア割り込み回数も増加する． データベースサーバは，リクエスト数を増加とともに，CPU 使用率，ディスク I/O 量も増加する．

2 つサーバとも飽和状態になる直前に，コンテキストスイッチ回数が急激に上がったことが観測された．従って，コンテキストスイッチの変化率から，コンテナの負荷状況を測定することができると考えられる．

追加するコンテナの負荷パターンから，最適な実行先を検出し実行する．更に，このノードの負荷状況を予測することが可能となる．

5 おわりに

本論文では，各コンテナイメージの特性がそれぞれ異なることが例について示した．特定のパラメータからコンテナの負荷を予測できることが判明した．

現在の実装では，コンテナイメージのモデル化手法はまたシンプルなため，今後は，機械学習を用いてより多くのコンテナイメージをモデル化し，Container Image Properties を拡充して行く．

参考文献

[1]田中 遼，田胡 和哉：“コンテナ型仮想化機構を用いた大学向けハイブリットドクラウドの構築”，情報処理学会全国大会第 77 回全国大会(5D-06，2015)

[2]Docker. Inc：“Docker” https://www.docker.com/ (2016/01/05)

[3]Google. Inc：“Kubernetes” https://kubernetes.io/ (2016/01/05)