2ZF-01

プログラミング演習のためのコンテナ型の実行環境の設計と仮実装

清水赳[†1]　花川直己[†2]　富永浩之[†3]

香川大学[†1]　NTT コミュニケーションズ株式会社[†2]　香川大学[†3]

1. はじめに

本研究室では，ゲーム戦略を題材とするプログラミング演習を実施している．カードゲームを題材とする WinT とボードゲームを題材とする WinG の 2 つの演習を実施している[1]．各演習では，それぞれ独自の実行環境を含む支援システムを提供している．大会運営サーバは単独の物理環境で運営している．運営サーバでは，受講者が作成した戦略コードを実行する．予備大会の期間を設け，受講者が作成したコードの提出を何度でも受け付ける．得点や順位を公開し，戦略の再検討による改良とコードの継続的な修正を促進する．この期間は大量の提出が行われ，実行結果の提示に大きな遅延が発生する．

本研究では，プログラミング演習のための実行環境として，コンテナ型仮想化を用いた分散実行基盤を提案している．それぞれの運営サーバが用意している実行環境を集約し汎用化する．分散実行による応答品質の向上を目指す．本論では，実際の負荷状況を確認し，実行基盤の設計と仮実装を行い，必要な機能を検討する．

2. プログラミング演習のための実行環境の目的

プログラム実行環境の必要性を述べる．ゲーム戦略を題材とするプログラミング演習における運営サーバは，以下を要件とする[2].

- サーバ側の非公開データの保護
- 大量の実行やエラーへの対応
- 実行リソースの複製やアレンジの容易性
- プログラムの挙動や性能の比較検証
- 挙動の予測と，例外的な処理への対応

現状では，この要件に基づいた実行環境を演習のシステムごとに実現している．しかし，単独サーバによる処理では，大量実行や非公開データの保護への対応などにおいて，問題が生じる．実際の演習では，大量の戦略や高度な実装の戦略の提出が行われ，応答品質が著しく低下した．そのため，これらの問題を解決するプログラミング演習のための実行環境が必要である．

Design and Prototype of Execution Environment using Container Type Virtualization for Programming Exercises

[†1] Takeru SHIMZU, Kagawa University

[†2] Naoki HANAKAWA, NTT Communications Corporation

[†3] Hiroyuki TOMINAGA, Kagawa University

3. 現在のプログラム実行環境の負荷状況

大会運営サーバのコード実行による負荷状況について述べる．WinG に注目する．図 1 は，2017 年度実践の日ごとの対戦数である．この演習では，対戦を間引き，削減する仕組みが導入されている．序盤では，戦略の提出が少なく，多くても 1 日 1000 件程度の対戦しか行われない．一方，終盤では，提出数が増加し，1 日 10000 から 20000 件程度の対戦が行われる．特に，最終日は，23541 件の対戦があった．さらに，1 対戦に要する時間を調査した．WinG では，現在，対戦時間の計測機能は未実装である．そのため，確認対戦の実行時間を目安とした．確認対戦とは，教授側が用意した簡易な 1 戦略との対戦である．この実行時間は，ログによる．2017 年度の実践では，確認対戦は 738 件行われた．これらは，平均 7.30 秒，中央値 4.82 秒，標準偏差 10.14 秒の実行時間を要した．これらより，単純な逐次実行では，処理が大きく遅延する可能性が高い．

4. プログラム実行基盤の提案

仮想環境を用いた独立安全かつ効果的にコードを実行できる環境 Cachalot を提案している(図 2)[2]．これは，運営サーバが個々に用意している実行環境を集約した実行用のサンドボックスである．運営サーバは，この環境に対し，実行設定ファイルとソースコード等を送信するだけで，実行結果を取得できる．この環境では，内部でコンテナ型の仮想環境を準備し，コンテナ内部でコードを実行する．これにより，独立性と安全性を確保する．さらに，複数の安価な計算機による分散実行を行い高速な処理を目指す．特に，分散実行では，個々の実行時間に大きくばらつきのある戦略コードを効果的に振分ける．

5. 実行基盤の設計と機能

Cachalot は 4 つのモジュールで構成される．運営サーバ管理モジュールでは，提出処理を行う支援システムを管理する．実行コンテナ管理モジュールでは，実行中のコンテナの管理を行う．リソース割当状況や実行時間を管理する．処理ノードモジュールは，実行コンテナを処理する複数のノードを管理する．負荷分散モジュールはノードの状況から処理の振分け先を決定する．

本論では，これら 4 つのモジュールを提供する

Cachalot サーバと，実際の処理を行うノードに設置してコンテナの実行とリソース状況の収集を行う Cachalot エージェントを仮実装した．

6. Cachalot のエージェントの仮実装

エージェントは，コード実行とリソース状況の収集の機能を有する．実装は，Go 言語を用いている．ノードには，コンテナ実行環境 Docker を必要とする．エージェントは，単バイナリである．複数の計算機への設置を考慮した．

コードの実行機能は，Cachalot サーバからコードを含む規定の HTTP リクエストを受け取ると，Docker コンテナを立ち上げる．このコンテナの中で，実行設定ファイルに従いコードを実行する．実行結果を Cachalot サーバに返却する．

リソース状況の収集機能では，ノードの CPU やメモリの使用状況，ロードアベレージを収集する．結果は規定のエンドポイントでサーバ側に返却する．また，ヘルスチェックのためのエンドポイントもある．これはサーバ側のサービスディスカバリ(SD)と死活監視の機能で利用する．

7. Cachalot のサーバの仮実装と動作試験

サーバは，実行基盤の 4 つのモジュールを提供する．実装は Ruby on Rails と Sidekiq を用いている．サーバに対して，実行設定ファイルとソースコードを送信すると，ジョブが発生する．これを一旦，キューに格納する．格納されたジョブは並列に取り出される．ジョブは，登録されたノードのうち，どれか 1 つに，コードを含む HTTP リクエストを送信する．エージェント側の実行結果を受け取り，DB に格納する．エンドポイントを用意し，実行状況と結果を支援システムに返却できるようにする．また，動作状況を Web 上で確認できる GUI を提供する．

さらに， SD 機能と，ノードの死活監視機能を有する．SD 機能は，IP アドレスやホスト名で探索範囲を設定すると，一定時間ごとにノードの存在を探索する．エージェントから規定の応答があり，そのノードが未登録の場合，新規に振分先に登録する．死活監視機能では，一定時間ごとに，エージェントにアクセスする．規定の応答の有無で死活を確認する．エージェントから応答が無い場合，振分先から除外する．

図 3 は，動作中の Web GUI である．サーバに対して，支援システムからのリクエストを模した HTTP リクエストを送信している．ノードが登録され，状況が管理されている．提出されたコードは，エージェントに振分けられ実行される．実行の結果と状況も管理されている．

現在，振分先のノードは，振分可能なノードの中から，ランダムに選んでいる．

8. おわりに

ゲーム戦略を題材とするプログラミング演習を実施している．本研究では，プログラミング演習のための実行環境として，コンテナ型仮想化を用いた分散実行基盤を提案している．ゲーム戦略を題材とするプログラミング演習における運営サーバの要件を列挙した．現在の負荷状況を分析し，分散実行環境の必要性を議論した．これらを踏まえ，仮想環境を用いた，独立安全かつ効果的にコードを実行できる環境 Cachalot を提案した．本論では，必要な機能と設計を検討し仮実装を行い，動作実験を行った．

今後は，各運営サーバへの適用を目指す．実際の運用を行い，実行時間やリソースの使用状況などを収集する．収集した情報を用い，適切な振分処理を実装する．これらを行うことにより，受講者に対する応答品質の向上を目指す．

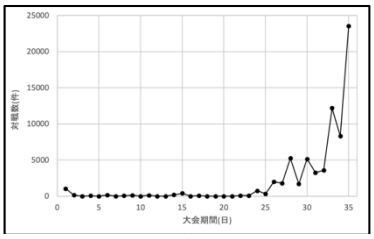

図 1 WinG の 2017 年度実践における対戦数

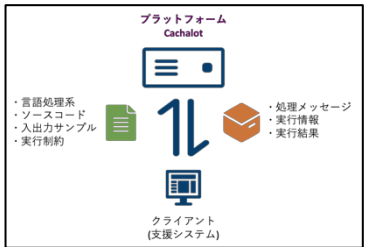

図 2 Cachalot の構成

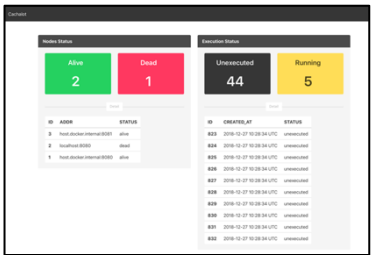

図 3 動作中の Cachalot サーバの GUI

参考文献

1) 花川直己, 玄馬史也, 富永浩之: ゲーム戦略を題材とする継続的なコード更新を促進する大会形式のプログラミング演習, 教育システム情報学会 第 42 回全国大会 講演論文集, Vol.42, No.I2-09,pp.317-318 (2017).

2) Naoki Hanakawa, Hiroyuki Tominaga: Container-Based Virtual Environment for Battle Execution of Round-robin in Applied Java Programming Exercise with Game Strategy and Contest Style, The International Journal of E-Learning and Educational Technologies in the Digital Media, Vol.3, No.2, pp. 83-98 (2017).