6ZA-03

ブロックチェーン技術を利用したセキュアなボランティアコンピューティングシミュレーターの実装

城島　翔太†　　金子晃介‡　　櫻井　幸一††

九州大学大学院システム情報科学府†　九州大学サイバーセキュリティセンター‡　九州大学大学院システム情報科学研究院††

1.概要

コンピュータで計算を行う際に，取り扱う問題の規模が大きい場合に，複数コンピュータで処理を分散させて解くことがある．これを分散コンピューティングといい，主に計算処理の速度の向上を目的に行われる場合が多い．本研究では，分散コンピューティングの一種であるボランティアコンピューティング(以下 VC)に着目する．VC とは有志者によって提供される計算資源で分散処理を行う手法である．VC は安価に実装できるというメリットがある一方で，信頼性の無い計算資源を利用する為に計算結果の信頼性が低い問題がある．そこで本発表では，この VC の問題を解決する仕組みの提案とその仕組みを検証するシミュレーターの実装方法について説明を行う．

2.導入

VC とは，ボランティアと呼ばれる有志者の提供する計算資源を利用して，分散処理を行う手法である．ボランティアが処理する問題は，サーバ側から提供される．提供される問題はプロジェクトによって異なる．プロジェクトによって，計算に貢献したボランティアにデジタルコンテンツ等の報酬が与えられるものがある．VC のプロジェクトに広く使われているミドルウェアとして BOINC が存在する[1]．BOINC を導入したプロジェクトでは，ボランティアはサーバに分割問題を要求し，解き終わると計算結果をサーバに返す仕組みとなる．VC は信頼性の担保されていない計算資源を用いて計算を行う為，誤った結果が返される可能性がある．その為，計算結果の信頼性を高める手法の提案が行われてきた．代表的な手法として Voting の説明を行う[2]．Voting とは，同じ問題を複数ノードに配布し，帰ってきた結果の中で最多の答えを採用する方法である．しかし，Voting は攻撃者が複数のアカウントを生成する Sybil Attack[3]に脆弱性を持つ．そこで信頼度という考え方を導入した手法[4]も提案されたが，脆弱性の問題は解決されていない[5]．

3.提案手法

計算結果の信頼性向上を目的に，以下の特徴を持つシステムの提案を行った[6]．このシステムを階級システムと呼ぶ事とする．

・ブロックチェーン導入と PoW による解の決定

Voting 等の既存研究では，ノード数により正答を決定する手法が採用されていた．その為，ノード数を増加させて有利に動く Sybil Attack 等の攻撃に脆弱性を持つ．そこで，ノード数によらない解の決定方法として，PoW を提案した．PoW とは，Proof of Work の略であり，ブロックチェーンシステムにおいてブロックを生成する際に，ある条件を満たすハッシュの計算を最初に解いたノードがブロック生成権を持つ手法である．VC のノード間でブロックチェーンを導入し，PoW で最初にハッシュ計算を解いたノードの解を採用することにより，ノード数によらず計算力によって解を決定する事が出来る．

・VC 参加者の階級分割

既存研究では，VC に参加するノードは問題を提供するプロジェクト側と，問題を処理するボランティア側しか役割が存在しなかった．本提案では，問題を処理するボランティア側のノードの階級を信頼度によって分ける事で，計算を担うノードの信頼性向上を目的としている．階級は以下の 3 つである．

計算ノード：プロジェクトノードによって提供される計算を担うノード．監査ノードによって結果の正誤が判断される．誤りと判断された場合は計算ノードから追放され待機ノードとなる．

監査ノード：計算ノードによって出された答えが正しいかどうか PoW を用いて決定する．計算ノードの答えと一致する場合は信頼度が向上し，計算ノードが欠けた場合，信頼度の高い監査ノードが計算ノードとなる．

待機ノード：どのプロジェクトにも属さずに待機を行うノード．プロジェクトの需要が発生すると監査ノードとなる．

上記の様に役割を分けることで，誤答を出すノードが計算に関与し難いシステムとしている

Implementation of Secure Volunteer Computing Simulator using Blockchain Technology
†Johjima Shota　Graduate School of Information Science and Electrical Engineering, Kyushu
‡Kaneko Kosuke　School of Science, Kyushu University
††Sakurai Kouichi　Faculty of Information Science and Electrical Engineering, Kyushu University
Security Laboratories, YY Corporation.

4. 実装内容

上記，階級システムによる VC の実装内容の説明を行なう．まず，VC に参加するノードは全て Node クラスを継承しており，IP アドレスや識別 ID 等の基本情報を有する．また，このシステムには問題を配布し，管理するプロジェクトノードが存在する．このプロジェクトノードはサーバ側となり，VC に参加するノードの階級管理と Send、Receive 関数により問題の送信，受信を行う．Distribute 関数で計算ノードには未解決の問題を送り，監査ノードには解決済み問題を一斉送信し，返信結果を見て信頼度を操作する．追放が起きた場合は補充を行う．このように，ノードの階級を操作する際に使用する操作を NodeManage クラスと Changenode 関数で行っている．本シミュレーターは一台で動作する想定であり，ノード管理はポインタにより行っている．その為，ポインタの処理が並列に行われた際，競合する可能性がある．その為，NodeManage クラスの関数は共通資源 Execheck を持ち，一つの関数しか実行できないよう管理している．問題の処理を行う計算ノードと監査ノードは，Receive 関数で問題が送信されるのを待ち，問題を受け取ると処理を行い，結果をプロジェクトノードに返すという処理を行う．また，プロジェクトノードから階級の変化があった際には Upgrade と Downgrade 関数で階級を変化させる．以下図 1 がノードの仕組みを表すクラス図である．

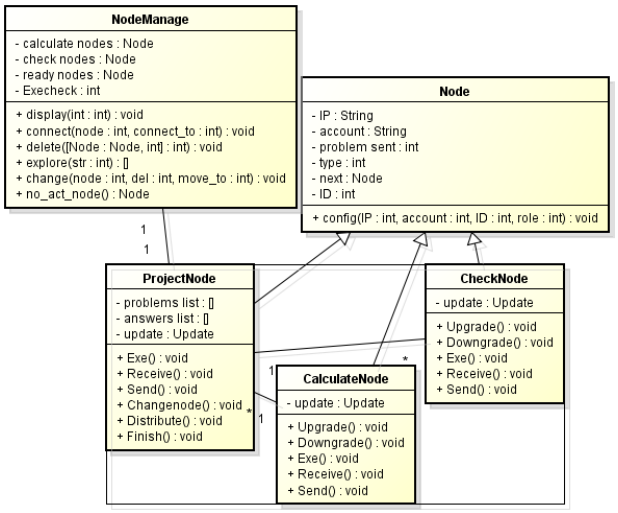

図 1. ノードのクラス図

また，上記の Node クラスを継承するクラスは update という変数を持つ．この update は後述の Update 型であり，問題を処理する際に実行される．この update は委譲関係となっており，クラスの階級が変化する事でそれに応じて update の中身も変化する． 前述の Node クラスは問題の送受信など，外部とのやり取りを行う関数であるのに対し，Update クラスは内部の処理を行うものである．このクラスは，計算ノードが監査ノードになることや，待機ノードが監査ノードになる場合の処理の違いに対応する為にこの形式にしている．計算ノードに対応する CalculateUpdate では，update 内で自身の計算の処理のフェーズを制御し，計算フェーズになると calculate で計算を行い，委譲元である Node クラスは getstate や getresult 関数を用いて処理フェーズや計算結果を取得する仕組みとなっている．監査ノードに対応する CheckUpdate は，CalculateUpdate と基本動作は同じであるが，calculate 関数内の処理が異なり，PoW によるハッシュ計算による計算の違いを想定した内容が追加されている．また，待機ノードに対応する ReadyUpdate であるが，このノードが内部で処理を行うことはないが，同一処理で行えるよう関数を持っている．以下が Update クラスとその継承元の仕組みとなっている．

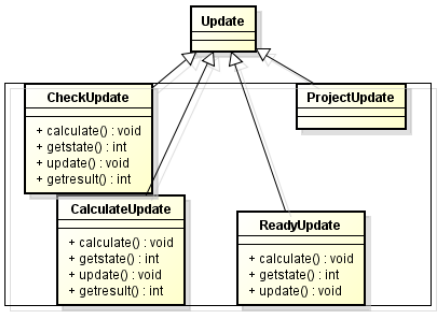

図 2. Update のクラス図

4. まとめ

本稿では，過去提案を行ったが実装はされていなかった階級システムによる VC のシミュレーターの実装を行い，システムと実装内容に関する説明を行なった．

謝辞 本研究は国立研究開発法人科学技術振興機構 (JST) 戦略的国際共同研究プログラム (SICORP) の支援並びに JSPS 科研費 JP18H03240 の助成を受けたものです，ここに感謝します．

[1]Anderson, David P. "Boinc: A system for public-ressource computing and storage." Grid Computing, 2004. Proceedings. Fifth

[2] L. F. G. Sarmenta, Volunteer Computing, Ph. D. thesis. Dept. of Electrical Engineering and Computer Science, MIT, Cambridge, MA, Dec, 2000.

[3]Douceur, John R. "The sybil attack." International workshop on peer-to-peer systems. Springer, Berlin, Heidelberg, 2002

[4]SARMENTA, Luis FG. Sabotage-tolerance mechanisms for volunteer computing systems. In: Cluster Computing and the Grid, 2001. Proceedings. First IEEE/ACM International Symposium on. IEEE, 2001. p. 337-346.

[5]渡邊寛, et al. “VC の妨害者対策における抜取検査併用の効果" 電子情報通信学会技術研究報告: 信学技報 111.408 (2012): 103-108.

[6]城島 翔太, 金子 晃介, 西田 裕輝, 堤 優亮, 櫻井 幸一, “ブロックチェーンを利用したセキュアな分散処理を実現するフレームワークの提案”, CSS2018, ホテルメトロポリタン長野 Oct. 22-25, 2018.