7H-05

エッジコンピューティング基盤 takuhai における レスポンス性を考慮したジョブ配置手法

三浦　和樹†　　廣津登志夫‡

法政大学情報科学部

1．序論

近年，インターネットに接続するデバイスが増加しており，クラウドコンピューティングで IoT デバイスを利用したデータの収集や処理を行う事例も増えてきている．これらの処理をクラウドで行う場合，クラウドへの負荷の集中，ネットワークの遅延が懸念される．この問題に対して IoT デバイスに近いネットワークにサーバを配置し，通信の遅延を抑制する技術のエッジコンピューティングが注目されている．

ここでは，典型的なエッジコンピューティングの環境としてプライベート環境である「ホームサーバ」，近隣にある計算機である「エッジノード」，ネットワーク的に距離のある計算機である「クラウドノード」からなる構成を考える．この構成では，一般的にホームサーバでプライベートなジョブを，クラウドノードでは AI の学習など負荷の大きなジョブ，エッジノードでは軽量な分析などのジョブが動くことが望ましい．

一般的に多数のノードとジョブを稼働させる際にはサービス運用フレームワークを用いる．そういった場合，ジョブを配置する一つの方法としては，ノードの負荷状態をみて配置を決めるという方法が考えられる．しかしノードの負荷を考慮した配置ではなく，個別の配置のポリシが必要なジョブがある．その中のレスポンスの求められるジョブはデバイスがデータを送ってレスポンスを返すまでの時間が短く済むことが望ましい．そのため，候補のノードの中から負荷・遅延のバランスで適切な配置をする必要がある．本研究はレスポンスの求められるジョブに関してレスポンス性を考慮した配置を行う．

2．関連研究

関連研究[1]ではノードの保持するデータ量を負荷の指標として，閾値を超えないように配置して負荷の分散を行っている．この配置では，デバイスから距離のあるノードに配置し，レスポンスを考慮した配置にならない場合がある．

A Job Scheduling Extension of Responsive Tasks for 'takuhai' Edge Computing Platform
†Kazuki Miura , Hosei University
‡Toshio Hirotsu , Hosei University

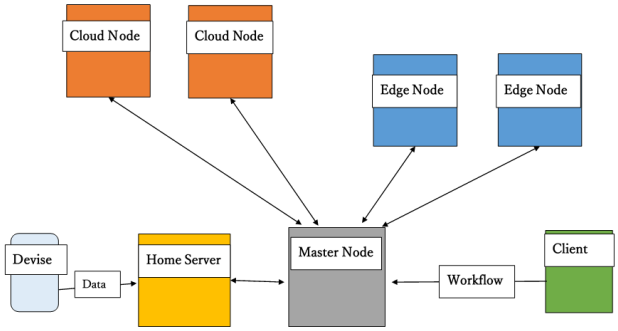

図 1 システムの構成

takuhai[2]はエッジコンピューティングの特徴を生かす配置のため作成されたワークフローエンジンである．ワークフローとはジョブの依存関係などを記述したもので，これを元に処理を行っている．ここでは依存関係の存在する複数のジョブをまとめてタスクと呼ぶ．takuhai ではジョブに対して，エッジに配置するなど強制的に指定することができる．この方法では，仮にあるノードにジョブが集中し，処理が遅くなっている場合でも同じノードに配置して，レスポンス時間を考慮した配置にならない場合がある．

3．設計

本研究のベースとなる takuhai において何らかのサービスを実行する際には，ワークフローをクライアントからマスタノードに登録する．ワークフローを構成する複数のジョブがジョブを実行するノードに配置された後，データがデバイスからホームサーバに送信されて，プライベートなジョブを行う．ジョブ実行の後，次のジョブの配置をマスターノードに問い合わせる．その宛先ノードに実行指令を出す．システムの構成を図 1 に示す．

takuhai で稼働するジョブについては，ジョブを分類し，プライベートジョブ，制約のないジョブ，レスポンスの求められるジョブの３つに分類して配置を考える．プライベートジョブはホームサーバに配置する．制約のないジョブについては使用可能な RAM が最大のノードに配置する．そして，レスポンスの求められるジョブやタスクに関しては，レスポンスにかかる時間が最短となるような配置を行う．レスポンスの求

められるタスクのレスポンスにかかる時間はタスクを構成する各ジョブの実行時間とその間のデータ転送の時間の合計で求めることができる．レスポンスの時間が最短となる配置を選択するためにはジョブのノードへの様々な配置の仕方におけるレスポンス時間を予測する必要がある．

レスポンスの求められるジョブの配置手法について説明する．ノードでの実行時間の予測は，そのノードにおいて同じジョブを実行した際の実行時間の実績から推定する．ノードが多数存在する場合を考え，ホームサーバとの通信時間で配置する候補のノードを絞ることにした．ジョブを配置する候補のノードの中にそれまでに同じジョブを実行したことがないノードがある場合，CPU の使用率が最低のノードを選択する．全てのジョブを候補のノード全てで一回実行した後は，過去の実行時間をもとにレスポンス時間を予測し，それが最小となるノードに配置する．

レスポンス時間を予測する配置を行う際には，過去の実行からジョブの実行時間とデータ転送の時間についてそれぞれ予測を立てる．ジョブの実行時間を予測するために，ジョブを実行する際に実行直前の CPU の使用率と処理に要した時間の組を記録する．そして，レスポンス時間の予測を用いた配置を行う際には，あるノードにおけるジョブの実行時間の予測は，そのノードのジョブ実行直前の CPU 使用率の近似する過去の実行時間とする．また，ジョブを実行するためのデータ転送の時間もデータの転送時に記録する．あるノード間のデータ転送時間の予測は，そのノード間での同じジョブの過去に計測したデータ転送時間の最も新しい記録とする．

4. 実装

本研究では，レスポンスを考慮した配置手法を takuhai のマスターノードに実装した．また，配置手法に用いる実行時間やデータ転送時間の計測などの機能をワーカーノードに実装した．

実行時間とデータ転送時間の計測と保存について以下に述べる．ジョブの実行時間とジョブに用いるデータ転送にかかる時間をマスターの保持するワーカーノードの情報に記録するようにした．ジョブ間のデータ転送の時間はワークフローの次のジョブに移る際に取得する．takuhai ではひとつのジョブが終わると，ジョブを終了したワーカーノードが次のジョブを行うノードをマスターノードに問い合わせ，返ってきた宛先に実行指令とデータを送信する．実行指令のリクエストに対し，レスポンスを返してリクエストとレスポンスの間の時間を計測する機能を加えた．データ転送の時間は，実行指令のリクエスト送信時の時刻とリクエストに対するレスポンスが届いた時刻との差とする．また，ジョブを行う際に実行時間や RAM の使用量，CPU 使用率を計測する．実行時間とデータ転送時間はまとめてマスターノードに送る．

4. 考察

レスポンスの求められるジョブ，制約のないジョブ，プライベートジョブを作成し，PC を複数台用意して実験を行った．本研究で追加した機能を用いた場合のレスポンスの求められるジョブの配置について説明する．クラウドノードとエッジノードの通信時間の差に比べて実行時間の差の少ないジョブであれば，通信時間を重視してエッジノードに配置することができた．また，通信時間の差が実行時間の差と比べて極めて短い場合には，実行時間を重視してクラウドノードに配置することができた．さらに，ほとんどの場合は近傍のエッジノードで実行するとレスポンス時間が短く済むが，負荷によってはクラウドや他のノードに配置するとレスポンス時間が短く済むようなジョブにもレスポンス時間が短くなるように柔軟な配置ができた．これは負荷を考慮した手法やノードの範囲の指定では対応できず，レスポンス性を考慮した配置としてメリットがあると考える．プライベートなジョブはホームサーバで実行され，制約のないジョブは使用可能な RAM の最大のノードで行われることが確認できた．

5. まとめ

本論分では takuhai に RAM や CPU などのノードのリソースやノード間の通信時間のデータをもとにレスポンス性を考慮した配置を行う機能を実装した．その考察として PC を複数台用意し，実際のジョブを用いた実験を行った．実験を通して，本研究の配置方法を用いることで，ジョブの実行時間と通信時間を考慮し，レスポンス性を重視した柔軟な配置が可能だと確認できた．

6. 参考文献

[1] R. Mogi, T. Nakayama and T. Asaka, ″Load Balancing Method for IoT Sensor System Using Multi-access Edge Computing, ″ 2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW), pp. 75-78, 2018

[2]佐藤琢斗, ”エッジコンピューティングのための軽量なワークフローエンジンの設計と実装”, 第 82 回全国大会講演論文集 , pp. 57-58, 2020