M-007

データセンタにおける Coflow スケジューラ Varys の改善手法の提案
Proposal of an Improvement Method of Varys, Coflow Scheduler in Data Center

中村 三郎† 相田 仁†
Saburo Nakamura Hitoshi Aida

1. はじめに

国際的にデジタルデータの量は飛躍的に増大しており，米国の調査機関によると 2021 年には約 40 兆 GB にも達すると予想されている．大量のデータを単一のコンピュータで処理することは難しく，複数のマシンで分散して処理を行っている．分散処理では処理能力を足し算的に増やすことができる一方，ノード間でデータを送受信する必要があり，通信による遅延が全体のパフォーマンスを下げる要因となっている．実際，近年の研究では分散処理を行う際に生じる中間データの通信遅延が全体の完結時間の 50%以上を占めることが明らかになっている．

分散処理の完結時間を決定するのは最も遅いフローの完結時間であり，すべてのフローの完結時間が重要なわけではない．したがってフローレベルで効率化を行う必要はなく，複数のフローをグループ化した Coflow という単位をスケジューリングして性能を上げる研究が行われている．Coflow スケジューラの性能評価は主に平均 Coflow 完結時間(平均 CCT)と制限時刻満足率の二つで行われる．

本研究では平均 CCT と制限時刻満足率の二つの向上が可能な Varys について，制限時刻満足率をさらに改善する手法を提案し，フローレベルシミュレータを用いて評価を行った．

2. Coflow スケジューラ Varys

Coflow とは 2012 年に考案された概念であり，分散処理での通信においてアプリケーションレベルで意味を持つ複数のフローのグループのことである[1]．分散処理全体の性能は一つ一つのフローで直接決まるわけではなく，それぞれのフローの中で最も遅いものや最も先のものによって決定される．したがって Coflow としてグループ化し，Coflow 内と Coflow 間のスケジューリングとして分けることで，複雑さを軽減するとともにアプリケーションレベルで意味のあるスケジューリングが可能となる．

Coflow スケジューラの評価は主に平均 CCT と制限時刻満足率の２つの観点から行われる．Coflow 考案後は平均 CCT の向上を目指す研究が主流であった．これは制限時刻が全てに設定されているわけではないなどの理由から平均 CCT の向上が優先されたと考えられる．その中でも平均 CCT だけでなく制限時刻満足率の向上もできる Coflow スケジューラとして Varys がある[2]．本研究では主に制限時刻満足率の向上に注目しているため，制限時刻満足率の向上ができる Varys をベースとしつつ更なる性能向上を目指した．

Varys は 2014 年に発表された Coflow スケジューラである[3]．平均 CCT の向上か制限時刻満足率の向上を目的としてスケジューリングを行うことができる．

平均 CCT の向上は Smallest-Effective-Bottleneck-First (SEBF)と呼ばれるヒューリスティックな優先度決定アルゴリズムと Minimum-Allocation-for-Desired-Duration (MADD)と呼ばれる割り当てアルゴリズムの二つが主な要因となっている．SEBF とは Shortest-Coflow-First という最も短い Coflow を先に処理するアルゴリズムを発展させたものである．ネットワーク状況を考慮することで Coflow の終了時刻を予測して優先度を決定している．一方 MADD とはその名の通り，必要最低限の割り当てだけを行うということである．最遅のフロー以外はどんなに早く終了しても意味がないため，最遅フローと同じ時間に終わるような帯域しか割り当てないということである．

制限時刻満足率の向上は主にアドミッション制御によって実現している．SEBF と同様に終了時刻を予想することで制限時刻に間に合うかどうかを判断し，間に合わなければ処理を一切行わない．これによりリソースを他の Coflow に集中させて制限時刻満足率の向上を果たしている．リソースの割り当ては平均 CCT モードと同じ MADD を用いている．

このように Varys は平均 CCT の向上と制限時刻満足率の向上のどちらかを実現している．しかし，後者は簡単なアドミッション制御のみしかしておらず，優先度の決定などでリソースを最適に使うようには設計されていない．

3. 提案手法

本研究では Varys の制限時刻満足率を改善させるべく，二つの変更を加えたスケジューラを提案する．

一つ目はアドミッション制御をより柔軟に行うということである．従来の Varys では Coflow の到着時に使えるリソースをすべて使った場合の最小完結時刻を予測し，それが制限時刻を下回るものだけを許可している．この許可と不許可は一度決定しすると変更されない．しかし，変更しないことで不利益が生じる場合がある．そこで今回提案するのは，制限時刻を超える Coflow を停止させてリソースを解放させるようにするということである．停止のタイミングは間に合わないことが予想できた時や時刻が制限時刻を過ぎた時などが考えられるが，今回は後者で停止するようにした．

二つ目は最初に割り当てた帯域を終了まで堅持するようにするということである．従来の Varys では Coflow 到着時に割り当てた帯域を無制限に与え続けるのではなく，一定時間が経過した場合に割り当ての量を減らすようになっている．これは一つの Coflow にリソースが集中してしまうよ

† 東京大学大学院工学系研究科
School of Engineering, The University of Tokyo

うな事態を防いでいる．どちらかというと平均 CCT の向上のためであると考えられるが，制限時刻に間に合うかどうかの予測の精度を落とすことになるため，最初に割り当てた帯域を終了まで保証するようにした．

4. 評価

4.1 評価方法

本研究ではフローレベルシミュレータの CoflowSim[4]を用いて評価した．用いたデータセットは Facebook で 2010 年ごろに実行された Hive/MapReduce のログをもとに作られたベンチマークである．3000 台のマシンが 150 のラックに入っており 10:1 のオーバーサブスクリプションである．合計の帯域は実際には 300Gbps であるが 100Gbps にスケールダウンしている．Coflow は 526 個であり一つのラックの中だけで完結しないように調整されている．また制限時刻については全ての帯域を利用できると仮定した際の予想最小 CCT に Uniform(1,2)を掛けたものを設定している．なお制限時刻の設定はすべての手法で同じである．

4.2 シミュレーション結果

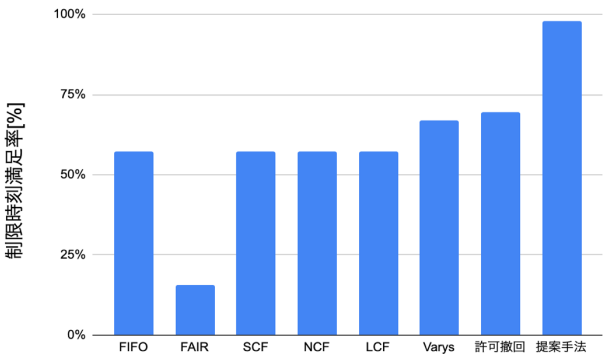

図 1. 各手法の制限時刻満足率の比較

図 1 は各手法での制限時刻満足率を示したものである．なお SCF, NCF, LCF は{Shortest or Narrowest or Lightest}-Coflow-First を示し，許可撤回は今回提案したもののうち許可を撤回するという提案のみ実装したものである．なお，Varys，許可撤回，提案手法では Coflow の到着時に拒否したものは割合の計算に入れていない．許可撤回を含む二つでは一度許可したものを撤回しているが，これは割合の計算に含んでいる．つまり処理を一度でも開始した中での満足率となっている．なお，許可撤回では 52，提案手法では 10 の Coflow を処理開始後に撤回している．

図 1 より提案手法では制限時刻満足率が大幅に改善しており，100%近くになっていることが分かる．また許可撤回についても Varys と比べてやや改善していることがわかる．しかし提案手法と比べると満足率は低く，許可撤回の効果は限定的と考えられる．

図 2 は制限時刻を満たした Coflow の数を比較したものである．制限時刻ありの Coflow スケジューリングでは割合で性能を評価するのが一般的ではあるが，それだけでは不十分であると考えて数での比較を行った．

図 1 と比較をすると Varys と許可撤回の手法において，SCF などの他の手法よりも満足数が低くなっていることが

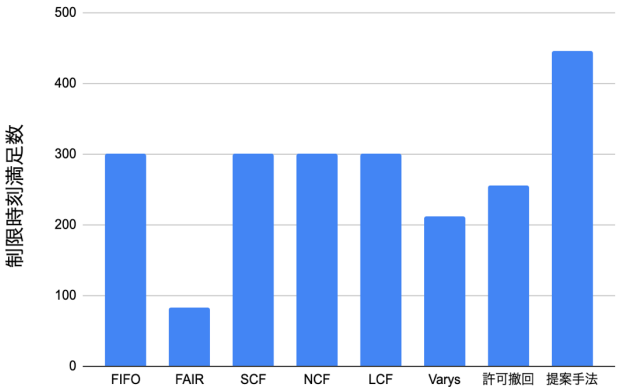

図 2. 各手法の制限時刻満足数の比較

分かる．つまり，従来の手法である Varys は割合の観点からは他の手法より優位であるが，絶対数では逆に劣るということが分かった．また許可撤回では Varys と比較した時に，割合に関しては改善の度合いは小さかったが，数に関しては大きく改善していることが分かる．これらの結果からも割合だけでの評価は不十分であり，数での議論が必要であると考えられる．一方，提案手法に関しては割合，数，どちらの観点からも従来手法よりも大きく優れていることが分かった．

5. おわりに

本研究では Coflow スケジューラ Varys について制限時刻満足率を改善させる新手法の提案を行い，フローレベルシミュレータで評価した．提案手法では制限時刻を満たす Coflow の割合と数の両方で優れた結果を示すことが分かった．また制限時刻満足数での比較も行なったことで，従来の割合のみでの評価は不十分であるということも分かった．

今後の課題としては次の三点が挙げられる．一つ目は提案手法のさらなる改善である．今回は許可を取り消せるようにしたが，そのタイミングを前倒すことや，逆に拒否したものを状況の変化に応じて許可に戻す改善案などが考えられる．また基本は MADD としつつもリソースに余裕のある場合には過剰に割り当てることで，後から到着するかもしれない Coflow にリソースを準備しておくという改善方法も考えられる．二つ目は制限時刻満足率以外の要素についても考える必要があるか検討することである．例えば，大小異なる Coflow を等価としてカウントして良いかや，リソースの占有の是非などについては慎重に検討すべきだと考えている．最後はより多面的に評価すべきということである．今回は Facebook のログをもとにしたデータだけで評価したが，Coflow のサイズや長さなどを変えることでより多くの要素から評価したいと考えている．

参考文献

[1] M. Chowdhury and I. Stoica, “Coflow: a networking abstraction for cluster applications.” in HotNets, pp.31–36(2012).
[2] S. Wang, J. Zhang, T. Huang, J. Liu, T. Pan, and Y. Liu, “A survey of coflow scheduling schemes for data center networks,” IEEE Communications Magazine, vol. 56, no. 6, pp. 179–185, (2018).
[3] M. Chowdhury, Y. Zhong, and I. Stoica, “Efficient coflow scheduling with varys,” in ACM SIGCOMM Computer Communication Review, vol. 44, no. 4. ACM, pp.443–454(2014).
[4] M. Chowdhury, “Flow-level simulator for coflow scheduling used in varys,” https://github.com/coflow/coflowsim.