TSUBAME 共同利用 平成 27 年度 学術利用 成果報告書

利用課題名 Lustre における並列入出力の高速化に関する研究
英文：Study of High Performance Parallel I/O on a Lustre File System

堀 敦史
Atsushi Hori

国立研究開発法人 理化学研究所 計算科学研究機構
RIKEN AICS
URL

邦文抄録（300 字程度）

大規模データを扱う並列計算における集団型 MPI-IO の高速化は今後益々重要な課題になってきている。代表的な MPI-IO 実装である ROMIO は様々な並列ファイルシステムをサポートしており、我々は ROMIO の実装内部の更なる高速化を図るために EARTH（Effective Aggregation Rounds with THrottling）と名付けた改変実装を提案している。本課題では EARTH を TSUBAME2.5 上に実装し、Lustre への集団型 MPI-IO に関して HPIO ベンチマークによる評価を 64 ノードを用いて 768 プロセスにより行ったところ、EARTH による性能向上の効果が見られるケースもあった。

英文抄録（100 words 程度）

Improving collective MPI-IO performance is one of the key issues for high performance data intensive computing. ROMIO is a well-known MPI-IO implementation and it is available many kinds of parallel file systems. We focus on further performance improvements of collective MPI-IO by implementing I/O request throttling scheme with stepwise data exchanges in our optimization framework named EARTH (Effective Aggregation Rounds with Throttling). We have implemented the EARTH optimization on the TSUBAME2.5 to examine its effectiveness on MPI-IO using Lustre. We have found a sort of improvements by the EARTH using 768 processes on 64 computing nodes.

Keywords: MPI-IO, ROMIO, two-phase I/O, aggregator, EARTH, throttling, stepwise data exchange

背景と目的

並列計算機の規模が増すにつれて並列計算により扱われるデータ量の増加が無視できなくなり、並列計算性能の向上におけるボトルネックの一つとなっている。並列計算における標準的な通信インタフェースである MPI では、並列 I/O を含む I/O インタフェースとして MPI-IO が定められており、代表的な MPI-IO 実装として ROMIO が広く利用されている。MPI-IO における集団型 I/O 機能により並列ファイルシステムを用いて高速な並列 I/O が利用できる。この中で ROMIO は Two-Phase I/O（以下、TP-IO）と呼ばれる高速化実装が用いられている。TP-IO は不連続なアクセスに対する集団型 I/O において、I/O に関与するプロセス間で通信を行うことでアクセスパターンに合わせたデータレイアウトの並べ替えを行い、連続なデータレイアウトでファイルシステムに対して I/O 操作を行うことで高速化を実現している。しかしながら、近年の計算機のノード数やノード内の CPU コア数の増加などに伴い、TP-IO による並列 I/O の性能がスケールしにくくなってきている。その大きな要因としてプロセス数増加に伴う並列ファイルシステムへの過剰な I/O 要求の発行とプロセス間データ通信のコスト増が挙げられる。

Lustre のような並列ファイルシステムでは、複数のストレージデバイスを束ねて大容量・高スループットを備えたファイルシステムを提供しており、クライアントからの I/O 要求がターゲットとなるストレージデバイスを管理する I/O サーバに送られて全体で高速な I/O を実現している。しかしながら一度にたくさんの I/O 要求が発行されると、I/O サーバ側の処理能力を超えてしまい、I/O 処理が待たされるクライアントが出てくる。また I/O サーバ側も多くの I/O 要求を同時に処理するために高負荷の状態となり、またターゲットとなるストレージデバイスへの I/O 処理も混雑してしまう。その結果として I/O 性能の低下を招いてしまう。

我々は以上の問題を解決するために EARTH(Effective Aggregation Rounds with Throttling)と名付けた高速化実装を提案している。EARTH は ROMIO における同時に発行するファイル I/O の数を

Throttling 機構により制限することで、上記で述べた過剰な I/O 要求による性能低下を回避し、かつ Throttling 機構に合わせた段階的なデータ通信を行うことで更なる性能向上を狙っている。EARTH は既に京コンピュータでの実装・評価を進めていたが、他の計算機システムでの有効性を確認する必要があり、その一環として TSUBAME2.5 を用いて InfiiniBand を用いた PC クラスタ環境での有効性を検証した。

概要

メニーコア混在型並列計算機用基盤システムソフトウェアの研究の一つとして、MPI-IO 実装である ROMIO における高速化を目指した最適化の実装と検証を進めている。特に集団型 I/O と呼ばれる入出力パターンにおける最適化である TP-IO の実装に関して、Lustre のような並列ファイルシステムを用いた大規模環境向けに最適化された実装を目指している。集団型 MPI-IO による Lustre への書込み処理における TP-IO 内部の処理の流れを図 1 に示す。この図では 4 プロセスが全て I/O 処理を行うアグリゲータとして機能している例を示しており，Lustre の各 Object Storage Target（OST）への書込みを行う前にデータの並べ替えを行った後に書き込みを行っている。このように TP-IO はデータ通信とファイル I/O を組み合わせて高速化を図っている。一方で，近年の CPU のコア数増加やノード内に複数の CPU ソケットが配置されていることに伴い、TP-IO においてアグリゲータの数も多くなり，Lustre の各 OST や OST 群を管理する Object Storage Server(OSS)への負荷が増加し、その結果 I/O 性能が低下する問題がある。我々は EARTH

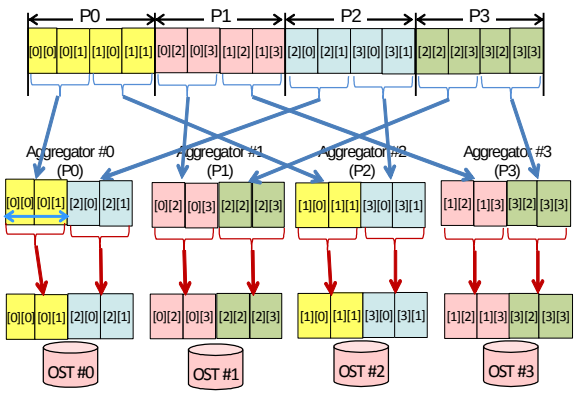

図 1： 集団型 MPI-IO による書き込み処理における TP-IO 内部の処理の流れの例

(Effective Aggregation Rounds with THrottling)と名付けた ROMIO に対する最適化実装を提案しており、ファイルシステムへの I/O 要求の Throttling と、Throttling と連動して動作する段階的通信によって高速化を図ることを目指している。

これらの機能は京コンピュータでも実装と評価試験を行っており、京での評価結果との比較検討を行うために、京とは異なる構成を持つ並列計算機システムとして、TSUBAME2.5 を利用して同様の評価を行うこととした。EARTH については、現在のところ論文執筆が完了していないため、実装方法について述べることを控えるが、上記のメカニズムによって、(1)Throttling によりファイルシステムへの I/O 要求の数を制限して高い I/O スループットを狙い、さらに、(2)I/O 要求発行に対する Throttling に伴い、早く I/O 要求発行を終了するプロセスから順に通信を開始できる段階的通信機構によって、一度に全プロセス間でデータ交換を行う際の通信混雑を避けることによる通信時間の短縮を狙っている。

結果および考察

実装した EARTH に対して HPIO ベンチマークによる評価試験を行った。HPIO は不連続アクセスパターンに対する集団型 MPI-IO の評価に用いられるベンチマークで、今回は MPI_File_write_all の性能を 64 台の Thin ノードを用いて 768 プロセスにより評価を行った。この評価では各ノードに 12 プロセスずつ起動しており、ランク配置はノード毎に詰めてゆく形式を使った。一方、使用した Lustre の OST の個数は計算ノードと同じ 64 個とした。またベンチマークではプロセス全体で約 137GB のファイルを生成するようにしており、7 回の I/O 性能計測値から最大値と最小値を除く値の平均値を記録した。Lustre へのアクセスではログインノード等と共通の InfiniBand ネットワークを経由することによる他のプログラムからの外乱による影響や、Lustre への高負荷状態が続くことによる他の利用者への影響をできる限り小さくする必要があったために、一度に多くの評価を行わず、日時を変えて評価を数回行い、その中の最大の平均値を採取して比較検討を行った。この計測での I/O 性能値を図 2 に示す。この評価では Throttling の I/O リクエスト数を 2 個に設定して動かしており、EARTH(wr:2)は書込みでの Throttling のみで、EARTH(wr,ex:2)は書込みで

の Throttling に加え Throttling に合わせた段階的通信を行っている。また比較のために現行 ROMIO（図中の orig）とアグリゲータ配置のみ最適化した実装（図中の agg_aw）での評価値も示している。この図で横軸はアグリゲータとなるプロセス数を示しており、256、384 および最大値である 768 プロセスの 3 点での性能値を示しているが、アグリゲータ数を増やしてゆく方が I/O 性能が高くなる傾向が見られた。

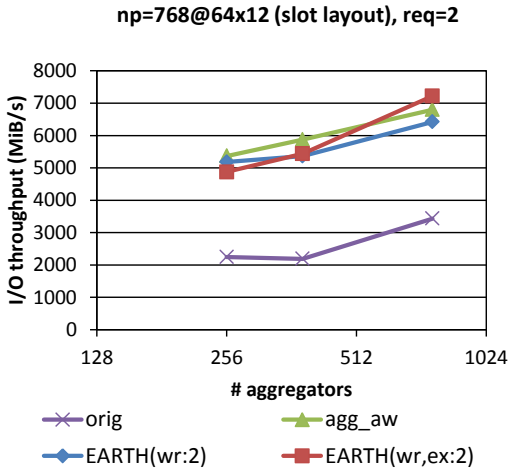

図 2： HPIO ベンチマークによる I/O 性能

また評価した 4 つのケースの中では、EARTH(wr,ex:2)が最も高い性能を示したが、アグリゲータ配置のみ最適化した実装との差はアグリゲータ数が 768 の場合でも 12%程度とそれほど大きな性能向上は得られていない。これは TP-IO 内の主要な 3 つの処理の中で通信処理が大部分を占めているためで、Throttling による書き込み性能の向上や段階的通信機構で得られる性能向上の効果が相対的に小さくなっているためである。今回試験を行ったアクセスパターンは非常に通信コストが大きくなるものを意図的に選んでいるが、アクセスパターンによっては通信コストが低減して、Throttling や段階的通信機構の有効性がもっと顕著になるケースもあるものと考えている。

次に、Throttling のリクエスト数を変えて性能への影響を調査した。その結果を図 3 に示す。Throttling のリクエスト数を 1、2、4 と増やしてゆくと、Throttling と段階的通信をリクエスト数が 2 で行った EARTH(wr,ex:2)のケースで全プロセスがアグリゲータになった時が最も高い性能を出していることが分かる。最適なリクエスト数を設定

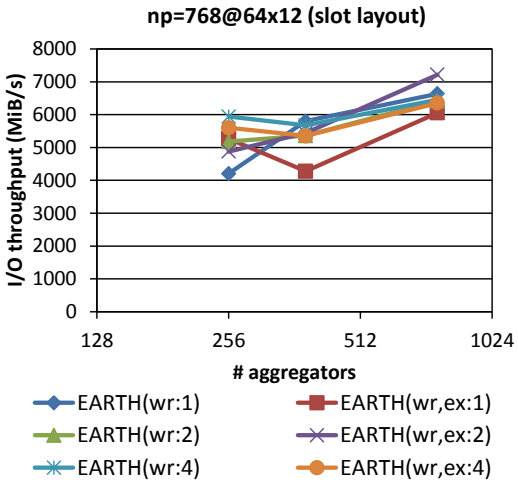

図 3： Throttling のリクエスト数を変えた際の I/O 性能の変化

すれば性能向上に繋がる可能性があることが分かったが、最適なリクエスト数を求める方法については今回の評価試験のデータからは分からなかった。これは今後の課題である。I/O のアクセスパターンやプロセス数、さらにノード毎に配置されるプロセスの数など多数の要因によって異なる振舞いをするものと思われるが、本研究とは別に京コンピュータでも同様の実装と評価試験を行っているが、こちらでは通信時間とファイル I/O の時間から、最適なリクエスト数を導きだせる可能性が見られるので、同様のアプローチで行えないかなど、今後検証を進めてゆくことも検討する予定である。

まとめ、今後の課題

我々は EARTH と名付けた ROMIO における高速化実装を進めるにあたり、ファイル I/O 処理への Throttling 機構の適用と、ファイル I/O に付随して行われるデータ通信に対する段階的送受信機構を提案している。この実装の有効性を検証するために、TSUBAME2.5 を利用させて頂き、HPIO ベンチマークを用いた性能評価を行った。その結果、Throttling および段階的送受信機構の有効性が見られる評価結果が得られたが、上記の 2 つの最適化における最適な I/O 要求数の求め方など、まだ分析が不十分な点が残っている。また本評価を行うにあたり、ファイル I/O 対象の Lustre ファイルシステムに関して、ユーザがログイン時やノード間通信などに利用されるネットワーク上に繋がっているため、性能評価の実施

日時を変えながら、その中で得られた性能の平均値を求めることで、できる限り上記のような性能のばらつきを小さく抑える配慮を行いつつ評価した。その結果、64 ノードで 768 プロセスを起動した場合における Throttling の効果を表すような評価結果が得られた。Throttling 機構での同時に発行する I/O リクエスト数の最適な値の算出方法については、HPIO ベンチマーク等による Lustre への大規模 I/O は高負荷な状態を招くので、他の利用者への影響等も配慮して長時間計測することは難しい状況であったため、評価に十分なデータを集めることが困難であった。よって今回は十分な検討が行えなかったが、これについては今後の課題として検討を続けてゆきたいと考えている。