九州大学学術情報リポジトリ
Kyushu University Institutional Repository

衝突削減のためのタスク配置最適化に関する研究

森江，善之
九州大学大学院システム情報科学府 | 九州システム情報技術研究所

末安，直樹
富士通株式会社ソフトウェア事業本部ミドルウェアコンポーネント事業部

松本，透
富士通株式会社ソフトウェア事業本部ミドルウェアコンポーネント事業部

南里，豪志
九州大学情報基盤センター

他

https://hdl.handle.net/2324/11878

出版情報：Symposium on Next Generation SuperComputing. 2007, pp.79-80, 2007-10
バージョン：
権利関係：

次世代スーパーコンピューティング・シンポジウム２００７

衝突削減のためのタスク配置最適化に関する研究

森江　善之†1†2, 末安　直樹†3, 松本　透†3, 南里　豪志†4, 石畑　宏明†5,

井上　弘士†6, 村上 和彰†6

†1 九州大学大学院システム情報科学府，†2 九州システム情報技術研究所，†3 富士通株式会社ソフトウェアコンポーネント事業部，†4 九州大学情報基盤センター，†5 富士通株式会社サーバシステム事業本部次世代 HPC 開発企画室，†6 九州大学大学院システム情報科学研究院

†1,†2y-morie@c.csce.kyushu-u.ac.jp，†3{sueyasu.naoki, matsumoto.tohru}@jp.fujitsu.com,

†4nanri @cc.kyushu-u.ac.jp，†5hiro.ishihata@jp.fujitsu.com，†6{inoue, murakami}@i.kyushu-u.ac.jp

概要：本稿では，通信の衝突を回避するためのタスク配置最適化の提案を行う．提案手法は通信パターンを抽出したプログラムを用いた性能評価実験において従来手法に対し最大68%の性能向上を示し，提案手法が有効であることを示した．

1　はじめに

近年の大規模並列計算機の計算ノード数の増加は著しい．計算ノード数が増加するとネットワークのためのハードウェアコストが莫大となる．このため，一般にメッシュやトーラス，ファットツリーなどのネットワークトポロジが用いられる．しかし，このようなネットワークトポロジでは，通信の衝突が頻発し，通信性能が悪化する．

著者らはネットワークトポロジに起因する通信の衝突を回避し，通信性能を向上させる研究を行っている[3]．本稿では，メッセージサイズの変化により，衝突を回避するタスク配置最適化の効果がどのように変化するかを調べた．

2　衝突削減のためのタスク配置最適化

本研究では，通信衝突による通信時間を加味することでより高性能な通信を可能とするタスク配置最適化の提案を行う．通信衝突は同時に同一リンクを同一方向に通るメッセージが複数存在する場合に発生する．このため，通信衝突を加味するには，どの通信が同時に実行されるかを知る必要がある．ここで，次の仮定を置くこととする．まず，メッセージサイズはすべて等しいとする．通信は完全に衝突するか全く衝突しないかのいずれかとなる．また，同時に実行される通信の集合を持つとする．この集合を Concurrent Communication Set とよび，CCSi で表わす．i は異なる CCS に属する通信の実行順序を表し，この実行順序を通信ステップとよぶ．同じ CCS に属す通信同士が同一リンクを同一方向に通過する時に衝突が発生するとする．

2.1　通信衝突を加味したタスク配置最適化

通信衝突による通信時間の増加を加味するため，式 1 に示す目的関数を用いる．この目的関数では，各通信ステップにおける各タスクの予想通信時間の最大値の総和を求める．これにより，各通信ステップにおける通信の衝突による通信時間の増加を加味する．

$$\sum_{t=0}^{N-1}\sum_{i=0}^{n-1} L(\pi(i), tork(i)) + (S(t,i,j) \times coll(t,\pi(i),tork(i))) / B(\pi(i), tork(i)) \quad (1)$$

t は通信ステップ，N は通信ステップの総数である．i はタスク，n はタスク総数である．$\pi(i)$ はタスク i が割り付けられている計算ノードを返す関数である．$tork(i)$ はタスク i の通信先のタスクを返す関数である．$L(p,q)$ と $B(p,q)$ はそれぞれ計算ノード p，q 間の通信遅延と通信帯域幅を返す関数である．$S(t,i,j)$ は通信ステップ t にお

けるタスク i からタスク j へのメッセージサイズを返す関数である．また，$coll(t,p,q)$ は通信ステップ t における計算ノード p，q 間の各経路で発生した通信の衝突回数+1 の最大値を返す関数である．

3　性能評価実験

3.1　実験概要

提案手法の有効性を示すため，カーネル部分の通信パターンを抽出したプログラムを作成して提案手法と従来手法によるタスク配置を適用した際のプログラムの実行性能の比較を行う．プログラムは recursive doubling，CG，umt2000 を用いた．

3.2　従来手法

T. Agarwal ら[1]や G. Bhanot ら[2]が提案している従来手法を比較対象とする．従来手法で提案している目的関数を式 2 に示す．

$$\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} C(i,j)\times H(\pi(i),\pi(j)) \quad (2)$$

i,j はタスク，n はタスク総数である．$\pi(i)$ はタスク i が割り付けられている計算ノードを返す関数である．$C(i,j)$ はタスク i からタスク j への通信量を返す関数である．また，$H(p,q)$ は計算ノード p から計算ノード q へのホップ数を返す関数である．

3.3　実験方法

提案手法と従来手法の目的関数の値を最小とするタスク配置を求める．この時，可能解すべての探索を行う．決定したタスク配置でプログラムを実行し，その合計所要時間を計測する．

実際に並列プログラムを実行する際，異なる CCS に属す通信が同時に実行される可能性がある．この時，異なる CCS の属する通信同士の衝突が発生しまい性能が悪化する可能性がある．このような状況を防ぐため，同期関数を挿入する．この時，通信ステップ毎に同期関数を挿入することになる．このため，1 回の同期関数にかかる時間と通信ステップ数の積が通信コストとして加わる．

3.3　実験結果

提案手法は従来手法に対し最大で 68%の性能向上を得た．図 1 から提案手法はメッセージサイズが大きくなると，性能向上が見込まれることが分かる．メッセージサイズが大きい場合には通信衝突の回避による性能向上が大きいと考えられる．メッセージサイズが小さい場合は，異なる CCS に属する通信が同時に実行されないよう挿入した同期関数によるコストの影響が大きいため，逆に性能が悪化している．同期関数のコストの低減が今後の課題である．

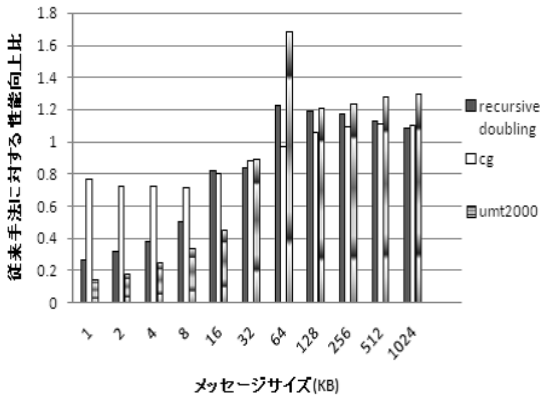

図 1　従来手法に対する性能向上比

謝辞

本研究は，「ペタスケール・システムインターコネクト技術の開発」プロジェクト（文部科学省「次世代 IT 基盤構築のための研究開発」の研究開発領域「将来のスーパー コンピューティングのための要素技術の研究開発」（平成 17～19 年度）の一つ）によるものである．

参考文献

[1]T. Agarwal, A.Sharma, L. V. Kale：Topology-aware task mapping for reducing communication contention on large parallel machines, IPDPS, pp.1-10, in 2006.

[2]G. Bhanot, A. Gara, P. Heidelberger, E. Lawless, J. Sexton, R. Walkup ： Optimizing task layout on the Blue Gene/L supercomputer, IBM J. Res.Dev. Vol.49, No. 2/3, pp.489-500, in 2005.

[3]森江 善之，末安 直樹，松本透，南里 豪志，石畑 宏明，井上 弘士，村上 和彰 ： 通信タイミングを考慮した衝突削減のための MPI ランク配置最適化技術，情報処理学会論文誌コンピューティングシステム(ACS19)，2007 年