2013 年 5 月 20 日 情報システム基盤学合同輪講発表資料

Process Mapping for MPI Collective Communications

著者： Jin Zhang, Jidong Zhai, Wenguang Chen, Weimin Zheng
出典： *Proceedings of the 15th International Euro-Par Conference on Parallel Processing, pp.81-92, 2009.*
発表者： 高性能コンピューティング学講座 本多・近藤研究室 1353002 大坂隼平

1 はじめに

複数のプロセス間での通信が必要な並列処理を行う際には、プロセス間の通信と実際のシステムが持つ通信路とをどうマッピングするかにより、性能に大きな影響がある。物理システムとアプリケーションを解析し、通信量の多いプロセスは通信性能が良い箇所に割り当て、通信量が少ないプロセスは通信性能が高くない箇所に割り当てることで、全体的に通信時間が短縮され性能の向上が見込めるようになる。既存の研究では自動的に最適なプロセスとプロセッサ間のマッピングを行う様々なアプローチが提案されている。しかし、これらはプロセス間の 1 対 1 通信のみを扱っている。一方で 1 対 1 通信とは違い、複数のプロセス間で通信を行う集団通信という通信方式がある。この集団通信に関しては未だに最適なマッピングアプローチが存在していない。

本論文では、並列計算機を利用する際に通信を記述する API の規格である MPI の集団通信関数においても最適なプロセスマッピングを可能とするアプローチとして Optimized Process Placement (OPP) を提案する。本アプローチは集団通信を一連の 1 対 1 通信に変換し、これに対して既存の 1 対 1 通信における最適なマッピング方式を適用することでプロセスマッピングを実現する。

2 従来のプロセスマッピングの手法

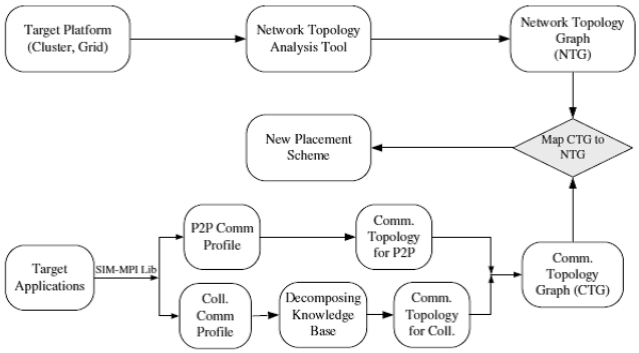


図 1: プロセスマッピングの手法

図 1 に従来のプロセスマッピングの手法の概要を示す。物理システムにおいてそれぞれのプロセッサ間のレイテンシやバンド幅を解析し、通信路の特徴をグラフに表す。このグラフを Network Topology Graph (NTG) と呼ぶ。アプリケーション側においても同様にそれぞれのアプリケーションにおけるメッセージの回数とメッセージの量を解析しグラフに表す。このグラフを Communication Topology Graph (CTG) と呼ぶ。CTG の解析結果から判明した通信量の多いプロセッサを NTG の解析結果から判明した通信性能の良いプロセッサに割り当てることで通信時間を短縮させる。

NTG は解析ツールを用いることで自動的に取得できる。CTG は 1 対 1 通信のアプリケーションの場合は MPIPP[1] 等の既存の手法を用いることで取得が可能であるが集団通信においては取得手段が存在しない。そこで、集団通信を一連の 1 対 1 通信に分割し、これを CTG に対応させる事で集団通信でもマッピングを可能にさせる。

3 MPI 集団通信における CTG の作成方法

3.1 Decomposition Knowledge Base

MPI 集団通信関数はライブラリ内で一連の 1 対 1 通信に分割される。しかし、この 1 対 1 通信への分割方法は単純ではない。MPI ライブラリが異なっていると実装の方法も異なっていたり、MPI ライブラリが同じであっても集団通信の実装に使用されるアルゴリズムによってはメッセージサイズやプロセスの数に依存する。これを解決するために、あらかじめ各集団通信関数で、使用される 1 対 1 への分割手法を記録しておく知識ベース Decomposing Knowledge Base (DKB) の作成を提案する。複数の MPI ライブラリを解析し、各 MPI ライブラリにおいて、メッセージサイズやプロセスの数によってどの分割アルゴリズムが使用できるかを示す分類表の作成を行う。

マッピングには、DKB を参照することで集団通信を実際の 1 対 1 通信に分割し、CTG を生成する。この方法により、最終的にアプリケーション全体の集団通信について 1 対 1 通信に分割された CTG を取得することができる。

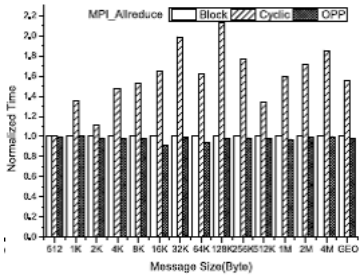
3.2 MPI Alltoall 関数の事例

集団通信関数の一つに MPI Alltoall 関数がある。この関数の実装の一つに Bruck アルゴリズムがあり、それを例に 1 対 1 通信への分割の事例を説明する。Bruck アルゴリズムは 8 つのプロセスで処理を行う場合 3 回のステッ

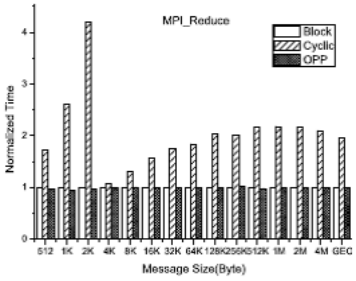
プで分割することができる。よって合計 24 個の 1 対 1 通信に分割でき、ここからメッセージ数と転送量を解析して CTG を得ることができる。図 2 はメッセージサイズが全ての要素を 10 バイトと仮定したときの、分割を行った後の CTG に対応する行列の出力結果である。この行列においては、80 バイトを転送するプロセス間を通信性能が良い通信路にマッピングをすることで性能向上が見込める。

P7	40	40	0	80	0	40	40	0
P6	40	0	80	0	40	40	0	40
P5	0	80	0	40	40	0	40	40
P4	80	0	40	40	0	40	40	0
P3	0	40	40	0	40	40	0	80
P2	40	40	0	40	40	0	80	0
P1	40	0	40	40	0	80	0	40
P0	0	40	40	0	80	0	40	40
	P0	P1	P2	P3	P4	P5	P6	P7

図 2: MPI Alltoall における分割結果



Allreduce (NP=64)



Reduce (NP=64)

図 3: IMB の評価結果の一例

4 性能評価

4.1 評価環境

評価は 4GB メモリを搭載する 2 ソケットを持つサーバを 1 ノードとする 16 ノードクラスタで行い、プロセッサは 1.6GHz Xenon dual core プロセッサを用いた。これによりクラスタのコア数は 64 コアとなることから、MPI のプロセス数 64 で評価を行った。評価には 2 つのベンチマーク Intel MPI Benchnark (IMB) と NAS Parallel Benchmark (NPB)、及び 3 つのアプリケーションを用いた。

4.2 IMB の評価結果

IMB はインテル社が開発したマイクロベンチマークで、MPI における通信性能を評価する。これを用いて全ての MPI 集団通信関数において最適なプロセスマッピングが可能であるかどうか調査する。提案手法である OPP と、プロセス番号順に各ノード内のプロセッサにまとめて割り当てる block 方式と、各ノードのプロセッサに 1 プロセスを割り当てたら、次のノードに割り当て、それを繰り返す cyclic 方式を比較評価する。その評価結果を図 3 に表す。評価結果から block 方式や cyclic 方式は集団通信関数によって差がある結果となったが今回の研究で提案したアプローチである OPP 方式においては全てにおいて最適なマッピングを見つけることができた。

4.3 NPB の評価結果

NPB は科学技術計算を行うプログラム一式で、並列コンピュータの実効性能を知る上で重要なベンチマークの一つである。今回は NPB に加えて 3 つのアプリケーションを用いてそれぞれのアプローチにおける評価を行った。その評価結果を表 1 に表す。評価結果から 1 対 1 通信のみのアプリケーションにおいては MPIPP と OPP が等しく良い性能を得ることができた。集団通信のみのアプリケーションにおいては OPP はどのアプリケーションにおいても block や cyclic の結果よりも 26%以上の性能向上を得られる結果となった。1 対 1 と集団通信両方含んでいるアプリケーションにおいては、MPIPP は集団通信を扱っていないため block や cyclic よりも性能が悪くなってしまっているが、OPP は性能向上が得られることがわかった。

表 1: NPB の評価結果

Name	Block(s)	Cyclic(s)	MPIPP(s)	OPP(s)	Speedup of OPP vs. Block
bt.C.64	95.47	103.76	90.79	90.66	1.05
cg.C.64	64.14	89.03	62.3	62.3	1.03
ep.C.64	11.12	11.12	11.12	11.09	1.00
lu.C.64	69.32	74.39	68.72	68.72	1.01
sp.C.64	143.97	151.40	132.12	132.12	1.09
mg.C.64	9.56	9.12	9.11	9.10	1.06
is.C.64	12.59	12.14	12.87	12.13	1.04
ft.C.64	31.52	23.20	N.A.	22.89	1.38
PAPSM.20	15.78	19.45	N.A.	12.54	1.26
GE.64	20.83	25.14	21.89	17.48	1.19
ASP.64	55.93	50.46	N.A.	50.16	1.11

5 まとめ

本研究では、集団通信関数でも最適なプロセスの割り当てを可能にするためのアプローチ OPP の設計、実装を行った。評価結果は実験を行った全てのアプリケーションにおいて従来の手法よりも通信速度が向上した。

参考文献

[1] Chen, H., Chen, W., Huang, J., Robert, B., Kuhn, H. “MPIPP: an automatic profile-guided parallel process placement toolset for SMP clusters and multiclusters.” proc. in ICS 2006, pp.353-360 (2006)