DCAST への新しい自己組織化機能の提案

The proposal of the new self-organization function for DCAST

中尾　昌広
Masahiro NAKAO

Abstract: "Dynamic Cluster Auto Setup Tool: DCAST" is a simple installation and administration software of calculation nodes in PC cluster. The feature of DCAST can design automatically the relation of the network mount between diskfull node and diskless node and divide a class in order to avoid a bottleneck. Therefore, DCAST's uers can also build PC cluster easily. In this paper, these functions shows that PC cluster construction DCAST's user's burden is reduced, and improvement in a performance of PC cluster.

1　はじめに

われわれは PC クラスタを構築，管理するツール Dynamic Cluster Auto Setup Tool（以下 DCAST）[1] の開発を行っている．DCAST は計算ノードのハードディスクの有無を判別する機能があり，DCAST を使用するユーザはディスクフルノードとディスクレスノードを同じ手順で構築することができる．これまでの DCAST は postmark[2] というベンチマークを用いてディスクフルノードのハードディスクの性能を調べる事により，ディスクフルノードとネットワークマウントの関係にあるディスクレスノードの割当数を決定していた．しかし，その手法では以下の点が問題となっている．

- postmark が出力する値の分散が大きい
- ベンチマークを行う時間が長い（1 ノード当たり約 30 秒）
- どのようなスペックのマシンで postmark を計測しても値がほとんど変わらない

そこで本研究では，連結されているスイッチ間に存在するボトルネックを避けるように計算ノードのグループ分けを行う新たな自己組織化機能を提案する．この機能により PC クラスタ構築者の負担を減らし，かつ PC クラスタの性能向上を目指す．

2　DCAST の概要

DCAST は適切に接続された PC ノードに対して，OS（Debian/GNU Linux）およびクラスタリング用ソフトウェアの自動インストール機能を提供する．PC クラスタの構築に際しては各ノードのスペックに応じたネットワークアーキテクチャ（論理な接続関係）を決定する必要があるが，DCAST では構築時に各ノードを調査して最適な構造を自動的に判断する．DCAST ではこの機能を自己組織化機能と定義している．また，DCAST はアップデートの際にはノード間の一貫性を保証するためにシステム全体の再インストールを行う．その際にノードの追加・変更等がなされていればアーキテクチャは再び変化する．このことより，設計・管理コストの大幅な削減や，遊休資源の容易な追加が可能になる．

3　新たな自己組織化機能の提案

3.1　大規模 PC クラスタシステム構築時の問題点

一般に大規模 PC クラスタシステムを構築する場合，スタックケーブルを用いてスイッチを多段結合し，一つのスイッチとして利用することが多い．その場合スイッチ内にボトルネックリンクが存在することになる．このような環境で PC クラスタ，特にディスクレスノードを構築する場合，ネットワークマウントの関係に注意をはらう必要がある．また，スタックケーブルを通して並列計算を行う場合，その部分がボトルネックになることも考えられる．

3.2　提案

前節で説明した問題点を克服するため，スイッチに存在するボトルネックを避けるように計算ノードのグループ分けを行うことを提案する．また，ネットワークマウントの関係もそのボトルネックを避けるように行う．この手法を用いる事により，PC クラスタ構築のネットワーク設計に要する負担を減らし，かつ PC クラスタの性能向上を目指す．

3.3　自己組織化機能の実現方法

Fig. 1 に示すような構成の PC クラスタの場合，スイッチの接続部にあるスタックケーブルが通信時のボトルネックになっていると考えられる．そこで，マスタノードから各計算ノードに対し，通信のバンド幅を測定する．得られた値によってクラス分けを行う．

Fig. 1 に示す構成と Table 1 に示すスペックを持つ PC クラスタにおいて，マスタノードから各計算ノード間の通

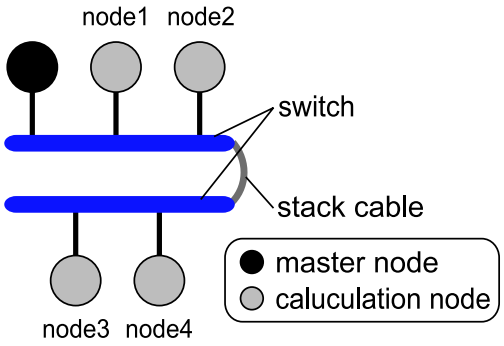

Fig. 1　PC クラスタの構成

Table 1　使用する PC のスペック

CPU	Pentium 4 2.4GHz
Memory	DDR-SDRAM 512MByte
Network	Fast Ethernet
OS	Debian GNU/Linux 3.0r3，kernel 2.4.27
Compiler	gcc-2.95，mpich-1.2.1
Switch	Cisco Systems Catalyst 3500 SERIES XL

信のバンド幅の計測を行った．計測には pingpong[3] というソフトウェアを用いた．pingpong のパラメータを調節することにより，256byte，512byte，768byte，1024byte の大きさのパケットを各 20 回ずつの通信を行う際のバンド幅を計測した．この値を用いた理由は，PC クラスタでよく用いられている MPI の通信プロトコルは TCP/IP であり，TCP/IP は 256byte から 1024byte の長さのパケットが使われるからである．ベンチマークの結果を Table 2 に示す．

この結果より，スタックケーブルを介した通信のバンド幅は介さない通信に比べ，半分程度しか性能が出ていないことがわかった．これにより，スイッチによるクラス分けが可能なことがわかった．なお，ベンチマーク測定に要した時間は 1 ノード当たり約 0.8 秒であった．

3.4　自己組織化機能の効果

提案する自己組織化機能を用いる事で性能が上がるかどうかを確かめるために，Fig. 1 と Table 1 に示す環境で姫野ベンチマーク [4] による計測を行った．同スイッチ

Table 2　pingpong ベンチマークの結果

通信経路	バンド幅
master → node1	6.09 Mbytes/sec
master → node2	6.09 Mbytes/sec
master → node3	3.67 Mbytes/sec
master → node4	3.68 Mbytes/sec

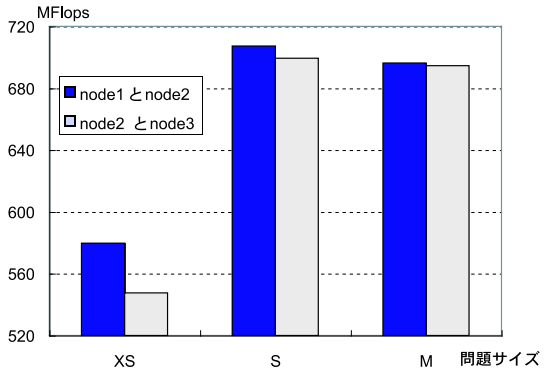

Fig. 2　姫野ベンチマークの結果

内にあるノード 1 とノード 2 を計算に使用した場合と，別スイッチに内にあるノード 2 とノード 3 を計算に使用した場合とを比較する．ベンチマークに用いた問題サイズは XS，S，M，分割数は（2,1,1），最適化オプションは O3 とした．ベンチマークの結果を Fig. 2 に示す．

この結果より提案する自己組織化機能に効果があることがわかった．問題サイズが小さい方が計測値の差が大きいのは、姫野ベンチマークは問題サイズが小さくなるほど単位時間当たりの通信回数が増えるからである．

4　考えられる問題点と今後の課題

今回提案した方法はスイッチが 2 つであると仮定しており，3 つ以上スイッチがある場合には対応していない．スイッチが 3 つあり，かつバンド幅を計測するマスタノードが真ん中のスイッチに接続されている場合，クラス分けができないと考えられる．解決方法として，マスタノードを必ず端のスイッチに接続するという制約をつけるか，もしくは DCAST 実行時にユーザがスイッチの数を指定することにより，まず二つにクラス分けを行った後，それぞれのクラスでもう一度クラス分けを行うという方法が考えられる．

今後は，今回説明した自己組織化機能の実装を DCAST に行い，動作実験を行うことで問題点を洗い出すことが挙げられる．

参考文献

1) DCAST ホームページ，http://mikilab.doshisha.ac.jp/dcast/

2) PostMark: A New File System Benchmark, http://www.netapp.com/tech_library/3022.html

3) Performance Modeling and Characterization (PMaC)，http://www.sdsc.edu/pmac/

4) 姫野ベンチマークホームページ，http://accc.riken.jp/HPC/HimenoBMT/