2004－DPS－116　(1)
2004／1／29

MPI/SPx におけるクラスタ間データ交換の実現

村 山 和 宏† 落 合 真 一†

近年，電話の交換機システムやレーダのデータ処理システムといった社会インフラを支えるシステムなどに HPC クラスタを採用する動きがある．このようなミッションクリティカルなシステムで HPC クラスタを動作させるために，我々は，高信頼化と高性能並列処理の両方を実現する MPI ミドルウェア：MPI/SP（MPI for Signal Processing) を開発した．そして，今回， MPI/SP クラスタを組み合わせてさらに大規模な演算を実現するために， MPI/SP を拡張したミドルウェア：MPI/SPx（MPI/SP extended）の設計を行なった． MPI/SPx では，複数の MPI アプリケーションの連携，データのパイプライン転送，クラスタ構成変更の隠蔽，システムの高信頼化を実現している．

The design of data exchange methods between clusters on MPI/SPx

Kazuhiro Murayama† and Shinichi Ochiai†

Recently, high performance computing (HPC) clusters can be applied to mission-critical systems, such as telecommunication systems and RADAR information processing systems. To use HPC clusters for those systems, we have developed MPI/SP (MPI for Signal Processing), which has achieved two functionalities; high performance computation and high availability. And now, to achieve more complexed and larger computation by connecting several MPI/SP systems, we have extended MPI/SP, which is called MPI/SPx (MPI/SP extended). MPI/SPx has three special features; (1) integration of MPI clusters. (2) pipeline communication between clusters, (3) high availability of larger CoC(Cluster of Clusters) system. In this paper, we describe the design of our middleware, MPI/SPx.

1. はじめに

近年，電話の交換機システムやレーダのデータ処理システムといった，社会インフラを支えるシステムに多数のプロセッサを組み合わせた HPC（High Performance Computing）クラスタを採用する動きがある．このようなミッションクリティカルなシステムに HPC クラスタを適用するために，我々は，高信頼化と高性能な並列処理を両立するクラスタリングミドルウェア：MPI/SP（MPI for Signal Processing）を開発した． MPI/SP では高信頼化のための機能を MPI のインタフェース内部に隠蔽しており，プログラマは高信頼化実現のための手続きを記述することなく，従来通り MPI アプリケーションを記述すれば高い信頼性を実現することが可能となった．

本研究では，これまでに開発した MPI/SP を応用し，MPI/SP で構築した信号処理クラスタを組み合わせることによってさらに大規模かつ複雑な演算の実現を目指している．今回，複数の MPI クラスタを接続した CoC（Cluster of Clusters）構成において，独立した MPI アプリケーション間のデータ交換を実現するために，複数の MPI/SP を統合化するミドルウェア：MPI/SPx（MPI/SP extended）を設計した．本稿では， MPI/SPx 上での複数の MPI クラスタ間メッセージパッシング，アプリケーションを変更することなくクラスタ構成を変更するための仕組み，およびシステム高信頼化の実現方法について述べる．

† 三菱電機（株）情報技術総合研究所
IT R&D Center, Mitsubishi Electric Corp

2. 背　　景

2.1 ターゲットシステムの特徴

図 1 は， MPI/SP に適用した信号処理システムの構成を示したものである．本システムは， 1 枚のマスタ CPU ボードと N 枚のスレーブ CPU ボードを持つ Compact PCI（C-PCI）ユニットをネットワークで接続したボードクラスタによって構成される．各 CPU ボードはネットワークインタフェースを持ち，これにより並列処理用のネットワークを構築する．クラスタを構成するプロセッサは，入力，演算，出力のように役割が決められており，デバイスからのデータ入力，演算，出力の一連の流れをパイプライン処理で繰り返し行なっている．

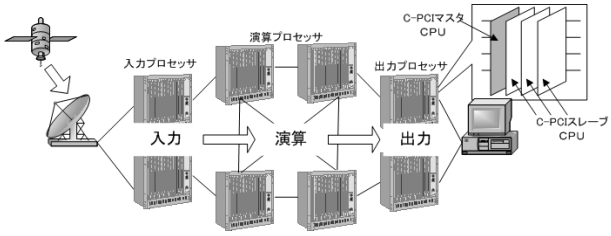


図 1　MPI/SP に適用した信号処理システムの構成

本研究で想定するシステムの構成を**図 2** に示す．本システムは，図 1 に示す信号処理クラスタシステム（以下，サブクラスタと呼ぶ）を組み合わせて構築した大規模信号処理システムであり，使用するレーダデバイスの台数や演算データの種別などに応じてサブクラスタ数の増減や使用サブクラスタの選択を行ない，システム構成を柔軟に変更できる点を特徴とする．

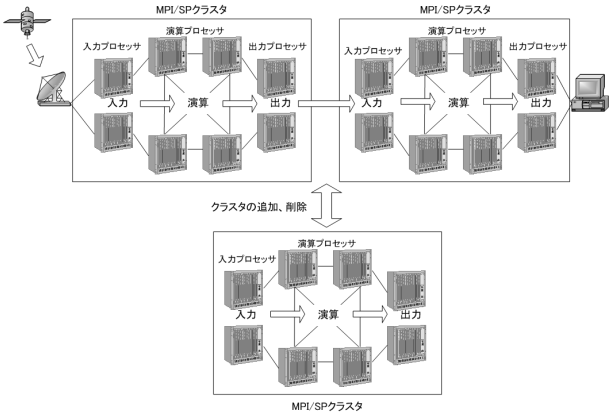


図 2　本研究で想定するシステムの構成

本ターゲットシステムの要求は以下の通りである．

複数のサブクラスタを組み合わせたパイプライン処理：　本ターゲットシステムでは，複数のサブクラスタを組み合わせ，データを段階的に処理する「パイプライン処理」によって全体の処理を実現する．

システム負荷，演算内容に応じたクラスタ構成の変更：　本ターゲットシステムでは，演算量や演算内容に応じてサブクラスタ単位で計算機構成を変更する．

システム高信頼化の実現：　本ターゲットシステムでは，演算結果を得るために長期間の連続運転を行なう．そのため，各クラスタで M+N 重化構成を採用することによりシステムの信頼性を高める．また，あるサブクラスタが動作不可能になった場合には，他のサブクラスタに通知し，システム全体を再起動する必要がある．

2.2　MPI/SP の特徴

これまでに我々が開発してきた高信頼プロセッサ間通信ミドルウェア：[1)]MPI/SP の特徴を以下に示す．

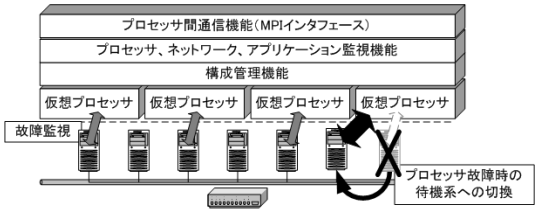


図 3　MPI/SP のソフトウェア構造

(1)　障害検出，復旧の自動化：プロセッサ，アプリケーション，ネットワークといったシステム構成要素の障害を検出し，障害箇所，障害の重度（一時的障害，故障など）に応じた適切な処置（再起動，停止，運用継続など）を行う．

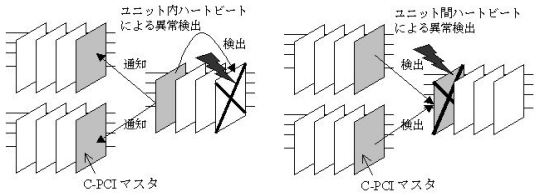


図 4　MPI/SP におけるプロセッサの障害検出

(2)　プロセッサ構成の動的変更の実現：MPI/SP ではプロセッサの冗長化（M+N 重化）構成に対応する．故障プロセッサと待機系プロセッサを入れ替える際，待機系プロセッサに対して故障プロセッサと同じランクを与えることにより，運転を継続しつつプロセッサを変更することができる．

(3)　システムの階層化管理：MPI/SP ではクラスタのマスタプロセッサが C-PCI マスタプロセッサの障害を検出し，C-PCI マスタが同一 C-PCI ユニット内の障害検出を行なう「階層型クラスタ管理」を行なうことにより，全 CPU の故障検出を可能にする．また，**図 5** に示すようなツリー構造を使用してシステム構成要素の故障に関する情報を送受信することにより，全プロセッサでのシステム構成情報の共有を実現する．

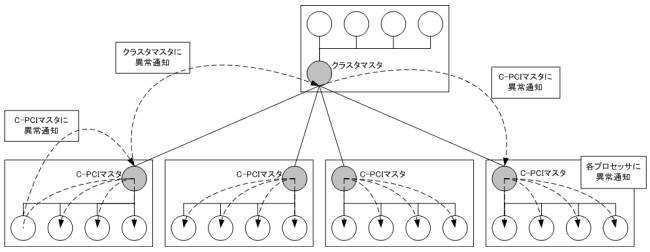


図 5　MPI/SP の階層化管理

(4)　大規模プロセッサ構成への対応：MPI/SP では数百台〜数千台のプロセッサ構成に対応する．システムの大規模化に伴って使用可能な OS 資源が少なくなってきた場合には，使用頻度の少ない OS 資源を解放することにより OS 資源を有効活用してシステムの大規模化に対応している．

(5)　MPI 1.2 準拠の API を提供：MPI/SP では，(1)〜(4) の機能を MPI[5)]（Message Passing Interface）1.2 版のインタフェース内に隠蔽する．これにより，従来通り MPI アプリケーションを記述するだけで並列処理とシステムの高信頼化を同時に実現する．

2.3　MPI/SP の拡張に向けた課題

ターゲットシステムのような，複数の並列計算機システムを接続させ，MPI のインタフェースを用いてメッセージ交

換を行なう手法として，異なる並列計算機を任意に組み合わせて並列演算を行なう通信ライブラリ：Stampi[3] や GRID 上での MPI によるプロセッサ間通信を実現するためのミドルウェア：MPICH-G2[4] などがある．

Stampi の構成および通信方法を**図 6** に示す．Stampi では，外部計算機と直接通信できないプロセッサからの通信を外部と通信可能なプロセッサに代行させることにより，直接ネットワーク接続できない計算機間の通信を実現している．

Stampi も MPICH-G2 も複数の並列計算機システムを連携させて単一の MPI_COMM_WORLD を構築しており，通信ドメインや計算機の違いを意識せずにプロセッサ間データ交換が可能である．

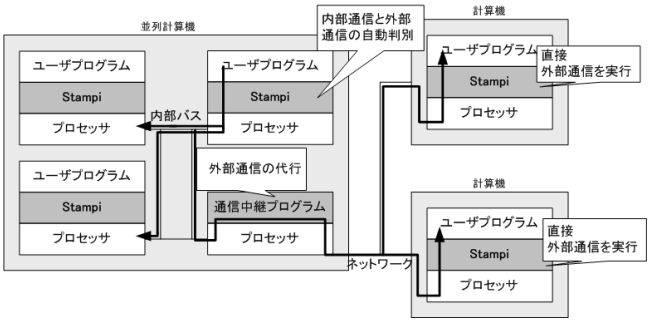


図 6 Stampi による計算機間通信

しかし，従来実装のような単一の MPI_COMM_WORLD によって本ターゲットシステムの要求を実現しようと考えた場合，以下の問題点が生じる．

- システム構成を変更すると全てのプロセッサのランクが変わるため，MPI アプリケーションを再実装しなければならない．
- 各サブクラスタの処理をまとめて 1 つの MPI アプリケーションで記述する必要があり，プログラムサイズが膨大となってメンテナンスが困難になる．

本研究でターゲットとするシステムでは，各サブクラスタの処理は独立して行なわれており，並列演算時にはサブクラスタ外のプロセッサに対して通信が行なわれることはない．従って，各クラスタでそれぞれ独立した MPI アプリケーションを作成し，それを組み合わせたほうがソフトウェアのメンテナンスが容易である．

これらのことから，本ターゲットシステムに対しては，各クラスタの独立性を保ちつつ，データ受け渡しのみをクラスタ間で連携させることによってシステム全体を構築するほうが望ましいと考えられる．

本研究では，MPI/SP を拡張することにより各クラスタで独立した MPI 環境を構築し，異なる MPI_COMM_WORLD 間でのデータ受け渡しやシステム構成の柔軟な変更を実現するための仕組みを提供する．

3. MPI/SPx の設計

今回設計した MPI/SPx（MPI/SP extended）を適用するためのプロセッサ構成を**図 7** に示す．図に示すように，各サブクラスタの演算プロセッサ群と隣接するサブクラスタの入力，出力プロセッサとをネットワーク接続することによりサブクラスタを連結している．

このようにシステムを構築することにより，入力プロセッサおよび出力プロセッサはサブクラスタへのデータ入力，サブクラスタからのデータ出力の両方を行なうこととなる．今後，これらのプロセッサを「入出力プロセッサ」と呼ぶ．

サブクラスタ間のデータ交換は入出力プロセッサがデータを中継することにより行ない，新規にサブクラスタを追加する際には，新規接続する入出力プロセッサと接続先の演算プロセッサとをネットワーク接続することにより実現する．

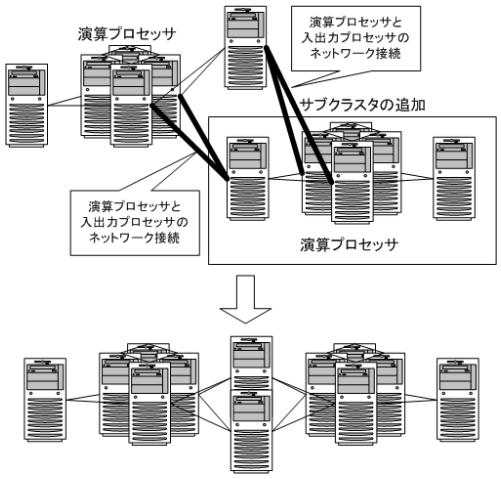


図 7 MPISPx を適用するためのシステム構成

このようなシステム構成にてシステム要求を満たすために，MPI/SPx では以下の拡張を行なった．

- 独立した MPI クラスタを連携させるために · · · プロセッサの複数の MPI_COMM_WORLD 加入
- クラスタ間データ交換を実現するために · · ·MPI_COMM_WORLD の切替によるクラスタ間データ交換
- アプリケーションの変更なくシステム構成変更を実現するために · · · プロセッサの役割の明確化
- システム高信頼化実現のために · · · システム階層化管理によるサブクラスタ異常情報の伝播の実現

以下，これらの設計内容について述べる．

3.1 複数の MPI_COMM_WORLD への加入

図 8 に示すように，MPI/SPx では，MPI クラスタ間連携を実現するため，入出力プロセッサが複数の MPI_COMM_WORLD グループに入ることにより複数の MPI クラスタを連結することとした．複数の MPI_COMM_WORLD に加入するために，MPI/SPx では以下のような拡張を行なっている．

複数のクラスタ構成テーブルの所有： 従来の MPI では，初期化時にクラスタ構成テーブルを読み込み，クラスタ構成テーブルに持つプロセッサ群をメンバとして MPI_COMM_WORLD グループを作成する．本研究では，入出力プロセッサが複数の MPI_COMM_WORLD グループに

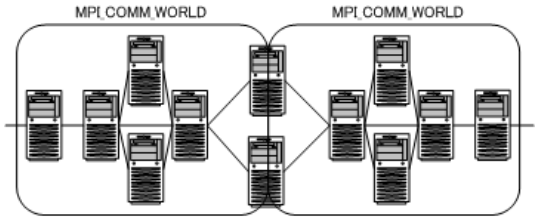


図 8　MPI/SPx における MPI_COMM_WORLD グループ構成

加入するため，自プロセッサが属する全ての MPI クラスタのクラスタ構成テーブルを持つこととした．

クラスタ構成テーブルへの入出力プロセッサの追加：　クラスタをシステムに新規接続する場合，接続される側のクラスタ構成テーブルに，新規追加するクラスタの入出力プロセッサを追加する．これにより，入出力プロセッサが複数の MPI_COMM_WORLD に属し，MPI クラスタが接続できる．

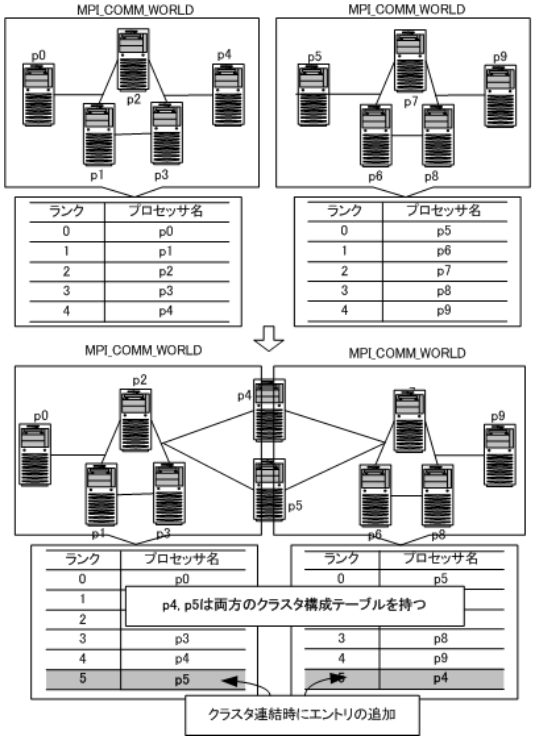


図 9　上段：従来の MPI のクラスタ構成テーブル
下段：MPI/SPx における複数のクラスタ構成テーブル

このように，入出力プロセッサに限って複数のクラスタ構成テーブルを持ち，すべてのクラスタ構成テーブルを読み込むことにより，複数の MPI_COMM_WORLD を持つことが可能になる．また，隣接するクラスタの入出力プロセッサをそれぞれ隣接するクラスタのクラスタ構成テーブルに示すことにより，入出力プロセッサを複数の MPI_COMM_WORLD グループに加入させ，MPI クラスタ間を接続することが可能となる．

3.2　MPI_COMM_WORLD の切替

前節に示すように，MPI/SPx では入出力プロセッサが複数の MPI_COMM_WORLD に属することにより隣接するクラスタ間を連結する．

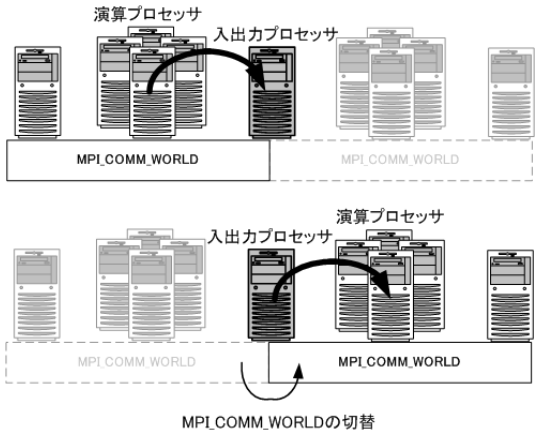


図 10　：MPI_COMM_WORLD の切替
（上段：データ受信時，下段：データ送信時）

このようなシステム構成においてクラスタ間通信を実現するために，入出力プロセッサは，データ受信時には送信側計算機が属する MPI_COMM_WORLD を使用し，データ送信時には送信側計算機が属する MPI_COMM_WORLD を使用する（**図 10**）．このような規則で入出力プロセッサ内部で MPI_COMM_WORLD を切り替えることにより，サブクラスタ間データ交換を実現する．

本方式では，入出力プロセッサと演算プロセッサは同一の MPI クラスタに属するため，各 MPI アプリケーションは，従来と同じように入出力プロセッサへ送信すれば自動的にクラスタ間パイプライン処理が実現できる．

3.3　プロセッサの役割の明確化

3.3.1　新規ランクの定義

前述のように，サブクラスタ同士の連結は，サブクラスタの入出力プロセッサと隣接する演算プロセッサ群をネットワーク接続し，各入出力プロセッサが隣接する MPI クラスタの MPI_COMM_WORLD グループに入ることにより実現する．

本拡張に伴い，以下の問題点が生じる．

- 新規参入した入出力プロセッサにデータを送受信するためには，新規参入するプロセッサのランクをアプリケーション作成時にあらかじめ知っておく必要がある．
- サブクラスタの追加などによりプロセッサ構成が変わると，各プロセッサに割り当てられているランクが変更となる．そのため，システム構成が変わるたびにアプリケーションを再作成する必要がある．

MPI/SPx では，問題点を解決するために，入出力プロセッサに対して MPISP_CLUSTER_OUT，MPISP_CLUSTER_IN という特殊なランクを与え，その他のプロセッサに 0 番から始まる従来通りのランクを与えることとする．

このランクの使用方法を以下に定める．

- MPISP_CLUSTER_OUT：各サブクラスタの演算用プロセッサが入出力プロセッサにデータを渡す際，MPI_Send()

の送信先ランクとして指定する．

- MPISP_CLUSTER_IN：各サブクラスタの演算用プロセッサが入出力プロセッサからデータを受け取る際，MPI_Recv() の送信元ランクとして指定する．

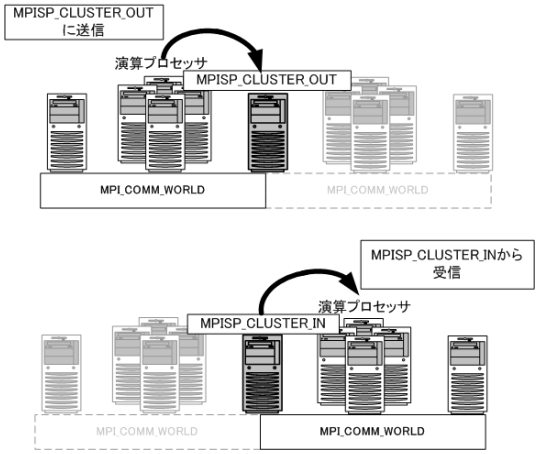


図 11 ：MPISP_CLUSTER_IN，MPISP_CLUSTER_OUT の使用方法

各サブクラスタに追加されるのは入出力プロセッサのみであるため，入出力プロセッサのランクをあらかじめ特殊なランクで固定化することにより，その他のすべてのプロセッサのランクは不変となる．

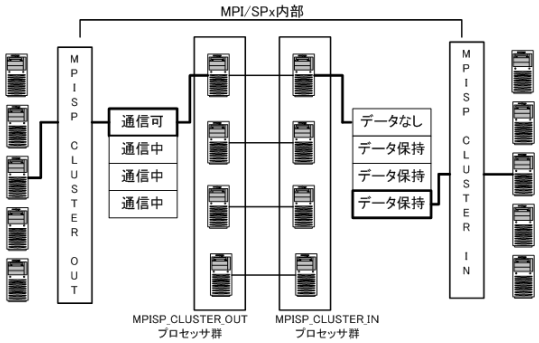
3.3.2 入出力プロセッサ群の隠蔽

前節のように，入出力プロセッサに特殊なランクを与えることにより，アプリケーションに影響を与えることなくシステム構成を変更することが可能となる．

しかし，サブクラスタを接続することにより，入出力プロセッサは複数個存在するため，従来の MPI とは異なり 1 つのランクに複数のプロセッサが割り当てられることになる．そこで，メッセージ送受信時には，複数の送受信プロセッサから 1 台を選ぶことが必要となる．本研究では，以下のように 1 台のプロセッサを選択してデータ送受信を行なうこととした．

演算プロセッサへの送信： MPI/SPx 内部で，入出力プロセッサ全体の使用状況を管理するテーブルを保持する．演算プロセッサが送信先を MPISP_CLUSTER_OUT と指定してデータを送信すると，テーブルに「送信可」と示される出力プロセッサから 1 台を MPI/SPx が選択し，そのプロセッサにデータが送信される．

入力プロセッサからの送信： MPI/SPx 内部で，入出力プロセッサのデータ保有状況を示したテーブルを保持する．MPISP_CLUSTER_IN を送信元に指定した場合，テーブルに「データ保持」と示される入力プロセッサから 1 台が MPI/SPx によって選ばれ，そのプロセッサがデータを送信する．演算プロセッサは，従来の MPI のランク定数：MPI_ANY_SOURCE を指定して任意のプロセッサからのデータを待つことにより，任意の入出力プロセッサからデータを受けることができ，サブクラスタ間のデータ交換が可能となる．



このように，MPI/SPx 内部でデータを送受信する入出力プロセッサを選択することにより，入出力プロセッサと通信する場合には前節に示した規則でランクを与えることにより入出力プロセッサとの間でデータ受け渡しが実現できる．

3.4 階層型クラスタ管理の拡張

MPI/SP の高信頼化機能により各サブクラスタ内部では高信頼化が可能である．しかし，現状では MPI/SP の高信頼化機能は互いに連携していないため，あるサブクラスタ全体が動作不能になったとしても，他のサブクラスタに知らせることができない．そこで，MPI/SPx ではサブクラスタも階層化管理することにより，MPI/SP の高信頼機能を連携させる．

(1) 各サブクラスタの入出力プロセッサから各サブクラスタのクラスタマスタを選択する

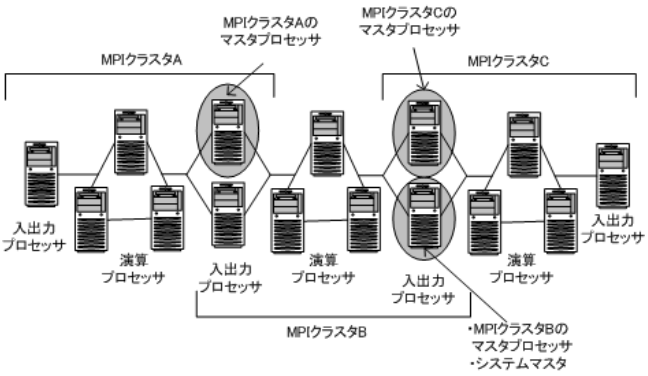


図 12 MPI/SPx におけるシステムマスタの選択方法

(2) (1) で選択したクラスタマスタからシステム全体を管理するシステムマスタを選択し，システムマスタを頂点とする階層構造を作成する．

(3) (2) で示したツリー構造に従ってサブクラスタの構成情報を伝達することにより，各サブクラスタが動作しているかどうかを伝達する．

(4) あるサブクラスタが動作不可能になった場合には，サブクラスタのマスタプロセッサから伝達が来ないことにより認識することができる．システムマスタは，サ

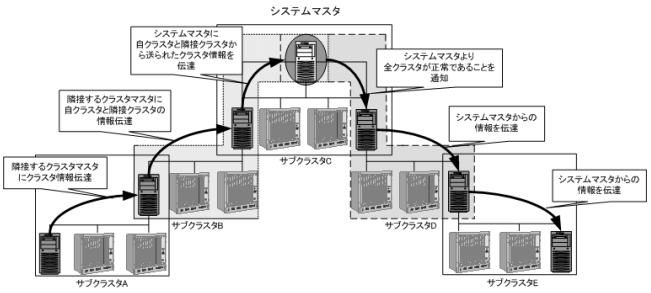


図 13 MPI/SPx におけるサブクラスタ情報の伝達

ブクラスタのマスタから伝達されたシステム障害の情報を再度サブクラスタに伝達することにより，システムを停止させることが可能になる．

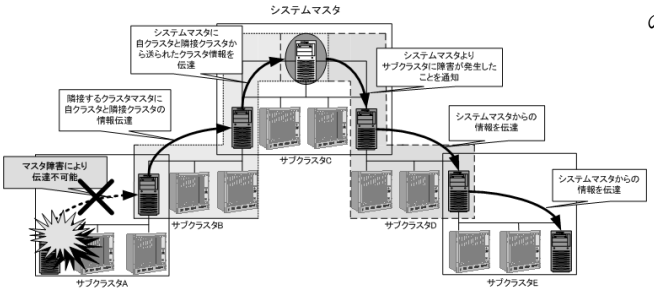


図 14 MPI/SPx における故障クラスタの検出

このように，すべてのクラスタ情報を１つのプロセッサに集中させ，再度配布することができるため，全部のプロセッサが１つのネットワークに接続されていなくてもクラスタの動作状況を伝達することができるため，あるクラスタが動作不可能になった場合には，全クラスタを停止させることができる．

3.5 実現に向けた課題

現在 MPI/SPx の実装中であるが，新たに以下の課題が発生している．

初期化時のサブクラスタ間連携： 各サブクラスタが MPI 環境の初期化を独立して行なった場合，システム全体の初期化完了時期を判別できないため，アプリケーションの実行をいつ開始してよいのかわからない．
そのため，MPI 環境の初期化時には，サブクラスタ間で同期を取る仕組みを MPI/SPx 内に実装する必要がある．実現手段としては，3.4 節に示したようなサブクラスタ間での情報交換による方法が考えられる．

入出力プロセッサの負荷分散の定義： MPI/SPx では，複数台ある入出力プロセッサを隠蔽し，使用するプロセッサは MPI/SPx 内部で自動的に選択している．そのため，入出力プロセッサ群の負荷分散をアプリケーションで意識する必要がない．しかし，負荷分散の方法はアプリケーションによって多様であるため，ユーザによって負荷分散のポリシーを設定できるような仕組みを MPI/SPx 内部に提供する必要がある．

システムマスタの高負荷化の回避: これまでの MPI/SP の設計では，任意の C-PCI マスタプロセッサがクラスタマスタ，システムマスタとなることができたが，本設計では，必ず入出力プロセッサがクラスタマスタ，システムマスタとなる必要がある．そのため，これらのプロセッサが高負荷となり，システム処理能力の低下の可能性がある．システムマスタの高負荷化を回避する方法としては，入出力プロセッサを多く配置することや，クラスタマスタ，システムマスタ専用プロセッサを配置することなどが考えられる．

4. おわりに

本研究では，Cluster of Clusters 構成で複数の MPI アプリケーションを統合化し，クラスタ間でパイプライン処理を実現する MPI/SPx の設計内容について述べた．MPI/SPx の特徴を以下に示す．

- 一部のプロセッサが複数の `MPI_COMM_WORLD` グループに属し，それらのプロセッサが `MPI_COMM_WORLD` を切り替えることによって MPI アプリケーション間のデータ交換を行なう．
- 入出力プロセッサ群に `MPISP_CLUSTER_IN`, `MPI_CLUSTER_OUT` という特殊なランクを与えることによりシステム構成変更を隠蔽し，アプリケーションを変更することなくシステム構成変更を実現する．
- 新規ランクを定義する以外に API を変更することなく MPI アプリケーション間の連携を実現する．

今後，3.5 節に示した課題について検討の上 MPI/SPx 内部の実装を行ない，通信性能やアプリケーション実装の容易性について検証を行なう．

参考文献

1) 村山　和宏，落合　真一：Cluster of Clusters 構成に向けた MPI/SP の拡張, FIT2003（第２回情報科学技術フォーラム）（2003）
2) 村山 和宏, 落合 真一： 大規模分散システムに向けた高信頼化機構の設計, 情報処理学会研究報告 2003-DPS-111, pp191–196.
3) T. Imanuma, Y. Tsujita, H. Koide and H. Takemiy ：” An Architecture of Stampi: MPI Library on a Cluster of Parallel Computers ”, LNCS 1908 Recent Advances in Parallel Virtual Machine and Message Passing Interface, Springer, pp.200–207.
4) N. Karonis, B.Toonen, and I.Foster: “ MPICH–G2 : A Grid-Enabled Implementation of the Message Passing Interface ”, *journal of Parallel and Distributed Computing(JPDC),* Vol.63, No.5, pp.551–563, May 2003.
5) MPI： *http://www.mpi-forum.org/*