グリッド環境を対象とした ログベース・フォールトトレランス手法の提案

服部 晃和　横田 隆史　大津 金光　古川 文人　馬場 敬信 †
宇都宮大学工学部情報工学科‡

1 はじめに

近年, 大規模かつ複雑な問題が様々な分野から発生している. その問題を解決するためのインフラストラクチャとして, グリッド [1] が注目されている. しかし, これらの大規模かつ複雑な問題を解決するためには, 大規模な計算資源を利用し, 長時間に渡って計算を行わなければならない. この場合, 処理中に計算資源にフォールトが発生する確率が, 1台の計算資源を使用する時と比べて高くなってしまう. そのため, 計算資源にフォールトが発生しても, クラッシュしたプロセスのリカバリを行い, 計算を自動的に続行する機能がグリッドに備わっている必要がある. 本稿では, 複数のサイトに分散した計算資源を活用し, 並列処理を行うグリッド環境を対象としたフォールトトレラントシステム, Eagleを提案する.

2 環境

我々が提案するEagleシステムは, 計算資源が複数のサイトに分散し, それらを同時に活用し, 数千から数百万に上るプロセス数で並列処理を行うグリッド環境を対象としている. プロセス間の通信はメッセージのみを用いる. このような大規模な環境でフォールトトレランスを確立するためには, プロセスの管理, 現実性の観点から, 以下の3つの特性を得る必要がある.

(1)プロセスは個別にチェックポイントを取り, 個別にリカバリを行える.
(2) あるサイトのプロセスのフォールトは他のサイトから隠蔽されている.
(3) サイト間通信を高信頼化する.

分散システムの研究で, (1) を満たす手法がいくつか考案されている. そのひとつに, pessimistic logging[2] と呼ばれるリカバリプロトコルがある. pessimistic loggingは, プロセスがメッセージを受信する前に, そのメッセージのログが信頼できる格納庫である, stable storageに格納されていることが保証されているプロトコルである. そのため, クラッシュしたプロセスは, チェックポイントから計算を再開し, メッセージのログを用いて, 個別にクラッシュした直前の状態までリカバリすることが可能である. しかし, この手法はメッセージ通信が高信頼であることを前提としている. サイト内通信では, この前提は有効であると考えられるが, サイト間通信はサイト内の通信と比べて, メッセージが消失する可能性が高いので, この前提は現実的でない. そのため, この手法を使用するためには, (3) の特性を得る必要がある. そ

* A proposal of a log-based fault tolerance method for grid environment
† Akikazu Hattori, Takashi Yokota, Kanemitsu Ootsu, Fumihito Furukawa, and Takanobu Baba
‡ Department of Information Science, Faculty of Engineering, Utsunomiya University

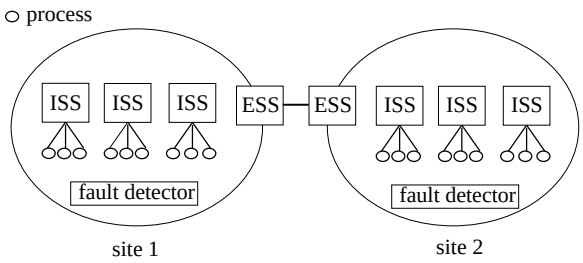

図 1: Eagleシステムの全体構成

こで, Eagleシステムはpessimistic loggingを基盤とし, stable storageの階層化を行うことによって, 上記の特性を得ている. 次節で, Eagleシステムについて具体的に説明する.

3 Eagleシステム

3.1 Eagleシステムの概要

Eagleシステムの全体構成を図1に示す. Eagleシステムはプロセスのチェックポイントと, メッセージのログを格納しておくISS(Internal Stable Storage), サイト間通信を信頼化するためのESS(External Stable Storage), プロセスのフォールトを検出するフォールトディテクタから構成される.

3.2 Internal Stable Storage

ISSは, 担当するプロセスのチェックポイントと, そのプロセス宛に送信されたメッセージを格納する. メッセージは直接プロセス宛に送信するのではなく, 一旦そのプロセスを担当するISSに送信され, そのメッセージのログが完了した後, 宛先のプロセスにメッセージを送信する. また, メッセージを送信する際に, プロセスはチェックポイントを取った後からの, メッセージの送信数をISSにカウントしておく. プロセスにクラッシュが発生した場合は, ここに格納されている情報を用いてリカバリを行う.

プロセスクラッシュとリカバリによってメッセージの送信先が変わってしまっても, ISSがメッセージをルーティングすることによって, 宛先が変わった旨を全プロセスに通知する必要がなくなる. この特性により, 他のサイトからフォールトを隠蔽することができる.

3.3 External Stable Storage

ESSはサイト間通信を高信頼化する. 別のサイトにあるプロセス宛のメッセージは, まず送信元のサイトのESSに格納される. その後ESSは, 送信先のサイトのESS宛にメッセージを送信する. その際に, 送信先のESSからACKを受け取るまで, 定期的にメッセージを送信し続ける. こうすれば, サイト間通信時にメッセージが消失してしまってもメッセージが再送されるので, 最終的に宛先

のサイトのESSはメッセージを受信することができる.

送信先のサイトのESSがメッセージを受信したら, 送信元のサイトのESSに向けてACKを返す. ACKがサイト間の通信中に消失してしまっても, 通信元のサイトのESSはACKを受け取るまでメッセージを再送するので, メッセージを受信する度にACKを送信することで, 最終的に送信先のESSから, 送信元のESSにACKが返される. この機構を用いることにより, サイト間通信を高信頼化することができる. 送信先のサイトのESSにメッセージが送信されたら, 宛先のプロセスを担当しているISS宛にメッセージを送信し, ログの完了後, プロセスにメッセージが到着する. サイト間通信時にメッセージの順番が入れ替わってしまうことが有り得るが, メッセージに, 送信元, 送信先別の通し番号を付けておくことで, 順番の入れ替わりを検出することができる. 順番の検出はISSで行う.

3.4 リカバリ

Eagleシステムでは, メッセージがISSに保存された後に, 宛先のプロセスにメッセージが送信される. そのため, プロセスにクラッシュが発生しても, クラッシュした直前の状態に至るまでのメッセージは, 全て担当のISSに格納されている. プロセスにクラッシュが発生した際は, ISSに格納されている, クラッシュしたプロセスのチェックポイントから計算を再開し, ISSに格納されているメッセージのログを用いて, プロセスにクラッシュが発生した直前の状態までロールバックする. この際に, ISSに格納されているチェックポイントからのメッセージの送信回数を参照することによって, メッセージの再送を防ぐ.

3.5 フォールトディテクタ

Eagleシステムは他のサイトのプロセスのフォールトから独立できるように設計されているので, フォールトディテクタは, サイト内に閉じてプロセスの状態を監視していれば良い. この特性により, フォールトの検出を高信頼化することができる.

4 評価

4.1 シミュレータの概要

我々は, Eagleシステムの有効性を評価するために, シミュレータEGsimの開発を行っている.

入力は,以下の2つのファイルとなる.
(1)シミュレーションパラメタファイル
(2)プロセスビヘイビアファイル
(1)は, サイト数, サイト内のプロセス数, ISSが担当するプロセス数, LANのバンド幅等, シミュレーションパラメタが記述されている. (2)はプロセスの振舞, すなわちプロセスのメッセージ送信, 受信, チェックポイント, クラッシュ等の, 各プロセスのスケジュールが記述されているファイルである.

出力は, 以下の3つとなる.
(1)フォールトトレランスを施さない場合の処理時間
(2)Eagleシステムを用いて処理を行った場合の処理時間
(3)フォールトが発生した場合のリカバリの平均時間

表 1: シミュレーションパラメタ

通常計算時間	3600[sec]
サイト数	10
プロセス数	各サイトに100
サイト内のESS数	1
LANのバンド幅	125 [Mbyte/s]
WANのバンド幅	12.5 [Mbyte/s]
ISSの書き込み速度	30 [Mbyte/s]
ESSの書き込み速度	30 [Mbyte/s]
メッセージサイズ	1 [Mbyte]
チェックポイントサイズ	100 [Mbyte]

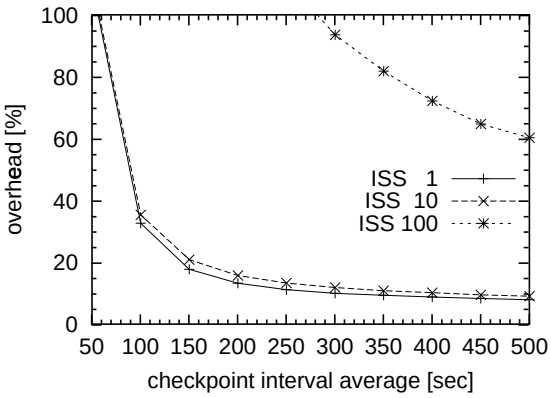

図 2: オーバヘッドの評価

これにより, Eagleシステムのオーバヘッド, リカバリ性能を見積もることができる.

4.2 予備評価

本稿では, 予備評価として, 表1に示すようなパラメタにおいて, ISSがそれぞれ, 1, 10, 100個のプロセスを担当した場合のEagleシステムのオーバヘッドを示す(図2). チェックポイントの平均間隔が500秒の時を見ると, ISSが100個のプロセスを担当した場合と比べて, ISSが1, 10のプロセスを担当した場合の方が, 45%程度オーバヘッドが小さいことが分かる.

5 おわりに

本稿では, 複数のサイトに分散した計算資源を活用し, 並列処理を行うグリッド環境を対象としたフォールトトレラントシステムである, Eagleシステムを提案した. またEagleシステムを評価するためのシミュレータEGsimを用いて予備評価を行った. 今後は, より詳細なシミュレーションパラメタを選定し, シミュレーション結果を比較し, Eagleシステムを評価する必要がある. また, Eagleシステムを実装し, 現実的に評価する必要がある.

謝辞 本研究は, 一部日本学術振興会科学研究費補助金(基盤研究(B)14380135, 同(C)14580362, 若手研究14780186)の援助による.

参考文献

[1] I. Foster, C. Kesselman and S. Tuecke: "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," International J. Supercomputer Applications 15(3), (2001).

[2] E. Elnozahy, L. Alvisi, Y. Wang and D. Johnson: "A survey of rollback-recovery protocols in message-passing systems," Technical Report CMU-CS-99-148, Carnegie Mellon University, June (1999).