M-045

P2P 型オブジェクト複製環境における 連携/非連携チェックポインティング手法の性能評価

Performance Evaluation of Coordinated and Independent Checkpointing on P2P-based Object Replication Environment

高橋 利幸 † 野口 尚吾 ‡ 高田 秀志 †
Toshiyuki Takahashi Shogo Noguchi Hideyuki Takada

1. はじめに

我々は, 同期型協調作業支援システムを実現するための P2P 型オブジェクト複製環境 “CUBE” [1] の開発を行っている. CUBE では, 各ノード上にオブジェクトの複製を生成し, それらのオブジェクトのふるまいを伝播することで, 複製オブジェクトの状態を同一に保っている.

一般的に, 分散システムの一部に障害が発生すると, システム全体として正常な運用ができなくなる. このため, 正常運用時にシステムの状態を保存しておき, 障害が発生した場合には, 保存した状態を用いて回復する機構が必要となる. この機構として, チェックポインティングと呼ばれる手法が存在し, 一般的に用いられている.

本稿では, P2P 型オブジェクト複製環境に, 一般的な分散環境での障害回復機構である連携/非連携チェックポインティング手法を適用することを考える. また, 複製環境におけるチェックポインティング手法について, 動作などの点から評価を行う.

2. P2P 型オブジェクト複製環境

P2P 型オブジェクト複製環境では, Peer-to-Peer(P2P) モデルを用いたネットワーク上で, 各ノードが他ノード上のオブジェクトの複製を保持している.

この環境では, あるノード上でオブジェクト間でメッセージパッシングが発生したときに, 他ノードが保持しているオブジェクトの複製にそのメッセージパッシングを伝播することで, 各ノード上のオブジェクトの複製を同一に保っている. 図 1 の通常時部分では, ノード B やノード C が他ノードに対して伝播を行う様子を示す. このような動作により, 各ノードのオブジェクトの状態は同一に保たれ, それらのオブジェクトで構築されるシステムの状態も各ノードで同一になる.

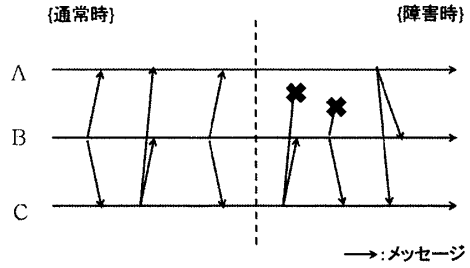

図 1: ノード間のメッセージパッシング

また, システムの一部に障害が発生した場合には, システムの運用に支障をきたす. この場合の例として, 図 1 の障害時部分に, ノード A への通信の切断が発生した場面を示す. この場合, ノード A には, 2 回分のメッセージパッシングの欠けが発生し, その後, オブジェクト A からのメッセージパッシングが伝播されるとシステムの一貫性に問題が起こる可能性がある. こうした場合には, システムの運用に支障のない状態に回復する必要があり, この回復を行うためには, システムの状態を保存する機構が必要となる.

3. P2P 型オブジェクト複製環境でのチェックポインティング手法

分散環境下でのチェックポインティング手法として, 連携手法と非連携手法の 2 種類が一般的に用いられている [2]. これら 2 つの手法を P2P 型オブジェクト複製環境に適用した場合について述べる.

3.1 連携チェックポインティング

連携手法では, チェックポイントを作成する場合, あるノードがチェックポイント作成開始を指示し, 他ノードが未処理のメッセージを処理した後に, チェックポイントを作成する. 例として図 2 のノード P がチェックポイント作成を指示するとすると, 最初にノード P がチェックポイントの作成開始を報せるメッセージをノード Q とノード R に送信する. それを受信したノード Q とノード R は, ノード P と同じ状態にするために未処理のメッセージパッシングを処理したあと, ノード P に対して準備完了のメッセージを送信する. ノード P は, ノード Q とノード R から準備完了のメッセージを受信したあと, ノード Q とノード R にチェックポイント作成実行メッセージを送信する. ここまでの動作が図 2 のチェックポイント作成前までに行われる制御メッセージのやり取りである. そして, 各ノードはチェックポイントを作成する. その後, ノード Q とノード R はノード P に完了メッセージを送信する. ノード P が全ノードから完了メッセージを受信した時点でチェックポイント作成が終了する. また, 回復処理時には, それぞれのノードが保持している最新のチェックポイントにロールバックすることで状態を回復する. これらの動作は一般的な分散環境下での連携手法と同じである.

3.2 非連携チェックポインティング

もう一つの手法である非連携手法では, 各ノードが他ノードとチェックポイントの作成のタイミングを合わせることなく独自にチェックポイントを作成する. この場合, 各ノードが持つチェックポイントのデータは, 2 節で説明したオブジェクト複製環境であることにより, 一貫したシステム状態となる. また, 回復処理時には, 障害から回復するノードが各ノードに最新のチェックポイント作成時間を問い合わせ, 最新のチェックポイントを持つ

†立命館大学 情報理工学部
‡立命館大学大学院 理工学研究科

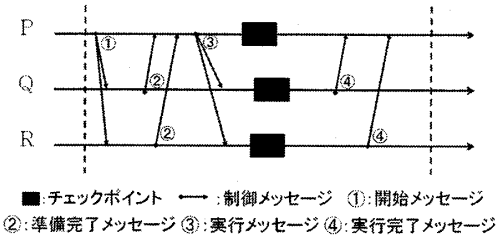

図 2: 連携手法のチェックポイント作成の動作

ノードから最新のチェックポイントデータを取得し, 状態の回復を行う. 非連携手法の回復時の動作を図 3 に示す.

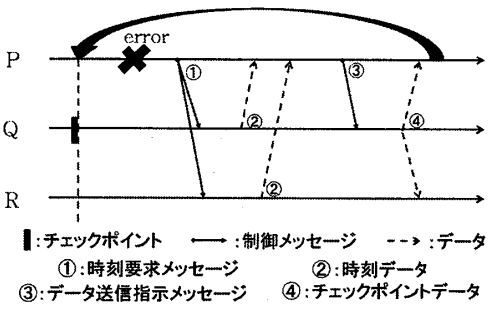

図 3: 非連携手法の回復時の動作

また, 回復後の状態を可能な限り最新の状態とするために, 綿密に状態を保存する必要がある場合, 一般的な分散環境では図 4 の上部分に示すように, 各ノードが連続でチェックポイントを作成しなければならない. しかし, P2P 型オブジェクト複製環境では, 各ノードの状態は他ノードの状態と同一になるため, 図 4 の下部分のように各ノードにチェックポイントを分散させることができる. これにより, 各ノードで連続にチェックポイント作成するための負担を軽減させることができる.

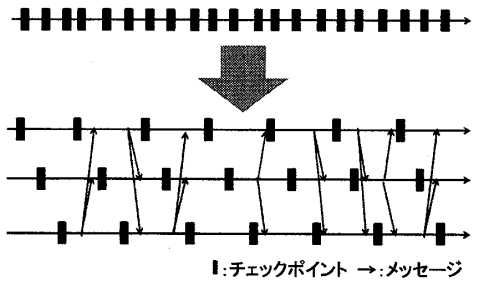

図 4: チェックポイントの分散による負担軽減

4. 評価と考察

本節では, P2P 型オブジェクト複製環境に連携/非連携チェックポインティングを適用した場合の評価を行う. また, 回復面についても考察する.

4.1 定性的評価

連携手法では, 3.1 節で述べたようにチェックポイント作成時に制御メッセージの送受信を行うが, この間, 各ノードは状態を同一にするために新たなメッセージパッシングの処理を行わない. つまり, システムを利用するユーザは一時的にシステムの利用を中断させられることになる.

非連携手法では, 3.2 節で述べたようにチェックポイントのデータは一貫したシステム状態となることにより, ノード間で一貫した状態を保持するチェックポイントの組を探索する必要がなくなる. そのため, 容易に状態の回復が行えると共に, 一般的な分散環境での非連携手法の問題点である “ドミノ効果”が発生しない. また, 分散環境では, 各ノードの状態が同一となっているチェックポイントを探索するために最新のチェックポイントのデータ以外にも数回分のチェックポイントのデータを保持する必要があったが, P2P 型オブジェクト複製環境下では, 前述したようにチェックポイントを探索する必要がなく最新のチェックポイントのデータだけで回復可能である.

4.2 考察

3 節で P2P 型オブジェクト複製環境に適用した場合の連携手法と非連携手法を述べた. この 2 つの手法における回復処理について考察を行う.

連携手法における回復方法として 2 つの方法が考えられる. 1 つ目の方法では, 障害の発生したノードが保持するチェックポイントを用いて回復する. この方法では, 障害が発生したあとに, 他ノードが新たにチェックポイントを作成した場合でも, 障害の発生したノードが保持するチェックポイントを用いる. そのため, 回復後の状態が最新の状態からかけ離れている場合がある. 2 つ目の方法では, 最新のチェックポイントを他ノードが保持している場合に, そのチェックポイントのデータを取得し, 回復する. そのため, 回復後の状態が最新の状態に比較的近いものになる. しかし, チェックポイントのデータを全ノードに送信するため, オーバヘッドがかかる.

非連携手法における回復方法では, 全ノードが保持するチェックポイントの中で最新のものを用いて, 回復する. この方法は, 最新のチェックポイントを用いるため, 回復後の状態が最新の状態に比較的近いものになる. しかし, データ送信のオーバヘッドがかかる.

これらの方法で回復したあとさらに, 状態を最新のものにするために, ログを用いる方法が考えられる.

5. おわりに

本稿では, 分散環境の中の一形態である P2P 型オブジェクト複製環境下で, 分散環境での一般的な連携/非連携チェックポインティングを適用し, 動作などの点から評価を行った.

今後は, 評価・考察からの問題を解決した P2P 型オブジェクト複製環境に最適なチェックポインティング手法の提案を目指す.

参考文献

[1] Shogo Noguchi, Hideyuki Takada, “CUBE: A Synchronous Collaborative Applications Platform Based on Replicated Computation”, CollabTech2009, 2009.

[2] Luis Moura e Silva, Joao Gabriel Silva, “Global Checkpointing for Distributed Programs”, Proceedings of the 11th Symposium on Reliable Distributed Systems, PP.155-162, 1992.