クラスタシステムの高信頼化技術

南谷崇　中村宏

情報理工学系研究科システム情報学専攻（先端科学技術研究センター）

1　はじめに

クラスタシステムは極めて低コストに高性能計算環境が構築可能であるため，大規模科学技術計算を中心に近年広く利用されている．一度の計算時間が数時間から数日に及ぶことも少なくない大規模科学技術計算では，この間の障害発生に対処する高信頼化技術は重要なものとなってきている．一方，大規模な HPC（High Performance Computing）クラスタシステムにおいては，構成要素となる商用既製品の数が多く，システムの故障率も大きくなる．そのようなクラスタシステムにおいては単一のノードのみの故障（単独故障）だけでなく，複数ノードの故障（多重故障）にも対応できる高信頼化技術が必要である．

我々は，低コストかつ高性能というクラスタシステムの利点を損なわないためには，高信頼化は空間的なハードウェア冗長化ではなく，システムソフトウェアによるチェックポインティングが適していると考えている．しかし，多重故障に対応するためにチェックポイントデータを単純に冗長化しても，チェックポイントオーバヘッドが増大し，高性能というクラスタシステムの利点が損なわれる．そこで，我々は，チェックポイント自体は冗長化せず，チェックポイントデータの保存先を毎回変更することでオーバーヘッドを最小限に抑えつつ，多重故障にも対応できる新しいチェックポインティング方式として Skewed Checkpointing[1, 2, 3] を提案する．

2　Skewed Checkpointing

2.1　多重故障に対応可能な従来手法

多重故障に対応したチェックポインティングとして MIR（Checkpoint Mirroring）[4] と CFS（Central File Server）[4] が提案されており，さらにその二つを組み合わせた 2-level Recovery Scheme[5] も提案されている．

MIR　MIR は k 重故障に対して k 個のチェックポイント mirror データを異なる k 個のノードに転送・保存し，自分自身もチェックポイントを保存する方法である（自分自身も保存するのでチェックポイントデータは k+1 個のノードに保存される.）k=1 の時，すなわち単独故障のみに対応した MIR（これを 1-mirror MIR と呼ぶ）であれば，それぞれのノードが保存する他ノードのチェックポイントデータは 1 個で良くチェックポイントオーバーヘッドは小さい．しかし，k 重故障まで対応した MIR （k-mirror MIR）の場合，転送・保存すべきチェックポイントの総量は k 倍となり，チェックポイントオーバーヘッドが大きくなる．

CFS　CFS は故障しない安全な共有ディスク (CFS:Central File Server) に全てのチェックポイントを保存する方法である．チェックポインティング時，リカバリ時に全ノードから CFS へ，もしくは CFS から全ノードへのデータ転送が行われるので，CFS へのアクセスが集中し，チェックポインティング・リカバリ時間は大きくなる．この時間は，CFS 1 台あたりのノード数に大きく影

響される．しかし，CFS は高価なため，このオーバーヘッド低減のために CFS を多く設置することはコストパフォーマンス上の問題が生じる．

2-level Recovery Scheme　低オーバーヘッドで多重故障に適応すべく 2-level Recovery Scheme が提案されている．発生確率の高い単独故障からすばやく復旧するには，チェックポイントオーバーヘッドが小さく，リカバリも高速な 1-mirror MIR を用いるのが最適である．しかし，1-mirror MIR では多重故障に対応できないので，オーバーヘッドは大きいが多重故障に完全に対応している CFS を 1-mirror MIR の数回おきに行う．そして単独故障時は最新の 1-mirror MIR のチェックポイントから回復し，多重故障時は CFS のチェックポイントから回復する．こうすることで，オーバーヘッドの大きい CFS の頻度を減らし，かつ発生確率の高い単独故障からはすばやく復旧可能になる．

2.2　提案手法：Skewed Checkpointing

我々が提案する Skewed Checkpointing の着眼点は 2-level Recovery Scheme と同様，発生確率の高い単独故障からはすばやく復旧可能で，かつ多重故障にも対応することである．

チェックポインティング　1-mirror MIR ではノード i のチェックポイントデータをノード $(i+1)modN$ に保存する．この場合ノード i とノード $i+1$ が同時に故障すると，ノード i のチェックポイントデータが失われ復旧させることができず，アプリケーションを再開させることは不可能となる．但し，ノード i とノード $k(k \neq i+1)$ が同時に故障した場合は再開は可能である．

提案する skewed checkpointing では冗長チェックポイントの保存先を定期的に変更する．P_1〜P_N までの N ノードがある場合，図 1 のようにそれぞれ CP_1〜CP_m で表される $m = \lfloor log_2 N \rfloor$ 種類の MIR チェックポインティングを実行する．CP_i では P_k のチェックポイントデータをローカルノードと $P_{(k+2^{i-1} modN)}$ に保存する．この時の 2^{i-1} を

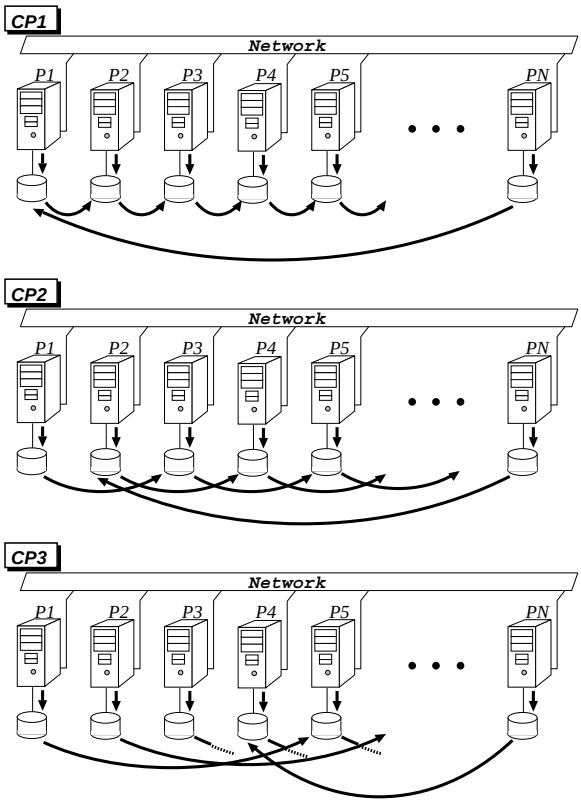


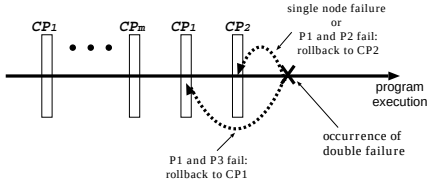
図 1: Skewed Checkpointing

skew distance と呼ぶ．それぞれのノードは他のノードが保存した冗長チェックポイントから回復可能であるように最新の m 個のチェックポイントを保存しておく．

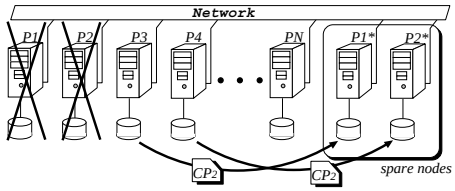
リカバリ　単一のノードの故障のとき図 2 に示すように最新のチェックポイントから回復する．

回復処理中にもう一つの故障が発生した場合を二重故障とする．この場合，故障ノードの場所に応じて，最新のチェックポイント，もしくは二つ前のチェックポイントから回復処理を行う．つまり，二つの故障ノードの最新のチェックポイントが，故障が発生していないノードから利用可能である場合は，最新のチェックポイントから回復する．そうでない場合，つまりある故障ノードに他の故障ノードの最新のチェックポイントが保存されていた場合，システムは二番目に新しいチェックポイントから回復する．

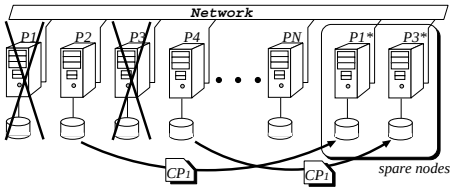
例えば最新のチェックポイントが CP_2 であった



(a) ロールバック



(b) P_1 と P_2 が故障した場合



(c) P_1 と P_3 が故障した場合

図 2: Skewed Checkpointing におけるロールバックリカバリ

場合は skew distance は 2 である．もし P_1 にと P_2 に故障が発生した場合，図 2 に示すようにシステムは P_1 と P_2 のチェックポイントデータを P_3 と P_4 から回復して CP_2 にロールバックする．もし P_1 と P_3 が故障した場合，システムは P_1 と P_3 のチェックポイントを P_2 と P_4 から回復して CP_1 にロールバックする．

定理　ノード数が N の場合，Skewed Checkpointing は $m = \lfloor log_2 N \rfloor$ ノードの多重故障を必ず回復することができる．つまり m ノードの多重故障は最新の m 個のチェックポイントのうち，少なくとも一つから必ず回復可能である．これは式で書くと以下で与えられる．

$$\exists d_k \in D, \forall P_{f_i}, P_{f_j} \in Z, \quad \neg \left(P_{f_i} \xrightarrow{d_k} P_{f_j} \right), \\ \text{where } P_{f_i} \xrightarrow{d_k} P_{f_j} \text{ represents } (f_i + d_k) mod N = f_j. \tag{1}$$

ここで，Z は故障した m ノードの集合，D は最新の m 個のチェックポイントの skew distance の集合を表わす．即ち，
$Z = \{P_{f_1}, P_{f_2}, \cdots, P_{f_m} (1 \le f_m \le N)\}$,
$D = \{\ d_i \mid d_i = 2^i (0 \le i \le m-1)\}$ である．

式 (1) は，故障した全てのノードのチェックポイントを保存されている skew distance が少なくとも一つは存在することを示しており，その存在する CP_k は Z 内の全ての故障ノードを回復できる．

証明　式 (1) の否定である式 (2) が成立しないことを示すことで証明する．式 (2) は最新の m チェックポイントのいずれからも回復不可能な m ノードの多重故障の集合が存在することを意味する．

$$\forall d_k \in D, \exists P_{f_i}, P_{f_j} \in Z, P_{f_i} \xrightarrow{d_k} P_{f_j} \tag{2}$$

ここで D は以下の特徴を持つ．ただし，D_1 と D_2 は $D_1 \cap D_2 = \emptyset$ を満たす D の部分集合である．

1. $\forall D_1, D_2,\ \sum_q d_q \neq \sum_r d_r\ (d_q \in D_1, d_r \in D_2)$.

これは 2 進表現が一意であることより明らか．

2. $\forall D_1, D_2,\ \sum_q d_q + \sum_r d_r\ < N\ (d_q \in D_1, d_r \in D_2)$.

これは $\sum_{i=0}^{m-1} d_i < N$ から明らかである．

3. $\forall D_1, D_2,\ \xrightarrow{\sum_q d_q \in D_1}$ は $\xrightarrow{\sum_r d_r \in D_2}$ や $\xrightarrow{-\sum_r d_r \in D_2}$ と等価ではない．

これは前述の 2 つの特徴から明らかである．

4. $\forall D_1, D_2,\ \forall P_a, P_b (0 \le a, b < N, a \neq b)$,

“$P_a \xrightarrow{\sum_q d_q \in D_1} P_b$” が成り立つ場合，“$P_a \xrightarrow{\sum_r d_r \in D_2} P_b$” と “$P_b \xrightarrow{\sum_r d_r \in D_2} P_a$” は成り立たない．

これは前述の D の特徴から明らかである．

最後に示した特徴より式 (2) を満たすためには少なくとも $m+1$ ノードが必要である．なぜなら m 種類の異なる “$\xrightarrow{d_k}$” や “$P_{f_i} \xrightarrow{d_k} P_{f_j}$” が全て成り立たなければならないからである．しかしながら Z の中にはたかだか m ノードしか存在しない．したがって，これは矛盾であるので仮説は成り立たないことが証明された．

同様に以下の補題が得られる．

表 1: チェックポイントとリカバリ時間

方式	CP time[sec]	recovery time[sec]
MIR(k=1)	60	39
MIR(k=2)	105	56
CFS	127.5	119
Skewed	60	39

表 2: 仮定するパラメータ

failure rate (λ)	0.000001
task length (Υ)	500000[sec]
CP size	500[MB]

補題 $k(1 \leq k \leq m)$ ノードの故障は最新の k チェックポイントから回復可能である.

3 評価

Skewed Checkpointing と他の方式の比較評価を行った. 評価は, 想定する故障率においてチェックポイントとリカバリを行った場合の実行時間の期待値を求め, チェックポインティングを行わず故障もなかった場合に対する実行時間の増分割合（オーバーヘッド）を用いて行う.

評価はまず, 単独故障のみ対応の MIR（$m = 1$）, 2 重故障まで対応の MIR($m = 2$), CFS, 5 回に 1 回 CFS を行う 2-LEVEL($r = 5$), 50 回に 1 回行う 2-LEVEL($r = 50$), Skewed について, チェックポイントと回復にかかる時間を 16 ノード構成のクラスタシステムで実測した. 次に, 表 1 に示す得られた値を用いて, 表 2 の仮定における実行時間の期待値を評価式より算出する. 評価式はマルコフモデルから導出したが, その詳細は文献 [3] を参照されたい.

評価結果を図 3 に示す. 横軸は各方式におけるチェックポインティングの間隔, 縦軸はオーバーヘッドを表わす. いずれの方式においても下に凸のグラフになる. これは, チェックポインティングの頻度を多くしすぎるとチェックポイント自体のオーバヘッドにより性能が低下し, 逆に頻度を少なくしすぎると故障発生時のロールバックで失う時間が大きくなるためである. また, 方式間の比較

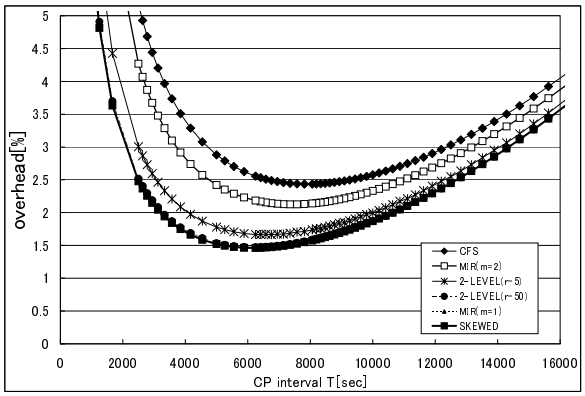


図 3: 性能オーバヘッド

では, Skewed Checkpointing がもっともオーバーヘッドが小さく, 有効であることが確認できる.

4 まとめ

本稿では Skewed Checkpointing という HPC クラスタシステムに適した多重故障対応の高信頼化技術を提案した. また, 評価式からオーバーヘッド求め, その有効性を示した. 今後はノード数を増やした場合の評価を行い, 大規模クラスタシステムにおける有効性を示したい.

参考文献

[1] 田島裕也、林田卓朗、近藤正章、今井雅、中村宏、南谷崇、“多重故障に適応した Skewed Checkpointing の提案”, 先進的計算基盤システムシンポジウム SACSIS2004, pp. 153-154, May, 2004

[2] 田島裕也、林田卓朗、近藤正章、今井雅、中村宏、南谷崇、“多重故障を考慮した計算機クラスタ向け Skewed Checkpointing の検討”, 信学技報 DC2004-19(2004-07), pp.37-42, 2004 (SWoPP2004)

[3] H. Nakamura, T. Hayashida, M. Kondo, Y. Tajima, M. Imai, and T. Nanya, “Skewed Checkpointing for Tolerating Multi-Node Failures”, Proceedings of IEEE SRDS’04, pp.116-125 , Oct. 2004

[4] J. S. Plank, “Improving the Performance of Coordinated Checkpoints on Networks of Workstations using Raid Techniques”, Proc. of SRDS’96, pp.76-85, 1996.

[5] N. H. Vaidya, “A Case for Two-Level Recovery Schemes”, IEEE Transactions on Computers, Vol. 47, No. 6, pp.656–666, June, 1998