# 高速なオンライン処理のためのＤＢバックアップ方式

4Ｇ−9

小林伸幸　大竹孝幸　梅田昌義
ＮＴＴ情報通信網研究所

## 1　はじめに

近年のリレーショナルデータベース（ＲＤＢ）の適用領域の拡大に伴い、従来より高速なＤＢアクセスを行うオンライン処理の必要性が高まっている。オンライン処理中でＤＢアクセスを行うトランザクションの処理速度に影響を与える要因の１つにＤＢのバックアップ処理がある。

ＤＢの運用に際しては、種々の障害への対策等のために、一定周期毎にＤＢデータのバックアップを取得することが多く行われる。このバックアップ処理を行う代表的な方式の１つに、ファジーチェックポイント方式[1]がある。この方式は、トランザクションとの排他制御が不要なため、他の方式に比べオンライン処理の処理時間の劣化が少ないことを特長としている。しかし、バックアップしたデータの一貫性を保証するためには、バックアップ処理中に走行するトランザクションのログを必要とする。ログの取得は、２次記憶への書き込みを行うＩ／Ｏが必要であり、さらにオンライン処理を高速化するときの問題点となる。

本稿では、ファジーチェックポイント方式における一貫性の保証に必要なログの取得方法について検討し、オンライン処理の処理時間への影響が少ないＤＢバックアップ方式について提案する。

## 2　一貫性を失う要因

本章では、ファジーチェックポイント方式で取得したデータが、一貫性を失う要因について分析する。通常のトランザクションは、アクセス対象のデータに対する排他制御を行うことにより、データの一貫性を保証している。しかし、ファジーチェックポイント方式では、この排他制御を超越して、バックアップを取得するため、取得したデータの一貫性が失われる。具体的には、以下に示す場合が考えられる。

(1)トランザクションがＩ／Ｏ中のデータを、バックアップ処理が取得した場合。
(2)更新前のデータをバックアップ処理が取得し、その更新トランザクションが正常終了した場合。（図１参照）
(3)更新後のデータをバックアップ処理が取得し、その更新トランザクションが異常終了した場合。（図２参照）

また、バックアップ処理中にシステムダウンやディスク媒体障害の発生により、一貫性が失われる場合も考えられる。

(4)更新後のデータをバックアップ処理が取得し、その更新トランザクションが完了以前に障害が発生した場合。
(5)バックアップ処理中にＩ／Ｏ中障害が発生した場合。

## 3　ログの取得方法

本章では、ログの取得においてＩ／Ｏを削減する方法を検討する。Ｉ／Ｏ削減には以下の方法が考えられる。

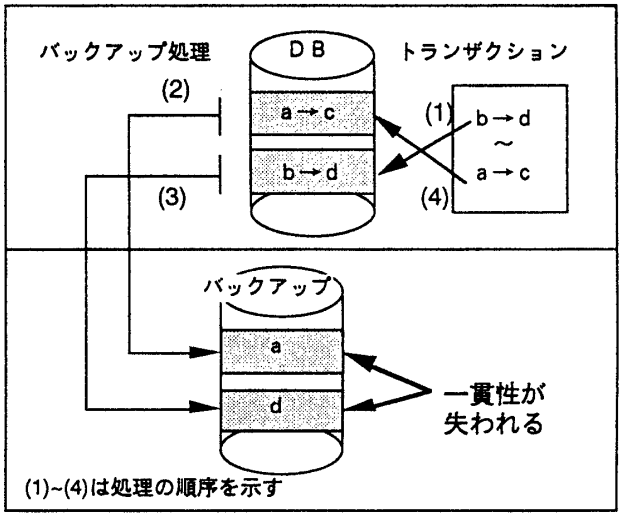


図１　一貫性の喪失

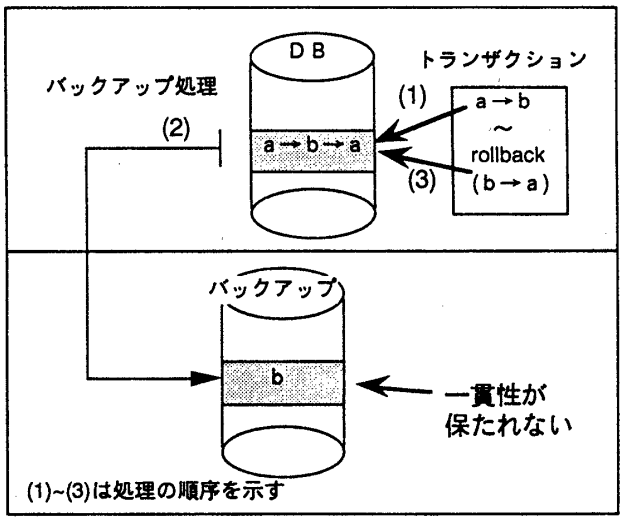


図２　一貫性の喪失

(a)ログの取得量を少なくする。
(b)ログの取得契機を減らし、一括して出力する。
(c)ログバッファのオーバーフローによるＩ／Ｏを抑止する。

このうち、(c)については、充分な大きさのログバッファを確保することにより対処できる。そのため、本稿では、(a)(b)について検討を加える。

### 3−1　通常のログ取得方法

はじめに、通常のトランザクションでのログ取得方法について検討する。ログは、更新対象データについて、挿入／削除／更新等の個々の更新系処理の前に２次記憶に出力され、ロールバック処理に使われる更新前の値（ＢＩ）と、終了時に一括して出力され、ロールフォワード処理に使われる更新後の値（ＡＩ）から構成される。しかし、本稿では、正常終了した場合の

---

DB backup method for on-line transactions.
Nobuyuki Kobayasi, Takayuki Ohtake, Masayosi Umeda.
NTT Network Information Systems Laboratories.

ＡＩのみ２次記憶へ出力し、障害発生時には、バックアップデータからのロールフォワード処理に用いる。ＢＩについては、２次記憶へは出力せず、メモリ上でトランザクションをロールバックするのに用いる。このことによりログの取得量は１／２になり、取得契機は、最大で各トランザクションの中で１回にできる。

### ３－２　バックアップ処理中のログ取得方法

次に、２章で述べた一貫性の喪失を防ぐために、バックアップ処理中に対象データを更新するトランザクションのログ取得方法について検討する。

２章の要因のうち、(1)(2)(3)について一貫性を保証するため、更新前後でＢＩおよびＡＩをログバッファ上に取得しておき、トランザクションが正常終了したときはＡＩを、異常終了したときはＢＩを２次記憶に出力する方法を提案する。こうして取得したログにより、更新対象データを正常終了したときは更新後の値に、異常終了した場合には更新前の値にすることができ、一貫性が保証される。また、取得契機は、トランザクションのなかで１回にすることができる。

(4)については、バックアップ処理で取得した値を、トランザクションの更新前に戻すことにより一貫性は保証される。そのためには、トランザクションがアボートすることを考えて、個々の更新系処理の前に２次記憶へ出力する必要があり、取得契機が増大する。そのため、本稿では、バックアップ用ファイルを２世代化することを提案する。そして、(4)が発生した場合は、バックアップ処理をアボートし、前世代のバックアップデータを用いることにより一貫性を保証する。

また、同様に(5)が発生した場合について、バックアップ処理をアボートし、前世代のバックアップデータを使用することにより一貫性を保証できる。

以上のログ取得方法により、ディスク内の格納効率は１／２になるが、トランザクション中におけるログの取得契機は、各トランザクション内で１回にすることができる。

## 4　バックアップ方式

前章の検討結果より、本稿で提案するＤＢバックアップ方式を以下に示す。

(1)バックアップ処理

ファジーチェックポイント方式に従い、オンライン処理の排他制御を超越して、バックアップ対象データを２次記憶に出力する。また、このとき出力先ファイルは２世代化しておき、交互に使用する。

(2)ログの出力処理

図３にのログの出力処理を示す。

本稿では、トランザクションの走行状態および終了形態により２次記憶に出力するログの取得方法を切り替える。以下に、それぞれの場合の示す。

(A)通常のトランザクション

通常のトランザクションでは、正常終了時にＡＩのみを取得し、ＢＩについては取得しない。

(B)バックアップ処理中のトランザクション

バックアップ処理と同時に走行するトランザクションでは、更新系処理の前後においてＢＩとＡＩをログバッファ上に取得しておく。そして、正常終了した場合はＡＩのみを、異常終了した場合はＢＩのみを取得する。

(3)バックアップ処理中の障害対策

バックアップ処理中にシステムダウンやディスクの媒体障害発生した場合、該バックアップ処理をアボートする。このときＤＢを復元する場合は前世代のバックアップファイルを使用する。

## 5　まとめ

本稿では、高速なオンライン処理のためのＤＢバックアップ処理について検討を行った。バックアップ方式として、オンライン処理への影響が少ないファジーチェックポイント方式を採用し、この方式において一貫性を保証するログの取得方法およびバックアップ方式について述べた。その結果、トランザクションの終了形態によりＡＩかＢＩの一方を２次記憶に出力すること、およびバックアップ用ファイルを２世代化することにより一貫性を保証し、かつ高速にＤＢをバックアップできることを示した。

今後は、ＤＢの更新方式等も含め、より詳細な定量評価を行う予定である。

【参考文献】

[1]D.J.Rosenkrantz,"Dynamic database dumping.",in Proc. SIGMOD Int. Conf. Management Data. ACM,1978,pp.3-8

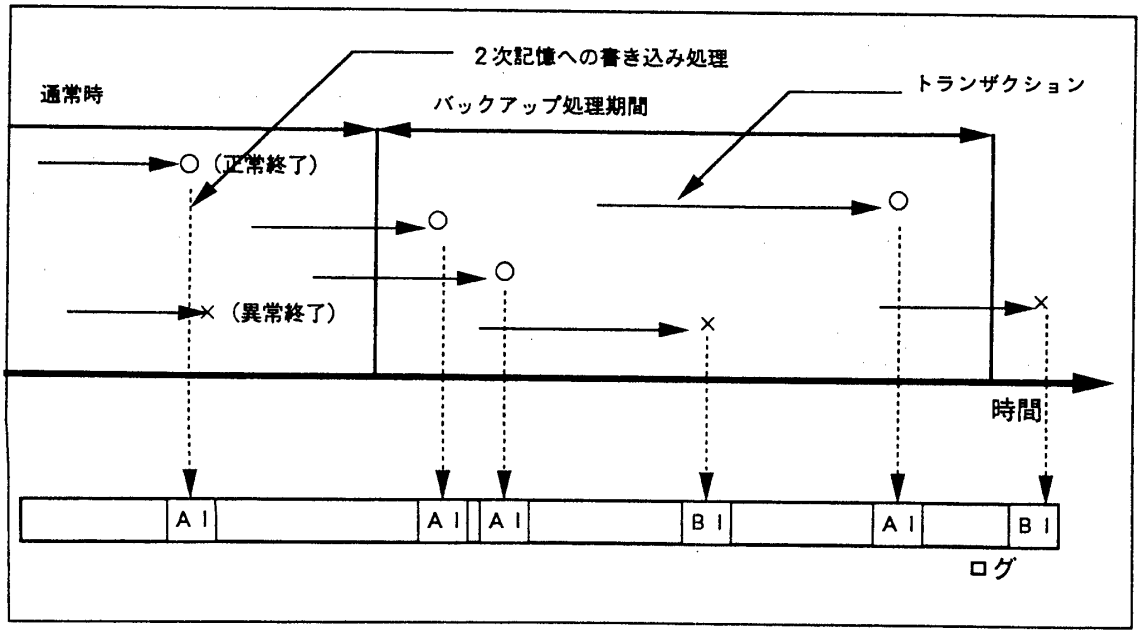


図３　ログの取得方法