関係データベースにおける構造劣化監視機構を用いた再編成スケジューラの提案

A Reorganization Scheduler on RDBMS Using Structural Deterioration Monitor

星野 喬♠ 合田 和生♦ 喜連川 優♦

Takashi HOSHINO Kazuo GODA
Masaru KITSUREGAWA

データベース更新が繰り返されると，二次記憶装置上のデータ格納構造が非効率になり，性能が低下する．このような構造劣化を除去するためにデータベース再編成は不可欠であるが，構造劣化している時間と空間を同定する必要がある．近年，データベースの大規模化による管理コスト増大，常時運用ニーズ増大などの背景があり，構造劣化監視，再編成実施のためのリソースが減り，管理者の経験則だけでは対処しきれないため，システム自身が自立的に構造劣化監視，再編成計画を行う必要が出てきた．本論文は，低オーバーヘッドなリアルタイム構造劣化モニタ及び，SLA ポリシに基づく再編成スケジューリングフレームワークを提案する．評価実験により，提案手法の有効性を明らかにする．

Database updates disorganize data stored physically in secondary storage and causes performance degradation, which is called structural deterioration. Database reorganization removes structural deterioration. Reorganization scheme requires to know when the database reorganization should be invoked, and what region of the database should be reorganized. Nowadays, database size increases, database administration becomes costly, and full-time database operation is required, then system resources which can be spent to monitor and maintain the database decrease, and administrator's heuristic reorganization scheduling becomes more difficult. In this paper, we propose a structural deterioration monitor with a little overhead and a framework of reorganization schedule based on SLA policies and evaluate them.

1. はじめに

近年,様々な分野において運用されているデータベース(以下，DB)が大容量化している．そのため，データベースシステム(以下，DBMS)における管理コストの増大が問題となっている．DBMS常時運用ニーズもまた増大しており，DBの状態監視および管理タスク実行のための時間やシステムリソースが少ない．また，人的操作によるミスが甚大な被害を引き起こす事例も増えており，人間の管理者のみによるきめの細かいDB管理に限界が見えてきた．この問題を解決するため，DB管理の自立化を目指す試みが増えている[3].

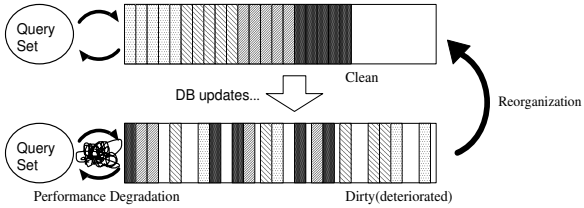

図 1 構造劣化と再編成
Fig. 1 Structural deterioration and reorganization of database

我々はDB再編成管理に着目し，その自立化を目的としている．DB表空間内のデータは，更新操作を繰り返すことによりその構造が劣化し，本来想定している配置と大きく乖離した状態となり，性能を大幅に低下させる場合がある(図1). このため，DB管理者は再編成を行うことにより，ストレージ内のデータを再配置し，性能低下を予め防ぐ必要がある．

常時運用に対応した再編成の自立化を実現するために，以下の課題を解決する必要がある．

・性能指標としての定量的な構造劣化モデルがない.
・構造劣化監視のオーバーヘッドが大きい.
・再編成実施判断のための枠組みがない.

まず，構造劣化の定量的な表現手法及び，それを常時監視する機構が必要であるが，観測される性能をそのまま構造劣化量として用いることは，構造劣化以外の性能低下原因との切り分けがしにくいため難しい．また，バッチ処理など時間のかかる処理の場合，実際にクエリを実行しないで構造劣化量を把握できることが望ましい．DBMSを含む現状のデータインテンシブアプリケーションでは，性能のボトルネックがストレージIOに起因することが多いため，我々はこれらのアプリケーションを対象に，ストレージにおけるデータ配置から期待されるIOコストを構造劣化量とし，高精度かつ低オーバーヘッドでリアルタイムに構造劣化を推定する構造劣化モニタを構築し，それを用いた再編成スケジューラを提案する．再編成スケジューラは，管理者，ユーザにとって分かりやすいSLA[1]ポリシに基づく再編成実施判断基準を用いて，再編成すべき時間と空間を出力する．提案手法は，DB再編成の自立化に大きく貢献すると期待される．

本論文は次のように構成される．まず，第2章で関連研究について述べ，次に第3章で自立再編成機構について述べる．その後，第4章で提案手法の評価を行い，最後に第5章にて結論と今後の課題を述べる.

2. 関連研究

DB 再編成に関する研究には，実行方式の高度化及び実行スケジューリングの最適化を中心に行われてきた．前者については，近年オンライン再編成方式についての研究が行われてきている[2],[6]．後者については，再編成スケジュールの最適化に関する研究が行われてきた[1]．これらの研究では，表空間を構成するストレージの性能特性は考慮せず，比較的簡単なDBモデルに基づいて再編成スケジューリング手法が

♠ 学生会員 東京大学大学院情報理工学系研究科
hoshino@tkl.iis.u-tokyo.ac.jp
♦ 正会員 東京大学生産技術研究所
{kgoda,kitsure}@tkl.iis.u-tokyo.ac.jp

[1] Service Level Agreement. ユーザもしくはアプリケーションから与えられる，システムが満たすべきサービス要件.

考察された．文献[4]は，DB バッファ等の影響を考慮して，索引と表を走査するのに必要な IO 数をコストとしてモデル化している．

著者らは，文献[7]において，DB の構造劣化の表現手法と，その定量的推定手法を提案し，インクリメンタルな構造劣化監視手法について提案した．本論文では，それらのモデルを定式化し，高精度，低オーバーヘッドなリアルタイム構造劣化モニタ及び，それを用いた再編成スケジューラを提案する．本手法では，これまであまり DBMS が考慮してこなかったストレージ性能特性を考慮した構造劣化を扱えるため，より正確な構造劣化量推定が可能になり，より高いレベルでのポリシー記述言語を適用した再編成スケジューリングが今後可能になると期待される．

3. 自立再編成機構

本論文で対象にする自立再編成機構の概要を図 2 に示す．本機構は，構造劣化モニタ，再編成スケジューラ，再編成エグゼキュータ[2]から構成される．構造劣化モニタは，ストレージ性能特性を用いて正確な構造劣化推定を行う．また，ストレージ上のデータ配置をそのまま用いるのではなく， DB 更新差分を用い，システムに対して低オーバーヘッドかつリアルタイムに構造劣化量を推定する．再編成スケジューラは，推定された構造劣化量，及びアプリケーションもしくは管理者から与えられた SLA ポリシに基づいて構造劣化領域を同定する．再編成エグゼキュータは同定された構造劣化領域を再編成対象として，再編成を実施する．

本論文では，表及び索引の実装方式として B+Tree 構造を対象にする．B+Tree の構造が変化する更新が起きる度に，DB エンジンに実装された B+Tree 更新差分抽出器[3]から共有メモリを介して必要な情報を抽出する．

以上のフレームワークにより，SLA ポリシに基づき，構造劣化による想定外の性能低下を防ぐことが可能になる．また，再編成空間を細かい粒度で同定することにより，再編成実施コストに対して性能改善効果の高い再編成が可能になることも期待される．本論文では，構造劣化モニタ及び再編成スケジューラについて述べる．再編成エグゼキュータの詳細については議論しない．

3.1. 構造劣化モニタ

本節では，構造劣化モニタの動作原理である，構造劣化のモデルおよび，更新差分による構造劣化量のインクリメンタル更新手法について述べる．表 1 に本章で用いる変数，定数及び関数の一覧を示した．

クラスタ表の B+Tree は，枝及び根ページにはクラスタ鍵(Clustered Key)と下位ページへのポインタが，葉ページにはデータ行が格納されている(図 3)．葉ページは，論理空間ではクラスタ鍵順に並んでいる．範囲検索は，対象となる範囲(例えば図 3 における範囲 $\mathbf{R}$)に属する葉ページ列 $\mathbf{S}=\{p_0, p_1, \cdots p_{N-1}\}$ を，クラスタ鍵順に読み込む走査を伴う．ここで，$|\mathbf{S}|=N$ とする．一般的には，B+Tree における葉ページの論理順序とストレージ内の物理順序が対応し，シーケンシャルアクセスとなることが期待されているが，更新によって，B+Tree の構造が変化すると，各ページの物理的

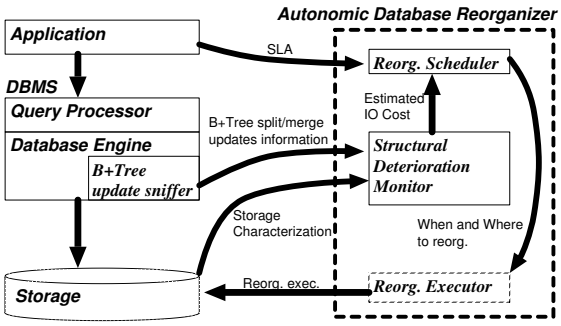

図 2 自立再編成機構

Fig. 2 Autonomic database reorganization framework

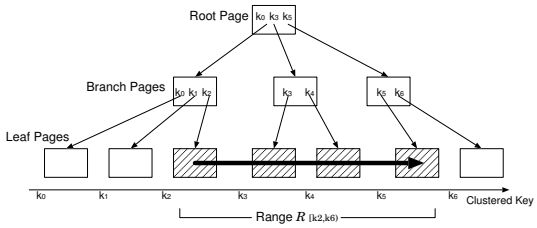

図 3 クラスタ表の B+Tree 構造

Fig. 3 B+Tree structure of clustered table

表 1 変数，定数及び関数一覧表

Table 1 Variables, constants and functions

変数	意味
$\mathbf{R}$	クラスタ鍵範囲
$\mathbf{S}$	$\mathbf{R}$ に対する葉ページ列
p	ディスク上のページアドレス
f	ページ列から IO コストを導く関数
$\mathbf{G}$	$\mathbf{S}$ に対する構造劣化分布
G_0	定数
C_{R}	範囲 R の範囲検索 IO コスト
X	許容構造劣化度
$\mathbf{T}$	再編成対象となる，$\mathbf{S}$ の部分集合

な配置が乱雑化する．このため，範囲検索のディスクアクセスは非シーケンシャル化し，性能が大幅に低下する．範囲検索において，個々の葉ページ p_i を読み込む IO コスト $G[i]$ は，p_i と直前に読み込んだページ p_{i-1} から，ディスクドライブ特性を考慮して IO コストとして近似でき，関数 f を用いて，

$$G[i]=\begin{cases} G_0 & i=0 \\ f(p_i - p_{i-1}) & i>0 \end{cases} \quad (1)$$

式(1)とする．関数 f の詳細については，文献[7]で詳しく述べている．このとき，範囲 $\mathbf{R}$ に対する範囲検索 IO コスト $C_{\mathbf{R}}$ は，式(2)で近似される．これを範囲 $\mathbf{R}$ の構造劣化量とする．

$$C_{\mathbf{R}} = \sum_{i|p_i \in \mathbf{S}} G[i] \quad (2)$$

G_0 は，ディスクドライブからのシーケンシャル読み込みにおけるページの読み込み IO コストを表すものとする．ディスクドライブにおいては， $G[i]$ の取りうる値の中で G_0 が最小値とする． $G[i]$ の列は $\mathbf{S}$ に対する構造劣化分布 $\mathbf{G}$ である．

次に，構造劣化の DB 更新差分によるインクリメンタル更新手法について述べる．DB 更新による B+Tree の構造変化は，ページの分割もしくは結合である． $\mathbf{S}$ に対してはページが追加されるか，削除されるかのどちらかである．このとき，

[2] 順に Structural Deterioration Monitor, Reorganization Scheduler, Reorganization Executor.

[3] B+Tree Update Sniffer

表 2　B+Tree 構造変化における C_R の差分

Table 2　C_R update with B+Tree update

	ページ q 追加	ページ削除
左端	$+f(p_0-q)$	$-f(p_1-p_0)$
右端	$+f(q-p_{\|S\|-1})$	$-f(p_{\|S\|-1}-p_{\|S\|-2})$
その他の位置 i_0	$-f(p_{i_0}-p_{i_0-1})$ $+f(q-p_{i_0-1})$ $+f(p_{i_0}-q)$	$-f(p_{i_0}-p_{i_0-1})$ $-f(p_{i_0+1}-p_{i_0})$ $+f(p_{i_0+1}-p_{i_0-1})$

C_R の変化量はそれぞれ境界での動作を含めて，表 2 に示した 6 つの場合に分けられる．

これにより，DB 更新を用いて，インクリメンタルに $\mathbf{S}$，$\mathbf{G}$ を，あらかじめ指定された範囲 $\mathbf{R}$ に対する C_R を計算し，最新の情報を保持することが出来る．全範囲を対象に $\mathbf{G}$ を保持すると，式(2)を用いて任意の範囲 $\mathbf{R}$ の構造劣化量 C_R が推定でき，リアルタイムに構造劣化を監視できる．

3.2. 再編成スケジューラ

再編成スケジューラが行うべき再編成実施判断は，再編成によって SLA が満たされるように構造劣化領域 $\mathbf{T}$ を求める問題に帰着できる．まず，SLA ポリシのインターフェースについて述べ，その後，SLA に基づく再編成領域同定手法について述べる．

本論文では SLA ポリシ記述インターフェースとして最も基本的な定義を行う．構造劣化量の許容基準として，構造劣化していない場合に期待される範囲検索クエリの応答時間に対する相対比を許容構造劣化度 X とし，以下の式で SLA を表す．

$$C_R(=\sum_{i|p_i\in\mathbf{S}} G[i]) < G_0|\mathbf{S}|X \qquad (3)$$

許容構造劣化度 X は $X>1$ で与えられる．式(3)は範囲検索 IO コスト C_R が構造劣化のない最高性能が期待できる場合における IO コストの X 倍未満であることを意味する．本論文では，任意の範囲 $\mathbf{R}$ に対して，式(3)を満たすものとする．ある時刻において，式(3)が満たされない場合，再編成スケジューラは構造劣化が許容範囲を超えていると判断し，再編成対象領域 $\mathbf{T}$ を決定し，再編成エグゼキュータに再編成実施命令を出す．$\mathbf{T}$ が与えられたとき，再編成操作によって，ページ列 $\mathbf{S}'$，その構造劣化分布 $\mathbf{G}'$ が得られるとする．このとき，式(4)が成り立つものとする．

$$G'[i]=\begin{cases} G_0 & p_i\in\mathbf{T} \\ G[i] & p_i\notin\mathbf{T} \end{cases} \qquad (4)$$

本論文では，再編成実施手法及び，その結果であるページ列 $\mathbf{S}'$ については議論しない．効率的な再編成スケジュールを行うため，与えられた SLA を満たすと同時に，再編成対象領域の大きさ $|\mathbf{T}|$[4]を最小にするような領域 $\mathbf{T}_{\min}$ が望ましい．図 4 に示すように $G[i]<G_0X$ が全ての p_i すなわち全ての領域に対して満たされることでのみ，式(3)を満たすことができる．以上より，再編成対象領域 $\mathbf{T}_{\min}$ は，$G[i]\geq G_0X$ を満たす p_i の集合である．

4. 評価

提案する構造劣化モニタついて評価を行った．また，再編成スケジューラの手法が実際に適用可能か考察を行った．実

[4] $\mathbf{T}$ に含まれるページ数．

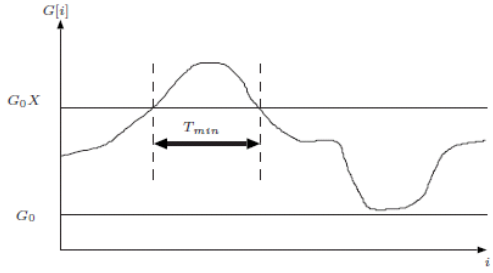

図 4　再編成対象領域

Fig. 4　Reorganization target area

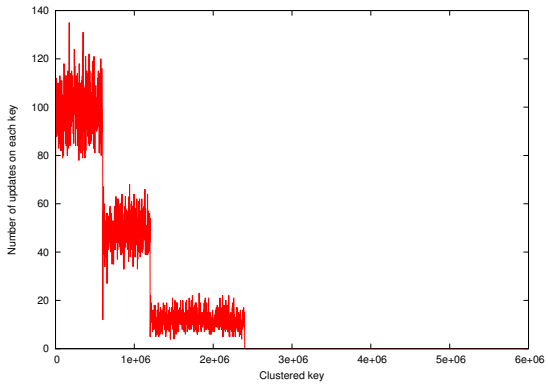

図 5　更新分布

Fig. 5　DB update distribution

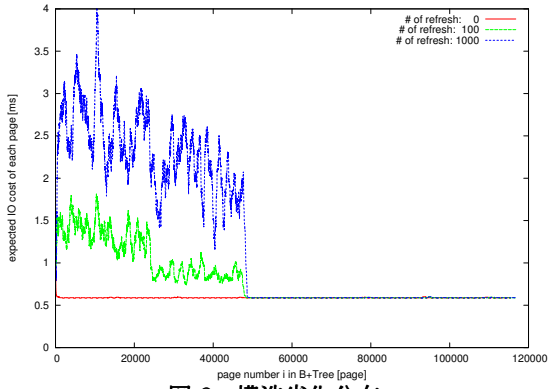

図 6　構造劣化分布

Fig. 6　Structural deterioration distribution

験環境として，Linux が動作する PC 上でディスクドライブ 1 台の上に MySQL[5] 4.1 InnoDB データベース表空間を作成した．ページサイズは 16KB とし，クラスタ表 T1 (int id clustered key, int id2, char str[255])を作成した．クラスタ鍵 id を，$\{id \mid 0\leq id<6000000\}$ として，重複がないように 600 万行ロードした．上記の環境で，以下の DB 更新及び，範囲検索を実行した．

DB 更新: クラスタ鍵に対して特定範囲を指定し，指定された範囲内でランダムにバルク行を削除，挿入する操作を，領域毎に異なる頻度で実行する．以上の操作を DB に対する更新 1 回とし，1000 回の更新を行った．1000 回後，各行が更新された回数を，横軸にクラスタ鍵値をとって図 5 に示した．DB 更新に伴い，構造劣化モニタを用いて，構造劣化分布 $\mathbf{G}$ を取得した(図 6)．$\mathbf{G}$ は更新分布と対応していることが確認できる．

範囲検索: 全クラスタ鍵範囲を，10%ずつ 10 の範囲に分け，それぞれ範囲検索クエリを実行し，応答時間を測定した．

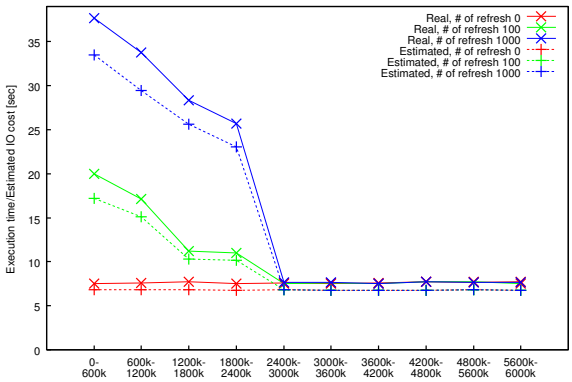

図 7　範囲検索性能と推定値

Fig. 7　Rangescan performance and estimation

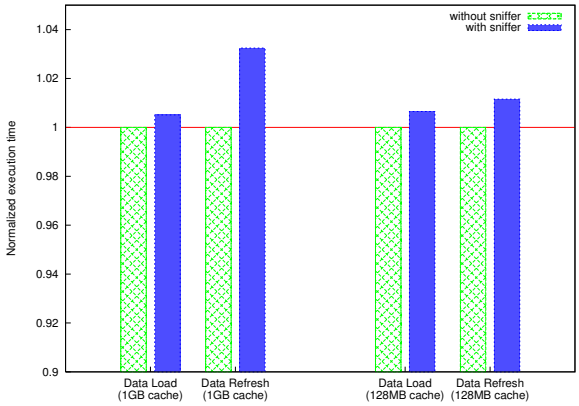

図 8　DB 差分抽出器のオーバーヘッド

Fig. 8　Overhead of DB update sniffer

範囲検索クエリの応答時間とその構造劣化モデルによる推定値を図 7 に示した．推定値との誤差は 8～17%程度であった．ただし，推定モデルは範囲検索における IO 時間しか考慮しておらず，MySQL 処理時間が 10%程度あることが分かっているため，実際の推定誤差は 7%未満である．

図 8 に，DB 更新差分抽出器のオーバーヘッドの評価結果を示した．バッファキャッシュを 1GB と 128MB の二通りの設定を用いて評価を行った．その結果，データロード時は 1GB キャッシュ時に約 0.5%，128MB キャッシュ時に約 0.7%であり，データ更新時は 1GB キャッシュ時に約 3.2%，128MB キャッシュ時に約 1.2%のオーバーヘッドが測定された．データサイズが約 2GB で，更新されるデータがその 40%であるから，1GB キャッシュ時には，データページはほぼ全てバッファキャッシュ上にあることになり，オーバーヘッドは取りうる最大値であると考えられる．以上により，低オーバーヘッドで，リアルタイムに構造劣化を監視できることが示された．

図 6 において，$X = 2$ を式(3)与えて SLA を設定すると，更新 100 回の時点で，論理空間 $\{i \mid 0 \le i < 24000\}$ すなわちクラスタ鍵範囲 $\{id \mid 0 \le id < 600000\}$ の空間がもはや SLA を満たさないであろうことが分かる．この領域を再編成スケジューラは再編成対象領域 **T** として同定できる．

5. おわりに

本論文は、DBMS において自立再編成機構を実現することを目的として，低オーバーヘッドなリアルタイム構造劣化モニタ，及び SLA ポリシに基づいた再編成スケジューラを提案した．評価において，構造劣化モニタが低オーバーヘッドで高精度に構造劣化を監視できることを示し，再編成スケジューラの適用方法について考察した．

今後，再編成実施手法についても考察し，コスト対効果の高い再編成スケジューラの構築を目指して研究を進めたい．

[謝辞]

本研究の一部は，文部科学省リーディングプロジェクト e-society 基盤ソフトウェアの総合開発「先進的なストレージ技術」の助成により行われた．協力企業である株式会社日立製作所より多くの有益なコメントを頂戴した．深謝する次第である．

[文献]

[1] Batory, D. S.: “Optimal file designs and reorganization points”. ACM TODS 7(1), pp.60-81 (1982).
[2](Ed.) Lomet, D.: “Special Issue on Online Reorganization”. IEEE Data Eng. Bull. 19(2) (1996).
[3] Lightstone, S., Schiefer, B., Zilio, D. and Kleewein, J.: “Autonomic Computing for Relational Databases: The Ten Year Vision”. In Proc. of AUCOPA2003 (2003).
[4] Mackert, L. F., and Lohman, G. M.: “Index Scans Using a Finite LRU Buffer: A Validated I/O Model”. ACM TODS 14(3), pp.401–424 (1989).
[5] MySQL: The World’s Most Popular Open Source Database. http://www.mysql.com/.
[6] 合田和生，喜連川優．：“データベース再編成機構を有するストレージシステム” 情報処理学会論文誌データベース，46(TOD (SIG8)) pp.130-147 (2005).
[7] 星野喬，合田和生，喜連川優．：“データベース更新差分を用いた範囲検索の IO コスト推定” DBSJ Letters, 4(2), pp.37-40 (2005).

星野 喬　Takashi HOSHINO
東京大学大学院情報理工学系研究科博士課程在学中．日本学術振興会特別研究員 DC. 2003 年東京大学工学部電子情報工学科卒業．2005 年東京大学大学院情報理工学系研究科修士課程修了．データベースシステムの研究に従事．情報処理学会，日本データベース学会学生会員．

合田 和生　Kazuo GODA
東京大学生産技術研究所特任助手．2000 年東京大学工学部電気工学科卒業．2005 年東京大学大学院情報理工学系研究科博士課程単位取得満期退学．博士(情報理工学)．並列データベースシステム，ストレージシステムの研究に従事．本会，情報処理学会，ACM，USENIX 会員．

喜連川 優　Masaru KITSUREGAWA
1978 年東京大学工学部卒．1983 年同大大学院工学系研究科情報工学博士課程了．工学博士．同年同大生産技術研究所講師．現在，同教授．2003 より同所戦略情報融合国際研究センター長．データベース工学，並列処理，Ｗｅｂマイニングに関する研究に従事．現在，日本データベース学会理事，情報処理学会，電子情報通信学会 各フェロー．平成 11-14 年 ACM SIGMODJapan Chapter Chair, 平成 9,10 年電子情報通信学会データ工学研究専門委員会委員長．VLDB Trustee (97-02), IEEE ICDE, PAKDD, WAIM などステアリング委員, IEEE データ工学国際会議(ICDE2005) General Chair.