

データベース性能診断システム

馬場 正和 中野 勝之 池田 幸雄

(N T T 横須賀電気通信研究所)

1. はじめに

大規模、高トラフィックなオンライン・データベース(DB)・システム分野では、性能が重視されるため、主にCODASYL型DBが採用されているが、①CODASYL型DB管理システム(DBMS)は言語仕様が複雑であること、②DB構築後のDB構造変更が困難なため、設計時に十分な評価、構造上の改善の実施が要求されること等の理由により、設計作業は専らDB設計専門家の手で行なわれている。しかし、DBシステム数の急増に伴いDB設計専門家の不足、DB設計品質の低下が深刻な問題となっており、非専門家でも簡易に性能の良いDBが設計できる支援機能の開発が望まれている。

従来、CODASYL型DBの設計支援法としては

①性能を考慮したスキーマを自動生成する方法

例えば、与えられた格納条件(データ項目とその量、項目間の連関、APがデータ項目へアクセスする頻度等)のもとで、レコードアクセス回数を最小にするようデータ項目をレコードタイプとセットハグループ化する方法^[1]

②利用者定義のスキーマのうち一部定義項目を最適化する方法

例えば、競合する設計目標(検索時間の短縮、データベース更新コストの低減、必要格納域の削減、システムトータルコストの低減)に対する優先度、重み付けのもとに各レコードタイプ毎に最適なレコードプレースメントを求める方法^[2]

等が提案されているが、これらを現実のDBに適用しようとする、①の方法では、DBの利用条件やスキーマの作成条件が十分表現できないため、現実の状況に適したスキーマが出力できない、表現の詳細化を試みると投入データ量が多くなり一般の設計者には負担が大き過ぎる、DBのデータ項目数が多くなると計算機時間が膨大となる、等の問題があり、実用化の段階には至っていない。また、②の方法では、利用者が定義したスキーマを対象とするので①の問題はないが、一部の定義項目にしか着目しないので、スキーマ構造全体を見渡した設計には機能不足である。

そこで、我々は、利用者定義のスキーマを対象として、スキーマ構造上性能面で少しでも改善の余地があれば、それを網羅的に検出し提示するという考え方にたち、DB設計専門家のスキーマの設計に関するノウハウを知識ベース化することにより、人手により設計されたスキーマ上の性能劣化要因を自動検出し、改善策を提示するDB性能診断システム^[3]を検討した。

本稿では、DB性能分析/改善の作業モデル、診断システムのシステムモデルと、システム実現のための中核となる性能診断知識、診断対象データの表現法、改善結果の競合解決法について述べる。

2. DB性能分析/改善の作業モデル

診断の対象であるCODASYL型DBは、レコード、レコードとレコード間の関係を表すセットで構成され、目的とするレコードへのアクセスは、セットを経由して行われる。すなわち、構造アクセスが行われ、セット選択、セット順序、セットポイントの選択、インデックスの有無等がDBアクセス性能に大きく影響するため、これらの修正によりアクセス性能を改善できる場合が多い。

DBスキーマの性能分析/改善は、一般的に図1に示すように、使用頻度が高く、問題点の存在しそうなアプリケーションプログラム(AP)に着目し該APのアクセスするスキーマの構成要素(セット/レコード等)について、設計者のノウハウに基づいた仮説(性能改善に有効だと思われるセット選択、セット順序等の修正及び構造の変更法)を立て、アクセス特性に関するデータ等をもとに、該仮説が成立するかどうか検証することによって行われている。

3. DB性能診断システムのモデル

DB性能診断システムは、上記仮説を知識ベース化することにより、性能上の問題点を自動検出し改善法を提示することを目的としたエキスパートシステムである。

DB性能診断システムのモデルを図2に示す。DB設計専門家から獲得した性能劣化要因の検出及び改善法に関する知識(性能診断知識)を知識ベースに格納する。推論機構は、この知識ベースを利用してDB設計者が作成したスキーマ、レコード、セット等のDBへの格納件数を指定

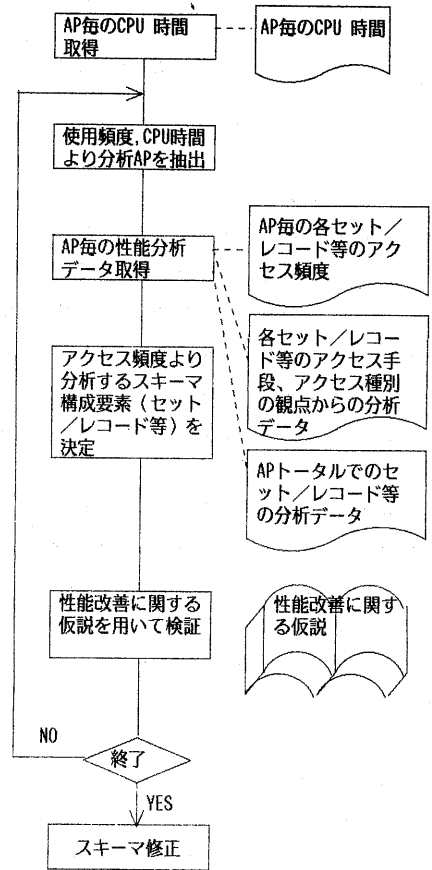


図1 性能分析/改善手順

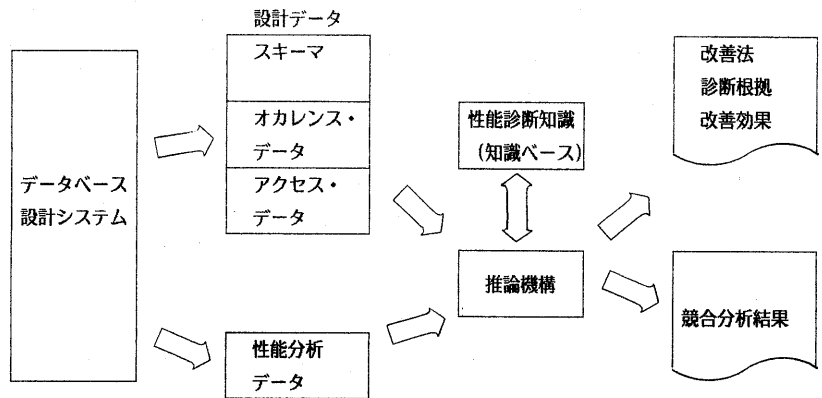


図2 システム・モデル

したオカレンスデータ、APがDBへアクセスするDB操作言語(DML)の系列をモデル化したアクセスデータ、及びそれらの情報をもとにDB性能分析に必要な特性データを収集する機構^[4]を用いて収集した性能分析データ(格納状況、アクセス特性を表わす情報)を用いて診断する。

スキーマに問題(性能ネック)箇所があれば検出し、改善法、推論内容を示す診断根拠を提示する。改善案の採否については、システム運用条件等、性能面以外の要因も考慮する必要があり、システムで自動的に判定するのは困難なため、スキーマの修正作業と共に設計者に委ねる。システムでは、そのための情報として、スキーマを修正した場合に得られる性能尺度(アクセス回数、I/O回数、DB容量)に関する改善効果量及びデータ記述言語の構文則上同時に適用可能かどうかの情報(競合分析結果)を提示する。

利用者は、それらの情報を用い、目的に応じてアクセス効率と記憶容量の両視点から評価し、スキーマ構造の最適化を行う必要がある。(例えばポインタ、インデックスを付与する場合、DBのアクセス時間は短縮できるが記憶容量は増加する。)

診断知識は、DB設計専門家の経験的ノウハウであり、体系的獲得が困難なため段階的に追加されるものである。したがって、性能診断システムの実現には、知識の追加、修正が容易に行えること、知識追加時プログラムの修正を伴わないこと、が要求される。

4. 性能診断知識の表現法

DB設計専門家は、2章で述べたように設計されたスキーマに対して、APのDB操作命令やレコード/セットのオカレンス件数等を用いて、DB操作命令の動作のシミュレーションやレコード/インデックス等の容量の計算を行い、該データを基に過去に遭遇した性能改善事例や、DBMSのマニュアル等から得た知識(仮説)に基づきスキーマの問題箇所を探索し、改善する。例えば、セットの特定レコードを検索する場合、通常インデックスの付与により性能が改善されるが、メンバレコード・オカレンス数が非常に多く、また、アクセス箇所が集中している場合は、インデックス経由のアクセスよりもセットチェーンサーチの方が性能上良いケースがある。したがってスキーマの性能診断知識はスキーマの問題箇所を特定する条件とその解決策から構成される。即ち、性能診断知識は、多くのエキスバ

性能診断知識

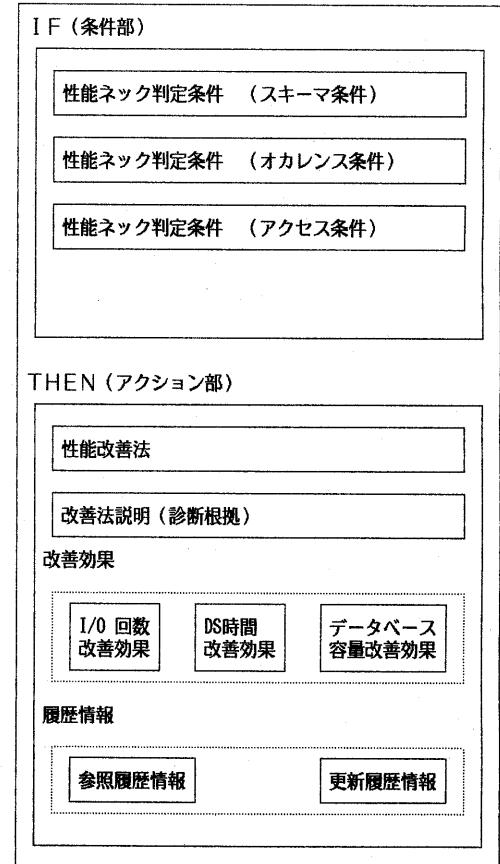


図3 性能診断知識の構成図

ート・システムで採用されているIF… THEN…の構文を持つプロダクション・ルール形式で表現可能である。また、プロダクションシステムとして実現することにより知識の追加、修正も容易になり前記システムへの要求条件が満足される。

図3に示すように、条件部に記述する性能ネック判定条件は、スキーマ条件、オカレンス条件、アクセス条件の論理積からなる。スキーマ条件はデータ記述言語の指定による問題箇所候補を検出する条件である。オカ

レンス条件、アクセス条件は、それぞれ格納状況を表わすデータ、アクセス特性を表わすデータを用いて指定するネック判定条件である。

即ち、各ルールで検出する性能ネック箇所はレコード、データ項目、セット等で指定されるデータ記述言語上の定義項目（例えば、セットのプライマポインタの選択）である。性能診断知識はルール作成容易化のため、原則として、1ルールではデータ記述言語上の1定義項目を診断する構成としている。知識ベースはこれらのルールの集合である。

アクション部には性能改善法、診断根拠説明、各性能尺度に関する改善効果、改善法の競合分析のための診断履歴情報を記述する。

性能診断知識は、データ記述言語の各定義項目に着目して表1に示す観点で抽出した。

表1 性能診断知識抽出の観点例

構造	改善対象	観 点 例
論理構造	レコード	オーナ・メンバレコードのエリア格納方法 (例、同一エリア、別エリア)
	データ項目	データ副記述項の選択(例、DEC(4)よりBIN(15)で表現した方がメモリ容量が少ない)
	セット	セット順序の選択 マルチメンバセットでのセット分離
格納構造	エリア	ページ数の見直し
	インデックス	インデックスの付与、削除
	レコード	ロケーションモードの変更
	セット	ポインタの付与、削除

5. 診断対象データの内容と表現法

診断対象スキーマは、通常、複数のセット、レコードで構成されるので、同一定義項目に関する診断すべき箇所が複数存在する可能性（例えば、セットA、セットBへのプライマポインタの付与）があり、1ルールで複数箇所を診断する必要がある。そのため、知識の表現は集合変数を用いて行い、診断実行時に変数にすべての具体値を代入してルールを評価する方法で実現することにした。これらの条件を考慮し、本システムでは、診断単位に関する情報をクラスとして定義し、その実現値をインスタンスとするフレーム表現により診断対象データを表わす。（プロダクションシステムでは、ルールがアクセスする情報、ここでは診断対象データはワーキングメモリに格納される。）診断知識中のネック判定条件は、このクラス名を参照する形で記述しておき、診断実行時には、対応インスタンスをワーキングメモリ

上で走査することにより、スキーマ全体の診断を行う。

DBスキーマの構成要素であるセット、レコード、データ項目等の情報間には種々の関係があり、診断知識表現にもこの関係が使用される。効率的な推論を行うためにはクラス間の関係を構造的に表現することが望ましい。(例えば、あるレコードへのトータルなアクセス情報とその内訳であるDML毎のアクセス情報は1:Nのクラス関係で表せる。)したがって、本システムは構造化されたワーキングメモリを持つプロダクションシステム^[5]上で実現する。

診断対象データは、データ記述言語で定義されたスキーマ情報とデータベース設計システムで収集される以下の性能分析データからなる。

(1) 動作状況分析データ

CODASYL型DBへのアクセスは、データ構造に従ったアクセスが主体であり、アクセス経路の選択によっては冗長になり、その冗長性を取除くことが性能改善に繋るので、目的レコードへのアクセスとそうでないものと区別することにより、冗長性の分析が容易となる。そこで、動作状況分析データはアクセス回数、実I/O回数を、目的とするレコードオカレンスへ辿り着くまでのアクセスと目的レコードへのアクセスとを区別したデータ、その他、APによりDBがアクセスされる際のアクセス手段、セット、レコードへのアクセス分布、アクセス種別等の動作状況を認識できるデータで構成される。

(2) 格納状況分析データ

DBは、レルムと呼ばれる記憶領域に分割され、更にレルムはページと呼ばれる単位に分割される。記憶効率を改善するためには、データの格納密度を向上させる必要があり、そのためレルムやページのレコード分布情報が必要となる。この情報は、ページが主記憶との転送単位であることからアクセス性能の分析にも有効である。以上の点を考慮して、格納状況分析データは、ページ内オカレンス数、セットオカレンスの占有するページ数等で構成される。

6. DB性能診断システムの競合解決法

プロダクションシステムは知識表現が簡易な反面、発火可能なルールが複数存在する場合、その中から1つのルールを選択する競合の解決法(conflict resolution)の設計が重要な課題である。

一般に、プロダクションシステムの競合解決法として、以下の様な手法があり、システムごとに、これらの簡単な競合解決手法をそれぞれ異なった組合せで使っている。^{[6], [7]}

- (1) ルールをリストにしておき、最初に照合したルールを実行する。
- (2) 最も厳しい条件を持つ(条件部の構成要素の数が多)ルールを優先する。
- (3) 最も最近に使われたルールを優先する。
- (4) 実行時に各ルールに対して評価値を計算し、最大の評価値が得られたルールを適用する。
- (5) 発火可能なルールはすべて発火させる

DB性能診断システムには以下に示す特徴があり、上述の方法の単なる組合せでは競合解決ができず、処理を2フェーズ化し、それぞれのフェーズで(5)と(1)の方法を別々に用いることによって解決している。

- ① 1ルールでは1改善法を提示する。
- ② 1スキーマには複数の改善候補があり、それらの組合せにより改善が行われる（一候補による改善もある）。
- ③ スキーマ構造上の制約により、改善候補を同時に適用できないケース（あるセットのセット順序の変更とセットの削除）がある

即ち、構造上の制約があるので、具体的なスキーマ上の改善箇所（発火後の全改善候補）を見ないと同時に適用可能かどうか判断できない。

そこで、DB性能診断システムでは図4に示すように、フェーズ1では発火可能なルールは全て発火させ全改善候補を洗いだし、アクション部（RHS）実行時に同時に適用可能かどうかをチェックするための履歴情報を取得する。フェーズ2では

履歴情報をもとに同時に適用可能かどうかのチェックを行い、競合解決を図っている。

本システムのような競合解決法の要求される分野にプロダクションシステムの適用域を広げるためには、プロダクションシステムの競合解決処理に、問題領域に依存した制御知識を指定できることが必要である。即ち、条件部（LHS）に性能ネックかどうか判定するための条件以外に競合解決に必要なデータを記述可能とし、さらに競合解決用のAP特有のロジックが組みこめる機構が用意されていれば、競合解決処理が1度の診断処理

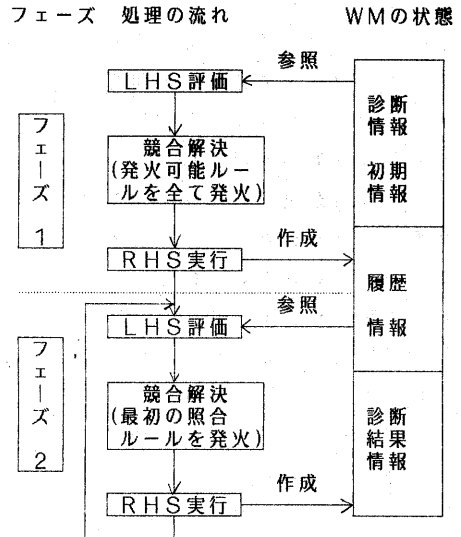


図4 診断処理の流れ

性能改善法 a, b の関係	参照/更新域の関係	備考
同時適用可	(ケース1) 参照域が重ならない R (a) R (b) 	両方の性能ネック改善法を適用可能
	(ケース2) 更新域以外の参照域が重なる R (a) R (b) 	
同時適用不可	(ケース3) 参照域が更新域に重なる R (a) R (b) 	いずれか一方の性能改善法を適用可能
	(ケース4) 更新域が重なる R (a) R (b) 	

R (a)、R (b) : 性能改善法 a、bの参照域
 U (a)、U (b) : 性能改善法 a、bの更新域
 参照域/更新域 : 性能改善法で参照/更新する内容

図5 性能改善法の同時適用性

フェーズで行えプロダクションシステムの適用領域をより広げることが可能と考える。[8]
次にフェーズ2で行っている本システム特有の競合解決の考え方について述べる。

複数の性能改善法を同時に適用しようとした場合、参照域（性能上問題があるかどうか判定するために参照するスキーマ定義情報）と更新域（性能を改善するために更新するスキーマ定義情報）、更新域と更新域が重なり矛盾が生じるケースがある。（図5参照）

本ケースを検出するためには、参照域と更新域の重なり、更新域同士の重なりを見出さなければならない。そのために参照および更新情報を履歴情報として保持する。

同時に適用可能かどうかの分析の方法としては以下に示すように履歴情報のパターン一致で分析可能なケースと分析にロジックが必要なケースがある。

(1) 履歴情報のパターン一致で分析可能なケース

①更新内容および選択肢の相違

スキーマ定義情報（データ記述言語の記述項、節を含む）の更新内容が従来の定義内容と異なる、および同一箇所への改善内容が異なるケース

②否定的表現

①で定義内容および更新内容を否定的に表現している場合、同時に適用できない可能性があるかと判定できるケース

③記述間の包含

データ記述言語の記述項（エリア、セット、レコード）に対し、現在定義されているものを削除する場合、この記述項の配下に定義されている節の記述は全て無効になる。この記述項の包含により同時に適用できないケース

(2) 分析にロジックが必要なケース

①構造を扱っている表現

ルール上の表現が下記のようなスキーマの構造を扱っている場合の競合の有無
・セット選択段数 ≤ 2 という表現、セット選択パスを $S_1 \rightarrow S_2 \rightarrow S_3$ にするという表現のケース

②内容の意味

表現上は異なっているが、データ定義言語上関連を持ち、仕様上の意味を考える必要があるケース

7. おわりに

DB設計専門家のノウハウを知識ベース化し、スキーマの性能劣化要因を検出し改善策を提示するDB性能診断システムについて述べた。従来、DBの性能評価/改善はDBMSの内部仕様を良く理解している専門家が机上で実施していたが、本システムを用いることにより、言語仕様を理解していれば、DBMSに通じていなくても高性能な設計が可能となる。

現在、本システムをKBMS^[5]上で実現し評価を実施中である。

本システムをさらに高度化するためには、DMLシーケンスの良否を評価するために必要な知識、複数の改善法の改善効果のトレードオフ分析を行うための知識等を導入していく必要があると考えている。

[参考文献]

- [1] Irani, K.B. etc. "A DESIGNER FOR DBMS-PROCESSABLE LOGICAL DATABASE STRUCTURE"
5th V.L.D.B pp.219-231, 1979
- [2] Jain, H.K. etc. "Computer-aided system for the database storage structure
design " Information management, Vol.6, No.6, pp.337-349, 1983
- [3] 馬場、池田 "データベース性能診断システムの実現法"
情報処理学会第30回全国大会、7U-3, 1985
- [4] 池田、中野、馬場、山多 "データベース設計システムDBDESIGN"
情報処理学会データベースシステム研究会、45-1, 1985
- [5] 服部、清水、和佐野他 "知識ベース管理システム(KBMS)"
情報処理学会知識工学と人工知能研究会、41-6, 1985
- [6] 田中、淵 監訳 "人工知能ハンドブック" 共立出版
- [7] 白井、辻井 "人工知能" 岩波講座情報科学-22
- [8] 馬場、中野、山多 "データベース性能診断システムにおけるルールの競合解決法"
情報処理学会第31回全国大会、9M-2, 1985