TC11-1

アウトオブオーダ型データベースエンジン OoODE の試作とディスクストレージを用いた実験的性能評価

合田　和生*　　早水　悠登　　山田　浩之（東京大学）
喜連川　優（国立情報学研究所，東京大学）

Prototypes of Out-of-Order Database Engine and
Their Experimental Performance Evaluation using Disk Storage

Kazuo Goda*, Yuto Hayammizu, Hiroyuki Yamada (The University of Tokyo)
Masaru Kitsuregawa (National Institute of Informatics / The University of Tokyo)

Out-of-Order Database Engine (OoODE), a high-speed database engine we have proposed, has a novel mechanism to introduce a new opportunity for the database engine to execute the query processing with a massive number of concurrent tasks and associated asynchronous IOs. This paper presents our work on prototype implementation of OoODE on the basis of open-source database software and investigates its performance benefits by showing our experimental results we obtained in enterprise-level disk storage environments.

キーワード：データベースエンジン，データベースシステム，問合せ処理，性能評価
(Keywords: database engine, database system, query processing, performance evaluation)

1. はじめに

データベースシステムを初めとするデータインテンシブな IT システムのハードウェア技術の潮流としては，多数のプロセッサコアを集積したマルチコアプロセッサと，多数のディスクドライブを集約したディスクストレージからなるシステムが主流であり，当該傾向はハイエンド環境からミッドレンジ環境へと波及している．データベースアプライアンス[10,11]と称される並列分散型のシステム構成が登場し，利用が拡大しているものの，各ノードはマルチコアプロセッサとディスクストレージから構成されており，同傾向は変わらない．市場においては，プロセッサコア数の増加は今後も進展するものとみられており，また，所謂ビッグデータブームに牽引され[8]，ストレージシステムの大型化・高密度化が著しい．これらの莫大な数のプロセッサコアとディスクドライブを有効活用することは，巨大データの管理を担うデータベース技術にとって，必須のものと言えよう．

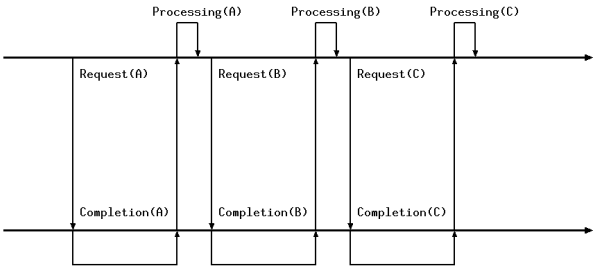
著者らは，アウトオブオーダ型データベースエンジン (OoODE) と称する高速データベースエンジンの開発を進めている[20-24]．当該データベースエンジンは，問合せ処理を動的にアンフォールドし，すなわち，タスク分解することにより，高多重のタスク並行実行ならびに高多重の非同期入出力発行を可能とする点に特徴を有している．マルチコアプロセッサの演算パワーとディスクストレージの入出力帯域を高効率に活用することを実現することにより，中程度の選択性を有する問合せにおいて，従来型の逐次的な実行方式によるエンジンと比べて，問合せ処理の高速化が期待されている．本論文では，これまで著者らがオープンソースのデータベースソフトウェアをベースとして進めてきたアウトオブオーダ型データベースエンジンの 2 例の試作実装を示すとともに，それぞれの実装に関して，ディスクストレージを用いた環境における実験により当該データベースエンジンの有効性を明らかにする．

2. アウトオブオーダ型データベースエンジン

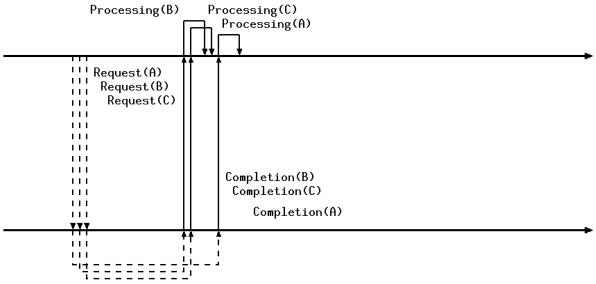
Ingres[13]等初期の実装以来，多くのデータベースエンジンにおいては，「インオーダ型」の実行方式によって問合せを処理してきた[4,18,20]．データベースエンジンは，問合せを受け付けると，問合せ実行計画を生成する．通常，問合せ実行計画は実行木と呼ばれる木構造で表され，データベースエンジンは実行木において演算ノードを逐次的に辿って実行することにより，当該問合せを処理する．演算ノードの実行においては，対象となるデータをデータベースから

取得する必要がある場合，データベースが格納されたストレージに入出力命令を発行しその完了を待って演算を実行することを，対象となる全てのデータを処理し終えるまで繰り返す．すなわち，当該方式においては，演算ノードにおいて入出力と演算を逐次的に繰り返し，実行時間としては，入出力にかかる応答時間が蓄積することとなる．

これに対して，著者らの考案した「アウトオブオーダ型」の実行方式では，問合せ処理の実行時に，演算ノードにおいて新たな入出力を発行する必要が生じると，都度にタスク分解を行い，分解された並行実行可能なタスク上で入出力と関連する演算を行う[20]．問合せ処理の実行時に，実行論理の許す限りにおいて，必要に応じてタスク分解が行われ，多数の非同期入出力がストレージに発行されることとなる．実際には，資源量は有限であるため，同時処理可能な非同期入出力の数の上限が制約されるものの，最近のサーバにおいては数百 GB 程度の主記憶容量は珍しくなく，またストレージシステムも数百のディスクドライブを搭載するに到っており[5,6,15]，従来に対してより多くのタスクと入出力の同時処理が可能となってきている[12,19]．従来のインオーダ型の実行方式の下では，プロセッサ性能と主記憶容量を節約するべく，極めて少量の入出力のみが発行されていたのに対し，アウトオブオーダ型の実行方式においては，ソフトウェアの非同期化によって[14]，実行論理が許す限り大量の入出力を発行することを可能とし，これにより，豊富な演算パワーを備えたマルチコアプロセッサコアと膨大な数のディスクドライブの並列アクセスによって，効果的な性能バランスの達成を狙う．



(a) インオーダ型実行



(b) アウトオブオーダ型実行

図 1. 実験結果＃ 1

Figure 1. Experimental Result #1

データベースのアクセスパスという観点では，データベースエンジンは，大きく分けて，リレーション全体の走査か，索引経由のアクセスかの選択を行う．リレーション間の結合は，問合せ実行時間の多くを占める場合が多い演算であるが，基底となるリレーションに対する選択性に基づき，問合せ最適化器によって，ハッシュ結合[2,7]とネステッドループ結合のうちいずれかが選択され，用いられてきた．前者は，リレーション全体の走査を基本とし，選択性が低い場合に有効とされるが，リレーション規模がテラバイトからペタバイトと巨大化するにつれ，単純な走査が可能な機会はより限られてくるはずであり，より選択性の高い問合せの重要性が高まる可能性が高い．すなわち，今後，ネステッドループ結合に頼る比重が増えるものと推察され，アウトオブオーダ型の実行方式は，当該結合方式に対して，圧倒的な高速化ポテンシャルを有している．

3. アウトオブオーダ型データベースエンジンの試作実装と当該実装を用いた実験による性能評価

著者らは，複数のオープンソースのデータベースソフトウェアをベースとしたアウトオブオーダ型データベースエンジンの試作実装を進めてきている．本論文では，まず，その一実装である MySQL をベースとする試作実装を紹介する[21,23]．

当該実装は, InnoDB ストレージエンジンデータフォーマット[9]によって構成されたデータベースに対する問合せを受け付け，これを処理するものである．この際，演算を実行するのに入出力を発行する必要が生じると，都度にタスク分解を行い，分解された並行実行可能なタスク上で入出力と関連する演算を行う．タスクとしては，オペレーティングシステムが提供するカーネルスレッドと，データベースエンジンのユーザ空間で提供するユーザスレッドを組み合わせて実現している[22]．

表 1 実験システム＃ 1

Table 1. Experimental System #1

サーバ装置 IBM X3850 M2	
プロセッサ	4x Intel Xeon (4p/24c)
主記憶	32GB （ただし，1GB のみ利用）
OS	RedHat Enterprise Linux 5.8 x86_64
HBA	8x Emulex 8Gbps FCs
ディスクストレージ装置 IBM DS5300	
コントローラ数	2
キャッシュ容量	8GB
ディスクドライブ	160x 15Krpm 450GB SAS HDDs

著者らは，当該実装の性能を評価するために，表 1 に示す実験システムを構築した．この際，ディスクアレイのコントローラにより 7D+1P のパリティグループを編成し，これによって構成された 20 個の LU を，オペレーティングシ

ステムの持つ RAID 機能によってパリティなしストライピングによる集約を行い，これをデータベース空間として利用した．

実験には TPC-H ベンチマーク[16]データセット（スケールファクタ 100）を用いることとし，また，問合せとしては当該ベンチマーク規定の Q.3 を簡易化したものを用い，その実行時間を計測した．また，比較対象として，代表的なオープンソースデータベースソフトウェアである MySQL 5.5 ならびに PostgreSQL 9.2 においても同じデータセットに対して同じ問合せを実行し，実行時間の計測を行った．

計測結果を図２に纏める．縦軸（実行時間）は対数でプロットされていることに注意されたい．MySQL 5.5 は結合方式としてネステッドループ結合を用いるのに対して，PostgreSQL 9.2 で当該方式に加えてハッシュ結合を備えていることからこの両方式と比較した．いずれの方式と比較しても，著者らの提案するアウトオブオーダ型データベースエンジンが，計測を行った選択率付近において，飛躍的な高速化を達成していることが見て取れる．

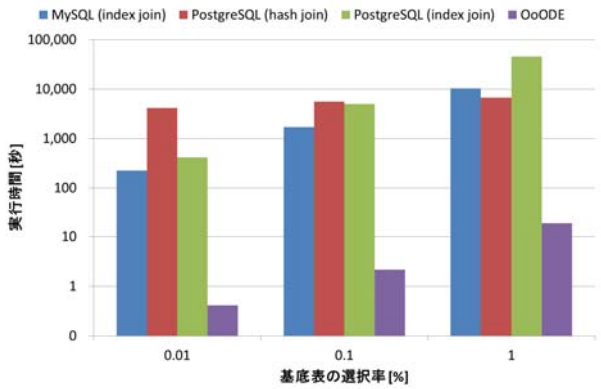


図２． 実験結果＃１

Figure 2. Experimental Result #1

上記は，所謂ノンクラスタ型のデータベースシステムとして MySQL ベースの試作実装であったが，著者らは，アウトオブオーダ型なる提案技術のより広範なシステムへの有効性検証のための一実装として，並列データ処理フレームワークである Hadoop をベースとするアウトオブオーダ型データベースエンジンの試作実装を進めてきている[24]．

当該実装は，既存の Hadoop エンジンに対して，アウトオブオーダ型実行方式による Hadooode なるエンジンを組込んだものである．当該エンジンにより，アウトオブオーダ型実行方式による Map/Reduce ジョブ[1]の実行を実現するほか，Hive 等の上位系に対応し HiveQL 等による問合せをアウトオブオーダ型実行方式によって処理する機能を備えている．なお，この際，索引アクセス等のデータベース技法を合わせて取り込んでいる．

著者らは，当該実装の性能を評価するために，表２に示す 21 ノードからなる実験システムを構築した．このうち，20 ノードを実行ノード，1 ノードをスーパバイザノードとして利用した．この際，各ノードにおいては，内蔵型のアレイコントローラにより 22D+2P のパリティグループを編成し，これによって構成された LU に対して，ファイルシステムを構成し，これをデータベース空間として利用した．

表２ 実験システム＃２

Table 2. Experimental System #2

20x 実行ノード，スーパバイザノード Dell PowerEdge R720xd	
プロセッサ	2x Intel Xeon (2p/16c)
主記憶	64GB
OS	CentOS Linux 5.8 x86_64
ディスクドライブ	2x 10Krpm 900GB SAS HDDs（OS 用） 24x 10Krpm 900GB SAS HDDs（データベース用）
アレイコントローラ	PERC H710p
NIC	Intel 1Gbps Ethernet
スイッチ Dell PowerConnect 5548	
ポート	48x 1Gbps Ethernet Ports

実験には同じく TPC-H ベンチマークデータセット（スケールファクタ 20,000）を用いることとし，また，問合せとしては当該ベンチマーク規定の Q.8 を簡易化したものを用い，その実行時間を計測した．また，比較対象として，Hadoop において同じデータセットに対して同じ問合せを実行し，実行時間の計測を行った．一般に，Hadoop は対象データセットの単純走査を基本としているが，昨今，Hadoop に索引アクセス等のデータベース技法が取り入れられている状況を鑑み[3,17]，ネイティブの Hadoop に加えて，Hadoop に独自に索引機構を取り込む実装を行い，当該実装とも比較を行った．

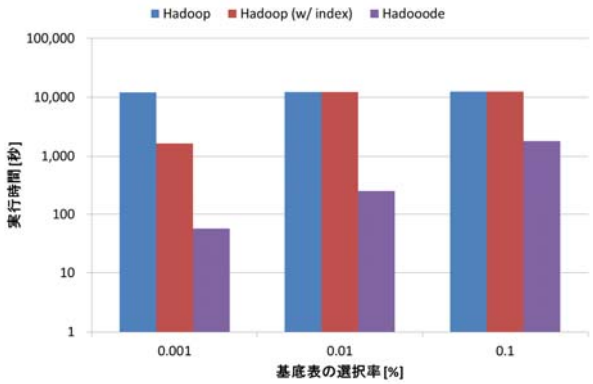


図３． 実験結果＃２

Figure 3. Experimental Result #2

計測結果を図３に纏める．同じく縦軸（実行時間）は対数でプロットされていることに注意されたい．Hadoop は，

先述の通り，対象データの単純走査を基本としており，また，結合方式としては整列併合結合を用いる．索引アクセス機能を付加した Hadoop 実装においては，加えて並列ネステッドループ結合を選択できるようにしてある．これに対して，Hadooode においては，更に加えて，アウトオブオーダ型実行方式による並列ネステッドループ結合が用いられる場合があり，特に選択性の高い問合せについては当該方式の有効性が高い．実験結果によれば，いずれの方式と比較しても，著者らの提案するアウトオブオーダ型実行方式により，計測を行った選択率付近において，著しい高速化を達成していることが見て取れる．

4. おわりに

本論文では，これまで著者らがオープンソース DBMS をベースとして進めてきたアウトオブオーダ型データベースエンジンの 2 例の試作実装を示すとともに，それぞれの実装に関して，ディスクストレージを用いた環境における実験により当該データベースエンジンにより従来型のデータベースエンジン実装と比較して著しい高速性が得られることを明らかにした．現時点では実装は荒削りなものであり，今後は更にその錬度を高めてゆきたい．また，本論文ではエンジンのカーネル部分に絞って，構成法と有効性を議論したが，他のコンポーネントである問合せ最適化器，更新・リカバリ機構，レプリケーション機構等に関しても，今後，順次報告してゆきたい．

謝辞

本研究の一部は，内閣府最先端研究開発支援プログラム「超巨大データベース時代に向けた最高速データベースエンジンの開発と当該エンジンを核とする戦略的社会サービスの実証・評価」の助成により行われた．

文　献

（1） J. Dean, S. Ghemawat: “MapReduce: Simplied Data Processing on Large Clusters", Proc. OSDI, pp. 137-150 (2004)

（2） David J. DeWitt, Robert H. Gerber, Goetz Graefe, Michael L. Heytens, Krishna B. Kumar, and M. Muralikrishna: “GAMMA - A High Performance Dataflow Database Machine”, In Proc. Int'l. Conf. on Very Large Data Base, pp. 228-237 (1986)

（3） J. Dittrich, J. A. Quiane-Ruiz, A. Jindal, Y. Kargin, V. Setty, and J. Schad. “Hadoop++: Making a Yellow Elephant Run Like a Cheetah (Without It Even Noticing)”, Proc. VLDB, pp. 515-529 (2010)

（4） R. Elmasri and S. Navathe: Fundamentals of Database Systems. Addison Wesley (1994)

（5） Kazuo Goda and Masaru Kitsuregawa: “The History of Storage Systems”, Proc. of IEEE, Vol. 100, No. Special Centennial Issue, pp. 1433-1440 (2012)

（6） Masaru Kitsuregawa, Kazuo Goda, and Takashi Hoshino: “Storage Fusion”, In Proc. of 2nd Int'l Conf. on Ubiquitous Information Mgmt. and Comm., pp. 287-294 (2008)

（7） Masaru Kitsuregawa, Hidehiko Tanaka, and Tohru Moto-Oka: “Application of Hash to Data Base Machine and Its Architecture”, New Generation Comput., Vol. 1, No. 1, pp. 63-74 (1983)

（8） McKinsey Global Institute: Big data: The next frontier for innovation, competition, and productivity (2011)

（9） Oracle Corp.: “MySQL: The World's Most Popular Open Source Database”, http://www.mysql.com/.

(10) Oracle Corp.: “Hybrid Columnar Compression (HCC) on Exadata”, An Oracle White Paper (2012)

(11) Phil Francisco: “The Netezza Data Appliance Architecture: A Platform for High Performance Data Warehousing and Analytics” IBM Redbooks (2011)

(12) Margo I. Seltzer, Peter M. Chen, and John K. Ousterout: “Disk Scheduling Revisited”, In Proc. USENIX Tech. Conf., pp. 313-323 (1990)

(13) M. Stonebraker, Eugene Wong, Peter Kreps, and Gerald Held: “The Design and Implementation of INGRES”, ACM Trans. Database Syst., Vol. 1, No. 3, pp. 189-222 (1976)

(14) Ivan Sutherland: “The tyranny of the clock”, Comm. ACM, Vol. 55, No. 10, pp. 35-36 (2012)

(15) N. Takahashi and H. Yoshida: “Hitachi TagmaStore Universal Storage Platform: Virtualization without Limits”, White Paper, Hitachi Ltd. (2004)

(16) Transaction Processing Performance Council: “TPC-H, an ad-doc, decision support benchmark”, http://www.tpc.org/tpch/.

(17) A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, N. Zhang, S. Antony, H. Liu and R Murthy. “Hive –A Petabyte Scale Data Warehouse Using Hadoop”, Proc. ICDE, pp. 996-1005 (2010)

(18) Eugene Wong and Karel Youssefl: “Decomposition - a strategy for query processing”, ACM Trans. Database Syst., Vol. 1, No. 3, pp. 223-241 (1976)

(19) Bruce L. Worthington, Gregory R. Ganger, and Yale N. Patt: “Scheduling Algorithms for Modern Disk Drives”, In Proc. ACM SIGMETRICS Conf., pp. 241-251 (1994)

(20) 喜連川優，合田和生：「アウトオブオーダ型データベースエンジン OoODE の構想と初期実験」，日本データベース学会論文誌, Vol. 8, No. 1, pp. 131-136 (2009)

(21) 合田和生，豊田正史，喜連川優：「アウトオブオーダ型データベースエンジン OoODE の試作とその実行挙動」，電子情報通信学会第 5 回データ工学と情報マネジメントに関するフォーラム／第 11 回日本データベース学会年次大会(DEIM2013), F3-1 (2013).

(22) 清水晃，徳田晴介，田中美智子，茂木和彦，合田和生，喜連川優：「アウトオブオーダ型データベースエンジン OoODE におけるタスク管理機構の一実装方式の評価」，電子情報通信学会第 5 回データ工学と情報マネジメントに関するフォーラム／第 11 回日本データベース学会年次大会(DEIM2013), F3-5 (2013)

(23) 早水悠登，合田和生，喜連川優：「アウトオブオーダ型データベースエンジン OoODE によるクエリ処理性能の実験的評価」，電子情報通信学会／日本データベース学会データ工学と情報マネジメントに関するフォーラム(DEIM), F3-3 (2013)

(24) 山田浩之，合田和生，喜連川優：「Hadoop におけるアウトオブオーダ型並列処理系の実装に関する一考察」，電子情報通信学会第 5 回データ工学と情報マネジメントに関するフォーラム／第 11 回日本データベース学会年次大会(DEIM2013), F3-3 (2013)