The University of Osaka
Institutional Knowledge Archive

Title	広帯域ネットワークにおけるデータベース移動に基づくトランザクション処理手法に関する研究
Author(s)	原, 隆浩
Citation	大阪大学, 2000, 博士論文
Version Type	VoR
URL	https://doi.org/10.11501/3178673
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

https://ir.library.osaka-u.ac.jp/

The University of Osaka

広帯域ネットワークにおけるデータベース移動に基づくトランザクション処理手法に関する研究

２０００年５月

原　　隆浩

内容梗概

近年の光ファイバを用いたネットワーク技術やATM（非同期転送モード：Asynchronous Transfer Mode）などの交換技術の発展により，データ通信のために利用できるネットワークの帯域幅が急速に拡大している．このような広帯域ネットワークでは，データの転送にかかる遅延よりもデータがネットワーク内を伝わるための遅延の方が支配的となるため，分散処理においてデータの転送にかかる遅延が処理のボトルネックであった従来の状況が一変する．つまり，従来の分散システムでは，分散処理のための転送データ量の削減が性能向上のための重要な要因であったが，広帯域ネットワーク上の分散システムでは，むしろ帯域幅を有効利用して通信回数を削減することで処理時間を短縮できる．

データベース分野においても，従来はデータベースは特定のサイトに固定され，それらに対する処理はメッセージによる処理依頼で行うなど，伝送データ量を削減する手法を開発することが一般的であった．しかし，広帯域ネットワークでは，データベース自体をネットワークを介して短時間で移動させること（データベース移動）も現実的に可能である．このような状況から，ネットワークを介してデータベースを移動させること（データベース移動）をデータベース処理に利用することも現実的に可能になりつつある．帯域幅の拡大やネットワークの広域化が益々進む今後は，データベース移動がデータベース処理における最も有効で強力な技術の一つになるものと考えられる．

そこで本研究では，広帯域ネットワーク上でのデータベース移動を分散データベースシステムにおけるトランザクション処理に応用する方法論を確立した．具体的には，データベース移動を用いてトランザクション処理を実行する方式を考案し，シミュレーション評価によってその有効性を検証した．また，従来の分散データベースシステムでは，並行処理性能の向上のためにデータベースの複製を作成することが一般的であるため，考案したトランザクション手法を複製が存在する環境に拡張した動的複製配置法を考案した．更に，データベース移動を実環境において利用することを考えた場合に重要な問題となる，データベースの位置管理についても効果的な手法を考案した．データベース移動を，公衆網やバックボーンネットワーク，LAN (Local Area Network) などが混在する異種WAN (Wide Area Network) 環境において利用することを考慮して，異種WAN環境に適した移動のスケジューリング手法についても考案した．

本論文は，6章より構成され，その内容は以下の通りである．

まず，1において，本研究の背景と動機について述べる．

第2章において，データベース移動に基づく分散データベースシステム DB-MAN (distributed database system based on DataBase Migration in ATM Networks) を提案する．DB-MAN は，これまでの分散データベースシステムにはない新たな機構として，トランザクション処理手法の選択機構と，データベースの移動を考慮した並行処理制御機構を有する．前者は，従来の二相コミットプロトコルに基づくデータベースを固定型のトランザクション処理手法（以後，固定処理と呼ぶ）と，データベース移動を利用したトランザクション処理手法とで，より効率的な方を適応的に選択する機構である．後者は，トランザクションが頻繁に発生する環境において処理効率が低下するのを防ぐために，従来の二相施錠規約で用いられる読出し施錠，書込み施錠に加えて，データベースの移動に関する新たな施錠を導入した並行処理制御機構である．これらの機構により，DB-MAN におけるトランザクション処理効率は，従来の分散データベースシステムと比べて大幅に改善できる．

次に，第3章において，データベース移動を用いてデータベースの複製を動的に再配置する手法を提案する．第2章で提案する手法では，データベース移動後に移動元のデータベースを削除することで複製の存在しない状態を実現していたが，ここで提案する動的複製配置法では，移動元のデータベースを削除しない．各サイトにおいて，制限された主記憶領域の範囲内で，利用価値の低い複製を現トランザクションで利用するデータベースの複製と置き換えることで，動的な複製の再配置を実現する．

第4章では，データベース移動を実環境において利用する際に必要となる，データベースの位置管理について議論する．特に，位置管理の重要性が高い WAN を想定して，従来の分散システム内の資源の位置管理や移動体計算機環境での移動体の位置管理に用いられている手法をデータベース移動に適用した手法，および，ATM ネットワークの特性を考慮した新たな手法を提案する．

第5章では，異種 WAN 環境においてデータベース移動を用いることを想定して，異種 WAN 環境に適した移動のスケジューリング手法を提案する．異種 WAN 環境では，トランザクション単位でのデータベース移動の制御およびバックボーン部以外での移動が困難なことから，第2章や第3章で提案した手法をそのまま用いることはできない．ここで提案する手法では，各データベース毎に，与えられたアクセス系列に対して，データベース操作のための通信所要時間を最短にする移動のスケジューリングを行う．

第6章では，本論文の成果を要約し，最後に今後の研究課題について述べる．

研究業績

1. 学術論文誌掲載論文

(1) 原 隆浩, 春本 要, 塚本 昌彦, 西尾 章治郎: "ATM ネットワークにおけるデータベース移動のためのデータベース位置管理手法," 電子情報通信学会論文誌 D-I, vol. J80-D-I, no. 2, pp. 137–145 (Feb. 1997).

(2) 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: "データベース移動を用いた ATM ネットワークにおけるトランザクション処理," 電子情報通信学会論文誌 D-I, vol. J80-D-I, no. 6, pp. 505–513 (June 1997).

(3) 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: "データベース移動に基づく分散データベースシステムとその並行処理制御機構," 電子情報通信学会論文誌 D-I, vol. J81-D-I, no. 6, pp. 819–828 (June 1998).

(4) T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio: "Database Migration: A New Architecture for Transaction Processing in Broadband Networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 10, no. 5, pp. 839–854 (Sept./Oct. 1998).

(5) 矢島 悦子, 原 隆浩, 塚本 昌彦, 西尾 章治郎: "相関性をもつデータ間の放送時間間隔について," 情報処理学会論文誌, vol. 40, no. 1, pp. 188–196 (Jan. 1999).

(6) 秋山 豊和, 原 隆浩, 春本 要, 塚本 昌彦, 西尾 章治郎: "アクセス情報に基づくデータベース移動を用いたデータベース再配置手法," 情報処理学会論文誌, vol. 40, no. 6, pp. 2765–2775 (June. 1999).

(7) 原 隆浩, 春本 要, 塚本 昌彦, 西尾 章治郎, 奥井 順: "広帯域ネットワーク上のデータベース移動に基づく動的複製配置法," 電子情報通信学会論文誌 D-I, vol. J82-D-I, no. 8, pp. 1049–1058 (Aug. 1999).

(8) 矢島 悦子, 原 隆浩, 塚本 昌彦, 西尾 章治郎: "データ間の相関性を考慮した放送データのスケジューリング法およびキャッシング法," 情報処理学会論文誌, vol. 40, no. 9, pp.

3577–3585 (Sept. 1999).

(9) 原 隆浩, 塚本 昌彦, 西尾 章治郎: "WAN 環境におけるデータベース移動のスケジューリング手法," 電子情報通信学会和文論文誌 D-I, vol. J82-D-I, no. 12, pp. 1369–1378 (Dec. 1999).

(10) 萩野 浩明, 新谷 剛, 原 隆浩, 塚本 昌彦, 春本 要, 西尾 章治郎: "移動体計算環境における連続メディア配送のための通信方式", 情報処理学会論文誌, vol. 41, no. 2, pp. 363–372 (Feb. 2000).

(11) 西澤 正稔, 萩野 浩明, 原 隆浩, 塚本 昌彦, 西尾 章治郎: "アドホックネットワークにおける片方向リンクを考慮したルーティング方式," 情報処理学会論文誌, vol. 41, no. 3, pp. 783–791 (Mar. 2000).

(12) Akiyama, T., Hara, T., Harumoto, K., Tsukamoto, M., and Nishio, S.: "Access Skew Detection for Dynamic Database Relocation," *International Journal of Computing and Informatics (Informatica)*, to appear (2000).

2. 国際会議会議録掲載論文

(1) T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio: "Database Migration for Transaction Processing in ATM Networks," in *Proc. 11th International Conference on Information Networking (ICOIN-11)*, vol. 1, pp. 1B-4.1–1B-4.10, Taipei, Taiwan, R.O.C. (Jan. 1997).

(2) T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio: "Location Management Methods of Migratory Data Resources in ATM Networks," in *Proc. ACM Symposium on Applied Computing (ACM SAC'97)*, pp. 123–130, San Jose, California, U.S.A. (Feb. 1997).

(3) T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio, and J. Okui: "Main Memory Database for Supporting Database Migration," in *Proc. 1997 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM'97)*, vol. 1, pp. 231–234, Victoria, B.C., Canada (Aug. 1997).

(4) H. Hagino, T. Hara, M. Tsukamoto, and S. Nishio: "A Location Management Method using Network Hierarchies," in *Proc. 1997 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM'97)*, vol. 1, pp. 243–246, Victoria, B.C., Canada (Aug. 1997).

(5) T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio: "DB-MAN: A Distributed Database System based on Database Migration in ATM Networks," in *Proc. 14th International Conference on Data Engineering (ICDE'98)*, pp. 522–531, Orlando, Florida, U.S.A. (Feb. 1998).

(6) T. Hara, E. Yajima, M. Tsukamoto, S. Nishio, and J. Okui: "A Scheduling Strategy of a Broadcast Program for Correlative Data," in *Proc. ISCA International Conference on Computer Applications in Industry and Engineering (ISCA CAINE'98)*, pp. 141–145, Las Vegas, California, U.S.A. (Nov. 1998).

(7) T. Akiyama, T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio: "Database Migration Based on Access Skew Detection," in *Proc. International Symposium on Database, Web, and Cooperative Systems (DWACOS'99)*, pp. 65–72 (Aug. 1999).

(8) T. Hara, M. Tsukamoto, and S. Nishio: "A Scheduling Method of Database Migration for WAN Environments," in *Proc. Brazilian Symposium on Databases (SBBD'99)*, pp. 125–136 (Oct. 1999).

(9) M. Nishizawa, H. Hagino, T. Hara, M. Tsukamoto, and S. Nishio: "A Routing Method Usin Uni-directional Link in Ad Hoc Networks," in *Proc. 7th International Conference on Advanced Computing and Communications (ADCOM'99)*, pp. 78–82 (Dec. 1999).

(10) Y.H. Loh, T. Hara, M. Tsukamoto, and S. Nishio: "A Hybrid Method for Concurrent Updates on Disconnected Databases in Mobile Comuting Environments," in *Proc. ACM Symposium on Applied Computing (ACM SAC 2000)*, vol. 2, pp. 563–565 (Mar. 2000).

(11) T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio: "Dynamic Replica Allocation Using Database Migration in Broadband Networks," in *Proc. IEEE International Conference on Distributed Computing Systems (ICDCS 2000)*, pp. 376–384 (Apr. 2000).

(12) T. Hara, M. Tsukamoto, and S. Nishio: "Database Migration in WAN Environments: How Can It Earn Good Performance?," in *Proc. International Conference on Database and Expert Systems Applications (DEXA 2000)*, to appear (Sept. 2000).

3. 国内重要会議会議録掲載論文

(1) 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: "データベース移動を用いた分散データベースシステムの並行処理制御について," 情報処理学会マルチメディア通信と分散処理ワークショップ論文集, pp. 179–186 (Oct. 1996).

(2) 萩野 浩明, 原 隆浩, 塚本 昌彦, 西尾 章治郎: "ネットワークの階層化を用いた移動体通信プロトコルについて," 情報処理学会マルチメディア通信と分散処理ワークショップ論文集, pp. 187–192 (Oct. 1996).

(3) 萩野 浩明, 原 隆浩, 塚本 昌彦, 西尾 章治郎: "移動体通信のための階層的位置管理方式におけるグループ構成手法," 情報処理学会マルチメディア通信と分散処理ワークショップ論文集, pp. 179–184 (July 1997).

(4) 矢島 悦子, 原 隆浩, 塚本 昌彦, 西尾 章治郎: "データ間の相関性を考慮した放送時間間隔の決定法," 情報処理学会マルチメディア通信と分散処理ワークショップ論文集, pp. 173–180 (Nov. 1998).

(5) 秋山 豊和, 原 隆浩, 春本 要, 塚本 昌彦, 西尾 章治郎: "トランザクション系列に基づくデータベース移動を用いたデータベース再配置手法," 情報処理学会マルチメディア通信と分散処理ワークショップ論文集, pp. 153–160 (Nov. 1998).

(6) 原 隆浩, 塚本 昌彦, 西尾 章治郎: "WAN 環境に適したデータベース移動のスケジューリングアルゴリズムの提案," 電子情報通信学会データ工学ワークショップ論文集, CD-ROM (Mar. 1999).

(7) 萩野 浩明, 原 隆浩, 塚本 昌彦, 西尾 章治郎: "移動体計算環境における階層的位置管理方式のシミュレーション評価," 電子情報通信学会データ工学ワークショップ論文集, CD-ROM (Mar. 1999).

(8) 酒井 仁, 秋山 豊和, 海貝 明道, 原 隆浩, 春本 要, 塚本 昌彦, 西尾章治郎: “移動機能を有する分散データベースシステム DB-MANαの設計と実装,” 電子情報通信学会データ工学ワークショップ論文集, CD-ROM (Mar. 1999).

(9) 西澤 正稔, 萩野 浩明, 原 隆浩, 塚本 昌彦, 西尾 章治郎: “アドホックネットワークにおけるリンク状態を考慮した片方向リンク対応ルーティング方式,” 情報処理学会マルチメディア, 分散, 協調とモーバイルシンポジウム論文集, pp. 273–278 (June 1999).

(10) 萩野 浩明, 原 隆浩, 塚本 昌彦, 西尾 章治郎: “アドホックネットワークのための蓄積型フラッディングプロトコルの提案,” 情報処理学会マルチメディア通信と分散処理ワークショップ論文集, pp. 61–66 (Dec. 1999).

(11) 酒井 仁, 秋山 豊和, 海貝 明道, 原 隆浩, 塚本 昌彦, 西尾 章治郎: “データベース移動に基づく分散データベースシステム DB-MANαにおける同時実行制御機構の設計と実装,” 日本ソフトウェア科学会プログラミングおよび応用のシステムに関するワークショップ論文集, URL: http://www.jaist.ac.jp/SPA2000/20000320/ (Mar. 2000).

(12) 原 隆浩, 西尾 章治郎: “アドホックネットワークにおけるアクセス可能性向上のためのデータ複製配置方式,” 情報処理学会マルチメディア, 分散, 協調とモーバイルシンポジウム論文集, 発表予定 (June 2000).

(13) 秋山 豊和, 海貝 明道, 酒井 仁, 原 隆浩, 塚本 昌彦, 西尾 章治郎: “DB-ManMos: データベース移動機能をもつ分散データベースシステムのためのモニリング機構,” 情報処理学会マルチメディア, 分散, 協調とモーバイルシンポジウム論文集, 発表予定 (June 2000).

(14) 萩野 浩明, 原 隆浩, 塚本 昌彦, 西尾 章治郎: “グループ移動を考慮した移動体通信方式,” 情報処理学会マルチメディア, 分散, 協調とモーバイルシンポジウム論文集, 発表予定 (June 2000).

4. 国内研究会, 全国大会発表論文

(1) 齋藤 淳, 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: “ASN.1 データベースシステムにおけるデータ格納方式,” 情報処理学会研究報告, vol. 95, no. 12, pp. 17–24 (Jan. 1995).

(2) 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: “ASN.1 データベースシステムにおけるインデックス機構の比較,” 情報処理学会研究報告, vol. 95, no. 147, pp. 17–24 (July 1995).

(3) 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: “ATM 環境におけるデータベース移動に基づくトランザクション処理手法,” 情報処理学会研究報告, vol. 96, no. 11, pp. 49–56 (Jan. 1996).

(4) 庄司加奈子, 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: “広帯域ネットワークにおけるデータベース移動の実現について,” 電子情報通信学会技術研究報告, vol. 96, no. 54, pp. 55–60 (May 1996).

(5) 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: “ATM 環境内で移動するデータベースの位置管理について,” 電子情報通信学会技術研究報告, vol. 96, no. 176, pp. 31–36 (July 1996).

(6) 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: “データベース移動に基づく分散データベースシステム DB-MAN の性能評価,” 情報処理学会第 54 回全国大会講演論文集 (3), pp. 3-243–3-244 (Mar. 1997).

(7) 萩野 浩明, 原 隆浩, 塚本 昌彦, 西尾 章治郎: “2 段階の階層化を用いた移動体通信方式の評価,” 情報処理学会第 54 回全国大会講演論文集 (3), pp. 3-469–3-470 (Mar. 1997).

(8) 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: “データベース移動に基づく動的複製配置法,” 電子情報通信学会技術研究報告, vol. 97, no. 160, pp. 107–112 (July 1997).

(9) 秋山豊和, 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: “データベース移動を用いた分散データベースシステムの設計と実装,” 情報処理学会第 55 回全国大会講演論文集 (3), pp. 3-469–3-470 (Sept. 1997).

(10) 矢島 悦子, 原 隆浩, 塚本 昌彦, 西尾 章治郎: “ライフポイントを用いた周期的放送データのキャッシング法,” 情報処理学会研究報告, vol. 97, no. 104, pp. 237–242 (Nov. 1997).

(11) 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: “広帯域ネットワークにおけるデータベースシステムアーキテクチャ,” インターネット技術研究委員会 (ITRC) シンポジウム, pp. 97–104 (Feb. 1998).

(12) 萩野 浩明, 原 隆浩, 塚本 昌彦, 西尾 章治郎: “移動体計算環境における動的グループ化を用いた位置管理方式,” 情報処理学会研究報告, vol. 98, no. 58, pp. 31–38 (July 1998).

(13) 矢島 悦子, 原 隆浩, 塚本 昌彦, 西尾 章治郎: “データ間の相関性を考慮した放送データの送受信方式について,” 情報処理学会研究報告, vol. 98, no. 57, pp. 95–102 (July 1998).

(14) 原 隆浩, 矢島 悦子, 塚本 昌彦, 西尾 章治郎: “アクセス要求発生の時間間隔を考慮した放送データのキャッシング方式,” 情報処理学会研究報告, vol. 99, no. 61, pp. 1–6 (July 1999).

(15) 秋山 豊和, 原 隆浩, 塚本 昌彦, 西尾 章治郎: “データベース移動を用いた分散データベースシステムにおける複製を考慮したデータベースの位置管理手法,” 情報処理学会研究報告, vol. 99, no. 61, pp. 327–332 (July 1999).

(16) Y.H. Loh, T. Hara, M. Tsukamoto, and S. Nishio: “Controlling Database Updates in Mobile Computing Environments with Frequent Disconnection,” 情報処理学会研究報告, vol. 99, no. 61, pp. 375–380 (July 1999).

(17) 北濱 秀基, 西澤 正稔, 萩野 浩明, 原 隆浩, 塚本 昌彦, 西尾 章治郎: “移動体通信のための転送制御方式の適応的選択について,” 情報処理学会第 59 回全国大会講演論文集 (3), pp. 253–254 (Sept. 1999).

(18) 海貝 明道, 酒井 仁, 秋山 豊和, 原 隆浩, 塚本 昌彦, 西尾章治郎: “データベース移動に基づく分散データベースシステム DB-MANαの性能評価,” 情報処理学会研究報告, vol. 99, no. 94, pp. 79–84 (Nov. 1999).

(19) 上向 俊晃, 萩野 浩明, 原 隆浩, 塚本 昌彦, 西尾 章治郎: リモートディスプレイ環境における携帯電話を用いた WWW ブラウジング方式, 情報処理学会ヒューマンインタフェース研究会・モーバイルコンピューティング研究会報告, vol. 99, no. 97, pp. 51–56 (Nov. 1999).

目 次

1 序章 **1**

1.1 研究の背景 1

1.2 アプローチ 3

1.3 本論文の構成 5

2 データベース移動に基づく分散データベースシステム **7**

2.1 まえがき 7

2.2 関連研究との比較 7

2.2.1 負荷分散のためのデータの再配置 8

2.2.2 Datacycle 手法 8

2.2.3 send-on-demand 手法 8

2.3 データベース移動を用いたトランザクション処理 9

2.3.1 分散環境のモデル 9

2.3.2 データベース移動を用いたトランザクション処理手法 10

2.4 DB-MAN システム 15

2.4.1 システムの概要 15

2.4.2 トランザクション処理手法の選択機構 17

2.4.3 DB-MAN における並行処理制御機構 21

2.4.4 リカバリ機能 25

2.5 シミュレーションによる性能評価 30

2.5.1 手法選択機構の性能評価 30

2.5.2 並行処理制御機構の性能評価 36

2.6 むすび 39

3 データベース移動に基づく動的複製配置法 41

3.1 まえがき 41

3.2 従来の複製手法 42

3.3 データベース移動を用いた動的複製配置法 43

3.3.1 システムの基本動作 43

3.3.2 DB 移動複製法 45

3.4 シミュレーションによる性能評価 50

3.4.1 シミュレーション環境 51

3.4.2 シミュレーション結果 53

3.5 むすび 57

4 データベースの位置管理 59

4.1 まえがき 59

4.2 移動体の位置管理手法の応用 60

4.3 移動追従型の位置管理手法の応用 60

4.4 ATM ネットワークの特性を考慮した手法 62

4.5 通信所要時間の比較 64

4.5.1 メッセージ通信に関するパラメータの定義 64

4.5.2 シミュレーション環境 66

4.5.3 シミュレーション方法と結果 68

4.5.4 シミュレーション結果の考察 70

4.6 むすび 73

5 異種 WAN 環境におけるデータベース移動 75

5.1 まえがき 75

5.2 最適な移動のスケジューリング 77

5.2.1 システム環境 77

5.2.2 移動のスケジューリング 80

5.2.3 実行例 82

5.3 アルゴリズムの正当性 83

5.4 実環境への適用 88

5.5 むすび 89

6 結論 **91**

6.1 本論文のまとめ 91

6.2 今後の課題 92

6.2.1 計算機の異種性 93

6.2.2 動的スキーマ生成 94

6.2.3 手法選択に適したトランザクション形式 94

6.2.4 データベースの最適単位 94

6.2.5 データベース移動に適した主記憶データベースの物理構造 95

謝辞 **97**

第1章

序章

1.1 研究の背景

近年，ATM（Asynchronous Transfer Mode: 非同期転送モード）などの交換技術や光ファイバなどの高速通信技術の発展により，数百 Mbps や数 Gbps の帯域幅をもつ広帯域ネットワークが普及しつつある．帯域幅の拡大により，大量のデータを短時間で転送することが可能になったため，さまざまなアプリケーションが盛んに構築され始めている [19],[23],[61],[65],[82]．現在盛んに研究が行われているサイエンティフィックデータベースや CALS などでは，マルチメディアデータを多く扱うことから，豊富な帯域幅や ATM 方式などにおける QoS (Quality of Service) の保証といった性質が有効になる．また，移動計算環境においてもバックボーンとして広帯域ネットワークを用いることにより柔軟性に富んだ高度な通信環境を提供できる [12],[16],[18],[60].

一方，帯域幅の拡大は，分散処理において通信コストがボトルネックであったこれまでの状況が一変することを意味している．つまり，計算機ネットワーク上の分散処理に関する研究は，伝送データ量の削減による性能向上から，帯域幅の有効利用による性能向上を目的とした技術の開発へと移行する必要がある．データベース分野においても，従来はデータベースは特定のサイトに固定され，それらに対する処理はメッセージによる処理依頼で行うなど，伝送データ量を削減する手法を開発することが一般的であった．しかし，広帯域ネットワークでは，データベース自体をネットワークを介して短時間で移動させること（データベース移動）も現実的に可能である [59].

ここで，データベース移動の対象となるのは，大型計算機センターなどで集中管理して

いるような位置依存性が高く，しかもギガバイト，テラバイトオーダにも至るような超大規模なデータベースではなく，ネットワークで相互接続された分散システムの各サイトで保持されている位置依存性が高くない百メガバイトオーダまでのデータベースである．その例としては，全世界規模のイントラネットを構築している企業で社内情報を共有するような場合が考えられる．近年のネットワーク技術，計算機システム技術の発展によって，ネットワークで相互接続されたデスクトップ型の計算機等で構成される分散システムを一つの大きなシステムと考えることにより安価で高性能なシステムが構築できることから，このような分散データベースシステムの重要性は高く，これから益々盛んに構築されていくものと考えられる．

このようなネットワーク環境においてデータベース移動を用いることにより，これまでの分散データベース技術とは異なる新たな可能性が出てくる．従来の二相コミットプロトコル（2PC）によってトランザクション処理を行った場合，各サイトに分散するデータベースに対して処理依頼メッセージによる処理を複数回行った後，二相コミットのためのメッセージ交換を二往復させる必要がある．一方，データベース移動によって，アクセス対象となるデータベースをトランザクション発生サイトに移動してしまえば，その後のメッセージ交換の必要がなくなる．広帯域ネットワークでは，大量のデータの転送は短時間で行えるが，伝搬遅延は従来のネットワークと殆んど差がないことから，データベース移動による処理時間の短縮が期待できる．

例えば，東京からニューヨークにある 100Mbytes のデータベースにアクセスする場合を考える．この二点間は，光ファイバのケーブルで接続され，ATM 方式を用いて 1Gbps の帯域幅でデータ交換が行われると仮定する．二点間の距離が約 1.2×10^7m でファイバ中を光が伝達する速度が 2.1×10^8m/s であるから，データの伝搬遅延はコネクションの設定時間，両サイトでの処理遅延および交換機でのルーチング遅延を無視しても 5.7×10^{-2}秒となる．一方，データベースの移動に要する時間は，8.0×10^{-1}秒である．従って，処理依頼メッセージや制御メッセージの通信回数が 15 回以上になる場合は，データベースをアメリカに固定したままで処理をするよりも，データベースを移動してローカルに処理を行った方が効率が良くなる．処理の遅延等を考慮すると，この閾値（15 回）は更に小さくなる．

一方，データベース移動を現実的なものにするためには，データベースアクセスの高速化やデータベースに付加していたインデックスの移動などの課題がある．低価格化，高性能化が急速に進むメモリ技術によって，近い将来には数 Gbytes の主記憶を持つマシンが一

般的になると考えられることから，前者は，主記憶データベース技術 [21],[22],[24],[55] を用いることで解消できる．後者に関しては，文献 [74] などにおいてデータ項目の移動を利用したシステムで採用されている二次インデックスの移動技術が利用できる．

このように，データベース移動のための技術課題が着実に解消されていき，更にネットワークの帯域幅が急速に拡大しているのとは対称的に，通信の伝搬遅延に関しては光の伝搬速度に上限があるために大幅な改善は不可能である．従って，ネットワークの広域化，データベース処理の多様化が進んでいくにつれて，データベース移動が強力で有効なデータベース技術の一つになり，さまざまな用途に有効に活用されるものと考えられる．

1.2 アプローチ

本研究では，広帯域ネットワーク上でのデータベース移動を分散データベースシステムにおけるトランザクション処理に応用することを考える．具体的には，データベース移動を用いてトランザクション処理を実行する方式を確立する．

まず，システム内にデータベースの複製が存在しないような単純な環境を想定し，トランザクションの複雑さやアクセスパターンに応じてデータベース移動を実行してトランザクションを処理する手法を考案する．更に，従来の分散データベースシステムでは，並行処理性能の向上のためにデータベースの複製を作成することが一般的であることを考慮して，考案したトランザクション手法を複製が存在する環境に拡張した動的複製配置法を考案する．

次に，これらの手法を実環境において運用するための技術課題のうち，最も重要と考えられるデータベースの位置管理について，広帯域ネットワークの特性を考慮した効果的な手法を考案する．

また，データベース移動を，公衆網やバックボーンネットワーク，LAN (Local Area Network) などが混在する異種 WAN (Wide Area Network) 環境において利用することを考慮して，異種 WAN 環境に適した移動のスケジューリング手法についても考案する．

本論文では，以上の事項を次のようなアプローチで実現する．

(1) データベース移動に基づく分散データベースシステムの提案

データベース移動に基づく分散データベースシステム DB-MAN (distributed database system based on DataBase Migration in ATM Networks) を提案する．DB-MAN は，

これまでの分散データベースシステムにはない新たな二つの機構として，トランザクション処理手法の選択機構と，データベースの移動を考慮した並行処理制御機構を有している．前者は，従来の二相コミットプロトコルに基づくデータベースを固定型のトランザクション処理手法（以後，固定処理と呼ぶ）と，データベース移動を利用したトランザクション処理手法とで，より効率的な方を適応的に選択する機構である．後者は，トランザクションが頻繁に発生する環境において処理効率が低下するのを防ぐために，従来の二相施錠規約で用いられる読出し施錠，書込み施錠に加えて，データベースの移動に関する新たな施錠を導入した並行処理制御機構である．これらの機構により，DB-MAN におけるトランザクション処理効率は，従来の分散データベースシステムと比べて大幅に改善できる．

(2) データベース移動に基づく動的複製配置法の提案

データベース移動を用いてデータベースの複製を動的に再配置する手法を提案する．(1) で提案する手法では，データベース移動後に移動元のデータベースを削除することで複製の存在しない状態を実現していたが，ここで提案する動的複製配置法では，移動元のデータベースを削除しない．各サイトにおいて，制限された主記憶領域の範囲内で，利用価値の低い複製を現トランザクションで利用するデータベースの複製と置き換えることで，動的な複製の再配置を実現する．

(3) データベースの位置管理手法の提案

データベース移動を実環境において利用する際に重要な問題となるデータベースの位置管理について議論する．特に，位置管理の重要性が高い WAN を想定して，従来の分散システム内の資源の位置管理や移動体計算機環境での移動体の位置管理に用いられている手法をデータベース移動に適用した手法，および，ATM ネットワークの特性を考慮した新たな手法を提案する．

(4) 異種 WAN 環境に適したデータベース移動のスケジューリング手法の提案

異種 WAN 環境においてデータベース移動を用いることを想定して，異種 WAN 環境に適した移動のスケジューリング手法を提案する．異種 WAN 環境では，トランザクション単位でのデータベース移動の制御およびバックボーン部以外での移動が困難なことから，(1) や (2) で提案する手法をそのまま用いることはできない．ここで提案す

る手法では，各データベース毎に，与えられたアクセス系列に対して，データベース操作のための通信所要時間を最短にする移動のスケジューリングを行う．

1.3 本論文の構成

本論文の構成は以下の通りである．まず，第2章において，データベース移動に基づく分散データベースシステム DB-MAN を提案する．特に，DB-MAN システムの特徴である，トランザクション処理手法の選択機構と，データベースの移動を考慮した並行処理制御機構について詳しく説明する．更に，提案したシステムの性能評価のために行ったシミュレーションの結果を示す．次に，第3章において，データベース移動を用いてデータベースの複製を動的に再配置する手法を提案する．更に，シミュレーション評価により，第2章の手法および提案する動的複製配置法が，それぞれどのような環境で有効かを示す．第4章では，データベースの位置管理のための7つの手法を提案する．シミュレーション評価により，与えられたシステム環境に対して，どの手法が最適となるかを示す．第5章において，異種 WAN 環境に適した，データベース移動のスケジューリング手法を提案する．更に，提案する手法が，与えられたアクセス系列に対して，データベース操作のための通信所要時間を最短にする移動のスケジュールを決定できることを証明する．最後に，第6章では，本論文の成果を要約し，最後に今後の研究課題について述べる．

なお，第2章は文献 [26], [28], [29], [32], [33], [35], [36], [37], [38], [39] で公表した結果に基づき論述する．第3章は文献 [34], [41], [44] で公表した結果に基づき論述する．第4章は文献 [27], [30], [31] で公表した結果に基づき論述する． 第5章は文献 [40], [42], [43], [45] で公表した結果に基づき論述する．

第2章

データベース移動に基づく分散データベースシステム

2.1 まえがき

本章では，データベース移動を用いてトランザクション処理を実行する分散データベースシステム DB-MAN を提案する．提案するシステムでは，トランザクション開始時に，トランザクションの性質に応じて従来の固定処理若しくはデータベース移動を用いた処理を選択して，トランザクションを実行する．

まず，分散システム内のデータの移動を考慮して提案された従来の手法と，本章で提案する手法を比較し，その差異を明確にする．更に，想定する環境とデータベース移動を用いたトランザクションの処理方法について述べた後，提案する DB-MAN システムについて説明する．更に，DB-MAN システムの性能評価のために行ったシミュレーションの結果を示す．

2.2 関連研究との比較

分散処理においてデータベースやデータ項目の移動を考えた研究は，これまでにいくつか報告されている．本章では，これらの研究と本研究とを比較することで，本研究の有効性について検証する．

2.2.1 負荷分散のためのデータの再配置

文献 [20],[51],[56],[57],[69],[81] などでは，相互接続されているサーバ間の負荷を均一化することを目的として，分散データの再配置を行っている．しかし，これらの研究では広帯域ネットワークは想定されておらず，データの移動は大きなオーバヘッドとなると考えられている．従って，データの移動を頻繁には行わず，本研究のようにトランザクション単位でデータベースを移動するものとは仮定が異なる．

2.2.2 Datacycle 手法

本研究同様に，広帯域ネットワークの特性をトランザクション処理に利用する研究としては，文献 [15], [48] などがある．文献 [15], [48] では，データベースを光ファイバのリング型ネットワーク上で定期的にブロードキャストする Datacycle と呼ぶ手法を提案し，読出し操作のみのトランザクションが中心的な環境で高いスループットを実現している．

Datacycle は，データベースの内容がネットワーク上を流れるという点では本研究と似ている．しかし，この手法では，書込み操作は特定の一つのサイトで行っているので，本質的には分散データベースではない．また，リング型ネットワーク上のみに特化した手法であるため，その他のトポロジのネットワーク上では実現できない．

2.2.3 send-on-demand 手法

広帯域ネットワークの特性をトランザクション処理に利用したその他の研究として文献 [13] がある．文献 [13] では，トランザクションに必要なデータ項目をトランザクション発生サイトに転送する send-on-demand という手法を提案している．この手法は，Datacycle が書込み操作が中心的なトランザクションにおいて処理効率が低下することに着目し，このようなトランザクションにおいて高い処理効率を実現するために提案されたものである．更に文献 [13] では，読出し操作のみのトランザクションでは datacycle を用い，書込み操作を伴うトランザクションでは send-on-demand を用いるといった混合型の手法を提案し，Datacycle と send-on-demand の両者の長所を生かしてトランザクション処理効率を向上することを図っている．しかし，これらの手法は，datacycle を基盤としているため，datacycle と同様にリング型のネットワーク上でしか利用できない．また，すべての書込み操作をトランザクション発生サイトへデータ項目を移動することで処理しているため，転送すべき

データ項目の総サイズが大きくなると処理効率が劣化する．従って，多くのデータベースが関わるような複雑な処理を必要とするトランザクションが頻繁に発生するような環境では，効率的に動作しない．

一方，本章で提案する DB-MAN システムは，ATM 方式などに基づく一般的な広帯域ネットワークを想定したシステムであるため，あらゆるトポロジのネットワークに適用可能である．更に，DB-MAN の処理手法選択機構（特に履歴統計モード）では，固定処理とデータベース移動の予測処理時間やトランザクションのアクセスパターンを考慮して固定処理若しくはデータベース移動を選択するため，変化のある環境下でより高い処理効率が得られるものと考える．

2.3 データベース移動を用いたトランザクション処理

本節では，まず想定する分散環境のモデルについて説明する．その後，その環境においてデータベース移動を用いてトランザクションを処理する方法を説明し，その効果について議論する．

2.3.1 分散環境のモデル

本章では，ネットワーク環境として，ATM ネットワーク上の仮想 LAN を想定する．ATM は，基本的にはコネクション型の交換方式であり，データ交換を行う前に通信したいサイトとのコネクション（ATM では仮想チャネルと呼ぶ）を確立する必要がある．コネクションの種類としては，ATM 交換機において予め経路を設定しておく PVC（Permanent Virtual Channel）と，サイトからの要求によって経路を設定する SVC（Switched Virtual Channel）があり，いずれのコネクションに対しても，一対一通信のためのポイント・ツー・ポイントコネクションと一対多通信のためのポイント・ツー・マルチポイントコネクションがある．SVC コネクションは，輻輳が起こった場合や通常 20 分程度の長い時間その通信路を使用しない場合にしか切断されない．また，PVC を利用する場合は，コネクションの設定時間を省略することができる．

ATM ネットワークでは，あるサイトから複数のサイトへ予めポイント・ツー・マルチポイントコネクションの PVC を設定しておくことにより，そのサイトをマルチキャストサーバ（MCS）として特定のグループへのマルチキャストが可能となる．つまり，物理的

な接続形態とは異なる仮想的なLANの構築が可能となり，仮想LAN内のデータのブロードキャストは，MCSに対してデータを送信することで実現できる（図2.1）．

本章では，この仮想LAN内における分散データベースシステムについて議論する．仮想LAN内のすべてのサイトが分散データベース処理に参加し，各々のサイトがローカルのデータベース管理システムをもつ．データベース全体は，複数のローカル・データベース（以後，簡単のためにこれをデータベースと呼ぶ）に分割され，初期状態として特定のサイトに分散して配置される．データベースの分割方法やデータモデルに関しては，ここでは特に指定しない．

このモデルに基づいて，本章で議論する分散環境では以下のような想定を設ける．

- 仮想LAN内でブロードキャストする場合を除いて，すべての通信は必要時に確立されるSVCのポイント・ツー・ポイントコネクションを用いて行う．
- トランザクションの処理のために確立されたSVCコネクションは，そのトランザクション内で再設定の必要がない．
- 仮想LAN内の各データベースには一意な識別子であるDB-idをもたせる．仮想LAN内のすべてのデータベースの集合を$\boldsymbol{D} = \{D_1, D_2, \cdots, D_m\}$と表す．ここで，$m$はデータベースの総数を，$D_j$ $(1 \leq j \leq m)$はDB-idを示している．
- 任意の二サイト間の通信の平均伝搬遅延をd_m，任意のサイトとMCS間の平均伝搬遅延をd_{MCS}とする．
- 仮想LAN内のデータベースは全て主記憶データベースとし，ディスクアクセスのオーバヘッドはないものと考える．また，文献[74]において提案されている二次インデックスの移動技術などを利用することにより，データに付加されているインデックスの高速な移動も可能であると想定する．

2.3.2 データベース移動を用いたトランザクション処理手法

本節では，前節の環境モデルに基づいて，データベース移動を用いてトランザクションを処理する方法を説明する．また，本章で提案する分散データベースシステムの処理手法選択部で利用するために，固定処理とデータベース移動を用いた処理の処理時間を定式化

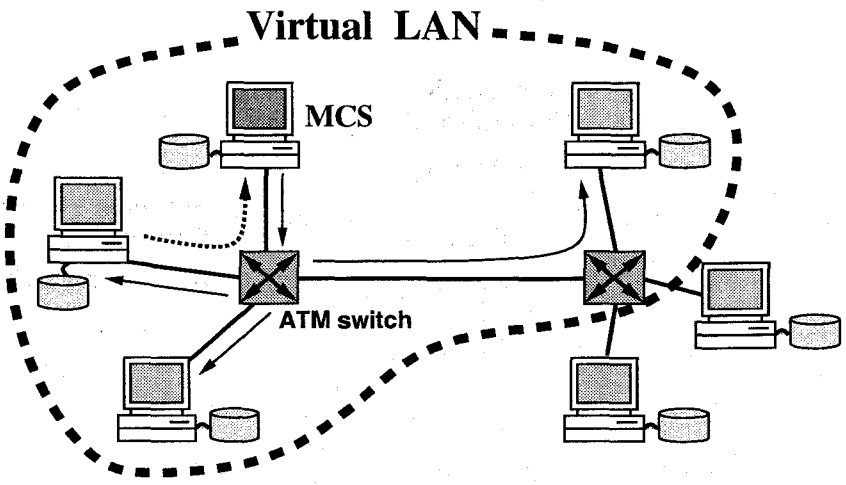


図 2.1: ATM の仮想 LAN

する．但し，この計算を動的に行う場合に，システムに対して大きな負担とならないように，簡単な式で近似的に表す．

まず，従来の固定処理を行う場合，図 2.2に示すように，トランザクションの処理に必要なデータベースを所持するサイトに対して，処理依頼メッセージによって処理を依頼し，その結果を受け取るといった操作を繰り返し行う．最終的な結果をトランザクション発生サイトがとりまとめ，トランザクションを発生したクライアント・アプリケーションへ返す．

従って，固定処理を行う場合の通信手順は図 2.3のようになる．最初に，処理操作に必要なメッセージ通信のために，トランザクション発生サイトからトランザクションを処理するためにに必要となるデータベースを所持する各サイト（サイト数 k）へのコネクションを設定しなければならない．これに要する時間を $C(k)$ と表す．コネクション設定後は，処理依頼メッセージの転送や結果の送信などに必要な通信が n 回行われ，これに nd_m の時間を要する．最後に，二相コミットのためのメッセージ交換が二往復行われるため，これに $4d_m$ の時間を要する．但し，広帯域ネットワークの特性を考慮すると，処理依頼メッセージや処理結果（中間結果を含む）のサイズは通信帯域幅に比べて小さいことから，各メッセージの転送遅延は 0 に近似する．また，2PC のための遅延は，トランザクション発生サイトとトランザクションに関係した複数（若しくは一つ）のサイト間の伝搬遅延のうち最大のものになるため，正確には d_m より大きくなるが，ここでは簡単のために d_m とする．

以上のことから，固定処理における通信所要時間 T_{fix} は次のように表される．

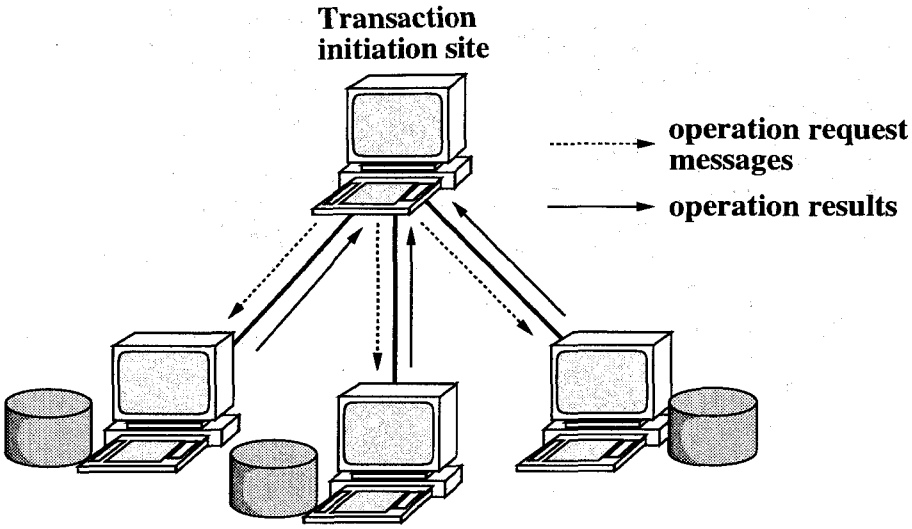


図 2.2: 従来のデータベース固定型処理

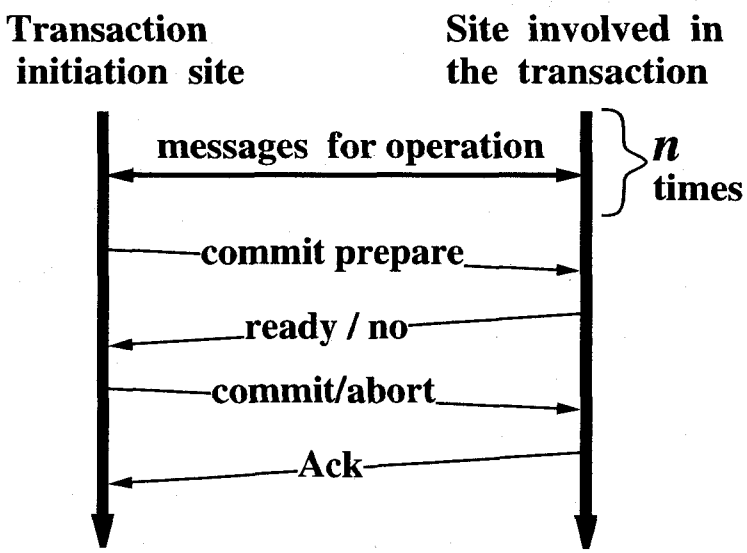


図 2.3: 固定処理の通信手順

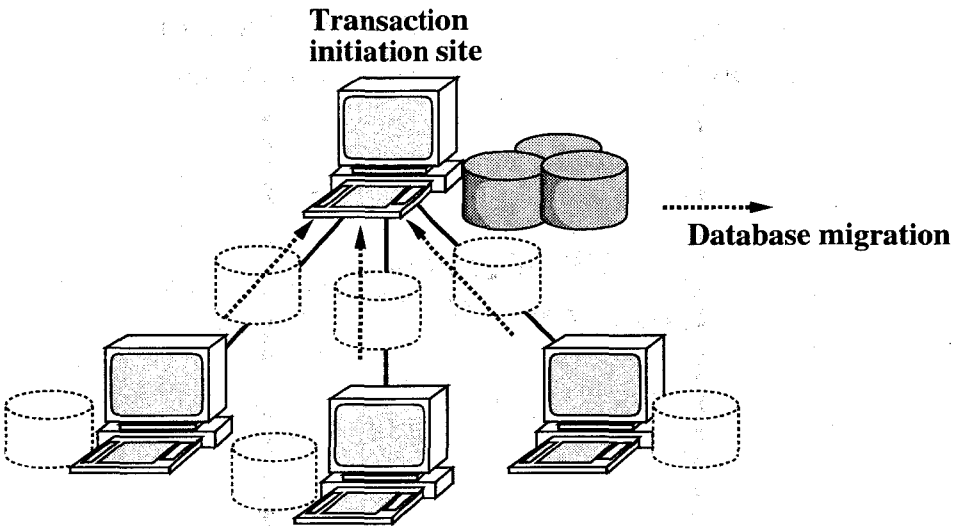


図 2.4: データベース移動を用いた処理

$$T_{fix} = (n+4) \cdot d_m + C(k) \tag{2.1}$$

一方，図2.4のように，トランザクション発生時にデータベース移動によって処理に必要なデータベースをトランザクション発生サイトに集めることで，トランザクション発生サイトでの集中処理が可能となり，メッセージ交換回数の削減による処理の高速化が期待できる．このようなデータベース移動を用いた処理を，図2.5に示すように，DB 要求，DB 移動，移動完了通知の 3 回の通信で実現する．ここで，本章では，トランザクションの開始時に，トランザクションに必要なデータベースが同定できるものと仮定する．

まず，DB 要求のためのメッセージを仮想 LAN 内の全サイトにブロードキャストする．これには，最初に MCS までメッセージが送られ，その後，全サイトへと転送されるため，$d_{MCS} + d_m$の時間を要する．但し，MCS から各サイトへの実質的な伝搬遅延は，必要なデータベースを所持しないサイトもあり，見積もることが困難であるため，簡単のために定数 d_mとする．また，転送遅延は固定処理の場合と同様に 0 に近似する．ここで，DB 要求のメッセージをブロードキャストするのは，全サイトが各データベースの正確な位置情報を保持することを可能にするためである．

次に，データベース移動を行うために，トランザクション発生サイトとデータベースを所持しているサイト（サイト数 k）との間に SVC コネクションを設定しなければならない．

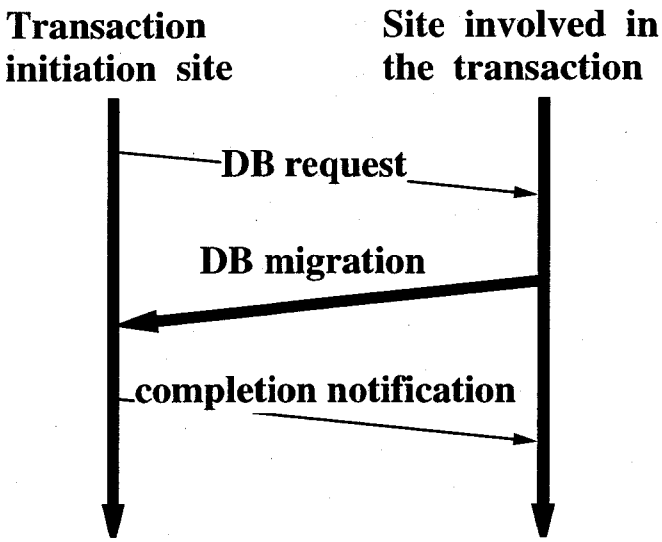


図 2.5: データベース移動を用いた処理の通信手順

これには固定処理と同じく $C(k)$ の時間を要する．コネクション設定後にデータベース移動を実行する．データベース D_jのサイズを $Size(D_j)$，トランザクションに必要でかつトランザクション発生サイトが所持していないデータベースの集合を$\boldsymbol{D}_N$，データベース移動のための予約帯域幅を B_Mとすると，データベース移動では転送遅延を無視できないため，これに要する時間が $d_m + \sum_{D_j \in \boldsymbol{D}_N} Size(D_j)/B_M$となる．ここでも実際の伝搬遅延は，2PC の場合と同様の理由で d_mより大きくなるが，簡単のため d_mとする．

最後に，移動完了通知をトランザクション発生サイトから仮想 LAN 内の全サイトにブロードキャストする．これには $d_{MCS} + d_m$の時間がかかる．このメッセージをブロードキャストすることで，全サイトに移動が完了したことを通知し，データベースを転送したサイトはそのコピーを消去することができる．その後は，メッセージ通信の必要がなく，トランザクション発生サイトでの集中処理が行われる．

以上のことから，データベース移動を用いた処理の通信所要時間 T_{DB}は次のようになる．

$$T_{DB} = 3d_m + 2d_{MCS} + C(k) + \sum_{D_j \in \boldsymbol{D}_N} Size(D_j)/B_M \tag{2.2}$$

近年の急速な計算機システムの高機能化を考慮すると，処理に要する通信所要時間がシステムの性能に大きく影響することは明らかである．そこで，両手法の通信所要時間を定式化した結果に注目すると，固定処理では処理に必要な通信回数 (n) が，データベース移

動を用いた処理では移動するデータベースの総サイズが，処理時間に大きく影響することが分かる．つまり，データベース移動をトランザクション処理に用いることにより，メッセージ通信が多数回必要となる複雑なトランザクションの処理時間が短縮されることが期待されるが，逆に，移動すべきデータベースのサイズが大きくなると処理効率が低下する．従って，効率的なトランザクションの処理を考えた場合，従来の固定処理とデータベース移動を用いた処理を，トランザクションの複雑度やデータベースの配置やサイズに応じて適応的に選択することが有効であるものと考えられる．

2.4 DB-MAN システム

本節では，従来の固定処理とデータベース移動を用いた処理を適応的に選択して，トランザクションの処理を行う分散データベースシステム DB-MAN を提案する．まずシステムの概要について説明し，その後，DB-MAN の手法選択機構を説明する．更に，トランザクションが頻繁に発生する場合にデータベース移動が処理のボトルネックとならないようにするための，並行処理制御機構について説明する．

2.4.1 システムの概要

DB-MAN のシステムの概要を図 2.6に示す．中間層には，固定処理とデータベース移動を用いた処理を適応的に選択するための手法選択部が存在する．クライアントアプリケーションが発生するトランザクションを中間層で解析し，適当な形式の問合せに変換して，実際にデータベースに対して処理を行う実行部に渡す．以下では，各部の機能について説明する．

トランザクション解析部： クライアント・アプリケーションからの問合せを実行部が処理可能な形式へと変換する．具体的には，問合せの解析，最適化，データベースの分散に基づいた部分問合せへの分割を行い，その結果を手法選択部に渡す．また，その問合せ全体（トランザクション）の処理に必要なデータベースやメッセージ通信の回数を調べ，部分問合せ，分割前の問合せと共にその結果も手法選択部に渡す．

手法選択部： DB 情報表として保持している過去のトランザクションのアクセスパタンやトランザクション解析部から受け取った現トランザクションに必要なデータベースや

通信回数の情報から，現トランザクションを固定処理若しくはデータベース移動を用いた処理のどちらで処理をするかを決定する．

固定処理で処理する場合，部分問合せの集合をコーディネータに渡す．データベース移動を用いて処理をする場合は，必要なデータベースのリストを DB 移動部に，分割前の問合せをコーディネータに渡す．

DB 移動部： トランザクション解析部から受け取った，移動すべきデータベースのリストに基づいて，DB 要求メッセージをブロードキャストする．すべての必要なデータベースの移動が完了した後，移動完了通知をブロードキャストする．

コーディネータ： 手法選択部から受け取った問合せに基づいて，必要な施錠の要求を行う．トランザクションを固定処理で処理する場合，手法選択部から受け取った部分問合せを適当なデータベースを保持するサイト（の実行部）へと転送し，その結果を受け取る．最後に，最終的な結果をとりまとめ，クライアント・アプリケーションに返す．データベース移動を用いて処理する場合，手法選択部から受け取った分割前の問合せに基づいた処理を，データベース移動完了後に自サイトの実行部に依頼する．そして，その結果をクライアント・アプリケーションに返す．

DB 情報管理部： DB 情報表を管理して，システムの各部に必要な情報を提供する．DB 情報表の管理法としては，ブロードキャストされる DB 要求メッセージ，移動完了通知，施錠要求メッセージなどを監視して，自サイトのデータベース位置情報やデータベースに関するその他の情報を更新する．

並行処理制御部： データベースに対する処理要求（書込み，読出し，移動）を，施錠の状態に基づいて，適当なタイミングで実行部に渡す．

実行部： データベースに対する直接の処理（書込み，読出し，移動）を管理する．具体的には，並行処理制御部を経由してトランザクション発生サイトのコーディネータから受け取った問合せ（部分問合せ）に基づいた処理を実行し，その結果を返す．更に，トランザクション発生サイト（の DB 移動部）から受け取った DB 要求メッセージ中のリストに，自サイトで保持しているデータベースが存在すれば，そのデータベースをトランザクション発生サイトへ転送する．移動完了通知を受け取った後，移動失敗時のために保持しておいたそのデータベースのコピーを消去する．

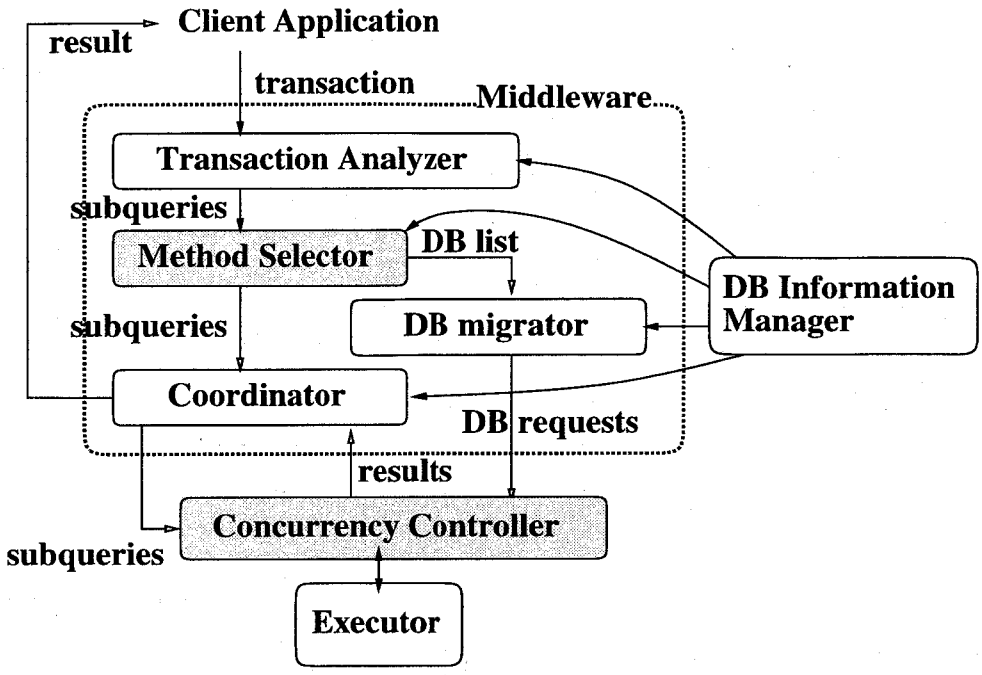


図 2.6: DB-MAN のシステムの概要

これらのうち，手法選択部と並行処理制御部がシステムの性能に最も大きく影響する．そこで以下では，手法選択部と並行処理制御部について更に詳しく議論する．

2.4.2 トランザクション処理手法の選択機構

2.3.2節で示したように，固定処理では処理に必要な通信回数 (n) が，データベース移動を用いた処理では移動するデータベースの総サイズが，処理時間に大きく影響する．また，トランザクションのアクセスパタンを考慮することで，更に効率的な手法選択が行えるものと考えられる．そこで DB-MAN では，トランザクションを処理する上でより効率的な手法を選択するために，手法選択部の機能として，単純モードと履歴統計モードの二つのモードを考える．単純モードは，発生したトランザクションを固定処理とデータベース移動を用いた処理で処理する場合のそれぞれの通信所要時間を見積もり，その値が小さくなる手法を選択する．一方，履歴統計モードでは，単独のトランザクションだけでなく，トランザクションのアクセスパタンも考慮した選択を行う．

単純モード

変数 t_1 を次のように定義する．

$$t_1 = T_{DB} - T_{fix} \tag{2.3}$$

このとき，単純モードでは，次のように処理手法を選択する．

if $t_1 < 0$ then	データベース移動を用いた処理
else	固定処理

ここで，T_{DB} を計算するためには，移動しなければならないデータベースの総サイズ $\sum_{D_j \in \boldsymbol{D}_N} Size(D_j)$ を知る必要があるため，仮想LAN内の各サイトで，DB-idとそのDB-idをもつデータベースのサイズ情報の表を保持するようにする．これにより，トランザクション解析部によって処理に必要なデータベースが同定されれば，$\sum_{D_j \in \boldsymbol{D}_N} Size(D_j)$ の値を計算することができる．また，各サイトにおいて，自分のサイトにあるデータベースのサイズが変動する度に全サイトに知らせるのは非効率的であるため，サイズ情報の表で保持している値と実際の値の差が一定値を越えた場合にのみその値をブロードキャストする．

履歴統計モード

単純モードでは，一つのトランザクションを処理する上で固定処理とデータベース移動のうち効率的な手法を選択した．しかし，一つのトランザクションで効率的である選択が，長期的には非効率的となる場合がある．例えば，現トランザクションだけで判断すると固定処理の方が効率的な場合でも，連続して同じサイトから同一のデータベースを用いるトランザクションが発生する場合には，先にデータベース移動で処理をしておいた方が効率的である．実際の環境においても，入れ子トランザクション [58] などのように，特定のデータベースを用いるトランザクションが連続的に発生することが予め予測できることは多い．

従って，以下のようなことを考慮することで，更に有効な手法選択を行うことができる．

- 一般に，分散システム内の各サイトはそれぞれ特有のトランザクション発生頻度やデータベースに対するアクセスパタンをもつため，データベースの利用履歴をとって，各データベースをその利用頻度が（特に最近）高いサイトに位置させることが有効であると考えられる．

DB-id	Size	Current Loc	Cnt	Usage-log				
D_1	50	S_1	1	S_1		$\cdots$	S_2	S_1
D_2	35	S_2	0	S_1	S_4	$\cdots$		
$\vdots$	$\vdots$	$\vdots$	$\vdots$			$\vdots$		
D_m	40	S_4	1			$\cdots$	S_2	S_1

図 2.7: DB 情報表

- 入れ子トランザクションやアプリケーション側で明示的に使用するデータベースを指定する場合など，近い将来に自サイトで発生するトランザクションのアクセスパタンが明らかである場合が多い．従って，サイト S_Iが特定のデータベース D_jを連続使用することが分かっている場合，サイト S_Iが D_jをもつ優先度を上げることが有効であると考えられる．

そこで，履歴統計モードでは，単純に式 (2.3) によって手法を選択するのではなく，履歴情報や連続使用の情報を保持することにより，上述の点を考慮した選択を行う．履歴統計モードで保持する DB 情報表を図 2.7に示す．図中の表の属性は，次の記法を用いている．

DB-id: データベース識別子

Size: DB-id で示されるデータベースのサイズ

Current Loc: データベースの現在位置

Cnt: データベースの連続使用宣言がされているか（0: されていない，1: されている）

Usage-log: データベースのアクセス履歴（一列が一つのトランザクションを示す）

Cnt と Usage-log はトランザクション毎に更新されるため，更新情報をトランザクション毎にブロードキャストする必要がある．データベース移動で処理する場合には，Usage-log は，DB 要求，移動完了通知メッセージの内容から直接決定される．Cnt の更新情報もこれらのメッセージに付加して転送することが可能である．一方，固定処理において，これらの更新をデータベース移動を用いた処理と同様の方法で全サイトに伝えるために，トランザクション発生時に最初に使用するデータベースと連続使用の情報をブロードキャストする．但し，これらの情報は,（後述する）デッドロック回避のためにブロードキャストする施錠要求メッセージに付加されるため，履歴統計モードだけでなく単純モードにおいても固定処理の通信所要時間は式 (2.1) より $d_m + d_{MCS}$だけ大きくなる．

$$T_{fix} = (n+5)\cdot d_m + d_{MCS} + C(k) \qquad (2.1)'$$

ここで，データベース D_jがサイト S_Iに存在する有効性を示す目的関数 $f(S_I, D_j)$ を次のように定義する．

$$f(S_I, D_j) = \alpha \cdot P' \cdot |L| + \sum_{l=1}^{|L|} \{k_l \cdot (|L| + 1 - l)\}. \qquad (2.4)$$

$$\alpha = \begin{cases} 1: & S_I\text{が } D_j\text{の連続使用を宣言している，あるいは} \\ & \text{現トランザクションで } D_j\text{の連続使用を宣言する．} \\ 0: & \text{その他．} \end{cases}$$

P': 連続使用優先係数

$|L|$: DB のアクセス履歴のサイズ
（トランザクション L 回分の履歴を残す）

$$k_l = \begin{cases} 1: & l\text{回前のトランザクションで } S_I\text{が } D_j\text{を利用．} \\ 0: & \text{その他．} \end{cases}$$

$f(S_I, D_j)$ は，前述の履歴統計モードで注目している二点のうち，第一項で第二点（連続使用宣言しているサイトの優先）を，第二項で第一点（利用頻度が高いサイトの優先）を反映している．ここで，**連続使用優先係数** P は，連続使用指定に対しての優先度の度合を表すものである．

次に，データベース D_jを現在位置するサイト S_Cからサイト S_Iに移動する有効性を示す移動評価関数 $G(S_I, D_j)$ を次のように定義する．

$$G(S_I, D_j) = f(S_I, D_j) - f(S_C, D_j) \qquad (2.5)$$

サイト S_Iが（所持していない）データベース集合$\boldsymbol{D}_N$を必要とするトランザクションを発生するとき，評価値 t_2を $G(S_I, D_j)$ の定義に基づいて各 $D_j \in \boldsymbol{D}_N$の $G(S_I, D_j)$ の平均値として次のように定義する．

$$t_2 = \frac{1}{|\boldsymbol{D}_N|} \sum_{D_j \in \boldsymbol{D}_N} G(S_I, D_j) \qquad (2.6)$$

この t_2は，現トランザクションをデータベース移動を用いて処理することの有効性を表している．

最後に，固定処理とデータベース移動を用いた処理のどちらを選択するかを示す値として t_{sel}を定義すると，履歴統計モードでの処理手法の選択方法は次のようになる．

$$t_{sel} = t_1 - K \cdot t_2 \tag{2.7}$$

if $t_{sel} < 0$	then	データベース移動を用いた処理
else		固定処理

K は t_2の係数であり，この K の値の設定がシステム全体の性能に大きく影響する．従って，ネットワークやシステムの管理者などの専門家が，ネットワークやデータベースの規模，アプリケーションの性質に応じてこの値を決定する．以下では K を**履歴依存係数**と呼ぶ．

2.4.3 DB-MAN における並行処理制御機構

本節では，DB-MAN の並行処理制御機構について説明する．データベース移動をトランザクション処理に利用する場合，データベース移動がデータベースといった大きなデータ単位を占有する処理であることを考慮すると，アクセスの繁雑時に次のような理由で処理効率が低下することが予測される．

- システム内のデータベース数は，読み書き操作の対象となるデータ項目の数に比べてかなり小さいために，データベース移動の要求の競合が頻繁に起こり，移動命令のデッドロックが発生する．
- データベースの移動にはある程度の時間がかかるため，その間，そのデータベース内のすべてのデータ項目に対する読み書き操作の処理を停止しなければならない．

更に，本システムがデータベース移動を用いた集中処理によってトランザクション処理の効率化を図っていいることや，DB-MAN における手法選択がデータベースの現在位置や履歴情報に基づいて実行されることを考慮すると，アクセス繁雑時には次のような問題が生じてしまう．

表 2.1: 施錠の排他関係

要求　状態	read	write	move	location
read	○	×	○	○
write	×	×	○	○
move	○	×	×	×
location	○	○	–	–

- データベースを集めて集中処理しているサイトのデータベースが，他サイトが発生したトランザクションによって移動されてしまい，集中処理の効果がなくなってしまう．
- 処理待ちのトランザクションが存在すると，データベースの現在位置や履歴情報から手法選択をしても，そのトランザクションの実行時にはデータベースの位置や履歴が変化してしまい，効果的な手法選択とならない．

これらの四つの問題点を解決するためにDB-MANでは，二相施錠規約を拡張し，データ項目に対する施錠として従来の書込み施錠（write lock），読出し施錠（read lock）にデータベース移動のための施錠として移動施錠（move lock）を加え，更に，データベース全体に対する施錠として位置施錠（location lock）を加える．移動施錠はデータベースを移動するためにかける施錠で，そのデータベースの全てのデータ項目に対して移動施錠がかけられた時点でデータベースの転送を開始することができる．位置施錠はデータベースの位置を固定するための施錠で，これがかけられている間はそのデータベースを他のサイトによって移動されることがない．それぞれの施錠の排他関係を表2.1に示す．

DB-MANでは，これらの施錠に基づき，以下の手順でトランザクションの処理を行う．

1. トランザクションの発生時に，全ての必要な施錠の要求をブロードキャストする．DB移動を用いて処理する場合はこの情報をDB要求メッセージに付加することで実現できるが，固定処理で処理する場合は個別の施錠要求メッセージが必要になる．
2. ブロードキャストされる施錠要求メッセージはMCSを介して全サイトに送信されるため，MCSにおいて連番のトランザクション識別子（Tid）を施錠要求メッセージ

に付加する．トランザクション発生サイトでは，ブロードキャストされて戻ってきたメッセージから自分のトランザクションに与えられたTidを知り，転送する処理依頼メッセージにそのTidを付加する．これにより，データベース所持サイトでは，到着した処理依頼メッセージがどのトランザクションのものか同定可能になる．

3. データベース所持サイトでは，一つのデータベースに対してデータ項目毎の施錠キュー（読出し，書込み，移動施錠が対象）を保持する．トランザクション発生サイトが発した施錠要求は各キューに挿入される．但し，移動施錠は，移動すべき各データベースの全データ項目の施錠キューに挿入される．

4. 各サイトにおいて，施錠されたデータ項目に対して処理が行われる．DB移動に関しては，移動すべきデータベースの全てのデータ項目に移動施錠がかけられた時点でデータベースの転送が開始される．表2.1から分かるように，書込み施錠がされている場合に移動施錠をかけることができないため，むやみに移動が起こったり，DB移動を用いて処理するトランザクションを優先的に扱うこともなく，ほぼトランザクションの発生順に処理が行われる．

以下では，このような手順によって上述の四点が実現されることを順に説明する．

(1) デッドロックの回避

各トランザクションに与えられるTidがMCSで付加される連番であることから，トランザクションの絶対順を定義することができる．

従って，何らかの原因で複数の施錠要求メッセージの到着順が逆転した場合にも，Tidを比べることで順序の変更が可能となる．しかし，待ち状態のトランザクションがないときに到着順が逆転すると，誤った順序で処理が開始することが考えられる．この場合，後から到着した施錠要求のTidが現在処理中のトランザクションのTidより小さければ，処理中のトランザクションを中止し，後から到着した方を優先的に処理することで解決する．従って，いかなる場合にも施錠のデッドロックは起こり得ない．

(2) 処理待ちトランザクションを考慮した手法選択

データベースの移動は，データベースの現在位置とトランザクション発生サイトのどちらにそのデータベースが存在する方が効率的かという点を考慮して決定されるため，各サイトは他のサイトがブロードキャストするDB要求（移動施錠要求）メッセージを監視す

ることで自分の発生するトランザクションの実行時には必要なデータベースがどこに存在するかを予測する．

単純モードの場合は各データベースの実際の現在位置とトランザクション実行時の予測位置を両方とも保持し，手法選択は後者の情報に基づいて行う．履歴統計モードの場合は，更に，(DB 要求メッセージと固定処理の両方の）施錠要求メッセージを監視して，処理待ちのトランザクションの情報を保持し，自分のトランザクションの実行時の各データベースの履歴情報を予測する．つまり，現在までに実際に実行された L 個の履歴情報だけでなく，自分のトランザクションの L 個前までに実行されるものの予測履歴情報を別に保持し，手法選択の際には後者の方を利用する．これらの予測情報は，トランザクションがアボートされたりデータベースの移動が失敗しない限り正確なものである．アボート時などの情報のリカバリについては後述する．

また，誤った予測情報を用いた手法選択によってデータベース移動が起こらないように，トランザクション発生サイトは移動施錠要求メッセージに，移動すべきデータベースの DB-id だけでなく，トランザクション実行時のそのデータベースの予測位置を記す．各データベースの所持サイトでは，最新の移動要求でそのデータベースがどのサイトに移動することになっているかという情報を保持しておき，新たに到着した移動施錠要求メッセージ内の予測位置がその情報と一致した場合にのみ，移動施錠が全データ項目の施錠キューに挿入される．一致しない場合はその要求は破棄される．

(3) 移動中のデータベースに対する処理のサポート

データベースの移動中は移動失敗時に備えてデータベースの送信元にそのコピーが保持されている．そこで，そのコピーに対して処理を行うことで，データベースの移動中においても書込み操作と読出し操作を継続することができる．コピーに対して行われた操作は，データベース移動完了後にその操作のログ情報を移動先へと送信することで実際のデータベースに反映される．また，ログ情報の送信後に更に到着した処理依頼メッセージに関しては，そのままデータベースの移動先へとフォワードする．

(4) 集中処理を行っているサイトの保護

データベースの全データ項目に移動施錠がかけられた時点で，データベースの移動が開始されるが，これと同時に各データ項目の移動施錠は解除され，データベース全体に位置施錠がかけられる．

表2.1に示すように，DB-MAN では位置施錠がかけられている状態で移動施錠をかけることができない．また，データベース移動を用いてトランザクションの処理を行っているサイトがそのトランザクションの実行している間，位置施錠はかけられたままになっている．従って，トランザクション発生サイトは，処理中に他のサイトによってデータベースを移動されることなく，効率的に処理が行える．

2.4.4 リカバリ機能

本節では，DB-MAN システムのリカバリ機能について説明する．DB-MAN システムは，障害発生時に備えてデータベースの移動を考慮した復旧機能を有する．また，データベース移動が失敗時に備えて，手法選択機構で用いる DB 情報の復旧機能を有する．

移動するデータベースの障害復旧

主記憶データベースを用い，前節の物理記憶管理法を採用することで，高速なデータベースの移動を実現できる．一方，主記憶が揮発性の記憶装置であることを考慮すると，さまざまな障害に対する対策が重要になる．

主記憶データベースの障害復旧法に関してはこれまでに多くの研究がなされており，最も一般的なものとして，redo ログのみを二次記憶に保存する手法が用いられている．つまり，トランザクション実行中には，redo ログと undo ログの両者を主記憶上に保持しておき，トランザクションがコミットされれば redo ログのみを二次記憶に書き込み，undo ログは消去する．これは，主記憶データベースでは主記憶障害時にデータベース自体が消去されてしまい，従来の二次記憶ベースのデータベースのように主記憶上のバッファと二次記憶中のデータベースの間の一貫性を保持する必要がないことから，undo 操作はトランザクションのアボート時にしか必要ないためである．ただし，この場合，チェックポイントはトランザクションの終了単位でしか行えない．

このような手法では，チェックポイント時と主記憶障害発生時に次のような処理を行う．

[チェックポイント時]

- チェックポイント時に最後に終了したトランザクションの識別子（もしくはコミット番号）の記録
- チェックポイント時の最終 redo ログの終端アドレスの記録

- 二次記憶上に保持されているデータベースのバックアップの更新

  （その後はそれまでの redo ログを適時消去する）

[主記憶障害発生時]

- 最終のチェックポイント以降の redo ログと二次記憶上のバックアップからデータベースを復旧

DB-MAN システムにおいても，基本的にはこのような従来の手法によって障害復旧を行う．但し,（主記憶上の）データベースが 移動することから，二次記憶上のバックアップも移動するかしないか，二次記憶に記録する redo ログは更新操作が行われた各サイトで保持するか，それとも，データベースが現在存在するサイトに集めておくかで，チェックポイント時や障害発生時の処理手順や処理時間が大きく異なってくる．

そこで DB-MAN システムでは，以下の三つの手法に基づいて障害復旧を行う．

(1) 完全追従法

データベース移動に伴い，二次記憶上のバックアップと redo ログの両方を，データベースの移動先へ移動する．移動元のサイトでは，データベース移動時に，最終のチェックポイント以降の redo ログおよびチェックポイント時に記録された情報を，データベースと一緒に移動先へと転送する．移動先のサイトでは，受け取ったデータベースを二次記憶に書き込みバックアップを作成し，更に，受け取ったその他の情報を二次記憶に書き込む．これらの情報は，次のデータベース移動やチェックポイント時の処理が高速に行えるように，主記憶上にも保持しておく．

従って完全追従法では，チェックポイント時と障害発生時にデータベース移動の導入に伴う特別な処理は必要なく，従来の手法と同等な処理をローカルに行えばよい（図 2.8(a)）．また，チェックポイント時のデータベースのバックアップの更新は，主記憶上の redo ログを用いることで効率的に実行できる．

(2) ログ追従法

データベースの移動時には，redo ログのみをデータベースの移動先へ移動する．つまり，移動先のサイトでは，受け取ったデータベースのバックアップを作成しない．それ以外の処理は完全追従法と同じである．

従ってログ追従法では，データベースのバックアップは特定のサイトで保持されているため，チェックポイント時と障害発生時に次のような処理が必要になる．

[チェックポイント時]

データベースのバックアップがリモートサイトにある場合は，そのサイトに対して最終のチェックポイント以降の redo ログを転送する．バックアップを保持するサイトでは，そのログに基づいてバックアップを更新する．

[主記憶障害発生時]

データベースのバックアップがリモートサイトにある場合は，そのサイトに対して最終のチェックポイント以降の redo ログを転送する．バックアップを保持するサイトでは，そのログに基づいてデータベースを復旧する（図 2.8(b)）．

(3) 非追従法

この手法ではデータベース移動時に，ログ情報の移動もバックアップの移動も行わない．redo ログは，最終のチェックポイント以降に更新が行われた各サイトに分散して保持されている．

従って，チェックポイント時と障害発生時に次のような処理が必要になる．

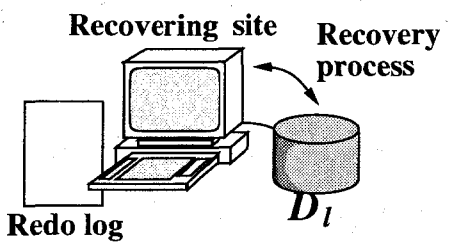
[チェックポイント時]

最終のチェックポイント以降にデータベースが移動してきたサイトを逆順にたどって，最終のチェックポイント以降の redo ログを収集する．それが完了した後は，データベースのバックアップを保持しているサイトに集めた redo ログを転送する．バックアップを保持するサイトでは，そのログに基づいてバックアップを更新する．

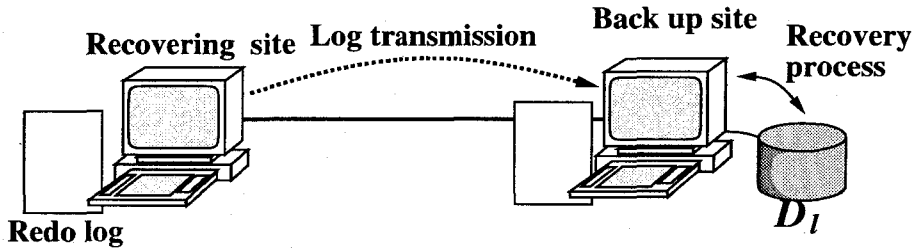
[主記憶障害発生時]

最終のチェックポイント以降にデータベースが移動してきたサイトを逆順にたどって，最終のチェックポイント以降の redo ログを収集する．それが完了した後は，データベースのバックアップを保持しているサイトに集めた redo ログを転送する．バックアップを保持するサイトでは，そのログに基づいてデータベースを復旧する（図 2.8(c)）．

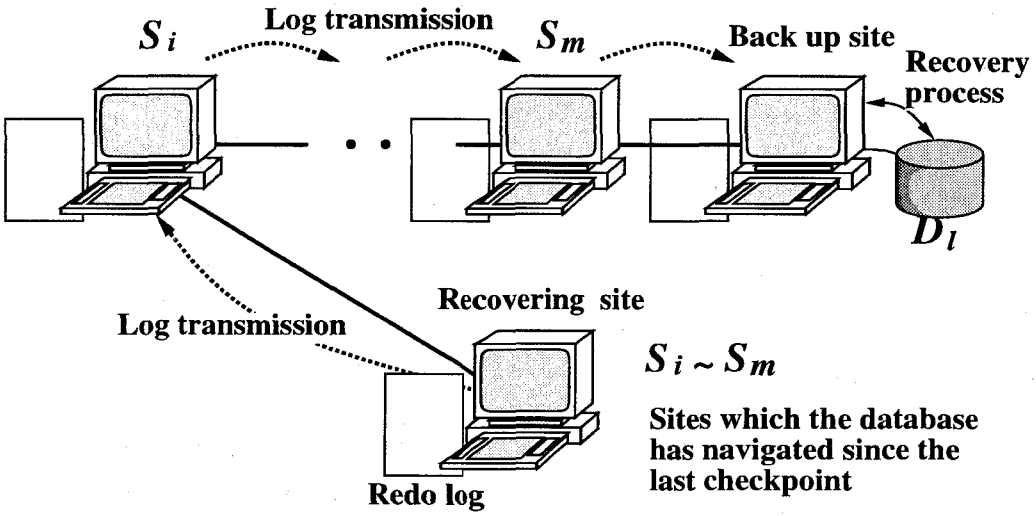
これらの三手法は，データベース移動時の処理とチェックポイント時，障害発生時の処理の手間が大きく異なる．データベース移動時の処理では，完全追従法，ログ追従法，非



(a) 完全追従法



(b) ログ追従法



(c) 非追従法

図 2.8: 各手法の障害復旧時処理

追従法の順で手間がかかり，特に，完全追従法ではデータベース全体を二次記憶へ書き込むため，他の二手法に比べてかなり処理時間が大きくなる．また，チェックポイント時，障害発生時の手間はその逆になり，特に，非追従法では多くの通信を要するために，他の二手法に比べてかなり処理時間が大きくなる．実際の環境では，分散システムの特性に応じて，これらの手法のいずれかを選択しなければならない．

DB 情報の復旧機能

DB-MAN では，履歴統計モードを用いている場合，手法選択の際にトランザクションが実行される時点でのデータベースの位置情報や履歴情報を予測する必要がある．そこで，トランザクションのアボートとデータベース移動の失敗が，履歴情報や位置情報に影響することを考慮すると，これらが生じた際のリカバリについて考える必要がある．DB-MAN では，リカバリを行う度合に応じて三つのレベルを設ける．

レベル 1

- データベース移動の失敗時にもトランザクションのアボート時にも，履歴情報，位置情報のリカバリを行わない．

レベル 2

- データベース移動失敗時は，トランザクション発生サイトがその旨を全サイトにブロードキャストする．各データベースの所持サイトでは，データベース移動を失敗したトランザクション以後のトランザクションの施錠要求をすべて破棄する．更に，全サイトにおいて，DB 情報表における予測履歴情報のうちそのデータベース移動で処理すべきトランザクション以後の情報をすべて破棄する．また，データベースの予測位置情報を実際の現在位置に変更する．そして，データベース移動失敗時に備えて保持しておいた処理済みの L 個の履歴情報とデータベースの現在位置を用いて，改めて処理手法の選択を行う．

- トランザクションのアボート時にはリカバリを行わない．

レベル 3

- データベース移動失敗時の処理はレベル 2 と同じ．

- トランザクションのアボート時にもリカバリを行う．アボートが発生した場合，トランザクション発生サイトはアボートメッセージをトランザクションに関わったデータベースの所持サイトだけでなく，全サイトにブロードキャストする．各データベースの所持サイトでは，アボートされたトランザクション以後のトランザクションの施錠要求をすべて破棄する．更に，全サイトにおいて，DB 情報表における予測履歴情報のうちそのデータベース移動で処理すべきトランザクション以後の情報をすべて破棄する．そして，データベース移動失敗時に備えて保持しておいた処理済みの L 個の履歴情報とデータベースの現在位置を用いて，改めて処理手法の選択を行う．

トランザクションの発生頻度が少なく，待ち状態のトランザクションがあまりないような環境では，リカバリのオーバヘッドが小さいため，レベル1が適している．トランザクションの発生頻度が高い環境では，リカバリのオーバヘッドが大きいので，リカバリのレベルを慎重に選択する必要がある．特に，データベースの位置情報の正確さが処理効率に大きな影響を与えることから，移動失敗時にリカバリを行うかどうかはオーバヘッドと処理効率の両者を考慮して決定しなければならない．具体的には，トランザクションの発生頻度が高く，移動失敗と問合せの発生の比が小さい場合は移動失敗時のリカバリを行わない．従って，この場合もレベル1が適している．一方，トランザクションの発生頻度が高く，移動失敗と問合せの発生の比が大きい場合，アボートと問合せの発生の比が小さければレベル2が，比が大きければレベル3が適している．

ここで，単純モードでは，履歴情報は手法選択に無関係であるため，リカバリのレベルは移動失敗のみに関する二つとなる．

2.5 シミュレーションによる性能評価

本節では，本研究において提案した DB-MAN システムの有効性を検証するために行ったシミュレーションの結果を示す．まず，DB-MAN のトランザクション処理手法選択機構の性能評価の結果を示し，次に，並行処理制御機構の性能評価の結果を示す．

2.5.1 手法選択機構の性能評価

本節では，DB-MAN のトランザクション処理手法選択機構の性能評価のために行ったシミュレーションの結果を示す．ここでのシミュレーションでは，並行処理制御機構を必要と

しないように，トランザクションは同時に発生することなく，一つのトランザクションが処理されると連続して次のトランザクションが発生する環境を想定している．また，履歴統計モードにおける連続使用優先係数 P，履歴依存係数 K の影響についても，シミュレーションによって示す．

選択モードの有効性の検証

DB-MAN の二つの選択モードを，シミュレーションによって性能評価し，固定処理およびデータベース移動のみを用いた処理と比較する．シミュレーションでは，10000 回のトランザクションを連続的に発生して，その処理に要する通信所要時間の総和を計算していく．

用いるパラメータの値を表2.2に示す．分散システム内のサイト数およびデータベース数はそれぞれ 20 とする．1000 回毎に各サイトのトランザクション発生頻度を p_1,p_2と変化させることで，各モードの適応性を検証する．通信回数を 1〜30 でランダムに変化させることで，簡単な書込み操作のみのトランザクションや結合操作を複数含むような複雑なトランザクションなどさまざまな性質のトランザクションを表現できる．また，データ転送に関するパラメータは，近い将来に実現されると思われる全世界規模の分散システムを想定したものである．つまり伝搬遅延は，光ファイバケーブルを用いた ATM ネットワークで各サイト間の平均ホップ数（経由する交換機数）が 10 程度，各交換機でのルーチング遅延が市販の ATM 交換機同様に 5 ミリ秒程度と仮定して設定している．データベース移動のための帯域幅は，ネットワーク全体の帯域幅が数 Gbps 程度になることを予想して 1Gbps と設定している．また，データベースのサイズは，アプリケーションの性質などに依存し，この値の設定がデータベース移動を用いた処理の通信所要時間に大きく影響するが，今回は，最も顕著な例として，固定処理の通信所要時間とデータベース移動を用いた処理の通信所要時間がほぼ等しくなるように設定した．なお，連続使用優先係数 P，履歴依存係数 K は，2.5.1節で示すシミュレーションによって選択した最適値を用いている．ただし，シミュレーションでは簡単のために，データベース移動の失敗や処理依頼メッセージの転送の失敗が起こらないような健全なネットワークを仮定し，更に，トランザクションのアボートも発生しないものと仮定する．

シミュレーションでは，次の手順を繰り返し実行する．

1. まずトランザクション発生頻度に応じてトランザクション発生サイトを決定する．

2. トランザクションに必要なデータベースの総数 $|\boldsymbol{D}_U|$ を 1 から $|\boldsymbol{D}|$ の範囲でランダ

表 2.2: シミュレーションのためのパラメータの設定

パラメータ	パラメータの意味	値
$\|\boldsymbol{S}\|$	仮想 LAN 内のサイト数	20（サイト識別子 $S_1, \cdots, S_{20}$）
$\|\boldsymbol{D}\|$	データベースの総数	20（DB-id $D_1, \cdots, D_{20}$）
$Size(D_j)$	各データベースのサイズ	45+1.5i [Mbytes] (i=0,$\cdots$,19)
p_1	トランザクション発生頻度 1	0.025 ($S_1, ..., S_{10}$), 0.05 ($S_{11}, ..., S_{18}$), 0.15 (S_{19}), 0.3 (S_{20})
p_2	トランザクション発生頻度 2	0.025 ($S_{11}, ..., S_{20}$), 0.05 ($S_3, ..., S_{10}$), 0.15 (S_2), 0.3 (S_1)
d_{MCS}	MCS への平均伝搬遅延	0.12 [s]
d_m	その他の 2 サイト間の伝搬遅延	0.12 [s]
B_M	データベース移動のための予約帯域幅	1 [Gbps]
$C(k)$	コネクション設定時間	$C(k) = 0.3k$ [s] と仮定
n	固定処理における通信回数	$n'\|\boldsymbol{D}_N\|/\|\boldsymbol{D}_U\|$（$n'$は 1〜30 でランダムに決定）
$\|L\|$	利用履歴のサイズ	20
P	連続使用優先係数	0.02
K	履歴依存係数	0.10

ムに決定する ($|\boldsymbol{D}_U| \geq |\boldsymbol{D}_N|$). もし，トランザクション発生サイト S_Iがあるデータベースに対して連続使用宣言をしている場合，$|\boldsymbol{D}_U|$ の値が連続使用宣言をしているデータベースの総数よりも小さければ，$|\boldsymbol{D}_U|$ をその値に再設定する．

3. トランザクションに必要なデータベースとして $|\boldsymbol{D}_U|$ 個のデータベースを選択する．但し，連続使用宣言をしているデータベースは必ずこれに含まれなければならない．

4. トランザクションの連続度βを 0,1,2 の中からランダムに決定する．この値は，現トランザクション以後に S_Iが何回連続してトランザクションを発生するかを表している．もし，この値が 0 より大きい場合は，以後β回，S_Iのトランザクション発生頻度を 1.00 増加させる．

5. 連続使用するデータベースを 3 で決定したデータベースの中からランダムに決定する．

6. t_{sel}の値に基づいて，固定処理若しくはデータベース移動を用いた処理を選択する．

7. 選択された手法に応じて，通信所要時間の総和を増加させる．但し，各手法およびモードでの通信所要時間は，式 (2.1)'，式 (2.2) より得られる計算値を実際の値と考える．

更に，n と $|\boldsymbol{D}_U|$ の決定条件のみを変えて，同様のシミュレーションを行った．n と $|\boldsymbol{D}_U|$ は（負の領域は排除した）正規分布に基づいて決定され，n では平均に 15，標準偏差に 2.5 を，$|\boldsymbol{D}_U|$ では平均に 10，標準偏差に 1.67 の値をそれぞれ用いる．つまり，この環境では，各トランザクションの n と $|\boldsymbol{D}_U|$ の値が平均値に近い場合が多く，そのため，固定処理とデータベース移動を用いた処理の通信所要時間も比較的近い値になる．

シミュレーションの結果を図 2.9, 図 2.10に示す．図 2.9は表 2.2の環境での結果を，図 2.10は n と $|\boldsymbol{D}_U|$ の決定条件を正規分布に基づくものに変えた場合の結果をそれぞれ示している．各グラフにおいて横軸はトランザクション処理数，縦軸は通信所要時間の総和である．また，'DB-migration' はデータベース移動を用いた処理，'Fixed-processing' は固定処理，'Simple mode' は単純モード，'Log-statistics mode' は履歴統計モードをそれぞれ表している．図 2.10では，グラフでは二本の線しか識別できないが，これは DB-MAN の単純モード，データベース移動のみを用いた処理，固定処理がほぼ同じ値を示しているために結果の線が重なってしまっているからである．

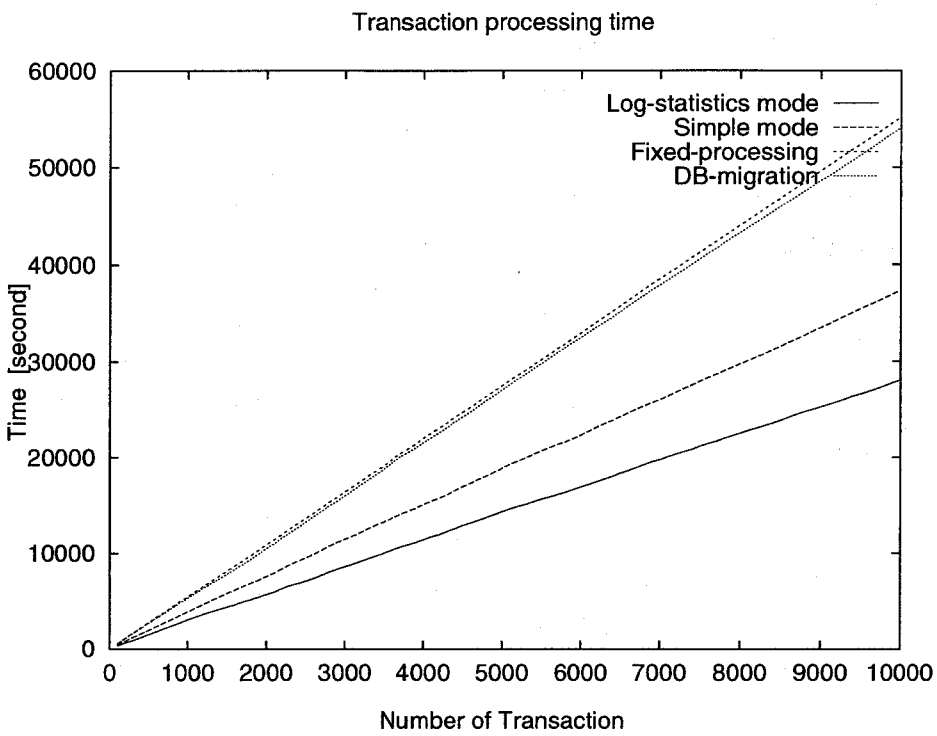


図 2.9: 各手法および各モードの通信所要時間の比較（表 4.1 の環境）

シミュレーションの結果より，DB-MAN における処理手法選択機構の有効性が認められる．しかし，図 2.10から分かるように，単純モードでは個々のトランザクションの通信所要時間のみに着目して手法選択が行われるため，固定処理とデータベース移動を用いた処理の通信所要時間が毎回ほぼ等しくなるような環境ではその有効性が小さくなる．一方，履歴統計モードでは，各サイトのデータベースの利用頻度やトランザクションの連続性も考慮して処理手法を選択するため，いずれの環境においても最も良い結果を示している．

連続使用優先係数 P，履歴依存係数 K の影響

本節では，連続使用優先係数 P，履歴依存係数 K が履歴統計モードを用いた場合の通信所要時間に与える影響をシミュレーション結果より考察する．

表 2.2に基づくシミュレーション環境において，P，K の値を P は 0 から 2 まで 0.05 ずつ，K は 0〜1 まで 0.05 ずつ変動させて履歴統計モードを用いた場合の通信所要時間を計測した結果を図 2.11に示す．グラフでは，x 軸が K の値，y 軸が P の値，z 軸が通信所要時間の平均値を表している．図 2.11の結果より，P，K の値によりトランザクション処理

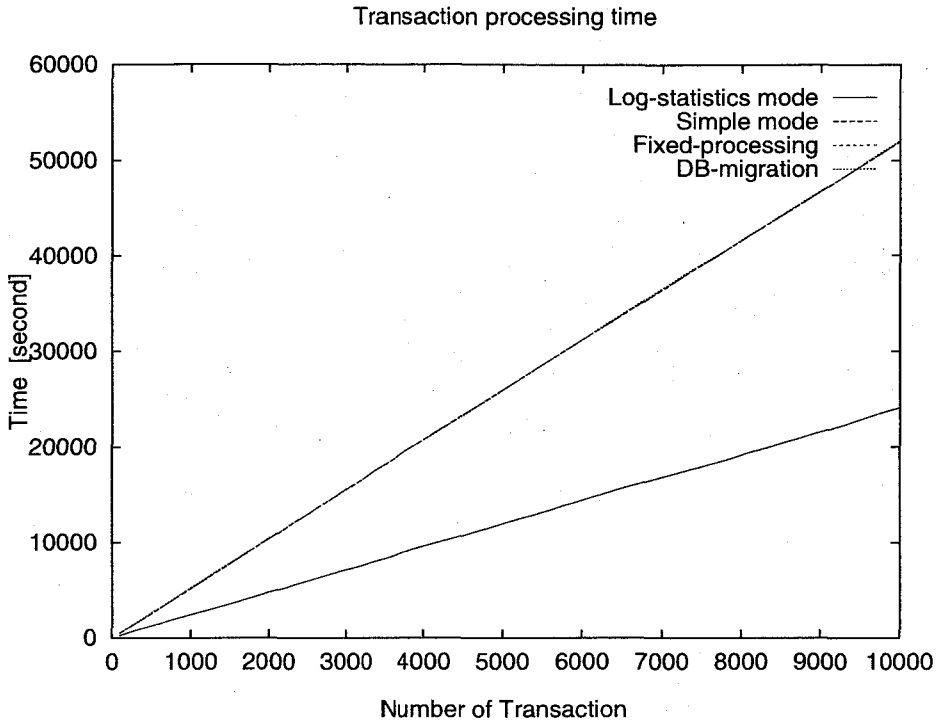


図 2.10: 各手法および各モードの通信所要時間の比較（正規分布に基づく場合）

時間が大きく変動することが分かる．この環境では，$K = 0.02 \sim 0.20$, $P = 0.00 \sim 0.20$ の範囲分）で良い結果が観測される．

そこで，更に詳しく観察するために，K を $0.02 \sim 0.20$ の範囲で 0.005 ずつ変動させ，P を $0.00 \sim 0.20$ の範囲で 0.005 ずつ変動させて，同様の環境でトランザクション 10000 回分の通信所要時間を計測した．その結果を図 2.12に示す．グラフでは，x 軸が K の値，y 軸が P の値，z 軸が通信所要時間の平均値を表している．

この結果から，$(K, P) = (0.090, 0.000 \sim 0.005)$, $(0.100, 0.000 \sim 0.040)$, $(0.105, 0.000 \sim 0.035)$, $(0.110, 0.000 \sim 0.010)$ の範囲で通信所要時間は最小値 2.81 秒を示している．実際のシステムにおいて履歴適応手法を実装し運用するに当たっては，P，K の値を詳細なシミュレーションによって決定する必要がある．但し，シミュレーションで実際の環境を正確に表すことは困難であるため，突出的によい性能を示す P，K 値は選択せず，広範囲でよい性能を示す P，K 値を実システムに採用すべきである．例えば，前節での P, K の値としては，0.02, 0.10 をそれぞれ用いた．

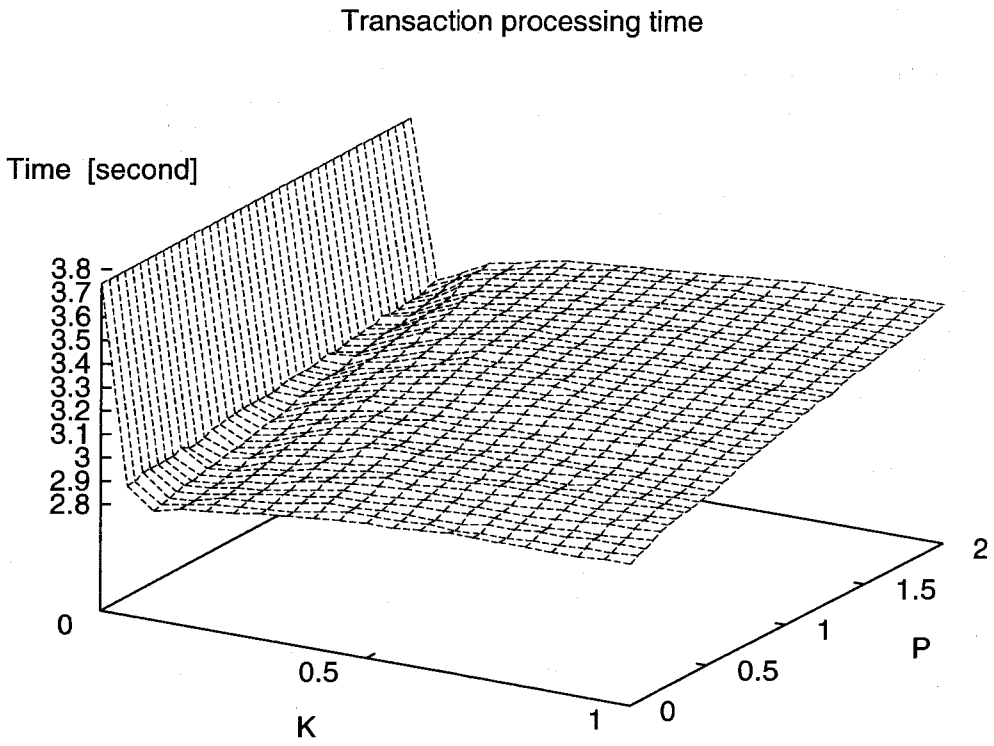


図 2.11: 履歴統計モードにおける通信所要時間に対する履歴依存係数 K および連続使用優先係数 P の影響

2.5.2 並行処理制御機構の性能評価

本節では，DB-MAN の並行処理制御機構の性能評価のために行ったシミュレーションの結果を示す．シミュレーションでは，基本的には，表2.2に基づいた環境を想定する．ただし，トランザクションは 2.5.1 節のシミュレーションのように連続的に発生するものではなく，指数分布に基づく発生間隔で発生するものとする．また，データベース移動のための帯域幅を 1.0Gbps と 2.0Gbps の場合でそれぞれシミュレーションをする．そして，トランザクションの発生間隔の平均値を変化させて，DB-MAN の単純モード，履歴統計モードでの処理時間と，純粋な二相施錠規約に基づく固定処理，データベース移動のみを用いた処理の処理時間を比較する．シミュレーションでは，10000 回のトランザクションの処理時間の平均値を計算する．データ項目，トランザクションの性質等に関しては，最も顕著な場合を想定して，以下のような条件をもつものとする．

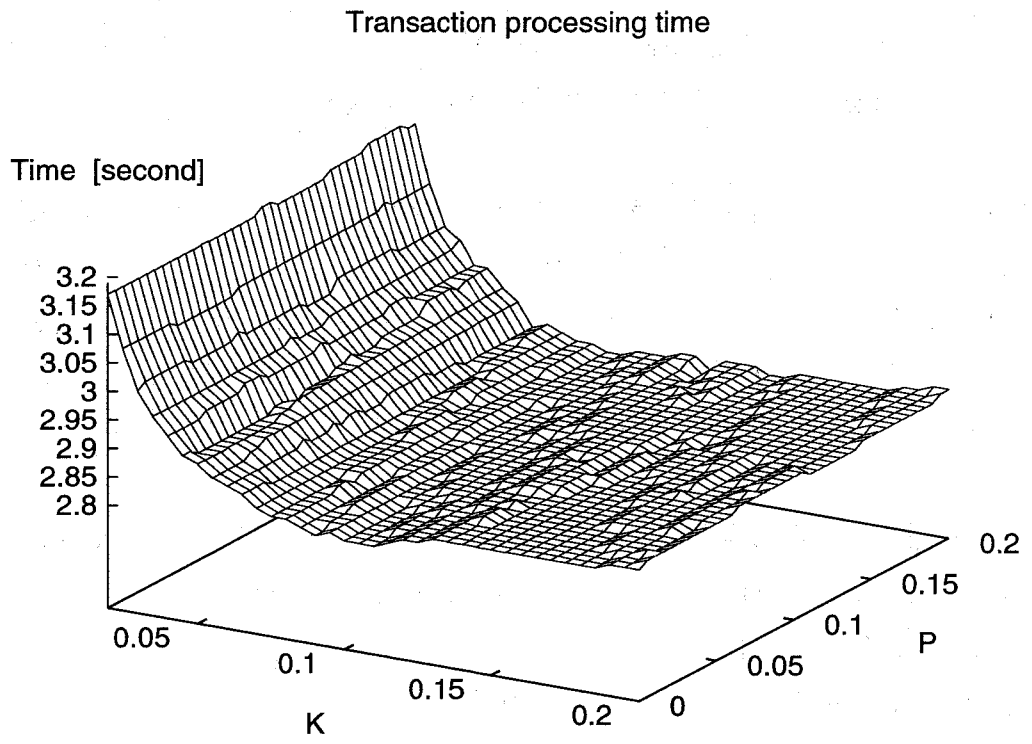


図 2.12: 履歴統計モードにおける通信所要時間に対する履歴依存係数 K および連続使用優先係数 P の影響（詳細）

- 各データベースは，30 個のデータ項目から構成される．各データ項目のサイズが 1.5Mbytes から 2Mbytes 程度になるため，各データ項目は一つのリレーションのようなものと考えられる．

- トランザクションはすべて，全順序関係をもつ書込み操作からなるものとする．従って，直前の書込み操作が完了するまで，次の書込み命令は実行されない．つまり，固定処理の場合，直前の書込み操作が完了した時点で，次の操作のための処理依頼メッセージが転送される．

- 主記憶データベースを想定していることから，書込み操作に要する処理時間は，通信にかかる時間に比べて極めて小さい値になる．従って，書込み操作に要する時間はすべて 0 に近似する．

ここで，それぞれの書込み操作は一つのデータ項目に施錠をかけて実行されるため，シミュレーションの手順として2.5.1節の手順3の後に，各書込み操作の対象となるデータベースとデータ項目を決定する次のような手順が必要になる．

3'. n'回の書込み操作の対象となるデータベースとそのデータベースのデータ項目を決定する．n'回中，$n'|\boldsymbol{D}_N|/|\boldsymbol{D}_U|$回は，使用するデータベースのうちトランザクション発生サイトが所持していないデータベースに対する操作で，それ以外は所持しているデータベースに対する操作とする．それぞれの範囲内で，操作対象となるデータベースとデータ項目はランダムに選択される．

以上のような条件と手順のもとでシミュレーションを行った結果を，帯域幅が1.0Gbpsと2.0Gbpsの場合でそれぞれ図2.13, 図2.14に示す．但し，シミュレーションでは，2.5.1節と同様に，通信の失敗などが生じない健全なネットワークを想定し，トランザクションのアボートも発生しないものとしている．これらのグラフは，横軸がトランザクションの発生間隔の平均値を，縦軸が各モードおよび手法の平均処理時間を表している．'Log-statistics mode'はDB-MANの履歴統計モード，'Simple mode'は単純モード，'Fixed-processing'は固定処理，'DB-migration'はデータベース移動のみを用いた処理をそれぞれ表している．

この結果より，まずデータベース移動を用いた処理は，処理時間と性能が劣化する点がともにデータベース移動のための帯域幅に大きく影響されることが分かる．しかし，帯域幅が1.0Gbps, 2.0Gbpsのどちらの場合でも，トランザクションの発生頻度が高くなると，データベース移動を用いた処理の処理性能がいち早く劣化している．これは，すべてのトランザクションがデータベース移動を用いて処理されるため，施錠の単位がデータベースになってしまうからである．一方，筆者らの提案したDB-MANの両モードでは，トランザクション処理手法の選択機構により，処理性能が劣化するまでは，固定処理とデータベース移動を用いた処理の場合よりも高い処理効率を示している．特に，履歴統計モードは，帯域幅が1.0Gbps, 2.0Gbpsのどちらの場合にも最も高い処理効率を示している．ここで，帯域幅が2.0Gbpsの場合にDB-MANの両モードとデータベース移動を用いた処理の処理時間にさほど差がないのは，殆どのトランザクションにおいてデータベース移動を用いた処理の方が固定処理よりも大幅に処理時間が小さくなり，DB-MANの処理手法選択機構がデータベース移動を選択することが多いためである．更に，並行処理制御機構により，データベース移動を導入した場合の問題点が解消され，固定処理と同等の並行処理性能を示して

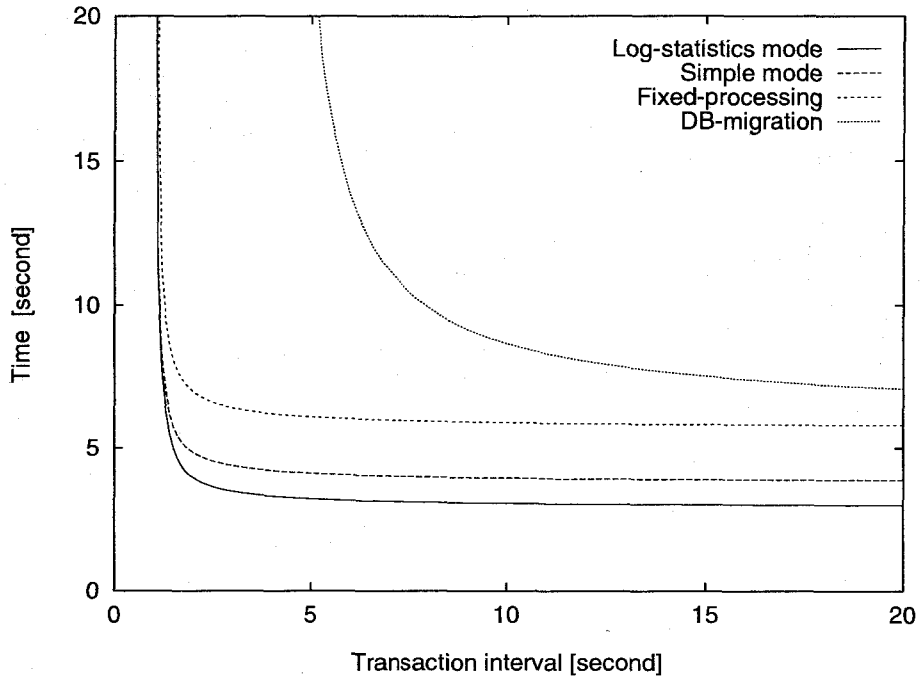


図 2.13: トランザクションの発生頻度と各手法の平均処理時間の関係（帯域幅= 1.0Gbps）

いる．以上のことから，DB-MAN の両機構の有効性が確認でき，DB-MAN の従来の固定処理に対する優位性が示された．

2.6 むすび

本章では，広帯域ネットワークの特性を有効に利用した，データベース移動に基づく分散データベースシステム DB-MAN を提案した．DB-MAN は，トランザクション処理手法の選択機構とデータベース移動を考慮した並行処理制御機構といった二つの特徴的な機構をもつ．前者は，従来の固定処理とデータベース移動を用いた処理とを適応的に選択する機構で，後者は，移動するデータベースに対するデータベース操作をサポートしたり，データベース移動の要求のデッドロックが生じないように移動を制御する機構である．また，移動するデータベースの障害復旧を効率的に行うために，redo ログとディスク上のバックアップを保持する方法として，三つの手法を提案した．これらは，システムの特性に応じて，使い分ける必要がある．

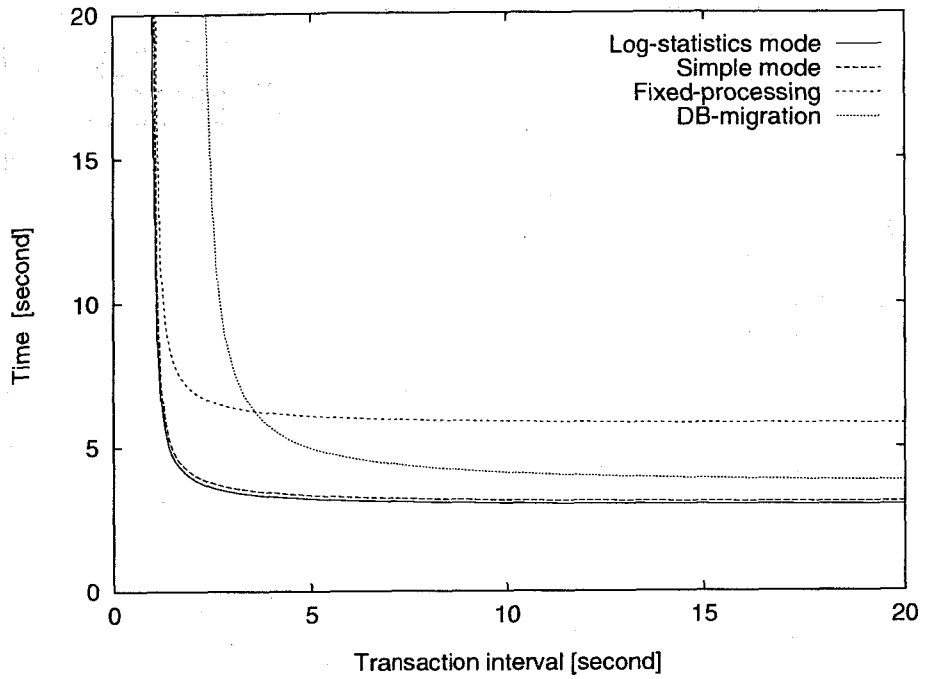


図 2.14: トランザクションの発生頻度と各手法の平均処理時間の関係（帯域幅= 2.0Gbps）

更に本章では，提案したシステムの有効性を検証することを目的として，シミュレーションによる性能評価も行った．手法選択機構の評価のためのシミュレーションの結果から，DB-MAN システムの履歴統計モードを用いることにより，処理効率が大幅に改善されることが確認できた．また，履歴統計モードの履歴依存係数と連続使用優先係数の設定が，処理効率に大きく影響することも分かった．更に，並行処理制御機構の評価のためのシミュレーションの結果から，DB-MAN で提案した並行処理制御法によって，データベース移動を導入することによる並行処理性能の劣化を防げることが確認できた．

第3章

データベース移動に基づく動的複製配置法

3.1 まえがき

第2章で提案した手法では，データベース移動後に転送元サイトのデータベースを削除することでデータベースの複製を作成しないようにしている．これは，複製を作成すると複製間の一貫性保持などの管理が複雑になり，そのためのメッセージ通信によって性能が低下することを考慮したものである．しかし，複製を作成することでデータベースへの読み書き操作のためのメッセージ通信を削減できることから，従来の分散データベースの分野では複製を用いたトランザクション処理手法が多く提案されている [4, 54, 72, 73, 78, 84].

複製を作成することの有効性は，ネットワークの帯域幅や各サイトの記憶領域の大きさなど，システムの特性に大きく依存する．筆者らが想定している広帯域ネットワーク上のシステムにおいても，システムの特性によって，複製を作成するか否かのどちらが有効であるかが決定する．また，前述のように広帯域ネットワークの有効利用がシステムの性能向上の重要な要因であることを考慮すると，従来の複製手法とは異なる，広帯域ネットワークの特性を生かした手法を考える必要がある．

本章では，広帯域ネットワークに適した複製手法として，データベース移動を用いて動的に複製を配置してトランザクションを処理する手法を提案する．更に，提案する手法をシミュレーションによって性能比較することで，提案する手法がどのような環境で有効かを検証する．

本章の以下の部分では，まず3.2節で従来の複製手法の問題点について述べる．3.3節でデータベース移動を用いた動的複製配置法を提案し，3.4節で性能評価のために行ったシミュ

レーションの結果を示す．最後に3.5節で本章のまとめとする．

3.2 従来の複製手法

これまでに提案されている複製手法は，ネットワークの帯域幅が狭いことを想定しているために，複製を静的に配置したり，ある程度長い観測期間内のアクセス履歴などの統計情報を用いて複製を再配置するものがほとんどである [54, 73, 78]．これらの手法では，各サイトのトランザクション発生頻度やデータベースのアクセスパタンが頻繁に変化する環境では，効果的な複製配置を行えないために性能が低下する．

この問題を解決するために，小さなデータの複製を作成して動的に配置する手法もいくつか提案されている [4, 72, 84] が，これらの手法では，複製の位置管理のためのオーバヘッドが大きいだけでなく，複製間の一貫性保持やデータベースの制約条件のチェックのために多くのメッセージ通信を必要とし性能が劣化する．本章で提案する手法では，豊富な帯域幅を前提とし，データベース移動によってデータベースという大きなデータ単位の複製をトランザクション毎に再配置することで，従来の手法の問題点を解決する．以下では，これまでに提案されている代表的な動的複製配置法の概要を示し，これらがもつ上記以外の問題点について述べる．

まず，文献 [84] では，データ項目の複製の作成・削除を各サイトで自律的に決定する複製の再配置法を提案している．この手法では，サイト j があるデータ項目に連続して二回の読出し操作を行い，その間に他のサイトからそのデータ項目に対して書込み操作が発生しない場合に，そのデータ項目の複製をサイト j に作成する．逆に，サイト j に存在するデータ項目（複製）に連続して二回の書込み操作があり，その間に読出し操作が発生しない場合に，サイト j の複製を削除する．文献 [84] では，読み書き操作に要するサイト間の通信回数をコストと定義して，システムの読み書き操作のパターンが均一の場合に提案した手法がコストを最適にすることを証明している．また，読み書きの操作系列が既知の場合に，動的に複製の作成・削除戦略を変更してコストを最小にする手法も提案している．しかし，これらの手法には，以下のような問題点がある．

1. 木構造のネットワークトポロジでしか動作しない．

2. 複製の作成に対する記憶領域に制限がない．

3. 更新操作が，複製の作成に関与した二サイト間を伝搬して各サイトに反映されるため，大きな時間がかかる．

4. 読み書き操作のパターンが均一な場合か既知な場合でしか，コストが小さくならない．

5. 通信回数（伝搬遅延）のみを考慮して，データの転送遅延を考慮していないことから，コストが最小になったとしても，システムの性能が最大になるとは限らない．

文献 [4] では，文献 [84] の手法を拡張した動的複製配置手法を提案している．この手法では，記憶領域に制限をもたせ，LRU アルゴリズムに基づいて複製の置き換えを行うことで，上記の問題点（2）を解決している．また，アクセス履歴をとり，読み書き操作のパターンを予測した複製の再配置を行うことで，上記の問題点（4）を解決している．しかし，この手法では，問題点（1）（3）（5）については解決できない．更に，複製の置き換えのアルゴリズムが単純であるため，制限された記憶領域を有効に活用できていない．

文献 [72] では，複製再配置を商業モデルに基づいて決定する手法が提案されている．この手法では，複製の売手と買手の間の契約によって，データの更新内容が通知される最大遅延を保証するが，厳密に複製間の一貫性を保持することはできない．また，複製再配置の具体的な方法は示されていない．

本章で提案する手法では，3.5節で示すように，これら従来の手法の問題点を全て解決できる．

3.3 データベース移動を用いた動的複製配置法

本章では，DB 移動複製法と呼ぶ，データベース移動を用いて複製を動的に配置してトランザクションを処理する手法を提案する．まず，本章で想定するシステム環境について述べ，その後，DB 移動複製法について説明する．

3.3.1 システムの基本動作

本章では，ATM の仮想 LAN などのように，メッセージのブロードキャストが可能な広帯域ネットワーク内での分散データベース処理を想定する．システム全体として一つのデータベースをもち，これを複数のローカル・データベース（以後は，これをデータベースと呼ぶ）に分割して，システム内の各サイトで保持する．なお本章では，第2章と同様に，高

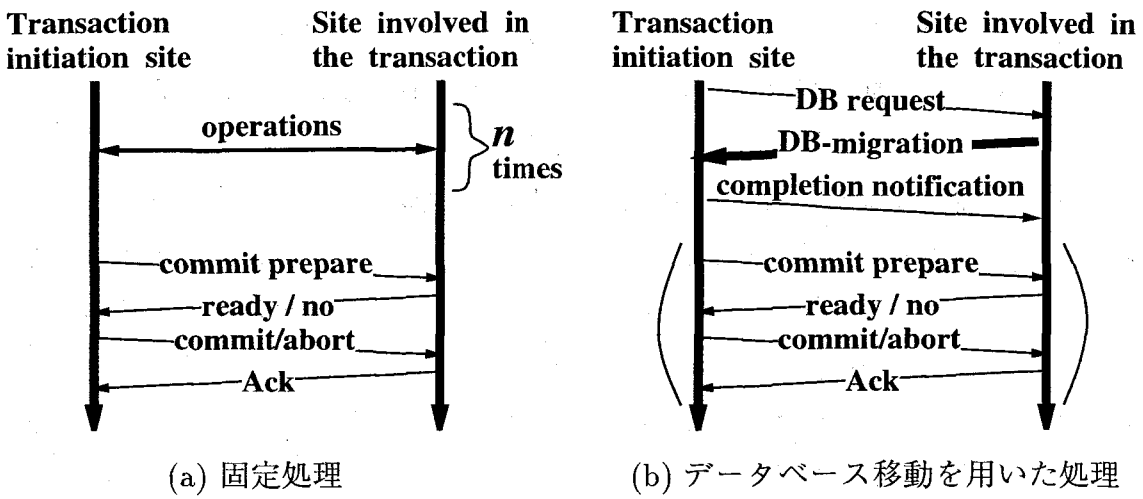
速なデータベース・アクセスとデータベース移動を実現するために，システム内のデータベースは主記憶データベースであるものと想定する．各サイトは，主記憶領域の範囲内で，複製の作成・削除を行うことができる．

データベース移動はデータベースの再配置を行うだけでデータベースの一貫性には影響を与えないため，システムの基本動作として，従来の二相施錠規約と二相コミット規約に基づいてトランザクションの処理を実行する．データベースに対する読出し・書込みの操作は，いずれの複製に対しても行えるが，書込みの場合は複製間の一貫性を保つためにすべての複製に書込み施錠を行わなければならない．但し，データベース移動がデータベースという大きなデータ単位をある時間専有する操作であることから，第2章で提案したデータベース移動を考慮した並行処理制御手法を用いる．この手法では，データベース移動の要求を含めた施錠要求のデッドロックが生じない機構を実現している．トランザクション発生サイトは，トランザクション開始時に施錠要求のメッセージを全サイトにブロードキャストする．このとき，特定のサイトを経由してそのサイトから施錠要求をブロードキャストすることで，そのサイトにおいてトランザクションの発生時刻の絶対順序を定義し，施錠要求の到着順序の逆転によって生じる施錠のデッドロックを防止する．各サイトでは，保持するデータベースのデータ項目毎に施錠要求キューを管理し，施錠の排他関係に基づいてデータベース操作を実行する．

トランザクションの処理手法としては，従来のデータベース固定型の処理（以後，固定処理と呼ぶ）とデータベース移動を用いた処理のいずれかを選択する．両手法の処理手順の概要を次に示す．

固定処理：固定処理における通信手順を図3.1(a)に示す．処理依頼メッセージの転送と結果の返送といった処理操作を行い，これに n 回の通信を要する．最後に，二相コミットのために二往復のメッセージ交換を行う．

データベース移動を用いた処理：データベース移動を用いた処理の通信手順を図3.1(b)に示す．まず，DB要求のためのメッセージをブロードキャストする．DB要求のメッセージをブロードキャストしているのは，全サイトが各データベースの正確な位置情報を保持することを可能にするためである．なお，本節では，トランザクションの開始時に，トランザクション処理に必要なデータベースが同定できるものと仮定している．次に，データベース移動を実行し，移動完了後に移動完了通知をトランザクション発生サイトからブロードキャストする．第2章のDB-MANシステムのトランザクション処理では複製を許さない



(a) 固定処理　　(b) データベース移動を用いた処理

図 3.1: 各手法の通信手順

ために，データベース移動を実行した後に転送元サイトのデータベースは消去していたが，本章で提案する手法ではこれを消去せず，複製としてそのサイトに残す．トランザクション発生サイトにおいてデータベース操作を実行した後，そのトランザクションが読出し操作のみでない場合は，操作対象となったデータベースのいずれかに複製が存在すれば，複製を保持するサイトに更新ログを転送し，固定処理同様に二相コミットを実行する．更新ログは，コミット準備メッセージに付加する．

データベース移動を用いた処理が選択されたとき，複製の削除および作成を実行する．トランザクション発生サイトにおいて，移動してくるデータベースの複製を作成するのに十分な主記憶領域を確保できるまで，そのサイトで保持していた他のデータベースの複製を削除し，その後，データベース移動によって新しい複製を作成する．

3.3.2 DB 移動複製法

DB 移動複製法では，DB-MAN システムのトランザクション処理手法と同様に，従来の固定処理とデータベース移動を用いた処理とを適応的に選択して，トランザクション処理を実行する．手法選択においては，現トランザクションを両手法で実行する際に要するそれぞれの通信所要時間と，トランザクション発生サイトのデータベースのアクセスパタンを考慮する．アクセスパタンを予測するために，データベースのアクセス履歴と，あるサイトが特定のデータベースを連続的に使用する場合に行う連続使用宣言の情報を用いる．

DB-MAN システムの手法では，データベースを現在所持しているサイトとトランザクション発生サイトのどちらにそのデータベースが置かれている方がシステム全体として効果的かという点が，移動の基準になっていた．しかし，移動後に転送元のデータベースが削除されない場合，現在所持しているサイトとの兼ね合いではなく，移動してくるデータベースとその主記憶領域の確保のために削除されるその他のデータベース（複製）のどちらを所持していることが有効であるかという点を基準に，データベース移動の実行を決定する必要がある．そこで，DB 移動複製法では，(1) 領域確保のために削除する複製の候補の決定，(2) 複製作成の有効性を考慮したデータベース移動実行の決定（トランザクション処理手法の選択），(3) 複製の作成・削除，といった DB-MAN システムの手法とは全く異なる仕組みが必要となる．以下では，これらの仕組みについて説明する．なお，DB 移動複製法の実行手順としては，(1) から (3) を順に実行した後，最後に，選択された処理手法に基づいてトランザクションを実行する．

削除するデータベースの候補の決定

まず，データベース移動の実行に必要な領域を確保するために削除しなければならない複製の集合$\boldsymbol{D_O}$の候補を選択する．各サイトでは，削除する候補の決定と手法選択に必要な情報として，次の属性をもつ DB 情報表を保持する．

DB-id: データベース識別子

Size: DB-id で示されるデータベースのサイズ

Current Loc: データベースの現在位置（複製が存在する場合はリストになる）

Cnt: データベースの連続使用宣言がされているか（0: されていない，1: されている）

Usage-log: データベースのアクセス履歴

これらの情報の更新は，すべてブロードキャストによって全サイトに反映される．ただし，メッセージ通信回数が増加しないように，データベース移動の要求メッセージなどその他のメッセージに追加して転送される．

ここで，サイト S_Iがデータベース D_jを保持する有効性を示す目的関数を，第 2章の $f(S_I, D_j)$ をそのまま用いて次のように定義する．

$$f(S_I, D_j) = \alpha \cdot P' \cdot |L| + \sum_{l=1}^{|L|} \{k_l \cdot (|L| + 1 - l)\}. \tag{3.1}$$

$$\alpha = \begin{cases} 1: & S_I \text{が} D_j \text{の連続使用を宣言している，あるいは} \\ & \text{現トランザクションで} D_j \text{の連続使用を宣言する．} \\ 0: & \text{その他．} \end{cases}$$

P': 連続使用優先係数

$|L|$: DB のアクセス履歴のサイズ
（トランザクション L 回分の履歴を残す）

$$k_l = \begin{cases} 1: & l \text{回前のトランザクションで} S_I \text{が} D_j \text{を利用．} \\ 0: & \text{その他．} \end{cases}$$

$f(S_I, D_j)$ は，第一項で D_jを連続的に使用することに対する優先度を表し，第二項で利用頻度に対する優先度を表している．ここで，連続使用優先係数 P'は，連続使用宣言に対しての優先度の度合を表すもので，DB-MAN システムの処理手法選択機構における連続使用優先係数 P と区別して P'としている．P'の値が大きいほど，連続使用が宣言された場合の目的関数 $f(S_I, D_j)$ の値が大きくなる．

各サイトにデータベース領域として割り当てられている主記憶領域のサイズを M，データベース D_jのサイズを $Size(D_j)$，現トランザクションの実行に必要でそのサイトが保持していないデータベースの集合を$\boldsymbol{D_N}$，そのサイトが所持しているデータベースおよび複製の集合を$\boldsymbol{D_H}$，そのサイトが保持している現トランザクションに不必要なデータベースおよび複製の集合を$\boldsymbol{D_C}$とする．$\boldsymbol{D_C}$の中から，次式を満たすようになるまで，下記のアルゴリズムにしたがって$\boldsymbol{D_O}$に加える複製を選択する．

$$M \geq \sum_{D_j \in D_H} Size(D_j) - \sum_{D_j \in D_O} Size(D_j) + \sum_{D_j \in D_N} Size(D_j). \tag{3.2}$$

アルゴリズム：

1. $\boldsymbol{D_O}$を空とする．

2. $\boldsymbol{D_C}$のなかで複製が他のサイトに存在するものから，$f(S_I, D_j)$ の値が小さい順に$\boldsymbol{D_O}$に加える．

3. $\boldsymbol{D_C}$のなかで複製をもたないものから，$f(S_I, D_j)$ の値が小さい順に$\boldsymbol{D_O}$に加える．ここで$\boldsymbol{D_O}$に加えられたデータベースの集合を$\boldsymbol{D_M}$とする．ここで選択されたデータベースは，削除する前に他のサイトに移動する必要がある．

手法選択アルゴリズム

トランザクション処理の実行に必要なデータベースの総サイズが，トランザクション発生サイトの主記憶領域より大きい場合は，強制的に固定処理を実行する．それ以外の場合では，固定処理とデータベース移動を用いた処理の通信所要時間と各サイトのデータベースのアクセスパタンを考慮して処理手法を選択する．

まず，データベース移動を実行する有効性，つまり，トランザクション処理に必要なデータベースの複製を作成することの有効性を表す評価値 E を，次のように定義する．

$$E = \sum_{D_j \in D_N} f(S_I, D_j) - \sum_{D_j \in D_O} f(S_I, D_j). \tag{3.3}$$

そこで，発生したトランザクションを，固定処理とデータベース移動を用いた処理のどちらで処理するかを，次の規則にしたがって選択する．

if $(T_{DB} - T_{fix}) - K' \cdot E < 0$	
then	データベース移動を用いた処理
else	固定処理

ここで，T_{DB}, T_{fix}は，それぞれデータベース移動を用いた処理と固定処理の通信所要時間の見積もり値であり，その算出法としては第2章での見積もりをベースとした次式を用いる．

$$\begin{aligned} T_{fix} &= (n+4) \cdot d + d_b + C \\ T_{DB} &= (1+4q) \cdot d + 2d_b + C + \textstyle\sum_{D_j \in D_N \cup D_M} Size(D_j)/B_M. \end{aligned} \tag{3.4}$$

d は通常の一対一通信に要する伝搬遅延，d_bはメッセージのブロードキャストに要する伝搬遅延，C は一対一通信のためのコネクション設定時間，B_Mはデータベース移動のためのネットワーク帯域幅を表している．q は D_Nの全てのデータベースが複製をもたない場合もしくは D_Nのデータベースに読込み操作しか行わない場合に 0，それ以外の場合は 1 の値をとる．

K'は E の値の優先度を表すもので，この値が大きいと通信所要時間の見積り値よりも履歴情報と連続使用情報を重視する．この K'を履歴依存係数と呼ぶ．また，K'としているのは，第2章の履歴依存係数 K と区別するためである．履歴依存係数 K'および前出の連続

使用優先係数 P'の値はシステムの性能に影響するため，DB 移動複製法を実環境で利用する際は，システムをモデル化し，シミュレーションなどによって詳細に評価した上でこれらの値を決定する必要がある．

複製の作成・削除アルゴリズム

手法選択によってデータベース移動を用いた処理が選択された場合，$\boldsymbol{D_O}$に含まれる複製およびデータベースを，移動してくるデータベースの主記憶領域を確保するために削除する（複製の削除）．ただし，複製が存在しないデータベース $D_m \in \boldsymbol{D_M}$に関しては，次の置き換えアルゴリズムを用いて他サイトに移動する．

1. 自サイトを除く全サイトからなる集合を$\boldsymbol{S_C}$とする．

2. $\boldsymbol{S_C}$に含まれる各サイト S_iに関する $f(S_i, D_m)$ を計算し，その値が最も大きいサイト S_oを D_mの移動先の第一候補とする．そして，S_oに対してデータベースの「受取り要求メッセージ」を転送する．

3. 「受取り要求メッセージ」を受信した S_oは，移動してくるデータベースのための主記憶領域を確保する．現段階で，主記憶領域に余裕がない場合には，そのサイトがもつ複製のうち他の実行中のトランザクションに使用されていないものの中から $f(S_I, D_j)$ が最も小さいものを削除する．主記憶領域が確保できた場合は（5）へ進む．主記憶領域が複製をもたないデータベースや実行中のトランザクションに使用されているデータベースによって占有されている場合には,「受取り拒否メッセージ」を返送して（4）へ進む．

4. 「受取り拒否メッセージ」を受信したトランザクション発生サイトは，S_oを$\boldsymbol{S_C}$から削除する．$\boldsymbol{S_C}$が空でない場合は（2）に戻り，空の場合はトランザクションの処理手法を固定処理に変更する．

5. S_oは，受取り要求を承諾したことを示す「受取り承諾メッセージ」を返送する．このメッセージは，通常のデータベース移動の移動要求メッセージと同様に，各サイトでデータベースの位置情報を保持するために全サイトにブロードキャストする．トランザクション発生サイトは，このメッセージを受信した後にデータベース移動を実行する．移動完了後は，通常のデータベース移動と同様に移動完了通知を行う．

以上のような処理が完了した後，あるいは，この処理に並行して，トランザクション処理の実行に必要なデータベース$\boldsymbol{D}_N$の複製を，他サイトからのデータベース移動によって作成する（複製の作成）.

なお，$\boldsymbol{D}_M$に含まれるデータベースの受取り先サイトが決定するまでに要するメッセージ交換の時間は，手法選択時のT_{DB}に加えることはできない．これは，実行中のトランザクションによってデータベースの受取りが可能かどうかが決定するため，手法選択時に受取り先サイトを同定できないからである.

3.4 シミュレーションによる性能評価

本節では，本章で提案したDB移動複製法の性能評価のために行ったシミュレーションの結果を示す．シミュレーションでは，DB移動複製法と，従来の静的な複製法，第2章のDB-MANシステムにおけるトランザクション処理手法の平均通信所要時間を比較する．計算機の処理能力が急速に発展していることを考慮すると，これからの分散処理の処理時間は通信所要時間に大きく影響を受けることから，本節での評価は平均処理時間の評価であるともみなせる．DB-MANシステムの並行処理制御手法を用いることで，データベース移動を導入することで生じる並行処理性能の低下を防ぐことができるため，ここではトランザクションは同時に発生しないものとする．従って，並行処理制御がシミュレーション結果に与える影響は，施錠要求のブロードキャストの通信時間のみとなる．比較対象とする静的な複製法とDB-MANシステムの手法の概要を示す.

静的複製法：主記憶領域の許す限り，各データベースの複製をランダムに配置する．ただし，配置は静的なもので，再配置や新たな複製の作成，削除は行わない．本章の手法と同様に，データベースに対する読出し，書込みの操作は，いずれの複製に対しても行えるが，書込みの場合は複製間の一貫性を保つためにすべての複製に書込み施錠をかけなければならない．トランザクションは，固定処理のみで実行される.

DB-MANシステムの手法：DB-MANシステムの履歴統計モードの手法を用いて，固定処理とデータベース移動を用いた処理を選択してトランザクションを実行する．データベース移動を用いた手法を選択した場合に，移動してくるデータベースに対して十分な主記憶領域が残っていない場合は，十分な領域が確保できるまで，DB移動複製法と同様の方法で，$f(S_I, D_j)$の値の小さい順に選択したデータベースを他サイトに移動する.

ここで，従来の動的複製配置法を比較対象としないのは，これらの手法がネットワーク

表 3.1: シミュレーションのためのパラメータの設定

パラメータ	パラメータの意味	値
$\|S\|$	仮想 LAN 内のサイト数	20（サイト識別子 $S_1, \cdots, S_{20}$）
$\|D\|$	データベースの総数	20（DB-id $D_1, \cdots, D_{20}$）
$Size(D_j)$	各データベースのサイズ	80 [Mbytes]
p_1	トランザクション発生率 1	0.025 ($S_1, ..., S_{10}$), 0.05 ($S_{11}, ..., S_{18}$), 0.15 (S_{19}), 0.3 (S_{20})
p_2	トランザクション発生率 2	0.025 ($S_{11}, ..., S_{20}$), 0.05 ($S_3, ..., S_{10}$), 0.15 (S_2), 0.3 (S_1)
d_{MCS}	MCS への平均伝搬遅延	0.12 [秒] ($d_b = 2d_{MCS}$)
d	その他の 2 サイト間の伝搬遅延	0.12 [秒]
C	コネクション設定時間	0.3 [秒]
n	固定処理における通信回数	$n'\|\boldsymbol{D}_N\|/\|\boldsymbol{D}_U\|$（$n'$は 1〜30 の範囲でランダムに決定する）
$\|L\|$	利用履歴のサイズ	20
P, P'	連続使用優先係数	3.50 (P), 1.40 (P')
K, K'	履歴依存係数	0.02 (K), 0.035 (K')

トポロジを制限していたり，複製間の一貫性を厳密に保持していないことから，同じ条件で評価できないためである．

3.4.1 シミュレーション環境

シミュレーションでは，ATM の仮想 LAN 内に構築された分散データベースシステムを想定する．ATM ネットワークでは，あるサイトから複数のサイトへあらかじめポイント・ツー・マルチポイントコネクションの PVC を設定しておくことにより，そのサイトをマルチキャストサーバ（MCS）として特定のグループへのマルチキャストが可能となる．つまり，物理的な接続形態とは異なる仮想的な LAN の構築が可能となり，仮想 LAN 内のデータのブロードキャストは，MCS に対してデータを送信することで実現できる．本研究のよ

うなデータベース移動を用いるシステムでは，バースト転送とメッセージのブロードキャストをサポートできるATMの仮想LANが最も適していると考えられる．ブロードキャスト以外の通信に関しては，すべてポイント・ツー・ポイントのSVCコネクションを用いる．また，一度確立されたSVCコネクションは一つのトランザクション内で再設定の必要がないものとする．

シミュレーションで用いるパラメータの値を表3.1に示す．データベースD_iの初期位置をS_iとする$(1 \leq i \leq 20)$．複製が存在する手法では，複製の初期位置として，主記憶領域が許す限り複製をランダムに配置する．データ転送に関するパラメータは，近い将来に実現されると思われる全世界規模の分散システムを想定したものである．つまり伝搬遅延は，光ファイバケーブルを用いたATMネットワークで各サイト間の平均ホップ数（経由する交換機数）が10程度，各交換機でのルーチング遅延が市販のATM交換機同様に5ミリ秒程度と仮定して設定している．また，データベースのサイズは，アプリケーションの性質などに依存し，この値の設定がデータベース移動を用いた処理の通信所要時間に大きく影響するが，今回は，最も顕著な例として，帯域幅が1.5Gbpsのときに固定処理の通信所要時間とデータベース移動を用いた処理の通信所要時間がほぼ等しくなるように設定した．データベースに対する操作回数n'を1〜30でランダムに変化させることで，さまざまな複雑度のトランザクションを表現できる．更に，1000回毎に各サイトのトランザクション発生確率をp_1, p_2と交互に変化させることで，提案した手法の環境の変化に対する適応性を検証する．システムや通信の障害は発生しないものと仮定し，各手法の通信所要時間には，T_{fix}とT_{DB}の計算値をそのまま用いる．但し，DB移動複製法とDB-MANシステムの手法において，領域確保のためのデータベース移動と移動先のサイトを見つけるのに要する通信時間を，各手法の通信所要時間に加えることにする．

DB-MANシステムの履歴統計モードにおける連続使用優先係数P，履歴依存係数K，および，DB移動複製法におけるP', K'の値としては，本節のシミュレーション条件をベースに帯域幅を1.5Gbps，主記憶領域のサイズを10（データベースサイズの整数倍）とし，P, K, P', K'の値を変化させて各手法の通信所要時間を調べることで得た最適値を用いている．

シミュレーションでは，次の手順を繰り返し実行する．

1．トランザクション発生率に応じてトランザクションの発生サイトS_Iを決定する．

2．トランザクションに必要なデータベースの総数$|\boldsymbol{D}_U|$を1から5の範囲でランダムに

決定する．もし，トランザクション発生サイト S_Iがあるデータベースに対して連続使用宣言をしている場合，$|\boldsymbol{D}_U|$ の値が連続使用宣言をしているデータベースの総数よりも小さければ，$|\boldsymbol{D}_U|$ をその値に再設定する．

3．トランザクションに必要なデータベースとして $|\boldsymbol{D}_U|$ 個のデータベースを選択する．但し，連続使用宣言をしているデータベースは必ずこれに含まれなければならない．

4．操作回数 n'を決定し，$n'|\boldsymbol{D}_N|/|\boldsymbol{D}_U|$ をデータベースのうちトランザクション発生サイトが所持していないデータベースに対する操作の回数とする．

5．トランザクションの連続度βを 0,1,2 の中からランダムに決定する．この値は，S_Iが以後何回連続してトランザクションを発生するかを表している．もし，この値が 0 より大きい場合は，以後β回，S_Iのトランザクション発生率を 1.00 増加させる．

6．連続使用するデータベースを 3 で決定したデータベースの中からランダムに決定する．

7．DB-MAN システムの手法および DB 移動複製法のみ，それぞれの選択法に従って，固定処理若しくはデータベース移動を用いた処理を選択する．

8．トランザクション処理を実行する．なお，サイト間のデータの伝播遅延，MCS へのデータの伝播遅延は，それぞれ d, d_{MCS}を固定的に用いる．

3.4.2 シミュレーション結果

前節で示した環境に基づき，まず読出し操作のみのトランザクションは発生しないものとして，データベース移動のための帯域幅と各サイトの主記憶領域のサイズを変化させて，各手法の平均通信所要時間を計算した．その結果を図 3.2，図 3.3，図 3.4に示す．グラフでは，x 軸がデータベース移動のための帯域幅，y 軸が各サイトの主記憶領域のサイズ（データベースサイズの整数倍）をそれぞれ表している．この結果から，従来の静的な複製法は，ネットワークの帯域幅の影響は受けないが，主記憶領域のサイズに大きく影響を受けることがわかる．文献 DB-MAN の履歴統計モードは，ネットワーク帯域幅によって性能が大きく変化し，また，主記憶領域のサイズが 5 以下になると性能が劣化する．一方，本章で提案した DB 移動複製法は，他の二手法ほどではないが，帯域幅と主記憶領域のサイズの両方の影響を受けている．

各手法の優劣を明確にするために，上記の結果に基づき，各ネットワーク帯域幅，主記憶領域のサイズにおいて最適（通信所要時間が最短）となる手法を調べた．その結果を図 3.5(a) に示す．更に，全トランザクション中の読出し操作のみのトランザクションの割合

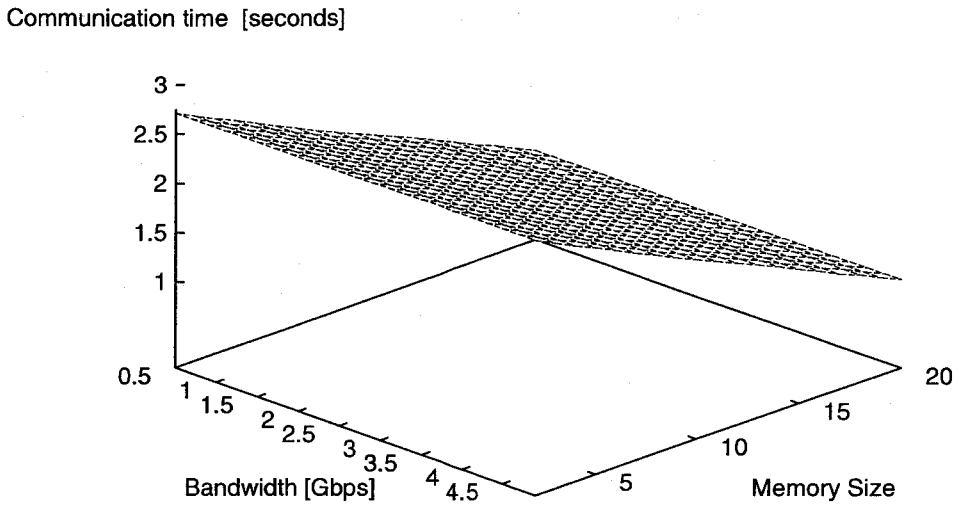


図 3.2: 従来の静的複製法の平均通信所要時間

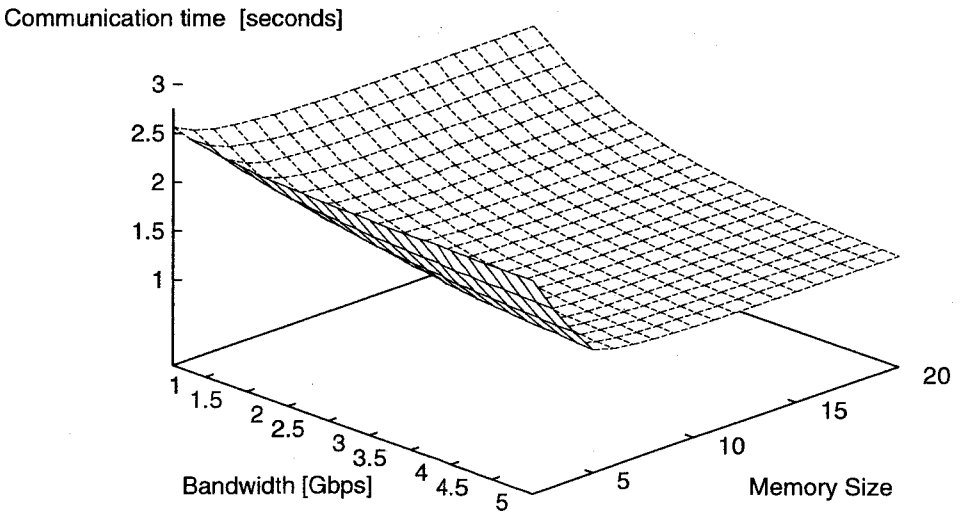


図 3.3: DB-MAN システムの履歴統計モードの平均通信所要時間

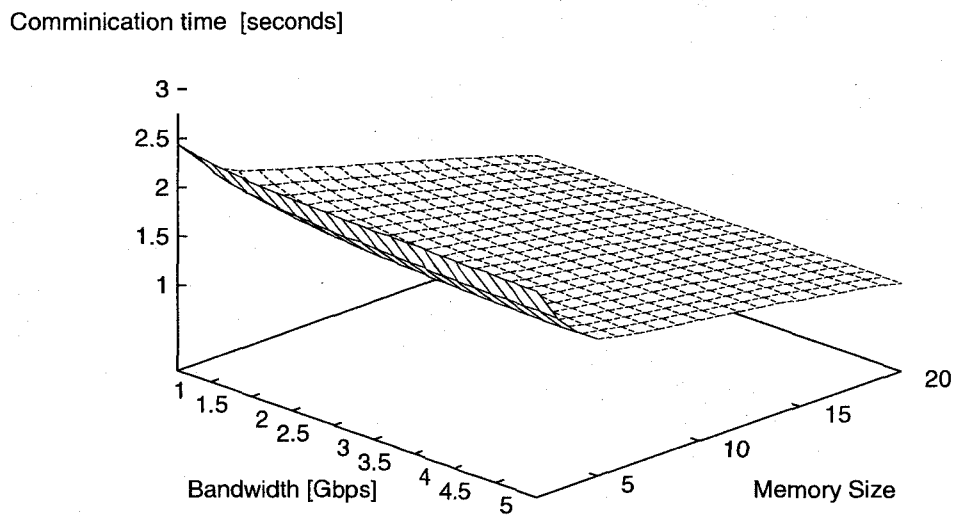
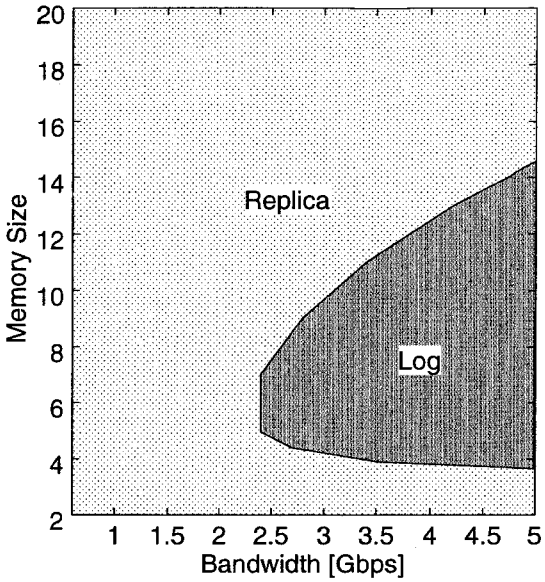


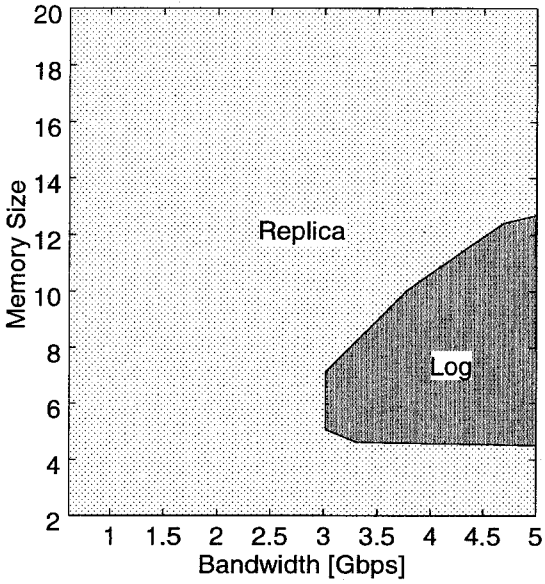
図 3.4: DB 移動複製法の平均通信所要時間

を 20%と 40%として，同様に最適となる手法を調べた．その結果をそれぞれ図 3.5(b)，図 3.5(c) に示す．図では横軸はデータベース移動のための帯域幅，縦軸は各サイトの主記憶領域のサイズ，図中の各領域は各手法が最適となる範囲を表している．また，'Log' は DB-MAN システムの履歴統計モード，'Replica' は DB 移動複製法をそれぞれ表している．この結果から，まず読出し操作のみのトランザクションが発生しない環境では，主記憶領域のサイズが 4 以上の範囲では，ネットワークの帯域幅が大きく主記憶領域のサイズが小さい場合に，DB-MAN システムの履歴統計モードが最も良い性能を示し，それ以外の場合では DB 移動複製法が最適となることがわかる．また，主記憶領域のサイズが 5 以下の範囲では，DB 移動複製法が良い性能を示している．なお，読出し操作のみのトランザクションの割合が大きくなると，結果の性質はほぼ同じであるが，DB-MAN システムの履歴統計モードが最適となる範囲が小さくなる．このような結果になる原因としては，次の 4 点が考えられる．

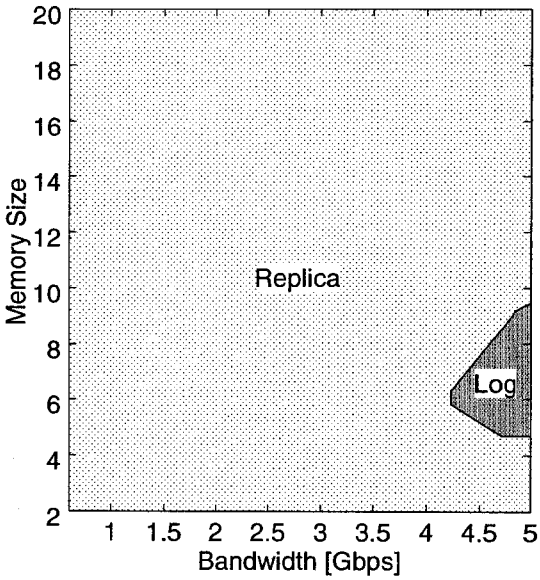
1. 複製を用いる手法では，主記憶領域のサイズが大きくなるとより多くの複製を保持できるため，トランザクション処理をほぼローカルに実行でき性能が向上する．



(a) read-only (0%)



(b) read-only (20%)



(c) read-only (40%)

図 3.5: 最適手法の分布

2. 読出し操作のみのトランザクションでは複製間の一貫性保持のための 2 相コミットの必要がないため，その割合が大きくなると，複製を用いる手法が最適となる範囲が大きくなる．

3. ネットワークの帯域幅が大きくなると，データベース移動のための転送遅延よりもメッセージの通信回数の方が性能に大きく影響するようになるため，複製を用いる手法における複製間の一貫性保持のための 2 相コミットが性能を低下させる．

4. 主記憶領域がかなり小さいとき，履歴統計モードでは，トランザクションが連続的に発生してこれをデータベース移動を用いた処理で実行する場合に，領域確保のためのデータベース移動が頻繁に発生するため性能が低下する．一方，DB 移動複製法では，先のトランザクションで移動したデータベースの複製が移動元に残されているため，次のトランザクションで異なるデータベースを移動してくる場合に，先のトランザクションで作成した複製を削除することで領域が確保でき，領域確保のためのデータベース移動は頻繁には起こらない．

以上の結果より，与えられたシミュレーション環境において広範囲の主記憶領域とネットワーク帯域幅で，提案した DB 移動複製法の優位性が確認できた．実環境においても，実際のシステムをモデル化し，このようにシミュレーションを行うことで，DB 移動複製法と DB-MAN システムの履歴統計モードで用いている手法の有効領域の境界値が分かり，両手法を使い分けることが可能となる．

3.5 むすび

本章では，データベース移動を用いて動的にデータベースの複製を再配置する手法を提案した．データベースという大きなデータ単位を動的に再配置することによって，従来の複製手法がもつさまざまな問題点を解決できる．本章で提案した手法は，従来の動的複製配置手法とは異なり，ネットワークトポロジに特に制限はなく，また，更新操作をブロードキャストすることによって全ての複製に更新内容が短時間で反映される．更に，操作のための伝搬遅延だけでなく，複製の作成に要する転送遅延も考慮して複製の再配置を行う．複製の置き換えに関しても，LRU より複雑な方法を用いることで，制限された記憶領域をより有効に活用できる．

シミュレーション評価の結果から，提案したDB移動複製法が広範囲にわたって良い性能を示すことが確認できた．実際の分散システムにおいては，ネットワークの帯域幅や各サイトの主記憶領域のサイズなどといったシステムの特性に応じて，DB移動複製法とDB-MANシステムで用いている手法を使い分けることが有効である．

第4章

データベースの位置管理

4.1 まえがき

本研究のようにシステム内のデータベースの移動を分散処理に利用する環境では，データベースの位置管理が必要となる．第2章と第3章では，ATMの仮想LANなどのようなメッセージのブロードキャストが可能な環境を想定しているため，データベースの移動時にシステム内の全サイトに新たな位置情報をブロードキャストするといった方法で位置管理を行っている．しかし，提案したDB-MANシステムや動的複製配置法をATMのWAN環境などに適応することを考えた場合，メッセージのブロードキャストができないため，異なる位置管理の手法が必要になる．

モーバイルコンピューティングの分野では，移動体の位置管理が重要な問題の一つと考えられ，これまでに多くの手法が提案されている [17],[49],[50],[52],[53], [62],[63],[64], [70],[76],[77], [79],[83]．また，分散システム内で負荷分散のためにデータ項目の再配置をする研究においても，データ項目の位置管理の手法が提案されている [81]．これらの位置管理手法は，データベースの位置管理にも応用できるものと考えられる．そこで本章では，まず，これまでに提案されているこれらの手法をベースとするデータベースの位置管理手法を提案する [28],[29],[30]．その後，ATMネットワークの特徴であるSVCコネクションの一定時間の定常性と豊富な帯域幅を有効に利用した新たな手法を提案し，従来の手法をベースとしたものとの比較をシミュレーションによって行う．そして，この結果を用いて，さまざまなデータベースへのアクセス頻度やデータベースの移動頻度などの環境においていずれの手法が最適であるかを同定することで，ATMのWAN環境におけるデータベースの位置管

理を効率的に行うことができる．

4.2 移動体の位置管理手法の応用

文献 [53, 79] では，移動体計算環境における移動体の位置管理手法として，メッセージのブロードキャストを用いた手法とデフォルトルータと呼ばれるルータに移動体の位置情報を保持させる手法を，合わせて五つ提案している．ここでは，ブロードキャストが不可能な ATM の WAN 環境を想定しているので，後者を応用した二手法を提案する．また，本節では，デフォルトルータをデータベースの位置管理のためのサーバ（位置管理サーバ）に置き換えて，これらの手法をデータベースの位置管理手法に変更する．但し，文献 [53, 79] では，各々の移動体に対してデフォルトサーバが設置されるが，本節では，すべてのデータベースの位置情報を一つの位置管理サーバで保持しているものとする．データベースの位置管理用に変更した手法の概要を以下に示す．

デフォルトフォワーディング（DF）方式： データベースの移動情報は，位置管理サーバのみに伝えられる．つまり，分散システム内の各データベースの位置情報は位置管理サーバのみが把握している．各サイトが特定のデータベースにアクセスする場合は，処理依頼メッセージを位置管理サーバに転送し，位置管理サーバがそのメッセージを目的のデータベースが存在するサイトへと再転送する．

デフォルト問合せ（DQ）方式： DF 同様に位置管理サーバのみが各データベースの位置情報を把握している．各サイトが特定のデータベースにアクセスする場合は，まず，位置管理サーバにそのデータベースが存在するサイトを問い合わせ，その結果を用いて，処理依頼メッセージを目的のデータベースが存在するサイトに直接転送する．

4.3 移動追従型の位置管理手法の応用

文献 [53, 79] 以外にも，移動体やデータ資源の位置管理の手法がいくつか提案されている．ここでは，文献 [49] で移動体の位置管理のために提案されているポインタ (*Pointer*) 方式を応用した**連鎖フォワーディング（CF）方式**と，文献 [81] で分散システム内のデータ項目の位置管理に用いられている手法を応用した**連鎖問合せ（CQ）方式**を提案する．こ

れらの手法は，処理依頼メッセージがデータベースの移動の軌跡をたどってデータベースの現在位置まで転送される．

CF, CQ の両方式では，まず全サイトに，システム内の各データベースの初期位置を記した位置情報表（以下では LL (Local Location) 表と呼ぶ）を持たせる．以後，LL 表は，各サイトにおいてローカルに更新されていく．その更新の規則として次のものを用いる．

更新規則：

(i) データベース D_lが，サイト S_iからサイト S_jに移動する場合，両サイトの LL 表の D_lの現在位置を S_jに書き換える．

(ii) D_lにアクセスするために，LL 表に従ってサイト S_iにメッセージを転送したが，既に D_lがサイト S_jに移動していることが判明した場合は，D_lの現在位置を S_jに書き換える．

従って，各サイトの LL 表は必ずしも正確な位置情報を把握しているわけではない．従って，CF 方式ならびに CQ 方式では次のようなデータベースアクセス方法を用いる．

CF 方式： 図 4.1のように，まず，自サイトの LL 表の情報に従って D_lが存在すると思われるサイトに処理依頼メッセージを転送する．これを受信したサイトでは，既に D_lが存在しない場合，送られてきたメッセージを LL 表に従って再転送する．このように，各サイトの LL 表に基づくメッセージ転送が連鎖的に起こった後，最終的に D_lが存在するサイトで処理依頼メッセージに相当する処理が D_lに対して行われ，その処理結果と D_lの現在位置が問合せ発生サイトに返送される．

CQ 方式： 図 4.2のように，まず，自サイトの LL 表に従って処理依頼メッセージをあるサイトに転送する．受信したサイトでは，既に D_lが移動している場合は，そのサイトの LL 表の D_lの位置情報を問合せ発生サイトへと返送する．問合せ発生サイトでは，返送された情報をもとに自サイトの LL 表の D_lの位置情報を更新し，更にその新たな位置情報を用いて処理依頼メッセージを再転送する．これを D_lが存在するサイトが見つかるまで繰り返し，最終的な処理結果が問合せ発生サイトに返送される．

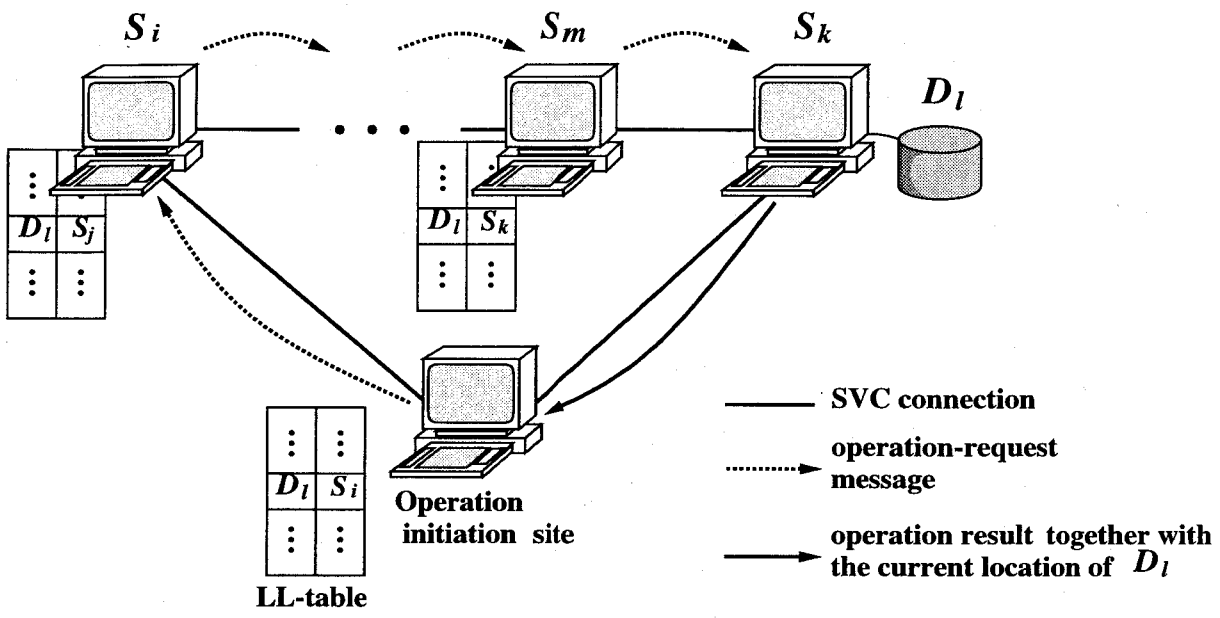


図 4.1: CF 方式でのメッセージの流れ

4.4 ATM ネットワークの特性を考慮した手法

定性的にみると，CF，CQ 方式では，データベースの移動が頻繁な場合に，特定のデータベースへのアクセスにデータベースの移動軌跡に沿った多くのメッセージ通信が必要になり，性能が劣化する．

そこで，この短所を補うことを目的として，新たなデータベース位置管理手法を提案する．提案する手法は，**拡張 CF（ECF）方式**と**拡張 CQ（ECQ）方式**の二つであり，これらは，ATM ネットワークの特徴である次の二点を利用している．

1. 広い帯域幅により，小量のデータの転送遅延は無視できる．
2. 輻輳が発生した場合を除いて，SVC コネクションは一定時間不使用にならない限り解放されない．

ECF，ECQ 方式は，データベースアクセスの際の処理依頼メッセージの流れは CF，CQ 方式と同じである．但し，CF，CQ 方式と異なり，LL 表で保持する情報として時刻印が加わる．LL 表の内容は，処理依頼メッセージと一緒に問合せ発生サイトから転送される．但し，ここでの時刻印は実際の時刻が記されているものではなく，データベースのメタデー

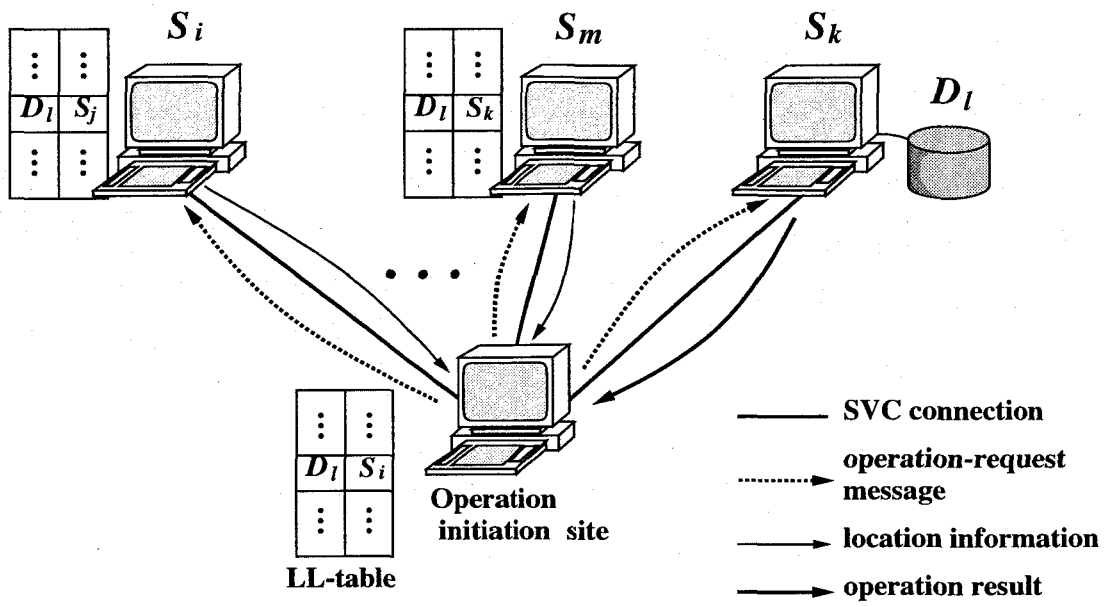


図 4.2: CQ 方式でのメッセージの流れ

タとして管理しているデータベースの移動回数番号である．移動回数番号は各データベース固有のメタデータで，そのデータベースが移動する度に一つずつ値が増加される．つまり，この値と LL 表の位置情報から，その位置が何回目の移動によるものかが判別可能なため，移動回数番号を比較することで位置情報の新しさ（信頼性）を比較できる．移動回数番号を時刻印として用いることにより，実際の時刻を用いる場合に生じる全サイトの時刻の同期をとらなければならないといった問題を解消できる．

ここで，LL 表の内容はさほど大きなデータではないため，広帯域ネットワークを用いることによりこの転送遅延は無視できる．ECF，ECQ 方式では次のような LL 表の更新規則を用いる．

更新規則：

(i) データベース D_lが，サイト S_iからサイト S_jに移動する場合，両サイトの LL 表の D_lの現在位置を S_jに書き換え，時刻印をそのデータベースの（値増加後の）移動回数番号にする．

(ii) LL表の内容を含んだメッセージを受信したサイトは，送られてきたLL表の内容と自サイトのLL表の時刻印を各データベースに関して比較する．両者の時刻印が同じでなければ，古い情報の方を新しいものに更新する．

ECF，ECQ方式では，処理依頼メッセージと一緒に問合せ発生サイトのLL表の内容も転送される．そして，目的のデータベースが存在するサイトおよびそのサイトまでに経由する各サイトにおいて，上述の(ii)の更新が行われる．処理依頼メッセージの転送完了時には，送られてきたLL表の内容は，問合せ発生サイト，経由したサイト，データベースが存在するサイトのすべてから得られる最も新しい情報となる．

その後，メッセージが転送されてきたSVCコネクションを用いて，逆向きに最新のLL表情報を，経由した全サイトおよび問合せ発生サイトに転送していく．この過程は，新たなコネクション設定を必要としないことから短時間で行える．最新の情報を受けとった各サイトは，自サイトのLL表を更新する．

これにより，データベースアクセス時に経由しなければならないサイトの数が減少することが期待される．ECF方式でのメッセージの流れの概要を図4.3に示す．

4.5 通信所要時間の比較

広帯域ネットワークでは，各手法の性能をトラフィック量よりむしろ，通信所要時間の方が重要と考えられる．従って，ここでは，このような考えに基づき，各手法の通信所要時間を評価する．

4.5.1 メッセージ通信に関するパラメータの定義

図4.1のように，通信を行う両サイト間のホップ数（経由するATM交換機の数）をhとし，両サイトでの処理遅延を定数p，ATM交換機間およびATM交換機-サイト間の伝搬遅延を定数d，各ATM交換機でのルーチング時間を定数rとすると，ATMネットワークにおけるコネクション確立後のメッセージ通信時間は$T(h)$は，式(4.1)のように表される．但し，本節では，位置管理やコネクション設定のための制御メッセージとデータベースに対する処理依頼メッセージなどといった比較的小さいメッセージの通信しか行わないため，メッセージの転送にかかる時間（転送遅延）は，広帯域ネットワークを用いることにより0に近似している．

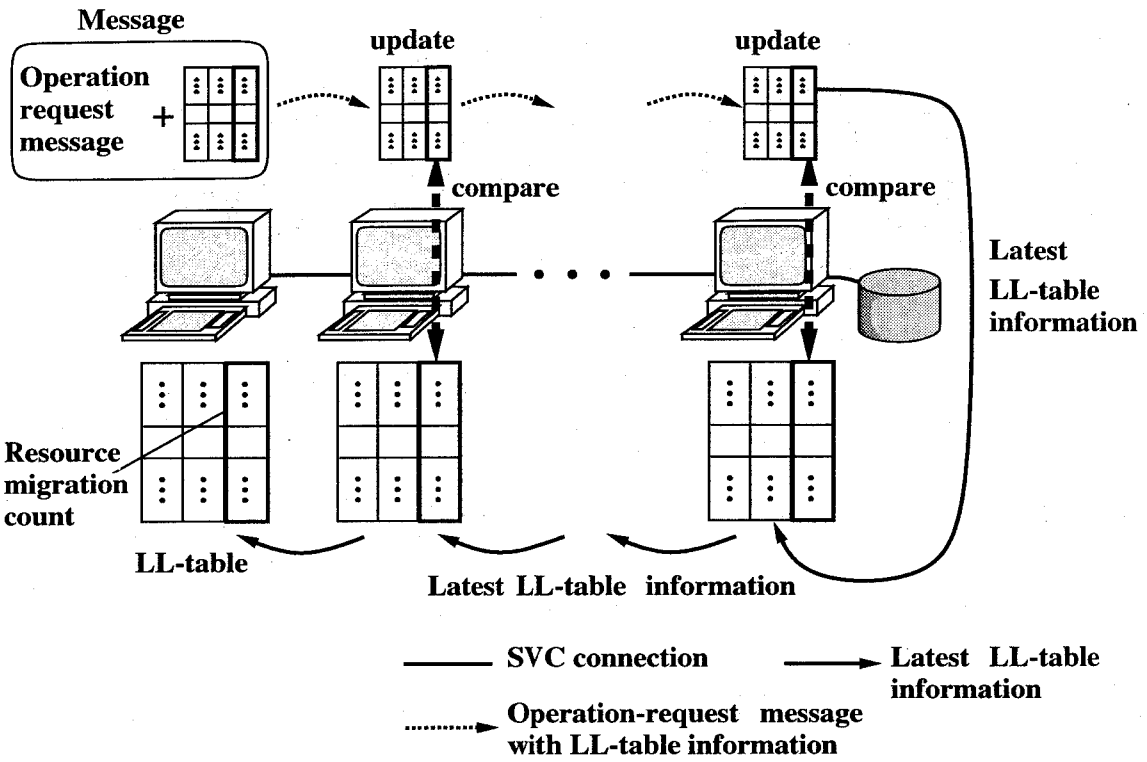


図 4.3: ECF 方式でのメッセージの流れ

$$T(h) = p + (h+1)d + hr \tag{4.1}$$

次に，二サイト間で SVC コネクションを設定する場合を考える．まず，コネクション要求サイトから目的のサイトまでコネクション要求シグナルが経路を設定しながら伝わっていく．そして，目的のサイトまでたどりつくと，そのサイトがコネクションの接続を許可するかを決定し，許可する場合は接続完了通知が往路で設定した経路をたどってコネクション要求サイトまで伝わっていく．従って，往路における各 ATM 交換機での平均経路設定時間を R，往路，復路を合わせた両サイトでの処理遅延を p'とすると，ホップ数 h の両サイト間での SVC コネクションの設定に要する時間 $C(h)$ は，次のように表される．

$$C(h) = p' + 2(h+1)d + h(r+R) \tag{4.2}$$

式 (4.1) と式 (4.2) を用いて，例えば，ホップ数 h の両サイト間で SVC コネクションを

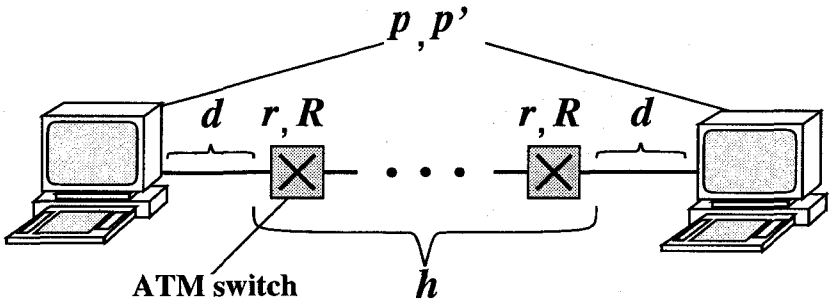


図 4.4: メッセージ通信における各パラメータ

設定して一つのメッセージの送信を行うための所要時間は，$C(h)+T(h)$ と表すことができる．

4.5.2 シミュレーション環境

シミュレーションでは，図4.5のような完全二分木のトポロジをもつ ATM ネットワークを想定する．その他のトポロジをもつネットワークに関しても，以下と同様の方法で性能評価が行える．完全二分木のネットワークでは，木の深さが n，つまり ATM 交換機が n 段の場合，ネットワーク内のサイト数は 2^n となる．ここで，図のように各サイトを $S_1, \cdots, S_{2^n}$ とする．また，DF, DQ 方式では，位置管理サーバが必要であるため，これらの方式では二分木の根の位置の交換機に位置管理サーバ S_0 を接続する．但し，位置管理サーバ自体はデータベースに対する処理を発生しない．

シミュレーションでは，各手法において，あるサイトが特定のデータベースにアクセスするために処理依頼メッセージを転送し，それに対する返答メッセージを受信するまでの一連の操作の通信所要時間を計算する．ここで，同じデータベースに対するそのサイトからの処理が連続することも考慮して，返答メッセージは，目的のデータベースが存在するサイトと処理発生サイトとの間に確立された SVC コネクションを用いて転送されるものとする．

各手法の所要時間を見積もるために，次のような記法を用いる．但し，$0 \leq i, j \leq 2^n$ とする．また，S_I は問合せ発生サイト，S_T は処理の対象となるデータベースが存在するサイトをそれぞれ示している．

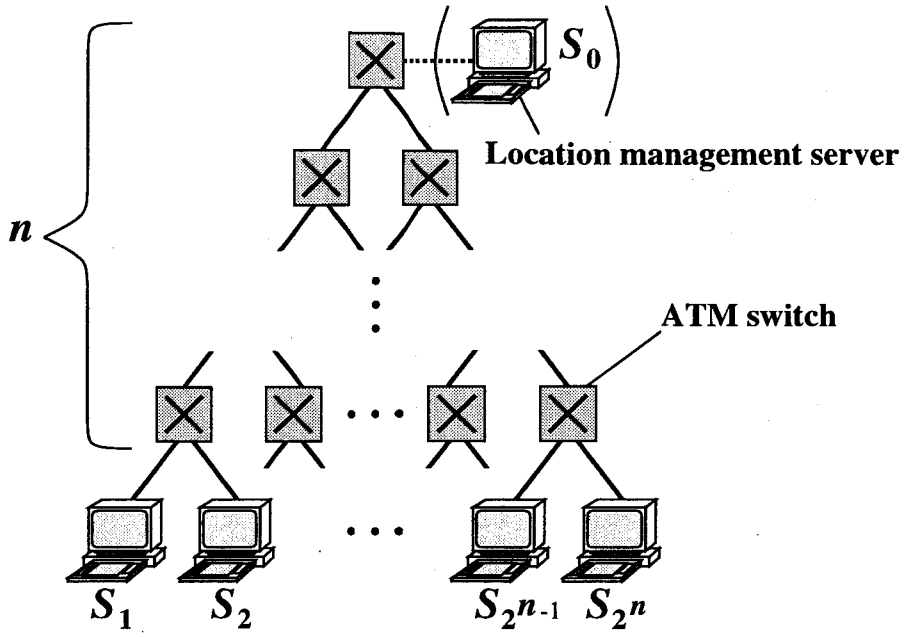


図 4.5: シミュレーションのネットワークトポロジ

$H_{i,j}$ サイト S_i-S_j間の最小ホップ数

$E_{i,j}$ 0 サイト S_i-S_j間の SVC コネクションが設定されている

1 設定されていない

$\boldsymbol{L}$ CF，CQ，ECF，ECQ 方式において，サイト S_Tにたどりつくまでに経由しなければならないサイトのリスト $\boldsymbol{L} = \{S_I, \cdots, S_T\}$

$|\boldsymbol{L}|$ $\boldsymbol{L}$ の要素数

L_i $\boldsymbol{L}$ の i 番目の要素のサイト番号（S_jなら j）

このような環境において，各手法の所要時間は，式 (4.1),(4.2) を用いてそれぞれ次のように表される．ここで，データベースの移動もデータベースに対する処理の一つと考えられるため，処理がデータベースの移動の場合は，DF，DQ 方式において移動後の位置管理サーバへの位置情報の通知のコスト（位置情報の通知と Ack メッセージの返送）も考慮している．

DF 方式:

$$t_{DF} = \begin{cases} 2T(n) + T(H_{I,T}) + (E_{I,0} + E_{0,T})C(n) + E_{I,T}C(H_{I,T}) \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(処理がデータベース移動の場合)} \\ 4T(n) + T(H_{I,T}) + (E_{I,0} + E_{0,T})C(n) + E_{I,T}C(H_{I,T}) \qquad \text{(その他)} \end{cases}$$

DQ 方式:

$$t_{DQ} = \begin{cases} 2T(n) + E_{I,0}C(n) + 2T(H_{I,T}) + E_{I,T}C(H_{I,T}) \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(処理がデータベース移動の場合)} \\ 4T(n) + E_{I,0}C(n) + 2T(H_{I,T}) + E_{I,T}C(H_{I,T}) \qquad \text{(その他)} \end{cases}$$

CF 方式:

$$t_{CF} = \sum_{k=1}^{|L|-1} \left\{T(H_{L_k,L_{k+1}}) + E_{L_k,L_{k+1}}C(H_{L_k,L_{k+1}})\right\} + T(H_{I,T}) + E_{I,T}C(H_{I,T})$$

CQ 方式:

$$t_{CQ} = \sum_{k=2}^{|L|} \{2T(H_{I,L_k}) + E_{I,L_k}C(H_{I,L_k})\}$$

ECF 方式:

$$t_{ECF} = \sum_{k=1}^{|L|-1} \left\{T(H_{L_k,L_{k+1}}) + E_{L_k,L_{k+1}}C(H_{L_k,L_{k+1}})\right\} + \max\left(T(H_{I,T}) + E_{I,T}C(H_{I,T}), \sum_{k=1}^{|L|-1} T(H_{L_k,L_{k+1}})\right)$$

ECQ 方式:

$$t_{ECQ} = \sum_{k=2}^{|L|} \{2T(H_{I,L_k}) + E_{I,L_k}C(H_{I,L_k})\} + \sum_{k=2}^{|L|-1} 2T(H_{I,L_k})$$

4.5.3 シミュレーション方法と結果

シミュレーションで用いるパラメータを表4.1のように設定する．分散システム内のデータベースの総数を 20 $(D_1, \cdots, D_{20})$ とし，D_iは初期状態としてサイト S_iに位置するものとする．簡単のため，各サイトは一様の確率で問合せを発生し，処理の対象となるデータベースも $D_1, \cdots, D_{20}$から同確率で選ばれるものとする．問合せおよびデータベース移動の発生

表 4.1: パラメータの設定

パラメータ	値 [秒]
p	0.01
p'	0.03
r	0.002
R	0.01
d	0.005

間隔は指数分布に基づくものとする．また，SVC コネクションは 20 分間不使用になると解放されるものとする．

まず，$n = 5$（サイト数は 32）の場合を考える．シミュレーションでは，問合せの発生間隔と，問合せとデータベース移動の発生頻度の比（問合せ/移動比）を変化させ，5000 回の問合せを処理し，その平均通信所要時間を各手法で計算した．そのシミュレーション結果を図 4.6～図 4.11に示す．図のグラフは，x 軸が平均問合せ発生間隔，y 軸が問合せ/移動比，z 軸は通信所要時間の平均値を表している．この結果より，DF 方式と DQ 方式は，問合せ/移動比の影響をあまり受けていないのに対し，他の四手法では，問合せ/移動比が小さくなる，つまり，データベースの移動頻度が高い場合に通信所要時間が大きくなることが分かる．特に CF，CQ 方式は，問合せ/移動比によって顕著に結果が変化する．

次に，図 4.6～図 4.11の結果から，各問合せ間隔，問合せ/移動比において最適（通信所要時間が最短）となる手法を調べた．その結果を図 4.12(a) に示す．図では，x 軸が平均問合せ間隔，y 軸が問合せ/移動比，図中の各領域が各手法が最適となる範囲を表している．また，$n = 7, n = 10$ の場合にも同様のシミュレーションを行い，その結果に基づいて最適手法を調べた．最適手法の分布を，$n = 7, n = 10$ の場合で，それぞれ図 4.12(b), (c) に示す．

$n = 5$ では，すべての場合で ECF 方式が最適となる．$n = 7$ では，問合せ/移動比がおよそ 5.5 を越える場合，つまりデータベース移動が頻繁には起こらない場合には，ECF 方式が最適となる．それ以外では，問合せ間隔が小さい場合若しくは問合せ/移動比が小さい場合に DQ 方式，その他は DF 方式が最適となる．$n = 10$ では，ECF 方式が最適となる領

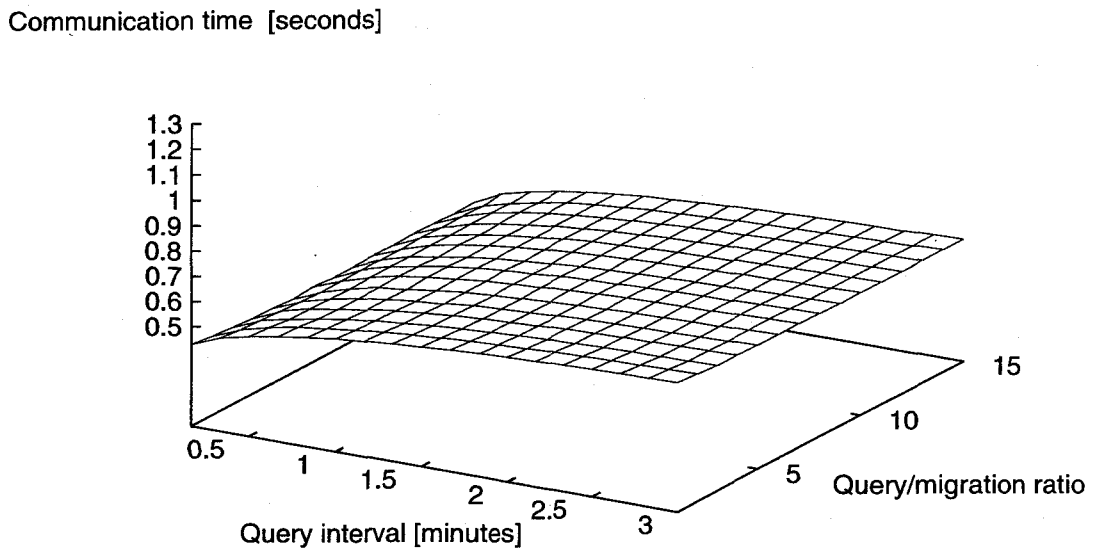


図 4.6: DF 方式の通信所要時間

域はシミュレーションを行った範囲内ではなくなり，問合せ間隔が小さい場合若しくは問合せ/移動比が小さい場合に DQ 方式，それ以外では DF 方式が最適となる．

4.5.4 シミュレーション結果の考察

DF，DQ 方式では，位置管理サーバに正確な位置情報が保持されているため，データベースの移動の影響を殆んど受けない．但し，DF 方式では三つの通信路，DQ 方式では二つの通信路を用いるため，コネクション設定時間の影響は DF 方式が大きい．従って，問合せの頻度が高い場合には，以前に設定した SVC コネクションを利用できる可能性が高いので，DF 方式の性能が良くなる．

一方，その他の四方式では，データベース移動の頻度が高くなると，目的のデータベースが存在するサイトまでに経由しなければならないサイトが増加するため性能が劣化する．また，ネットワークの規模が大きい場合もデータベースの移動が可能なサイトが増加するため性能の低下が著しい．

ECF, ECQ 方式に着目すると，サイトの航行数を削減することによる性能向上の効果が顕著に現れている．これらの手法では，処理に無関係なデータベースの位置情報も更新されるため，システム内のデータベース数が多い場合は，更にその効果が大きくなるものと

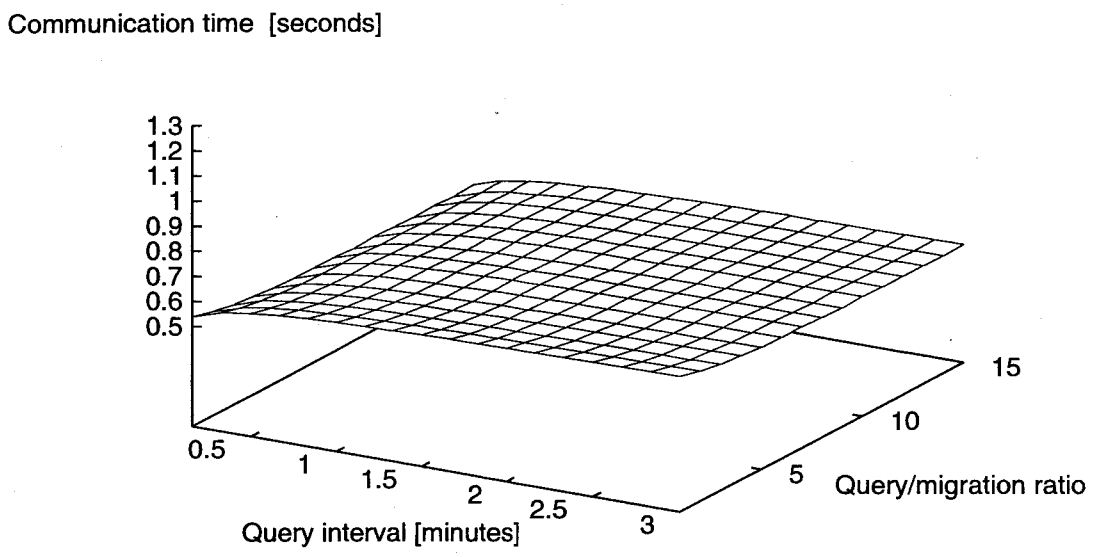


図 4.7: DQ 方式の通信所要時間

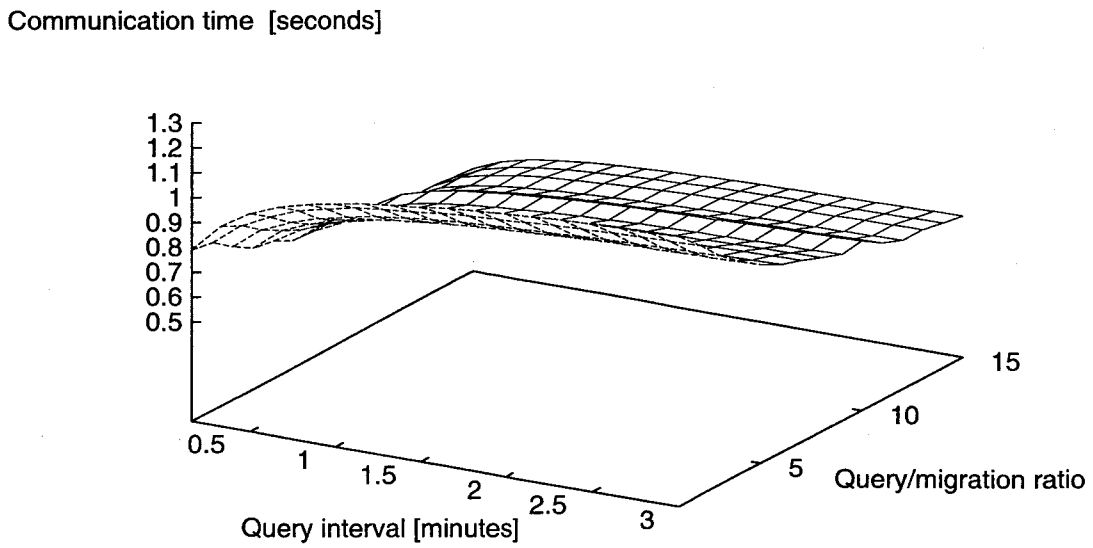


図 4.8: CF 方式の通信所要時間

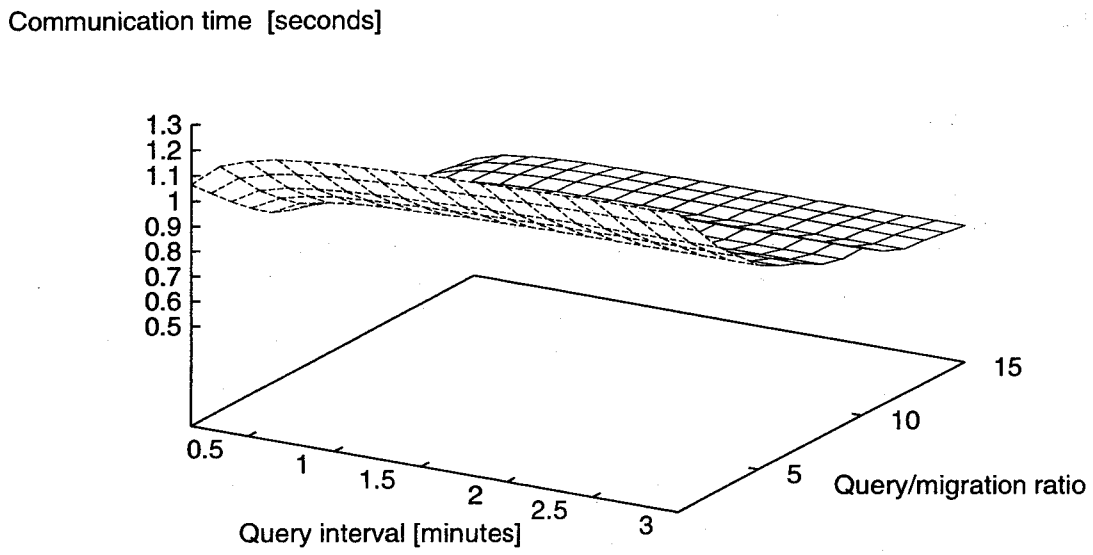


図 4.9: CQ 方式の通信所要時間

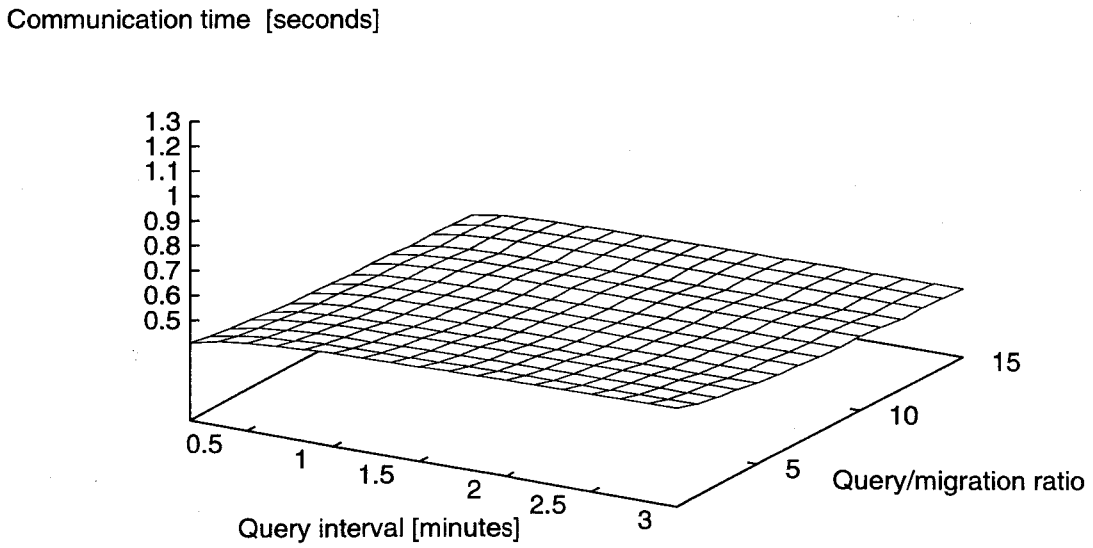


図 4.10: ECF 方式の通信所要時間

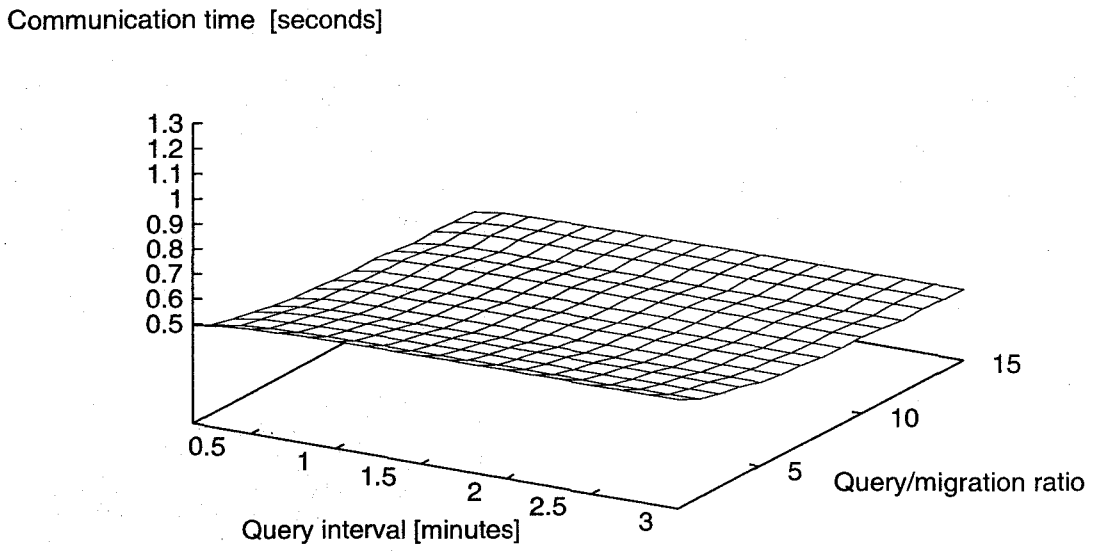


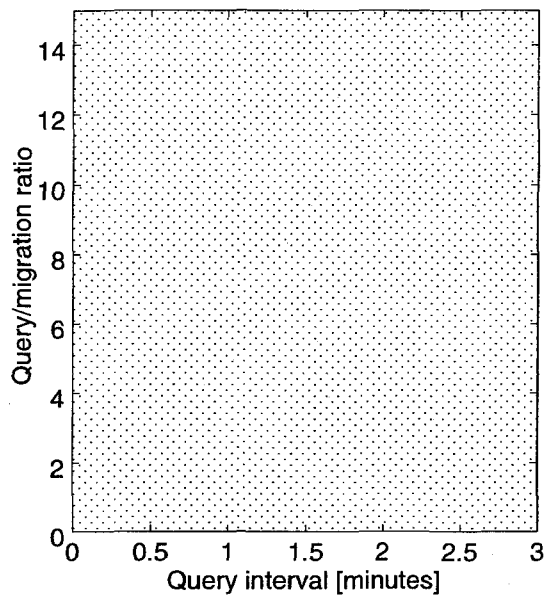
図 4.11: ECQ 方式の通信所要時間

考えられる．

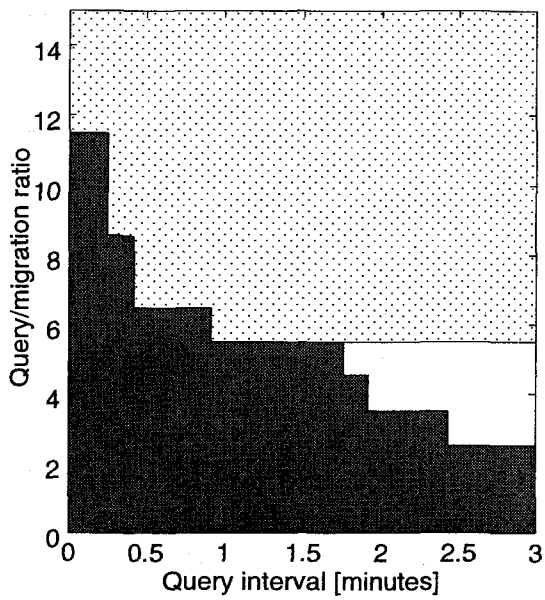
また，ECF 方式と ECQ 方式を比べると，すべての場合において ECF 方式の方が良い性能を示している．これは，ECQ 方式では，更新された LL 表の情報を経由した各サイトに伝達する際に，必ず問合せ発生サイトを経由する必要があるために，通信回数が増え，性能が低下するからである．

4.6 むすび

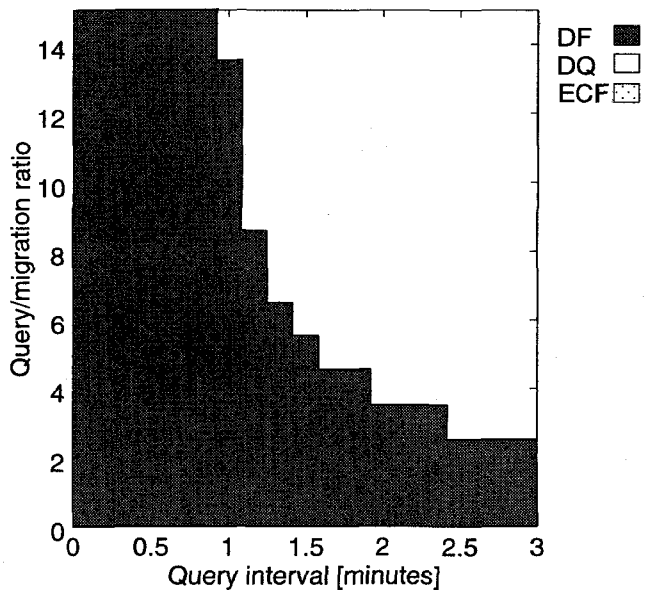
本章では，ATM の WAN 環境においてデータベース移動を利用することを想定して，7 つのデータベース位置管理手法を提案した．5 つの手法は，従来の分散システム内の資源の位置管理や移動体計算機環境での移動体の位置管理に用いられている手法ベースとするもので，残りの 2 つの手法は，ATM ネットワークの特性を考慮して新たに提案したものである．更に，シミュレーション評価により，データベースの移動頻度やシステムの規模，問合わせの発生頻度といったシステムの特性に応じて，提案したいずれの手法が最適になるかを調べた．この結果を利用することで，与えられたシステム特性に応じて，最適な位置管理手法を選択することができる．



(a) n=5



(b) n=7



(c) n=10

図 4.12: 最適手法の分布

第5章

異種WAN環境におけるデータベース移動

5.1 まえがき

第2章と第3章で提案した手法では，ATMの仮想LANなどネットワーク全体がある程度均一で広帯域な環境内のシステムを想定していた．一方，データベース移動をLANや公衆網，バックボーンなどが混在する異種WAN環境で用いる場合には，次のような理由からこれまでとは異なる手法を考える必要がある．以下では簡単のため，異種WAN環境をWAN環境と呼ぶ．

- ネットワークがLANや公衆網，バックボーンなどさまざまな部分ネットワークから構成されるため，任意の2サイト間でデータベース移動を実行するのは困難である．特に，狭帯域の公衆網を経由してデータベース移動を実行するのは現実的に不可能である．

- 複数のデータベース移動は並行して実行可能であるため，トランザクション発生時に，トランザクションのアクセス情報や必要なデータベースの（並行）移動に要する通信時間を考慮して，データベース移動の実行を決定することが有効である．つまり，移動は個々のデータベース毎ではなく，トランザクション単位で制御することが好ましい．しかし，WAN環境では，システム内のサイト数が多く，データベースの移動可能範囲がそのデータベースが存在するバックボーン内に限られていることから，各サイトがトランザクション発生時に，トランザクションの実行に必要なデータベースに対する他サイトのアクセス状況や，データベースの移動可能範囲を全て把握すること

は非現実的である．従って，各サイトがトランザクション単位で，複数のデータベースの移動を自律的に決定するのは困難である．

そこで本章では，WAN環境においてデータベース移動を利用することを想定し，バックボーンネットワーク上の各データベースに対して，与えられたアクセス系列のデータベース操作に要する通信時間を最短にする移動のスケジューリング手法を提案する．更に，本章では，提案する手法を現実のWAN環境に適用する方法について議論する．一般に，各サイトが自律的に動作している分散データベースシステムにおいて，データベースに対するアクセス系列が前もって完全に与えられるという想定は現実的ではない．しかし，アクセス混雑時には，アクセス系列は処理待ちのアクセス要求のリストとして部分的に与えられる．また，アクセス閑散時には，過去のアクセス履歴やシステム全体としてのアクセスの特徴から，ある程度アクセス系列を予測することは可能である．このように，実環境においてある程度の精度のアクセス系列が得られることから，与えられたアクセス系列に対して，通信所要時間を最短にする移動のスケジューリングを行うことは重要である．

これまでにも，データベース操作の処理時間を短縮するためにデータの再配置を行う研究として，文献[4, 84]などが報告されている．しかし，これらの研究では，リモートサイトに存在するデータ項目への読込み操作が連続して実行される時に，読み込んだデータ項目を複製として保持しておくという方法がとられている．このような手法では，システム内に小さなデータ項目の複製が多数存在することになり，表全体などといった大きなデータ単位に対する操作や，データベースに制約条件が定義されている場合の整合性チェックに多くのメッセージ通信を必要とするため，システムの性能が低下する．更に，データの移動単位が小さいと，トランザクション内で条件分岐などがある際に，操作対象となる移動単位がその操作の実行時にしか決定しない．その結果，同じ移動単位に対する連続する操作や，複数の並列実行可能な操作が存在する場合にも，効果的な移動が行えない．一方，本研究では，広帯域ネットワークを有効利用し，データベースといった大きなデータを移動単位とすることで，これらの問題を解決している．

本章では，書込み操作時に複製間の一貫性保持のためのメッセージ通信によって性能が低下することを考慮して，データベースの複製は作成しない．しかし，第3章で述べたように，実際には複製を作成することでデータベース操作のためのメッセージ通信を削減できるため，複製を作成することが有効かどうかは，ネットワークの帯域幅や書込み操作の頻度などのシステム特性に依存する．WAN環境における複製の有効性や複製の配置スケ

ジュールの検討については，今後の課題と考えている．

本章の以下の部分では，5.2節でデータベース移動のスケジューリング手法を提案し，5.3節で提案した手法が最適なスケジューリングを行えることを証明する．5.4節で提案した手法の実環境への適用法について議論する．最後に，5.5節で本章のまとめとする．

5.2 最適な移動のスケジューリング

本節では，まず本章で想定するシステム環境について述べ，その後，データベース操作のための通信時間を最短にするWAN環境に適したスケジューリング手法を提案する．

5.2.1 システム環境

本章では，図5.1に示すようなWAN環境内に構築された分散データベースシステムを想定する．ネットワークは，LAN（広帯域），公衆網（狭帯域），バックボーン（広帯域）の3つの要素から成り，LANからバックボーンへの通信は公衆網を経由しなければならない．また，図ではバックボーン部のネットワークの線の長さがさほど長く描かれていないが，実際は日本・米国間といったようにかなり長いものである．

このようなネットワーク上のシステムにおいて，システム内の各サイトは一意に識別できるものとし，各サイトをサイト識別子 S_i $(i = 1, 2, \cdots)$ で表す．特定のデータベースへの m 個の操作要求からなるアクセス系列 A に対して，データベース移動のスケジュールを $[S_{h1}, \cdots, S_{hm}]$ として与える．但し，スケジュール中の S_{hj} $(j = 1, \cdots, m)$ は，j 番目のデータベース操作の直前に，そのデータベースをサイト $S_{h(j-1)}$ から S_{hj} に移動することを示す．

本章で提案する手法では，各データベース毎に，与えられたアクセス系列に対して，そのデータベースの操作に要する通信時間を最短にするように移動を決定する．各データベース毎に移動を決定するのは，5.1節で述べたWAN環境の特徴を考慮すると，WAN環境ではトランザクション単位の移動が困難なためである．

想定するシステム環境の詳細を以下に示す．

- データベース移動はバックボーン部のみで実行される．これは，LAN内ではデータ転送の伝搬遅延が小さいためデータベース移動の効果が小さく，公衆網内ではネットワークの狭帯域性からデータベース移動に大きな時間を要するからである．但し，こ

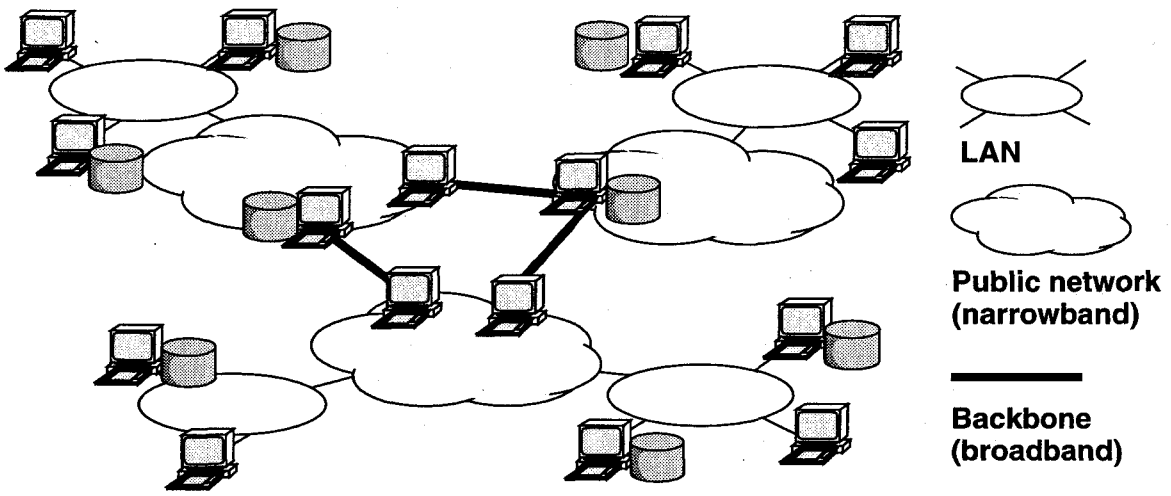


図 5.1: WAN 環境における分散データベースシステム

の仮定は，LAN 内や公衆網上のサイトにデータベースを配置すべきではないと主張するものではない．実際のシステムでは，システムの設計時に，特定の LAN 内や公衆網上のサイトからのアクセスが大部分であるデータベースはその LAN 内やそのサイトに，さまざまなサイトからアクセスされるデータベースはアクセスの特性を考慮して適当なバックボーン内のサイトに配置するなど，適切な配置を行うことがシステムの性能上有効である．

図 5.1では，1 つのバックボーンで相互接続された 2 つのサイトから成る部分ネットワーク上，および，2 つのバックボーンで相互接続された 3 つのサイトから成る部分ネットワーク上でデータベース移動が可能である．以下では，バックボーンで相互接続された部分ネットワークを，バックボーン部分ネットワークと呼ぶ．

- バックボーン部分ネットワーク内において，サイト S_i-S_j間でデータベース移動を実行するときの移動時間を $D_{i,j}$とする．これは，サイト間の帯域幅や伝搬遅延に依存して決定する．また，サイト S_i-S_j間でデータベース操作などの小さなメッセージ通信に要する通信時間を $d_{i,j}$とする．これは，サイト間の伝搬遅延に等しい．移動時間および伝搬遅延には，対称性 $D_{i,j} = D_{j,i}$, $d_{i,j} = d_{j,i}$が成立するものとする．さらに，それぞれ三角不等式 $D_{i,k} \leq D_{i,j} + D_{j,k}$, $d_{i,k} \leq d_{i,j} + d_{j,k}$が成立するものとする．この想定は，特定のサイトにメッセージ通信やデータベース移動を行う際に，一つ若しくは

複数のその他のサイトを経由するよりも，そのサイトへ通信や移動を直接行う方が通信時間が短いことを示している．この想定は，通常の通信方式を考えると一般的なものである．

- ディスクアクセスがデータベース移動のボトルネックとならないように，システム内のデータベースは全て主記憶データベース [21, 22] とする．

- データベースの複製は作成しないものとする．従って，データベース移動の後に，移動元のデータベースを削除する．

- データベースのアクセス系列は，アクセス（操作）要求を発生したサイトのリスト $\mathtt{A} = [S_{i1}, \cdots, S_{im}]$ $(m > 1)$ として与えられれる．リストの先頭は，最初に操作要求を発生したサイトの識別子であり，以降，操作要求を発生した順にサイト識別子が並ぶ．アクセス系列の一部を，アクセス部分系列と呼ぶ．

  但し，操作要求を行ったサイトがそのデータベースが存在するバックボーン部分ネットワーク外に位置する場合，S_{im}はそのサイトではなく，バックボーン部分ネットワーク内のサイトのうち，その操作要求がバックボーン部分ネットワーク内に到着する際に経由するサイトである．つまり，各データベースに対して，システム全体は，そのデータベースが存在するバックボーン部分ネットワークと，部分ネットワーク内のサイトから成る仮想的なシステムと考えることができる．

- m 個のデータベース操作から成るアクセス系列に対する移動のスケジュールは，m 個のサイト識別子から成るリスト $\mathtt{S} = [S_{h1}, \cdots, S_{hm}]$ で与えられる．ここで，リスト中の S_{hj} $(1 \leq j \leq m)$ は，先述のように，アクセス系列中の j 番目の操作の実行直前に，データベースをサイト S_{hj}に移動することを表している．スケジュールの系列の一部を，部分スケジュールと呼ぶ．

- 現在位置が S_Iであるデータベースに対するアクセス系列 $\mathtt{A} = [S_{i1}, \cdots, S_{im}]$ を，スケジュール $\mathtt{S} = [S_{h1}, \cdots, S_{hm}]$ に基づいてデータベース移動を実行して処理した場合の通信時間を $T(S_I, \mathtt{A}, \mathtt{S})$ と表す．簡単のために，これをスケジュール $\mathtt{S}$ のコストと呼び，次式のように表す．但し，S_{h0}を S_Iとする．

$$T(S_I, \mathtt{A}, \mathtt{S}) = \sum_{j=1}^{m} \left\{ D_{h(j-1),hj} + d_{ij,hj} \right\}. \tag{5.1}$$

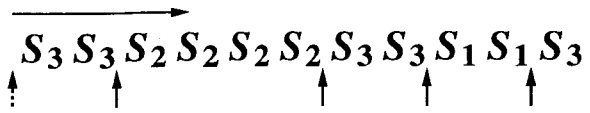


図 5.2: アクセス系列と変化点

5.2.2 移動のスケジューリング

本節では，与えられたアクセス系列に対して，そのデータベース操作に要する通信時間を最短にする移動のスケジューリングについて議論する．図5.2は，アクセス系列の一例である．図5.2における上向きの実線矢印は，アクセス要求を発生するサイトが別のサイトに変化する点を表しており，これを変化点 (turning point) と呼ぶ．また，変化点の直後のサイトを変化点サイトと呼ぶ．最初のアクセス要求がデータベースの初期位置とは異なるサイトから発生する場合は，アクセス系列の先頭も変化点となる（図の点線矢印）．

与えられた系列に対して，全てのアクセス点であらゆるサイトに移動するケースを考慮して最適な移動のスケジューリングを行うと，操作対象となるデータベースが存在するバックボーン部分ネットワーク内のサイト数を N，アクセス系列の要素数を m としたとき，その計算量は N^2m と膨大なものになる．そこで本章では，より少ない計算量で最適な移動のスケジューリングを行う手法を提案する．手法のアルゴリズムを以下に示す．

アルゴリズム：

1. データベースの初期位置を表す節点を根節点とし，これをレベル1節点と呼ぶ．また，節点値を0とする．

2. アクセス系列を走査して変化点に到達したら，現在の最大レベルの各節点（レベル i 節点）に，部分ネットワーク内のサイトを表す最大 N 個の子節点を作成する．これらの子節点をレベル $i+1$ 節点とする．

   ここで，N は注目している部分ネットワーク内の全サイト数である．但し，子節点を作成するのは，その子節点が表すサイトと変化点サイト間の伝搬遅延が，親節点が表すサイトと変化点サイト間の伝搬遅延と同じ若しくは小さくなるもののみである．従って，親節点が変化点サイトを表す場合は，子節点として変化点サイトを表す1つの節点のみが作成される．

3. サイト S_j を表す子節点の節点値を，親の節点値に $D_{p,j} + L \cdot d_{t,j}$ を加えたものとする．但し，親節点と同一のサイトを表す子節点では，$L \cdot d_{t,j}$ のみを加える．ここで，親節点が表すサイトを S_p，変化点サイトを S_t，注目している変化点から次の変化点（以降に変化点がない場合はアクセス系列の最後点）までに含まれるアクセス数を L としている．

   アクセス系列中で，以降に変化点が存在しない場合は手順（5）に，存在する場合は手順（4）に進む．

4. 全てのレベル $i+1$ 節点の中で，同じサイトを表しているものの中から，節点値が最小となるもののみを残し，それ以外は削除する．節点値が最小となる節点が複数存在する場合は，それらのうち 1 つのみを残して，それ以外は削除する．

   これらの操作の終了後，手順（2）に戻る．

5. レベル $i+1$ 節点中で節点値が最小となるものを見つける．根からその節点を結ぶ道を調べ，各変化点において，対応する節点が表すサイトにデータベースが位置するように，移動のスケジューリングを決定する．(アルゴリズムの終了)

与えられたアクセス系列中の変化点の個数を M とすると，アルゴリズムの終了時には，データベースの初期位置を表す節点を根とした，各レベル（最大レベル以外）の節点の総数が最大 N で深さが M の木が作成される．アルゴリズムの手順（3）において，子の節点値として与えられる値の親節点の節点値からの増加分 $(D_{p,j} + L \cdot d_{t,j})$ は，注目している変化点から次の変化点までのデータベースアクセスをデータベース移動によってサイト S_j に移動して実行した場合に要する通信時間を表している．従って，各節点の節点値は，アクセス系列の先頭から注目している次の変化点までのデータベースアクセスを，根からその節点にたどり着くまでの道に対応するようにデータベース移動を実行して処理した場合の通信時間を表している．

手順（4）から分かるように，最大レベル以外の各レベルでは，同一のサイトを表す節点のうち最小の節点値をもつもの以外は削除されるため，各レベルでは $\sum_{i=1}^{N}(i-1)$ 回の節点値の比較により N 個の節点が残る．また，最大レベルでは，$\sum_{i=1}^{N} i$ 個の節点が存在し，$\sum_{i=1}^{N} i - 1$ 回の節点値の比較が行われる．従って，このアルゴリズムの時間的な計算量は N^2M オーダとなる．ここで，一つのトランザクションは複数のデータベース操作を含むため，あるサイトから特定のデータベースにアクセスが連続して発生することが多い．つ

まり，アクセス系列中の要素数を m とすると，一般に $m >> M$ となる．従って，提案アルゴリズムによって，最適なスケジュールを発見する計算量が大幅に削減できるものと考えられる．

5.2.3 実行例

本節では，提案したアルゴリズムの実行例を示す．まず，システム環境としては，図5.3に示すようなバックボーンの部分ネットワークを想定する．サイト間の伝搬遅延は S_1-S_2, S_2-S_3, S_3-S_1間でそれぞれ 0.12 秒, 0.08 秒, 0.10 秒とする．また，サイト間のデータベース移動に要する通信時間は S_1-S_2, S_2-S_3, S_3-S_1間でそれぞれ 0.44 秒, 0.36 秒, 0.40 秒 とする．ここで，データベース移動に要する時間は，データベースのサイズを 50Mbytes，バックボーンの帯域幅を 2Gbps と想定したものである．データベース移動を安全に行うためには，移動元サイトから移動先サイトへのデータベース転送と，移動先サイトから移動元サイトへの移動完了通知の 2 回の通信が必要になるため，上記の値はこれを移動完了通知の転送遅延を 0 に近似して計算したものである．

ここで，図 5.4の上部に示すように，データベースの初期位置を S_1，アクセス系列を $\mathtt{A} = [S_3, S_3, S_2, S_2, S_2, S_2, S_3]$ としたときに，提案したアルゴリズムによって移動のスケジューリングを行う．図5.4のアクセス系列中に示された上向きの矢印は，変化点を表している．

図 5.4の下部に，アルゴリズムの実行結果を示す．実行結果のグラフにおいて，節点の左に記されているのがその節点の節点値であり，枝に記されているのがその枝が結ぶ親節点と子節点の節点値の差分である．各レベルにおいて，変化点サイトとの伝搬遅延を増加しないサイトを表す節点を作成し，そのサイトに移動した場合のアクセスに要する通信時間を計算する．レベル 3 では，同じサイトを表す節点のうち最小の節点値となるもの以外を削除する．最終的に，レベル 4 において最小の節点値となる節点を見つける．この結果，(S_1, S_2, S_2, S_2) が最小の節点値を与える経路となるため，各変化点においてこの経路に対応するように移動を実行する．つまり，この場合の最適な移動のスケジューリングは，最初の変化点においてデータベースを S_2へ移動し，その後の変化点では移動を行わないものである ($\mathtt{S} = [S_2, S_2, S_2, S_2, S_2, S_2, S_2]$).

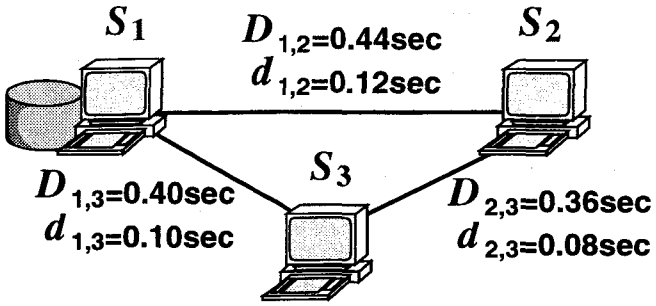


図 5.3: システム例

5.3 アルゴリズムの正当性

本節では，提案したアルゴリズムによって，与えられたアクセス系列に対して通信時間を最短とする移動のスケジューリングが行えることを証明する．

まず，変化点に関する性質を示す．

定理 1：

変化点以外の移動を含まない最適なスケジュールが存在する．

定理 1 の証明：

証明にあたり，まず次の補題を示す．

補題 1：

変化点を含まないアクセス部分系列中の一点で移動を行う場合，アクセス部分系列の両端以外の点での移動が通信時間を最短にするときは，両端での移動も通信時間を最短にする．

補題 1 の証明：

ここで，サイト S_iからの b 回のデータベース操作から成る単純なアクセス系列 $\mathtt{A}_1$を考える．図 5.5に示すように，a 番目の操作後にサイト S_{j0}（初期位置）から S_{j1}への移動を実行する場合，このスケジュール $\mathtt{S}_1$は $\mathtt{S}_1 = [S_{j0}, \cdots, S_{j0}, S_{j1}, S_{j1}, \cdots, S_{j1}]$ で表される．ここで，スケジュールの 1 番目から a 番目までの要素は全て S_{j0}で，$a+1$ 番目以降の要素は全て S_{j1}である．このとき，スケジュール $\mathtt{S}_1$のコスト $T(S_{j0}, \mathtt{A}_1, \mathtt{S}_1)$ は，式 (5.1) より次式のようになる．

$$
\begin{aligned}
&T(S_{j0}, \mathtt{A}_1, \mathtt{S}_1) \\
&\quad = D_{j0,j1} + a \cdot d_{i,j0} + (b-a) d_{i,j1}.
\end{aligned}
\tag{5.2}
$$

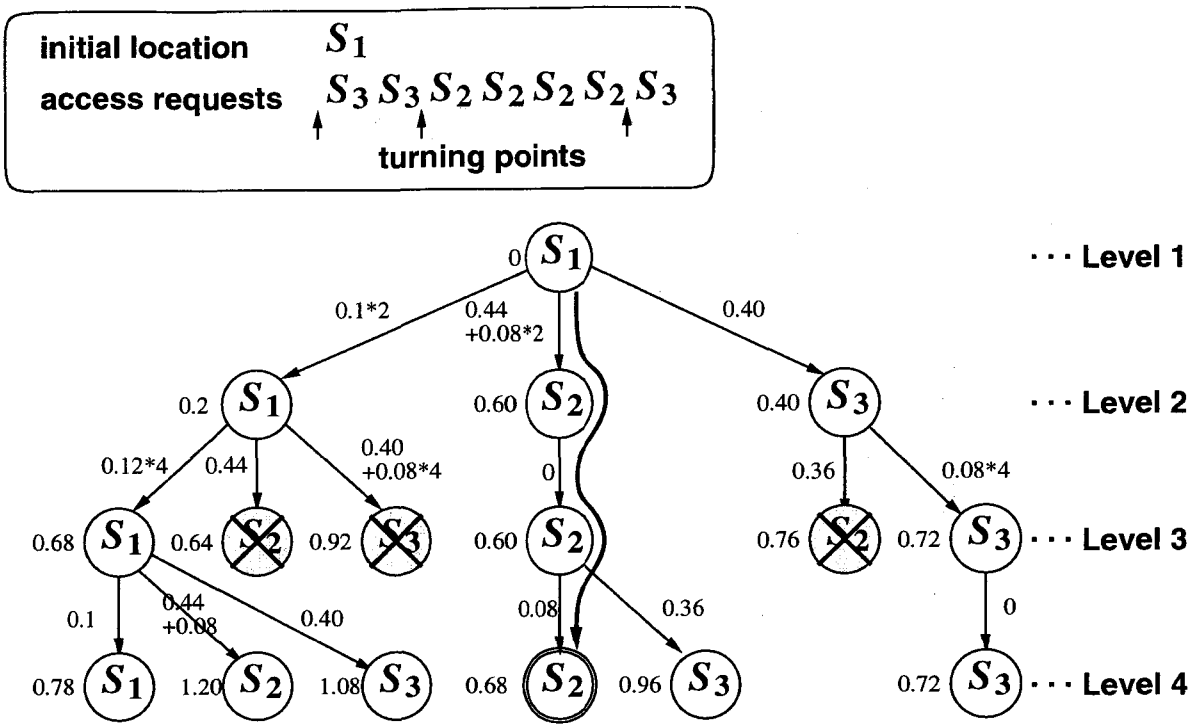


図 5.4: アルゴリズムの実行例

式 (5.2) において，a に依存するのは第二項と第三項である．この式の値を最小にする a $(0 \leq a \leq b)$ の値は，次のようになる．

$$\begin{array}{ll} 0 & (d_{i,j0} > d_{i,j1}\text{のとき}) \\ b & (d_{i,j0} < d_{i,j1}\text{のとき}) \\ \text{任意} & (d_{i,j0} = d_{i,j1}\text{のとき}). \end{array}$$

従って，アクセス部分系列中の両端でない点における移動によって通信時間が最短となるのは $d_{i,j0} = d_{i,j1}$のときであり，この場合は任意の点での移動が同じ通信時間となるため，アクセス部分系列の両端の移動においても通信時間は最短となる．□

補題1により，$d_{i,j0}$と $d_{i,j1}$がどのような値でも，0 から b までのアクセス部分系列の両端のいずれかは a の最適位置となる．

ここで，図 5.6に示すように，両端を変化点とし内部には変化点を含まない，サイト S_i からの b 回のアクセスから成るアクセス部分系列を考える．データベースの初期位置を S_{j0} とし，アクセス部分系列中の n 個の点で移動を実行するものとする．n 個中の k 番目の点

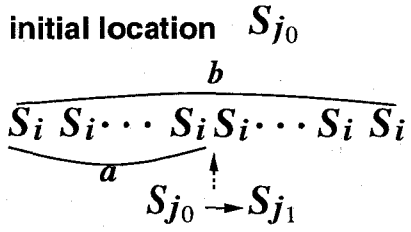


図 5.5: 補題 1 の証明における記号の定義

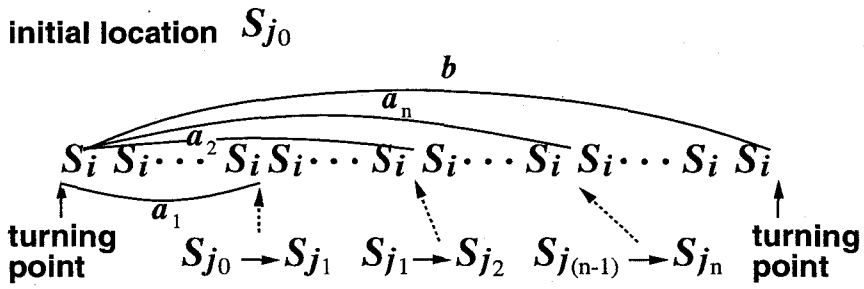


図 5.6: 定理 1 の証明における記号の定義

は，アクセス部分系列の先頭からの距離が a_kで，その点でサイト $S_{j(k-1)}$から S_{jk}への移動が実行される．

まず，先頭から a_2までの領域において，最初の a_2回のデータベース操作に要する通信時間を最短にする a_1の値を考える．ここで補題 1 より，$a_1 = 0$ もしくは $a_1 = a_2$のいずれかが a_1の最適値となる．$a_1 = 0$ が a_1の最適値となる場合は，a_1での移動は図 5.6のアクセス部分系列の先頭（変化点）で実行される．$a_1 = a_2$が a_1の最適値なる場合は，a_1での移動が a_2で実行されるため，結局 a_2では S_{j0}から S_{j2}までの 2 回の移動が実行されることになる．仮定より，この時は S_{j0}から S_{j2}へ直接移動した方が通信時間が短くなるため，実際には 1 回の移動となる．

次に，a_2の最適値を考える．a_1の最適値が 0 の場合，0 から a_3の領域のある一点で S_{j1}から S_{j2}への移動を実行するときに，a_3回のデータベース操作のための通信時間を最短にする a_2が，a_2の最適値である．a_1の最適値が a_2の場合，同じ領域で S_{j0}から S_{j2}への移動を実行するときに，a_3回のデータベース操作のための通信時間を最短にする a_2が，a_2の最適値である．いずれの場合も，補題 1 より，$a_2 = 0$ もしくは $a_2 = a_3$のいずれかで通信時間が最短

となる．

同様にして，a_k点 $(2 < k < n)$ までの k 回の移動は，そのうち最初の k_1 回 $(0 \leq k_1 \leq k)$ をアクセス部分系列の先頭で，それ以降の移動を次の点 a_{k+1}で実行することにより通信時間が最短となることを帰納的に示すことができる．最終的に，a_n点までの n 回の移動は，最初の n_1回 $(0 \leq n_1 \leq n)$ をアクセス部分系列の先頭で，それ以降の移動をアクセス部分系列の最後の点で実行することにより通信時間が最短となる．ここで，仮定より，複数の移動より1回の移動で特定サイトに移動する方が通信時間が短くなるため，アクセス部分系列の先頭で S_{j0}から $S_{j(n_1)}$への移動が実行され，最後の点で $S_{j(n_1)}$から S_{jn}への移動が実行される．n_1の値に関係なく，変化点上での移動が通信時間を最短にする．

以上のことから，変化点を両端とする内部に変化点を含まないアクセス部分系列では，両端の変化点のみでのデータベース移動を考慮することで，データベース操作に要する通信時間を最短とするスケジューリングが可能となる．□（定理1の証明終）

定理1により，本章で提案したアルゴリズムの手順（2）において，各変化点での移動のみを考慮していることの正当性が保証される．

以下では，リスト a とリスト b の追加 (append) 操作を，a + b と表す．

定理2：

変化点サイトまでの伝搬遅延がデータベースの現在位置のサイトよりも大きくなるサイトへの移動は，通信時間を最短にする移動のスケジュールには含まれない．

定理2の証明：

証明にあたり，まず次の補題を示す．

補題2：

現在位置 S_Iのデータベースのアクセス系列 A がアクセス部分系列 a_1と a_2から成り $(\mathrm{A} = \mathrm{a}_1 + \mathrm{a}_2)$，A に対するスケジュール S が a_1と a_2に対する部分スケジュール s_1と s_2からそれぞれ成るとき $(\mathrm{S} = \mathrm{s}_1 + \mathrm{s}_2)$，次式の関係が成立する．但し，式中の S_Jは，スケジュール s_1を実行した後のデータベースの位置である．

$$\begin{aligned} T(S_I, \mathrm{A}, \mathrm{S}) &= T(S_I, \mathrm{a}_1 + \mathrm{a}_2, \mathrm{s}_1 + \mathrm{s}_2) \\ &= T(S_I, \mathrm{a}_1, \mathrm{s}_1) + T(S_J, \mathrm{a}_2, \mathrm{s}_2). \end{aligned}$$

補題2の証明：

s_1と s_2が，それぞれ m_1個および $m - m_1$個のサイト識別子から成るものとすると，式 (5.1) より，

$$
\begin{aligned}
T(S_I, \mathtt{A}, \mathtt{S}) &= \sum_{j=1}^{m} \left\{ D_{h(j-1),hj} + d_{ij,hj} \right\} \\
&= \sum_{j=1}^{m_1} \left\{ D_{h(j-1),hj} + d_{ij,jk_j} \right\} + \sum_{j=m_1+1}^{m} \left\{ D_{h(j-1),hj} + d_{ij,jk_j} \right\} \\
&= T(S_I, \mathtt{a}_1, \mathtt{s}_1) + T(S_J, \mathtt{a}_2, \mathtt{s}_2). \qquad \square
\end{aligned}
$$

ここで，現在位値 S_Iのデータベースのアクセス系列 $\mathtt{A} = \mathtt{a}_1 + [S_i] + \mathtt{a}_2$（$S_i$は変化点サイト）に対して，$\mathtt{a}_1$後の変化点において $d_{i,j0} < d_{i,j1}$となるサイト S_{j1}への移動を含むスケジュール $\mathtt{S} = \mathtt{s}_1 + [S_{j1}] + \mathtt{s}_2$を考える．但し，$\mathtt{s}_1$および $\mathtt{s}_2$は，それぞれアクセス部分系列 $\mathtt{a}_1$, $\mathtt{a}_2$に対する部分スケジュールであり，$\mathtt{s}_1$実行後のデータベースの位置を S_{j0}とする．

更に，注目している変化点においてデータベース移動を実行せず，その次の変化点でサイト S_{j0}から S_{j1}の移動を実行すること以外は，スケジュール S と等しい別のスケジュールを S' $= \mathtt{s}_1 + [S_{j0}] + \mathtt{s'}_2$とする．スケジュール S において，注目する次の変化点でも別のデータベース移動（S_{j1}から S_{j2}）が実行される場合，スケジュール S' では，次の変化点で S_{j0}から S_{j1}への移動と S_{j1}から S_{j2}への移動が連続して実行されるものとする．(実際には，S_{j0}から S_{j2}への 1 回の移動が実行される．) $\mathtt{s'}_2$は，S' におけるその変化点以後の部分スケジュールである．

このとき，S と S' のコストはそれぞれ次式で表される．

$$
\begin{aligned}
T(S_I, \mathtt{A}, \mathtt{S}) &= T(S_I, \mathtt{a}_1 + [S_i] + \mathtt{a}_2, \mathtt{s}_1 + [S_{j1}] + \mathtt{s}_2) \\
&= T(S_I, \mathtt{a}_1, \mathtt{s}_1) + D_{j0,j1} + n \cdot d_{i,j1} + T(S_{j1}, \mathtt{a}_2, \mathtt{s}_2) \\
T(S_I, \mathtt{A}, \mathtt{S'}) &= T(S_I, \mathtt{a}_1 + [S_i] + \mathtt{a}_2, \mathtt{s}_1 + [S_{j0}] + \mathtt{s}'_2) \\
&= T(S_I, \mathtt{a}_1, \mathtt{s}_1) + n \cdot d_{i,j0} + D_{j0,j1} + T(S_{j1}, \mathtt{a}_2, \mathtt{s}_2).
\end{aligned}
$$

n は注目している変化点とその次の変化点の間に含まれるデータベース操作数である ($n > 0$)．ここで，$d_{i,j0} > d_{i,j1}$より，$T(S_I, \mathtt{A}, \mathtt{S}) > T(S_I, \mathtt{A}, \mathtt{S'})$ となる．□（定理 2 の証明終）

定理 2 により，提案したアルゴリズムの手順 (2) において，子節点が表すサイトと変化点サイト間の伝搬遅延が，親節点を表すサイトと変化点サイト間の伝搬遅延以下になる子節点のみを作成していることの正当性が保証される．

定理 3：

実行後のデータベースの位置を S_{j0}とする部分スケジュール s に対して，実行後のデータベースの位置を S_{j0}とし，スケジュールのコストが s よりも小さくなる別の部分スケジュールが存在するとき，s は通信時間を最短にするスケジュールの一部とはならない．

定理3の証明：

現在位置S_Iのデータベースのアクセス系列$\mathtt{A} = \mathtt{a}_1 + \mathtt{a} + \mathtt{a}_2$に対するスケジュール$\mathtt{S} = \mathtt{s}_1 + \mathtt{s} + \mathtt{s}_2$を考える．$\mathtt{s}_1$, $\mathtt{s}$, $\mathtt{s}_2$は，それぞれアクセス部分系列$\mathtt{a}_1$, $\mathtt{a}$, $\mathtt{a}_2$に対する部分スケジュールである．また，$\mathtt{s}_1$, $\mathtt{s}_2$実行後のデータベースの位置を，それぞれS_{j0}, S_{j1}とする．

ここで，$T(S_{j0}, \mathtt{a}, \mathtt{s}) > T(S_{j0}, \mathtt{a}, \mathtt{s'})$となり，実行後のデータベースの位置を$S_{j1}$とする部分スケジュール$\mathtt{s'}$を含むスケジュール$\mathtt{S'} = \mathtt{s}_1 + \mathtt{s'} + \mathtt{s}_2$を考える．このとき，次の関係が成立する．

$$
\begin{aligned}
T(S_I, \mathtt{A}, \mathtt{S}) &= T(S_I, \mathtt{a}_1, \mathtt{s}_1) + T(S_{j0}, \mathtt{a}, \mathtt{s}) + T(S_{j1}, \mathtt{a}_2, \mathtt{s}_2) \\
&> T(S_I, \mathtt{a}_1, \mathtt{s}_1) + T(S_{j0}, \mathtt{a}, \mathtt{s'}) + T(S_{j1}, \mathtt{a}_2, \mathtt{s}_2) \\
&= T(S_I, \mathtt{A}, \mathtt{S'}).
\end{aligned}
$$

□（定理3の証明終）

定理3により，提案したアルゴリズムの手順（4）において，各レベルで同じサイトを表している子節点のうち節点値が最小とならないものを削除することの正当性が保証される．

以上の議論より，次のような系が導かれる．

系：

提案したアルゴリズムによって，通信時間を最短にする移動のスケジューリングを行える．

5.4 実環境への適用

本節では，提案したアルゴリズムを，実際のWAN環境に適用するための実現法について議論する．

アルゴリズムのWAN環境への適用を考えたとき，次のような問題が生じる．

- バックボーン部を移動するデータベースの位置管理をどのようにするのか．
- データベースへのアクセス系列をどのようにして得るのか．
- 移動のスケジューリングをどのサイトが行うのか．

これらの問題に対しては，バックボーン部の各データベースに，その位置情報の管理やスケジューリングを行う特定のサイト（以後，ホームサイトと呼ぶ）を設定するという解決策が考えられる．ホームサイトとデータベースが現在位置するサイトの間では，小さな

メッセージの交換しか行われないため，ホームサイトは必ずしもそのデータベースが存在するバックボーン部内のサイトでなくてもよい．しかし，メッセージ通信の伝搬遅延が大きくならないように，そのバックボーン部の近くに位置することが望ましい．

この場合，ホームサイトでデータベースの位置管理を行うために，データベースが移動する度に，移動に関与したサイトが，ホームサイトにそのデータベースの新しい位置を通知する．従って，ホームサイトは常に担当するデータベースの現在位置を把握できる．WAN 環境中の全サイトは，そのデータベースの現在位置を意識することなく，アクセス要求をホームサイトへと転送する．アクセス要求を受け取ったホームサイトは，その要求をデータベースの現在位置へとフォワードする．このような位置管理手法は，4章において DF（デフォルトフォワーディング）方式として提案されおり，サイト数が多い WAN 環境に適した手法である．

アクセス系列の獲得は，データベース操作のためのキューをホームサイトで保持し，処理待ち状態の操作要求の情報をアクセス系列情報として利用することで実現できる．なお，操作要求の並べ換えによりデータベース操作に要する通信時間を短縮することが可能であるが，これを行わないものとする．WAN 環境では，多数のサイトがさまざまなトランザクションを発生するため，操作要求の順序を変更することによって，容易にデッドロックが生じるからである．また，待ち状態の操作要求がない場合，つまりトランザクションの発生頻度が小さい場合は，最近の履歴情報などから後のアクセス系列を予測する．この場合は，移動にかかる時間がデータベース操作の処理スループットに与える影響が小さいため，アクセス系列の予測が不十分でも大きな問題とならない．

このようにして得たアクセス系列情報に基づき，ホームサイトにおいて，提案したアルゴリズムを用いて移動のスケジューリングを行う．

この他にも，バックボーン部分ネットワーク内の全てのサイトでデータベースの位置情報やアクセス情報を保持する方法なども考えられる．ここでは，これ以上の詳細は示さないが，提案アルゴリズムの実現法はシステム特性に応じて決定する必要がある．

5.5 むすび

WAN 環境ではトランザクション単位でのデータベース移動の制御およびバックボーン部以外でのデータベース移動が困難なことを考慮して，バックボーン部においてデータベース操作に要する通信時間を最短にするための，データベース移動のスケジューリングアル

ゴリズムを提案した．このアルゴリズムでは，バックボーン部分ネットワーク内のサイト数N，アクセス系列中の変化点数Mに対して，N^2Mオーダの計算量で最適な移動のスケジューリングが可能である．

更に，提案したアルゴリズムを実際のWAN環境に適用するための実現法について議論した．各データベースに対してホームサイトを設置し，これにデータベースの位置管理や操作要求情報を管理させることで，提案したアルゴリズムの実現が可能となる．

第 6 章

結論

6.1 本論文のまとめ

本論文では，広帯域ネットワーク上でのデータベース移動を分散データベースシステムにおけるトランザクション処理に応用する方法について議論した．

まず第 1 章では，広帯域ネットワーク上の分散データベースシステムでは帯域幅の有効利用による通信回数の削減がシステムの性能向上の重要な要因であることを述べた．その上で，データベース移動が有効な手段であることを言及し，本論文の論点を明らかにした．

第 2 章では，データベース移動に基づく分散データベースシステム DB-MAN を提案した．DB-MAN は，これまでの分散データベースとは大きく異なり，トランザクション処理手法の選択機構とデータベース移動を考慮した並行処理制御機構といった二つの特徴的な機構をもつ．更に，提案したシステムの有効性を検証することを目的として，シミュレーションによる性能評価も行った．手法選択機構の評価のためのシミュレーションの結果から，DB-MAN システムの履歴統計モードを用いることにより，処理効率が大幅に改善されることを確認した．履歴統計モードの履歴依存係数と連続使用優先係数の設定が，処理効率に大きく影響することも分かった．また，並行処理制御機構の評価のためのシミュレーションの結果から，DB-MAN の並行処理制御機構によって，データベース移動を導入することによる並行処理性能の劣化を防げることを確認した．

第 3 章では，データベース移動を用いたデータベース複製の動的再配置手法を提案した．この手法では，第 2 章で提案した手法とは異なり，データベース移動後に移動元のデータベースを削除せずに複製として残すことで，データベース操作のための通信回数の削減を

図っている．各サイトにおいて，制限された主記憶領域の範囲内で，利用価値の低い複製を現トランザクションで利用するデータベースの複製と置き換えることで，効果的な複製の再配置を実現している．シミュレーション評価により，複製を作成しない第2章の手法，および，静的な複製配置法との性能比較を行った．その結果，提案した動的複製配置法は記憶領域がある程度大きい場合や非常に小さい場合に有効で，第2章の手法はネットワーク帯域幅が大きい場合に有効であることを確認した．

第4章では，データベース移動を実環境において利用する際に重要な問題となるデータベースの位置管理について，ATM の WAN 環境を想定して7つの手法を提案した．提案した位置管理手法のうち，5つは従来の分散システム内の資源の位置管理や移動体計算機環境での移動体の位置管理に用いられている手法をデータベース移動に適用したものである．残りの2つの手法は，ATM ネットワークの特性を考慮して新たに提案したものである．更に，シミュレーション評価により，データベースの移動頻度やシステムの規模，問合わせの発生頻度といったシステムの特性に応じて，提案したいずれの手法が最適になるかを調べた．また，本論文では記述していないが，筆者らの研究グループでは，システム内にデータベースの複製が存在する環境を想定し，第4章の手法を拡張したいくつかの位置管理手法を提案している．その成果は，文献 [8] において公表している．

第5章では，異種 WAN 環境においてデータベース移動を利用することを想定し，バックボーン部分ネットワーク上の各データベースに対して，与えられたアクセス系列のデータベース操作に要する通信時間を最短にする移動のスケジューリング手法を提案した．この手法では，バックボーン部分ネットワーク内のサイト数 N，アクセス系列中の変化点数 M に対して，N^2M オーダの計算量で最適な移動のスケジューリングが可能である．更に，提案した手法を現実の WAN 環境に適用するために，各データベースに対してホームサイトと呼ぶ特定のサイトを決定し，ホームサイトにおいてそのデータベースの位置管理と移動のスケジューリングを行うに方法について議論した．

6.2 今後の課題

今後の課題としてまず挙げられるのが，本研究で提案した DB-MAN システムおよび動的複製配置法，位置管理手法の実システム上での実現である．本研究では，提案したシステムと手法の有効性をシミュレーション評価によって示しているが，サイトやネットワークが健全であるなど理想的な環境を想定している．しかし，実環境では，例えばサイトの

負荷やネットワークの輻輳などから，同一サイト間でもネットワークの遅延がシミュレーションで設定したような一定値とならないことが一般的である．このようなシミュレーションモデルと実環境との差異の影響や，提案したシステムと手法のプロトコルオーバヘッドは，これらを実装し実測によって評価する必要がある．更に，その評価の結果から，提案したシステムと手法の設計上の問題点を同定し，改善策を検討しなければならない．なお，筆者らの研究グループでは，現在，DB-MAN のプロトタイプシステムの設計および実装を進めており，現段階での成果を文献 [5], [6], [7], [9], [10], [11], [67], [68], [80] で公表している．今後は，このシステムの開発を継続し，提案した手法のこのシステム上で実現する予定である．

また，本研究ではデータベース移動を実環境で利用する際の技術課題について位置管理のみに着目したが，その他にもいくつかの技術課題が残されている．今後は，これらについて，具体的な手法を考案し解消策を検討する必要がある．更に，これらの手法を実装し実測評価することによって，提案した手法の有効性，ならびに，データベース移動の現実性について検証する必要がある．以下に，今後検討予定である技術課題を列挙し，その解決策の概要について述べる．

6.2.1 計算機の異種性

異種計算機環境においてデータベース移動を効率的に実行するためには，データベース内のデータの統一的な表現が必要になる．異種計算機環境におけるデータ通信のために，構造化データの統一的なバイナリ表現として，いくつかのものが規定されている．例えば，国際標準化機構 (International Standardization Organization: ISO) によって規定されている抽象構文記法 1 (Abstract Syntax Notation One: ASN.1)[1, 2] や，RPC (Remote ProcedureCall) のために独自に開発された XDR (eXternal Data Representation)[75]，構造化文書のための SGML (Standard Generalized Markup Language)[3] などがこれにあたる．

筆者らは，これまでに ASN.1 をデータ表現形式とするデータベースシステムを提案しており [25, 46, 47, 66]，このシステムのアーキテクチャを応用することで異種計算機環境におけるデータベース移動を実現できる．

6.2.2 動的スキーマ生成

より柔軟なデータベース移動を実現するために，データベースの受信側でそのデータベースのスキーマを知らなくてもデータベースの移動が行えるようにすべきである．そのためには，データベースの中身（データ）を転送する前に，スキーマ情報やデータベースのサイズ，あるいは，データベースに付加されているインデックスなどを受信側に転送する必要がある．受信側では，これらの情報を受け取った後，データベースの転送に備えて，スキーマを動的に生成しデータベースのためのメモリ領域を確保しなければならない．

6.2.3 手法選択に適したトランザクション形式

提案したDB-MANシステムと動的複製配置法では，トランザクション発生時に，データベース操作の対象となるデータベースやそのトランザクションを固定処理で処理した場合の通信回数などを予め知る必要がある．しかし，一般的にはトランザクション内で条件分岐などが存在するため，これらの情報を予測することは困難である．データベースの分割法を工夫して，条件分岐の対象となるデータ項目同士を一つのデータベースとしたり，データベースのサイズを大きくすることで，このような問題を緩和できるが，完全に解決することは困難である．従って，DB-MANシステムや動的複製配置法では，例えば，一つのトランザクション内では条件分岐が存在しないとか，そのトランザクションでの最悪な場合（用いるデータベースのサイズが最大，通信回数が最大）のデータベースサイズと固定処理での通信回数を指定するといったように，特定のトランザクションの形式が必要になる．

6.2.4 データベースの最適単位

前節で述べたように，データベースの分割法として，各データベースのサイズを大きくとることで，データベース操作の対象となるデータベースやその他の手法選択に必要な情報をトランザクションの開始時に高い確率で同定できる．また，属性値に関して相互に制約をもつ複数の属性を一つのデータベースに含ませることで，制約条件のチェックに要する通信を削減できる．更に，大きなデータベース単位は，小さな場合に比べて，データベースの位置管理のためのオーバヘッドが小さい．しかし，データの転送遅延やデータベース移動の並行処理性能を考えると，明らかにデータベース単位が小さい方が有利である．

従って，システムの性能を最大にするためには，上記のような様々なシステム特性を考

慮して，最適なデータベース単位を選択する必要がある．また，一般にシステム特性は時間経過とともに変化するため，これらを検出する手法も必要となる．

6.2.5 データベース移動に適した主記憶データベースの物理構造

より高速なデータベースの移動を実現するためには，移動時に必要な処理が少なくなるような物理構造にしなければならない．そこで，分散システム内の各データベースに対して，それぞれが存在するサイトにおいて連続した主記憶上の領域を与えるようにすればよい．このように，各データベースの物理記憶上の領域を連続的なものにすることで，移動時にデータの再構成などが必要なく，そのまま主記憶上からデータを連続的に読出しながら転送することが可能となる．従って，高速な DB 移動を実現できる．

しかし，データベースに対して連続的な領域を確保してそのまま転送する手法により，高速な移動を実現可能であるが，明らかな問題点もある．一般に，物理記憶中のデータはポインタを用いて特定の物理アドレスを参照しており，データベースに付加されているインデックスも物理アドレスを指し示すポインタを用いている．従って，前節の手法では，データベース（インデックスを含む）の移動先において，移動元に配置されていたアドレスとは異なる領域にデータベースが配置されるため，ポインタをそのまま移動するとデータベース中のポインタが誤ったアドレスを指すことになる．

これらの問題は，確保している領域の先頭アドレスからの相対ポインタなど，直接物理アドレスを参照しないポインタを用いることで解決する必要がある．

謝辞

本研究の全課程を通じて，懇切なる御指導，御助言と格別なる御配慮を賜りました大阪大学大学院工学研究科情報システム工学専攻 西尾章治郎教授に謹んで深謝の意を表します．

本論文をまとめるにあたり，貴重な時間を割いて頂き，懇切なる御指導と有益な御助言を賜りました大阪大学大学院工学研究科情報システム工学専攻 村上孝三教授，ならびに，大阪大学サイバーメディアセンター 下條真司教授に心より感謝致します．

大学在学中には講義・学生生活を通じて，在職後は職務を通じて，基礎的な学問ならびに学問に取り組む姿勢をご教授頂きました大阪大学大学院工学研究科情報システム工学専攻 白川功教授，藤岡弘教授，薦田憲久教授，赤澤堅造教授，電子情報エネルギー工学専攻 岸野文郎教授に厚く感謝申し上げます．

本研究を推進するにあたり，直接の御指導，御助言，御討論を頂きました大阪大学大学院工学研究科情報システム工学専攻 塚本昌彦助教授に衷心より感謝申し上げます．

本研究の大部分の過程において，ともに研究を進め有益な御指導，御討論を頂いた大学大学サイバーメディアセンター 春本要講師に深謝致します．

本研究の初期の過程において，ともに研究を進め御協力頂いた株式会社日立製作所 庄司加奈子氏に感謝の意を表します．

本研究の研究成果を実システムとして構築するにあたり，とりわけ御尽力頂いた大阪大学サイバーメディアセンター助手 秋山豊和助手，ならびに，キヤノン株式会社 酒井仁氏に感謝致します．

本研究に関する有益な御討論，御助言を頂きました近畿大学生物理工学部電子システム情報工学科 奥井順教授に心より感謝致します．

本研究を行なうにあたり，ともに研究を進め御協力頂いた大阪大学大学院工学研究科情報システム工学専攻博士前期課程 海貝明道氏，ならびに，京都大学大学院情報学研究科システム科学専攻 和田充史氏に感謝の意を表します．

本研究に関して有益な御討論を頂いた大阪大学大学院工学研究科情報システム工学専攻西尾研究室の諸氏に厚く御礼申し上げます.

最後に，研究生活を送るうえで，暖かい御支援と多大なる御理解を頂いた両親に心からの感謝と御礼を申し上げます.

参考文献

[1] ISO 8824: *Information Processing Systems — Open Systems Interconnection — Specification of Abstract Syntax Notation One (ASN.1)* (1987).

[2] ISO 8825: *Information Processing Systems — Open Systems Interconnection — Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)* (1987).

[3] ISO 8879: *Information Processing — Text and Office Systems — Standard Generalized Markup Language (SGML)* (1986).

[4] S. Acharya and S.B. Zdonik: "An efficient scheme for dynamic data replication," Technical Report CS-93-43, Brown University, Providence, RI (Sept. 1993).

[5] 秋山 豊和, 原 隆浩, 春本 要, 塚本 昌彦, 西尾 章治郎: "データベース移動を利用した分散データベースシステムの設計と実装," 情報処理学会第 55 回全国大会講演論文集 (3), pp. 469–470 (Sept. 1997).

[6] 秋山 豊和, 原 隆浩, 春本 要, 塚本 昌彦, 西尾 章治郎: "トランザクション系列に基づくデータベース移動を用いたデータベース再配置手法," 情報処理学会マルチメディア通信と分散処理ワークショップ論文集, pp. 153–160 (Nov. 1998).

[7] 秋山 豊和, 原 隆浩, 春本 要, 塚本 昌彦, 西尾 章治郎: "アクセス情報に基づくデータベース移動を用いたデータベース再配置手法," 情報処理学会論文誌, vol. 40, no. 1, pp. 188–196 (June 1999).

[8] 秋山 豊和, 原 隆浩, 塚本 昌彦, 西尾 章治郎: "データベース移動を用いた分散データベースシステムにおける複製を考慮したデータベースの位置管理手法," 情報処理学会研究報告, vol. 99, no. 61, pp. 327–332 (July 1999).

[9] T. Akiyama, T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio: "Database migration based on access skew detection," in *Proc. Int'l Symposium on Database, Web, and Cooperative Systems (DWACOS '99)*, pp. 65–72 (Aug. 1999).

[10] T. Akiyama, T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio: "Access skew detection for dynamic database relocation," *International Journal of Computing and Informatics (Informatica)*, to appear (2000).

[11] 秋山 豊和，海貝 明道，酒井 仁，原 隆浩，塚本 昌彦，西尾 章治郎: "DB-ManMos: データベース移動機能をもつ分散データベースシステムのためのモニリング機構," 情報処理学会マルチメディア，分散，協調とモーバイルシンポジウム論文集, 発表予定 (June 2000).

[12] H. Armbrüster: "The flexibility of ATM: supporting future multimedia and mobile communications," *IEEE Commun. Mag.*, vol. 33, no. 2, pp. 76–84 (Feb. 1995).

[13] S. Banerjee, V.O.K. Li, and C. Wang: "Distributed database systems in high-speed wide_area networks," *IEEE Journal on Selected Areas in Commun.*, vol. 11, no. 4, pp. 617–630 (May 1993).

[14] S. Banerjee and P.K. Chrysanthis, "Network latency optimizations in distributed database systems," in *Proc. 14th Int'l Conf. on Data Engineering*, pp. 532–540 (Feb. 1998).

[15] T.F. Bowen, G. Gopal, G. Herman, T. Hickey, K.C. Lee, W.H. Mansfield, J. Raitz, and A Weinrib, "The datacycle architecture," *Communication of the ACM*, vol. 35, no. 12, pp. 71–81 (Dec. 1992).

[16] M.H. Callender: "Future public land mobile telecommunication systems," *IEEE Personal Commun.*, vol. 1, no. 4, pp. 18–22 (1994).

[17] K.G. Carlberg: "A routing architecture that supports mobile end systems," in *Proc. IEEE MILCOM '92*, pp. 159–164 (1992).

[18] S. Chia: "The Universal mobile telecommunication system," *IEEE Commun. Mag.*, vol. 30, no. 12, pp. 54–62 (Dec. 1992).

[19] M. Daily, J. Gabrynowicz, S. Narumalani, K. Nygard, W. Perrizo, P. Ram, S. Reichenbach, G.A. Seielstad, and W. White: "Accessing earth system science data and applications through high-bandwidth networks," *IEEE Journal on Selected Areas in Commun.,* vol. 13, no. 5, pp. 793–805 (June 1995).

[20] R. Devine: "Design and implementation of DDH: A distributed dynamic hashing algorithm," in *Proc. 4th Int'l Conf. on Foundations of Data Organization and Algorithms (FODO)*, Chicago (1993).

[21] D. DeWitt, R. Katz, F. Olken, L. Shapiro, M. Stonebraker, and D. Wood: "Implementation techniques for main memory database systems," in *Proc. ACM SIGMOD '84*, pp. 1–8 (June 1984).

[22] H. Garcia-Moline, R.J. Lipton, and J. Valdes: "A massive memory machie," *IEEE Trans. Comput.*, vol. C-33, pp. 391–399 (May 1984).

[23] A. Guha, A. Pavan, J. Liu, R. Ajay, and T. Steeves: "Supporting real-time and multimedia applications on the mercuri testbed," *IEEE Journal on Selected Areas in Commun.*, vol. 13, no. 4, pp. 749–763 (May 1995).

[24] R. Hagmann: "A crash recovery scheme for a memory resident database system," *IEEE Trans. Comput.*, vol. C-35, pp. 839–843 (Sept. 1986).

[25] 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: "ASN.1 データベースシステムにおけるインデックス機構の比較," 情報処理学会研究報告, vol. 95, no. 147, pp. 17–24 (July 1995).

[26] 原隆浩，春本要，塚本昌彦，西尾章治郎: "ATM 環境におけるデータベース移動に基づくトランザクション処理手法," 情報処理学会研究報告，vol. 96, no. 11, pp. 49–56 (Jan. 1996).

[27] 原隆浩，春本要，塚本昌彦，西尾章治郎: "ATM 環境内で移動するデータベースの位置管理について," 情報処理学会研究報告，vol. 96, no. 68, pp. 111–116 (July 1996).

[28] 原隆浩，春本要，塚本昌彦，西尾章治郎: "データベース移動を用いた分散データベースシステムにおける並行処理制御について," 情報処理学会研究報告，vol. 96, no. 1, pp. 179–186 (Oct. 1996).

[29] T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio: "Database migration for transaction processing in ATM networks," in *Proc. 11th Int'l Conf. on Information Networking (ICOIN-11)*, vol. 1, pp. 1B-4.1–1B-4.2 (Jan. 1997).

[30] 原隆浩，春本要，塚本昌彦，西尾章治郎: "ATM ネットワークにおけるデータベース移動のためのデータベース位置管理手法," 電子情報通信学会論文誌 D-I, vol. J80-D-I, no. 2, pp. 137–145 (Feb. 1997).

[31] T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio: "Location management for migratory data resources in ATM networks," in *Proc. ACM Synposium on Applied Computing (ACM SAC '97)*, pp. 123–130 (Feb. 1997).

[32] 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: "データベース移動に基づく分散データベースシステム DB-MAN の性能評価," 情報処理学会第 54 回全国大会講演論文集 (3), pp. 3-243–3-244 (Mar. 1997).

[33] 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: "データベース移動を用いた ATM ネットワークにおけるトランザクション処理," 電子情報通信学会論文誌 D-I, vol. J80-D-I, no.6, pp. 505–513 (June 1997).

[34] 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: "データベース移動に基づく動的複製配置法," 電子情報通信学会技術研究報告, vol. 97, no. 160, pp. 107–112 (July 1997).

[35] T. Hara, K. Harumoto, M. Tsukamoto, S. Nishio, and J. Okui: "Main memory database for supporting database migration," in *Proc. 1997 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM '97)*, vol. 1, pp. 231–234 (Aug. 1997).

[36] T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio: "DB-MAN: A distributed database system based on database migration in ATM networks," in *Proc. 14th Int'l Conf. on Data Engineering*, pp. 522–531 (Feb. 1998).

[37] 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: "広帯域ネットワークにおけるデータベースシステムアーキテクチャ," インターネット技術研究委員会 (ITRC) シンポジウム, pp. 97–104 (Feb. 1998).

[38] 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: "データベース移動に基づく分散データベースシステムとその並行処理制御機構," 電子情報通信学会論文誌 D-I, vol. J81-D-I, no. 6, pp. 819–828 (June 1998).

[39] T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio: "Database migration: A new architecture for transaction processing in broadband networks," *IEEE Trans. on Knowledge and Data Eng.*, vol. 10, no. 5, pp. 839–854 (Sept./Oct. 1998).

[40] 原 隆浩, 塚本 昌彦, 西尾 章治郎: "WAN 環境に適したデータベース移動のスケジューリングアルゴリズムの提案," 電子情報通信学会データ工学ワークショップ論文集, CD-ROM (Mar. 1999).

[41] 原 隆浩, 春本 要, 塚本 昌彦, 西尾 章治郎, 奥井 順: "広帯域ネットワーク上のデータベース移動に基づく動的複製配置法," 電子情報通信学会論文誌 D-I, vol. J82-D-I, no. 8, pp. 1049–1058 (Aug. 1999).

[42] T. Hara, M. Tsukamoto, and S. Nishio: "A scheduling method of database migration for WAN environments," in *Proc. Brazilian Symposium on Databases (SBBD '99)*, pp. 125–136 (Oct. 1999).

[43] 原 隆浩, 塚本 昌彦, 西尾 章治郎: WAN 環境におけるデータベース移動のスケジューリング手法, 電子情報通信学会和文論文誌 D-I, vol. J82-D-I, no. 12, pp. 1369–1378 (Dec. 1999).

[44] T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio: "Dynamic replica allocation using database migration in broadband networks," in *Proc. International Conference on Distributed Computing Systems (ICDCS 2000)*, pp. 376–384 (Apr. 2000).

[45] T. Hara, M. Tsukamoto, and S. Nishio: "Database Migration in WAN Environments: How Can It Earn Good Performance?," in *Proc. International Conference on Database and Expert Systems Applications (DEXA 2000)*, to appear (Sept. 2000).

[46] K. Harumoto, M. Tsukamoto, and S. Nishio: "A database system based on ASN.1: Its system architecture and database programming language, " in *Proc. Int'l Symposium*

on Advanced Database Technologies and Their Integration (ADTI '94), pp. 160–167 (1994).

[47] 春本 要, 塚本昌彦, 西尾章治郎: "OODBMS を用いた ASN.1 データベースの実現とその評価," 電子情報通信学会論文誌 D-I, vol. J79-D-I, no. 11, pp. 975–983 (Nov. 1996).

[48] G. Herman, G. Gopal, K. Lee, and A. Weinrib: "The datacycle architecture for very high throughput database systems," in *Proc. ACM SIGMOD '87*, pp. 97–103 (1987).

[49] T. Imielinski and B.R. Badrinath: "Querying in highly mobile distributed environments," in *Proc. 18th Int'l Conf. on Very Large Data Bases (VLDB '92)*, pp. 41–52 (Aub. 1992).

[50] J. Ioannidis, D. Duchamp, and C.Q. Maguire Jr.: "IP-based protocols for mobile internetworking," in *Proc. ACM SIGCOMM '91*, pp. 235–245 (1991).

[51] T. Johnson and P. Krishna: "Lazy updates for distributed search structure," in *Proc. ACM SIGMOD '93*, pp. 337–346 (1993).

[52] S.M.H. Joseph and F.A. Ian: "Local anchor scheme for reducing location tracking costs in PCNs," in *Proc. ACM MOBICOM '95*, pp. 181–193 (1995).

[53] R. Kadobayashi and M. Tsukamoto: "Traffic-based performance comparison of mobile support strategies," ACM-Baltzer Mobile Networks and Nomadic Applicatons (NOMAD): *Topical Journal on Mobility of Sytems, Users, Data and Computing*, vol. 1, no. 1, pp. 57–65 (1996).

[54] N. Krishnakumar and A.J. Bernstein: "High performance escrow algorithms for replicated databases," in *Proc. 18th Int'l Conf. on Very Large Data Bases (VLDB '92)*, pp. 175–186 (Aug. 1992).

[55] T.J. Lehman and M.J. Carey: "A recovery algorithm for a high performance memory resident database system," in *Proc. ACM SIGMOD '87*, pp. 104–117 (May 1987).

[56] W. Litwin, M.-A. Neimat, and D.A. Schneider: "LH* – Linear hashing for distributed files," in *Proc. ACM SIGMOD '93*, pp. 327–336 (May 1993).

[57] G. Matsliach and O. Shmueli: "An efficient method for distributing search structures," in *Proc. 1st Int'l Conf. on Parallel and Distributed Information Systems (PDIS)* (1991).

[58] J.E.B. Moss: *Nested Transactions: An Approach to Reliable Distributed Computing*, MIT Press, Cambridge, MA (1985).

[59] 西尾章治郎, 塚本昌彦: "広帯域ネットワークにおけるマルチメディア情報ベース," 電子情報通信学会論文誌 D-II, vol. 79, no. 4, pp. 460–467 (Apr. 1996).

[60] T. Norp and J.M. Roovers: "UMTS integrated with B-ISDN," *IEEE Commun. Mag.*, vol. 32, no. 11, pp. 60–65 (Nov. 1994).

[61] J.P. Nussbaumer, B.V. Patel, F. Scaffa, and J.P.G. Sterbenz: "Networking requirements for interactive video on demand," *IEEE Journal on Selected Areas in Commun.*, vol. 13, no. 5, pp. 779–787 (June 1995).

[62] C. Perkins and P. Bhagwat: "A mobile networking system based on Internet Protocol," *IEEE Personal Communications*, vol. 1, no. 1, pp. 32–41 (1994).

[63] C. Perkins: "IP mobility support," IETF RFC2002 (1996).

[64] S. Rajagopalan and B.R. Badrinath: "An adaptive location management strategy for Mobile IP" in *Proc. ACM MOBICOM '95*, pp. 170–180 (1995).

[65] R. Rooholamini and V. Cherkassky: "ATM-based multimedia servers,"*IEEE Multimedia*, vol. 2, no. 1, pp. 39–52 (Spr. 1995).

[66] 齋藤 淳, 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: "ASN.1 データベースシステムにおけるデータ格納方式," 情報処理学会研究報告, vol. 95, no. 12, pp. 17–24 (Jan. 1995).

[67] 酒井 仁, 秋山 豊和, 海貝 明道, 原 隆浩, 春本 要, 塚本 昌彦, 西尾章治郎: "移動機能を有する分散データベースシステム DB-MANαの設計と実装," 電子情報通信学会データ工学ワークショップ論文集, CD-ROM (Mar. 1999).

[68] 酒井 仁, 秋山 豊和, 海貝 明道, 原 隆浩, 塚本 昌彦, 西尾 章治郎: "データベース移動に基づく分散データベースシステム DB-MANαにおける同時実行制御機構の設計と

実装," 日本ソフトウェア科学会プログラミングおよび応用のシステムに関するワークショップ論文集, URL: http://www.jaist.ac.jp/SPA2000/20000320/ (Mar. 2000).

[69] C. Severance, S. Pramanik, and P. Wolberg: "Distributed linear hashing and parallel projection in main memory databases," in *Proc. 16th Int'l Conf. on Very Large Data Bases (VLDB '90)*, pp. 674–682 (1990).

[70] 重野寛, 大島浩, 松下温: "インタネット上でホスト移動をサポートするプロトコル HMSP," 電子情報通信学会論文誌 B-I, vol. J80-B-I, no. 3, pp. 111-120 (1997).

[71] 庄司加奈子, 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: "広帯域ネットワークにおけるデータベース移動の実現について," 電子情報通信学会技術研究報告, vol. 96, no. 54, pp. 55–60 (May 1996).

[72] J. Sidell, P.M. Aoki, A. Sah, C. Staelin, M. Stonebraker, and A. Yu: "Data replication in Mariposa," in *Proc. 12th Int'l Conf. on Data Engineering*, pp. 485–494 (Feb. 1996).

[73] M. Stonebraker: "Concurrency control and consistency in multiple copies of data in distributed INGRES," *IEEE Trans. Software Eng.*, vol. 3, no. 3, pp. 188–194 (May 1979).

[74] M. Stonebraker, P.M. Aoki, R. Devine, W. Litwin, and M. Olson: "Mariposa: A new architecture for distributed data," in *Proc. 10th Int'l Conf. on Data Engineering*, pp. 54–65 (1994).

[75] Sun Microsystems, Inc., "XDR: External Data Representation Standard," RFC1014 (June 1987).

[76] F. Teraoka, Y. Yokote, and M. Tokoro: "A network architecture providing host migration trasparency," in *Proc. ACM SIGCOMM '91*, pp. 209–220 (1991).

[77] F. Teraoka, K. Uehara, H. Sunahara, and J. Murai: "VIP: A protocol providing host mobility," *Communication of the ACM*, vol. 37, no. 8, pp. 67-75, p. 113 (1994).

[78] R.H. Thomas: "A majority consensus approach to concurrency control for multiple copy databases," *ACM Transactions on Database Systems*, vol. 4, no. 2, pp. 180–209 (June 1979).

[79] M. Tsukamoto, R. Kadobayashi, and S. Nishio: "Strategies for query processing in mobile computing," T. Imielinski and H.F. Korth (Eds): *Mobile Computing*, Kluwer Academic Publishers, pp. 595–620 (1996).

[80] 海貝 明道, 酒井 仁, 秋山 豊和, 原 隆浩, 塚本 昌彦, 西尾章治郎: "データベース移動に基づく分散データベースシステム DB-MANαの性能評価," 情報処理学会研究報告, vol. 99, no. 94, pp. 79–84 (Nov. 1999).

[81] R. Vingralek, Y. Breitbart, and G. Weikum: "Distributed file organization with scalable cost/performance," in *Proc. ACM SIGMOD '94*, pp. 253–264 (1994).

[82] D.J. Wetherall, W.F. Stasior, J.F. Adam, H.H. Houh, M. Ismert, D.R. Bacher, B.M. Phillips, and D.L. Tenninhouse: "View station applications: Implications for network traffic," *IEEE Journal on Selected Areas in Commun.*, vol. 13, no. 5, pp. 768–778 (June 1995).

[83] H. Wada, T. Yozawa, T. Ohnishi, and Y. Tanaka: "Mobile computing environment based on Internet packet forwarding," in *Proc. Winter USENIX*, pp. 503–517 (1993).

[84] O. Wolfson and S. Jajodia: "Distributed algorithms for dynamic replication of data," in *Proc. ACM PODS '92*, pp. 149–163 (June 1992).