UNIX ワークステーションにおけるディスク・アクセス特性とディスク・キャッシュの考察†

畑下豊仁†† 志賀稔††
風間成介†† 渡辺治††

高度の応答性を要求されるワークステーションにとってディスク装置と主記憶装置のアクセス時間のギャップを克服することは重要な課題である．この課題の解決策の1つとしてディスクキャッシュが挙げられる．筆者らは UNIX ワークステーションの性能向上を目指して，ディスクキャッシュの導入を図るため，以下の実験を行った．ディスクアクセス特性をディスク I/O トレースデータから分析し，さらにこのトレースデータを基にディスクキャッシュによる効果をシミュレーションを用いて予測した．次にこの結果を用いて，ディスクキャッシュの仕様を確定し，その仕様に基づきディスクキャッシュをインプリメントし，実測により性能評価を行った．以上の実験結果から，シミュレーションによる予測が実測に良く一致すること，ディスクキャッシュによりディスク I/O 時間を約半分に短縮できることが確かめられた．さらに，Read/Write 比，順アクセスの占める割合，平均ディスク I/O 時間等のディスクアクセス特性，最適なディスクキャッシュ方式，現行システムの問題点などが明らかになった．これらのことは他のワークステーション，UNIX システムに対しても有益であると思われる．

1. はじめに

近年，画像処理，CAD/CAM，科学技術計算，オフィスオートメーション等の種々の用途に使われるワークステーション（以下 WS と記す）が注目を浴びている．WS はマンマシン・インタフェースを司るため，応答時間の短縮化が強く要求されている．システムの応答性を向上させるため，大型機やミニコンではディスクキャッシュを使用している例が見受けられるが[1),2)]，ディスクキャッシュの手法は，筆者らが対象としている WS の応答性の向上にも有効と思われる．なぜなら，この WS はディスクへの依存度が高い UNIX[3)]* を OS とし，巨大なアプリケーション・プログラムと大容量の画像ファイルをアクセスする必要があるためである．

一方，ディスクキャッシュの効果はシステムの状態，ファイル・アクセスの動特性に依存するため，実稼働時のディスクの利用状況をモニタリングし，そのデータを基にしたシミュレーションによりディスクキャッシュの効果を推定する方法が報告されている[4)~8)]．

本論文では，上記手法を用い UNIX・WS にディスクキャッシュをインプリメントした結果について報告する．さらに，過程において明らかになったディスクアクセス特性，ディスクキャッシュの一構成例およびディスクキャッシュの効果についての考察を述べる．

以後，2章では対象 WS の概要について述べ，3章ではディスク I/O トレースデータの採取と WS のディスクアクセス特性の解析結果について述べる．4章では採取したトレースデータを基に，ディスクキャッシュ・シミュレータによりディスクキャッシュの効果を推定し，ディスクキャッシュ仕様の決定を行う．最後に5章ではインプリメントしたディスクキャッシュの効果について考察する．

2. 対象 WS

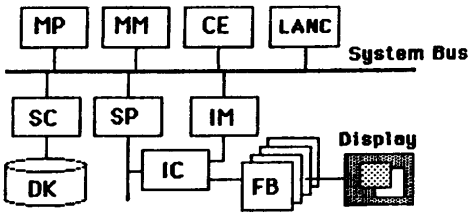
2.1 対象 WS の構成と利用形態

筆者らが測定対象とした WS はオフィスオートメーション用に開発されたもので[9)]，OS は UNIX SYSTEM Ⅲ にマルチウィンドウ機能，LAN により WS・サーバー間のファイルの高速転送を行う通信機能などを拡張したものを採用している．この WS の構成を図1に示す．UNIX 処理用の主プロセッサに 32 ビット・アーキテクチャ・マイクロプロセッサを使用し，主記憶は 1.25 Mbyte を実装している．フォーマット時約 66 Mbyte の容量を持つ8インチ固定ディスク装置を1台実装しており，ディスクは SMD (Storage Module Drive) インタフェースでディスク

† A Consideration on Disk Access Characteristics and Disk Cache on UNIX Workstation by TOYOHITO HATASHITA, MINORU SHIGA, SHIGEYUKI KAZAMA and OSAMU WATANABE (Information Systems and Electronics Development Laboratory, Mitsubishi Electric Corporation).

†† 三菱電機(株)情報電子研究所

* UNIX は米国ベル研究所で開発されたオペレーティング・システムの名称である．



MP: 主プロセッサ, MM: 主記憶, CE: 圧伸装置,
LANC: LAN コントローラ, SC: ディスク制御装置,
DK: ディスク装置, SP: 従プロセッサ,
IM: イメージメモリ, IC: イメージコントローラ,
FB: フレームバッファ

図 1 対象 WS の構成
Fig. 1 Composition of the target workstation.

制御装置とつながりシステムバスに接続されている.

利用形態は, 同時に使用するのは一人という環境を想定している. 主な処理は, 文書処理, 電子メールなどである.

2.2 ディスク管理

対象 WS で使用している固定ディスク装置の諸元を表 1 に示す. 1つの固定ディスク装置を3つの論理装置に分割して使用しており, ディスクの先頭部をルートファイルシステム(dk 1)*, ディスク中央部をスワップデバイス, 残りの領域をユーザファイルシステム (dk 2) として使用している. UNIX SYSTEMⅢのファイルシステムは 512 byte 長のブロックで構成され, 入出力処理はブロック単位で行われる[10]. 汎用計算機のように1回の I/O 要求で複数のセクタをアクセスすることはない. また, ブロックサイズは対象 WS では1物理セクタの大きさに等しい. スワップデバイスに対する入出力処理は 16 Kbyte 単位でディスクとの転送を行っている. セクタインタリーブは行っていない.

3. ディスクアクセス特性

3.1 測定方法

ディスクアクセス動作を解析するために, ディスク制御装置に植え付ける動作モード (Read/Write), トラック/セクタアドレス, データ転送長, 起動時刻, および転送終了時刻を順次採取した. このトレースデータを記録するためにトレースメモリを付加して, トレースメモリ領域を UNIX Kernel のデータ空間に配置した. また, 時刻の測定精度は 1 msec であり, この測定によるオーバヘッドは測定精度より十分小さく, 無視できる.

対象 WS では最も多用されると思われるアプリケーション・プログラムである『文書処理』を測定対象に選び, 文字と図形ベクトルを含む文書1ページを新規に作成する環境で測定を行った. この計測以外のプロセスは走っていない.

表 1 固定ディスク装置の諸元
Table 1 Disk unit specification.

ヘッド数	7
シリンダ数	573
セクタ数/トラック	34
セクタ容量	512 byte
ディスク総容量	約 66 Mbyte

* ファイルシステムとは, デバイスを論理的に扱うための構造のことをいい, スーパ・ブロック, i-リスト, データブロック領域より構成される.

3.2 ディスクアクセス特性の解析

測定したトレースデータを (1) Read/Write 比, (2) アクセス領域の連続性, (3) シーク動作, (4) 再利用セクタ, (5) ディスク I/O 時間の5点から評価を行った. ディスク制御装置の起動回数 (以下, ディスク I/O 回数と記す) の総数は 6581 回, そのうちファイルシステムに対するアクセスは 5290 回, スワップデバイスに対するアクセスは 1291 回であった. なお (3) 以外はスワップデバイスに対するアクセスによるデータを解析の対象からはずしている.

(1) Read/Write 比

Read/Write=7.3 であった. すなわち 88% はディスクリードであり, 読出しの比率が高いことが解る.

(2) アクセス領域の連続性

本論文では順アクセス長を物理的に連続した n 個のセクタを順にアクセスした時の順アクセス長を n セクタと定義する. UNIX では順アクセス長 n セクタ

表 2 順アクセス長の分布
Table 2 Distribution of sequential access length.

順アクセス長	度数	全アクセスに占める割合
1 sec	1415	26.7%
2 sec	335	12.7%
3 sec	132	7.5%
4 sec	41	3.1%
5 sec	31	2.9%
6 sec～10 sec	45	6.6%
11 sec～1/2 tr	8	2.0%
1/2 tr～1 tr	9	4.1%
1 tr～2 tr	8	7.8%
2 tr～ 3tr	2	3.5%
5 tr～10 tr	3	15.5%
10 tr～	1	7.6%

(sec はセクタ, tr はトラック)

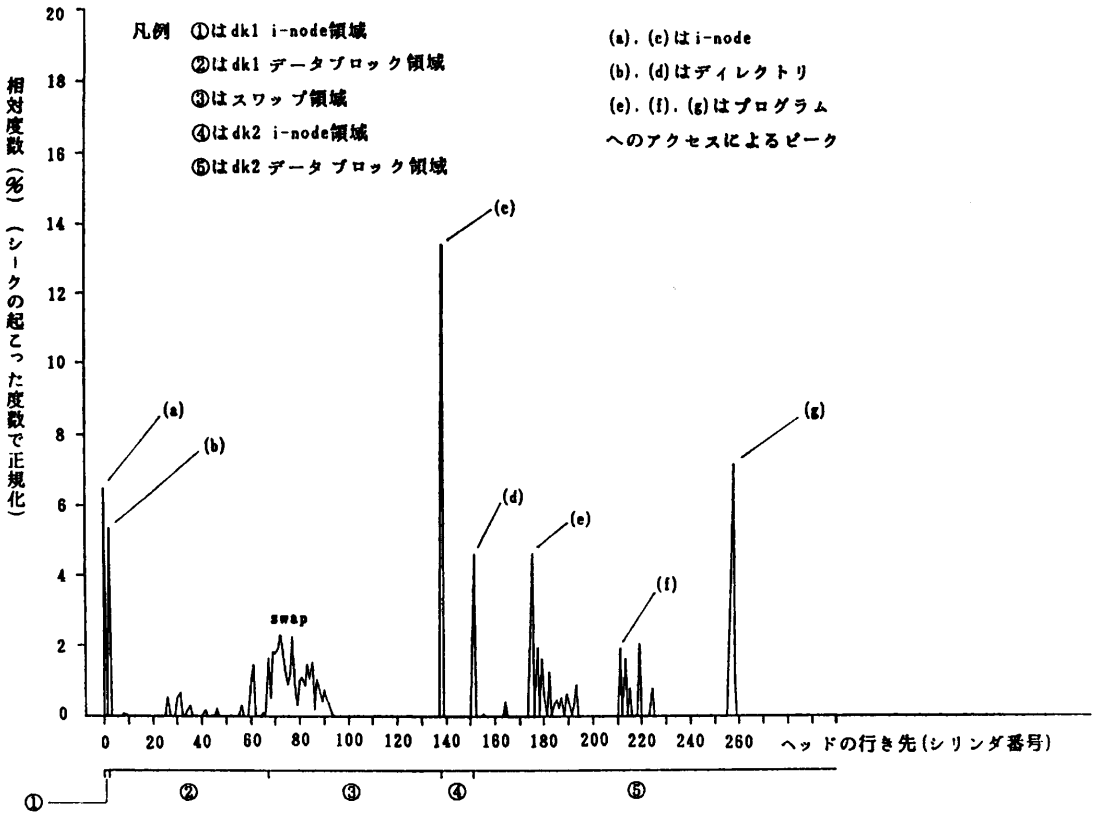


図 2 ヘッドの行き先の分布
Fig. 2 Distribution of head movement by seeking.

の時のディスク I/O 回数は n 回となる．表2に順アクセス長の度数分布を示す．順アクセス長が4セクタまでのものの全ディスク I/O 回数に占める割合は約5割，1/2 トラックを超えるものが約4割を占めることが読取れ，順アクセスの占める割合が高いと判断できる．順アクセス長が1セクタのうち約8割は i-node*・ディレクトリへのアクセスであり，順アクセス長が 1/2 トラックを越えるものはプログラムへのアクセスであった．

（3） シーク動作

図2にシーク時のヘッドの行き先の分布を，図3にシーク距離の分布を示す．なお，シークを伴うアクセスは全アクセスの約4割であった．i-node・ディレクトリへのシークは 30%，スワップ領域へのシークは 28%，ユーザファイルシステムのデータブロック領域へのシークは 35% をそれぞれ占める．図3からは上記領域間のシークに対応するピークが確認できる．これらから，i-node・ディレクトリへのシークの割合が高いと考えられる．

* ファイル管理情報が格納されている構造体のことをいう．

（4） 再利用セクタ

同一セクタに対する2度以上のアクセスの累計は全アクセス回数の約3割であった．

（5） ディスク I/O 時間

本論文ではディスク I/O 時間を次のように定義する．ディスク制御装置に起動をかけるディスク起動ルーチンに制御が移ってから，転送終了割込処理ルーチンが終了するまでの時間をディスク I/O 時間とする．図4にディスク I/O 時間の分布を示す．図から 16 msec の所に顕著なピークが見られる．16 msec はディスク1回転分の時間に相当し，物理的に連続した領域へアクセスする時に生じる回転待ちにより，このピークができると考えられる．また，平均ディスク I/O 時間は 19.7 msec であった．

4. ディスクキャッシュ・シミュレーションと考察

4.1 ディスクキュッシュ・シミュレータ

採取したディスク I/O トレースデータを用い，ディスクキャッシュ・シミュレータによるシミュレーシ

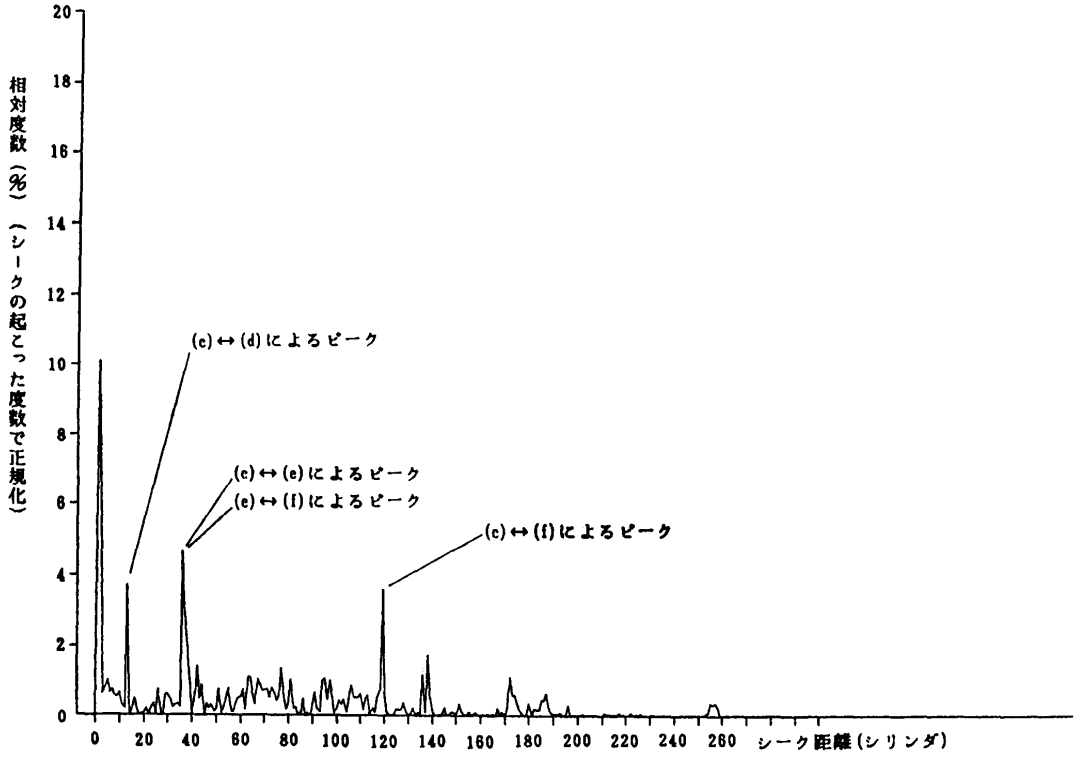


図 3 シーク距離の分布
Fig. 3 Distribution of seek distance.

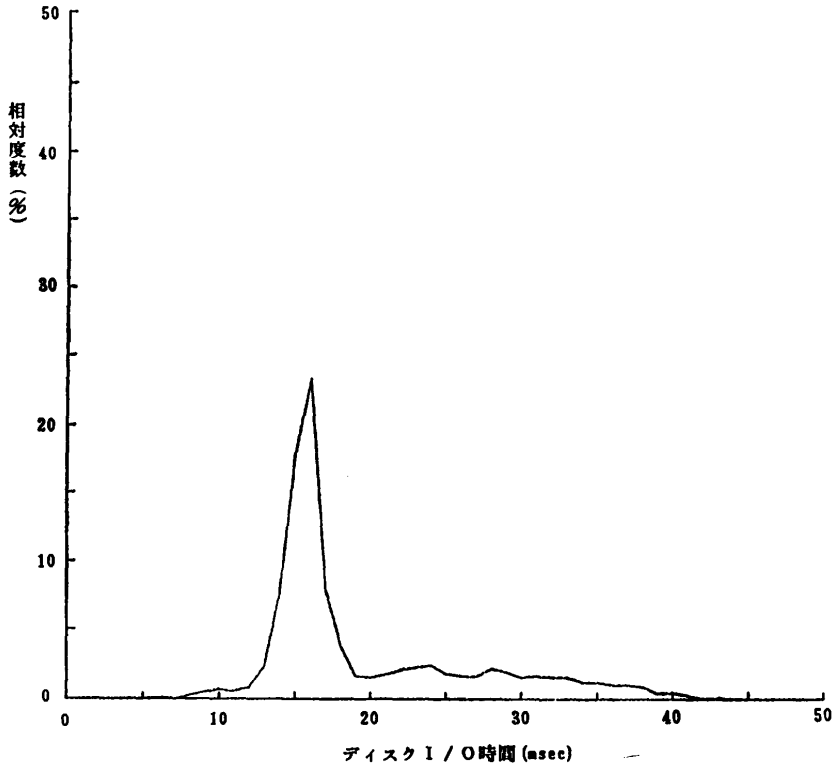


図 4 ディスク I/O 時間の分布
Fig. 4 Distribution of disk-I/O time.

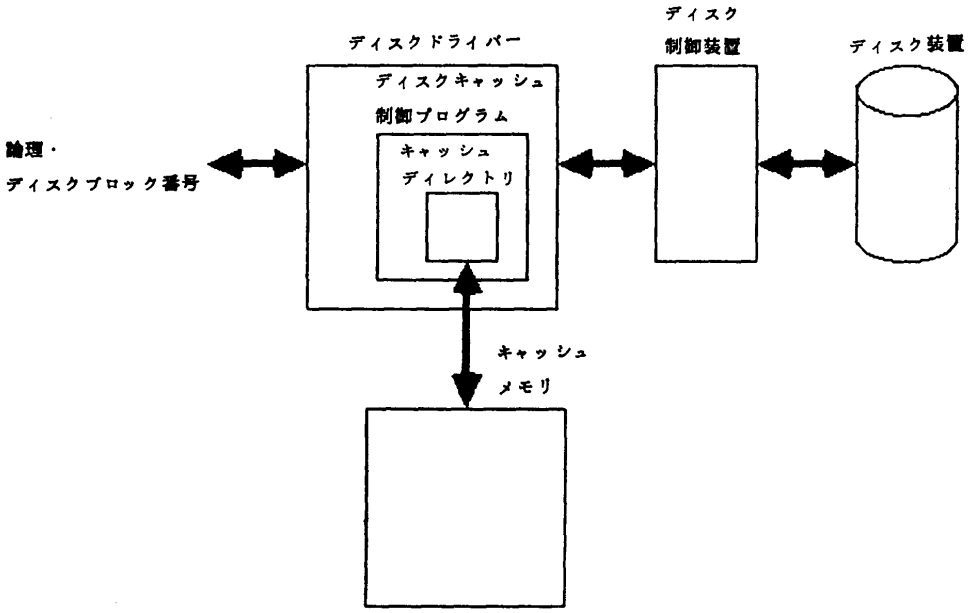


図 5 ディスクキャッシュ・モデル
Fig. 5 Disk cache model.

ョンからディスクキャッシュの効果を推定し，仕様を決定することを考える．このシミュレータがシミュレートするディスクキャッシュのモデルは図 5 のようになる．このようなディスクキャッシュを考えたのは以下の 3 つの理由による．

（1） 採取したトレースデータを有効に利用できる．

（2） インプリメント時に OS の改造量が少なく，容易にディスクキャッシュを実現できる．

（3） ハードウェアの制限より，主プロセッサがキャッシュの処理を行う．

キャッシュメモリは n セクタ (n=1, 2, 4, 8, 16, 32) を 1 ブロックとした単位で管理され，ミス時に要求ディスクブロックを先頭とする物理的に連続した n セクタをステージング* する．このようなステージングを行うのは 3.2 節から順アクセスの占める割合が高いと判断できるので，このステージング方式により高いヒット率を期待できるからである[11]．なお，本論文ではヒット率をディスク制御装置に植え付けるすべての入出力コマンドに対する，ヒットした入出力コマンドの割合と定義する（スワップデバイスに対する入出力処理は除く）．キャッシュとディスクとのブロックマッピングはフルアソシアティブ方式としている．

* 磁気ディスクからキャッシュメモリへデータをコピーすることをいう．

また，このシミュレータはスワップデバイスに対する入出力処理を対象としていない．その理由は次の 4 点による．

（1） スワップデバイスに対する入出力処理は 16 Kbyte 単位でディスクとの転送を行っておりかつ，スワップは集中して発生するので転送量が多く，本方式のように主処理プロセッサがすべてのキャッシュの処理を行う場合，UNIX の処理が止まってしまう可能性がある．

（2） 16 Kbyte 単位の転送だとディスク I/O 時間のうちデータ転送時間の占める割合が約半分程度になる．一方，キャッシュの処理をソフトウェア的に行う本方式ではデータ転送スピードの向上はあまり期待できないので，ヒット時の効果はファイルシステムに対するアクセスに較べて小さい．また，ミス時はキャッシュ処理によるオーバヘッド（キャッシュ↔主記憶間のデータ転送）が多大なものになる．したがって，ファイルシステムに対する入出力と較べてディスクキャッシュの効果をあまり期待できない．

（3） i-node や UNIX ファイルは 512 byte 単位でディスクとの転送を行っており，2 つの転送単位を管理するとキャッシュ管理オーバヘッドの増加につながる．

（4） メモリ管理方式は，将来スワップ方式から

ページング方式に移行すると思われる.

4.2 シミュレーション結果と考察

シミュレーションによりリプレースメント・アルゴリズム，書き込み方式，キャッシュサイズ，キャッシュ・ブロックサイズの4項目の比較を行い，ディスクキャッシュ方式を選択した.

(1) リプレースメント・アルゴリズムの比較

以下の3つのアルゴリズムについて比較を行う. なお，書き込み方式はライトバック方式としてシミュレーションを行った.

(a) LRU (Least Recently Used) アルゴリズム

(b) 疑似 LRU アルゴリズム (FINUFO (First In Not Used First Out) アルゴリズム[12]☆

(c) FIFO アルゴリズム

図6にリプレースメント・アルゴリズムの違いによるヒット率の変化を示す. 各アルゴリズムによるヒット率の差は最大 3.2% (キャッシュサイズ 512 Kbyte・ブロックサイズ 0.5 Kbyte の時) であり大きな差は見られない. この結果からリプレースメント・アルゴリズムはリプレースメント処理時間が小さく，LRU アルゴリズムと同等のヒット率が期待できる疑似 LRU アルゴリズムが適していると考える.

(2) 書き込み方式の比較

(a) ライトスルー方式と (b) ライトバック方式の2つの方式の比較を行った. 本論文では両方式を以下のような方式とする.

(a) ライトスルー方式：書き込み時，当該ディスクブロックがキャッシュ中に存在すれば，キャッシュに書き込みを行い，ディスクにも書き込む. 存在しなければ，キャッシュには何も行わず，ディスクにのみ書き込む.

(b) ライトバック方式：書き込み時，当該ディスクブロックがキャッシュ中に存在すれば，キャッシュにのみ書き込みを行う. 存在しなければ，ステージングを行い，キャッシュにのみ書き込みを行う. ディスクに対して書き込みを行うのは，汚れブロック* を置き換える時のみである.

ライトスルー方式では書き込み時，常にディスクに対して書き込みを行い，ライトバック方式では汚れブロックを置き換える時にディスクに対して書き込みを行うので，両方式の性能比較を行うには，ディスクに対して実際に何回 I/O を行うかで比較する必要がある.

リプレースメント・アルゴリズムは疑似 LRU とし，同様のシミュレーションを行いディスク I/O 回数を調べた. **図7**にキャッシュがない時のディスク I/O 回数を基準とした時のディスク I/O 回数の割合を示す. 図から最大で約 11 ポイント (キャッシュサイズ 1024 Kbyte，ブロックサイズ 1 Kbyte の時)，キャッシュサイズ 128 Kbyte 以上の場合は，ブロックサイズ 16 Kbyte の時を除いて，ライトスルー方式の方がディスク I/O 回数が多いことが解る. これから，ライトスルー方式の方が性能的に劣ると判断する. 11

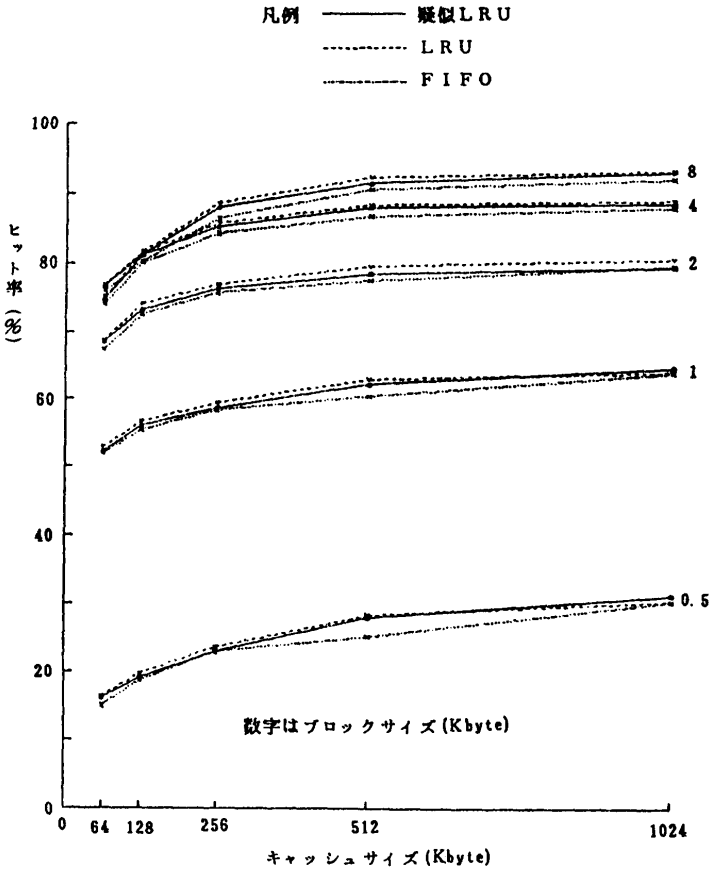


図6 置き換えアルゴリズムによるヒット率の比較
Fig. 6 Difference of hit-ratio by replacement algorithm.

☆ 付録参照

* 書き込み操作が行われたブロックのことをいう.

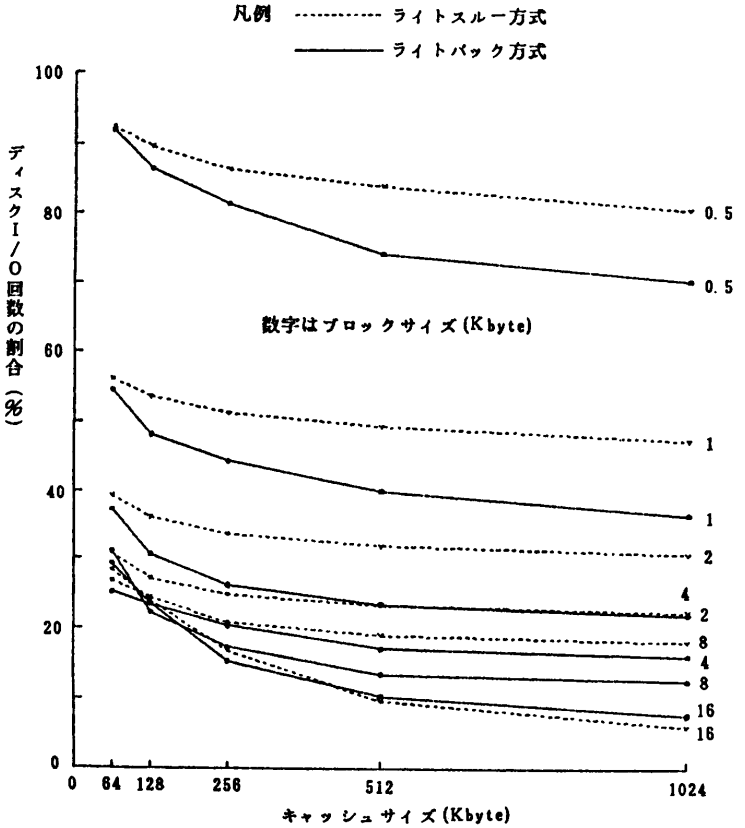


図 7 書き込み方式によるディスク I/O 回数の比較
Fig. 7 Difference of disk-I/O counts between write-through and write-back.

T：キャッシュ付きディスク I/O 時間
R_h：Read-hit 率
R_m：Read-miss 率
W_h：Write-hit 率
W_m：Write-miss 率
T_{se}：検索時間（ブロック数に比例）
T_{tr}：キャッシュ↔UNIX バッファ間転送時間
T_{rp}：置き換えブロック決定時間（ブロック数に比例）
T_s：シーク・回転待ち時間等
T_{dtr1}：ディスク→キャッシュメモリ間転送時間（ブロックサイズに比例）
T_{dtr2}：UNIX バッファ→ディスク間転送時間

1 例としてキャッシュサイズ 256 Kbyte，ブロックサイズ 4 Kbyte の時を挙げると，$T_{se}=1.0$，$T_{tr}=1.0$，$T_{rp}=2.0$，$T_s=20.6$，$T_{dtr1}=3.9$，$T_{dtr2}=0.5$（単位：msec）

以上はディスクドライバの処理ステップ数から試算している．上記例では，ディスク I/O 時間はキャッシュ無の場合の 47% に短縮できる（5.2 節参照）．図 8 にこの推定によるディスク I/O 時間を示す．図 6, 7, 8 からキャッシュの処理をソフトウェア的に行う本方式では，大型汎用機での実測[11]とは異なり，ヒット率の向上は必ずしもディスク I/O 時間の短縮に結びつかず，キャッシュ処理によるオーバヘッドを常に考える必要があることが解る．

この推定からインプリメントを行うキャッシュのサイズを考えると，同一ブロックサイズでは 1Mbyte～256 Kbyte で大きな差が見られないため，256 Kbyte で十分と思われる．

このキャッシュサイズでブロックサイズを考えると，ブロックサイズは 16 Kbyte の時，ステージング・コストの増大で 8 Kbyte より性能が劣化していることが解る．ブロックサイズを大きくとるとブロック数が減少し，アプリケーション・プログラムの違いによるアクセスパターンの変化に対してヒット率が不安定になると考えられ，性能劣化を招く可能性がある．以上からブロックサイズは 4 Kbyte を選択する．

ポイントの差は書き込みの比率にほぼ等しい．また，キャッシュサイズが小さくなると，汚れブロックの置き換えが増え，両方式の差が小さくなる．この減少はブロックサイズが大きいほど顕著である．本論文では，性能的にはライトバック方式よりも劣るが，事故によるキャッシュメモリ内容の消失時にデータ消失が起きないため信頼性が高く，ミス時の処理が単純なためインプリメントが容易なライトスルー方式を採用することにした．誤操作による電源断の可能性があるオフィス環境で使うことを想定した対象 WS では，ライトバック方式は採用できない．

（3） ディスクキャッシュの効果の推定

疑似 LRU リプレースメント・アルゴリズム，ライトスルー方式を採用した時のディスクキャッシュの効果を次式により推定する．

$$T = R_h * (T_{se} + T_{tr}) + R_m * (T_{se} + T_{tr} + T_{rp} + T_s + T_{dtr1}) + W_h * (T_{se} + T_{tr} + T_s + T_{dtr2}) + W_m * (T_{se} + T_s + T_{dtr2})$$

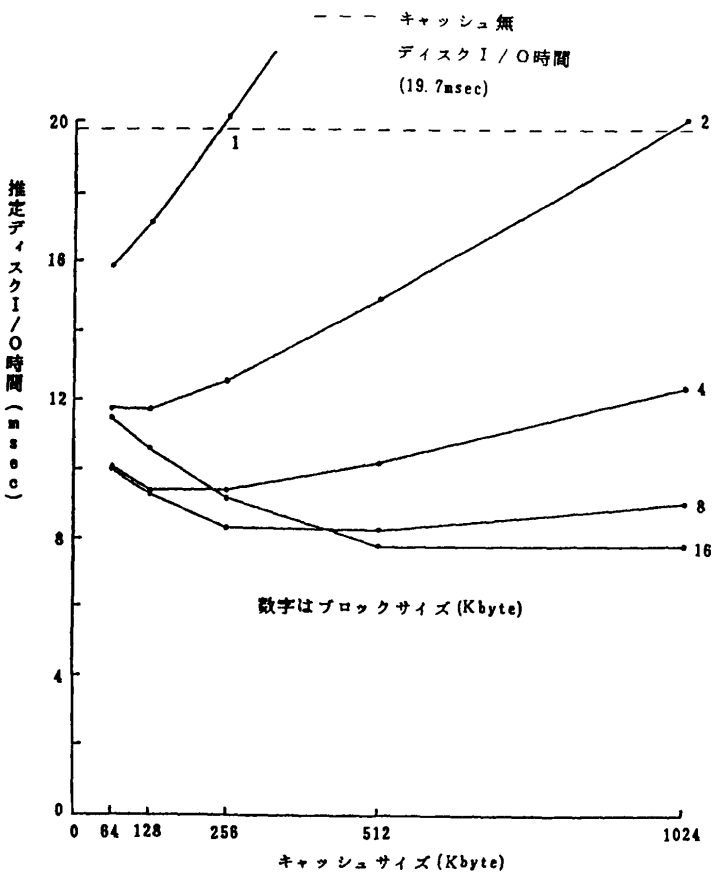


図 8 ディスクキャッシュの効果の推定
Fig. 8 Estimation of effects by disk cache, relation between cache-size and block-size.

5. ディスクキャッシュのインプリメントと考察

5.1 ディスクキャッシュのインプリメント

インプリメントしたディスクキャッシュの諸元は以下のとおりである.

(1) キャッシュの管理・実行は UNIX の処理を行う主プロセッサが行う.
(2) ディスクドライバにキャッシュを置く.
(3) リプレースメント・アルゴリズムは FINUFO アルゴリズム
(4) キャッシュとディスクとブロックマッピングはフルアソシアティブ方式
(5) ヒット・ミスの判定はソフトウェアによる逐次検索方式
(6) キャッシュ・ブロックサイズは 4 Kbyte
(7) キャッシュサイズは 256 Kbyte
(8) キャッシュはスワップデバイスに対する入出力処理を対象にしない.

インプリメントは, 従来のディスクドライバにキャッシュ管理部を追加する形で行った.

5.2 測定結果と考察

ディスクキャッシュを実装し, キャッシュ無の場合と同様のトレースデータを測定した. ディスクキャッシュの評価を (1) ディスク I/O 時間, (2) ビジー率と単位時間当たりの I/O 回数, (3) レスポンス時間の 3 点から行う. また, 総ディスク I/O 回数などの測定結果を**表 3** に示す. 総ディスク I/O 時間はキャッシュ無の場合の 45.5% に短縮でき, シミュレーションによる予測値 (47%) とよく一致し, 手法 6)〜8) の有効性が実測により確認できた.

(1) ディスク I/O 時間

図 9 にディスク I/O 時間の分布を示す. 2〜3 msec のピークはキャッシュ・ヒット時のものである. 1 msec の幅があるのは検索時間の差によるものと思われる. キャッシュ無の場合と比較して最大ディスク I/O 時間が大きくなっているのは, ステージング時に 4 Kbyte 単位に読み出すためであろう.

(2) ビジー率と単位時間当たりの I/O 回数

単位時間 (1 秒) に占めるディスク I/O 時間の総和の割合をビジー率と定義する. **図 10** にビジー率と単位時間当たりの I/O 回数の関係を示す. 比較のためキャッシュ無の場合を併記する. なお, スワップデバイスに対する入出力処理によるデータは除いてある. 図から以下の 6 点のことが解る.

表 3 測定結果
Table 3 Summary of measurement.

	キャッシュ無	キャッシュ有
ディスク I/O 回数	5290	5379
Read/Write	7.3/1	7.3/1
キャッシュリード回数	—	4009
ヒット率	—	75%
スワップ回数	1291	1314
平均ディスク I/O 時間	19.7 msec	8.8 msec

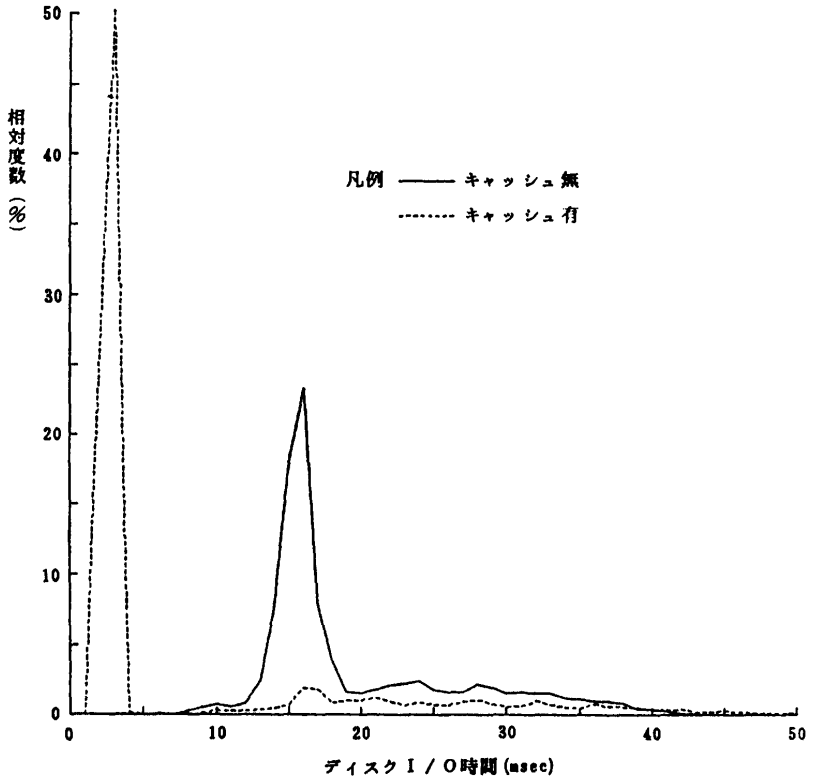


図 9 ディスク I/O 時間の分布
Fig. 9 Distribution of disk-I/O time with disk cache.

(a) キャッシュを実装することにより，ビジー率が全体に低下している．

(b) キャッシュ無の場合 20 msec/回の所に相関がある．これは平均ディスク I/O 時間に等しい．キャッシュ有の場合は 9 msec/回の所に弱い相関がある．キャッシュの場合相関が弱いのはミス時や書き込み時の処理時間とヒット時の処理時間の差が大きいためである．

(c) ディスクのスループット（単位時間当たりの I/O 回数）は最大値で比較した場合で約 1.3 倍に向上している．

(d) キャッシュ有の場合，75 回/秒・ビジー率 40 % の所に一集団を形成しているのが解る．これは順アクセスを行っている場合の集団で，ビジー率が40%と低くなっている．この要因は，UNIX の特徴である Kernel 空間↔User 空間のデータコピーを主プロセッサが行っているためである[13]．このことは，アセンブラステップ数による試算から確かめられた．したがって，この時主プロセッサはほぼフル稼働状態になっていると考えられ，この集団はシステム全体で見た時のディスクのスループットの限界点を示している．

(e) (d) から，ディスクキャッシュ専用の CPU を持ちキャッシュの処理とデータコピーを並列に行う．CPU・メモリを高速化してデータコピーを高速に行う．User 空間と直接物理 I/O を行う等の方法でこのデータコピーにかかる時間を無視できるようにすれば，キャッシュ無の場合のビジー率の最大値まではディスクを使用できると思われる．順アクセス時，スループットはビジー率に比例して増加するので，上記仮定が成立すれば，(d)項で述べた順アクセスによる集団は，約 170 回/秒，ビジー率 90% の位置に移動すると考えられ，ディスクのスループットをキャッシュ無の場合と較べて約 3 倍に向上できる可能性がある．

(f) キャッシュ無の場合約 58 回/秒がスループットの最大値であるが，これはビジー率が 90% 近くありデバイスによる制限と考えられる．これは明らかにディスクネックである．

(3) レスポンス時間

UNIX のユーティリティ・プログラムの time コマンドで測定したシステム全体の処理時間を表 4 に示す．キャッシュを持つことにより全処理時間 (real) を 13.7% 短縮することができた．sys (Kernel 処理

表 4 time コマンドによる全処理時間の測定
Table 4 Measurement of all processing time by time-command.

単位 (分：秒)

	キャッシュ無	キャッシュ有
real	5：57	5：08
user	23	23
sys	2：31	2：22

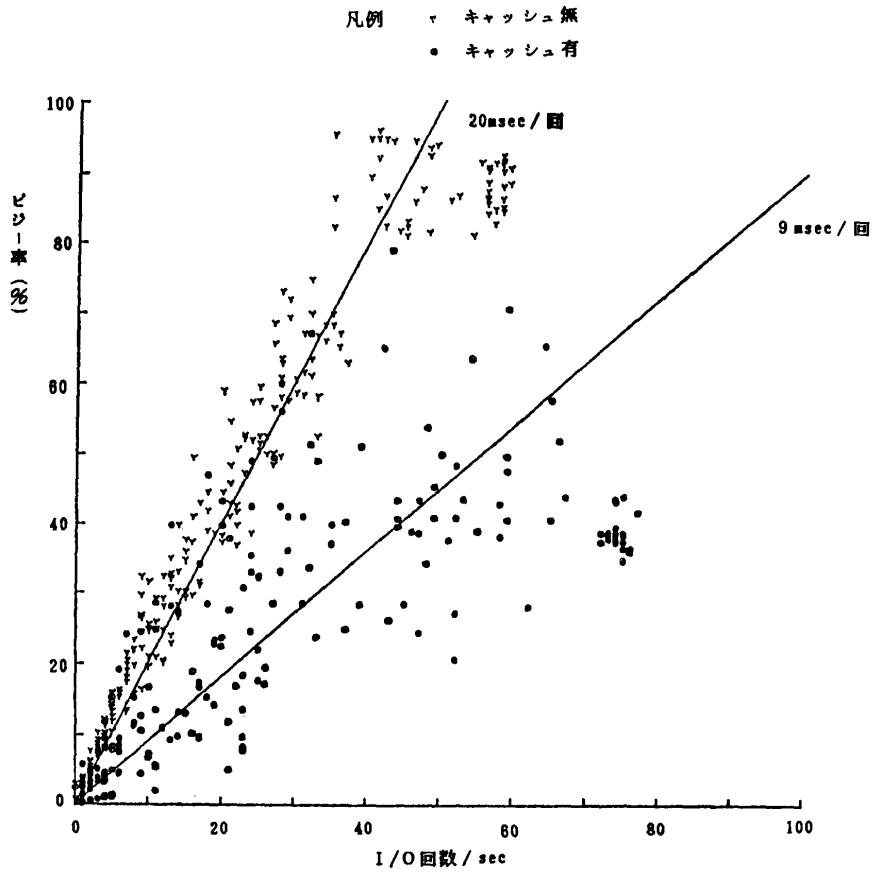


図 10 ビジー率と単位時間当たりの I/O 回数
Fig. 10 The relation between busy-ratio and I/O counts per second.

時間）の減少は，読み出し時のキャッシュ・ヒットの場合にはディスクに対して実際に I/O を行わないため，プロセススイッチが減少するためである．また，各操作コマンドのレスポンス時間でも平均 10% 程度の時間短縮が見られた．

6. まとめ

UNIX ワークステーションのディスク I/O トレースデータを採取し，ディスクアクセス特性の解析を行った．次に解析結果を基にしてディスクキャッシュのインプリメントを行い，性能評価を行った．

解析結果から，読み出しの比率が 88% と高いこと，順アクセスの占める割合が高いこと，平均ディスク I/O 時間は 19.7 msec であることなどが解った．

次にディスクキャッシュの性能評価の結果，シミュレーションによる効果予測と実測の結果がよく一致し，手法の有効性が確かめられ，平均ディスク I/O 時間をキャッシュ無の場合の半分以下の 8.8 msec にすることができた．また，キャッシュ実装時に順アクセスを行う場合，Kernel 空間↔User 空間のデータコピーがディスクのスループットのボトルネックになることを示し，このボトルネックを解決すればディスクのスループットをキャッシュ無の場合の 3 倍程度に向上できることを示した．さらに，全処理時間を 13.7%短縮でき，応答性が約 10% 向上することを確認できた．

本報告では UNIX SYSTEMⅢを OS とするワークステーションについて測定を行った．現在 AT & T は SYSTEM-V をリリースしているが，ブロックサイズが 1 Kbyte になった以外ディスクアクセス特性が大きく変わるような変更は行われていないので，本報告のディスクキャッシュは SYSTEM-V に対しても有効であると思われる．

今後の課題としては，UNIX SYSTEM-V でのディスクキャッシュの効果の測定，OA 向け WS では必須である大容量イメージデータの取扱い方法の検

討等が挙げられる.

謝辞 最後に，本研究の機会を与えていただいた，当社情報通信システム開発部山崎英蔵部長，ならびにご助言をいただいた三石彰純氏，伊藤均氏に深謝いたします.

参考文献

1) システム性能向上への効果が実証されたディスクキャッシュと半導体ディスク，日経エレクトロニクス，1985年3月11日号，No. 364，pp. 159-187 (1985).
2) 金子 悟：汎用大型機のディスクアクセスを高速化するディスクキャッシュの方式と性能評価，日経エレクトロニクス，1985年3月11日号，No. 364，pp. 205-232 (1985).
3) デビット F. ハイナント：UNIX システム・ベンチマーク，日経バイト，1984年11月号，No. 2，pp. 91-104 (1984).
4) 畑下豊仁，志賀 稔，風間成介，渡辺 治：ワークステーションのディスクアクセス特性の解析，昭和61年度電子通信学会総合全国大会講演論文集分冊7，1752，p. 84 (1986).
5) 畑下豊仁，志賀 稔，風間成介，渡辺 治：ワークステーションのディスクキャッシュの評価，昭和61年度電子通信学会総合全国大会講演論文集分冊7，1753，p. 85 (1986).
6) 管 隆志，上田尚純，田中千代治：ディスクキャッシュ装置のシミュレーションによる効果測定，情報処理学会論文誌，Vol. 22，No. 1，pp. 22-28 (1981).
7) 宮地泰造，三石彰純，溝口徹夫：階層型ディスク・キャッシュ・サブシステムの性能評価，信学論 (D)，Vol. J 68-D，No. 9，pp. 1609-1616 (1985).
8) 三石彰純，宮地泰造，溝口徹夫：バッファ内蔵型ディスク装置とその性能評価，信学論 (D)，Vol. J 67-D，No. 11，pp. 1301-1308 (1984).
9) 渡辺 治，水野忠則，坂下善彦，風間成介，伊藤均：高機能ワークステーション，三菱電機技報，Vol. 59，No. 12，pp. 31-35 (1985).
10) 石田晴久：UNIX，共立出版，東京 (1983).
11) 小畑征二郎，松沢 茂，宮崎正俊，表 俊夫，神山 典，高橋壮幸：入出力時間から見たディスクキャッシュの効果について，情報処理学会，オペレーティング・システム研究会，25-5 (1984. 12. 14).
12) 益田隆司：仮想記憶の制御方式，情報処理，Vol. 21，No. 4，pp. 369-377 (1980).
13) McKusik, M. K., Joy, W. N., Leffer, S. J. and Fabry, R. S. : A Fast File System for UNIX, *ACM Trans. Comput. Syst.*, Vol. 2, No. 3, pp. 181-197 (1984).

付 録

FINUFO (First In Not Used First Out) アルゴリズムについて

このアルゴリズムは以下のようなものである.

すべてのブロックを一つの円周上に並べ，参照されたブロックには参照ビットをセットする．ブロックの置き換えが発生した時，時計の針のような「現在の位置」を示すポインタを用い，ポインタを順に回転させ，最初に見つけた参照ビットが‘0’のブロックを置き換えの対象とする．新しく読み込まれたブロックは同位置にセットし，ポインタを回転させていく過程で参照ビットが‘1’にセットされているものは‘0’にリセットする.

(昭和61年7月 7 日受付)
(昭和62年4月15日採録)

畑下 豊仁

昭和36年生．昭和59年東京工業大学工学部電気・電子工学科卒業．同年三菱電機(株)入社．同社情報電子研究所システム・ソフトウェア開発部所属．ワークステーション・アーキテクチャの開発に従事．電子情報通信学会会員.

志賀 稔 (正会員)

昭和28年生．昭和47年直江津工業高等学校電気科卒業．同年三菱電機(株)入社．現在，同社情報電子研究所システム・ソフトウェア開発部に所属．分散システム，ワークステーション・アーキテクチャの開発に従事.

風間 成介 (正会員)

昭和24年生．昭和46年大阪大学工学部電子工学科卒業．昭和48年同大学院修士課程修了．同年三菱電機(株)入社．同社情報電子研究所システム・ソフトウェア開発部所属．現在ワークステーションの研究・開発に従事．電子情報通信学会会員.

渡辺 治 (正会員)

昭和18年生．昭和42年大阪大学工学部応用物理学科卒業．同年三菱電機(株)入社．同社情報電子研究所システム・ソフトウェア開発部所属．現在分散処理，ワークステーション等の研究開発に従事．電子情報通信学会会員.