データベース・システム　73－3
オペレーティング・システム　44－3
（１９８９．９．８）

ＤＢ統合型ＯＳの構造と性能評価

高崎繁夫*、平林元明**、原田　淳***、荒井弘治*、山口康隆*

*　（株）日立製作所　システム開発研究所
**　（株）日立製作所　ソフトウェア工場
***　日立ソフトウェアエンジニアリング（株）

中小型ＯＳは、小規模ホストマシンの管理から大規模計算機システムの部門プロセッサを管理するＯＳとして拡大しつつある。その鍵となるのがＯＳとＤＢ機能の統合であり、ここではＤＢ機能を統合したＯＳの開発が行われた。その目的はファイルシステムの一元化による使い易さと性能向上である。この性能向上を達成するため、大容量メモリを利用した仮想データ空間制御方式を開発した。本方式の特徴は、（１）ファイルデータのＬＲＵ管理によるメモリへの常駐化、（２）先読み、後書き機能による入出力の効率的使用、にある。以下、その方式と性能評価結果について述べる。

An OS integrated with DB facilities and its evaluation

Shigeo TAKASAKI*, Motoaki HIRABAYASHI**, Atushi HARADA***,
Kouji ARAI*, Yasutaka YAMAGUCHI*

*　Systems Development Lab.,Hitachi,Ltd.
1099,Ohzenji,Asao-ku,Kawasaki,215,Japan
**　Software Works,Hitachi,Ltd.
5030,Totsuka-cho,Totsuka-ku,Yokohama,244,Japan
***　Hitachi Software Engineering Co.,Ltd.
6-81,Onoe-cho,Naka-ku,Yokohama,231,Japan

Middle-scale Operating System will be widly used in department computer in large computer complex in addition to host computer of small-scale system. It is a key feature that DB facilities integrate into OS. We have designed an OS that contains some DB facilities in kernel component.
Purposes of the integration are improvements of usability and performance by means of unified relational file system. To achieve the purposes,we have developed a new method of Virtual Data Space Control(VDSA) that is based on a large amount of memory. Characteristics of the method are (1) maintaining of file data in memory controlled by LRU (Least Recently Used) strategy, (2) pre-read and deferred-write operation to perform efficient I/O.
This paper describes the new method and its performance results.

1. はじめに

近年の情報処理技術の発達に伴い汎用中小型計算機の分野も従来の小規模ホストコンピュータのイメージから、ネットワーク上の分散ホストや大規模システム内の部門プロセッサとして進化し始めている。このような環境では、従来のＥＤＰ要員中心からエンドユーザ中心の利用環境となり、使い勝手とプログラムの開発生産性、およびネットワークへの対応のし易さがキーポイントとなってくる。これを支える技術としてデータベース（ＤＢ）の導入は不可欠であり、リレーショナルＤＢをその基幹ＤＢに採用する方式が主流となってきた[1]。当社においても、このような新しい利用環境に対応し、革新するハードウェア技術を有効に利用するため中小型機用のオペレーティングシステム(ＯＳ)としてＶＯＳ Ｋ(Virtual-storage Operating System Kindness)を新たに開発した。このＶＯＳ Ｋは、ＨＡＡ(Hitachi Application Architecture)に準拠した最初のＯＳであり、その特徴は次の機能をシステムの基本構造に備えたことにある。

(1)リレーショナルＤＢ機能
(2)バッチと対話の統一操作環境を実現した対話制御機能
(3)ＯＳＩに準拠した拡張ＨＮＡ通信制御機能

本論文では、リレーショナルＤＢ機能と一体になったＶＯＳ Ｋの制御構造とその性能評価について述べる。

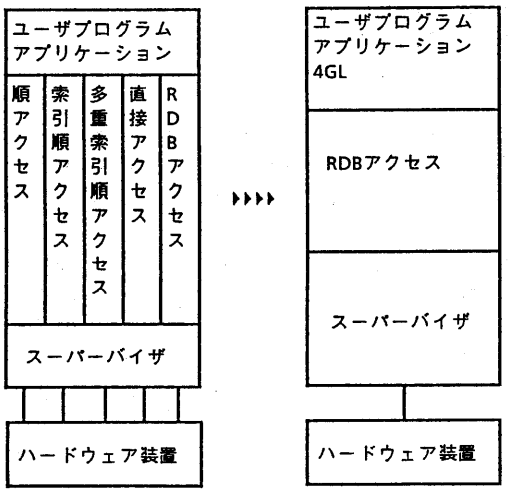

図1 アクセス法の統一

2. ＶＯＳ Ｋの制御構造と方式

2．1 リレーショナルＤＢ機能とＯＳの統合

リレーショナルＤＢは、データベースをフラットなテーブルの集合として扱い、そのデータの論理的な関係だけを用いてデータの蓄積および取り出しを行う機能である。ＶＯＳ ＫではこのリレーショナルＤＢを基幹ＤＢとして採用し、ディスク上のファイルをこれで一元管理している。その狙いは次の通りである。

（１）アクセス法の統一（図１）

従来システムでは大型ＯＳと同様に各ファイル編成に応じてアクセス法が用意され、利用者側がその編成法や用途に応じて選択する方法を取っていた。しかし、この方法では、各アクセス法で重複する機能もあり、利用者側の選択の負担になると共に、ＯＳ側にとってもファイルシステムの構造を複雑化する一因となっていた。ＶＯＳ Ｋでは、このような多種のアクセス法をＲＤＢアクセスに統一することにより、制御構造の簡略化とファイルシステムへの統一したアクセス環境を提供することを狙いとしている。ただし、高速処理が必要な順アクセスについては、順ファイル制御として専用インターフェイスを設けることで高速化を図っている。

（２）階層形ディレクトリ構造の実現（図２）

従来のＪＣＬ（ジョブ制御言語）方式では、ファイルアクセス時にはそのファイルが存在する装置やボリュームの指定が必要であったが、

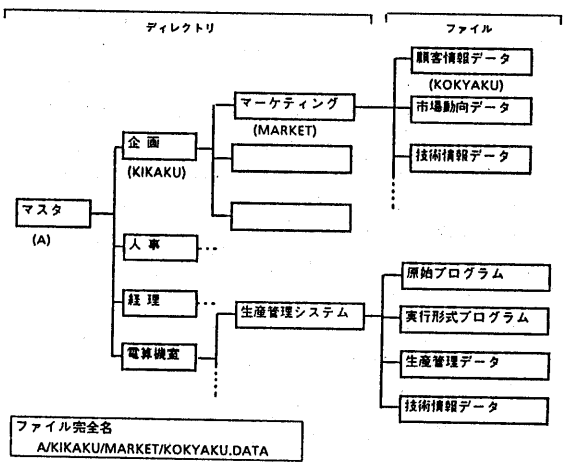

図2 階層形ディレクトリ

そのような繁雑な装置管理の無駄を省き、部門間でのファイルを論理的、かつ階層的に管理することが部門プロセッサ本来の役割に一致し、利用者の使い勝手向上になると考え階層形ディレクトリ構造を採用した。ここでは各管理部門やグループがファイルシステム内のノードとして扱われ、このノード名を完全または省略名（カレントノード名までをパスとして登録）で指定する。図2の例では以下のようになる。

- ﾏｽﾀ(A)：最上位のﾉｰﾄﾞ名(ﾌｧｲﾙｼｽﾃﾑ名)
- 企画(KIKAKU)：ﾉｰﾄﾞ名
- ﾏｰｹﾃｨﾝｸﾞ(MARKET)：ﾉｰﾄﾞ名
- 顧客情報ﾃﾞｰﾀ(KOKYAKU)：ﾌｧｲﾙ名
- ﾌｧｲﾙﾀｲﾌﾟ(例ではDATA)：ﾌｧｲﾙの種別ｺｰﾄﾞ

そして、このファイルはファイル完全名
.A/KIKAKU/MARKET/KOKYAKU.DATA でアクセスできる。

このような階層形ディレクトリ構造の利点は次の通りである。

①部門別にノードを設定することで部門内のファイル管理が容易となり、あたかも部門でボリュームを占有しているかのように使用できる。

②ノードを使用者別に分けることで、ファイル名が重複しても他ノードには影響しない。

③ノード以下の複数ファイルを一括して処理することができる。

この階層形ディレクトリ構造は、木構造であるためリレーショナルＤＢのインデクス管理と同様の方式で扱うことができる。

（３）ＯＳ制御の簡略化

ファイルの一元管理はアクセス法の統一と共に、物理的なデータ構造も統一することになる。例えば従来は、ディスク内のファイルの割り当てがユーザ指定になっていたため、ブロック長やスペース量など性能面を考慮したボリューム管理が可能であったが、その反面無駄な空きスペースの発生やファイル編成別の入出力チャネルプログラムの作成などデメリットとなる点も多かった。ＶＯＳ Ｋではリレーショナルデータベースの管理の下、ディスク上のデータブロックはすべてページ（４ＫＢ）単位に統一している。これによる利点は次の通りである。

①ディスクのスペース管理が従来のエクステント管理からビットマップの統一管理となり、無駄スペースを排除することができる。ただし、ビットマップ管理で問題となる連続領域の割り当ては領域予約の概念を入れて、３２ページ単位（図４におけるサブセル）に事前確保できるようにしている。

②入出力チャネルプログラムをファイル管理（もしくはデータ管理）側で作る必要がなく、ページ番号を用いた論理的なＩ／Ｏインターフェイスで実現できる。実際の入出力チャネルプログラムはＯＳの核ルーチン（ＳＰＶ）で作成する。

③Ｉ／ＯがＳＰＶ管理の元でページ単位のＩ／Ｏとなるため、ページングロジックを活用できる。

このようなリレーショナルＤＢによるファイルシステムの一元化により、従来に比べファイルシステムの構造が簡素化され、使用者側は物理的な配置や構造を意識する必要がなくなり使い勝手の良いファイルシステムを構築することができた。一方、性能面から見るとすべてのファイルアクセスがリレーショナルＤＢを経由することになり性能低下が懸案されるが、その対策としてファイルのページを主記憶上に積極的に保持し、入出力のオーバーヘッドを削減する仮想データ空間制御（ＶＤＳＡ：Virtual Data SpAce control）方式を採用したので、以下にその概略を示す。

２．２ 仮想データ空間制御方式[2]

仮想データ空間の概念を図３に示す。仮想データ空間はプログラム空間（仮想アドレス空間）とは別に設けられたデータ専用空間であり、ディスク装置当たり最大８ＴＢ（テラバイト）の領域を持つ論理的な空間である。この仮想データ空間内はすべてページサイズに区切られ、そのページアドレスはディスク上のブロック（ページ）アドレスに一致する。そして、主記憶（実アドレス空間）上に保持されたＶＤＳ領域をＶＤＳバッファと呼び、プログラム空間と同

様にＬＲＵ方式で管理する。このＶＤＳＡを使用した入出力処理動作を図３により示す。
①ユーザからレコード単位の入出力要求は、リレーショナルＤＢ制御（ＲＤＢＦ）によりページ単位にブロッキングおよびデブロッキングされる。
②ＲＤＢＦでは、共用バッファ中にこれらのページを一時的に保持しておき、入出力が必要と判断した場合には、ＳＰＶにそのデータタイプやアクセスタイプに応じた入出力要求マクロを発行する（主要マクロを表１に示す）。
③ＳＰＶでは、要求ページがＶＤＳバッファ上に保持されている場合には、ページ転送またはページテーブル書き換えで共用バッファとＶＤＳバッファ間のページ移動を行い、順入力の場合にはトラック単位の先読み動作を起動する。

このＶＤＳＡの狙いは、大容量化する主記憶を有効に利用してファイルの入出力性能を改善し、リレーショナルＤＢによるファイルの一元化を性能面から支援するものである。更にこのＶＤＳＡを有効に機能させるために、次のような制御方式を設けている。

（１）データの先読み制御

ファイルの順入力のような、次に入力すべきデータページが予見できるものについては、ユーザからの要求とは非同期に先読みの入力動作を行う。先読みの単位はトラックまたは指定ページ数である。

（２）データの後書き制御

出力要求についても遅延出力が可能なデータページについては、ユーザからの出力要求とは非同期に、入出力装置やチャネルの空きに応じてまとめ書きをする手段を備えている。

（３）一時ファイル（出力抑止）モードの設定

コンパイラ・ワークなど一時的ファイルについてはディスク出力を極力抑止し、入出力回数の削減を図っている。

（４）優先入出力処理

入出力処理でも先読み、後書き処理は低優先と考えられ、これらの低優先の入出力処理中にページングまたは実読み（同期型の入力要求）など高優先の入出力要求が発生した場合には、低優先処理を中断し、高優先処理をサービスするようにしている。

（５）データページのＬＲＵ管理

プログラム空間と同様に仮想データ空間のページもＬＲＵ管理を行い、参照頻度の高い（例えばＤＢのディレクトリやインデクス）ページは主記憶中に残るようにして、ヒット率を高めている。

この仮想データ空間を適用したディスクは図４および表２に示すような領域で管理される。同空間へのアクセスの最小単位であるページの上位概念として、サブセル、セル、ユニットおよび論理ディスクが存在する。仮想データ空間は論理ディスク単位に設けられ、基本的には１仮想データ空間＝１（論理）ディスクである。

３．　性能評価

３．１　評価ツール

ＶＯＳ Ｋは新規開発のＯＳであり、また上記ＶＤＳＡのような新しい制御方式を採用していることから、完成したＯＳが所定のハードウェア性能を充分に引き出しているか否かは重要な関心事項であり、性能評価が必要とされた。そのため、設計段階での方式選択を主にしたデザイン評価[3]とは別に、システムが実際に動き始めるテスト段階から性能データの収集を行い、問題箇所については迅速な対策を取れるようにした。この実システムをベースにした性能評価では次のツールを使用した。

（１）命令トレーサ

計算機の実行する命令を逐一追跡、収集するプログラムであり、オーバーヘッドはかかるがシステムの実際の動きを収集できることからシステムの動作解析や動的ステップ数測定には強力なツールである。ＶＯＳ Ｋのアーキテクチャは、大型ＯＳと同じＭアーキテクチャ（３１ビットアドレッシングのＭ／ＥＡモード）を採用しているために、大型ＯＳでの開発支援ツールＯＳＴＤ（Operating System Test Driver）[4]と、このＯＳＴＤに用意されている命令トレーサ[5]を使用することができた。この命令トレーサの利用によりＯＳマクロ別の動的ステ

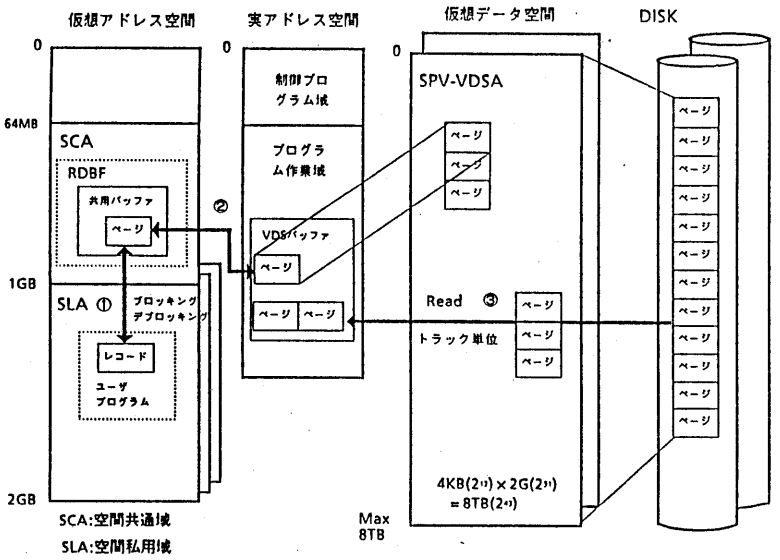

図3 仮想データ空間の概念図

表1 VDSA主要アクセスマクロ

マクロ名	動作	バッファリング	同期/非同期	主な用途
DCELSRED	指定ページ(連続)のVDSバッファ経由の読み込み	○	同期(実読み) 非同期(先読み)	順ファイルの高速アクセス
DCELRRED	指定ページ(不連続)のVDSバッファ経由の読み込み	○	同期(実読み) 非同期(先読み)	順ファイル以外のファイル入力
DCELDRED	指定ページの直接読み込み	×	同期	直接アクセス用
DCELBWRT	指定ページのVDSバッファ経由の書き出し	○	非同期	ファイルのデータ部出力
DCELTWRT	指定ページのVDSバッファ経由の書き出し	○	同期	ディレクトリ出力
DCELDWRT	指定ページの直接書き出し	×	同期	ジャーナル
DCELWWRT	指定ページのVDSバッファへの書き出し	○	--	一時ファイル
DCELFLUS	出力保留ページの一括出力	--	同期	非同期出力の確認

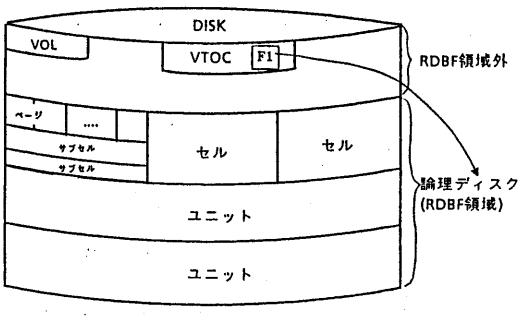

図4 DISKの領域管理

表2 DISK領域の管理単位

区分	大きさ	目的
ページ	4096B	仮想データ空間へのアクセス単位
サブセル	32ページ =128KB	スペース予約時の予約単位 予約しなければページ単位で自動増分
セル	256ページ =1MB	仮想データ空間の構成単位 仮想データ空間=nセル
ユニット	64セル =64MB	スペース管理上の概念、1ページで領域管理情報を格納出来る最大の大きさ
論理ディスク	nセル	仮想データ空間のオープン単位 (VTOC中のF1ラベルで管理)
ディスク	n論理ディスク +ラベル	物理ディスク

ップ数など基本性能単価を得ると共に、性能上の問題点の早期対策と目標性能達成の確認が可能になった。

（２）ソフトウェア・モニタ

核ルーチンの一部として動き、システムの内部情報をイベント発生毎、または時間間隔で収集するソフトウェア・モニタは、多少のオーバーヘッドは介在する（通常５%以下）が、システム固有な性能情報を収集できることから、命令トレーサとは異なる側面で性能評価には有効な手段である。ＶＯＳ Ｋにおいても０Ｓの開発と同期して、このソフトウェア・モニタの開発を進めていった。収集する情報については、ＣＰＵやタスク、Ｉ／Ｏなどの一般情報以外に特徴的な情報としてページカラー・マップ情報がある（図５参照）。 このページカラー・マップ情報は、主記憶ページの割り当て情報をビジュアル化したものであり、サンプリングした全ページの割り当て情報から色コードを与え、ＷＳ画面上に表現するものである。例えば以下のような色コードの割り当てとなる。

・赤色：ＳＰＶの固定ページ
・青色：ＳＰＶ以外のユーザページ（ページング対象のページ）
・黄色：データ（ＶＤＳ）ページ
・水色：固定化（Ｆｉｘｅｄ）ページ
・緑色：空きページ

このページカラー・マップの狙いは、次の通りである。

(a)メモリ依存型のＯＳ構造（特にＶＤＳＡのサポート）のため、メモリの利用状況を把握することが性能評価の重要ポイントとなる

(b)単に数値でなく、動きのあるビジュアル表現にすることで認識力を高める

(c)従来の合計値では判断できなかったローカルな情報が得られる

このページカラー・マップでの問題点は、メモリが増大すればする程大きくなる情報収集と表示時間にあるが、測定負荷に応じてサンプリング幅を増やし、また表示は高精細ディスプレイのＷＳを用いることで対処した。

以上の性能評価ツールを製品開発に同期して適用することで問題点の製品への早期フィードバックを図った。

３．２ 性能評価例

従来システム（ＶＯＳ０／ＥＳ）との性能比較として、順ファイルの入力時間を図６に示す。性能をレコード当たりの処理時間で表すと、従来システムでは次の式で近似できる。

$$y = \mathrm{Max}\{S_1\alpha + S_2(1-\alpha), t_1(1-\alpha)\} \quad \text{---(1)}$$

ここで、S_1：ﾊﾞｯﾌｧにﾋｯﾄした時のCPU時間
S_2：ﾊﾞｯﾌｧにﾋｯﾄしなかった時のCPU時間
α：ﾊﾞｯﾌｧﾋｯﾄ率(1-ﾚｺｰﾄﾞ長/4096)
t_1：1ﾍﾟｰｼﾞ入力時のI/O時間

である。

つまり、従来システムではＣＰＵとＩ／Ｏがページ単位で並列動作するため、レコード当たりの処理時間は処理の長い方に隠れてしまうことになる。

一方、新システム（ＶＯＳ Ｋ）では、メモリ上にある場合にはメモリ内の転送で済み、メ

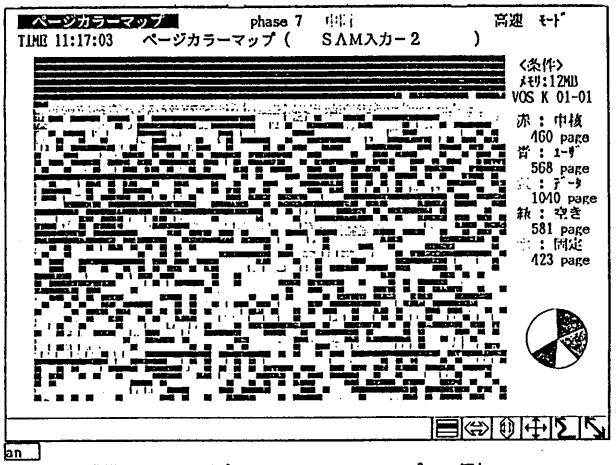

図5 ページ･カラー･マップの例

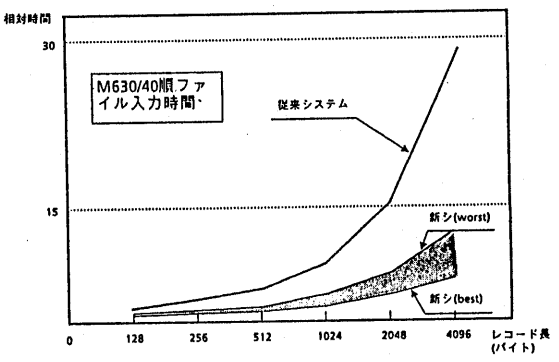

図6 従来システムとの性能比較

モリ上にない場合にはトラック単位の一括入力となるため、次に近似できる。

$$y = S_1\alpha + S_2(1-\alpha)\beta + t_2(1-\alpha)(1-\beta) \quad --(2)$$

β :VDSAでのヒット率

t_2 :1トラック入力時のI/O時間

ただし、ＣＰＵ時間を表すS_1，S_2はＯＳが異なると動的ステップ数が異なるため同一の扱いにはできないが、図６では命令トレーサによる測定結果をベースにしてS_1，S_2を計算している。また、ＶＤＳＡでのヒット率βについては、１トラック当たり６ページ入り、トラックの切れ目で次のトラックへの入力要求を起動しているため、メモリ中に全くない場合でも$\beta = 5/6$となるが、これを図６ではＷｏｒｓｔケースとして表わしている。ファイルのページがすべてメモリ中にある場合（例えばファイルの出力後に入力ジョブを実行する場合）には$\beta = 1$となり、ＣＰＵだけの処理で済むためＢｅｓｔケースとして表わしている。これにより従来システムではレコード長が長くなればなる程、入出力装置側の性能に依存してくるが、新システムではトラック単位の一括入力とＬＲＵ管理によるファイルのメモリ上での保持によりヒット率が高められ、性能向上が図られていることがわかる。これらの結果は実測結果とも一致している。

４．　終りに

ＤＢとＯＳを融合したシステムの一例としてＶＯＳ Ｋの方式を示した。その特徴はファイルシステムの統一化による操作性の向上と、それを性能面から支援するための仮想データ空間制御方式の実現にある。この仮想データ空間方式はソフトウェアで実現しており、ハードウェア支援に較べて柔軟な対応が可能になっているが、今後も性能面については更に強化を図っていく所存である。

参考文献

[１] 細田：ＲＤＢを軸に簡易化進む小型汎用機ＯＳ、日経コンピュータ、Ｎｏ．１９４、ｐｐ．９３－１０１（１９８９）

[２] 荒井、佐藤他：中小形システムＶＯＳＫ（２）－基本制御方式－、第３８回情報処理学会全国大会、６Ｎ－２、１９８９

[３] FERRARI,D.: "Computer Systems Performance Evaluation" ,PRENTICE-HALL,INC. (1978)

[４] 久保、吉澤他：汎用テスト支援システムＯＳＴＤ（１）－概要－、第３３回情報処理学会全国大会、７Ｖ－４、１９８６

[５] 高崎、大町：仮想計算機技術を利用した命令トレーサの開発、第２２回情報処理学会全国大会、１９８１