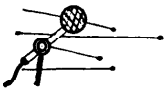
講　演



今後10年間のコンピュータ・サイエンスの研究†

鈴　木　則　久††

ご紹介いただきました鈴木でございます．今日はこれから10年間において，計算機科学の分野でどのような研究が重要であるかという予測をいたしてみたいと思います．

どんな研究が重要になるかということは，5年，10年先の計算機，技術および需要を考えていかなければならないと思います．

まず計算機科学が，物理の研究あるいは化学の研究などと違っていることは，計算機科学というものは人間のつくった計算機というものを研究する，あるいはそれを使って研究するものだということで，ハードウェアの進歩，テクノロジの進歩に非常に密接に関係しています．そのためにハードウェアの進歩によって研究する必要のなくなる分野も出てきますし，一方ハードウェアの進歩によって新しく生まれ，非常に重要になる研究もあります．ですから，これから5年先，10年先のテクノロジの進歩はどうか，ハードウェアはどんなものが出てくるか，コンピュータシステムはどんなものを使うようになるかということは，将来の研究を予測するのに重要な前提になります．

過去の研究をみますと，ハードウェアの進歩によって不要になった研究もあるわけです．たとえば30年前ですとドラム・コンピュータというものがありました．そのころのソフトウェアの研究といいますと，ドラムの上のどこに命令を置くのが最適かというようなことでした．命令の実行時間とドラムのスピンの時間を考慮して，ウェイトなしに命令が実行できるような命令配置を考えることでした．また4，5年前には，オプティカル・ディスクがはやり始めました．当時は，オプティカル・ディスクとしてはライトワンス型が主要であると考えられていましたので，そういうライトワンスのオプティカル・ディスクを使って読み書き自由にできるディスクを作るには，どうするかというような研究も行われておりましたけれども，現在ではだれも研究対象としてはおりません．

一方，ここ10年間に生まれて非常に重要な研究対象になりましたものにユーザ・インタフェースがあります．これはパソコンの出現，特にビット・マップ・ディスプレイの進歩によって，人間に使いやすいコンピュータが手軽に入手できるようになり，いかに使いやすいソフトを作るかという研究が大変盛んになりました．また，ローカル・エリア・ネットワーク，および光通信の進歩により，分散システムが大変重要な研究分野になってきています．あるいは一方，LSIの進歩によって大規模並列コンピュータというものが実際的なものになってきたので，それのソフトウェア，およびアーキテクチャの研究が大変重要になってきています．

1. テクノロジ進歩の予測

それではこれから10年，コンピュータ用のキー・テクノロジが，どう進歩するだろうかみていきます．

まず，演算素子としてはやはり基になるものは，なんといってもシリコンであってCMOSであろうと予測します．

シリコンCMOS論理回路では，何がスピードの重要な要因となるかといいますと，演算スイッチ回路のスピードと，配線の遅延があるわけですね．演算スイッチ回路のスピードは，LSIの初歩の本を見ていただくとわかりますけれども，ソースからドレーンまでの電子の移動時間できまる．これは $\tau=\frac{L^2}{\mu V_{ds}}$ という式で表せる．すなわち電子が移動する時間はゲートの二乗に比例する．MOSの論理スイッチ回路は，チャネル幅が半分になるとほぼ4倍速くなるだろうと考えられます．それに最近のように高温度の超伝導物質が盛んに発明されてきますと，配線による遅延も無視できるようになってくるということで，ますますCMOS，マイクロプロセッサの全盛時代が到来するだろうと思われます．

それでは，MIPS値から将来のCMOSマイクロプ

† 第34回 全国大会特別講演
（昭和62年3月18日　日本大学理工学部習志野校舎）
†† 日本IBM

ロセッサをみましょう．現在，8 MIPS，10 MIPS のマイクロプロセッサが市場に出始めております．88年ごろには0.7ミクロンのチャネル幅のマイクロプロセッサが市場に出回るようになるでしょう．そうすると，ほぼ 20 MIPS，91年には 0.5 ミクロンで 40 MIPS．10年後の1990年代の中ごろには，100 MIPS シングル・チップ・マイクロプロセッサが出てくるだろうと予測されます．

一方，メモリ・チップも今まで3年ごとに4倍の容量になっておりますので，1990年代の中ごろには，64メガビットの DRAM が普及するだろうと予測されます．

それから計算機を構成する重要な要素として2次記憶装置があります．現在は磁気ディスクですけれども，この分野でもいくつも新しいテクノロジの可能性が考え出されていて，密度の高速の進歩が考えられています．たとえば現在先端のテクノロジとしては薄膜ヘッド，あるいは薄膜ディスクが使われております．

これが全面的に取り入れられてくるとか，あるいは垂直磁気が実用化してくるとか，あるいは磁気ヘッドも大変性能のいいマグネトレジスタンス・ヘッドが実用化してくるとか，あるいはオプティカル・ストレッジもアクセス速度，データ・レートも進歩するというようなファクタが予測されまして，これらを考えて少なくとも10年後には現在の 100 倍ぐらいの容量の進歩が予測できます．

また，現在使っているようなビット・マップ・ディスプレイがどのくらいのものになりますかといいますと，現時点で普通に買えるものは 1000×1000 で，特殊な用途には 3000×3000 のものが得られます．たぶん10年後には3000×3000が普通のものになって，特殊用途には 1万×1万 ぐらいのものが実用化するでしょう．これらもリクイッド・クリスタルでつくられるでしょう．

2. システム進歩の予測

テクノロジの予測はこれぐらいにして，次にはシステムの予測をいたしましょう．

これから 10 年後にはどんなコンピュータ・システムが現れるかと申しますと，まず，パーソナル・コンピュータというと安いものでは 10 万円ぐらいで買うわけですけれども，10年後に 10 万円ぐらいで買えるようなパーソナル・コンピュータには，32 ビット・マシーンで演算速度から言うと 10 MIPS，主記憶は 100 メガバイト，2次記憶として1ギガバイト，そしてディスプレイとしては 3000×3000 というようなディスプレイをもつようなものがラップトップとしても，机上用としても使われるようになると思われます．

一方，ミニコンピュータとかワーク・ステーションの性能ですけれども，ワード・サイズとしては 32 ビット，あるいは48ビット，演算速度としては40からあるいは 1000 MIPS になります．1000 MIPS と申しますのは，これはマイクロプロセッサ1台でつくられるのではないということです．そしてディスクは 10 ギガバイト程度のものが使われるようになる．

ここで特筆すべきことはパラレル・コンピュータが現実のものになります．10年後には 40 MIPS のシングル・チップ・マイクロプロセッサを10から20台使い，シェアード・メモリのマルチプロセッサが，普通にワーク・ステーション，あるいはミニコンピュータの中に取り入れられるだろうと考えられます．

そうすると，メインフレーム・コンピュータというものはどうなるのでしょうか．要するに皆さんがローカル・エリア・ネットワークに付けて使うワーク・ステーションが 1000 MIPS ということですから大変な速度になるわけです．

一方，メインフレームはなかなか速くならない．それでは，ミニコンピュータが出てきたからメインフレーム・コンピュータは果たしていらなくなるだろうかということですけれども，メインフレームは非常にリライアブルで，非常にセキュリティが高くて，そして非常に多くの人から使えて，また非常に多くのインフォメーションをもっているというようなコンピュータ・ユティリティと呼ばれる用途に使われるだろうと予測されます．

さて，次は，スーパコンピュータの将来ですけれども，これは非常に高速になると予測されます．特殊用途の計算機をつくるということがフィージブルになる．すなわち，ある特殊問題を解くコンピュータをつくることが経済的に可能な時代になってきます．

たとえば現在たぶん一番速いスーパコンピュータとしては GF 11 というのがあります．これは IBM のワトソン研究所でつくられているコンピュータですが，これは 11 ギガフロップ出るということで GF 11 という名前をつけてあるコンピュータなんですけれども，これは QCD (Quantum Chromo Dynamics) という理論物理では非常に重要な計算がありまして，それを解くためにつくられたコンピュータです．これを使

ってプロトンの重量を大変精度高く計算できます．しかしそれには GF 11 を使って1年間計算にかかります．しかしクレイを使うと100年かかるということで，こういうコンピュータを使わないとこの計算はできないということです．

一方，10 年後には 1 テラフロップスのスーパコンピュータはつくられるであろうと思われます．1テラフロップスと申しますと，たとえば先ほどの予測のように，シングル・チップで 100 MIPS ですと 1万台つなげばいいわけです．1万台つないだコンピュータというのは現在の接続技術の延長で可能ですから，1テラフロップスもハードウェアとしては可能な範囲だと考えられます．

3. 重要な研究課題

これで，テクノロジおよびシステムの予測をおえ，どういう研究が重要であるかを考えましょう．基本的には，これからどんどん進歩するテクノロジをいかにうまく利用するシステムをつくり，また使うかということを考えることが重要な研究です．

パソコンでは 10 MIPS，ワーク・ステーションのでは 1000 MIPS を使えても現在私がコンピュータを使うのはコンピュータ・メイルとエディタですけれども，その両方が今までの 1,000 倍速くなってもほとんど有り難くないわけです．1,000 倍の MIPS を使うような需要をどういうところで生み出すか，ということが大変重要な研究になってきます．例えば人工知能であるとか，あるいはコンピュータ・グラフィックスであるとか，あるいは音声とかあるでしょうけれども，特に重要なテーマ，あるいは解決しなければならないテーマを4つ選びますと，まず一つはコンカレンシの問題です．コンカレンシの問題は，先ほど申しましたとおり，1000 MIPS，あるいは1テラフロップスのコンピュータはマルチプロセッサになるからです．

2番目にはマルチメディア・インフォメーション・システムをあげます．現在のコンピュータではグラフィックスや音声が使えるようになってきています．これからのコンピュータではこれらをさらに進めたマルチメディアが計算機のアトラクティブなセリング・ポイントになります．これらはまた MIPS を消費することにもなります．

次に重要な研究テーマはヘトロジニアス・オペレーティング・システムです．世の中にはいろいろのオペレーティング・システムがありますが，これらは統一されることはありえないでしょう．しかしながらコンピュータ間のコミュニケーションは非常に重要になってくる．たとえば東京のコンピュータを使って計算しながら，ときどきボストンにあるコンピュータで演算をして結果をもらって何かをしたいという使い方が増えてきます．そのためには複数の機種間でデータのやりとり，プログラムの呼びあいなどを実現せねばならず，解決策としてヘトロジニアス・オペレーティング・システムというものが重要になるわけです．

最後は，AI がどれほど成熟化して実用化されるかということです．確かに計算機が現在よりも賢くなり，かつ多くの MIPS を使うということは非常に多くのお金を使って研究しているわけですけれども，残念ながら AI ビジネスは予測よりも延びていませんし，その進歩も予測に達していないわけです．

それで AI テクノロジを成熟させ役に立つものにし，かつ大きなビジネスにするのが重要な研究です．

4. コンカレンシ

最近コンカレンシに関する研究・開発が爆発的な勢いで進歩してきました．特にアメリカで非常に多くのコンカレンシを商品とするビジネスが始まっています．なぜ今急にコンカレンシかというと 32 ビットの高性能マイクロプロセッサが得られるようになったからです．

なぜ 32 ビットであるかといいますと，16 ビットですとメモリ・アドレス・ケイパビリティは非常に限られる．ということは，10 台つないでもメモリのリミテーションからたいして大きなプログラムは走らせない．32 ビットになりますと，メモリが 4 ギガバイトまでアドレスでき非常にメモリがたくさん使えます．そうすると，コンピュータというのはご存じのとおりメモリにマイクロプロセッサがついたものですから，メモリが大きくなるとその周りにたくさんのマイクロプロセッサを付加できるようになる．たとえば 16 ビットですと 64 キロバイトの周りにマイクロプロセッサをつけても，マイクロプロセッサばかりの集合になってしまってちっとも性能が出なかったわけです．それが 32 ビットのマイクロプロセッサが出てきたわけです．これを単につなげることによって，非常に高性能なミニコンピュータ，あるいはワーク・ステーションが簡単にできるようになったわけです．これらの構成で共通していることはバス結合であること，そしてまた多くのものはシェアード・メモリであること，ま

た4台から20台の小規模マルチプロセッサであるということです．

なぜこういうアーキテクチャが選ばれたかと申しますと，もっとも重要な要因はソフトウェアのユニプロセッサからマルチプロセッサ，あるいはマルチプロセッサからユニプロセッサへの移行がもっとも簡単だからということです．ということで，もっとも汎用性の高いマルチプロセッサであるといえます．

まずバス結合の話をしましょう．一番重要な研究課題は，メモリ・バスへのアクセスを少なくするにはどうするかということです．たとえばMC68000ではメモリ・バスへのアクセスはバス容量の80%にも達するというデータがあります．ということは2台つなげると，もうそれによって1台当たりの性能は，60%に落ちてしまうということです．メモリ・バスのトラフィックをいかに少なくするかということは，ここ5，6年の研究だったわけです．

その一つの解法はライト・バック・キャッシュを使うことです．ライト・バック・キャッシュというのは通常のキャッシュ・メモリよりも，主記憶へデータを書く割合がずっと小さく性能の高い方式です．キャッシュ・メモリを使いますとバスへのアクセスが減ってマルチプロセッサの性能が大幅に上がります．

そうすると，ご存じのようにメインフレームでも昔からありましたけれども，一番問題になるのはキャッシュ・コンシスタンシィといいまして，いくつもキャッシュがありまして，そこの一つのマイクロプロセッサが一つのキャッシュに書き込むと，同じアドレスをもっていたほかのキャッシュの内容が違ってくる可能性が出てくるのを避けなければなりません．

それでいろいろメインフレームでも研究がなされてきたわけですけれども，メインフレームとこのようなワーク・ステーション，あるいはミニと違うことは，特にワーク・ステーションではユーザが1人であることです．ということは，すべてのアプリケーション・プログラムをコントロールできます．一方，メインフレームでは，どんなアプリケーションが自分と同時に走っているか，ほとんど予測できないので最適化がむずかしいわけです．

一方，ワーク・ステーションではお互いに協調できるプロセスの走っているシステムだということで，比較的楽観的にキャッシュのアーキテクチャを設計できまして，その一つとして最近特に注目されているのは，スヌープ・キャッシュと申しまして，これはXEROXのドラゴン，あるいはDECのファイアーフライというマルチプロセッサ・ワークステーションで使われております．これは，1981年に私がC. サッカーと世界で初めて設計したものです．論文も昨年あたりから出始めましたのでご存じかと思いますけれども，どういうものかといいますと，キャッシュ・メモリがありまして，このメモリの上のほうにプロセッサがあって下のほうにメモリ・バスがあるわけです．このキャッシュ・メモリを，メモリとプロセッサと両方で同時に中身をみてアップデートを行います．要するに，キャッシュ・メモリのコンシスタンシィを管理する機能を完全に分散化するアーキテクチャです．

一方，こういうキャッシュを使いますと，シンクロナイゼーションも低価に実現できます．マルチプロセッサの基本操作としてロッキングが不可欠です．これは一つのメモリ領域を二つ以上のプロセッサでアクセスする可能性があるのでアクセスしているときはほかからのアクセスは排除しなければいけない．マルチプロセッサのロックのやり方の一つにスピン・ロックというのがあります．これはビジー・ウェートでして，あるフラッグを見て，そこがセットされているとループして待っている方法です．

これは非常に簡単で性能もいい．特にいくつも小さなプロセスが動いているときには非常に性能がいいということは実証されているんですけれども，シェアード・メモリ・マルチプロセッサ上に実現しますと性能が落ちる．なぜかと申しますと，待っているプロセッサがときどきフラッグを見にいかなきゃならない．そうすると計算していないプロセスがメモリ・バス・トラフィックを起こすので，システムの性能を落としてしまう．しかし，キャッシュを使いますと，スピンして待っているフラッグがキャッシュの中に入ってしまう．そうすると，待っているプロセッサは自分のキャッシュをリファレンスするだけで済むので一切メモリ・バス・トラフィックを起こさないので，大変能率的で簡単なシンクロナイゼーションも実現できるということです．

次の研究課題は，こういうマルチプロセッサの上にどういうプログラミング・ランゲージ，およびプログラミング・システムをつくるかということです．最も普及しているCとかフォートランなどもこういうようなマルチプロセッサの上で動かなければいけないし，あるいは大変ハイレベルのファンクショナル・ランゲージも動くでしょう．

そういういろんなプログラミング・ランゲージについて研究していかなけりゃいけないのですけれども，将来の研究方向を示す一例として，われわれが行っているコンカレント・ランゲージの研究のお話をいたします．

昨年の12月「CACM」にピーター・デニングが，「コンカレンシの4世代」についての論文を書いていますが，大変良く発展をとらえています．この中で彼は第1世代はメインフレームのマルチプロセッサであって，これは各違うタスクをそれぞれ空いているプロセッサに実行させるOSによって実現された．これは現在成功して幅広く使われている．

2番目のコンカレンシの時代というのはメッセージ・パッシングであると言っています．これはトニー・ホアーのコミュニケーティング・プロセスで代表される言語で，完全に違うアドレス空間にあるプロセスがメッセージを送って演算を実行するものである．プログラマは並列性を意識してプログラムを作成し，直列用と並列用との間でプログラムの互換性はない．これも実用化されている．

第3世代とは，直列用のプログラムが最小の変換で並列用になるものをいいます．これは基本的にシェアード・メモリのコンカレンシをモデルとしてもっていて，データを並列に動くプロセス（最近ではスレッドと呼ぶことが多い）間で直接シェアできる．

第4世代では，たとえばファンクショナル・プログラミングのように，プログラマは直列用か並列用かは意識しないで書いて自動的に並列用プログラムが作成されるのを言う．現在の技術動向は第2世代から第3世代に移っていこうとしているところだという論文です．そういう意味ではわれわれの考えている言語，ユニファイド・メッセージ・センドと呼んでいますけれども，これは第3世代です．これをLISPの中で実現することを考えています．どういうようなものであるかといいますと，これはエクスプレッション・ベースのコンカレンシであります．旧来のコンカレンシはスレッド・ベースのコンカレンシと呼びましょう．スレッドはプロセスという概念と比較してつくられた概念です．たとえばUNIXにあるようにプロセスはフォークを実行して，今までと完全にアドレス・スペースは別であって，しかしながらパイプでコミュニケートできるという実行環境です．それは第2世代のコンカレンシであるわけです．

第3世代になりますと，そのように並列に動く実行環境を軽くつくらなければならないのと，コミュニケーションも簡単にできなければならないという要求が出てきます．アドレス空間は完全にシェアしているというものはこれは第3世代のコンカレンシで，これを最近のターミノロジではスレッドと申します．

このスレッド・ベースのコンカレンシに対して，われわれの創案しましたのはエクスプレッション・ベースのコンカレンシです．何かといいますと，要するにLISPの操作を一つの見方でしますとこのようにツリーがありまして，このツリーを書き換えるんです．そうするとたとえばこのように，左にA右にBというサブツリーをもったツリーを考え，今あるLISPのファンクションが，これをaとbというサブツリーをもったツリーに変換するとします．これを普通の直列型演算でどうやりますかというと，まずAをaに書き換えて次にBをbに書き換えます．これを並列に実行すると，両方のサブツリーを同時に書き換えます．ですからプログラム上でフォークを実行し，左を書き換えるプロセスをつくるのと右を書き換えるプロセスをつくるのでなく，単にエクスプレッション上で書き換えるという意味で，これをエクスプレッション・ベースのコンカレンシと言います．

これをユニファイド・メッセージ・センドと言いましたからにはまた他に理由があります．LISPで式を書き換えるのに，そのタイミングをみてみましょう．「書き換えろ」と言って「はい」と言って書き換えていって，結果が変わりましたというのはこれはイミーディエットであると言うことにしましょう．「書き換えろと言って」，そうするとどこかにほかのもの，あるいはほかのところで計算し始めるというものは，いつか未来に終わるということからフューチャと言うことにしましょう．この中間として，「書き換えろ」と言っても，その演算は本当にその値をほしいときにはじめて実行されるものも考えられます．これは昔レイジ・エバリーエーションというものが発明されましたけれども，その実行の方法と同じです．そのほかにも計算のモードというものはあるかもしれませんけれども，この三つのモードのみを式の評価方法とし，これを全部一つのメッセージ・センドの上にオーバ・ロードしたものが，われわれの言っているユニファイド・メッセージ・センドです．イミーディエットのメッセージ・センドは(F A)と書きます．フューチャのメッセージ・センドでは(future F A)と書き，レイジのメッセージ・センドは(lazy F A)と書きます．

そして，かつこれにオブジェクトの概念を取り入れます．そうして lazy も future もすぐ結果を返すようにします．

この結果は何かと言いますと，たとえば future の場合には，FUTURE というクラスのインスタンスが返ってきます．これは何かと言いますと，プロセス・コントロール・ブロックのようなもので，その一部に実行のためのコードが含まれていて，その他に実行環境を含んでいてその下で計算しています．ですから結果をすぐツリーの中にアサインしても計算は続行されているわけです．ツリーがあって，左側のサブツリーAにまず future で呼んで，するとすぐ結果が返ってきますからそれをアサインしておけばいい．普通のフォーク，ジョインのように，計算が完了するまで待って，戻った答をアサインするということをしなくてすみます．しかしながら，future はすぐそれ自身では値が得られませんから，次にそれにまたメッセージを送ると，そのときはまだ計算が完了していないとそのメッセージは待たされることになります．

一方，lazy も同様で，Aというサブツリーに lazy で呼びますと，すぐ結果が返ってきます．その結果はクラス LAZY のインスタンスです．たとえば大きなディクショナリをつくりたいときにはこれが便利です．レイジ・エバリエーションでどんどん結果をつくっていって，そのディクショナリにどんどんアサインしていきます．その値は何かと調べに行ったとき初めて計算が実行されます．

またこうしますと，エクセプションも大変きれいに扱えます．並列用プログラム言語では，いくつもプロセスが動いていてどこかでエクセプションがあったとします．そのエクセプションがどこでもキャッチされないと一番上のプログラムも通り抜け，どこにコントロールを移すか困ることがあります．

しかし，このようにユニファイド・メッセージ・センドを使いますと，フューチャのインスタンスの中でエクセプションが起こって，どこでもキャッチされないと，そのフューチャのインスタンスをクラス EXCEPTION のインスタンスに変えます．

次に値をとりにいったときに，初めてディバッガに入るというようなセマンティックスにできます．

それでは最後ですが，最も重要な研究項目として，このような並列をうまく使うアプリケーションをつくる研究があります．

一つはエキスパート・システムです．エキスパート・システムはどれだけ並列度があるかということが一つの問題点ですけれども，カーネギー・メロン大学のある学生がプロダクション・システムの並列度を調べたところ，ほぼ log N のスピードアップで，かつ10ぐらいまでの並列度しかなかったということです．ということは，10台がほぼエコノミカルな限界であると考えられます．

また，われわれ IBM 東京基礎研究所でも機械翻訳について，どのぐらい並列度があるかということを調べてみましたけれども，だいたい10ぐらいが並列度の限度という結果がでました．ということから，この10台程度の並列性をもったコンカレント・マシンというのは今後非常に重要な計算システムになるだろうと考えられます．しかしながら，一方，500台程度の並列性をもった並列計算システムというのも重要になるでしょう．これらの計算機では，ロボット，イメージ処理，グラフィックスのような分野で多くの使用が考えられます．

また，非常に大規模の並列性になりますと，コネクション・マシンというものがあります．将来は何テラフロップスというような計算機がこのように非常に大規模な並列を利用してつくられるでしょう．するとアプリケーションはシミュレーション，セマンチック・ネットワークの実現，あるいはグラフィックスというような応用に使われるだろうと思われます．結論としては，コンカレンシの研究ではアーキテクチャの研究，オペレーティング・システムの研究，プログラミング言語の研究およびアプリケーションの研究があります．特にアプリケーションの研究は非常に重要なことですので，これからは非常に熱心に研究されなければならない分野だと思います．

5. マルチメディア・インフォメーション・システム

2番目の重要な研究分野としてはマルチメディア・インフォメーション・システムをあげましょう．これはアプリケーション・ドリブンな研究であります．一言で言いますと，われわれはいまだにキャラクタの世の中に束縛されています．ですからメッセージを送るにも，ワープロを使うにも，キャラクタで入力してキャラクタを出力しています．コンピュータシステムはキャラクタだけはうまく扱えるようにつくられているわけです．

しかしながら，これをもっとアトラクティブにして

経済価値の高いものにするには，イメージ，グラフィックス，音声，あるいはアニメーション，ビデオなどが自由に使えるようにならなければならないわけです．

そのように，いろいろな情報形態を自由に使えるシステムを完成するには非常にいろんなアプリケーションを書かなければならないわけです．ですから研究としては，いかに多くのアプリケーションの統一をとるとか，あるいはつくるときの手間を少なくするかということを考えなければならないわけです．

たとえばいくつものインフォメーションのフォームが出てくるわけですけれども，これらを現在のキャラクタと同じように，いろんなことができるようにしなければなりませんし，Aの機械からBの機械にいろんなデータを伝送するにも，キャラクタと同じような便利さで送られなければなりません．もしもできないと，情報交換の世の中ではローウェスト・コモン・ディノミネータに収束しますから，いくら1台だけは非常に便利なものがあっても，ほかの機械が全然そういうものが使えないとやはり情報交換はキャラクタのみでということになります．

そうすると，オペレーションとしてはインプット，ディスプレイ，プリンティング，トランスミッティング，ファイリング，データベース・クエリなどというオペレーションは全部サポートしなければなりません．研究ではこういう機能を実現するのに，いかに少ない量のプログラムで実現するかということが重要になってきます．それについてわれわれの知っている解法の一つとしてはジェネリックがあります．コンピュータ・サイエンスの専門用語ですけれども，これは何かというと，たとえばプリントという重要な操作は，これはたとえばキャラクタ・データにプリントと言っても，イメージ・データに言っても，ボイス・データに言っても，やはり同じような動作をしてほしいというのが基本思想です．

そうすると，このようにプリントという操作をジェネリックに定義しましても，各情報形態に対するプリントの実現は個々に書かなければならないわけですけれども，プリントを使ったアプリケーション，たとえばデータ・ベース・マネージメント・システム，あるいはその上につくるオフィス・インフォメーション・システムなどは，それぞれの情報形態に重複してつくることなく一つのプログラムで全部の情報形態を同じように扱うことができます．

そのマルチメディア・インフォメーション・システムの中でどのようにクリエイティブなアプリケーション・ソフトウェアをつくるかということはこれは非常に重要な研究ですけれども，それと同時に，このようなシステムを開発する計算機言語が非常に重要になってきて，その最も重要な機能の一つがジェネリックです．IBM の東京基礎研究所でも COB という言語を設計しています．これからのシステム用言語ではジェネリックと，オブジェクト，あるいは，モジュラリティというような機能が不可欠になるでしょう．

6. ヘトロジニアス・オペレーティング・システム

3番目に重要なテーマはヘトロジニアス・オペレーティング・システムで，これは異機種コンピュータ間のコミュニケーションをうまくしていくにはどうするかという研究です．現在だいたい二つの分野でこれがうまくいっていまして，要するにワークステーション・ワークステーション間を LAN を使って通信することと，パソコンとメインフレーム間のホスト PC リンクというものですが，それでも1社の機種だけに限られるわけです．これがたとえば UNIX があって，VM があって，MVS があって，MS/DOS があってという世の中で，全部のコンピュータ間でリソースがシェアできて，かつリモート・プロシージャ・コールを可能にしたいわけです．これを非常に簡単に実現するということがこれからの研究に重要な課題になるわけです．

それではどうするかというと，現在考えられている一つの方法は，各オペレーティング・システムの上に薄いネギの皮みたいなものをつけて実現しようということです．それにはポータブルで移植の簡単なものをつくりまして，それを UNIX 上に移植し，また MS/DOS あるいは VM の上にも移植しますと，それから自由に UNIX マシーンをリモート・プロシージャ・コールで使えるようになるわけです．そのようなものを考え出そうというのが非常に重要な研究テーマになると考えられます．

7. 人工知能

最後に人工知能です．人工知能の研究は非常に盛んなわけですけれども，経済的にはあまり芳しくない．ですから AI というのはなるほど非常にアトラクティブな分野だけれどもこれが現実になるにはかなりなギ

ャップがあります．たとえば知識リプレゼンテーションとか知識アクイジションとか，これは10年どころでは解決しなく100年もかかるだろうと研究者が言うような分野もありますけれども，これから10年間で実用度が急速に進む分野は何であるかというと，AI言語とシェル，エキスパート・システム，自然言語，ロボットなどがあるわけです．

AI言語普及の課題は，一般のソフトとのコネクションがなければならない．そういう意味で，現在たとえばシェルだけがLISPから独立してMS/DOSの上で動くというようなものは，普及への方向の一つだと考えられます．単独にLISPマシーンがあって，そこで単にプロトタイプをつくるというような使い方は，経済的，あるいは世界に対するインパクトという意味では非常に小さいものです．

エキスパート・システムが重要になるには，現在のようなシャロ・リーズニングだけでは高度なことができず，すぐ限界にくると思われます．たとえば現在のリーズニングだとだいたい3レベルのインファレンスぐらいが一番複雑なエキスパート・システムで実現されているわけですけれども，これがたとえば9とか10とかのようなレベルまでになると，かなり高度のことができると思われます．しかしそうなると非常につくるものも大変ですし，また検証の問題も出てきます．それから考えると，このように非常に賢いエキスパート・システムは，オペレーティング・システムでのUNIXのように世の中に一つか二つしかこれから10年間には開発されなくて，アプリケーションというのはそれを単にパラメタを変えて使うだけになる可能性がかなり高いだろうと考えられます．

自然言語については，これは非常に日本では重要なものですけれども，マシン・トランスレーションが一番重要な応用でしょう．これは非常にむずかしい問題ですけれども，人間が翻訳結果をチェックしないでも動くというようなものでないかぎり実用にならないでしょう．もう一つの技術的方向は，システムが自分で結果を評価してくれるようなものがほしいです．これは翻訳したけれども正確かどうか自信がないとか，これは非常に自信があるとかいうようなことをシステムが言うわけです．こうするとチェックの努力がずっと減ります．

結論を申しますと，今後10年間にも半導体を中心に，テクノロジの進歩は大変目覚ましいものがあると思われます．ですからコンピュータが現在の1,000倍，1万倍の能力になるのは夢ではないと思われます．しかしそのような強力なものを十分使いこなすには，いかにすばらしいアプリケーションを考え出せるかによるわけです．特にグラフィックスとか，音声認識，人工知能などの分野の進歩を考えなければならないであろうと思われます．

そのような非常にすばらしいコンピュータシステムができてきた世の中で，コンピュータが真に使いやすくなるにはいくつもの重要な研究分野がありまして，そのうち特に重要だと考えられる4つとして，今日お話ししましたコンカレンシ，マルチメディア・インフォメーション・システム，ヘトロジニアス・オペレーティング・システムとあとAIのテクノロジの研究を成功させることだと考えられます．

大変時間をオーバしまして，最後のほうは大変速くとばしましたけれども，これで終わりにいたしたいと思います．（拍手）