OR メモランダム

このコラムは，ORにかかわる概念，知識(手法，原理)，それらの図解，よい教材や問題，実学ORの実施経験，そこから得られた知恵やアドバイス，失敗談と教訓，新しい観点，視座，フレームワーク，未だ解けていない問題，面白い研究テーマなどを，“新鮮に”，しかも，“コンパクトに”表現し，提示していただくものです．ユニークなアイディア，フレッシュな見方，発想，だれかと意見をたたかわせたい問題提起など，ふるってご投稿ください．(原稿は，刷り上がり，半ページから3ページに納まるようにお書きください．簡素に！加筆訂正をお願いする場合があります)

OR-CASEのすすめ

八巻　直一

ORの定義のひとつに，次のようなものがある．

『科学的な方法，手法および用具を体系の運用に関する問題に適用して，運用を管理する人に問題に対する最適な解を提供することである』

この定義にしたがえば，OR屋の仕事は科学的な方法，手法をそろえて，管理者の提出する問題の最適解を求めることとなろう．方法をさらに磨き上げることも，大切な仕事ではある．そして，今日ではそれらの手法は，例外なくコンピュータのソフトウェアとして，実現されているのである．

さて，ORを企業などの中で専門的に業務とする人たちが，はたしてどれだけ存在するかは不明であるが，おそらくそう多くはないと想像できよう．たぶんそれゆえに，コンピュータ・メーカはOR屋にははなはだ冷淡であるように見える．情報処理学会の発表内容にも，それは端的に現われている．最近の情報処理学会の発表件数は，ゆうに1000件を越えるが，数理計画法に関係するものはほんのわずかにすぎない．講演プログラムからそれらをさがす作業は，まるで大相撲の幕下以下の番付を虫眼鏡で見るようである．

しかしながら，これからの社会活動の中で，ORの役割が急速に増大するであろうことは，疑う余地がないであろう．そして，ORにおけるコンピュータの重要性もまた，途方もない重さを加えていくであろう．コンピュータ・メーカおよびソフトウェア・ハウスには，ORのためのコンピュータ利用技術の整備にもっと目を向けてもらいたいとの，切なる希望を表明したい．

やまき　なおかず　㈱システム計画研究所
〒150 渋谷区桜丘町2—9

一方では，コンピュータ・サイエンスの発展はめざましく，ハードウェア，ソフトウェアともに，まさに日進月歩である．数値処理の分野では，ベクトル・プロセッサや並列処理が，ソフトウェアの分野では，AIやファジーの利用がめざましい．その他の分野でもいちいち挙げていられないほど多くの技術革新が，日夜起こっているのである．

ソフトウェアの設計・製造技術の分野で，最近話題になっている技術に，CASEといわれるものがある．CASEは，Computer Aided Software Engineeringの略であり，従来の紙と鉛筆による設計，エディタとコンパイラだけにたよったプログラミング，手探りの検証作業などを，一貫した機械環境のもとで，コンピュータの積極的な助けを受けながら実施することで，革命的な生産性の改善を狙うシステムである．CASEは，まだ決定的な実用化には至っていないけれども，CASEの考え方はソフトウェアの設計・製造のみにとどまらず，いろいろな分野に急速に拡大するであろうと予想されている．

ORの世界でも，このCASEの考え方は，十分に考慮する価値があると思われる．CASEと呼ぶべきかどうかには，異論があるかもしれないが，仮にそれをOR-CASEと言うことにし，以下に糸口となりそうな事柄を示そう．

ORの世界でも，たとえばいわゆる表計算ソフトウェア（スプレッドシート）を上手に使って，プログラムをいちいち書かなくとも，かなりたくさんの手法を試すことのできるようにした工夫などが，OR-CASE の萌芽といえるかもしれない．スプレッドシートを用いることによって，CASEらしきものを比較的安易に実現す

ることができるというわけである．スプレッドシートを使うと，ソフトウェア生産性が，コンパイラを用いた場合の数百倍に達することさえあるのだそうだ．

この事実は，おおいなるヒントである．問題の記述および解法の記述が，専用の言語によってできるならば，問題の解析作業の，生産性と信頼性は間違いなく飛躍的に高まるであろう．つまり，OR-CASE の要素として，問題記述専用の言語は不可欠である．つけ加えるならば，その言語はFORTRANと十分に近い（互換性がある）ことが望ましい．なぜならば，これまでの手法の蓄積の大部分が，FORTRANで書かれているからである．

次にほしいのは，数式処理の道具である．手法の開発や問題の分析には，微分積分などの数式処理が欠かせない作業であるが，数式処理言語を積極的に用いることは，ほとんどないのが現状であろう．もし，数式処理言語が，使いやすい形で，さりげなく使えるように工夫されているならば，問題分析などの生産性，信頼性はこれもまた飛躍的に高まるであろう．

DTPといわれる文書編集システムは，最近めざましい発展をとげた．ORの作業を考えるとき，レポートの作成作業の比重が非常に大きいといえる．数理的レポートを作るための，使いやすいDTPシステムは，ぜひ必要である．

その他，外部のデータにアクセスするための通信システムや，自分のデータを管理するためのデータベース機能などがOR-CASE の主要な要素となろう．

考えてみると，上にあげた要素の多くは，現在でも世の中に存在するものである．しかしながら重要なのは，これらを統合的に，一元化したコンセプトのもとで使えることである．バラバラに要素を集めても，必ずしも強力な道具とはなり得ない．やはり，OR屋の立場をよく理解したうえで，全体を統合的に考えたシステムを構築するべきであろう．

たぶん，OR-CASE は，ワークステーションの上に搭載されるであろう．そして，X-window などのマルチウィンドウを用いたシステムが使われるだろう．各ウィンドウは，ワープロかスプレッドシートのようになっていて，文書，表，データベース，通信，問題の分析あるいは手法のプログラミングまでが，一元的に行なえる雑記帳となるであろう．

レポートを書きつつあるときに，式変形や方程式を解く必要がたびたび起こるであろう．そのとき，ボタンひとつでそれが実行できて，あたかもワープロで書いたかのようにレポート上に書き込まれれば，どんなにうれしいか．ちょっと知りたいことがあれば，OR-CASE にむかって質問しよう．すると彼（彼女？）は，内部または外部の適当なデータベースから，適切な答えを見つけてきて，レポートにしてくれるのである．筆者らの夢は，こんな道具を作ることである．そして，少しずつこしらえてはいるのである．コンピュータに詳しい皆さんの多くが，OR-CASE の構築に興味を示してくださることが，われわれの願いである．

筆者らが開発を進めている最適化問題解法ソフトウェア ASNOP や，アルゴリズム記述言語 LAMAX™ は，このようなOR-CASE の要素として考えられたものである．ASNOPは，非線形最適化問題を記述すると，問題に最も適した解法プログラムを生成する．多くの問題は，非線形最適化問題に帰着するが，OR技術者にとって，それらに対していちいちプログラム作成することは，大変な手間であるばかりか，場合によっては難しい論文を読解したり，分厚い専門書を読み下したりしなければならない局面さえあり得る．ASNOPは，その手間を軽減して，OR技術者の生産性を高めることを目的として作られた．ASNOP については，本誌1988年11月号に紹介されているので，ここではこれ以上は触れないでおく．

LAMAXは，FORTRANを用いた場合に，特に最近問題となる点を解決する．

FORTRANの問題点は，スーパーコンピュータの性能を引き出すためのチューニングにある．チューニングには，相当専門的な，しかも各コンピュータ特有のノウハウが必要であり，それらにいちいち対処することは非生産的である上に，なかなか最高のチューニングはできない．また，手法の多くには，行列表現が現われるので，FORTRANの配列に書き直す作業も伴う．

LAMAXは，これらの作業を除去して，プログラミングの生産性，信頼性を飛躍的に向上させるとともに，スーパーコンピュータの性能を最高に引き出すプログラムの生成を実現する．たとえば，3月号のORメモランダム『一般化逆行列を用いてLPの一般解を求めよう』の中で，以下のような記述が見える．記号の説明は，当該記事を参照していただくことにする．

『問題』 制約条件 $\boldsymbol{d} \leqq \boldsymbol{A}\boldsymbol{x} \leqq \boldsymbol{b}$ のもとに，$\boldsymbol{c}^T\boldsymbol{x}$ を最小にせよ．

これに対する一般解は，次のように表わされる．

```
      REAL A:RECTANGULAR[M,N],G:RECTANGULAR[N,M]
      REAL I:DIAGONAL[N]
      REAL x:VECTOR[N],u:VECTOR[M],v:VECTOR[N]
      REAL d:VECTOR[M],b:VECTOR[M],c:VECTOR[N]
      REAL z:VECTOR[M]
C
      DO 100 L = 1 , N
        I[L:L] = 1
  100 CONTINUE
C
      CALL MINPUT(A)
      CALL MINPUT(d)
      CALL MINPUT(b)
      CALL MINPUT(c)
C
      G = GINVRS(A)
C
      z = G' * c
      DO 200 L = 1 , M
        IF( z[L] .GT. 0 ) u[L] = b[L]
        IF( z[L] .LT. 0 ) u[L] = d[L]
        IF( z[L] .EQ. 0 ) u[L] = 0.5*(d[L]+b[L])
  200 CONTINUE
C
      CALL MINPUT(v)
      x = G * u + ( I - G * A ) * v
C
      CALL MPRINT(x)
      f = c' * x
      WRITE(*,*) ' OPTIMAL VALUE=',f
      END
```

$x^*=A^-u^*+(I-A^-A)v$

ここに，vは任意のn次元ベクトルである．また，u^*は，以下のようなベクトルである．ただし，$z=(A^-)^Tc$とする．

$(u^*)_i=b_i$ ただし $z_i>0$

$(u^*)_i=d_i$ ただし $z_i<0$

$(u^*)_i=(1-t_i)d_i+t_ib_i\ (0\leqq t_i\leqq 1)$ ただし $z_i=0$.

これをＬＡＭＡＸで表現する場合，ほとんどそのまま書きくだせばよいのである．かたや，ＦＯＲＴＲＡＮでプログラミングしようとすると，かなり苦労しなければならないだろう．さらに，ベクトル演算を意識したコーディングをやろうとすると，ハードウェア特性をよく知っていなければ不可能である．

ＯＲ-ＣＡＳＥ の要素としてのＬＡＭＡＸは，ここでも大きな助けとなる．ＬＡＭＡＸは，ＦＯＲＴＲＡＮと比較して，より抽象度の高い言語であるから，ベクトル演算向きの最適化を，プログラムに書き込まれた意味を把握した上で，自動的に行なうことができるのである．

ＬＡＭＡＸは，ひとつの例であるが，問題記述向きの言語を作ろうという動きは，最近割合盛んになりつつあり，今後は数理的な解析用にいろいろな言語が提供されることが期待される．

最後に，上の記述を ＬＡＭＡＸ で表現した例を示して，この稿を終えたい

上のプログラム中で，任意ベクトルの入力は省いて，特殊解 G*u と（Ｉ－G*A）を出力したほうがよいだろう．また，ＧＩＮＶＲＳ(Ａ)はＬＡＭＡＸの組み込み関数で，ムーア・ペンローズ型の一般化逆行列を求める．

なお，ＬＡＭＡＸの紹介は，いくつかの場所で行なわれているが，一番最近では，1990年春期情報処理学会の論文集（並列処理言語）を参照することができます．

注）ＬＡＭＡＸは㈱システム計画研究所の商標です．