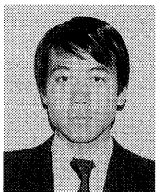
大沢文夫："微生物の行動"，ヒューマンサイエンス，3，生命現象のダイナミズム，中山書店，pp. 37-60 (1984).
4) 脳神経系の膨大なフィードバックによる調整機能は次の文献に詳しい．
R. グラニット："目的を持つ脳"，海鳴社(1978).
5) 部品化によるシステム構成法，要素化によるシステム解析法に対する批判については次の文献を参照してほしい．
アーサー．ケストラー編著："還元主義を越えて"，工作舎 (1984).
6) 大脳皮質の構築，コラム間，領野間の線維連絡については次の文献を参照してほしい．
ノーマン．D. クック："ブレイン・コード"，紀伊國屋書店 (1988).

(平成5年4月7日受付)



熊沢 逸夫 (正会員)

1959年生．1981年東京工業大学工学部電気電子工学科卒業．1986年同大学院博士課程修了．同年同大学工学部情報工学科助手．1990年同学科助教授．パターン認識，信号画像処理，ニューラルネットワークの研究に従事．工学博士．電子情報通信学会，国際ニューラルネットワーク学会各会員．E-mail: kumazawa@cs.titech.ac.jp.

並列プログラミング†

上田 和紀††

正しく，しかも効率のよい並列プログラムの作成には，記述の道具であるプログラミング言語と，並列化の方法論が重要な役割を果たす．本稿では，この二つの側面において，理論と実際がどう関わっているかを，個人的経験をまじえながら考えてみたい．

1. 並列プログラミング言語とその処理系

ひと昔前まで，プログラミング言語とそれを実行する計算機アーキテクチャとの間のセマンティック・ギャップは，次第に減少するものと期待する向きが多かった．たとえば高級言語マシンの研究は，この方向を追求するものであった．

しかし，最近の RISC や並列計算機のめざましい進歩と普及は，この期待とは逆の方向をいくものであったと言える．これは，むしろ歓迎すべき方向であったと思う．なぜなら，プログラミング言語と計算機アーキテクチャそれぞれに対するわれわれの理解を深め，さらには両者をつなぐ方法論の進歩をもたらしたからである．

この動きの背景には，コンパイラ技術の近年のめざましい進歩がある．これは，理論と実際がよくかみ合っている分野といえよう．すでに逐次言語については，人間が容易に想像できないような機械語を最適化コンパイラが出すようになった．レジスタ割付けのようなスケジューリング問題は，本来計算機が得意とする問題であったわけである．さらに，RISC や並列計算機の出現は，文や命令のスケジューリング技術をも大きく進展させた[2]．これらのスケジューリングは，単に最適化問題を解けばよいというものではない．むしろ，最適化問題を切り出すためのプログラム解析技術がきわめて重要である．より詳細で，より大域的な解析がとりいれられるようになった結果，高度な最適化が可能になったわけである．

このように，既存のプログラミング言語を所与のものとして，自動並列化を含む最適化をしようとする試みの一方で，新たな並列プログラミング言語の設計も盛んである．これは，より簡潔で汎用性の高い理論を軸として，並列処理へのエレガントな方法論を構築しようとするものである．しかし，既存の言語のもつ巨大な文化と慣性に対抗するためには，強固な理論的基盤と実用性とを同時に示す必要がある．

筆者は長年，並行・並列プログラミング言語 GHC および KL 1[9] の研究開発にたずさわってきた[10]が，そこで感じたことは，言語設計やその後の発展に際して，処理系作成，記述力，理論的扱いの三者を，バランスよく考えることの重要性で

† Gaps Between Theory and Practice: Parallel Programming by Kazunori UEDA (Department of Information and Computer Science, Waseda University).
†† 早稲田大学理工学部情報学科

ある．諸側面の相互交流が重要であると言うこともできる．

例をあげよう．プログラミング言語の理論研究の一つの柱は，その形式的意味論の研究である．最適化コンパイラの正当性など，意味論なしでは議論できないはずであるのに，残念ながら形式的意味論の研究がこれまでに実践に及ぼした影響はあまり大きいとは言えない．その一因は，意味論研究からのフィードバックの不足にあるのではないかと思う．与えられた言語を正確に記述することは当然重要であるが，それだけでは一方通行に終ってしまう．言語設計者の立場からすれば，理論研究から言語設計への積極的なフィードバックがほしいのである．

処理系作成や記述力といった実用的側面と，言語設計との相互交流も重要である．理論的計算モデルから発生した言語が下手に実用性を目指すと，理論研究のための版と実用版とができ，双方がほとんど遊離してしまいかねない．GHC/KL 1の一つの設計目標は，この遊離を最小限にとどめるということであった[10]．理論版である GHC と実用版である KL 1 の間に差異はあるが，その差異と関係を明確にするように努めた．

先述のプログラム解析であるが，新たな並列言語の場合も，既存の手続き型プログラムに伍する性能を得るために重要な技術となる．ここで，処理系を高速化するだけならば高度なプログラム解析の枠組を作ればよい．しかし言語設計まで含めて考えると，プログラム解析がアドホックにしかできないよりは，静的な(つまりコンパイル時の)型検査のように系統的にできるほうがはるかに望ましい．つまり，理論的定式化を，言語機能，つまりは概念の形にして明示することができれば大きな前進である．

言語設計，理論，処理系作成の間のこのような相互交流が比較的うまくいった例を一つあげよう[4]．GHC では，並行プロセスの間の通信に，論理型言語の特徴であるユニフィケーションという演算を使っている．この演算は，双方向の情報の流れを許す強力なものであるが，並列言語の通信機能として使うときは，複雑な情報の流れが静的に解析できると，処理系作成の上で好都合であるし，記述力が大きく犠牲になることもない．そこで，GHC プログラムの入出力モードの体系と，それに基づく静的解析の技法を提案した[8]．これはプログラム解析手法の提案であると同時に，言語機能の提案にもなっている．またこれは，新たな最適化手法の実現を可能にした[8]．このような相互交流は，理論と実践を分業でやっていたら容易ではないであろう．

2. 並列処理と超並列処理

最近になってようやく，日本の企業も並列計算機の開発に熱心になってきた．だが，当面利用可能になりそうな数十〜数百プロセッサ規模の並列計算機の効率を活かすのに，これまでの並列アルゴリズムの研究は役に立たない，という不満もよく耳にする．

このギャップは，現実の並列プログラミングと大多数の並列アルゴリズム研究の問題設定がまったく異なるからであるような気がしてならない．多くの並列プログラマにとっての当面の目標は，良い逐次アルゴリズムをもとに，できるかぎりの台数効果を引き出すこと，つまり n 台の計算機で理想的には n 倍の性能向上を図ることである．これに対して，並列アルゴリズム研究の主な関心は，十分な台数のプロセッサが利用できるときの時間計算量にあった．

筆者は，アルゴリズムに内在する最大並列度以上の台数のプロセッサでの処理を考える分野を**超並列処理**，それに比べて十分少ない台数のプロセッサでの処理を考える分野を**並列処理**と呼び，明確に区別するのがよいと考える．後者においては，プロセッサ台数を，与えられた問題の入力サイズや最大並列度とは独立のパラメタとして扱うものとする．もちろん，現実のプロセッサ台数をこう単純に分類できるものではないが，この両極端をよく考えることは意味があるし，具体的なプロセッサ台数でこの二つの用語を分類しようとする試みに比べれば，はるかに合理的である．

すると，並列処理では，逐次処理の時間計算量を改善するどころではなく，いかにそれを悪化させずに台数効果をあげるかが重要課題であることが分かる．もとのアルゴリズムに $100p\%$ の逐次部分があると，$1/p$ 倍以上の台数効果は望めないという **Amdahl の法則**[5] がある．並列処理への理論的消極論としてよく引用されるが，問題のサイズを大きくすれば p は一般に小さくなるではな

いかという，通常の反論だけに終ってしまうのも寂しい．重要なのは，もとの逐次アルゴリズムの中にある，強すぎる本質的逐次性を緩和して，p の値を下げる方法論である．まだ現在は，問題ごとの個別的な対応が主であるが，たとえば**見込み計算** (speculative computation)[7]という技法が有効であることは具体的に示されてきている．これをより一般的な並列プログラミング・パラダイムとして定着させることは，近未来の並列処理にとって非常に重要であろう．

並列処理における理論と実際の橋渡しを目指した計算モデルも提案されてきている．その一つがバルク同期並列 (bulk-synchronous parallel, BSP) モデル[11]であり，逐次計算の文化における von Neumann モデルのような役割を目指している．

一方，超並列処理のセッティングは，理論的可能性の追求の場としては非常に魅力的であり，実際並列アルゴリズムや超並列人工知能[3]の研究から得られる知見は大変興味深い．また，超並列計算機による問題解法として，問題の構造をダイレクトにマッピングする方式が有望視されている．これらの研究や考え方に共通するのは，論理的な見通しのよさの追求であり，これはアルゴリズム設計の出発点としては確かに重要であろう．超並列のパラダイムは，ソフトウェアの質的変革を起こす可能性をもっている．

しかし，大多数の人にとって，超並列処理の第一目的は速度向上であるにちがいない．実用化が近づき，性能への要求が強まるにつれて，現実の計算機の物理的側面への配慮をはじめとする，非常に多くの工学的な工夫が，理論の枠組の中でも必要となろう．超並列計算量のモデルとして代表的なのは PRAM (Parallel Random Access Machine)[1]であるが，これに対する批判的代案の一つとして，三次元空間での実現における制約を設計原理とした Spatial Machine[6] というモデルが出てきている．

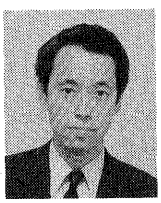
他分野から例をひくと，(計算機でない) 機械設計においても，機械の動作が高速化するにつれ，機構学だけでなく力学的な配慮が重要となってきた．この力学と力学的配慮とに対応するものの発展が，超並列処理の分野でも望まれる．

逐次アルゴリズムの文化を考えてみるに，工学的意義の追求によって得られた蓄積の比重はきわめて大きい．このような蓄積を図ることが，超並列処理が成熟するために必要で，それによって，並列処理による改革的アプローチと超並列処理による革新的アプローチの比較が，真に興味深いものとなろう．

参考文献

1) 小特集「並列アルゴリズムの現状と動向」，情報処理，Vol. 33, No. 9, pp. 1023-1066 (Sep. 1992).
2) 笠原博徳：並列処理技術，コロナ社 (1991).
3) 北野宏明：超並列人工知能，人工知能学会誌，Vol. 7, No. 2, pp. 244-262 (1992).
4) 二木厚吉他：理論は実践を導けるか，実践は理論を生かせるか？ 情報処理，Vol. 33, No. 3, pp. 272-289 (Mar. 1992).
5) Amdahl, G. M.: Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities, In *Proc. AFIPS Spring Joint Computer Conference*, Vol. 30, pp. 483-485 (1967).
6) Feldman, Y. and Shapiro, E.: Spatial Machines: A More Realistic Approach to Parallel Computation, *Comm. ACM*, Vol. 35, No. 10, pp. 60-73 (1992).
7) Halstead, R. H. Jr.: Parallel Symbolic Computing, *IEEE Computer*, Vol. 19, No. 8, pp. 35-43 (1986).
8) Ueda, K. and Morita, M.: A New Implementation Technique for Flat GHC, In *Proc. Seventh Int. Conf. on Logic Programming*, MIT Press, pp. 3-17 (1990).
9) Ueda, K. and Chikayama, T.: Design of the Kernel Language for the Parallel Inference Machine, *The Computer Journal*, Vol. 33, No. 6, pp. 494-500 (1990).
10) Ueda, K.: The Fifth Generation Project: Personal Perspectives, *Comm. ACM*, Vol. 36, No. 3, pp. 65-76 (1993).
11) Valiant, L. G.: A Bridging Model for Parallel Computation, *Comm. ACM*, Vol. 33, No. 8, pp. 103-111 (1990).

(平成5年5月12日受付)



上田 和紀 (正会員)

1956年生．1978年東京大学工学部計数工学科卒業．1986年同大学院情報工学博士課程修了．工学博士．1983年日本電気 (株) 入社．1985年～1992年(財)新世代コンピュータ技術開発機構へ出向．1993年4月より早稲田大学理工学部情報学科助教授．プログラミング言語の設計と実装，並行・並列処理，論理プログラミングの研究に従事．日本ソフトウェア科学会，人工知能学会，ACM, IEEE Computer Society, Association for Logic Programming 各会員．Journal of Logic Programming 言語・システム分野エディタ．