3P−6

並列処理におけるプログラミング支援環境

渡辺　高，坂尾　和浩，成田　良一，橋本　伸，神谷　幸男
富士通　(株)

1．はじめに

科学技術計算の分野でも，マルチプロセッサによる並列処理システムが実用化されてきている．しかし，そのようなシステムが広く一般ユーザに普及するほどには，マルチプロセッサの利用技術は成熟していない．依然として，ユーザによる並列処理プログラミングは困難なものであり，コンパイラによる自動並列化には限界がある．

このような現状を打破する方法として，並列処理プログラミング支援環境を整えることが，広く考えられている．ユーザ支援システムを構築することで，ユーザによる並列処理アプリケーション開発を容易なものにしよう，というアプローチである．

我々が開発したＨＰＰシステム[1][2]においても，このアプローチをとっている．本稿では，ＨＰＰシステム向けに構築した支援システムの概観について報告する．

2．Phil言語[1]

科学技術計算プログラム向けのシステムを考える場合，対象とする言語はFortran とするのが妥当である．しかし，Fortran は言語仕様が大きく，プロトタイプシステムの対象には適していない．そこで，我々はPhilという並列処理言語を用意し，支援システムはこの言語を対象として構成することにした．

Phil言語は，Fortran ライクな手続き型言語で，以下のような特徴がある．

(1) 並列処理，階層記憶利用の記述を持つ
(2) post/wait 等の同期制御文を持つ
(3) コンパクトな言語仕様

なお，Phil言語における並列処理記述は，手続きの並列呼び出しに限定してある．

3．プログラミングストーリー

支援システムを構成する場合，まずはその使われ方を考えなければならない．特に並列処理においては，アプリケーションプログラミングの手法が確立されていないため，プログラミングのプロセスを明確にすることから始めなければならない．

並列処理プログラミングの手法としては，アルゴリズム段階から並列性を意識するものも考えられる．しかし現段階では，既存の逐次プログラムを並列化することがほとんどであるため，並列プログラムの開発過程を概念的に，図1のように想定した．

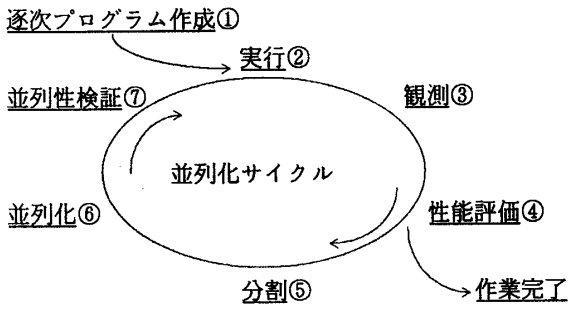

図1　プログラミングサイクル

この手順は次のようなものである．

① ユーザはまず，逐次で動作するプログラムを作成する．

② 実行する．このときにシステムがトレース情報を収集する．

③ 各プロセッサで，どのプロセスがどのように実行されたのかを観測する．

④ ③で観測した実行状況を定量的に評価する．③での観測結果と，ここでの評価結果から，並列化する部分を決定する．すでに満足な結果がでていたのなら，ここで並列化の作業は終了する．

⑤ 並列化したい部分を，手続き呼び出しに変換する．この作業はPhil言語に依存したものだが，並列実行の単位を決定する，という意味では一般的な意義を持つ．

⑥ 並列処理指定の記述を追加する．

⑦ ⑥で記述した並列実行部分が正しく実行されるか否かの判定を行う．

⑧ ⑦で，正常に並列実行可能の判定が出た場合には，並列化したプログラムを実行する（②に戻る）．正常実行不可能の判定がでた場合には，他の並列化候補を捜す（③に戻る）．もしくは同期制御文を挿入して，プログラムを実行する（②に戻る）．

なお，この手順は，Fortran を対象とした場合にも，適用できるものである．

4．並列処理プログラミング支援システムの実現

上述のストーリーに沿った支援システムを用意すれば，ユーザによる無理のない並列処理プログラム開発が可能になると考え，各作業を支援するツール群を作成した．その

A Programming Support System for Parallel Processing
Takashi WATANABE, Kazuhiro SAKAO, Ryoichi NARITA,
Shin HASHIMOTO, Sachio KAMIYA
FUJITSU, Ltd.

構成は以下のとおりである．

ｉ）実行表示ツール（図２）

各プロセッサごとのプロセス実行状況を，グラフ状に図示する．３の③を支援する．拡大，縮小等の機能を備えており，任意の時間帯を任意の幅で観測できる．階層メモリ間のデータ転送状況も合わせて表示する．

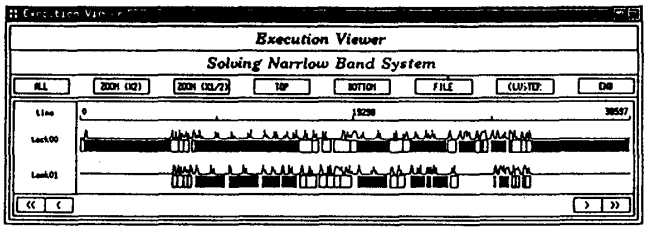

図２　実行表示ツール

ii）コスト解析ツール

プログラム内の各所で費やされた実行時間を表示する．３の④を支援する．効率評価，データ転送量評価等の機能を合わせ持っている．

iii）クラスタ化支援ツール

プログラム内の指定領域を手続き（クラスタと呼ぶ）呼び出しに変換する．３の⑤を支援する．変数の再配置（引数化，手続き内の局所変数化）機能を合わせ持っている．

iv）並列記述追加ツール

並列記述（pardo,parcall [1]）を付加する．３の⑥を支援する．

v）並列性検証ツール（図３）

並列実行記述部分が正しく並列実行されるか否かの判定を行う．３の⑦を支援する．正しい並列実行が不可能な場合には，アクセス競合が起こっている変数，及び場所を表示する．

これらのツールのうち，ｉ）ii）は実行時のトレース情報を入力としており，iii）v）は静的なデータフロー解析情報を入力としている．

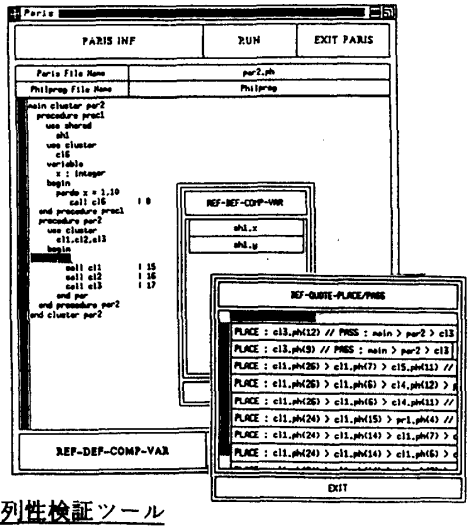

図３　並列性検証ツール

５．実プログラムへの適用例

本システムを幾つかの実プログラムに適用し，その効果を確認した．一例として，土木工学分野のニールセン橋の構造解析プログラム（ＮＩＥＬ，プログラム規模は1.3Kステップ程度）の場合について述べる．

プログラムＮＩＥＬは，あるＤＯループが実行時間の８割をしめ，かつそのＤＯループは並列化可能というものである．したがって，そのＤＯループを並列化することで，十分並列処理の効果を得ることができる．

まずは，本システムを用いずに（既存のツールは用いた），並列実行が完了するまでの作業を行った．作業に費やした時間は，担当者１名で１週間であった．特に，並列化の正当性の検証に多くの時間を割いた．

次に別担当者が，本システムを用いて並列化を行った．プログラム構造が単純であったこともあるが，この場合は一日で同等の結果を得ることができた．さらに，並列実行の効率等，支援ツールなしでは知りえない情報も知ることができた．

なお，各ツールは次のように用いられた．コスト解析ツールで並列化するＤＯループを決定し，クラスタ化支援ツールと並列性記述追加ツールで並列化を行い，並列性検証ツールでその正当性を確認し，並列化したプログラムを実行した後に，実行表示ツールで実行の様子を把握した．これは，概ね想定したストーリーに沿ったものである．

また，本稿では触れなかったが，各ツールは階層記憶利用に関する機能も持っている．データ配置に関する最適化においても，本システムの効果を確認することができた．

６．終わりに

本支援システムは，実運用を考えるといくつかの問題があるものの，プロトタイプとしては一応の成功を収めた．

今後，並列処理システムが普及していくにつれ，多くのユーザが並列マシンの性能を引き出せるような開発支援環境を整えていくことが，ますます重要になってくる．我々も，今回の経験をもとに，より使いやすい並列処理プログラミング支援システムを開発していきたい．

謝辞　ソースコードＮＩＥＬを提供してくださった京都大学工学部土木工学教室渡邊教授に感謝致します．

なお，本研究は通商産業省工業技術院大型プロジェクト「科学技術用高速計算システムの研究開発」の一環として新エネルギー・産業技術総合開発機構（ＮＥＤＯ）から委託を受けて，実施したものである．

【参考文献】

(1) 橋本，他：階層記憶型並列処理，第41回情報処理学会全国大会予稿集，1990

(2) 加藤，他：階層記憶型並列処理の制御ソフトウェア，第41回情報処理学会全国大会予稿集，1990