研究課題別研究評価

1. 研究課題名:　近未来の並列処理に適した実装用言語

2. 研究者名:　八杉 昌宏

3. 研究のねらい:
プロセッサ間結合方式の異なる様々な並列計算機上で、信頼性・再利用性・実行効率の高いソフトウェアを作成するには、並列処理のための高水準プログラミング言語が必要である。また近い将来製造可能となるオンチップマルチプロセッサなどにおける新しい細粒度のスレッド管理方式や通信・同期方式を高水準言語実装で利用する必要がある。一般に高水準言語コンパイラの実装には言語階層を用いる。特にC言語を経由することで、低レベルの最適化やレジスタ割り付けなどの処理をCコンパイラに担当させることができる。しかしながら、C言語は本来逐次処理用の言語であるため、並列処理のための実装用言語として用いるには記述性・実行効率ともに問題がある。本研究では、並列処理に適した実装用言語をC言語に拡張することにより開発する。特に最適化されたコード内での他のプロセッサからの要求と自プロセッサの処理の細粒度多重実行を実現する。

4. 研究結果及び自己評価:
(概要)　本研究では、C 言語を改良・補強したよりよい言語仕様をもつ並列処理に適した実装用言語を目指して並行協調拡張C言語 XC-cube を設計した。具体的には、そのような協調のための機能として(1)プロセッサ間で通信・同期をするための機能、(2)仕事（計算）の実行手順の調整機能、をどのように言語仕様に加えていけばよいかの研究を行った。また、XC-cube 言語の実装 (XC-cube 言語から機械語へのコンパイル手法)、つまり、XC-cube で追加した機能を効率よく実現する手法について研究した。さらにXC-cube言語の利用 (高水準言語からXC-cube言語へのコンパイル手法)についても研究を進めた。

(通信・同期機能)　従来の C 言語では、Dekker のアルゴリズムによる排他制御等が効率よく記述できないばかりでなく、データ領域とその領域への書き込み完了を表すフラグを用いるような単純な同期でさえ効率よく記述できない。C 言語のレベルで提供されるべき仮想機械のモデルが直接提供されておらず、逐次計算機用の C コンパイラが翻訳したコードが並列計算機上に動いているという変なモデルを C プログラマが考えなくてはならなかった。本研究では、通信・同期について、共有メモリの概念に基づいて XC-cube 言語レベルでこれを記述する方式を提案した。実行主体間で通信・同期をするための機能として、共有メモリ向けプリミティブを一通り設計した。(1)メモリ操作完了順序を指定するメモリバリア用プリミティブ、(2)粒度保証同期変数アクセスプリミティブ、を提供する。さらによく使う機能として(3)ロックのプリミティブを提供する。

通信・同期機能の実装については、最初のステップとしてあるいは普及の戦略として、これを組み込んだ XC-cube コンパイラを開発する前に、GNU C コンパイラの拡張機能を利用したプリミティブの実装を行った。Alpha、MIPS、Pentium、PowerPC、SPARC-V9、といった現在代表的なプロセッサアーキテクチャ/マルチプロセッサシステムの仕様の詳細なサーベイを行い、対象とした。

XC-cube 言語の通信・同期機能を利用することで、さまざまな高速な同期が C 言語上で記述できるようになった。

(実行手順の調整機能) 従来、呼び出し中に呼び出し元は見えなかったが、各プロセッサが、他のプロセッサとの協調のために、一旦、呼び出し元にさかのぼって自身の（元々設定した目標に従って）やりかけの処理さえも変更可能とする言語機能を提案した。本研究では、各手続きには、半固定化された高速な実行手順以外に「普段は使わないが、いざというときのための手続き」を含ませておく。この手続き中の手続き（入れ子手続き）が呼び出されたときに「元手続きがどういうふうにやりかけであったか」が分かるようになっていればよい。これで、普段は半固定された手順で高速に実行することを可能とするが、いざというときは、この入れ子手続きを呼び出すことで、やりかけの手続きでさえ見直しを可能とできる。C 言語の場合、手続きのことを関数と呼ぶが、入れ子関数は関数内で入れ子に定義された関数であり、その関数ポインタにより定義時の変数束縛環境を伴う入れ子関数のクロージャを利用することができ、呼出し元での変数にアクセスする手段が提供できる。

入れ子関数は GCC でも実現されているが、入れ子関数の関数ポインタについて、通常の関数ポインタとの互換性のためにいわゆるトランポリンという技法を用いていたり、他のプロセッサからも呼び出せるようにするために効率的なレジスタの使用が妨げられるなどオーバヘッドも大きい。XC-cube では、同等の機能を生成コストなどが少ない軽量クロージャとして実現する技法を開発している。これは、関数ポインタとは別扱いすることでトランポリンなどを不要とするとともに、実際に呼び出されるまでは変数の場所がレジスタ上であってもよいとするものである。

また仕事（計算）の実行手順の調整機能を利用した、遅延タスク生成に基づく負荷分散、マルチスレッドなど、高水準言語が提供すべき機能を実現する手法について研究した。マルチスレッドの実現については、以前から研究していた高水準言語であるオブジェクト指向並列言語 OPA のためのコード生成手法が変換手法の基礎として利用できた。遅延タスク生成に基づく負荷分散、マルチスレッドとも XC-cube 言語を利用すれば効率よく実現できることを示した。

(まとめ) 最終的に、得られた成果の一つは、共有メモリのモデルを C 言語のレベルで与えるにはどうするかということだった。この分野は、元々ハードウェアのレベルで議論されることが多かったが、モデルができあがってから、改めて調べてみると Java 言語においても Java 言語の提供する共有メモリモデルは何かということで研究がされている。ある意味では、似たようなモデルになっているが、逆にいえば、他の言語でもやはりモデルが必要であり、他の言語にも本研究の成果を活かせることがわかる。

もう一つ得られた成果の一つは、入れ子手続きを C 言語の機能として追加しておけば、C 言語のプログラムへうまく翻訳することで、マルチスレッドや自動負荷分散を提供する高水準言語の実装ができることに気が付いたことだった。このアイデアは非常に面白く、この研究で行った並列処理以外にも、ごみ集め、マイグレーション、スタットトレース、一級継続、例外処理、Non-strict 計算など広く応用できる。

今後は、XC-cube 言語処理系を完成させていくとともに、XC-cube 言語を実際に高水準言語の実装に用いたアプリケーションの記述や評価を行う予定である。

5. 領域総括の見解:

C 言語は、本来逐次処理用の言語であり、並列処理の実装用言語として用いるには記述性・実効性に問題がある。八杉昌宏は、並列処理に適した実装用言語を C 言語を改良・拡張することで、この問題の解決を図ったことは評価できる。特に、最適化されたコード内での他のプロセッサからの要求と自プロセッサの処理の細粒度多重実行を実現した。

6. 主な論文等:

八杉 昌宏, 馬谷 誠二, 鎌田 十三郎, 田畑 悠介, 伊藤 智一, 小宮 常康,
湯淺 太一, "オブジェクト指向並列言語 OPA のためのコード生成手法",
情報処理学会論文誌：プログラミング, Vol. 42, No. SIG 11 (PRO 12), November 2001.

*八杉 昌宏, 田畑 悠介, 小宮 常康, 湯淺 太一,
"共有メモリ向けプリミティブとその GCC を使った実現",
情報処理学会論文誌：プログラミング, Vol. 43, No. SIG 1 (PRO 13), January 2002.

*田畑 悠介, 八杉 昌宏, 小宮 常康, 湯淺 太一, "入れ子関数を利用したマルチスレッドの実現",
情報処理学会論文誌：プログラミング, Vol. 43, No. SIG ? (PRO 14), March 2002. (採録予定)

*八杉昌宏, 田畑 悠介, 小宮常康, 湯淺太一,
"入れ子関数を利用した動的負荷分散", 情報処理学会プログラミング研究会, January 2002.