要 旨

ストリーム駆動プロセッサと
その行列計算への応用に関する研究

山崎 弘法

現代の多くのマイクロプロセッサはプログラムカウンタを基準に命令実行を制御している。このため、プログラム実行に必要なデータの参照・更新をメモリに要求する方式をとる。よって、メモリとプロセッサ間においては、キャッシュブロックやページ/セグメントといったプログラム実行とは本質的に異質なデータ集合を転送しており、時間的／空間的局所性が低いプログラムの実行時には、メモリバンド幅を有効に活用できない。

本研究ではこの問題を解決するために、ストリーム内の個々のデータの可用性に基づいて順次命令をパイプライン的に駆動して実行するストリームフロープログラミング SFP (Stream-Flow Programming) モデルを考案した。この計算モデルを直接的に実現するマイクロアーキテクチャでは、メモリが能動的にプロセッサへ本質的に処理に必要なデータのみを供給する、メモリ主導型のマイクロアーキテクチャとなるため、上述の問題を原理的に解決できる可能性がある。よって、著者らはこれを、ストリーム駆動プロセッサ SDP（Stream-Driven Processor）アーキテクチャと命名して、その検討を行ってきた。

本稿では、SFP モデルとそれを直接的に実現する SDP の構成法を述べる。さらに、SDP の LSI 設計を通した評価、ならびに、SDP を行列計算へ応用した場合の性能評価結果を示す。

キーワード　並列処理, 並列プログラミングモデル, データ駆動/ストリーム駆動プロセッサ

Abstract

A Study on Stream-Driven Processor and Its Application to Matrix Computations

Hironori YAMASAKI

In most of modern microprocessors, instruction execution control scheme is based on the program counter so that the processor issues read/write requests of a necessary set of data to the memory module, e.g., cache or main memory. Here, transferred data sets such as cache block, segment, or page are essentially different from data sets required from the program. Therefore, in case of programs with low temporal and spatial locality, bandwidth between processor and memory cannot be utilized efficiently. To solve this problem, we proposes a stream flow programming (SFP) model where instructions are driven in pipelined manner based on the availability of individual data stream. Furthermore, as a micro-architecture realizing SFP directly, we designs stream-driven processor (SDP) architecture in which a kind of functional memory supply essential data set to the processor and the processor passively executes instructions according to the supplied stream of data.

In this paper, we introduce the SFP and SDP. After that, preliminary evaluation results through LSIdesign of a simple SDP and its application to a matrix computation program are discussed.

key words parallel processing, parallel programming, data-driven/stream-driven processor.