ALU-Net を用いることによるデータ移動の効率化

吉瀬 謙二，中村 友洋，辻 秀典，安島 雄一郎，田中 英彦
東京大学大学院工学系研究科

1 はじめに

並列性の利用に関する制約から、スーパースカラ・プロセッサのアーキテクチャによる性能向上に限界が見え始めている。スーパースカラ方式においてより多くの並列性を抽出するためには、命令ウィンドウのサイズを大きくする必要があるが、このことはより多くの並列性を利用する利点に対してサイクル時間が増大するという欠点をもたらし十分な性能向上につながらない。このような背景から、サイクル時間の増大を抑えながら、命令ウィンドウのサイズを拡大することで多くの並列性を抽出し、*ALU-Net* と呼ばれる多数の ALU のネットワークに命令を割り付けて処理を進める Very Large Data Path(VLDP) アーキテクチャ[1] の開発をおこなっている。本稿では、最小限のスイッチとラッチを介し 2 つの ALU 間でデータを転送するという ALU-Net の利点について議論する。

2 VLDP アーキテクチャと ALU-Net モジュール

ALU-Net を説明する前に、VLDP アーキテクチャの構成と、ALU-Net の位置付けを整理する。

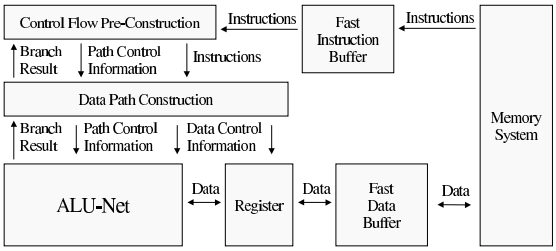

図 1: Very Large Data Path Architecture

VLDP アーキテクチャは、近い将来に必要となる命令やデータを格納する **Buffer モジュール** (*Fast Instruction Buffer*,*Fast Data Buffer*, *Register* という 3 つの要素から成る。)、複数のパスから投機的に命令をフェッチする **Control Flow Pre-Construction モジュール** [2]、フェッチした命令のデータ依存関係を解析しデータ移動のタイミングや実行のタイミングを決定する **Data Path Construction モジュール**、そして VLDP アーキテクチャの処理装置である **ALU-Net モジュール** という 4 つのモジュールから成る。図 1 に VLDP アーキテクチャのブロック図を示す。

Intra data access of ALU-Net on VLDP architecture
Kenji KISE, Tomohiro NAKAMURA, Hidenori Tsuji, Yuichiro AJIMA, Hidehiko TANAKA
Graduate school of Engineering, The University of Tokyo

2.1 ALU-Net モジュール

ALU-Net は多数の ALU とそれを接続するネットワークから成る。1 次元状に ALU を配置する ALU-Net の例を図 2 に示す。この ALU-Net の構成はハードウェアの利用率を高く保つことが可能だが、ALU の数が増加するに従いスイッチ・マトリックスが複雑となり、データ供給がボトルネックとなる。

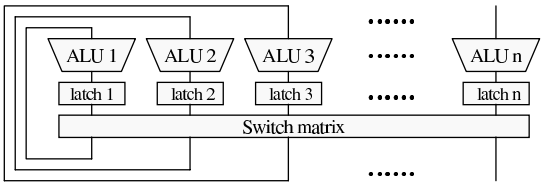

図 2: ALU-Net implementation of one dimension array

VLDP アーキテクチャでは、データ参照の局所性を積極的に利用する 2 次元状の ALU 配置の利用を検討している。この構成より、ある ALU における実行の結果を他の ALU の入力として必要最低限のスイッチとラッチを経由して接続することが可能となる。幅 3、高さ 2 で配置した 6 個の ALU を自由度の異なる 3 つのネットワークで接続した ALU-Net を図 3 に示した。ここで自由度 (degree) とは ALU に接続されている latch の数を意味し、図 3 の左のグラフから自由度 1、2、3 の ALU-Net に対応する。

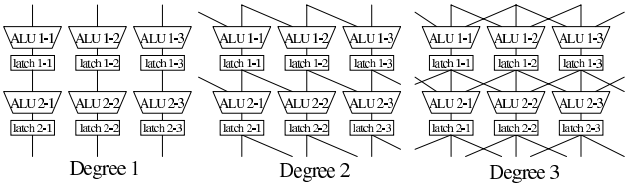

図 3: Three examples of ALU-Net

図 3 右に示す自由度 3 の ALU-Net において、例えば latch1-1 に格納されているデータを利用する命令を実行する場合には、ALU2-3, ALU2-1, ALU2-2 のいずれかでその演算を処理すれば、必要とする latch1-1 のデータをレジスタを経由することなく利用できる。今 ALU2-1 で計算が完了したとすればその演算結果は latch2-1 に格納され、次に ALU2-1 で実行が完了するまでこのデータは接続された ALU(ALU1-3, ALU1-1, ALU1-2) で利用可能となる。ALU-Net における、latch から ALU へのデータ・アクセスを *intra data access* と呼び、Register を経由するその他のデータ・アクセスを *inter data access* と

呼ぶことにする。VLDP アーキテクチャでは intra data access を用いることで ALU-Net と Register で転送しなければならないデータ量の削減を目指す。

3 ALU-Net における Intra Data Access の割合

本章では、ALU-Net の自由度を変化させ、全データアクセスに対する intra-data access の割合を評価する。評価は SPEC92 で用いられる cc1, compress, espresso, sc に合成ベンチマークの dhrystone を加えた 5 種類を対象に、各プログラム 1 千万命令のトレース •データを用いておこなう。アーキテクチャは SPARC architecture version 8 である。評価において、理想的なメモリ •システムと理想的なフェッチ機構の存在を仮定する。理想的なフェッチ機構を仮定しているため、フェッチは単一パスからおこない、命令ウィンドウの全てのエントリは正しいパスからフェッチされた命令で満たされているとする。

3.1 ALU-Net の構成と割り当て戦略

命令を ALU に割り当てる場合には、できる限り intra data access の割合が増えるように割り付ける必要があるので、各ラッチに格納されているデータのタグと実行する命令のオペランドのタグを比較し、一致した場合にはそのラッチに接続されている ALU の任意の一つに命令を割り付ける。もし、接続されている全ての ALU に命令が割り付けられている場合、または、必要とするデータがどのラッチにも存在しない場合には適当な位置に命令を割り付けることにする。

今回評価するネットワークの構成は図 3 に示した自由度 1,2,3 の ALU-Net と自由度を ALU の数だけ持つ ALU-Net (complete network) の 4 つで評価する。ALU の数は 16(4x4), 25(5x5), 36(6x6), ..., 100(10x10) という 7 つの構成を用いる。1 サイクルに割り当てることができる命令の数は命令ウィンドウのサイズにより決まる。命令ウィンドウのサイズは 20, 40 という 2 つの設定で評価する。

3.2 評価結果

命令ウィンドウのサイズを 20 とした時の評価結果を図 4 に、命令ウィンドウのサイズを 40 とした時の評価結果を図 5 に示す。図 4,5 の横軸は ALU の数を示し、縦軸に全データ •アクセスに対する intra data access の割合を示す。図 4 と図 5の比較より ALU の数が大きいところでは、命令ウィンドウのサイズは intra data access の割合にそれほど影響を与えないないことがわかる。ネットワーク構成で比較すれば、完全なネットワークをもつ ALU-Net では、81 個の ALU-Net を用いることで 80%以上のデータ •アクセスが intra data access として ALU-Net 内で吸収できることがわかる。しかし、自由度 3 の現実的な ALU-Net では割合が 60%程度に減少する。自由度 2 と自由度 3 の違いは数%と少ないが、自由度 1 の ALU-Net では intra data access の割合は 50%まで減少することがわかる。

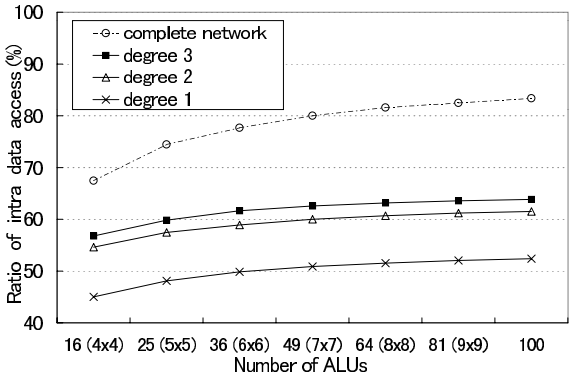

図 4: Ratio of intra data access (window size 20)

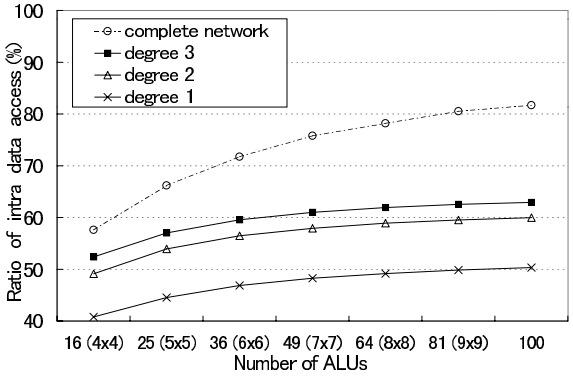

図 5: Ratio of intra data access (window size 40)

4 まとめ

最小限のスイッチとラッチを介し 2 つの ALU 間でデータを転送するという ALU-Net の利点を説明し、幾つかの ALU-Net 構成において intra data access として ALU-Net 内で吸収されるデータ •アクセスの割合を求めた。評価の結果、命令ウィンドウのサイズを 40 とした時、100 個の ALU からなる自由度 3 の ALU-Net において intra data access の割合は 62%という結果を得た。今後、今回の評価で理想化していた Buffer モジュールや分岐予測機構のシミュレータと組み合わせて評価をおこなっていく。

本研究の一部は文部省科学研究費 (一般研究 (B) 課題番号 07458052 「大規模データパスプロセッサの研究」) による。

参考文献

[1] 中村友洋, 吉瀬謙二, 辻秀典, 安島雄一郎, 田中英彦. 大規模データパスプロセッサの構想. 情報処理学会研究報告 97-ARC-124, pp. 13–18, June 1997.

[2] 辻秀典, 中村友洋, 吉瀬謙二, 安島雄一郎, 田中英彦. 大規模データパス •プロセッサにおけるフェッチ機構の検討. 情処研報 97-ARC-126, pp. 37–42, October 1997.