1D-7

高並列推論エンジン実験環境PIEEE上へのFLENGの実装

小池 汎平, 島田 健太郎, 朝海 寛, 田中 英彦
(東京大学 工学部)

1. はじめに

高並列推論エンジン実験環境PIEEE[1,2,3]の推論コプロセッサVector-UNIRED[3]は、PIEの基本処理である単一化及び縮退をパイプライン処理により高速実行することを目的として設計された。Vector-UNIREDは、ゴールと定義節を2本のベクトルとみたて同時にパイプラインに流し込むことにより論理型プログラム言語を高速に直接実行するシリコンインタプリタである。一方、我々は現在PIEEE上にCommitted-Choice言語FLENG[4]の実装を進めている。FLENGの処理系はC言語で記述し、PIEEEのCPUで実行させている。これは、Vector-UNIREDはOR並列処理向きゴール書き換え処理しかサポートしていず、FLENGを実装するためにVector-UNIREDを十分活用することができないためである。そこで、本稿では、FLENGの実行に適した新しいVector-UNINIREDの構成法について検討してみる。

2. 現在のVector-UNIREDの問題点

Vector-UNIREDで、FLENGを実行する場合に考慮しなければならない点として、次のことが挙げられる。

1. 単一化の最中に、他IUにあるデータをアクセスしなければならない場合がある。しかし、他IU内のデータのアクセスにはパイプラインサイクルに比べ比較的長いターンアラウンド時間を要する。
2. FLENGを始めとするCommitted Choice言語では、グローバルな変数に対する書き出しをactiveなunify述語によって行なう。この処理は、頻度の高いものであり、効率的にサポートする必要がある。
3. 単一化の最中にサスペンドする可能性がある。ゴールのスケジューリングによりサスペンド頻度を下げることも重要だが、UNIREDとしてもサスペンドの際のコンテクストスイッチングを効率的にサポートしなければならない。

また、現在のVector-UNIREDの一般的な問題点として、

1. UNIRED自体がインデクシングをサポートしていないため、常に、PIEEEのCPUが、インデクシングを行ない、GF、DTを指すポインタをUNIREDに供給する必要があった。このため、縮退が終了してから、次の単一化が開始されるまで、かなり長い時間を要しパイプラインを有効に利用できなかった。
2. 縮退処理によるゴールフレーム生成の際に必ずメモリを消費してしまう。
3. ローカル変数をメモリ上にとったため、変数のアクセスのたびに必ずメモリアクセスが必要となった。

などがあった。

3. Vector-UNIREDのプリミティブオペレーション

FLENG向きVector-UNIREDは、FLENGの実行に必要な次の基本処理を行なう。

1. Passive Unification
2. Active Unification
3. Reduction
4. Overwrite Reduction

Passive Unificationは、定義節の頭部での単一化を行なうものである。Active Unificationは、Commit後に行なう外部変数への書き出しを伴う単一化である。Reductionは、定義節のボディ部のゴールを縮退し新しいゴールフレームを生成する処理である。Overwrite Reductionは、リテラル領域を再利用しながら新しいゴールフレームを生成する処理であり、ゴールフレーム用メモリ領域の消費速度を低下させるとともに、上書きされるゴールフレームが、生成するゴールフレームと同じ情報を持っている場合そのデータを書き出さず、ローカルメモリのアクセストラフィックを低下させる働きをもつ。

また、定義節中の変数はUNIRED内のレジスタに格納されることが望ましい。

4. DMAとの協調動作と複数コンテクスト混合処理

現在のVector-UNIREDは他IU内のデータ(変数、構造データ)を直接アクセスする機能を持っていない。このような場合

Implementaion of FLENG on PIEEE
Hanpei KOIKE, Kentaro SHIMADA, Hiroshi ASAKAI, Hidehiko TANAKA
University of Tokyo

はDMAに依頼し、ネットワークを経由してデータをフェッチしてこなければならない。しかし、単一化の最中に他IU内のデータのアクセスをDMAに要求した場合、データが到着し処理を続行できるまでに、パイプラインサイクルに比べかなり長い時間待たなければならない。そこで、この間Vector-UNIREDのパイプラインを無駄にしないためにDMAに通信処理を要求したら、Vector-UNIREDは直ちにコンテクストを切り換え別のゴールの処理を開始することにする。

5. Active Unifyアノテーションの導入

グローバル変数に対する書き出しをわざわざunify述語のゴールを縮退して実行するのは無駄な処理が伴う。そこで、定義節の頭部に、Commit後Active Unifyを行なうことを指示するActive Unifyアノテーションを導入する。Passive Unification実行中にこのアノテーションのついたセルのペアに出会う毎にこれらのセルをUNIRED内に蓄えてゆき、Passive Unificationが成功し、定義節がCommitされた後に、蓄えておいた情報をもとにして、Active Unificationをまとめて実行することにする。

Active Unifyアノテーションの導入により、1回のInference毎に少なくとも平均1回以上実行されるActive Unification処理を、unify述語を介さずに直接連続実行することができ、オーバヘッドをなくすことができる。

6. インデクシング条件キャッシュ

Vector-UNIRED内で、同一の述語が末尾再帰呼び出しによって繰り返し実行されているような場合、縮退が終ってから再び単一化が起動されるまでの時間が長いと、Vector-UNIREDのパイプラインが有効に利用できない。

この時間の大部分は、出来上がったゴールをもとにして、次に適用可能な定義節を決定するインデクシング処理の時間である。しかもここで考えているような末尾再帰呼び出しによる繰り返し処理では、繰り返しが続くあいだ同一のインデクシング条件が成立すると考えられる。

そこで、最後に行なったインデクシングの条件をVector-UNIRED内のキャッシュに保存し、UNIREDで継続実行することになったゴールの縮退と並行してインデクシング条件の一致(述語が同じで、第一引数の型も同じかどうか)を判定することにする。条件が成立した場合、直ちに新しいゴールと同じ定義節との単一化を準備すれば良い。この機能により、縮退から次の単一化への移行時間を最小化できる。

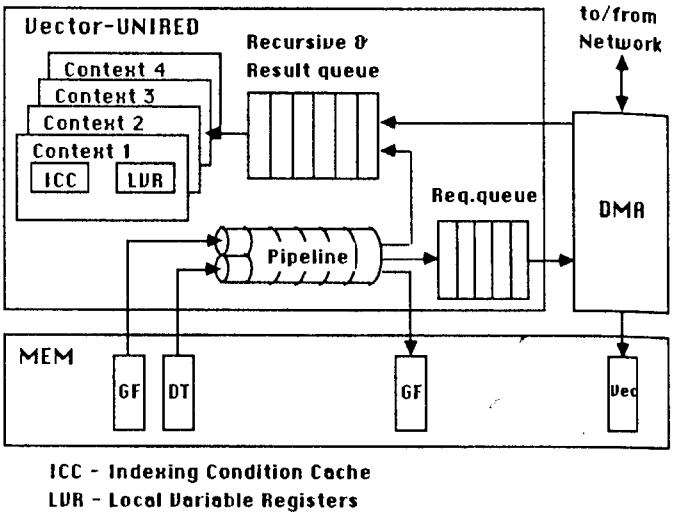

Figure 1: Vector-UNIREDのイメージ

7. FLENG向き Vector-UNIREDのイメージ

以上の検討に基づいた、FLENG向きVector-UNIREDのイメージを示すと上図のようになる。

8. おわりに

高並列推論エンジン実験環境PIEEEの推論コプロセッサVector-UNIREDをCommitted Choice言語FLENGに適応させるにはどうすればよいかについて検討した。この結果に基づき、FLENGの実行に適したVector-UNIREDの構成のイメージを示した。今後、本検討に基づき、1セル当たりの単一化及び縮退処理を平均1.5クロック以内の速度で行なうことを目標に、より詳細な仕様を決定していきたいと思っている。

References

[1] 小池, 山内, 田中, "高並列推論エンジン実験環境PIEEEの概要", 第33回 情報処理学会 全国大会, 5B-5 (1986).

[2] 小池, 山内, 野田, 田中, "高並列推論エンジン実験環境PIEEE - 全体構成 -", 第34回 情報処理学会 全国大会, 4P-3 (1987).

[3] 野田, 小池, 山内, 田中, "高並列推論エンジン実験環境PIEEE - 推論ユニット -", 第34回 情報処理学会 全国大会, 4P-4 (1987).

[4] Nilsson M. and Tanaka H., "FLENG Prolog - Turning supercomputers into Prolog machines", Proc. Logic Prog. Conf. '86, Tokyo, June, 1986.