式評価順序の最適化による微視的並列度向上

2F−4

竹内陽一郎, 境隆二

(株) 東芝　情報処理・機器技術研究所

1　はじめに

並列命令実行を前提にした場合、構文木をなるべくバランスするよう式の評価順序をきめ、木の高さを出来るかぎり低くした方が、全体のクリティカルパス長が短くなり高速となる。このような最適化（以下、ハイトリダクションと呼ぶ）は論理回路設計や、大規模並列処理の分野では、重視されてきたが、命令セットレベルでの並列度をあげるための最適化手法としてはあまり取り上げられていない。これは、従来、ハイトリダクションの適用が一つの式、あるいは、基本ブロック内に限定して考えられていたため、実質的な性能向上があまり大きくなかったためと考えられる。

しかし、現在では、データフロー解析、ループアンロール等の手法と組み合わせることにより、従来考えられていたよりもはるかに大きな式を並列化の対象とできる。このため、ハイトリダクションの適用による性能向上は、従来の数１０％のレベルから、数倍のレベルまで来ていると考えられ、並列処理の性能を左右する重要な技術になると思われる。以下、命令セットレベルでの並列最適化を前提に、アルゴリズムの検討とその拡張および問題点について延べる。

2　ハイトリダクションの基本的なアルゴリズム

並列最適化では、命令スケジュールのために、データフローダイアグラムを作成するため、これを利用すれば、新たに構文木を作成する必要はない。

まず、データフローダイアグラムのクリティカルパスに添って、命令をスキャンし交換、結合法則によってクリティカルパス長を削減できる箇所をさがす。交換、結合法則を適用して演算順序を変更する。このさい、対象となる命令が他のノードからも参照されている場合、コピーを作成する。パス短縮の結果、別の箇所が全体のクリティカルパスとなった場合、対象パスを変更する。以下、クリティカルパス長が削減できなくなるまで繰り返す。

分配法則を適用した結果、一方の演算が定数評価、共通不変式削除でなくなるケースがありうる。この場合、実質的に交換法則と同じであり、この枠組みに取り込める。

3　分配法則を含む拡張

交換、結合法則による基本的なアルゴリズムだけでは、異なる演算が交互に現われる状態に陥り、あまり効果が現われない。例えば、ＬＦＫ (Livermore Fortran Kernel) の１ｏｏｐ５をアンロールして得られる式は、２のアルゴリズムでは、全く短縮できない。

この場合、分配法則を適用すれば、結合法則が適用できるようになり、さらにパスの短縮を進めることができる。しかし、分配法則の適用は、次の問題を引き起こす。

（１）分配法則を適用していくと、演算数が極端に増加する。並列実行できる命令数が限られている、実際のマシンでは、逆に性能が落ちることもありうる。

（２）分配法則を適用する場所、タイミングによって結果が異なり、最適化が難しい。特に、分配法則の適用を制限した方が、パス長が最短になるケースがありうる。

これらの問題に、現実的に対処する方法として、以下のアルゴリズムが提案されている。（参考文献２）以下、この手法を２分分配法と呼ぶ。

（１）それを起点とする部分式に式全体の約半分の項が含まれるノードを捜す。

（２）分配法則、結合法則を交互に適用して、このノードを木の根まで引き上げる。

（３）引き上げたノードで式を二つにわけ、それぞれの部分式に、（１）からを再帰的に適用していく。葉にいきついたら終了。

4　条件処理を含む式の処理

条件処理を、条件分岐を使用せず、条件選択演算に置き換えることで、このようなケースでもハイトリダクションを適用できる。データフロー解析を行なえば、命令の副作用（特に、演算エラーや、メモリフォルトなど）が無視できる範囲内で、条件分岐の条件選択演算への置き換えが可能である。条件選択演算は、ハード命令でサポートされていることが望ましいが、マスク生成と論理演算とで合成することもできる。例として、ＬＦＫ　１ｏｏｐ２４をアンロールした結果は、条件選択演算を用いると図１のようになる。

Optimization of expression evaluation order to enlarge fine-grained parallelism

Yoichiro TAKEUCHI , Ryuji SAKAI

TOSHIBA CORPORATION

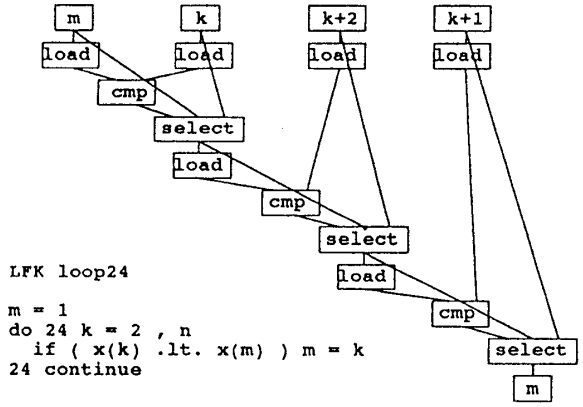

図１　LFK loop24 の条件選択演算によるアンロール

条件選択演算に対し、図２の代数法則を適用することで、他の演算同様、ハイトリダクションの枠組みの中に組み入れられる。これによって、条件処理が並列化の妨げにはならなくなり、性能向上に寄与するだけでなく、ハイトリダクションの適用できる範囲が大幅に拡大する。従来、条件分岐が多発するため並列化が非常に困難とされてきたソーティングプログラムなどが、これによって、並列化のターゲットになりうる可能性がある。

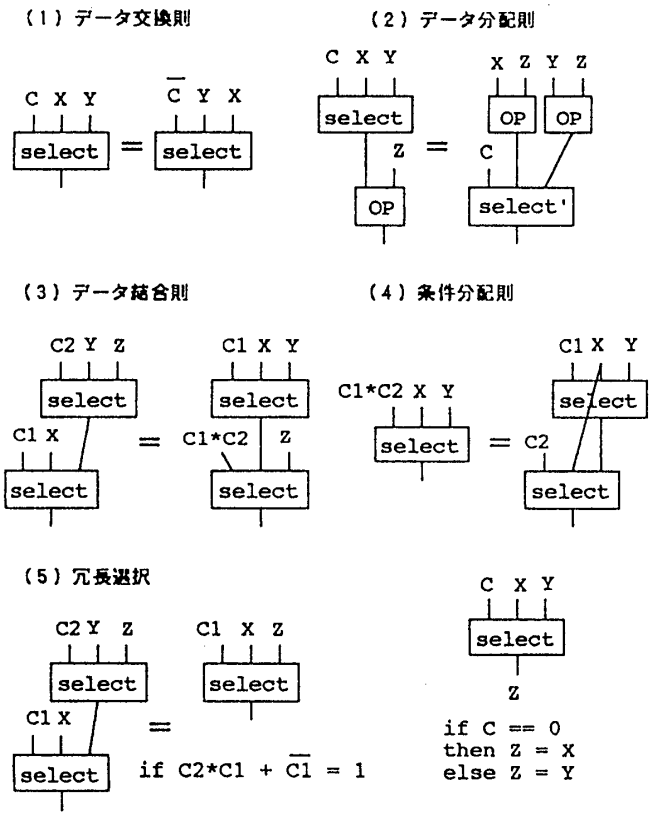

図２　条件選択式の変形規則

例として、ＬＦＫ　ｌｏｏｐ２４にハイトリダクションを適用した結果を図３に示す。従来、人手でおこなわざるを得なかった、ｏ（ｎ）アルゴリズムからｏ（ｌｏｇｎ）アルゴリズムへの変換が、機械的にできる点に、特に注目する必要がある。ただし、選択条件の合成で、条件選択後の値の比較を用いず、完全に論理演算に展開してしまうため、比較演算の数がｏ（n^2）になってしまう点が実用上解決すべき問題として残る。（これは、条件選択演算に分配法則を適用していることから生じている。）

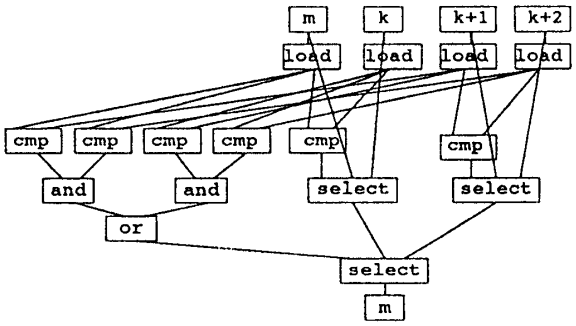

図３　ハイトリダクション適用後の LFK loop24

5　今後の課題

ＶＬＩＷなど実際のマシンでは、ハード的な並列度は比較的小規模（～１０）であるため、分配法則を適用した際の、演算数の増加をどう抑えるかが実用上大きな問題となる。２分分配アルゴリズムとハイトリダクション後の共通不変式削除でかなり抑えられると考えられるが、さらに検討が必要である。

条件処理を論理演算で置き換えた場合、条件が成立しない場合の処理も同時に実行されるため、条件のネストが深いと、効率が極端に落ちる。これに対処するため、ハイトリダクション後、命令スケジューリングの段階で、再び条件分岐構造にもどすことが考えられるが、具体的なアルゴリズムは検討中である。

6　おわりに

ループアンロール、条件処理の論理演算化、ハイトリダクションの組み合わせが、命令セットレベルでの並列度を上げるうえで、極めて強力な手段であることがわかった。今後、この枠組みをさらに拡大して行くことで、大規模並列処理マシンへの適用など、並列処理技術のブレークスルーとなる可能性もある。

参考文献

（１）竹内，境：微視的並列度向上のための中間コード最適化戦略，情報処理学会第４３回全国大会予稿集，１９９１

（２）R.Brent,"The Parallel Evaluation of General Arithmetic Expresions", Journ. of ACM, Vol.21, No.2, pp.201-206, Apr.1974