6W-5

並列処理システム－晴－における実行時エラーの処理

萩本猛　草野義博　山名早人　村岡洋一
（早稲田大学　理工学部）

1．はじめに

我々は，科学技術計算用並列処理システム－晴－（-HARRAY-：Hybrid ARRAY）を提案している[1]．－晴－は，科学技術計算用にFORTRANで記述されたプログラムを高速に実行することを目的とし，要素プロセッサを1024個持つ並列処理システムである．－晴－の実行方式は，プログラムをコンパイル時にマクロブロックという単位に分割し，マクロブロック間をコントロールフロー，マクロブロック内をデータフローで処理を行うＣＤフロー方式である[2]．

データフローのプログラムでは，後述するゲート後置を行うと，計算機資源が無限にあると仮定したとき，実行速度が約３倍向上することを確認している[3]．しかし，計算機資源は有限であるため，－晴－では，プログラムの並列度が計算機資源よりも小さい部分でゲート後置を行い，この部分の実行速度を向上させる．

しかし，制御ゲートが実行時エラーを回避させるために設けられているとき，ゲート後置を行うと，その先行評価部分で，ゲート後置が原因の実行時エラーが発生する場合がある．この実行時エラーは，ユーザのプログラムの誤りが原因でないため，ユーザに報告することはできない．したがって，ゲート後置が原因となりえる実行時エラーが発生したとき，その発生原因がゲート後置であるのかプログラムの誤りであるのかを判断する必要がある．

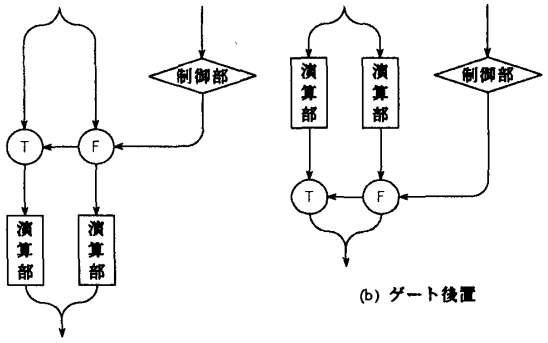
本稿では，ゲート後置が原因となりえる実行時エラーが発生したとき，その発生原因を判断する方式を提案する．

2．実行時エラーの処理

2.1 実行速度とゲート後置

データフローのプログラムでは，制御ゲートを演算部分の後方に配置すると，プログラムの並列度が増加し，計算木の高さが減少する（図１参照）．この手法，すなわち，制御ゲートを演算部分の後方に配置することを，ゲート後置と呼ぶ．ゲート後置は，演算されるノード数を増加させるが，増加したプログラムの並列度よりも計算機資源が十分にあるとき，実行時間を短縮させる．

－晴－では，プログラムの並列度が計算機資源よりも十分小さい部分で，実行時間短縮のために，コンパイル時にマクロブロックの範囲内でゲート後置を行う．

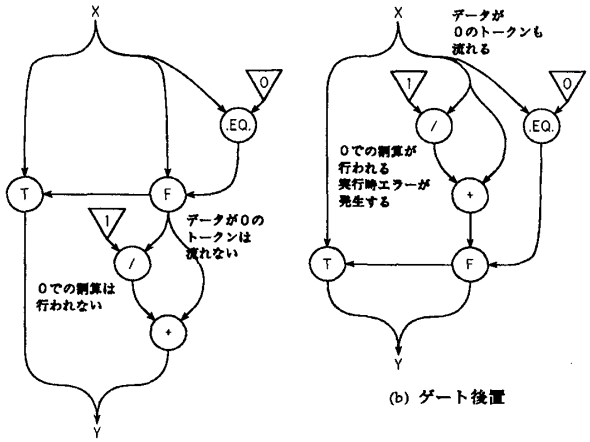


(a) 通常　　(b) ゲート後置

図１ 制御ゲートを含むデータフローグラフ

2.2 ゲート後置による実行時エラー

制御ゲートが実行時エラーを回避させるために設けれれているとき（例：割り算における割る数のチェック），ゲート後置を行うと，その先行評価部分で，回避させるべき実行時エラーが発生する場合がある（図２参照）．この実行時エラーは，発生原因がユーザのプログラムの誤りではなくコンパイラが行ったゲート後置であるため，ユーザに報告することはできない．したがって，ゲート後置が原因となりえる実行時エラーが発生したとき，その発生原因がゲート後置であるのかプログラムの誤りであるのかを判断し，前者ならば処理を続け，後者ならばユーザに実行時エラーを報告する必要がある．



(a) 通常　　(b) ゲート後置

図２ 制御ゲートを含むデータフローグラフの例
(IF (X.EQ.0) THEN Y=X ELSE Y=1/X+X)

A Run-time Error Handling Scheme
for Parallel Processing System -Harray-
Takeshi HAGIMOTO,Yoshihiro KUSANO,
Hayato YAMANA,Yoichi MURAOKA
WASEDA University

ゲート後置が原因となりえる実行時エラーは，要素プロセッサで発生するエラーであり，種類として以下に示すものがある．

①演算結果のオーバフロー
②不当な演算（０での割算など）
③メモリアクセスエラー（アドレスの誤りなど）

－晴－では，このような種類の実行時エラーが発生したとき，次に述べるエラートークンという特殊なトークンを使用して，実行時エラーの発生原因を判断する．

2.3 エラートークンによる判断方式

ゲート後置が原因となりえる実行時エラーが発生したとき，その発生原因を判断するために，エラートークンという特殊なトークンを使用する方式を提案する．この方式は，実行時に処理の流れと共に，自然に実行時エラーの発生原因を判断する方式である．以下にエラートークンの具体的な使用方法を述べる．

ゲート後置が原因となりえる実行時エラーが，ある演算ノードで発生すると，そのノードからエラートークンを出力させる．エラートークンは，エラートークンであることを示すフラグが立ち，実行時エラーが発生したノード番号と，発生した実行時エラーの種類を示すコード番号を持ち，アーク上を通常のトークンと同様に流れる．

エラートークンによって次の演算ノードが発火すると，入力されたエラートークンをそのまま出力させ，エラートークンを伝達させる（図３参照）．

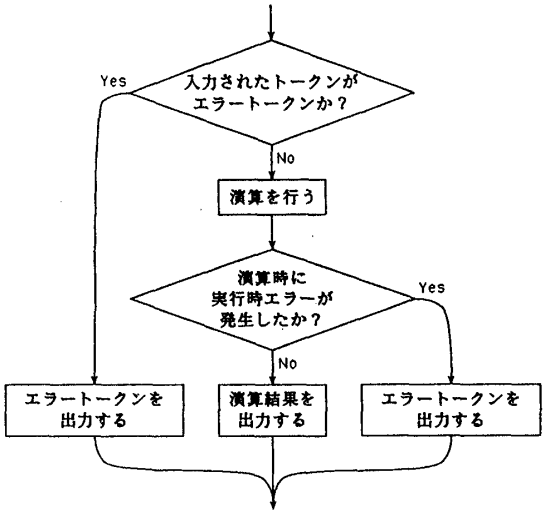
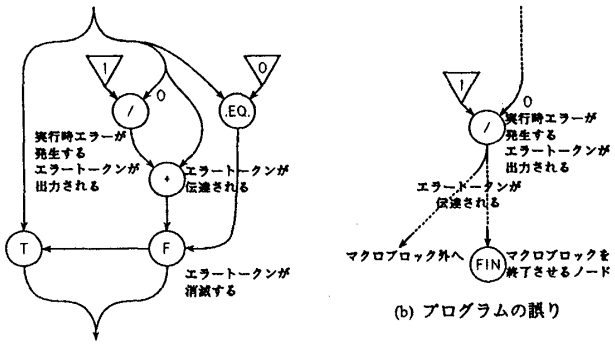


図３ 演算ノードでの処理

エラートークンが，マクロブロック外に出力されるか，マクロブロックを終了させるノードを発火させたとき，ユーザに実行時エラーを報告する．

この方式を用いると，コンパイラが行ったゲート後置が原因の実行時エラーが発生したとき，エラートークンを必ず制御ゲートで消滅させるようにコンパイルを行うことにより，処理を続けることができる（図４.a参照）．ユーザのプログラムの誤りが原因の実行時エラーが発生したとき，エラートークンは，制御ゲートで消滅せずに，マクロブロック外に出力されるか，マクロブロックを終了させるノードを発火させるため，ユーザに実行時エラーを報告することができる（図４.b参照）．



(a) ゲート後置　　(b) プログラムの誤り

図４ 実行時エラーの発生原因の判断の例

３．おわりに

－晴－では，実行速度向上のためにゲート後置を行うので，ゲート後置が原因の実行時エラーが発生する場合がある．この実行時エラーは，プログラムの誤りが原因の実行時エラーと区別する必要がある．そこで，ゲート後置が原因となりえる実行時エラーが発生したとき，エラートークンという特殊なトークンを使用して実行時エラーの発生原因を判断する方式を提案した．

この方式を用いると，制御ゲートが実行時エラーを回避させるために設けられているときでも，実行速度向上のために，ゲート後置を行うことができる．

今後は，次の２つの条件を満たすコンパイル方式について検討を進めていく．

①ゲート後置によるエラートークンをすべて制御ゲートで消滅させる．
②プログラムの誤りによるエラートークンをすべて制御ゲートで消滅させない．

参考文献

1) 丸島：“ 並列処理システム－晴－の実行方式 ”，情処研報，88-CA-69-2，pp.9-16（1988）
2) H.Yamana：“ System Architecture of Parallel Processing System -Harray- ”，Proc. of Int. Conf. on Supercomputing，pp.76-89（1988）
3) 萩原：“ 並列処理システムにおけるＦｏｒｔｒａｎプログラムのマクロブロック化の評価 ”，情処第35回全大，1C-4，pp.99-100（1987）