8L−5

先行評価に適したマクロタスク生成手法

山名早人　安江俊明　石井吉彦　村岡洋一
早稲田大学理工学部

1．まえがき

従来提案しているマクロタスク間先行評価方式[1]におけるマクロタスク構成方法について報告する．先行評価方式とは，プログラム中の条件分岐文を越えて実行を進める方式である．

マクロタスク生成の目的は，(1) 変数の2重定義に伴う副作用問題の回避，及び (2) 仮実行（投機的実行）に必要なプロセッサ数の削減の2点である．先行評価によって生じる副作用は，先行評価中に，同一データに対する2重定義が行われることによって生じる．本稿では，2重定義を回避するために，各マクロタスクへのデータ依存関係が制御によらず一意になるようにマクロタスクを構成する．次に，実行時のプロセッサ数を削減するため，マクロタスク生成においては，データ依存と制御依存の関係を用いて，マクロタスクを融合した場合も，先行評価の効果を失わない部分を1つのマクロタスクとする．これは，従来のマクロタスク生成手法が制御依存のみを考えていたのに対し[2]，データ依存を考えた生成手法として新規性を持つ．

2．対象モデル

2．1　対象プログラム

以下の条件を満たすプログラムを対象とする．

(1) フローグラフ[3]が簡約可能[3]．
(2) 副プログラムはインライン展開済．
(3) エラー回避のための条件分岐は先行評価の対象外．

2．2　マクロタスクの定義

マクロタスク（以下MT）とは，プログラムの一部分であり，制御の入点となる文をその先頭に1つ持つ．従って，マクロタスクの最小単位は，単一の文となる．また，マクロタスク集合は，その制御フローグラフが非循環有向グラフで表せるとする．プログラムを非循環の複数のグラフに変換する手法としては，HTG[4]がある．本稿では，プログラムがこのような非循環の複数のマクロタスク集合に分割されていることを前提とする．

2．3　多段先行評価の実行方式

1MTはMTの粒度に応じた複数のPEに割り当てられ，マクロタスク内部で細粒度の並列処理を行うものとする．そして，各MTは，実行開始条件・制御決定条件・実行停止条件の3条件（以下MT制御条件と呼ぶ）により実行制御されるものとする．以下に各条件成立時の処理内容を示す．

(1) 非実行状態のMTは，実行開始条件あるいは制御決定条件が成立した段階で実行を開始する．実行開始条件によって実行を開始した場合を仮実行，制御決定条件によって実行を開始した場合を本実行と呼ぶ．

(2) 仮実行中のMTについて，実行停止条件が成立した段階で，MTの実行を停止しする．

3．先行評価に適したマクロタスク生成

3．1　マクロタスク生成の目的

先行評価に適したMT生成の目的は，(1) 変数の2重定義に伴う副作用問題の回避，及び (2) 仮実行に必要なプロセッサ数の削減の2点である．

(1) に挙げた変数の2重定義による副作用は，制御の確定しない段階でMTの実行を開始（仮実行）することによって生じる．例えば図1(a)の例では，MT2及びMT3において同一変数aが定義されるため，これらMT2及びMT3を仮実行すると，MT4において参照される変数aの値がどちらのMTで定義されたものであるかを保証することができない．これは，制御依存によって，データの依存関係が変るにもかかわらず，制御依存関係を無視して実行を開始している為に起こる．仮実行によるこのような副作用を防ぐためには，図1(b)に示すように，制御に関わらずMT間のデータ依存関係が一意になるようにMTを複製すればよい．

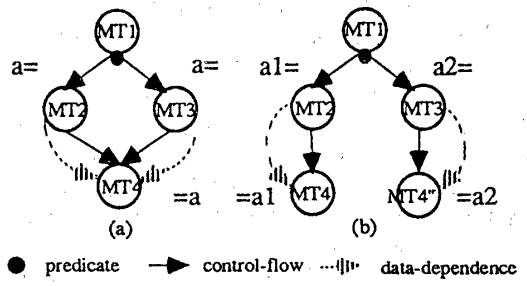

図1 MT制御条件単一化のためのMT複製

(2) に挙げた仮実行に必要なプロセッサ数は，文献[5]で示されているように，先行評価段数をnとするとプログラムのデータ依存関係によって決定し，必ずしもO(2ⁿ)にはならない．すなわち，制御依存を無視して実行しても，データ依存の存在により先行評価の効果が表れない部分が存在する．このような部分を1つのMTとすることにより，実行時のプロセッサ数を削減する．具体的には，データ依存と制御依存の関係を用い，マクロタスクを融合した場合も，先行評価の効果を失わない部分を1つのMTとする．また，この融合によりMT数が減少し，MT制御に伴うオーバヘッドも削減できる．MT制御に伴うオーバヘッドは，MTの起動・実行停止に伴う制御オーバヘッドであり，集中制御を行う場合，MT数に比例して制御オーバヘッドも増大する．

3．2　マクロタスク生成手順

MT生成手順は，以下の6手順である．

(1) HTG[4]の作成(基本タスク生成)
(2) 制御フローグラフの作成
(3) 制御依存グラフの作成
(4) データ依存グラフの作成
(5) 基本タスク制御条件単一化のための基本タスク複製
(6) 仮実行に必要なプロセッサ数削減のための基本タスク融合

手順 (1) ～ (4) は，文献[4]に示される手順に等しい．手順 (5) ～ (6) は，先行評価に適したマクロタスク生成のために，新たに提案する手法である．

A Macro-tasking Scheme for Eager Evaluation
Hayato YAMANA , Toshiaki YASUE, Yoshihiko ISHII and Yoichi MURAOKA
Waseda Univ.

(1)HTG[4]の作成(基本タスク生成)

プログラムを非循環グラフに変換するために，HTG (Hierarchical Task Graph)をプログラムから作成する[4]．HTGにおけるノードを基本タスク(BT)と呼ぶ．

(2)制御フローグラフ(CFG)の作成

制御フローグラフは，HTGから，同一レベルに存在するノードを抜き出したグラフであり，CFG=(V,CFE)で表現する．各頂点v∈VはHTGにおいて同一レベルに存在する基本タスク(BT)を，有向辺cfe=(u,v)∈CFEはBTuからBTvへの制御フローを表す．

(3)制御依存グラフ(CDG)の作成

制御依存グラフは，有向グラフCDG=(V,CDE)で表現される．CDGの各頂点v∈Vは基本タスク，有向辺cde=(u,v)∈CDEはBTuからBTvへの制御依存を表す．また，各有向辺cde=(u,v)∈CDEに対して，ラベルu-aを付ける．ここで，ラベルu-aは，CFG内のBTuにおいてBTaが選択されることを示す．

(4)データ依存グラフ(DDG)の作成

データ依存グラフは，基本タスク間のデータ依存関係を示したものであり，有向グラフDDG=(V,DDE)で表現する．有向辺dde=(u,v)∈DDEはBTuからBTvへのデータ依存を表す．

(5)基本タスク制御条件単一化のための基本タスク複製

BT間のデータ依存を制御依存に関わらず一意に定めるためにノードvの複製が必要かどうかを決定する条件は，DDG上のノードv∈Vに対して入力しているデータ依存辺dde(u,v)のソースノードの集合をDDS(v)={u | dde(u,v)∈DDE}，ノードvが実行されると仮定した時に必ず実行されるノード集合をDN(v)とした時，DDS(v) ⊆DN(v)が成立するかどうかであり，DDS(v)⊆DN(v)の時は，複製の必要がなく，DDS(v)⊈DN(v)の場合は，複製しなければならない．ここで，DN(v)は，次のようにして求めることができる．

CDG上，ノードvに至るある1本のパスをp(v)=<$a_0,a_1,...,a_n$=v>,a_i∈V(ただしa_0は入力に制御依存辺を持たない)で表すとすると，パスp(v)が選択される時，パスP(v)内の全ての制御辺cde(a_i,a_{i+1})は，真となるラベルa_i-c_iを持つ．ここで，パスp(v)に属するラベル集合をL(p(v))={a_i-c_i | cde(a_i,a_{i+1})がラベルa_i-c_iを持つ,<a_i,a_{i+1}>⊆p(v)}とする．一般に，CDG上のノードv∈Vは，ノードvに至るパスを複数持つ．ノードvに至るパス集合をP(v)={p(v) | p(v)=<$a_0,a_1,...,a_n$=v>,a_i∈V}とする．ノードvが選択される時，∃p(v)∈P(v)が選択され，L(p(v))に含まれる全てのラベルが真となる．今，ノードvが選択される時に必ず真となるラベル集合をTL(v)とすると，TL(v)={a_i-c_i | ∀p(v)∈P(v), a_i-c_i∈L(p(v))}となる．

[補題1]　ノードvが選択される時，必ず実行されるノード集合DN(v)は，DN(v)={n | ∃p(n)∈P(n),L(p(n))⊆TL(v)}となる．

(証明)　∃n∈DN(v)が実行されないと仮定する．ノードnが実行されないための必要十分条件は，∀p(n)∈P(n)について，ラベル集合L(p(n))内のラベルの内少なくとも1つが疑になることである．ところが，∃p(n)∈P(n),L(p(n))⊆TL(v)であり，TL(v)は，ノードvが選択される時に必ず真となるラベル集合であるので，同一のラベルが真と疑の2つの状態を持つことになる．ラベルは真あるいは疑の一状態しかもたないので，仮定は矛盾しており，補題1が成立する．

以上から，ノードvが，ノード集合DN(v)以外からデータ依存を持つ場合，ノードvを複製し，全てのノードvに対して，DDS(v) ⊆ DN(v)が成立するようにグラフ変換を行う．詳細なアルゴリズムは文献[6]を参照していただきたい．

(6)仮実行に必要なプロセッサ数削減のための基本タスク融合

実行に際して必要となるプロセッサ数を削減するために(5)で作成されたBT間で融合を行う．

ある2つの基本タスクBTaとBTbが融合できるための条件は，融合後のBTが単一のMT制御条件を持つことである．すなわち，融合後にできたBTは，一組の，実行開始条件・制御決定条件・実行停止条件を持たなければならない．

まず，実行開始条件が融合前後で変らないためには，BTa及びBTbへ入力するデータ依存辺が次の何れかの条件を満たせばよい．

{z | dde(z,BTa)} = { z | dde(z,BTb)} …… (1)

BTa及びBTbへ入力するデータが全く等しい場合であり，実行開始条件は融合の前後で変化しない．

∃dde(BTa,BTb), ∀z, z∈{z | dde(z,BTb)}→dde(z,*)dde(*,BTa)

但dde(z,*)dde(*,BTa)は，zからBTaへのデータ依存がBT* (*は0個以上のBT) を経由して存在することを示す …… (2)

BTaの実行終了によりBTbへ入力するデータ依存を保証できる場合であり，融合後のBT内でBTbへのデータ依存を保証でき，実行開始条件はBTaの実行開始条件で代用できる．

次に，制御決定条件が融合前後で変らないためには，BTa及びBTbへ入力する制御依存辺が次の何れかの条件を満たせばよい．

{z | cde(z,BTa)} = { z | cde(z,BTb)} …… (3)

BTa及びBTbへ入力する制御依存が全く等しい場合であり，制御決定条件は融合の前後で変化しない．

∀z, z={z|cde(z,BTb)}→z=BTa …… (4)

BTa内の分岐評価によりBTbの制御が保証できる場合である．BTa内の分岐評価終了時（BTa終了時)に，BTbの実行が未開始の場合に限り，融合後のBT内でBTbの制御を保証でき，BTb制御決定条件は，BTaの制御決定条件で代用できる．

実行停止条件は，制御決定条件の否定であるため，上記(3)(4)と同様となる．以上より，融合できる条件は，(1)かつ(3), (2)かつ(3), (2)かつ(4)の3条件となる．この結果得られたブロックをマクロタスク（MT）と呼ぶ．

4．むすび

本稿では，先行評価に適したマクロタスクの生成手法について述べた．今後は，これまで提案しているマクロタスク間の先行評価方式[1]と共に，実機上での性能評価を行っていきたいと考えている．

文献

[1] 山名他:"先行評価を用いたマクロタスクの多段仮実行方式の提案", 並列処理シンポジウムJSPP'92予稿集,pp.117-122, 1993

[2] 本多他:"OSCAR上でのFortran並列処理系のインプリメントと性能評価",信学技報, CPSY-89-57, pp.75-80, 1989

[3] AV.Aho R.Sehi and JD.Ullman:"Compilers:Principles, Techiniques and Tools", Addision-Wesley, 1986

[4] Milinf Girkar and C.D.Polychronopolulos: "Automatic Extraction of Functional Parallelism from Ordinary Programs", IEEE Trans. on Parallel and Distributed Systems, Vol.3, No.2, pp.166-178,1992

[5] U.Banerjee et.al:"Fast Execution of Loop with IF statements", IEEE Trans. on Comp.,pp.1030-1033, 1984

[6] 山名他:"並列処理システムにおけるマクロタスク間先行評価方式", 電子情報通信学会論文誌分冊D投稿中