並列処理アルゴリズムを用いた多段合成による論理関数の主項の生成

5B－5

伊藤　貴・　牧野　克友希　鈴木　浩文
松島　秀人　凌　暁萍　後藤　公雄

神奈川工科大学情報工学科

１．緒言

論理関数の主項を高速に生成する手法として多段分割法が既に開発され，少変数においての有効性が示されたが，多変数に対応できない問題が残されている［1］．本研究では多段分割法並列処理手法を取り入れることにより，多変数にも効率良く処理する並列アルゴリズムを提案し，シミュレーションによりその有効性を示す．

２．多段分割法

論理関数が多変数になって行くに従って主項を生成するのに非常に多くの時間と記憶容量を必要とする．そこで最初に与えられた論理関数を任意の数で分割することにより，主項生成時の変数を減らし，演算時間と記憶容量の短縮を図る．分割された関数（以後分割関数と呼ぶ）は，上位分割段内でグループ分けされ，各グループ内では分割関数自身と相互に隣接する分割関数同士から求めた合成関数に対して仮主項（分割関数・合成関数で得られる主項はその関数の中では主項かもしれないが，元の論理関数の主項とは必ずしも言えないので，仮主項とする）を生成する．この時点で分割関数の仮主項と合成関数の仮主項とで吸収・削除の判定を行う．生成された各グループ内の仮主項の集合は上位段の分割関数の一つとして最小項番号の書き戻しを行う．この手続きを各分割段で行い，最上段に達した時に終了する．なお，合成関数の仮主項を生成する場合に，分割関数の仮主項同士を組み合わせて合成関数の仮主項とする方法（規則１）と，分割関数を一旦最小項の集合に戻してからその集合同士で直接合成関数を生成した後仮主項を求める方法（規則２）の２方法がある．

３．多段分割法の並列処理

各段で分割するのではなく，Ｌ段で合成するならば，与えられた論理関数は，2^L個に分割される．そして各分割関数での仮主項の生成処理を並列に行う（これをＢプロセスと呼ぶ）．次に，隣接する分割関数２つのみで行う合成処理を並列に行う（これをＧプロセスという）のだが，この合成処理の中で次に述べる２つの反復処理（ルート）があるので，その部分も並列に処理する．

a．各分割関数でもとまった仮主項を一旦セル集合形に書き戻し，共通要素を取り出して合成関数の共有主項を生成する．共有主項を書き戻し，合成関数の仮主項を生成し，他の仮主項に吸収されないかどうかの判定を行う．

b．各分割関数からもとまった仮主項を合成関数の仮主項として書き戻し，１で得られた仮主項に吸収されるかどうかの判定を行う．吸収判定後に残った仮主項が，次の段で合成するときの分割関数の仮主項となる．

以上の処理を最終段まで行ない，最終段で吸収判定後に残った仮主項が最初に与えられた論理関数の主項になる．

ＢプロセスとＧプロセス以外の逐次的に処理される部分をＳプロセスと呼ぶ．

４．並列化のための理論的解析

前述の並列処理手法を用いた多段分割法並列処理アルゴリズムの理論的な解析による評価を行う．

各パラメータを次のように設定する．

t_b：Ｂプロセスを処理する時間の重み
T_B：Ｂプロセス集合を処理する時間
t_g：Ｇプロセスを処理する時間の重み
T_G：Ｇプロセス集合を処理する時間
t_s：Ｓプロセスを処理する時間の重み
T_S：Ｓプロセス集合を処理する時間
t_c：プロセス間データ転送コストの重み
T_c：データ転送コスト
T　：全処理時間
M　：プロセッサの数

全処理時間Ｔは

$$T = T_B + T_G + T_c + T_S \quad (1)$$

となるのは明らかである．T_Bについて検討してみると，Ｂプロセスは全て並列処理できるので，処理効率は単純にプロセッサ数に左右されると考える．十分なプロセッサが用意されていれば（つまり，$M \geqq 2^L$），各プロセスは同時に始まり，同時に終わると考える．故に，Ｂプロセス集合の処理時間はＢプロセスの処理時間に等しい（$T_B = t_b$）．プロセッサが足りない時にはここでは最も簡単な折り畳み法を用いた．

この方法では，2^L個のＢプロセス集合は結果的にはY_B個のサブ集合に分けて実行される．

$$Y_B = \lceil 2^L / M \rceil \quad (2)$$

ここで，$\lceil a \rceil$はａより大きい最も小さな整数を意味する．よってT_Bは，

$$T_B = \lceil 2^L / M \rceil \times t_b \quad (3)$$

次にT_Gだが，ｋ段におけるＧプロセスの全処理時間T_g'はT_Bと同様に求められる．

$$T_g = \lceil (2^L / 2^k) / M \rceil \cdot t_g \quad (4)$$

Prime Impricants Generation of Logic Function by Multi-Level-Synthsis Method using Parallel Processing
Department of Information and Computer Sciences, Kanagawa Institute of Technology.

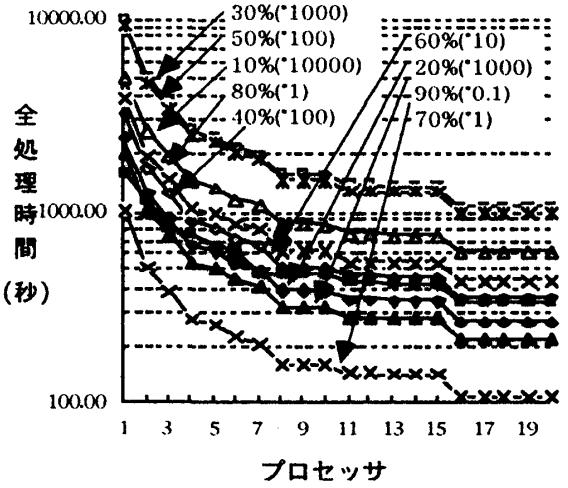

図１　濃度による並列処理時間の変化
（１４変数，６４分割）

よってGプロセス集合の処理時間を求める式は次のようになる．

$$T_g = \sum_{k=1}^{L} \left| \lceil (2^L / 2^k) / M \rceil \cdot t_g \right|$$

$$= \sum_{k=1}^{L} \left| \lceil 2^{L-k} / M \rceil \cdot t_g \right| \qquad (5)$$

処理全体に渡るプロセス間交信にかかるコストT_cを求める式は次のように仮定する．

$$T_c = \sum_{k=1}^{L} \left| \lceil 2^{L-k} / M \rceil \cdot 2 \right| \cdot t_c \qquad (6)$$

また，Sプロセス集合の全処理時間T_Sは次のようになる．

$$T_S = \sum_{s \in S}^{L} t_s \qquad (7)$$

5．シミュレーションによる評価

多段分割法の逐次処理アルゴリズムは既にSUN4上に（C言語により）実装され，様々な濃度または変数の数で測定され，評価が行われていた．ここで，前節で述べた論理的解析結果を基に並列処理を行う場合の処理時間をシミュレートすることにより，多段分割法の並列処理効果を示す．

シミュレーションはC言語で書かれ，SPARCステーション5（サンマイクロシステムズ社製）上に実装した．前記各プロセスの処理時間単位は逐次処理アルゴリズム時の測定値を参考して設定した．その結果を図１と図２に示す．

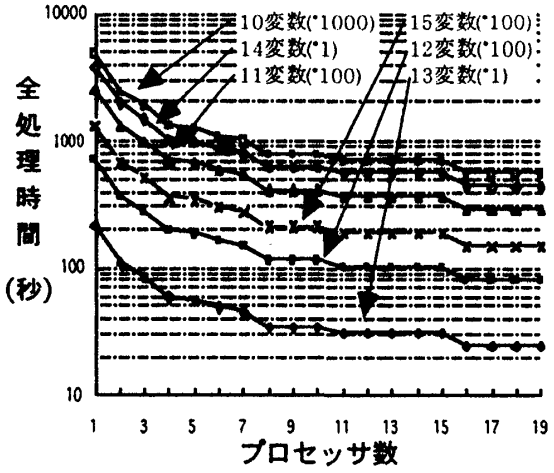

図２　変数による並列処理時間の変化
（濃度８０％，６４分割）

図１では１４変数の論理関数に対する処理で，変数の濃度を１０％から９０％まで変えて得られたプロセッサ台数効果を示した．図２は濃度が８０％の時の変数の数を１０変数から１５変数まで変えて得られた台数効果を示す．これらの図より次の結果が得られた。

- 並列処理の台数効果はいずれの場合も１６台までであり，８台までは顕著である．
- 粗粒度での並列処理を行う場合，与えられた論理関数の変数の数や最小項の濃度などに関係なく，使用するプロセッサの数に多く左右される．

6．むすび

結果として、変数が多いほど、あるいは濃度が高いほど、その効果がある（処理時間が１０分の１以下に短縮される）ことが確認された。またプロセッサ数が十分にあれば分割数が多いほど効果が発揮される。

参考文献

[1] Goto, K., Tatumi, H., Ling, X.P. and Takahashi, S. : "Improvement of Prime Impricants Generation of Logic Function by Multi-Level-Synthsis Method", Proc. of the IEEE TENCON'94, Vol.2, p.938, Aug, 1994.