2004－AL－94　(13)
2004／3／19

ブロック化分岐プログラムにおける変数順序と表現能力の関係

小関一弘　武永康彦†

ブロック化分岐プログラムとは分岐プログラムの変数を読む順序に制約を設けたモデルの一つである。本稿ではブロック化分岐プログラムに於ける変数順序と表現能力の関係について調べる。k 回読み l 変数順序ブロック化分岐プログラムに於いて、ブロックの変数順序を入れ替えた場合のサイズの変化について、隣接するブロックでの入れ替えにより多項式サイズで表現出来る関数のクラスが変化しないとの仮定がもたらす結果を明らかにし、また、定数幅で表される関数のクラスは変数順序の入れ替えにより変化しない事を示す。

The Relation between Variable Order and Computational Power of Blockwise Branching Programs

Kazuhiro KOSEKI and Yasuhiko TAKENAGA †

A blockwise branching program is a kind of branching program with restrictions on the order to read variables. In this paper, we discuss the relation between the variable order and the computational power of the blockwise branching programs. We show some results from assumptions that the exchange of neighboring blocks doesn't affect the class of functions represented by polynomial size read-k-times l-variable-order blockwise branching programs. Also we prove that the class of functions represented by constant width blockwise branching programs doesn't change by exchanging variable orders of its blocks.

1. はじめに

論理関数の複雑さを明らかにする事は理論計算機科学に於ける重要な課題の一つである。論理関数の表現方法としては、組み合わせ回路、論理式、分岐プログラム等が代表的なものとして挙げられる。種々のアプリケーションに於いて、論理関数の内部表現としてしばしば制限を加えた分岐プログラムであるOBDD (Ordered Binary Decision Diagram) が利用されている[1]。OBDD の特徴として変数順序を固定すると表現が一意に定まり、演算が高速であるという点が挙げられる。また、多くの論理関数を小さなサイズで表現出来、記憶領域が少なくて済むという利点もある。

本研究の背景として、OBDD は扱う論理関数の規模が大きくなるにつれて、実用的に効率良く扱えるサイズで表現する事が難しくなって来ているという状況がある。最近ではOBDD よりコンパクトな論理関数表現が求められる様になり、OBDD の制約を緩和した分岐プログラムの研究[2)~6)] が行われている。分岐プログラムへの制約の付け方には、入力を読む回数や変数順序に制約を加えるというものがある。これ迄に1回読みの分岐プログラムに関する研究は多く行われているが、2回以上入力を読む分岐プログラムに関しては、まだ分かっていない事が多い。

関連する研究としては、深さに制限を加えた分岐プログラムを考えて、その下界を調べたもの[5)] や、n ブロック全てで同一の変数順序を取る分岐プログラムの表現能力を調べたり、2 ブロックのブロック化分岐プログラムによって言語 k-PLANE が任意の k について表現可能である事を示したりしたもの[2)]、変数順序を l 通り迄許した分岐プログラムを考えて、言語3-S.PLANE を多項式サイズの3回読み2変数順序ブロック化分岐プログラムによって表す事が出来るのならば、言語 k-S.PLANE が多項式サイズの k 回読み $(k-1)$ 変数順序ブロック化分岐プログラムで表現出来る事を示したもの[3)] 等、制限を加えた分岐プログラムについての研究が行われている。

本研究では変数順序と表現能力の関係について調べる。具体的には変数順序の順列を入れ替える事で、分岐プログラムのサイズがどの程度大きくなるのかについて、二つの分岐プログラムのモデルを用いて考える。一

† 電気通信大学 電気通信学部 情報工学科
Department of Computer Science,The University of Electro-Communications

つは k 回読み l 変数順序ブロック化分岐プログラムである。この k 回読み l 変数順序ブロック化分岐プログラムに於いて、任意の連続する 2 ブロックの変数順序の順列を入れ替えても多項式サイズで収まるとした場合と、$k=2$ の時にこの 2 ブロックの変数順序の順列を入れ替えても多項式サイズで収まるとした場合の二通りの仮定を立てて、それぞれの仮定が成り立つ時に、k 回読み l 変数順序ブロック化分岐プログラムに於いて、変数順序の順列を入れ替えた時のサイズの増加について考える。

もう一つのモデルは、k 回読み l 変数順序定数幅ブロック化分岐プログラムである。このモデルについても、$k=2$ である 2 回読み定数幅ブロック化分岐プログラムでの変数順序の順列を入れ替えた時のサイズの変化が定数サイズで抑えられる事を用いて、k 回読み l 変数順序定数幅ブロック化分岐プログラムのサイズの変化を調べる。

以下 2章では種々の分岐プログラムの定義を行い、それらの間の関係を示す。次に、3章では変数順序の順列を入れ替えた時の分岐プログラムのサイズの変化について述べる。最後に、4章をまとめとする。

2. 分岐プログラム (Branching Program)

2.1 分岐プログラム

分岐プログラムは論理関数を有向非巡回グラフで表現したものである。1 個以上の非終端節点と 2 個の終端節点から成り、それぞれ変数、定数でラベル付けされていて、変数節点、定数節点と言う。また、変数節点のうち入次数が 0 のものを根と言う。変数節点の出次数は 2 で、定数節点の出次数は 0 である。根から定数節点迄、各変数節点にラベル付けされた変数の変数割り当てに従って枝を選択してパスを構成し、到達した定数節点にラベル付けされた値がその関数の値となる。

根となる節点からある節点迄の最長パス上の節点の数をその節点のレベルと言う。同一レベルにある節点の数のうち、最大のものがその分岐プログラムの幅となる。分岐プログラムのサイズは、その分岐プログラムの幅に依存する。また、一つのブロック内での変数の現れる順序の事を変数順序と呼ぶ。変数順序は変数の添字上の置換によって表される。各変数は一つのブロック内の任意のパス上で高々 1 回、そのブロックの変数順序に逆らわない順番で現れる。

2.2 種々の分岐プログラム

分岐プログラムには変数を読む順序や変数の出現回数、変数順序等に制限を加えた様々なモデルが提案されている。

k 回読み分岐プログラム[1] 根となる節点から定数節点迄の任意のパス上で、どの変数も高々 k 回現れる時、この分岐プログラムを k 回読み分岐プログラムと呼ぶ。

Oblivious 分岐プログラム[5] 任意の節点から出ている 2 本の枝のどちらもが次のレベルの節点を指していて、且つ同じレベルの全ての節点が同一変数でラベル付けされている分岐プログラムを oblivious 分岐プログラムと呼ぶ。

ブロック化分岐プログラム[2] Oblivious 分岐プログラムに於いて変数の数を n とすると、$((i-1)n+1)$ レベル $(i \geq 1)$ から in レベル迄の連続する n レベルを一まとまりとしたものをブロックと言い、根となる節点が含まれるブロックから数えて i 番目のブロックを第 i ブロックの様に言う。この様なブロックに分けられ、且つ全ての変数が各ブロック内で丁度一つのレベルにのみ現れる時、この分岐プログラムをブロック化分岐プログラムと呼ぶ。

k 回読み l 変数順序ブロック化分岐プログラム[3] k 回読みブロック化分岐プログラムはブロック毎に異なる変数順序を取る事が許されている。即ち、k 通りの変数順序を取る事が出来る。分岐プログラムが l 通り $(1 \leq l \leq k)$ の変数順序によって構成されている時、l 変数順序ブロック化分岐プログラムと呼ぶ。

$(k,^*)$- プログラム[5] k 回読みブロック化分岐プログラムに於いて、全てのブロックで同じ変数順序が割り当てられている時、即ち、k 回読み 1 変数順序ブロック化分岐プログラムの時、この分岐プログラムを $(k,^*)$- プログラムと呼ぶ。

OBDD(Ordered Binary Decision Diagram)[1] k 回読みブロック化分岐プログラムに於いて、$k=1$ の時、即ち 1 回読みブロック化分岐プログラムの時、この分岐プログラムを OBDD と呼ぶ。

ブロック化分岐プログラムに於いて変数順序の表記については、$\Pi_i (1 \leq i \leq k)$ を第 i ブロックの変数順序とし、$\Pi = (\Pi_1, \Pi_2, \cdots, \Pi_k)$ と表す事とする。また、第 i ブロックの変数順序 Π_i は、l 変数順序の時、その中の一つの変数順序 $\pi_j (1 \leq j \leq l)$ が割り当てられており、それを $\Pi_i = \pi_j$ と表す事とする。

今回主に扱うのはブロック化分岐プログラムである。ブロック化分岐プログラムの入力を読む回数と変数順序

表 1　多項式サイズで表現出来る言語のクラスの表記

表記	モデル
P_{BPk}	k 回読み分岐プログラム
P_{oBP}	oblivious 分岐プログラム
P_{bBP}	ブロック化分岐プログラム
P_{bBPk}	k 回読みブロック化分岐プログラム
P_{lBP}	l 変数順序ブロック化分岐プログラム
P_{lBPk}	k 回読み l 変数順序ブロック化分岐プログラム
P_{BPk*}	$(k,^*)$- プログラム
P_{OBDD}	OBDD
$P_{bBPk\Pi}$	k 回読みブロック化分岐プログラム (順列:Π)
$P_{lBPk\Pi}$	k 回読み l 変数順序ブロック化分岐プログラム (順列:Π)

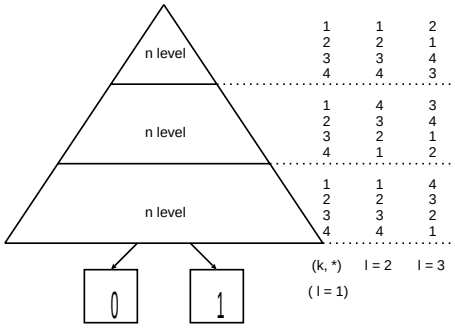

図 2　入力を読む回数、変数順序の数の違いの例

の数についてまとめると**図 1** の様になる。図にも書いてある通り、図中にて $[a,b]$ とあるのは a 回読み b 変数順序ブロック化分岐プログラムを表している。$a=k, b=1$ の時、そのブロック化分岐プログラムは $(k,^*)$- プログラムとなる。$a=k, b=l(1\leq l\leq k)$ の時、そのブロック化分岐プログラムは k 回読み l 変数順序ブロック化分岐プログラムとなる。$a=b=k$ の時、そのブロック化分岐プログラムは k 回読みブロック化分岐プログラムとなる。そして、$a=b=1$ の時、そのブロック化分岐プログラムは OBDD となる。

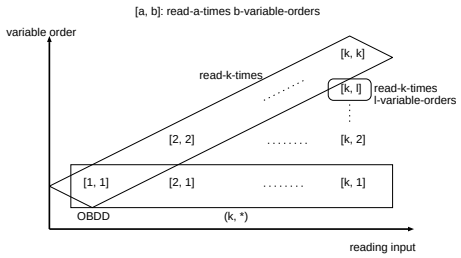

図 1　入力を読む回数、変数順序の数の違いによるモデルの違い

それぞれの分岐プログラムのモデルによって、多項式サイズで表現出来る言語のクラスを**表 1** の様に表す事とする。

また、1 ブロックに 4 変数ずつある 3 回読み、即ち 3 ブロックのブロック化分岐プログラムを例にとり、$(k,^*)$- プログラム、k 回読み l 変数順序ブロック化分岐プログラム、そして k 回読みブロック化分岐プログラムの三つのモデルの違いを示したものが**図 2** である。図中に於いて、1 から 4 の数字で 4 変数を表している。

3. 変数順序の順列と表現能力の関係

3.1 2 変数順序に於ける順列の入れ替え

分岐プログラムの表現能力を調べるに当たり、二つの分岐プログラムのモデル間に於ける表現能力の差を明らかにする事が重要な目標の一つである。表現能力の差を示す為には、各分岐プログラムのモデルに於ける良い下界を求める必要がある。1章でも触れたが、1 回読みの様々なモデルに於いては既にいくつもの指数的下界が示されている。しかし、2 回読み以上の良い下界を求める事は難しいと考えられる。

本章では、各ブロックの変数順序の並び方を変える事で、分岐プログラムのサイズがどの程度大きくなるのかについて考える。二通りの変数順序の並び方があった時多項式のクラスで表現能力に差が生じる可能性を持った言語は見つかるものの、超多項式的下界の証明が困難な事から、クラスの違いは示せてはいない。ここではその様な可能性のある言語の例を一つ挙げる。

ADD-and-COUNT　長さ $2n(n\geq 1)$ の二進数列 $a_1a_2\cdots a_{2n}$ があった時、$a_1a_2\cdots a_n+a_{n+1}a_{n+2}\cdots a_{2n}$ の値を $a'_0,a'_1,\cdots,a'_n$ とする。$\alpha=\sum_{i=0}^{n}a'_i$ とした時 $a'_{2\alpha}=1$、且つ $\sum_{j=1}^{2\alpha}a_j=\alpha$ が成り立つ時、且つその時に限り 1 となる。

この言語は、入力を第 1 ブロックで $a_n,\ a_{2n},\ a_{n-1},\ a_{2n-1},\ \cdots a_1,\ a_{n+1}$ と読み、第 2 ブロックでは $a_1,a_2,\cdots a_{2n}$ の順に読めば、2 回読みブロック化分岐プログラムによって多項式サイズで表現する事が出来る。しかし、第 1 ブロックと第 2 ブロックそれぞれでの入力を読む順序を入れ替え、第 1 ブロックで $a_1,a_2,\cdots a_{2n}$ と読み、第 2 ブロックでは $a_n,a_{2n},a_{n-1},a_{2n-1},\cdots a_1,a_{n+1}$ の順に読んだ場合は、指数サイズが必要ではないかと予想される。

この様に、ある言語が分岐プログラムの一つのモデルによってどの程度のサイズで表現出来るのかを示す事は大変に難しい問題となっている。従って以下では k 回読みのモデルに於いて、第1 ブロックから第 k ブロック迄の変数順序の並び方を変換した時に、そのサイズがどの程度大きくなるのかについて考える事とする。先ず、変数順序の順列を入れ替える事によるサイズの増加が多項

式サイズで抑えられる様な仮定を設け、その上で任意の変数順序の順列を別の順列に変換した時のサイズの変化について考える。次にサイズの増加は幅に依存する事から、幅が固定された場合の分岐プログラムについて考える。

3.2 k 回読み l 変数順序ブロック化分岐プログラム

k 回読み l 変数順序ブロック化分岐プログラムに於いて、変数順序の異なる順列間での分岐プログラムのサイズを調べる為に、各ブロックの変数順序を入れ替える事を考える。ここでは二つの仮定を用意し、それぞれの仮定を踏まえてそれぞれで定理を導く。

先ず一つ目は k 回読み l 変数順序ブロック化分岐プログラムに於いて、以下を仮定する。

仮定 1 $\Pi = (\Pi_1, \Pi_2, \cdots, \Pi_{i-1}, \Pi_i, \Pi_{i+1}, \cdots, \Pi_k)$, $\Pi' = (\Pi_1, \Pi_2, \cdots, \Pi_{i-1}, \Pi_{i+1}, \Pi_i, \cdots, \Pi_k)$ とすると、$P_{lBPk\Pi} = P_{lBPk\Pi'}$ が成り立つ。

この時、以下の定理が成り立つ。

定理 1 仮定 1が成り立つ時、以下が成り立つ。任意の定数 k に対し、k 回読み 2 変数順序ブロック化分岐プログラムに於いて、変数順序 π_1 が c 個、変数順序 π_2 が $(k-c)$ 個の時に構成される任意の変数順序の順列 Π, Π' に対して、$P_{lBPk\Pi} = P_{lBPk\Pi'}$ となる。

証明 仮定 1から、任意の連続する 2 ブロックの変数順序の順列を入れ替える事により、サイズは多項式的な増加で済む。二つの変数順序から成る順列を任意の順列へと変換する回数はブロックの数に依存する。ブロックの数 k は定数である事から、交換も定数回で済む。よって任意の順列への変換を行う事が可能である。

□

次に 2 回読みブロック化分岐プログラムに於いて、以下の様に仮定する。

仮定 2 $P_{bBP2(\Pi_1,\Pi_2)} = P_{bBP2(\Pi_2,\Pi_1)}$ が成り立つ。

この時、次の定理が成り立つ。

定理 2 仮定 2が成り立つ時、以下が成り立つ。任意の偶数の定数 $k(k \geq 6)$ に対し、k 回読み 2 変数順序ブロック化分岐プログラムに於いて、変数順序 π_1 が c 個、変数順序 π_2 が $(k-c)$ 個の時に構成される変数順序の任意の順列 Π, Π' に対して、$P_{2BPk\Pi} = P_{2BPk\Pi'}$。

証明 仮定 2では、1 ブロックが n 変数から成る 2 ブロックのブロック化分岐プログラムに於ける変数順序の入れ替えが許されている。

そこで、第 $(2m+1)$ ブロック $(0 \leq m \leq \frac{k-1}{2})$ から第 k ブロック迄の $(k-2m)$ ブロックを $(\frac{k}{2}-m)$ ブロックずつ二つに分け、各々を一つの大きなブロックと見なして、1 ブロック $(\frac{k}{2}-m)n$ 変数から成る 2 ブロックのブロック化分岐プログラムと見なす。一つの大きなブロック内に於いては、同じ変数でラベル付けされていても、異なるレベルにある場合は異なる変数同士として扱う。例えば、第 j ブロックのある一つのレベル全てにラベル付けされている変数 x_i を x_{ij} とする。すると、このブロック内に於いても任意のパス中に各変数は 1 回しか現れない事となり、ブロック化分岐プログラムとなっている。$m \geq 1$ の時はこの 2 ブロックのブロック化分岐プログラムの根となる節点は、元の k ブロックのブロック化分岐プログラムの第 $(2m+1)n$ レベルの節点であり、一般に複数存在する。

ここで、$m = 0$ の時はこの 2 ブロックのブロック化分岐プログラムの根となる節点も、元の k ブロックのブロック化分岐プログラムの根となる節点も、同じであり唯一である。この 1 ブロック $\frac{kn}{2}$ 変数から成る 2 ブロックのブロック化分岐プログラムに於いて、仮定 2より多項式サイズで変数順序の順列を入れ替える。

$m \geq 1$ の時は 2 ブロックのブロック化分岐プログラムに於いて、根となる節点が複数存在する事となるが、高々多項式個なので仮定 2を用いて変数順序の順列を入れ替えた後も、多項式サイズで収まる。

そして、入れ替えた後はまた、1 ブロック n 変数から成る $(k-2m)$ ブロックのブロック化分岐プログラムと見れば良い。つまり、変数も x_{ij} としていたものを第 j ブロックの x_i の様に元に戻す。こうする事によって、第 $(2m+1)$ ブロック $(0 \leq m \leq \frac{k-1}{2})$ から第 k ブロック迄の $(k-2m)$ ブロックの変数順序の順列の変換が可能である事が分かる。

ここで、第 $(k-2m+1)$ ブロックから第 k ブロック迄の $2m$ ブロックの変数順序の順列を上記の方法で入れ替える関数を δ_m と表す事とする。例えば、$k = 4, m = 4, \Pi = (a, b, c, d)$とすると、

$$\delta_4(a, b, c, d) = (c, d, a, b)$$

また、$m = 2$ では、

$$\delta_2(a, b, c, d) = (a, b, d, c)$$

となる。

次に δ_m を用いて変数順序の任意の順列間に於いて、変換が行える事を数学的帰納法を用いて証明する。

(i) $k = 6$ の時、先に示した入れ替えの方法を用いて変数順序の任意の順列から別の任意の順列へと変換が可能である事を**図 3**、**図 4**、**図 5** に示す。三つの図に於いて、囲まれている六つのカンマで区切られた数字の並びは、左から昇順に第 i ブロック $(1 \leq i \leq 6)$ に割り当てられている変数順序の順列を表している。2 変数順序であるので、二種類の一方を 1、他方を 2 と表す。矢印についている添字は、前述の入れ替えを行う関数 δ の添字を表している。

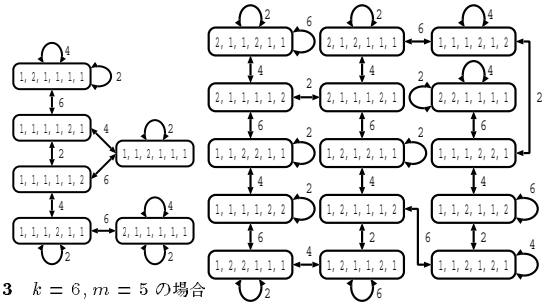

図 3　$k=6, m=5$ の場合

図 4　$k=6, m=4$ の場合

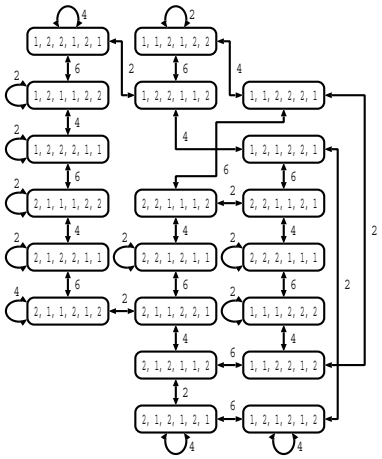

図 5　$k=6, m=3$ の場合

これらの図より、$k=6$ の時、変数順序の任意の順列から別の任意の順列へと変換出来る事が分かる。ちなみに$k=6, m=2$、$k=6, m=1$のそれぞれの場合は、それぞれ図 4、図 3 に於いて図中の変数順序を表している数字 1 と 2 を入れ替える事で示す事が出来る。

(ii)　一般の k の時、変数順序の任意の順列から、別の任意の順列への変換が可能であると仮定する。

この時、k の次の偶数である $k+2$ を考える。第 3 ブロックから第 $(k+2)$ ブロック迄の k ブロックに注目すると、k の時の仮定より、この k ブロック内で変数順序の順列は任意に変換する事が出来る。よって、変換前の変数順序の順列を Π、変換後の順列を Ψ とすると、$\Pi_{\frac{k}{2}+2}=\Psi_1, \Pi_{\frac{k}{2}+3}=\Psi_2$となっている様な変数順序のある順列にも変換可能である。

次に 1 ブロック n 変数の $(k+2)$ ブロックを 1 ブロック $(\frac{k}{2}+1)n$ 変数の大きな 2 ブロックと見なし、先の仮定 2を用いて変数順序の順列を入れ替えると、$\Pi_1=\Psi_1, \Pi_2=\Psi_2$となる。よって、$\Pi_1, \Pi_2$ は求める順列 Ψ を満たしているので、以後は第 3 ブロック以降の k ブロックについて考えれば良い。k の時の仮定から、k ブロックは変数順序の任意の順列に変換可能である。従って、第 3 ブロック以降も求める順列 Ψ を満たす変数順序の順列に変換出来る。

以上より、$(k+2)$ ブロックでも変数順序の任意の順列から別の任意の順列へと変換する事が可能である。

この一連の変数順序の順列を入れ替える回数は k にのみ依存する。ここで k は定数である事から、このブロック化分岐プログラムのサイズは多項式的な増加で収まっている事が分かる。

□

定理 2では k が偶数の場合のみを考えたが、奇数の場合は第 1 ブロックの変数順序を変える事が出来ない。しかし、第 2 ブロックから第 k ブロック迄の $(k-1)$ ブロックは偶数個である事から、その部分に関しては変数順序の順列を任意に変換する事が可能である。

3.3　k 回読み l 変数順序定数幅ブロック化分岐プログラム

ブロック化分岐プログラムの幅が定数の場合を考える。ある二つのブロック間で変数順序の順列を入れ替えた時、ブロック化分岐プログラムのサイズ、即ち幅は前述の通り、一般には n に関してどの様に増加して行くのか分かっていない。前節 3.2では、仮定として変数順序の順列の入れ替えを行った後のブロック化分岐プログラムの幅は、多項式的な増加で抑えられるとして定理を導いた。

よって今度は幅が定数で固定されているブロック化分岐プログラムを考え、その変数順序の順列を変換した時この定数幅ブロック化分岐プログラムのサイズがどの様に増加するかについて考える。

k 回読み l 変数順序定数幅ブロック化分岐プログラムに関して、次の定理が成り立つ。

定理 3　　k 回読み l 変数順序定数幅ブロック化分岐プログラムの変数順序の順列を任意に変換する事によるサイズの増加は定数で抑えられる。

証明　先ず幅が d である 2 回読み定数幅ブロック化分岐プログラムを考える。変数順序は $\Pi=(\Pi_1, \Pi_2)$ とすと、この 2 回読み定数幅ブロック化分岐プログラムに於いて、変数順序を $\Pi'=(\Pi_2, \Pi_1)$ の様にその順番を入れ替える。

この定数幅ブロック化分岐プログラムが表している関数の部分関数を考える。幅が d である事から第 2 ブロックの第 1 レベルの節点数は高々 d である。この高々 d 個の節点を　　$\cdots, n_d$ とする。全てのパス上には、ある一つの $n_i (1 \leq i \leq d)$ が必ず存在するので、第 1 ブロックの根となる節点から n_i 迄を f_i とし、n_i に到達する時 $f_i=1$、そうでない時 0 とする。そして第 2 ブ

ロックの n_i の表す関数を g_i とする。この時 f_i は第1ブロックより小さく、第1ブロックに割り当てられた変数順序に於いて幅は d 以下である。g_i も同様に第2ブロックより小さく、第2ブロックに割り当てられた変数順序に於いて幅は d 以下である。この2回読み定数幅ブロック化分岐プログラムの表す関数 F は、次の様に表す事が出来る。

$$\begin{aligned} F &= \sum_{i=1}^{d} f_i \cdot g_i \\ &= \overline{g_1} \cdot \overline{g_2} \cdot \cdots \cdot \overline{g_d} \cdot 0 \\ &\quad + \overline{g_1} \cdot \overline{g_2} \cdot \cdots \cdot \overline{g_{d-1}} \cdot g_d \cdot f_d \\ &\quad + \overline{g_1} \cdot \overline{g_2} \cdot \cdots \cdot \overline{g_{d-2}} \cdot g_{d-1} \cdot \overline{g_d} \cdot f_{d-1} \\ &\quad + \quad \cdots \\ &\quad + g_1 \cdot \overline{g_2} \cdot \overline{g_3} \cdot \cdots \cdot \overline{g_d} \cdot f_1 \end{aligned}$$

一つの部分関数 $g_i (1 \leq i \leq d)$ の幅は高々 d。これが d 個あるのでそれぞれの和項について、その幅は d^d であり、それが $(d+1)$ 個あるので入れ替えた後の第1ブロックの幅は高々 $(d^d)^{d+1} = d^{d(d+1)}$ となる。また、第2ブロックの幅は高々 d である。d は定数より、入れ替えた事によるサイズの増加は定数で抑えられる事が分かる。

次に k 回読み l 変数順序定数幅ブロック化分岐プログラムにおいて、任意の隣接する2ブロックの変数順序を入れ替える場合を考える。この2ブロックのみに注目すると、根となる節点が高々 d 個あり、このうち一つだけを考えると、葉となる節点を高々 d 個持っている。これに対して変数順序を入れ替えると、各々の葉に対応する関数が高々 $d^{d(d+1)}$ の幅の分岐プログラムで表される事より、全体で幅は高々 $d^{d^2(d+1)}$ となる。これを高々 d 個ある全ての根に対して行うと、全体で幅は高々 $d^{d^2(d+1)+1}$ となる。

ここで、ある変数順序の一つの順列から別の順列へ変換を行う際のサイズの増え方を考える。2ブロック間の変数順序の順列を入れ替えた時、サイズは高々 $d^{d^2(d+1)+1}$ であった。従って、ある変数順序の順列から別の順列へは隣接する2ブロックの変数順序の入れ替えを高々 $\frac{k^2}{2}$ 回行えば良い事から、そのサイズは高々 $(d^{d^2(d+1)+1})^{\frac{k^2}{2}} = d^{(d^2(d+1)+1)\frac{k^2}{2}}$ と分かる。k も定数であるから、高々定数である。よって k 回読み l 変数順序定数幅ブロック化分岐プログラムの変数順序の順列の任意の変換によるサイズの増加は定数で抑えられる。

□

4. ま と め

本稿では二つの分岐プログラムのモデルに関して、変数順序の順列を変換する事によるサイズの変化について考えた。

k 回読み2変数順序ブロック化分岐プログラムに於いては、隣接するブロック間の変数順序を入れ替える際の少し異なる二つの仮定の下で、定理1では任意の定数 k について、定理2では任意の偶数の定数 k について、変数順序の順列を任意に変換しても多項式のサイズで抑えられるという結果が得られた。

また、k 回読み l 変数順序定数幅ブロック化分岐プログラムに於いては、ブロック間で変数順序を入れ替えても表現される関数のクラスは変わらない事が示された。

参 考 文 献

1) R. E. Bryant, "Graph-based algorithm for Boolean function manipulation", IEEE Trans. Comput., vol.35, No.8, pp.677-691, 1986.
2) Yasuhiko TAKENAGA and Shuzo YAJIMA, "On Read-k-times-only Branching Programs", Technical Report of IEICE, COMP94-51, 1994.
3) Kazuya HIROTA, Kazuyoshi TAKAGI and Naofumi TAKAGI, "Computational Power of Read-k-times-only l-variable-ordering Block-wise Branching Programs", Technical Report of IEICE, COMP2001-12, 2001.
4) 市村昌一, 武永康彦, "幅に制限を加えたOBDDの等価性判定", Technical Report of IEICE, COMP2000-11, 2000.
5) Matthias Krause, "Lower Bounds for Depth-Restricted Branching Programs", Information and Computation, pp1-14, 1991.
6) Beate Bollig, Martin Sauerhoff, Detlef Sieling and Ingo Wegener, "ON THE POWER OF DIFFERENT TYPES OF RESTRICTED BRANCHING PROGRAMS", FORSCHUNGSBERICHTE RESEARCH REPORTS, Forschungsbericht Nr. 562, 1994