A-28

# $k$ 回読み blockwise 分岐プログラムについて

小関一弘　　武永康彦 (電気通信大学)

## 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おける最重要課題のひとつである。論理関数の表現方法のひとつに分岐プログラムがある。

最近では強い制約を持つ分岐プログラムであるOBDD(Ordered Binary Decision Diagram) [1] が論理設計支援の分野で広く用いられる等実用面でも注目され、OBDD の制約を緩和した分岐プログラムの研究も行われている。そのひとつに blockwise 分岐プログラムがある。Blockwise 分岐プログラムは、異なる変数順序で変数を複数回読める様に OBDD の制約を緩和したモデルである。この blockwise 分岐プログラムの計算能力を調べる為に言語 $k$-PLANE [2] が考えられている。また、blockwise 分岐プログラムの変数順序に制約を加えた時の計算能力を調べる為に、言語 $k$-$S$.PLANE [3] も考えられている。

言語 $k$-PLANE に対し、これまでに $k$-PLANE を表す 2 回読み blockwise 分岐プログラムの構成方法が示されているが [2]、これは $n \bmod (k-1) = 0$ の時にのみ可能である。

そこで本研究では任意の定数 $k$ に対し $n \bmod (k-1) \neq 0$ の場合でも $k$-PLANE $\in P_{bBP2}$ となる事を示す。

## 2 Blockwise 分岐プログラム

分岐プログラムとは論理関数を有向非巡回グラフで表現したものである。非終端節点は変数で、終端節点は定数 0, 1 でラベル付けされており、それぞれ変数節点、定数節点と言う。変数節点、定数節点の出次数はそれぞれ 2、0 である。変数節点のうち入次数が 0 のものをソースと呼び、ソースは 1 個しかない。出枝は一方が 0、他方が 1 でラベル付けされており、それぞれ 0 枝、1 枝と呼ぶ。変数への値の割り当てが与えられた時、各変数節点にラベル付けされている変数の値によって 0 枝または 1 枝に進む。ソースから定数節点まで辿り、到達した終端節点のラベルが関数の値となる。

ソースから定数節点まで辿る間に各変数が高々 $k$ 回現れる分岐プログラムを $k$ 回読み分岐プログラムと言う。また、ソースからある節点までに通過した節点の数の最大値をレベルと言い、連続する $n$ レベルをひとまとまりとしたものをブロックと言う。分岐プログラムの任意の節点から出ている 2 本の枝の両方が次のレベルの節点を指しており、かつ同じレベルの全ての節点が同じ変数でラベル付けされ、全ての変数がブロック内で丁度 1 回だけ現れる時、この分岐プログラムを blockwise 分岐プログラムと呼ぶ。変数順序はブロック毎に異なっても良い。

分岐プログラムを構成する節点の数をその分岐プログラムのサイズとする。$k$ 回読み blockwise 分岐プログラムによって多項式サイズで表現される言語のクラスを $P_{bBPk}$ と書く。

## 3 言語 $k$-PLANE

$k$-PLANE は以下の様な言語である。

**定義 1** $A(d_1, d_2, \cdots, d_k) \in \{0,1\}^{n^k}$ $(1 \leq d_i \leq n)$ を $k$ 次の論理行列とする。$A|_{d_i=t}$ を $A(d_1, d_2, \cdots, d_k)$ に $d_i = t$ とする事で得た $k-1$ 次の行列とする。任意の $t$ $(1 \leq t \leq n)$ に対し $A|_{d_i=t}$ の全ての要素が等しくなる様な $i$ $(1 \leq i \leq k)$ が存在する時、かつその時に限り $k$-PLANE$(A) = 1$ となる。

図 1は 3-PLANE$(A)$ = 1 となる場合の例を表している。

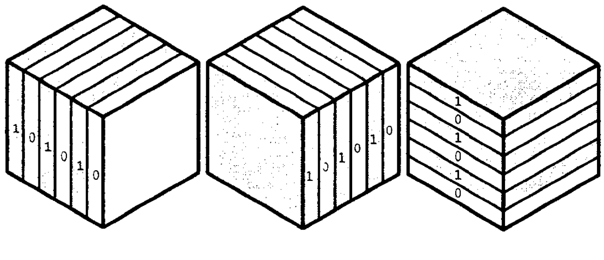


図 1: 3-PLANE$(A) = 1$ の例

言語 $k$-PLANE は $n \bmod (k-1) = 0$ の時、2 回読みの blockwise 分岐プログラムによって多項式サイズで表される。この 2 回読み blockwise 分岐プログラムの構成方法は、入力である $k$-PLANE を $(k-1)$ 個の部分 cube に分割し、1 回目でその対応する要素の比

較を行いながらある1次元の方向において$k$-PLANEが1となるかどうかを調べ、2回目で残る$(k-1)$個の次元を$(k-1)$個の部分cubeでそれぞれ調べる様にする。しかし$n \bmod (k-1) \neq 0$の場合は$(K-1)$個の部分cubeに渡って対応する要素が存在する部分とそうでない部分とがある為、そのまま当てはめる事が出来ない。よって、本論文ではその場合の構成方法を示し、次の結果を得た。

**定理1** 任意の$n, k$に対し、以下が成り立つ。
$k$-PLANE $\in P_{bBP2}$

## 4 3-PLANEを表す2回読みblockwise分岐プログラム

構成方法の概要としては、1回目で先ず最大で$(k-2)$個現れる余り同士でも対応する要素が等しいかどうかを調べる。そして残っている部分の対応する要素同士を余りを調べた時と同じ方向で比較する。次に2回目で$(k-1)$個の次元を$(k-1)$個の部分cubeで調べる。

簡単の為、先ず$k=3$の場合を考える。2個の部分cubeに分割するので、$n \bmod 2 = 1$の場合、部分cubeは$A_1 = A|_{d_1=1,2,\cdots,\frac{n-1}{2}}$、$A_2 = A|_{d_1=\frac{n+1}{2},\frac{n+3}{2},\cdots,n}$とする。任意の$s$ $(1 \le s \le \frac{n-1}{2})$, $t, u$ $(1 \le t, u \le n)$において、$A(s,t,u)$と$A(s+\frac{n-1}{2}, t, u)$を対応させると$A|_{d_1=n}$に対応する要素がなく、この部分が余る。第1ブロックでは最初にこの余りの部分を調べる。その結果、以下の3通りに場合分けされる。

$(\alpha)$ $A|_{d_1=n}$の要素全てが同じ値

$(\beta)$ 任意の$t$ $(1 \le t \le n)$において$A|_{d_1=n,d_2=t}$の要素全てが同じ値（但し$(\alpha)$の場合は除く）

$(\gamma)$ それ以外

続いて$A_1$と$A_2 - A|_{d_1=n}$の対応する要素を1組ずつ順に調べる。

$(\alpha)$の場合に$k$-PLANE$(A) = 1$となる為には、任意の$t (1 \le t \le n)$において、$A|_{d_1=t}$を構成する要素の値が全て同じかどうかを調べる。$A|_{d_1=t}$と$A|_{d_1=t+\frac{n-1}{2}}$の2個のまとまりを並行して調べる変数順序となる。成り立つ場合はこの時点で$k$-PLANE$(A) = 1$となり、成り立たない場合は0となる事が分かる。即ち、$(\alpha)$の場合は1回読みで十分である。

$(\beta)$の場合は、任意の$s$ $(1 \le s \le n-1)$, $t$ $(1 \le t \le n)$に対し、$A|_{d_1=s,d_2=t}$と$A|_{d_1=s+\frac{n-1}{2},d_2=t}$を並行して調べる変数順序となる。成り立つ場合は第2ブロックへ進む。

$(\gamma)$の場合は既に$d_1, d_2$に関して値が0となる事が判明している。$A_1$と$A_2 - A|_{d_1=n}$の対応する要素がどれも等しい時のみ、$d_3$に関して値が1となる可能性が残っているので、この条件を満たすかどうかを調べ、満たす場合は第2ブロックへ進む。

第2ブロックは$(\beta)$の場合を$A_1$で、$(\gamma)$の場合を$A_2$で調べるが、$(\beta)$の場合は$A_2$の一部である余りの部分との対応も調べなければならない為、$A_2$を調べる際に$A_1$を調べる部分を挿入する変数順序とする。$A_1$と余りである$A|_{d_1=n}$の部分において、$d_2$の値の等しい要素がそれぞれで同じ値である場合か、または$A_2$において、$d_3$の値の等しい要素がそれぞれで同じ値である場合に$k$-PLANE$(A) = 1$となる。

## 5 $k$-PLANEを表す2回読みblockwise分岐プログラム

次に任意の定数$k$について考える。$n \bmod (k-1)$の値は$0, 1, \cdots, k-2$となるが、0の時は従来の方法[2]で、1の場合は$k = 3$の場合と同様の方法で解けるが、$2, \cdots, k-2$の場合は余りが複数個になり、その部分の構成方法を追加する必要が出てくる。追加の仕方は対応部分の様に全ての余りを対応させて、全ての余りの対応している要素が等しい場合にのみ、先に進む。以降は$k=3$の時と同様に構成する。

以上より、任意の$k$に対して$k$-PLANEを2回読みblockwise分岐プログラムで表す事が可能である事が示された。

## 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 Blockwise Branching Programs", Technical Report of IEICE, COMP2001-12, 2001.