設 計 自 動 化 74－3
（1995．3．8）

疑似クロネッカー決定グラフに基づく AND-EXOR 多段論理回路構成法

笹尾勤, 浜地啓, 和田精二
九州工業大学情報工学部 電子情報工学科
〒820 飯塚市大字川津 680-4

あらまし: 疑似クロネッカ決定グラフ (PKDD) を用いた論理関数の表現法が知られている. PKDD では二分決定グラフ (BDD) に比べて節点数が 2 ～ 3 割小さいため, 多段論理回路の初期解として使用できる. 本稿では, PKDD に局所的変換法を適用した, AND, OR, EXOR, およびインバータを用いた多段論理回路の合成法を示す.

和文キーワード: 多段論理合成, EXOR, リード・マラー展開, 疑似クロネッカ決定グラフ, BDD, 局所的変換法

Multi-level Logic Synthesis Based on Pseudo-Kronecker Decision Diagrams and Local Transformation.

Tsutomu Sasao, Hiraku Hamachi, and Seiji Wada
Department of Computer Science and Electronics
Kyushu Institute of Technology, Iizuka 820, Japan

Abstract: This paper presents methods to derive multi-level logic networks using AND, OR, EXOR and inverters. The design method first generates a pseudo-Kronecker decision diagram (PKDDs) from given functions. Then, it generates multi-level logic networks consisting of AND, OR, EXOR and inverters. Finally, it simplifies the networks by using local transformation. Experimental results using MCNC benchmarks are shown.

Key words: multi-level logic synthesis, EXOR, Reed-Muller expansion, pseudo Kronecker decision diagram, BDD, local transformation.

1 まえがき

現在の論理合成システムは, AND, OR, 及び NOT を基本とした設計法を用いており, 制御回路等では人手設計の回路に匹敵する品質の回路を生成できる. しかし, 算術演算回路, 誤り訂正回路, 通信用回路等に関しては, 人手設計の回路に比べ, まだまだ改良の余地がある. これらの回路を能率よく合成するには AND, OR, NOT の他に, EXOR を効果的に活用した論理合成法の開発が必要である.

論理関数の表現法としてして, BDD(Binary Decision Diagram : 二分決定グラフ) がよく使用されている. 関数 f を表現する BDD の各節点を二入力のセレクタで置き換えると, 関数 f を実現する多段論理回路が得られる. ここでは, EXOR 演算を用いた決定グラフとして, PKDD(Pseudo Kronecker Decision Diagram : 疑似クロネッカ決定グラフ) を用いる. PKDD は, BDD を一般化した判定グラフであり, 関数を表現するための節点数は BDD よりも少ない. [15, 17] では, 与えられた関数を PKDD で表現するアルゴリズムを示した. PKDD では BDD に比べ節点数が 2 ~ 3 割小さくできる. 本稿では, 与えられた関数を AND, OR, EXOR 及びインバータを用いた多段論理回路で実現する問題について考察する. 与えらえた論理関数 f を PKDD を用いて表現し, 各節点を EXOR を含んだモジュール (図 1) で置き換えることにより, f を実現する多段論理回路が合成できる. ただし, このようにして生成した回路には, 冗長な部分を多く含むため, 局所的変換法を適用しゲート数を削減する.

本稿では, この方法を多数のベンチマーク関数に適用した結果を報告する.

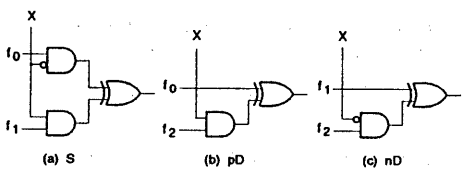

図 1: 各展開の論理回路による実現

2 二分決定グラフ (BDD)

任意の論理関数 f は

$$f = \bar{x}_1 f_0 \oplus x_1 f_1 \quad (1)$$

$$f = f_0 \oplus x_1 f_2 \quad (2)$$

$$f = \bar{x}_1 f_2 \oplus f_1 \quad (3)$$

の形で展開可能である. ここで, $f_2 = f_0 \oplus f_1$ である. (1) をシャノン展開, (2) を正極性ダビオ展開, (3) を負極性ダビオ展開と呼ぶ. 関数 f が与えられた時, f と展開すべき変数 x_1 が決まれば, 部分関数 f_0, f_1, f_2 は一意的に定まる. また, f_0, f_1 に対してシャノン展開を再び適用すれば

$$f_0 = \bar{x}_2 f_{00} \oplus x_2 f_{01},$$
$$f_1 = \bar{x}_2 f_{10} \oplus x_2 f_{11}$$

と展開できる. 同様に f_{00}, f_{10} に対して同じ展開を行うと

$$f_{00} = \bar{x}_3 f_{000} \oplus x_3 f_{001},$$
$$f_{01} = \bar{x}_3 f_{010} \oplus x_3 f_{011},$$
$$f_{10} = \bar{x}_3 f_{100} \oplus x_3 f_{101},$$
$$f_{11} = \bar{x}_3 f_{110} \oplus x_3 f_{111} \text{ を得る.}$$

図 2 はこの展開を示した二分決定木 (Binary Decision Tree) である. この決定木は, 次の二つの規則を用いて簡単化できる.

1. 部分関数が変数 x に依存しない場合は, x に関しては展開しない.
2. 部分関数が同じ時, 部分木を共有する.

この方法で簡単化したものが二分決定グラフ (Binary Decision Diagram : BDD) である. 関数 f と展開する変数の順序が与えられたとき, BDD は一意的に定まる. BDD を処理する高速アルゴリズムが開発されており, 論理合成では盛んに用いられている [2].

3 PKDD

f の展開木の各節点において, (1)~(3) の任意の展開を許すと, 例えば, 図 3 のような決定木が得られる. この決定木では, x_1 に関してはシャノン展開, x_2 に関し

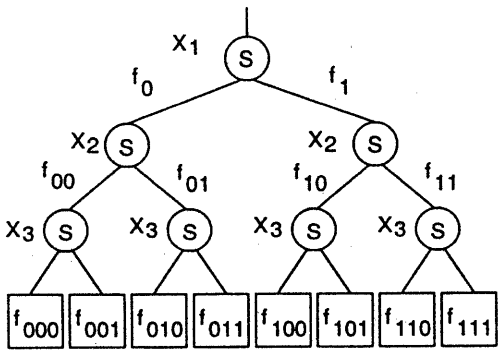

図 2: シャノン展開を用いた論理関数の表現

ては, 正極性ダビオ展開と負極性ダビオ展開, x_3 に関しては全ての展開法を用いている. このような展開を疑似クロネッカ展開 (pseudo-Kronecker expansion) と呼び, 簡単化した決定グラフを疑似クロネッカ決定グラフ (PKDD) という. 変数の順序を固定した時, 一つの関数に対して高々 3^{2^n-1} 個の PKDD が存在する. 節点数最小の PKDD を最小 PKDD と呼ぶ.

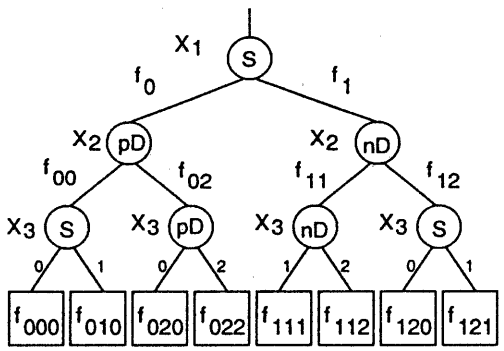

図 3: 疑似クロネッカ展開を用いた論理関数の表現

4 論理回路の生成

4.1 展開の方針

論理関数を展開する際, 部分関数の包含関係を調べることにより展開の方針を決定できる. ここでは, EXOR ゲートは, 同じ入力数の AND や OR に比べ高価であると仮定する.

定理 4.1 論理関数 f を $f = \bar{x}f_0 \oplus xf_1$ と展開したとき,

$$f_0 \subseteq f_1 \text{ ならば } f = f_0 \vee xf_1 \quad (4)$$

$$f_0 \supseteq f_1 \text{ ならば } f = \bar{x}f_0 \vee f_1 \quad (5)$$

と展開できる.

定理 4.2 論理関数 f を $f = g \oplus x^*h$ と展開したとき,

$$g \cdot h = 0 \text{ ならば } f = g \vee x^*h \quad (6)$$

$$g = 0 \text{ ならば } f = x^* \cdot h \quad (7)$$

$$g \supset h \text{ ならば } f = \overline{(x^* \cdot h)} \cdot g \quad (8)$$

と展開できる. ここで x^* は x あるいは $\bar{x}$ を表す.

(4) の場合 f は変数 x で正, (5) の場合 f は変数 x で負である. これらの条件が成立する場合, EXOR を用いない方が得策なので, PKDD を生成する際, シャノン展開を用いる. また, 多段論理回路を生成する際, シャノン展開の EXOR 演算子は OR に置き換える. (6) の条件は, (4) または (5) の条件と同じである. (7) は (6) の特別の場合である. (8) では, EXOR ゲートを AND ゲートとインバータ置き換えることができる.

4.2 回路の生成法

PKDD の各節点を, それぞれ図 1 の回路で置き換えると, f を実現する回路が得られる. 例えば, 表 1 の論理関数を PKDD に変換し, 各節点を図 1の回路で置き換えると, 図 4 が得られる. 論理関数を表現する場合, PKDD は BDD よりも節点数が少なくてよいので, PKDD に基づく決定グラフを用いることによって必要なゲート数を少なくできる.

5 局所的変換法

本構成法で使用した局所的変換法は, 以下の通りである.

- AND, OR の簡単化 (図 6)
  定数除去
  直列のゲート併合.
- インバータ除去 (図 7)
  直列インバータの削除.

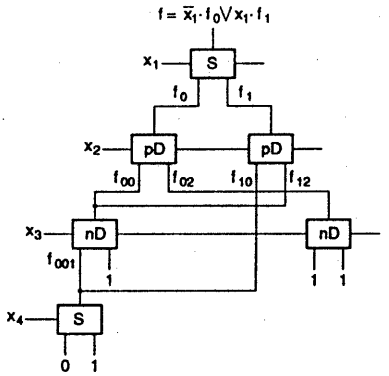

図 4: 表 1 の関数の回路実現

表 1: 4 変数関数

x_1	x_2	x_3	x_4	f
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

- EXOR の簡単化 (図 8)
  定数除去
  直列のゲート併合
  否定の抽出.
- AND(OR) と EXOR の簡単化 (図 9)
  カスケードにおける同一入力の除去.
- ゲート出力を考慮した簡単化 (図 10)
  定数関数検出
  同一出力の共有.

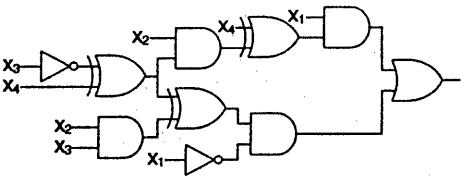

図 5: 図 4 の回路に局所的変換法を適用した結果

- 入力関数の包含関係を考慮した簡単化 (図 11)
- 入力関数と出力関数の包含関係を考慮した簡単化 (図 12)

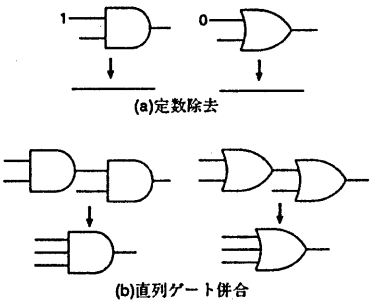

図 6: AND, OR の簡単化

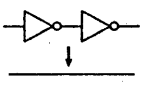

図 7: 直列インバータ除去

図 4 に示す回路の各モジュールを図 1 の回路で置き換え, 次に本章の局所的変換法を繰り返し適用すると, 図 5 の回路が得られる.

6 実験結果

与えられた関数を PKDD に変換するプログラム, 及び, PKDD を多段論理回路に変換するプログラムを開発

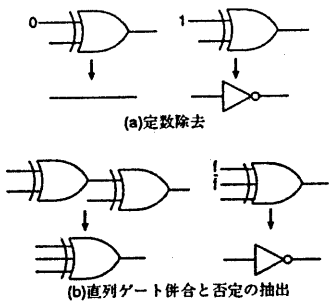

図 8: EXOR の簡単化

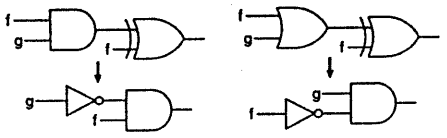

図 9: カスケードの簡単化

した. 表 2, 3 で, 第 4 列目は各種ベンチマーク関数を表現する BDD および PKDD の非終端接点数を示す. 各グラフに対して, 接点数がなるべく少なくなるような入力変数の順序を求めた. 入力順序の最適化には, シミュレーテッド・アニーリング法を用いた. BDD の接点数を最小化する変数順序は, PKDD では, 必ずしも接点数を最小化しない.

なお PKDD は 4.1 で述べた方針で作成したので, 節点数は必ずしも最小にはなっていない. (4) または (5) の条件が成立するときには, 図 12 の簡単化規則を用いてゲートを 1 つ削減できる.

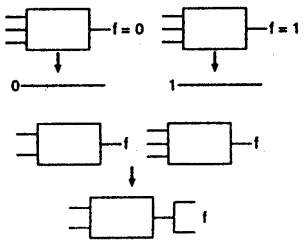

図 10: 定数関数の検出, 同一出力関数の共有

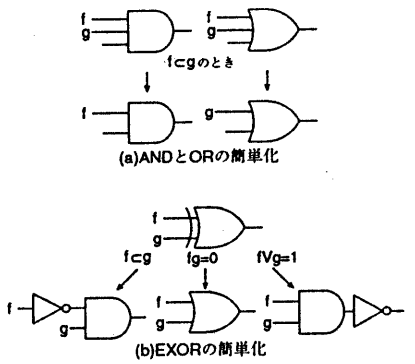

図 11: 入力関数の包含関係を考慮した簡単化

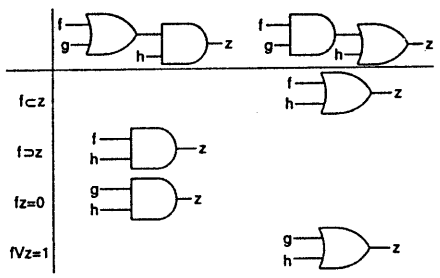

図 12: 入力関数と出力関数の包含関係を考慮した簡単化

表の 5 列目以降は生成した回路に局所的論理変換法を適用したもののゲート数, ファンイン数および段数を示す. 算術演算回路では, 段数は多少増加するものの, 既存の方法に比べゲート数を大幅に削減できた. 制御回路では, 既存の方法に比べゲート数を多数必要とする回路が多かった. 本手法では, 論理段数が増える傾向にあるが, その原因として次のことが考えられる.

1. 二入力ゲートを基本論理素子としているため, 多入力のゲートを基本とする手法に比べ, ゲート数や論理段数も増える.

2. 段数の計算法が異なる. 本手法では, AND, OR, EXOR, インバータのみの使用を許し, それぞれを一段と数えている. NAND, NOR, EXNOR 等の使用を許すことにより, ゲート数や段数を削減できる.

BDD に基づく回路と PKDD に基づく回路を比較す

ると, ゲート数やファンイン数は平均 2 割小さくなった. また add6, radd, rd73, rd84, t481, xor5 などでは, PKDD に基づく回路はゲート数, ファンイン数とも大幅に小さくなった.

7 あとがき

本稿では PKDD に局所的変換法を用いた多段論理合成法について述べた. PKDD は BDD よりも節点数が少なくてよく, 演算回路等の実現法として有望である.

与えられた関数が節点数の少ない判定図で表現できる場合, 本構成法は有効である. しかし, 関数によっては, AND-OR 論理和形を基本とした設計法の方が簡単になる場合もある. 実際の応用では, 与えられた関数の性質を判定し, 制御系などの浅い論理に対して, 従来の AND-OR 二段論理を基本とする合成法を適用し, 算術演算系などの深い論理では, BDD や PKDD などの決定グラフを基本とした合成法を適用すべきである. これは, 論理関数の最適表現法を見つける問題にも関係している.

参考文献

[1] S. D. Brown, R. J. Francis, J. Rose, and Z. G. Vranesic, *Field Programmable Gate Arrays*, Kluwer Academic Publishers, Boston 1992.

[2] R. E. Bryant, "Graph-based algorithms for Boolean function manipulation," *IEEE Trans. Comput.* Vol. C-35, No. 8, Aug. 1986, pp. 677-691.

[3] M. Davio, J-P Deschamps, and A. Thayse, *Discrete and Switching Functions*, McGraw-Hill International, 1978.

[4] R. Drechsler, A. Sarabi, M. Theobald, B. Becker and M. A. Perkowski, "Efficient representation and manipulation of switching functions based on ordered Kronecker Functional Decision Diagrams," Fachbereich Informatik, Universitat Frankfurt, Interner Bericht 14/93.

[5] K. Enomoto, S. Nakamura, T. Ogihara, and S. Murai, "LORES-2: A logic reorganization system, " *IEEE Design and Test*, Oct. 1985, pp.35-42.

[6] D. Gregory, K. Bartlett, A. de Geus, and G. Hachtel, "Socrates: A system for automatically synthesizing and optimizing combinationallogic," *Design Automation Conference 1986*, pp.79-85, June 1986.

[7] N. Ishiura, H. Sawada, and S. Yajima, "Minimization of binary decision diagrams based on exchange of variables," *ICCAD-91*, pp. 472- 475, Nov. 1991.

[8] U. Kebschull, E. Schubert and W. Rosenstiel, "Multilevel logic synthesis based on functional decision diagrams," *EDAC 92*, 1992, pp. 43-47.

[9] S. Minato, N. Ishiura, and S. Yajima, "Shared binary decision diagram with attributed edges for efficient Boolean function manipulation," *27th Design Automation Conference*, pp. 52-57, June, 1990.

[10] L. McKenzie, L. Xu and A. Almaini, "Graphical representation of generalized Reed-Muller Expansions," *IFIP 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, Hamburg, Sept. 16-17, 1993.

[11] T. Sasao, "Input variable assignment and output phase optimization of PLA's," *IEEE Trans. on Comput.* vol. C-33, No. 10, pp. 879-894, Oct. 1984.

[12] T. Sasao and P. Besslich, "On the complexity of MOD-2 Sum PLA's", *IEEE Trans.* on Comput. Vol.39. No.2,Feb.1990.

[13] T. Sasao (ed.), *Logic Synthesis and Optimization*, Kluwer Academic Publishers(1993-01).

[14] T. Sasao, "EXMIN2:A simplification algorithm for exclusive-OR Sum-of-products expressions for multiple-valued input two-valued output functions," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol.12, No.5, May 1993, pp.621-632.

[15] T. Sasao and J. T. Butler, "A design method for look-up table type FPGA by pseudo-Kronecker expansion," *Proc. International Symposium on Multiple-Valued Logic*, May 1994, pp.97-106.

[16] 笹尾勤, "FPGA の論理設計法", 情報処理, vol.35, No.6, pp.530- 534, 1994.

[17] 笹尾勤, 松浦, "EXOR を用いた決定グラフについて," 電子情報通信学会 VLSI 設計技術研究会, 情報処理学会設計自動化研究会, 1994 年 10 月.

表 2: BDD に局所的論理変換法を適用したもの

関数名	入力	出力	節点数	ゲート数				ファンイン数			段数
				NOT	AND	OR	XOR	AND	OR	XOR	
5xp1	7	10	68	21	33	30	0	68	61	0	10
add6	12	7	78	17	104	59	0	208	119	0	24
adr4	8	5	29	11	15	13	0	30	26	0	13
alu2	10	8	88	13	74	52	0	154	112	0	17
apla	10	12	102	10	81	51	0	185	103	0	16
bw	5	28	108	9	94	78	0	193	161	0	10
clip	9	5	105	18	97	58	0	196	129	0	17
co14	14	1	27	14	37	13	0	75	26	0	27
con1	7	2	64	6	12	10	0	25	21	0	7
dc2	8	7	64	7	53	35	0	116	70	0	13
dist	8	5	152	28	144	98	0	293	202	0	17
dk17	10	11	62	10	45	21	0	105	42	0	11
duke2	22	29	377	21	280	157	0	652	323	0	25
ex5	8	63	288	22	157	140	0	352	321	0	12
f51m	8	8	67	23	38	30	0	76	61	0	13
in2	19	10	232	17	269	163	0	565	334	0	22
in7	26	10	98	16	63	42	0	140	88	0	23
inc	7	9	72	16	58	39	0	127	79	0	13
misex1	8	7	38	10	29	20	0	67	40	0	9
misex3	14	14	585	50	505	319	0	1127	663	0	26
mlp4	8	8	141	12	160	111	0	330	224	0	17
radd	8	5	39	11	44	26	0	88	53	0	16
rd53	5	3	23	9	20	12	0	40	24	0	10
rd73	7	3	43	15	35	24	0	70	50	0	15
rd84	8	4	59	20	49	32	0	101	67	0	17
risc	8	31	67	8	49	21	0	107	45	0	8
rot8	8	5	75	15	57	42	0	121	86	0	15
sao2	10	4	85	10	77	43	0	161	100	0	17
sex	9	14	49	10	30	21	0	66	46	0	9
sqr8	8	16	233	33	253	171	0	518	343	0	16
sym6	5	2	15	5	11	8	0	22	18	0	7
sym9	9	1	33	9	31	20	0	62	44	0	14
t481	16	1	32	18	31	15	0	64	30	0	29
tial	14	8	692	82	609	377	0	1244	819	0	27
ts10	22	16	146	6	96	48	0	288	96	0	7
vg2	25	8	194	19	131	102	0	307	212	0	23
xor5	5	1	9	8	8	4	0	16	8	0	10
合計			4664	637	3891	2509	0	8390	5254	0	588

表 3: PKDD に局所的論理変換法を適用したもの

関数名	入力	出力	節点数	ゲート数				ファンイン数			段数
				NOT	AND	OR	XOR	AND	OR	XOR	
5xp1	7	10	33	12	17	5	14	35	10	30	12
add6	12	7	57	0	41	1	21	86	6	52	11
adr4	8	5	15	0	7	3	7	14	6	14	10
alu2	10	8	70	19	58	27	8	128	57	16	14
apla	10	12	96	10	81	51	0	185	103	0	16
bw	5	28	91	14	86	43	20	185	87	40	8
clip	9	5	79	19	62	39	12	127	87	29	15
co14	14	1	26	14	24	6	7	48	12	14	26
con1	7	2	15	6	13	9	0	27	19	0	7
dc2	8	7	56	8	39	21	5	90	44	11	12
dist	8	5	132	21	105	59	26	213	122	57	15
dk17	10	11	61	11	46	20	0	107	40	0	11
duke2	22	29	343	37	268	112	16	634	227	33	23
ex5	8	63	228	39	166	85	18	372	194	37	15
f51m	8	8	30	10	17	4	13	34	8	28	12
in2	19	10	205	23	253	149	2	536	301	4	23
in7	26	10	84	19	53	17	17	126	35	38	20
inc	7	9	65	12	53	27	11	115	55	23	11
misex1	8	7	34	9	29	18	1	66	36	2	9
misex3	14	14	500	36	437	222	90	953	452	185	26
mlp4	8	8	99	9	86	30	27	176	64	69	13
radd	8	5	26	0	16	1	10	34	4	23	7
rd53	5	3	13	0	8	2	6	16	4	13	7
rd73	7	3	21	0	14	2	12	28	6	25	9
rd84	8	4	29	0	18	2	16	39	6	33	10
risc	8	31	55	9	42	12	0	91	26	0	8
rot8	8	5	64	17	59	31	6	122	62	12	15
sao2	10	4	76	10	64	33	6	135	80	12	17
sex	9	14	44	14	34	11	0	81	23	0	9
sqr8	8	16	171	19	144	63	51	293	126	122	14
sym6	5	2	15	7	12	7	0	24	16	0	8
9sym	9	1	26	10	20	1	17	40	4	36	12
t481	16	1	17	10	9	0	3	22	0	9	6
tial	14	8	413	39	377	231	42	774	545	85	25
ts10	22	16	146	6	96	48	0	288	96	0	7
vg2	25	8	191	24	137	94	0	315	196	0	22
xor5	5	1	5	0	0	0	1	0	0	5	1
合計			3655	511	3003	1490	485	6590	3167	1057	492