アルゴリズム 33−14
(1993. 5. 29)

単項演算に対する局所計算可能な符号化とその符号長に関する考察

永浦 渉　　安浦 寛人

九州大学 大学院総合理工学研究科 情報システム学専攻

〒816 福岡県春日市春日公園 6-1

E-mail: {nagaura, yasuura}@is.kyushu-u.ac.jp

ある符号化Cの下で，演算結果の各桁がオペランドの高々k桁だけに依存するような演算の計算規則が作れるとき，この演算は符号化Cの下でk−局所計算可能であるという．本論文では，全てのn入力n出力論理関数をある符号化の下で局所計算可能とする際に必要な符号長の下界を関数の数を調べることにより求めた．その結果，入出力数nを増加させると，符号化に必要な符号長は指数関数的に増加することがわかった．

A Consideration on Locally Computable Coding and Its Code Length

Wataru Nagaura　Hiroto Yasuura

Department of Information Systems
Interdisciplinary Graduate School of Engineering Sciences
Kyushu University
Kasuga-shi, Fukuoka 816 Japan

E-mail: {nagaura,yasuura}@is.kyushu-u.ac.jp

When an operation defined on a finite set can be realized under a coding C by a computation rule in which each digit of the result of the operation depends on at most k digits of the operand, the operation is said to be k−locally computable under the coding C. In this report, we show a lower bound of the code length which is needed when all logical functions with n inputs and n outputs are locally computable under a coding. The result is that the code length increases as an exponential function of n.

1 はじめに

1960年台初頭，Avizienisは算術演算回路の設計において，冗長符号化が高速アルゴリズムの実現に利用できることを指摘した[1]．通常の2進表現では，加算の場合，最上位桁は加数と被加数の全ての桁に依存するので少なくとも桁数の対数に比例する時間が必要となる．しかし，冗長符号化を利用すると，桁上げ伝搬のない加算器が実現でき，桁数に無関係に一定の時間で計算が可能である[4]．ある演算において，演算の計算時間とその演算結果の各桁が依存するオペランド数は深い関係にある．多くのオペランドに依存する桁の計算に要する時間は大きくなる．演算結果の各桁に依存するオペランド数をある一定数以下に制限するように符号化を工夫すれば，符号長は大きくなるが全体の計算時間は小さくできると考えられる．この考えに基づいて局所計算可能性という概念が提案された．この局所計算可能性を実現するには非冗長符号化では一般に実現できないことが示されている[2]．ある符号化$\mathcal{C}$の下で，演算結果の各桁がオペランドの高々k桁だけに依存するような演算の計算規則が作れるとき，この演算は符号化$\mathcal{C}$の下でk−局所計算可能であるという[2]．冗長2進表現を用いた各種算術演算アルゴリズムなど，冗長符号化を利用して局所計算可能にしたアルゴリズムがいくつか示されている[2][3]．さらに，局所計算可能性は，順序回路の高速化や，多出力組合せ回路の検査容易設計などでの応用が期待されている．

ここでは，ある有限集合とその上で定義された複数の単項演算に対してk−局所計算可能な符号化を与えることを考える．要素数nの任意の集合とその上で定義された任意の単項演算集合に対して2−局所計算可能にできる符号化が存在し，その符号長は$O(n^2 \log n)$となることが知られている[2]．しかし，これでは符号長が膨大となる．そこで，符号長をもっと小さくできる符号化を求めたい．任意の集合とその上で定義された単項演算を冗長符号化によりk−局所計算可能な関数として実現するときに要する符号長の上下限がわかれば符号化を考える際に重要な情報となる．この問題に対する1つのアプローチとして，本論文では，全てのn入力n出力論理関数をある単一の符号化によりk− 局所計算可能な論理関数として実現する場合に必要な符号長の下界を求める．

本論文の内容としては，2章では，本論文で用いる語句の定義を行なう．3章では，全てのn入力n出力論理関数をある単一の符号化によりk− 局所計算可能な論理関数として実現する場合に必要な符号長の下界を求める．4章では，3章で求めた符号長の下界についての考察を行なう．

2 諸定義

この章では，本論文で用いる語句の定義を行なう[2][3][5][6]．

2.1 冗長符号化

Sを有限集合とする．$OP = \{op_1, op_2, \cdots, op_m\}$を$S$上で定義された単項演算の集合とする．$\sum$を符号を構成する有限のアルファベットとする．本論文では$\sum = \{0, 1\}$とする．$\sum^n$(nは正整数)を$\sum$上の長さnの系列の集合とする．nを符号長と呼ぶ．Sに対する$\sum^n$上での符号化$\mathcal{C}$を次のように定義する．

- 符号化$\mathcal{C}$は$\sum^n$の部分集合からSへの全写である．
- $\forall s \in S$に対し$\exists \mathbf{c} \in \sum^n$があり，$s = \mathcal{C}(\mathbf{c})$である．

$\mathbf{c} \in \sum^n, s \in S$について$\mathcal{C}(\mathbf{c}) = s$であるとき，$\mathbf{c}$を$s$に対する符号語という．$S$の要素数を$|S|$とする．以下，任意の集合$A$の要素数を$|A|$と表記する．符号長$n = \lceil \log(|S|) \rceil$のとき，$\mathcal{C}$を最短符号化と呼ぶ．すべての$s$に対応する符号語が唯一であるとき，$\mathcal{C}$は単一符号化であるといい，ある$s$に対応する符号語が2つ以上あるとき，$\mathcal{C}$は多重符号化であるという．最短かつ単一の符号化を非冗長符号化と呼び，それ以外を冗長符号化と呼ぶ．

S上の単項演算opと関数$\mathbf{f} : \sum^n \to \sum^n$に対し

$$\mathcal{C}(\mathbf{f}(\mathbf{c})) = op(\mathcal{C}(\mathbf{c}))$$

が成り立つとき$\mathbf{f}$は符号化$\mathcal{C}$の下でopを実現するという．

2.2 関数と部分関数

$\mathbf{c}$を$\sum^n$の要素とすると$\mathbf{c}$は

$$\mathbf{c} = (c_1, c_2, \cdots, c_n), (c_i \in \sum)$$

とベクトル表現できる．c_iを第i桁と呼ぶ．$\mathbf{f}:\sum^n \to \sum^n$において，入力を$(x_1,x_2,\cdots,x_n)$，出力を$(y_1,y_2,\cdots,y_n)$とすれば

$$\begin{aligned}(y_1,y_2,\cdots,y_n) &= (\mathbf{f}(x_1,x_2,\cdots,x_n)) \\ &= (f_1(x_1,x_2,\cdots,x_n), \\ &\quad f_2(x_1,x_2,\cdots,x_n), \\ &\quad \cdots, \\ &\quad f_n(x_1,x_2,\cdots,x_n))\end{aligned}$$

と書ける．但し，$f_i:\sum^n \to \sum$である．したがって$\mathbf{f}$は

$$\mathbf{f}=(f_1,f_2,\cdots,f_n)$$

とベクトル表現できる．$f_i(i=1,\cdots,n)$を$\mathbf{f}$の部分関数と呼ぶ．

$\sum^n$上で定義された関数の集合を

$$F^n=\{\mathbf{f}|\mathbf{f}:\sum^n \to \sum^n\}$$

とする．$\mathbf{f}\in F^n$の部分関数の集合を

$$Fp^n=\{f|f:\sum^n \to \sum\}$$

とする．

$f\in Fp^n$の変数をベクトル

$$\mathbf{x}=(x_1,x_2,\cdots,x_n)$$

また，変数の集合を

$$X=\{x_1,x_2,\cdots,x_n\}$$

とする．

$\mathbf{x}=(x_1,x_2,\cdots,x_n)$において，$X=\{x_1,x_2,\cdots,x_n\}$から$X_i=\{x_{i_1},x_{i_2},\cdots,x_{i_m}\},(X_i\subseteq X)$を取り除いて，$\mathbf{x}'=(x_1,\cdots,x_{i_1-1},x_{i_1+1},\cdots,x_{i_2-1},x_{i_2+1},\cdots,x_{i_m-1},x_{i_m+1},\cdots,x_n)$としたとき，$\mathbf{x}'$を$\mathbf{x}_{<I>},(I=\{i_1,i_2,\cdots,i_m\})$と表記する．$X$に対する$X_i$の補集合を$\overline{X_i}(=X-X_i)$と表記する．

$f,f'\in Fp^n$において，$f\equiv f'$とは，$\forall \mathbf{c}\in\sum^n$に対し，$f(\mathbf{c})=f'(\mathbf{c})$が成立することである．逆に，$f\not\equiv f'$とは，$\exists \mathbf{c}\in\sum^n$があって，$f(\mathbf{c})\neq f'(\mathbf{c})$が成立することである．

$f\in Fp^n$が全ての変数に依存しているとき，fは非縮退であるという．また，fは真にn変数に依存する関数であるという．そうでないとき，fは縮退しているという．または，縮退可能であるという．

$f\in Fp^n$において，$f(x_1,x_2,\cdots,x_i=1,\cdots,x_n)$を$f_i$，$f(x_1,x_2,\cdots,x_i=0,\cdots,x_n)$を$f_{\overline{i}}$と表記する．

$\forall f\in Fp^n$において，shannon展開により

$$f(\mathbf{x})=x_if_i+\overline{x_i}f_{\overline{i}}$$

と書ける．このとき，$f_i\neq f_{\overline{i}}$のときfはx_iに依存するという．$f_i=f_{\overline{i}}$のときfはx_iに関して独立であるという．また，x_iを冗長な変数という．

$f\in Fp^n$がx_iに関して独立であるとき，

$$f_i=f_{\overline{i}}=g(\mathbf{x}_{<\{i\}>})$$

となる$g\in Fp^{n-1}$が存在する．このとき

$$\begin{aligned}f(\mathbf{x}) &= x_i\cdot f_i+\overline{x_i}\cdot f_{\overline{i}} \\ &= x_i\cdot g(\mathbf{x}_{<\{i\}>})+\overline{x_i}\cdot g(\mathbf{x}_{<\{i\}>}) \\ &= (x_i+\overline{x_i})\cdot g(\mathbf{x}_{<\{i\}>}) \\ &= g(\mathbf{x}_{<\{i\}>})\end{aligned}$$

が成立するので，fとgを同一視し，$f\approx g$または，$g\approx f$と表記する．また，gはfを冗長な変数x_iに関して縮退した関数という．

2.3 局所計算可能性

$\mathbf{f}\in F^n$の各部分関数$f_i(i=1,\cdots,n)$すべてがそれぞれ高々k個の入力変数にしか依存しないとき，$\mathbf{f}$はk−局所計算可能(k−locally computable)であるという．以下,「局所計算可能」を「$l.c.$」と略記する．kを$\mathbf{f}$の入力に対する依存度と呼ぶ．さらに，部分関数fにおいてちょうどl個の入力変数に依存する場合fは"真に$l-l.c.$な部分関数"という．

$\mathbf{f}$が符号化$\mathcal{C}$の下でopを実現し，かつ，$\mathbf{f}$が$k-l.c.$であるときopは符号化$\mathcal{C}$の下で$k-l.c.$であるという．

定義 1

S上の単項演算の集合$OP=\{op_1,op_2,\cdots,op_m\}$が符号化$\mathcal{C}$の下で$k$−弱局所計算可能*(weakly k−locally computable)*であるとは，各演算op_iがk−局所計算可能であることである．

定義 2

S上の単項演算の集合$OP=\{op_1,op_2,\cdots,op_m\}$が符号化$\mathcal{C}$の下で$k$−弱局所計算可能であるとする．各演算$op_i$を実現する関数$\mathbf{f}_i(i=1,2,\cdots,m)$において，第$j$桁を計算する部分関数が依存する$k$個の入力変数を共通にすることができるとき，単項演算の集

合OPは符号化Cの下でk−強局所計算可能 *(strongly k−locally computable)* であるという．

F^nの中の関数で$k-l.c.$なものの集合を

$$F^{n,k}=\{\mathrm{f}|\mathrm{f}\in F^n \text{かつ} k-l.c.\}$$

と表記する．同様に，Fp^nの中の部分関数で$k-l.c.$なものの集合を

$$Fp^{n,k}=\{f|f\in Fp^n \text{かつ} k-l.c.\}$$

と表記する．$Fp^{n,k}$の中で，依存する変数の集合が$X_i(\subset X)$である要素の集合を$Fp^{n,k}_{X_i}$，真に$j-l.c.$な要素の集合を$Fp^{n,\widetilde{j}},(j\leq k)$と表記する．$Fp^{j,\widetilde{j}}$は単に$Fp^{\widetilde{j}}$と表記する．

S上の単項演算の集合$OP=\{op_1,op_2,\cdots,op_m\}$が符号化$C$の下で$k$−局所計算可能であるとする．このときの符号長が$n$のとき$OP$の各要素$op_i$はそれぞれ符号化$C$の下で$F^{n,k}$の中のある要素により実現されている．よってこれを$OP$は符号化$C$の下で$F^{n,k}$のある部分集合$F^{n,k}_{OP,C}$で実現されているという．

3 局所計算可能な符号化に必要な符号長

単項演算の集合OPと依存度kが与えられたとき，$k-l.c.$な符号化の符号長をOPとkの関数として表す問題を考える．しかし，この問題は，OPの各要素の性質を考慮に入れて議論しなければならず，難しい問題となる．そこで，問題を簡単化するために，$S=\sum^n$として，全ての$\sum^n$上の単項演算すなわちF^nを単一な符号化Cにより$k-l.c$な関数$F^{n_c,k}_{F^n,C}$で実現することを考える．ここで，n_cは符号化Cによる符号長である．これより，符号化Cの対象となる$\sum^n$のベクトル長nを単にnと表記し，符号化Cによる符号長n_cを単にn_cと表記する．

符号化Cは，F^nの各要素全てを$k-l.c.$な関数とするものであるから，もし存在したとしてもn_cはかなり大きくなるのではないかと思われる．そこで，本論文では，n_cの下界を求め，符号化Cについて議論することにする．

n_cの下界が満たす必要条件としては，$F^{n_c,k}_{F^n,C}$は$F^{n_c,k}$の部分集合であるから，$F^{n_c,k}$の要素数$|F^{n_c,k}|$がF^nの要素数$|F^n|$以上でなければ$F^{n_c,k}_{F^n,C}$がF^nを実現することは不可能である．したがってn_cは最低でも

$$|F^{n_c,k}|\geq|F^n| \qquad (1)$$

の条件を満たさなければならない．

よって，この章では関数の数$|F^{n_c,k}|$,$|F^n|$を調べることにより，条件式(1)を満たすことでn_cの下界を求めることにする．

補題 1

$|F^n|=2^{n\cdot 2^n}$である．

(証明)

$\mathrm{f}\in F^n$はn個の部分関数から構成されている．よってfは部分関数の集合Fp^nの$|Fp^n|$個の要素を重複を許してn個取って並べたものと考えることができる．したがって$|F^n|$は

$$|F^n|=|Fp^n|^n=\{2^{2^n}\}^n=2^{n\cdot 2^n} \qquad (2)$$

となる．ただし，$|Fp^n|=2^{2^n}$である[5][6]. ■

F^nの要素は，Fp^nの要素から構成されている．同様に，$F^{n,k}$の要素は，$Fp^{n,k}$の要素から構成されている．ここで，$Fp^{n,k}$の性質を簡単にまとめると，次のようになる．

- $Fp^{n,k}=Fp^{n,\widetilde{k}}\cup Fp^{n,\widetilde{k-1}}\cup\cdots\cup Fp^{n,\widetilde{1}}\cup Fp^{n,\widetilde{0}}$かつ，任意の2つの集合$Fp^{n,\widetilde{i}},Fp^{n,\widetilde{j}}\subset Fp^{n,k}$に対し，$i\neq j$ならば$Fp^{n,\widetilde{i}}\cap Fp^{n,\widetilde{j}}=\emptyset$である．
- $Fp^{n,\widetilde{j}}=Fp^{n,\widetilde{j}}_{X_1}\cup Fp^{n,\widetilde{j}}_{X_2}\cup\cdots\cup Fp^{n,\widetilde{j}}_{X_m},(X_i\subset X,|X_i|=j,X_i\neq X_k(i\neq k))$である．また，任意の2つの集合$Fp^{n,\widetilde{j}}_{X_i},Fp^{n,\widetilde{j}}_{X_k}$に対し，$i\neq k$ならば，$Fp^{n,\widetilde{j}}_{X_i}\cap Fp^{n,\widetilde{j}}_{X_k}=\emptyset$である．
- $\forall Fp^{n,\widetilde{j}}_{X_i}\subset Fp^{n,\widetilde{j}}$の各要素は，$Fp^{\widetilde{j}}$の要素と1対1対応となっている．

$|F^{n,k}|$を求めるには，$|Fp^{n,k}|$を求める必要がある．そこで，まず，上に述べた$Fp^{n,k}$の各性質が成り立つことを証明する．次に，それらの性質を用いて$|Fp^{n,k}|$を求める．そして，$|F^{n,k}|$を求める．

補題 2

$f\in Fp^{n,k}$において，fに独立な変数の集合をX_iとする．$f(\mathbf{x})$をX_iに関して縮退した関数を$g(\mathbf{x}_{<I>})$とすれば，$\forall\mathrm{c}\in\sum^n$に対して$f(\mathrm{c})=g(\mathrm{c}_{<I>})$が成立する．ただし，$I$は，$X_i=\{x_{i_1},x_{i_2},\cdots,x_{i_m}\}$としたとき，$I=\{i_1,i_2,\cdots,i_m\}$とする．

(証明)

縮退の定義より明らか. ■

補題 3

$f, f' \in Fp^{n,\widetilde{j}}_{\widetilde{X_i}}$において，$f, f'$をそれぞれ$X_i$に関して縮退した関数を$g, g' \in Fp^{\widetilde{j}}$とすれば，次のことが成立する．

$$f \equiv f' \text{ならば } g \equiv g' \quad (i)$$

$$f \not\equiv f' \text{ならば } g \not\equiv g' \quad (ii)$$

(証明)

$f(\mathbf{x}) \approx g(\mathbf{x}_{<I>}), f'(\mathbf{x}) \approx g'(\mathbf{x}_{<I>})$ より，$\forall \mathbf{c} \in \sum^n$に対して

$$f(\mathbf{c}) = g(\mathbf{c}_{<I>})$$

$$f'(\mathbf{c}) = g'(\mathbf{c}_{<I>})$$

が成立する．ここで，Iは，$X_i = \{x_{i_1}, x_{i_2}, \cdots, x_{i_{n-j}}\}$としたとき，$I = \{i_1, i_2, \cdots, i_{n-j}\}$とする．

(i)のとき$f \equiv f'$より，$\forall \mathbf{c} \in \sum^n$に対して

$$f(\mathbf{c}) = f'(\mathbf{c})$$

が成立しているから，$\forall \mathbf{c} \in \sum^n$に対して

$$g(\mathbf{c}_{<I>}) = g'(\mathbf{c}_{<I>})$$

が成立する．すなわち，$g \equiv g'$となる．

(ii)のときは，

$$f(\mathbf{c}) \neq f'(\mathbf{c})$$

となる$\mathbf{c}$が存在する．よってその$\mathbf{c}$に対しては，

$$g(\mathbf{c}_{<I>}) \neq g'(\mathbf{c}_{<I>})$$

が成立する．すなわち，$g \not\equiv g'$となる． ■

補題 4

$\forall f \in Fp^{n,\widetilde{j}}_{X_i}, \forall f' \in Fp^{n,\widetilde{j}}_{X'_i}$において，$X_i \neq X'_i$ならば$f \not\equiv f'$である．

(証明)

fはX_iに，f'はX'_iに依存する．$X_i \neq X'_i$であるから，$X_i \cap X'_i$に属さない$x_j, (x_j \in X_i, x_j \notin X'_i)$が少なくとも1つは存在する．$f, f'$はともに，

$$f(\mathbf{x}) = x_j f_j + \overline{x_j} f_{\overline{j}}$$

$$f'(\mathbf{x}) = x_j f'_j + \overline{x_j} f'_{\overline{j}}$$

と書くことができる．fはx_jに依存しているから$f_j \neq f_{\overline{j}}$であり，$f'$は$x_j$に独立であるから$f'_j = f'_{\overline{j}}$である．もし，$f \equiv f'$と仮定するならば，明らかに矛盾している．従って，$f \not\equiv f'$である． ■

補題 5

$|Fp^{n,\widetilde{j}}| = {}_nC_j |Fp^{\widetilde{j}}|$である．

(証明)

$Fp^{n,\widetilde{j}} = Fp^{n,\widetilde{j}}_{X_1} \cup Fp^{n,\widetilde{j}}_{X_2} \cup \cdots \cup Fp^{n,\widetilde{j}}_{X_m}$ である．ただし，$X_i \subset X, |X_i| = j, X_i \neq X_k (i \neq k)$である．補題3より，$Fp^{n,\widetilde{j}}_{X_i}$の各要素は，$Fp^{\widetilde{j}}$の要素と1対1対応となっている．また，$Fp^{n,\widetilde{j}}_{X_i}$の中に$Fp^{\widetilde{j}}$の要素に対応する要素が必ず1つ存在する．従って，$Fp^{n,\widetilde{j}}_{X_i}$の要素数と$Fp^{\widetilde{j}}$の要素数は等しくなり，

$$|Fp^{n,\widetilde{j}}_{X_i}| = |Fp^{\widetilde{j}}|$$

となる．これは他の集合に対しても同様である．

さらに，$\forall f \in Fp^{n,\widetilde{j}}$は，補題4より複数の集合の要素となることはなく，その関数が属する集合は唯一決まる．従って，任意の2つの集合$Fp^{n,\widetilde{j}}_{X_i}, Fp^{n,\widetilde{j}}_{X_j} \subset Fp^{n,\widetilde{j}}$に対して$X_i \neq X_j$ならば，$Fp^{n,\widetilde{j}}_{X_i} \cap Fp^{n,\widetilde{j}}_{X_j} = \emptyset$となる．これより，$|Fp^{n,\widetilde{j}}| = m|Fp^{\widetilde{j}}|$となる．さらに集合の数$m$は$n$個の変数から$k$個選ぶ組合せの数となるから$m = {}_nC_j$となる．従って，

$$|Fp^{n,\widetilde{j}}| = {}_nC_j |Fp^{\widetilde{j}}|$$

となる． ■

補題 6

$|Fp^{n,k}| = \sum_{i=0}^{k} ({}_nC_i |Fp^{\widetilde{i}}|)$である．

(証明)

$Fp^{n,k} = Fp^{n,\widetilde{k}} \cup Fp^{n,\widetilde{k-1}} \cup \cdots \cup Fp^{n,\widetilde{1}} \cup Fp^{n,\widetilde{0}}$である．$f \in Fp^{n,k}$は，依存する変数の集合は唯一決まるから，その変数の数がjであれば，fは$Fp^{n,\widetilde{j}}$に属する．このように，$\forall f \in Fp^{n,k}$は，複数の集合に属することは決してあり得ない．すなわち，任意の2つの集合$Fp^{n,\widetilde{i}}, Fp^{n,\widetilde{j}} \subset Fp^{n,k}$に対し，$i \neq j$ならば$Fp^{n,\widetilde{i}} \cap Fp^{n,\widetilde{j}} = \emptyset$となる．よって，$Fp^{n,k}$の要素数は各集合$Fp^{n,\widetilde{i}}$の要素数の和となる．従って，

$$|Fp^{n,k}| = \sum_{i=0}^{k} |Fp^{n,\widetilde{i}}| = \sum_{i=0}^{k} ({}_nC_i |Fp^{\widetilde{i}}|)$$

となる． ■

補題 7

$|Fp^{\widetilde{k}}| = 2^{2^k} - \sum_{i=0}^{k-1} ({}_kC_i |Fp^{\widetilde{i}}|)$,
$|Fp^{\widetilde{0}}| = 2$である．

(証明)

$|\widetilde{Fp^k}| = |Fp^k| - |Fp^{k,k-1}|$ は明らかである．また，$|Fp^k| = 2^{2^k}$ も明らかである．$|Fp^{k,k-1}|$ は補題6より $|Fp^{k,k-1}| = \sum_{i=0}^{k-1}({}_kC_i|\widetilde{Fp^i}|)$ である．従って，

$$|\widetilde{Fp^k}| = |Fp^k| - |Fp^{k,k-1}| = 2^{2^k} - \sum_{i=0}^{k-1}({}_kC_i|\widetilde{Fp^i}|)$$

である．なお，$|\widetilde{Fp^0}|$ は，$f=1, f=0$ の2通りであるから，$|\widetilde{Fp^0}| = 2$ である． ■

これより，条件式(1)

$$|F^{n_c,k}| \geq |F^n|$$

により，n_cの最低限必要な符号長を求める．

定理 1

F^nの全要素をを符号化Cにより弱−局所計算可能な関数として実現する際に必要な符号長の下界は，

$$\left\{\sum_{i=0}^{k}({}_{n_c}C_i \cdot |\widetilde{Fp^i}|)\right\}^{n_c} \geq 2^{n\cdot 2^n} \qquad (3)$$

を満たす最小の整数n_cである．

(証明)

k−弱局所計算可能な関数の集合を $F_{weak}^{n_c,k}$とする．$\forall \mathrm{f} \in F_{weak}^{n_c,k}$の各出力 $y_i, (i=1,2,\cdots,n_c)$ は部分関数が $k-l.c.$であるという条件さえ満たしておけば良い．よって，$F_{weak}^{n_c,k}$の要素は，$k-l.c.$な部分関数の集合 $Fp^{n_c,k}$の要素を重複を許して n_c個取って並べる順列の総数通り作ることができる．したがって，$F_{weak}^{n_c,k}$の要素数 $|F_{weak}^{n_c,k}|$ は

$$|F_{weak}^{n_c,k}| = |Fp^{n_c,k}|^{n_c} = \left\{\sum_{i=0}^{k}({}_{n_c}C_i \cdot |\widetilde{Fp^i}|)\right\}^{n_c} \qquad (4)$$

となる．条件式(1)より n_cの下界は

$$|F_{weak}^{n_c,k}| \geq |F^n|$$

を満たす最小の整数である．よって，定理1が証明された． ■

定理 2

F^nの全要素をを符号化Cにより強−局所計算可能な関数として実現する際に必要な符号長の下界は，

$$n_c \geq \frac{n\cdot 2^n}{2^k} \qquad (5)$$

を満たす最小の整数n_cである．

(証明)

k−強局所計算可能な関数の集合を $F_{strong}^{n_c,k}$とする．$\mathrm{f} \in F_{strong}^{n_c,k}$の各出力 $y_i, (i=1,2,\cdots,n_c)$ は y_iの部分関数の入力が $\{x_{i_1}, x_{i_2}, \cdots, x_{i_k}\}$に依存したとすると，他の関数全てにおいて y_iに依存する入力変数は集合 $\{x_{i_1}, x_{i_2}, \cdots, x_{i_k}\}$ の要素でなければならない．$\mathrm{f} \in F_{strong}^{n_c,k}$の要素を $\mathrm{f} = \{f_1^{n_c,k}, f_2^{n_c,k}, \cdots, f_{n_c}^{n_c,k}\}$ とし，$f_i^{n_c,k}$に依存する入力変数の集合を $\mathbf{x}_i = \{x_{i_1}, x_{i_2}, \cdots, x_{i_k}\}$とする．入力変数の組合せが $\mathbf{x}_i$と決まっている場合，$\mathbf{x}_i$を 入力変数とする $k-l.c.$な部分関数は $|Fp^k| = 2^{2^k}$ 個作れる．なぜならば，Fp^kの入力はk個あり，それぞれの入力変数を $\mathbf{x}_i$とすれば良いからである．したがって，$\mathrm{f} \in F_{strong}^{n_c,k}$は $\mathbf{x}_1$に依存する部分関数から $\mathbf{x}_{n_c}$に依存する部分関数までを1つづつ選んで並べる順列通り作るここができるので，$F_{strong}^{n_c,k}$の要素数 $|F_{strong}^{n_c,k}|$ は

$$|F_{strong}^{n_c,k}| = |Fp^k|^{n_c} = 2^{n_c\cdot 2^k} \qquad (6)$$

となる．

条件式(1)より n_cの下界は

$$|F_{strong}^{n_c,k}| \geq |F^n|$$

を満たす最小の整数である．すなわち，

$$2^{n_c\cdot 2^k} \geq 2^{n\cdot 2^n}$$

$$\therefore\ n_c \geq \frac{n\cdot 2^n}{2^k}$$

である．よって，定理2が証明された． ■

4 考察

4.1 弱局所計算可能性

表4.1に n が1から7までの $k-l.c.$な部分関数の数，真に $k-l.c.$な部分関数の数及び，式(4)より弱局所計算可能な関数の数求めたものを示す．

表からわかるように k(以下 kと略記する) が5以上のときは

$$|\widetilde{Fp^k}| \simeq |Fp^k| = 2^{2^k}$$

と近似することができる．$k \geq 5$ のとき

$$\begin{aligned} &{}_nC_k \cdot |\widetilde{Fp^k}| - {}_nC_{k-1} \cdot |\widetilde{Fp^{k-1}}| \\ &\simeq\ {}_nC_k \cdot |Fp^k| - {}_nC_{k-1} \cdot |Fp^{k-1}| \\ &=\ {}_nC_k \cdot 2^{2^k} - {}_nC_{k-1} \cdot 2^{2^{k-1}} \end{aligned}$$

$$= \frac{n!}{k!\cdot(n-k)!}\cdot 2^{2^k} - \frac{n!}{(k-1)!\cdot(n+1-k)!}\cdot 2^{2^{k-1}}$$

$$= \frac{n!}{k!\cdot(n-k)!}\cdot 2^{2^k}\left(1-\frac{k}{n+1-k}\cdot\frac{1}{2^{2^{k-1}}}\right)$$

$$\simeq \frac{n!}{k!\cdot(n-k)!}\cdot 2^{2^k} = {}_nC_k\cdot 2^{2^k}$$

$$k\geq 5 \text{ のとき } 1 >> \frac{k}{n+1-k}\cdot\frac{1}{2^{2^{k-1}}}$$

したがって，$k\geq 5$ のとき

$$\begin{aligned}|Fp^{n,k}| &= \sum_{i=0}^{k}({}_nC_i\cdot|Fp^{\widetilde{i}}|)\\ &= {}_nC_k\cdot|Fp^{\widetilde{k}}| + {}_nC_{k-1}\cdot|Fp^{\widetilde{k-1}}| + \cdots\\ &\quad + {}_nC_0\cdot|Fp^{\widetilde{0}}|\\ &\simeq {}_nC_k\cdot|Fp^{\widetilde{k}}|\\ &= {}_nC_k\cdot 2^{2^k}\end{aligned}$$

と近似できるので弱局所計算における条件式 (3) はこの近似式を使って，kが 5 以上のときは

$$\left({}_{n_c}C_k\cdot 2^{2^k}\right)^{n_c}\geq 2^{n\cdot 2^n} \qquad (7)$$

となる．

n と kを決めたとき，n_cの下界を条件式 (3)($k=1,2,4$ のとき) 及び条件式 (7)($k=8$ のとき) より求めたものを図 1に示す．この図から n_cは指数関数的に増加することがわかる．

4.2 強局所計算可能性

式 (6) より n が 1 から 7 までの強局所計算可能な関数の数を求めたものを表 4.1に示す．

任意の集合 $S(|S|=2^n)$ とその上で定義去れた任意の単項演算に対し 2−強局所計算可能とする符号化が存在する．その符号長は $O(|S|^2\log|S|)$ となる．そこで $k=2$ のときはこの符号長を n_cの上界とすると

$$|S|^2\log|S| = (2^n)^2\log 2^n = n\cdot 2^{2n}$$

であるから条件式 (5) より，$k=2$ のときの n_cの範囲は

$$n\cdot 2^{2n}\geq n_c\geq n\cdot 2^{n-2} \qquad (8)$$

となる．式 (8) より，n_cの上界は下界の 2^{n+2}倍となるので n が増すと指数関数的に範囲が広がることがわかる．

表 1: $|\widetilde{Fp^n}|, |Fp^{n,k}|, |F^{n,k}_{weak}|, |F^{n,k}_{strong}|$ 一覧表

n	1	2	3	4
$\|\widetilde{Fp^n}\|$	2	10	218	6.46×10^4
k	1	1 2	1 2 3	1 2 3 4
$\|Fp^{n,k}\|$	4	6 16	8 38 256	10 70 942 6.55×10^4
$\|F^{n,k}_{weak}\|$	4	36 256	512 5.49×10^4 1.68×10^7	1.00×10^4 2.40×10^7 7.87×10^{11} 1.84×10^{19}
$\|F^{n,k}_{strong}\|$	4	16 256	64 4.10×10^3 1.68×10^7	256 6.55×10^4 4.29×10^9 1.84×10^{19}

n	5					6					
$\|\widetilde{Fp^n}\|$	4.29×10^9					1.84×10^{19}					
k	1	2	3	4	5	1	2	3	4	5	6
$\|Fp^{n,k}\|$	12	112	2.29×10^3	3.25×10^5	4.29×10^9	14	164	4.52×10^3	9.73×10^5	2.58×10^{10}	1.84×10^{19}
$\|F^{n,k}_{weak}\|$	2.49×10^5	1.76×10^{10}	6.33×10^{16}	3.64×10^{27}	1.46×10^{48}	7.53×10^6	1.94×10^{13}	8.57×10^{21}	8.51×10^{35}	2.93×10^{62}	3.94×10^{115}
$\|F^{n,k}_{strong}\|$	1.02×10^3	1.05×10^6	1.10×10^{12}	1.21×10^{24}	1.46×10^{48}	4.10×10^3	1.68×10^7	2.81×10^{14}	7.92×10^{28}	6.28×10^{57}	3.94×10^{115}

n	7						
$\|\widetilde{Fp^n}\|$	3.40×10^{38}						
k	1	2	3	4	5	6	7
$\|Fp^{n,k}\|$	16	226	7.86×10^3	2.27×10^6	9.02×10^{10}	1.29×10^{20}	3.40×10^{38}
$\|F^{n,k}_{weak}\|$	2.68×10^8	3.01×10^{16}	1.85×10^{27}	3.09×10^{44}	4.85×10^{76}	5.99×10^{140}	5.28×10^{269}
$\|F^{n,k}_{strong}\|$	1.64×10^4	2.68×10^8	7.21×10^{16}	5.19×10^{33}	2.70×10^{67}	7.27×10^{134}	5.28×10^{269}

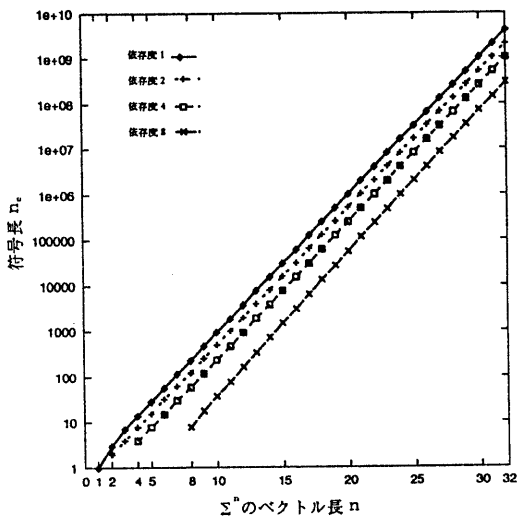

図 1: 弱局所計算可能における依存度と符号長の関係

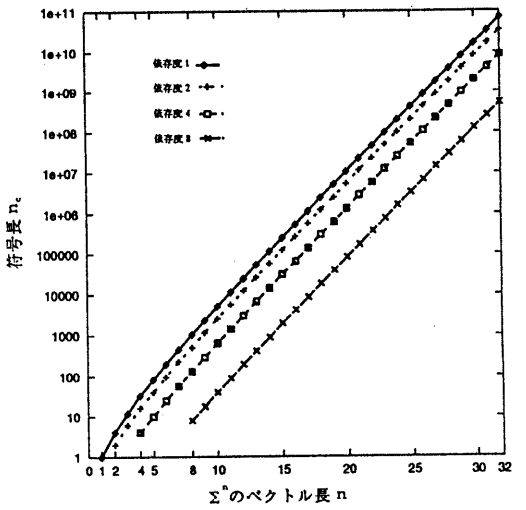

図 2: 強局所計算可能における依存度と符号長の関係

n と kを決めたとき, n_cの下界を条件式 (5) より求めたものを図 2に示す. この図から弱局所計算可能な場合と同様に, n_cは指数関数的に増加することがわかる.

5 おわりに

全ての n 入力 n 出力論理関数をある単一の符号化により $k-$ 局所計算可能な論理関数として実現する場合に必要な符号長の下界を関数の数を調べることより求めた. 本論文では問題を簡単にするために全ての論理関数を $k-$局所計算可能とする万能な符号化に限定して議論した. 今後, 与えられた論理関数集合に対する符号長の上下限及び演算の性質を利用した符号化法について検討していきたい.

謝辞

日頃ご討論頂く九州大学 大学院総合理工学研究科 村上和彰講師, ならびに. 安浦研究室の諸氏に感謝致します.

本研究は, 一部文部省科学研究費補助金重点領域研究「超並列アルゴリズム設計のためのデータ構造と計算モデルに関する研究」による.

参考文献

[1] Avizienis,A. : Signed-Digit Number Representation for Fast Parallel Arithmetic, IEEE Trans.Elec.Comput, Vol.EC-10,No.3,pp.389-400,1961.

[2] 安浦寛人 : 単項演算に対する局所計算可能な符号化, 情報処理学会誌, Vol.31,No.5,pp.740-747,May 1990.

[3] 安浦寛人, 高木直史, 矢島脩三 : 冗長符号化を利用した高速アルゴリズムについて, 電子情報通信学会論文誌, Vol.1.J70-D,No.3,pp.525-533,May 1987.

[4] 安浦寛人 : 計算機システムの基本演算のためのハードウェアアルゴリズム, コンピュータ, vol.3,No.1,pp.23-34,July 1984.

[5] 室賀三郎, 茨木俊秀, 北橋忠宏 : しきい論理, 産業図書, June 1976.

[6] 室賀三郎, 笹尾勤 : bit 別冊 論理設計とスイッチング理論 —LSI,VLSI の基礎—, 共立出版, July 1981.