

単項演算に対する局所計算可能な符号化

京大工学部 安浦 寛人 (Hiroto Yasuura)

1. はじめに

並列アルゴリズムの設計においては、いかに計算の並列性を抽出し、並列に計算できる部分を独立に並列計算するかが重要な問題である。さらに、各々の独立した計算が局所的な情報のみ依存するようにできれば、個々の独立した計算は単純なものとなり、計算全体における通信量も小さくなる。このように、並列計算を局所的な通信のみを必要とする単純な計算の集まりとすることができれば、通信や計算に要する時間や資源を小さくすることができる。

このようなアイデアに基づき、(1)では、局所計算可能性(Local Computability)という概念を提案し、それを実現するために符号化の工夫が有効であることを示した。集合 S とその上での演算 f が与えられたとき、ある符号化 C のもとで、 f による演算結果の各桁が、オペランドの高々 ℓ 桁にのみ依存するような計算規則が作れるとき、演算 f は符号化 C のもとで ℓ -局所計算可能であるという。(S, f) が有限可換群であれば、 S の要素数にかかわらず $\ell = 28$ となる 2 値の等長符号化が存在する⁽¹⁾。この符号化は S の各要素に 2 つ以上の符号を割り当てる冗長符号化であり、この冗長性が本質的に、計算の局所化に寄与してい

る。

一般的に、与えられた代数系に対し、局所計算可能性を実現する符号化を見つける問題は、並列アルゴリズムの設計の基本問題の一つである。上記の結果は、有限可換群とその上の2項演算だけが対象であったが、より広い枠組みの上での議論が必要である。ここでは、最も単純な場合の一つとして、有限集合とその上で定義される複数の単項演算を対象として、局所計算可能性を実現する符号化について議論する。

2. 諸定義

S を有限集合とする。 S の要素数を n とする。 $\{OP_1, OP_2, \dots, OP_m\}$ を S 上で定義された単項演算とする。すなわち、 $OP_i: S \rightarrow S$ である。 Σ を有限のアルファベットとし、その要素数を a とする。 S に対する Σ 上の符号化 \mathcal{C} は、 Σ^* の部分集合から S への全射（上への写像）として定義する。 ℓ を符号長と呼ぶ。ここでは、等長符号のみを考える。

$c \in \Sigma^*$, $s \in S$ について $\mathcal{C}(c) = s$ であるとき、 c を s に対する符号という。 Σ^* の要素 c で $\mathcal{C}(c)$ が定義されていない c を非符号語と呼ぶ。 $\ell =$

$\lceil \log_a n \rceil$ のとき、 \mathcal{C} を最短符号化と呼ぶ。また、ある s に対応する符号が2つ以上あるとき、 \mathcal{C} は冗長であるという。

$f: \Sigma^* \rightarrow \Sigma^*$ に対し、 $\mathcal{C}(f(c)) = OP(\mathcal{C}(c))$ が成り立つとき、 f は符号化 \mathcal{C} の下で S 上の単項演算 OP を実現するという。 f は

$$(z_1, z_2, \dots, z_k) = f(x_1, x_2, \dots, x_k)$$

$$= (g_1(x_1, x_2, \dots, x_k), g_2(x_1, x_2, \dots, x_k), \dots, g_k(x_1, x_2, \dots, x_k))$$

と書ける。すなわち、

$$z_j = g_j(x_1, x_2, \dots, x_k)$$

とかける。各 $g_j (j=1, 2, \dots, k)$ が高々 l 個の入力変数にしか依存しないとき、 f は l 一局所計算可能 (l -locally computable) であるという。S 上の単項演算 OP が符号化 σ の下で l 一局所計算可能であるとは、 OP を実現する l 一局所計算可能な関数 f が存在することである。

[定義 1] S 上の単項演算の集合 $\{OP_1, OP_2, \dots, OP_m\}$ が符号化 σ の下で l 弱一局所計算可能 (weakly l -locally computable) であるとは、各演算 OP_i が l 一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 $\{OP_1, OP_2, \dots, OP_m\}$ が符号化 σ の下で l 弱一局所計算可能であるとき、各演算 OP_i を実現する関数 $f_i (i=1, 2, \dots, m)$ において、各 $z_j (j=1, 2, \dots, k)$ が依存する l 個の入力変数を共通にすることができるとき、単項演算の集合は符号化 σ の下で l 強一局所計算可能 (strongly l -locally computable) であるという。

以下の議論は簡単のため 2 値符号化 ($\Sigma = \{0, 1\}$) を仮定するが、一般性は失われない。

3. 非冗長な最短符号

$k = \lceil \log_2 n \rceil$ かつ非冗長符号で、最も一般的に用いられている符号化について考える。強い制限のため、局所計算可能性は一般には小さくできない。

[定理 1] 非冗長な最短符号化のもとでは、弱局所計算可能性に関して、 $l < \ell$ ($= \lceil \log_2 n \rceil$) とはできない単項演算の集合が存在する。

(証明) $n = 2^p$ (p は正整数) とする。図 1 のようにある演算 OP で s_0 は s_0 、その他の状態は s_1 であるとする。 $l < \ell$ と仮定する。 s_0 に割り当てられる符号を $(0, 0, \dots, 0)$ としても一般性を失わない。 s_1 に割り当てられる符号を (a_1, a_2, \dots, a_k) とする。このベクトル中で i 番目の要素が 0 でなかったとする。ここで、仮定より x_i が依存しない変数が少なくとも 1 つ存在するから、それを x_j とする。符号の中で j 番目の要素だけが 1 であり、他のすべての要素が 0 であるものを考える。この符号を割り当てられた S の要素を s_j とすると、状態変数 x_i は x_j に依存しないから、演算 OP の下で s_j と s_0 の次状態の x_i の値は一致しなければならない。しかし、 $a_i = 1$ と選んだので矛盾である。□

4. 非冗長符号

最短符号の制限をはずすと符号化の自由度は大きく増大する。実用的にも、1 ホット符号等は広く用いられている。

要素	単項演算	符号語
	.. OP...	$x_0 x_1 \dots x_i \dots x_j \dots x_{k-1}$
s_0	s_0	0 0 ... 0 ... 0 ... 0
s_1	s_1	$a_1 a_2 \dots 1 \dots a_j \dots a_{k-1}$
s_2	s_1	
s_j	s_1	0 0 ... 0 ... 1 ... 0
s_{n-1}	s_1	

図 1 . 定理 1 の証明

[定理 2] 複数の単項演算が定義された任意の集合 S に対し, 1-弱局所計算可能となる符号化が構成できる.

(証明) 非冗長符号化のもとでは, 符号の各桁 x_i は, S の上の 1 つの 2 ブロック分割に対応している. 任意の 2 ブロック分割 $\{S_1, S_2\}$ について, 各演算における遷移に対応して, S_3 の各要素は S_1 中の要素に遷移し, S_4 中の要素は S_2 へ遷移するような 2 ブロック分割 $\{S_3, S_4\}$ が存在する. すべての 2 ブロック分割に対応するように各桁を選んだ符号長 2^n の非冗長符号は, 任意の単項演算の集合を 1-弱局所計算可能とする. 実用的には, 与えられた単項演算の集合に対して 2^n より小さな必要

最小限の符号長を求めることができる。□

[定理3] m 個の単項演算が定義された任意の集合 S に対し, $\min(m, \lceil \log_2 n \rceil)$ 一強局所計算可能となる符号化が構成できる。但し n は S の要素数である。

(証明) 定理2の証明と同様の議論で、一般に、

$$z_i = g_j(x_1, x_2, \dots, x_k) \\ = OP_1 x_{i1}^* + OP_2 x_{i2}^* + \dots + OP_m x_{im}^* \quad (1)$$

(x^* は x の肯定または否定を表す)となるように符号化を決めることができる⁽²⁾。
 $m < \lceil \log_2 n \rceil$ の時は(1)式を満たす符号化、それ以外は任意の最短符号化でよい。

□

定理3で与えられる Q の最小値の上界が上限となるような集合がある。

[定理4] 非冗長符号化のもとでは、強局所計算可能性 Q を $\min(m, \lceil \log_2 n \rceil)$ より小さくできない集合 S が存在する。

(証明) $\lceil \log_2 n \rceil < m$ の場合は、次のような演算が定義された集合を考える。但し、 $n = 2^p$ (p は正整数)とする。 S の要素 s_j ($j=0, 1, \dots, n-1$) の演算 OP_i ($i=0, 1, \dots, \lceil \log_2 n \rceil - 1$) における遷移先は、 j の $\lceil \log_2 n \rceil$ ビットの2進数表現の 2^i の桁が0のとき s_0 、1のとき s_1 である。状態 s_0 と s_1 を分離する符号語の桁 z は少なくとも1個存在する。このとき、 $s \neq s'$ なる任意の S の要素について、演算 OP_i ($i=0, 1, \dots, \lceil \log_2 n \rceil - 1$) のいずれかで、これらの遷移先は異なるので、 z はすべての要素を

分離するのに必要な少なくとも「 $\log_2 n$ 」個の入力変数に依存する（図2参照）。次に「 $\log_2 n \geq m$ 」の場合を考える。同様に n 要素の集合 S' （ $n=2^p$ p は正整数）を考える。 S' の状態 s_j （ $j=0,1,\dots,n-1$ ）の入力 OP_i （ $i=0,1,\dots,m-1$ ）における遷移先は、 j の「 $\log_2 n$ 」ビットの2進数表現の2^位の桁が0のとき s_0 、1のとき s_1 である。要素 s_0 と s_1 を分離する桁 z は少なくとも1個存在する。 S' の要素は 2^m 個の部分集合に分かれ、異なる集合に属する状態においては、ある適当な演算によって遷移先が異なる。すなわち、 z は 2^m 個の部分集合を分離するのに必要な m 個以上の変数に依存する。□

要素	単項演算	符号語
	$OP_0 OP_1 OP_2 \dots OP_{p-1} \dots OP_m$	$\dots z \dots$
s_0	$s_0 s_0 s_0 \dots s_0$	0
s_1	$s_1 s_0 s_0 \dots s_0$	1
s_2	$s_0 s_1 s_0 \dots s_0$	
s_3	$s_1 s_1 s_0 \dots s_0$	
s_n	$s_1 s_1 s_1 \dots s_1$	

図2. 定理4の証明

5. 冗長符号

冗長符号化は、1つの要素に複数の符号を割り当てることを許すもので、符号化の自由度は飛躍的に増大する。この自由度を利用して局所計算可能性を小さくできる。複数の単項演算が定義された任意の集合 S に対し、2-強局所計算可能とするような符号化が構成できる。この結果から、順序機械のパイプライン化に関する一般的な結果が導ける。

[定理5] 複数の単項演算が定義された任意の有限集合 S に対し、その演算の集合を2-強局所計算可能とするような冗長符号化が構成できる。

(証明) 符号化には、並列プレフィックス計算 (PPC) を利用する⁽³⁾⁽⁴⁾。以下に、単項演算を実現する順序回路の構成法を示す。

1) 集合 S に対し、 S から S の中への自己写像全体の集合 M を考える。 S の単項演算は、 M の中の写像の1つに対応する。自己写像の合成を2つの写像に対する2項演算と見ると、 M はこの演算によって、モノイドをなす。 M の要素は n^n 個であるから、その符号化には $n \log_2 n$ ビットあればよい。モノイドの演算を計算する2入力論理素子のみで構成した組合せ回路 (以下 I 回路とよぶ) の段数を D とすると、 $D = O(n \log_2 n)$ とできる。

2) 次のような組合せ回路 C を考える。まず、単項演算 OP_i から対応する M の要素に変換する回路 (I_p 回路) を D 個用意する。これらを葉として、高さ $\lceil \log_2 D \rceil$ の I 回路からなる2分木を作る。根の出力をさらに別の I 回路 (最終 I 回路という) の入力とし、その出力は自分自身のもうひとつの入力とする。 C は適当にダミー

の素子を入れて，入力から，最終 I 回路の出力までのすべての経路の段数を一致させておく（同期化）。

3) 回路 C を構成するすべての論理素子の出力に記憶素子を挿入する（パイプライン化）。 C 中のすべての記憶素子は高々 2 つの他の記憶素子の出力にしか依存していない。

4) 入力された演算 OP_i を記憶する長さ D のシフトレジスタを用意する。また，長さ D の 1 ビットの巡回シフトレジスタを用意し，その中の一つのビットを基準ビットとする。巡回シフトレジスタには，1ヶ所のみ 1 をセットしておく。基準ビットが 1 のとき，入力シフトレジスタの内容が一斉に I_p 回路の入力に送られるように回路を組む（パイプライン入力の同期）。

このようにして組んだ順序回路のすべての記憶素子（ C 中のものとシフトレジスタ中のもの）を符号の各桁に対応させる。各記憶素子（桁）は高々 2 個の記憶素子（桁）にしか依存しないので，2 一局所計算可能となっている。また，この回路が集合 S 上の任意の単項演算の集合を計算することは容易に示される。□

定理 5 の証明の構成法では，最悪の場合 n の指数オーダー以上の符号長となる。現実的には，与えられた単項演算の集合から必要となる最小限の自己写像だけを対象として，より短い符号長で実現できる。この定理から，次の系が導かれる。

[系] 任意の順序機械は，入次数制限のある論理素子モデルのもとで，入力周期を素子 1 段分と記憶素子の遅延時間程度になるように 1 相のクロックのもとでパ

イプライン化できる⁽²⁾。

6. 応用と課題

今回は符号長についてはあまり議論しなかったが、明らかに符号長と局所計算可能性の間にはある種のトレードオフが存在すると考えられる。今後、このトレードオフについても考察を行ないたい。また、強局所計算可能性は、2項演算の弱局所計算可能性と密接な関係がある。この点についても考察していく。

謝辞 日頃御指導頂く本学田丸啓吉教授に感謝いたします。また、御討論頂いた京都産業大学岩間一雄助教授ならびに本学の矢島脩三教授はじめ諸氏に深謝いたします。本研究は一部文部省科学研究補助金62750321「局所計算可能な符号を用いた並列アルゴリズムの研究」による。

参考文献

- (1)安浦, 高木, 矢島, "冗長符号化を利用した高速並列アルゴリズムについて", 信学論 vol.J70-D, no.3, March 1987.
- (2)安浦寛人, "順序回路の局所計算可能な状態符号化について", 電子情報通信学会コンピューテーション研究会資料, COMP87-51, Nov. 1987.
- (3)Unger.S.H., "Tree Realization of Iterative Circuits," IEEE Trans. on Comput., vol.C-26, no.4, pp.365-383, April 1977.
- (4)Ladner, R.E. and Fischer, M.J., "Parallel Prefix Computation," J. of ACM. vol 27, no.4, pp.831-838, Oct. 1980.