2008－AL－117 (13)
2008／3／7

膜計算における基本演算アルゴリズム

立石 竹志　藤原 暁宏
九州工業大学 情報工学部 電子情報工学科

概要： 近年，新しい計算パラダイムの1つである膜計算が注目を集めている．本研究では，この膜計算において，論理演算，加算，及び，ソートを実行するアルゴリズムを提案する．まず，最初に，膜計算において論理演算を実行するアルゴリズムを2つ示す．1つ目のアルゴリズムは，2入力1出力の論理演算を $O(1)$ 種類のオブジェクトを用いて $O(1)$ ステップ実行するアルゴリズムであり，2つ目のアルゴリズムは，n 入力の論理演算を $O(n)$ 種類のオブジェクトを用いて $O(1)$ ステップで実行するアルゴリズムである．次に，膜計算において加算を実行するアルゴリズムを示す．このアルゴリズムは，2つの m ビットの2進数の加算を $O(m)$ 種類のオブジェクトを用いて $O(1)$ ステップで実行可能である．最後に，膜計算においてソートを行うアルゴリズムを示す．まず，2つの m ビットの2進数の比較交換操作を $O(m)$ 種類のオブジェクトを用いて $O(1)$ ステップで実行するアルゴリズムを示す．次に，そのアルゴリズムを用いて，奇偶転換ソートのアイデアに基づき，n 個の m ビットの2進数のソートを $O(mn)$ 種類のオブジェクトを用いて $O(n)$ ステップで実行するアルゴリズムを示す．

Procedures for basic operations in membrane computing

Takeshi Tateishi, Akihiro Fujiwara
Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology

Abstract: Recently, membrane computing has considerable attention as one of new paradigms of computations. In this paper, we propose three procedures, which are for logic operations, addition and sorting, in membrane computing. We first propose two procedures for logic operations in the membrane computing. The first procedure computes logic operations that has two inputs and one output in $O(1)$ steps using $O(1)$ objects, and the second procedure computes AND, OR, and EXOR for n boolean inputs, and works in $O(1)$ steps using $O(n)$ objects. We next propose a procedure for an addition in membrane computing. The procedure computes the addition of two binary numbers of m bits in $O(1)$ steps using (m) objects. We finally we propose a procedure for sorting in membrane computing. We propose a sub-procedure for a compare-and-exchange operation of two bianary numbers of m bits. The sub-procedure works in $O(1)$ steps using $O(m)$ objects. Then, we propose a procedure for sorting using the sub-procedure. The procedure is based on an idea of the odd-even transposition sort, and executes sorting of n binary numbers of m bits in $O(n)$ steps using $O(mn)$ objects.

1 はじめに

近年，新しい計算パラダイムの1つである膜計算 (P システム) が注目を集めている．膜計算は，Păun[4] によって提案された，生物の細胞活動を計算に用いる計算モデルである．生物の細胞内では，それぞれの膜で区切られた細胞内の要素 (核，細胞膜，液胞など) は独自に生命活動を行っており，膜計算では，この生命活動を並列計算と考えて計算モデルとしている．

膜計算では，細胞内の要素をオブジェクトと呼び，各オブジェクト，及び，膜に対して進化規則を適用し，オブジェクト，及び，膜を進化させることによって計算を行う．この計算モデルは，データ量が増加すると指数的に計算時間が増大するような問題に威力を発揮し，従来の計算機では指数的な時間を必要とする NP 完全問題を多項式ステップで解くことが可能となる．このモデルの提案以降，様々な NP 完全問題を解くアルゴリズム [2, 3, 5, 6] が提案されている．

一方，膜計算を NP 完全問題だけではなく，より広い問題の解法に利用するためには，論理演算，及び，算術演算のような通常の計算機が備えている基本的な演算を実行するアルゴリズムも必要である．このような基本演算に関するアルゴリズムは，文献 [2] において，オブジェクトを用いた2進数表記法，及び，多入力の論理和と加算を実行するアルゴリズムが提案されている．前者のアルゴリズムは，n 個の論理値に対して論理和を求めるものであり，2分木演算のアイデアを用いて $O(\log n)$ ステップで実行可能となっている．後者のアルゴリズムは，2つの m ビットの2進数に対して加算を行うものであり，各ビット毎の論理演算を用いて計算を行うことにより，$O(m)$ ステップで実行可能となっている．

本研究では，この膜計算において，より効率の良い論理演算，加算，及び，ソートを実行するアルゴリズムの提案を行う．

まず最初に，膜計算においてオブジェクトを用いて2進数を表すデータ構造を提案する．提案するデータ構造は，文献 [2] の表記法を拡張したものであり，2進数をアドレスとビット毎に1つのオブジェクトを用いて表現するものである．このデータ構造を用いることにより，膜計算において n 個の m ビットの2進数を $O(mn)$ 種類のオブジェクトにより表現することができる．

次に，上記のデータ構造を用いて，論理演算を実行するアルゴリズムを2つ示す．1つ目のアルゴリズムは，入力に対して進化規則を論理演算の解にな

るように設定することにより2入力1出力の論理演算を計算するアルゴリズムであり，$O(1)$ 種類のオブジェクトを用いて $O(1)$ ステップで実行可能である．2つ目のアルゴリズムは，多入力の論理演算を実行するアルゴリズムであり，論理値に注目した進化規則を設定することにより，n 入力1出力の論理積，論理和，及び，排他的論理和を計算する．このアルゴリズムは，$O(n)$ 種類のオブジェクトを用いて $O(1)$ ステップで実行可能である．

また，膜計算をおいて加算を実行するアルゴリズムを示す．このアルゴリズムは，それぞれのビット毎に半加算器，桁上がり，及び，桁上がりの伝播を計算する進化規則により構成されており，2つの m ビットの2進数の加算を $O(m)$ 種類のオブジェクトを用いて $O(1)$ ステップで実行可能である．

最後に，膜計算においてソートを行うアルゴリズムを示す．このアルゴリズム実現のために，まず，2つの値の比較交換操作を行うアルゴリズムを示す．このアルゴリズムは，2つの値の減算を実行し，その解を元に値の交換を行うか否かを判断するという進化規則で構成されており，2つの m ビットの2進数の比較交換操作を $O(m)$ 種類のオブジェクトを用いて $O(1)$ ステップで実行可能である．最後に，提案した比較交換操作を用いて，ソートを行うアルゴリズムを提案する．このアルゴリズムは，奇偶転換ソートのアイデアに基づくものであり，入力された n 個の m ビットの2進数をそれぞれ偶数ステップ，奇数ステップで比較交換操作を n 回並列に実行するという進化規則で構成されており，n 個の m ビットの2進数のソートを $O(mn)$ 種類のオブジェクトを用いて $O(n)$ ステップで実行可能である．

2 膜計算モデル

2.1 膜構造

本節では，膜計算の基本要素である膜構造について簡単に説明する．膜計算において用いられる膜構造は，生物の細胞等をモデル化したものである．

ここで，膜計算で用いられる膜構造の例を図1に示す．この膜構造は階層的にとらえることができ，最も外側の膜 (それ以上外側に膜がない膜) をスキン膜といい，逆に，最も内側の膜 (それ以上内側に膜がない膜) を初期膜という．また，それぞれの膜によって仕切られた空間を膜空間といい，スキン膜の外の空間を環境という．また，それぞれの膜は整数のラベルにより識別される．本研究では，整数 j によりラベル付けされた膜を，膜 j と表記する．

次に，膜構造の表記法について説明する．膜構造の表記法は膜に対応したラベルを添字とする左括弧と右括弧の組を用いて表記する．例えば，図1の膜構造は，以下のように表される．

$$[_1\,[_2\,]_2\,[_3\,[_4\,]_4\,[_5\,]_5\,]_3\,]_1$$

また，各膜は，電気的極性 $e \in \{-, 0, +\}$ を持つ．例えば，図1の膜5が $+$ という極性を持つ場合以下のように記述する．

$$[_1\,[_2\,]_2\,[_3\,[_4\,]_4\,[_5\,]_5^+\,]_3\,]_1$$

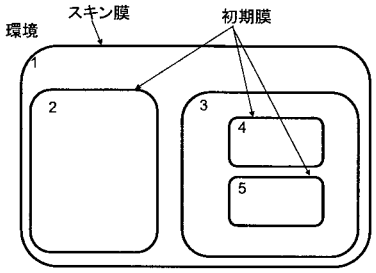

図 1: 膜構造

なお，一般的に，電気的極性 $e = 0$ である場合は，極性の表記を省略する．

次に，オブジェクトについて説明する．各膜はオブジェクトと呼ばれる要素を含む．例えば，図1の膜5の中にオブジェクト a が含まれるとすると，膜構造は以下のように表記される．

$$[_1\,[_2\,]_2\,[_3\,[_4\,]_4\,[_5\,a\,]_5\,]_3\,]_1$$

また，同じ膜内にオブジェクト a, b, c が存在すると，abc のように文字列として表記し，同じオブジェクトが複数存在する場合は，a^n と表記することにより，オブジェクト a が n 個存在することを表す．

最後に，膜計算の重要な要素である進化規則について説明する．進化規則とは，オブジェクトが進化するための規則であり，各膜において進化規則に従ってオブジェクトが進化していくことにより計算を行う．例えば，$a \to b$ という進化規則があった場合，これはオブジェクト a が次ステップでオブジェクト b に進化することを表している．この進化規則については次節で詳しく説明する．

2.2 計算モデルと進化規則

本稿では，2.1節で説明した膜の性質を厳密に定義した計算モデルとして，文献 [2] で提案されている P システムを用いる．1つの P システム Π は以下のように定義される．

$$\Pi = (O, \mu, \omega_1, \omega_2, \cdots, \omega_m, R_1, R_2, \cdots, R_m, i_{in}, i_{out})$$

ここで，各要素の定義は以下の通りである．

O: 使用する全てのオブジェクトの集合

ω_j: 膜 $j \in H$ の中に初期状態で存在するオブジェクトの集合

R_j: 膜 $j \in H$ に対して適用される規則の集合

i_{in}: 入力膜のラベル

i_{out}: 出力膜のラベル

ここで，集合 H は使用するすべての膜のラベルの集合である．

本稿では，進化規則としては，文献 [2] に基づき，オブジェクト進化規則，内部通信規則，外部通信規則，膜分解規則，及び，膜分割規則という 5 種類の進化規則を用いる．以下にこの 5 つの進化規則の定義を示す．

オブジェクト進化規則
オブジェクト進化規則は以下のように定義される．

$$[_h\, a\,]_h^{e_1} \rightarrow [_h\, b\,]_h^{e_2}$$

ただし，$h \in H, e_1, e_2 \in \{+,-,0\}, a,b \in O$ である．

オブジェクト進化規則は，最も基本的な規則であり，各膜に存在するオブジェクトを別のオブジェクトに進化 (変化) させる規則である．また，オブジェクト進化規則は，オブジェクトだけが進化する場合は，膜のラベルと電気的極性を省略する．例えば，

$$[_h\, a\,]_h^0 \rightarrow [_h\, b\,]_h^0$$

という規則の場合は，

$$a \rightarrow b$$

と省略して記述する．

内部通信規則
内部通信規則は以下のように定義される．

$$a[_h\,]_h^{e_1} \rightarrow [_h\, b\,]_h^{e_2}$$

ただし，$h \in H, e_1, e_2 \in \{+,-,0\}, a,b \in O$ である．

内部通信規則は，オブジェクトを進化させると同時に内側の膜に移動する規則である．

外部通信規則
外部通信規則は以下のように定義される．

$$[_h\, a\,]_h^{e_1} \rightarrow b[_h\,]_h^{e_2}$$

ただし，$h \in H, e_1, e_2 \in \{+,-,0\}, a,b \in O$ である．

外部通信規則は内部通信規則と逆の働きをする進化規則であり，オブジェクトを進化させると同時に外側の膜に移動する規則である．

膜分解規則
膜分解規則は以下のように定義される．

$$[_h\, a\,]_h^{e} \rightarrow b$$

ただし，$h \in H, e \in \{+,-,0\}, a,b \in O$ である．

膜分解規則は，オブジェクトを進化させると同時に，そのオブジェクトが存在する膜を分解 (消去) する規則である．膜が消去されることにより，その膜の中のオブジェクトは，1 つ外側の膜の中に存在することになる．なお，スキン膜を分解することはできないので，そのような膜分解規則を定義することはできない．

膜分割規則
膜分割規則は以下のように定義される．

$$[_h\, a\,]_h^{e_1} \rightarrow [_h\, b\,]_h^{e_2} [_h\, c\,]_h^{e_3}$$

ただし，$h \in H, e_1, e_2, e_3 \in \{+,-,0\}, a,b,c \in O$ である．

膜分割規則は，オブジェクトを進化させると同時に，そのオブジェクトが存在した膜を分割 (分裂) させる規則である．なお，スキン膜を分割することはできないので，そのような膜分割規則を定義することはできない．

さらに，膜計算には 2 つの大きな特徴がある．それは，最大並列則と非決定性である．最大並列則とは，ある時点において適用できるすべての規則が適用されるというものであり，これにより膜計算を並列計算とみなすことができる．また，非決定性とは，ある時点においてあるオブジェクトに適用できる規則は複数存在する場合は，適用される規則が非決定的に選択されるというものである．

本研究では，ある時点で，適用可能なすべての規則を適用し，次の状態に進化することを遷移と呼び，その遷移の計算量を $O(1)$ ステップと定義する．また，計算の終了は，適用できる規則がなく，それ以上進化が進まないときに計算が終了するものとする．

3 2 進数を表すデータ構造

以下では，膜計算におけるオブジェクトを用いた 2 進数を表すデータ構造について説明する．多くの膜計算モデルにおいては，値がオブジェクトの長さで表現されているが，文献 [2] においては，オブジェクトを用いた 2 進数表現が提案されている．本研究では，文献 [2] で提案された 2 進数表現を改良し，n 個の m ビット 2 進数を $O(mn)$ 個のオブジェクトを用いて 2 進数を表すデータ構造を提案する．以下にオブジェクトの詳細を示す．

2 進数の 1 ビットを表すオブジェクトは以下の様に定義される．

$$\langle A_i, B_j, V_{i,j} \rangle$$

ここで，$i(0 \leq i \leq n-1)$ は 2 進数の格納されるアドレスを表し，$j(0 \leq j \leq m-1)$ は 2 進数の中のビット番号を表す．さらに，$V_{i,j}(\in 0,1)$ は，表すビットの論理値を表している．

上記のオブジェクトを用いて，アドレス i に格納された 2 進数の値 V_i は，

$$\langle A_i, B_{m-1}, V_{i,m-1} \rangle, \langle A_i, B_{m-2}, V_{i,m-2} \rangle, \cdots, \langle A_i, B_0, V_{i,0} \rangle$$

という m 個のオブジェクトで表すことができる．ここで，

$$V_i = \sum_{j=0}^{m-1} V_{i,\,j} \times 2^j$$

である．例えば，アドレス 8 に格納された 2 進数 1011 は，以下の 4 つのオブジェクトにより表される．

$$\langle A_8, B_3, 1 \rangle, \langle A_8, B_2, 0 \rangle, \langle A_8, B_1, 1 \rangle, \langle A_8, B_0, 1 \rangle,$$

従って，n 個の m ビットの 2 進数は，$O(mn)$ 個のオブジェクトを用いて表すことができる．

4 論理演算アルゴリズム

本節では, 表1に示すような2入力1出力の論理演算を実行するアルゴリズムを示す. なお, $\alpha \in \{0,1\}$ は, 入力に対して, 適切な論理演算の解であるような論理値である.

表 1: 真理値表

V_{in1}	V_{in2}	V_{out1}
0	0	α_{00}
0	1	α_{01}
1	0	α_{10}
1	1	α_{11}

以下に論理演算を実行する P システム Π_{logic} を示す.

$$\Pi_{logic} = (O, \mu, \omega_1, R_1, i_{in}, i_{out})$$

ここで, Π_{logic} を構成するそれぞれの集合は以下ようになる.

- $O = \{\langle A_x, B_0, V_{x,0}\rangle, \langle A_y, B_0, V_{y,0}\rangle | V_{x,0}, V_{y,0} \in \{0,1\}\}$
- $\mu = [_1\,]_1$
- $\omega_1 = \{\langle A_x, B_0, V_{x,0}\rangle\langle A_y, B_0, V_{y,0}\rangle\}$
- $R_1 = \{\langle A_x, B_0, 0\rangle\langle A_y, B_0, 0\rangle \to \langle A_x, B_0, \alpha_{00}\rangle,$
  $\langle A_x, B_0, 0\rangle\langle A_y, B_0, 1\rangle \to \langle A_x, B_0, \alpha_{01}\rangle,$
  $\langle A_x, B_0, 1\rangle\langle A_y, B_0, 0\rangle \to \langle A_x, B_0, \alpha_{10}\rangle,$
  $\langle A_x, B_0, 1\rangle\langle A_y, B_0, 1\rangle \to \langle A_x, B_0, \alpha_{11}\rangle\}$
- $i_{in} = 1, i_{out} = 1$

この計算モデルでは入力膜, 及び, 出力膜ともに膜1である. また, 入力である論理値 x, 及び, y はそれぞれアドレス x の0ビット目, 及び, アドレス y の0ビット目に格納されており, 出力はアドレス x の0ビット目に上書きされる.

このアルゴリズムの入出力は以下の通りである.

入力 2 つの論理値を表すオブジェクト $\langle A_x, B_0, V_{x,0}\rangle$, $\langle A_y, B_0, V_{y,0}\rangle$

出力 論理演算の解を表すオブジェクト $\langle A_x, B_0, V_{x,0}\rangle$

論理演算を実行するアルゴリズムは, 入力された2つのオブジェクトに対して, 上記の R_1 に示す進化規則を適用することで実行される.

上記の, 論理演算を実行する P システム Π_{loggic} は, 定数回の進化規則の適用によりアルゴリズムが終了するので, 以下の定理が得られる.

定理 1 2つの論理値を表すオブジェクトを入力とし, 論理演算を計算する P システム Π_{logic} は, $O(1)$ 種類のオブジェクト, $O(1)$ 個の膜, 及び, $O(1)$ 個の進化規則を用いて, $O(1)$ ステップで実行可能である. □

また, n 個の論理値を表わすオブジェクトに対して $\frac{n}{2}$ 対の論理演算を実行するアルゴリズムは, 上記で説明した P システム Π_{logic} において, 使用するオブジェクト, 及び, 進化規則の数を $O(n)$ 種類準備することにより実行できる. よって, 以下の定理が得られる.

定理 2 n 個の論理値を表すオブジェクトを入力とする $\frac{n}{2}$ 対の論理演算は, P システムにおいて $O(n)$ 種類のオブジェクト, $O(1)$ 個の膜, 及び, $O(n)$ 個の進化規則を用いて, $O(1)$ ステップで実行可能である. □

5 多入力論理演算アルゴリズム

本節では, まず, 最初に, 多入力の論理積を実行する P システムを定義し, アルゴリズムの計算量を検証する. 次に, 多入力の排他的論理和を実行する P システムを定義し, そのアルゴリズムの計算量を検証する. なお, アルゴリズムの詳細な動作説明については, 紙面の関係上省略する.

5.1 多入力の論理積アルゴリズム

以下に, n 入力の論理積を実行する P システム Π_{n-AND} を示す.

$$\Pi_{n-AND} = (O, \mu, \omega_1, \omega_2, R_1, R_2, i_{in}, i_{out})$$

ここで, Π_{n-AND} を構成するそれぞれの集合は以下ようになる.

- $O = \{c(k), \langle A_i, B_0, V_{i,0}\rangle | 0 \le i \le n-1, 0 \le k \le 2, V_{i,0} \in \{0,1\}\}$
- $\mu = [_1\,[_2\,]_2]_1$
- $\omega_1 = \phi$
- $\omega_2 = \{\langle A_0, B_0, 0, V_{0,0}\rangle\langle A_1, B_0, 0, V_{1,0}\rangle \cdots \langle A_{n-1}, B_0, 0, V_{n-1,0}\rangle\}$
- $R_1 = \phi$
- $R_2 = \{c(0) \to c(1), c(1) \to c(2),$
  $[_2\langle A_0, B_0, 0\rangle]_2 \to [_2\langle A_0, B_0, 0\rangle]_2^+,$
  $[_2\langle A_1, B_0, 0\rangle]_2 \to [_2\langle A_0, B_0, 0\rangle]_2^+,$
  $\vdots$
  $[_2\langle A_{n-1}, B_0, 0\rangle]_2 \to [_2\langle A_0, B_0, 0\rangle]_2^+,$
  $[_2c(1)\langle A_0, B_0, 0\rangle]_2^+ \to \langle A_0, B_0, 0\rangle[_2\,]_2^+,$
  $[_2c_2]_2 \to \langle A_0, B_0, 1\rangle[_2\,]_2\}$

- $i_{in} = 2, i_{out} = 1$

ここで，入力膜は膜2であり，出力膜は膜1である．また，入力される n 個の論理値は，それぞれアドレス0から $n-1$ の0ビット目に格納されており，出力は，アドレス0の0ビット目に上書きされる．また，オブジェクト $c(0), c(1), c(2)$ はカウンタの役割を働きをする．

以下に n 個の論理値の論理積を計算するアルゴリズムの概要を示す．

入力 n 個の論理値を表す以下の n 個のオブジェクト
$\langle A_0, B_0, 0, V_{0,0}\rangle\langle A_1, B_0, 0, V_{1,0}\rangle \cdots \langle A_{n-1}, B_0, 0, V_{n-1,0}\rangle$

出力 入力されたオブジェクトに論理値が0を表すオブジェクトが存在すれば値0を表わすオブジェクト $\langle A_0, B_0, 0\rangle$ を出力し，そうでなければ値1を表わすオブジェクト $\langle A_0, B_0, 1\rangle$ を出力

Step 1 入力されたオブジェクトに値が0を表すオブジェクトが存在するかどうかを調べる．

Step 2 Step 1 で0が存在すれば0を出力し，そうでなければ1を出力する．

上記の，n 個の論理値の論理積を計算する P システム Π_{n-AND} は，定数回の進化規則の適用によりアルゴリズムが終了するので，以下のような定理が得られる．

定理 3 n 個の論理値を表すオブジェクトを入力とし，論理積を計算する P システム Π_{n-AND} は，$O(n)$ 種類のオブジェクト，$O(1)$ 個の膜，及び，$O(n)$ 個の進化規則を用いて，$O(1)$ ステップで実行可能である． □

また，n 入力の論理和を実行するアルゴリズムも上記と同様の方法で実現可能であり，以下の定理が得られる．

定理 4 n 個の論理値を表すオブジェクトを入力とし，論理和を計算する P システム Π_{n-OR} は，$O(n)$ 種類のオブジェクト，$O(1)$ 個の膜，及び，$O(n)$ 個の進化規則を用いて，$O(1)$ ステップで実行可能である． □

5.2 多入力の排他的論理和アルゴリズム

以下に，n 個の論理値の排他的論理和を計算する P システム Π_{n-XOR} を示す．

$$\Pi_{n-XOR} = (O, \mu, \omega_1, \omega_2, \omega_3, R_1, R_2, R_3, i_{in}, i_{out})$$

ここで，Π_{n-XOR} を構成するそれぞれの集合は以下ようになる．

- $O = \{c_1(k), c_2(h), \langle A_i, B_0, V_{i,0}\rangle, \alpha, \beta, \gamma \mid 0 \le k \le 3, 0 \le h \le 4, 0 \le i \le n-1, V_{0,1} \in \{0,1\}\}$
- $\mu = [_1\,[_2\,[_3\,]_3\,]_2\,]_1$
- $\omega_1 = \phi$
- $\omega_2 = \{c_1(0)\}$
- $\omega_3 = \{\langle A_0, B, 0, V_{0,0}\rangle\langle A_1, B, 0, V_{1,0}\rangle \cdots \langle A_{n-1}, B, 0, V_{n-1,0}\rangle\}$
- $R_1 = \phi$
- $R_2 = \{c_1(0) \to c_1(1), c_1(1) \to c_1(2), c_1(2) \to c_1(3), c_0(3) \to \gamma, [_2\alpha \to \langle A_0, B_0, 1\rangle[_2]_2^+, [_2\,\gamma\,]_2 \to \langle A_0, B_0, 0\rangle[_2\,]_2\}$
- $R_3 = \{\langle A_0, B_0, 1\rangle \to \alpha, \langle A_1, B_0, 1\rangle \to \alpha, \vdots, \langle A_{n-1}, B_0, 1\rangle \to \alpha, \alpha\alpha \to \beta, c_2(0) \to c_2(1), c_2(1) \to c_2(2), [_3\,[_2\,c_2(2)\,]_2\,]_3 \to [_2\,c_2(3)\,]_2\}$
- $i_{in} = 2, i_{out} = 1$

入力膜は膜3であり，出力膜は膜1である．また，入力はアドレス0から $n-1$ の0ビットにそれぞれ格納された n 個の論理値であり，出力はアドレス0の0ビット目に上書きされる．また，オブジェクト $c_1(k)$，及び，$c_1(h)$ はカウンタの役割をし，特定のステップ数で解を出力し，計算を終了する働きをする．

以下に n 個の論理値の排他的論理和を計算するアルゴリズムの概要を示す．

入力 n 個の論理値を表すオブジェクト
$\langle A_0, B_0, 0, V_{0,0}\rangle\langle A_1, B_0, 0, V_{1,0}\rangle \cdots \langle A_{n-1}, B_0, 0, V_{n-1,0}\rangle$

出力 入力されたオブジェクトに値が1を表すオブジェクトが偶数個存在すれば値0を表わすオブジェクト $\langle A_0, B_0, 0\rangle$ を出力し，奇数個ならば値1を表わすオブジェクト $\langle A_0, B_0, 1\rangle$ を出力

Step 1 以下のサブステップを実行することにより，入力されたオブジェクトに1を表すオブジェクトが偶数個存在するか，奇数個存在するかを調べる．

(1-1) 1を表すすべてのオブジェクトを，同じオブジェクト α に進化させる．

(1-2) オブジェクト α に対して，$\alpha\alpha \to \beta$ という進化規則を適用する．この時，オブジェクト α の個数が偶数個なら，オブジェクト α はすべてオブジェクト β に進化し，オブジェクト α の個数が奇数個なら，オブジェクト α が1つ残ることになる．

Step 2 Step 1 の結果により，以下のサブステップ (2-1) オブジェクト α が 1 つ残っている場合，(2-2) オブジェクト α が残ってない場合のいづれかの動作を行う．

(2-1) オブジェクト α が存在することにより，1 が奇数個存在することがわかるので，$\alpha \rightarrow$ (出力オブジェクト) という，進化規則を適用し，出力を行う．

(2-2) オブジェクト α が存在しないことにより，カウンタの役割をするオブジェクトが γ まで進化する．さらに，$\gamma \rightarrow$ (出力オブジェクト) という進化規則を適用し，出力を行う．

上記の，n 個の論理値の排他的論理和を計算する P システム Π_{n-XOR} は，定数回の進化規則の適用によりアルゴリズムが終了するので，以下の定理が得られる．

定理 5 n 個の論理値を表すオブジェクトを入力とし，排他的論理和を計算する P システム Π_{n-XOR} は，$O(n)$ 種類のオブジェクト，$O(1)$ 個の膜，及び，$O(n)$ 個の進化規則を用いて，$O(1)$ ステップで実行可能である． □

6 加算アルゴリズム

本節では、2 つの m ビットの 2 進数の加算を実行する P システム Π_{add} を示す．なお，アルゴリズムの詳細な動作説明については，紙面の関係上省略する．

P システム Π_{add} を定義する前に，排他的論理和を実行する P システム Π_{xor} を以下のように定義する．

$$\Pi_{xor} = (O_{xor}, \mu_{xor}, \omega_{xor}, R_{xor}, i_{xor-in}, i_{xor-out})$$

この P システム Π_{xor} は，2 つの m ビット進数を入力とし，それぞれのビット毎に排他的論理和の解を出力するものでる．

以下に，2 つの m ビットの 2 進数の加算を実行する P システム Π_{add} を示す．ただし，入力 2 進数の最上位ビットは 0 であるとする．

$$\Pi_{add} = (O, \mu, \omega_1, \omega_2, \omega_3, \omega_4, \omega_5, R_1, R_2, R_3, R_4, R_5, i_{in}, i_{out})$$

ここで，Π_{add} を構成するそれぞれの集合は以下ようになる．

- $O = \{O_{xor}, c_0(d), c_1(e), c_2(f), c_3(g), \langle A_i, B_j, V_{i,j}\rangle, D | 0 \le d \le 1, 0 \le e \le 1, 0 \le f \le 2, 0 \le g \le 3, i \in \{a, b, c, x, y, p, s\}, 0 \le j \le m-1\}$
- $\mu = [_1\, [_2\, [_3\, [_4\, [_5\,]_5\,]_4\,]_3\,]_2\,]_1$
- $\omega_1 = \phi$
- $\omega_2 = \phi$
- $\omega_3 = \{c_1(0)^m c_2(0)^m c_3(0)\langle A_c, B_0, 0\rangle \langle A_c, B_1, 0\rangle \cdots \langle A_c, B_{m-1}, 0\rangle\}$
- $\omega_4 = \{c_0(0)^{2m}\langle A_a, B_0, V_{0,0}\rangle\langle A_a, B_1, V_{0,1}\rangle \cdots \langle A_a, B_1, V_{0,m-1}\rangle \langle A_b, B_0, V_{1,0}\rangle\langle A_b, B_1, V_{1,1}\rangle \cdots \langle A_b, B_1, V_{1,m-1}\rangle\}$
- $\omega_5 = \{D\}$
- $R_1 = \phi$
- $R_2 = R_{xor}$
- $R_3 = \{[_3\langle A_x, B_j, V_{x,j}\rangle]_3 \rightarrow \langle A_x, B_j, V_{x,j}\rangle [_3\,]_3^+,$
  $[_3 c_1(0)\langle A_c, B_j, 0\rangle\langle A_c, B_{j-1}, 0\rangle \cdots \langle A_c, B_k, 0\rangle \langle A_p, B_j, 0\rangle\langle A_p, B_{j-1}, 1\rangle \cdots \langle A_y, B_k, 1\rangle]_3^+ \rightarrow [_3 c_1(1)\langle A_c, B_j, 1\rangle\langle A_c, B_{j-1}, 1\rangle \cdots \langle A_c, B_k, 0\rangle]_3^+,$
  $[_3 c_2(1)\langle A_y, B_{j-1}, 1\rangle\langle A_c, B_j, 0\rangle]_3^+ \rightarrow [_3 c_2(2)\langle A_c, B_j, 1\rangle]_3^+,$
  $[_3 c_2(0)]_3^+ \rightarrow [_3 c_2(1)]_3^+,$
  $[_3 c_3(0)]_3^+ \rightarrow [_3 c_3(1)]_3^+,$
  $[_3 c_3(1)]_3^+ \rightarrow [_3 c_3(2)]_3^+,$
  $[_3 c_3(2)]_3^+ \rightarrow [_3 c_3(3)]_3^+,$
  $[_3 c_3(3)]_3^+ \rightarrow [_3\,]_3^-,$
  $[_3\langle A_c, B_j, V_{c,j}\rangle]_3^- \rightarrow \langle A_c, B_j, V_{c,j}\rangle [_3]_3^-\}$
- $R_4 = \{[_4 c_0(0)\langle A_x, B_j, V_{x,j}\rangle]_4 \rightarrow [_4 c_0(1)\langle A_x, B_j, V_{x,j}\rangle]_4\langle A_x, B_j, V_{x,j}\rangle,$
  $[_4 c_0(0)\langle A_y, B_j, V_{y,j}\rangle]_4 \rightarrow [_4 c_0(1)\langle A_y, B_j, V_{y,j}\rangle]_4\langle A_y, B_j, V_{y,j}\rangle,$
  $[_4 c_0(1)\langle A_x, B_j, 0\rangle\langle A_y, B_j, 0\rangle]_4 \rightarrow \langle A_p, B_j, 0\rangle[_4]_4,$
  $[_4 c_0(1)\langle A_x, B_j, 0\rangle\langle A_y, B_j, 1\rangle]_4 \rightarrow \langle A_p, B_j, 0\rangle[_4]_4,$
  $[_4 c_0(1)\langle A_x, B_j, 1\rangle\langle A_y, B_j, 0\rangle]_4 \rightarrow \langle A_p, B_j, 1\rangle[_4]_4,$
  $[_4 c_0(1)\langle A_x, B_j, 1\rangle\langle A_y, B_j, 1\rangle]_4 \rightarrow \langle A_p, B_j, 0\rangle[_4]_4\}$
- $R_5 = \{[_5 D]_5 \rightarrow [_5 D]_5^+[_5 D]_5^-$
  $[_5\langle A_a, B_j, 0\rangle\langle A_b, B_j, 0\rangle]_5^+ \rightarrow \langle A_x, B_j, 0\rangle[_5]_5^+,$
  $[_5\langle A_a, B_j, 0\rangle\langle A_b, B_j, 1\rangle]_5^+ \rightarrow \langle A_x, B_j, 1\rangle[_5]_5^+,$
  $[_5\langle A_a, B_j, 1\rangle\langle A_b, B_j, 0\rangle]_5^+ \rightarrow \langle A_x, B_j, 1\rangle[_5]_5^+,$
  $[_5\langle A_a, B_j, 1\rangle\langle A_b, B_j, 1\rangle]_5^+ \rightarrow \langle A_x, B_j, 0\rangle[_5]_5^+,$
  $[_5\langle A_a, B_j, 0\rangle\langle A_b, B_j, 0\rangle]_5^- \rightarrow \langle A_y, B_j, 0\rangle[_5]_5^-,$
  $[_5\langle A_a, B_j, 0\rangle\langle A_b, B_j, 1\rangle]_5^- \rightarrow \langle A_y, B_j, 0\rangle[_5]_5^-,$
  $[_5\langle A_a, B_j, 1\rangle\langle A_b, B_j, 0\rangle]_5^-$

$$\rightarrow \langle A_y, B_j, 0\rangle[_5]^-_5,$$
$$[_5\langle A_a, B_j, 1\rangle\langle A_b, B_j, 1\rangle]^-_5$$
$$\rightarrow \langle A_y, B_j, 1\rangle[_5]^-_5\}$$

- $i_{in} = 5, i_{out} = i_{xor-out} = 1$

ただし，$0 \leq j \leq m-1$，$0 \leq k \leq m-1$ であり，$[_2\,]_2 = [_{xor}\,]_{xor}$ である．また，入力膜は膜 5 であり，出力膜は膜 1 である．また，オブジェクト c はカウンタの働きをし，オブジェクト D は，膜 5 を分割するために必要なオブジェクトである．

以下に 2 つの m ビットの 2 進数の加算を実行するアルゴリズムの概要を示す．なお，アルゴリズムの入力 2 進数 a，b のそれぞれのビットを $a_i, b_j (0 \leq j \leq m-1)$ と表している．

入力 2 つの m ビットの 2 進数 a，b を表したオブジェクト

$$\langle A_a, B_0, V_{a,j}\rangle, \langle A_a, B_1, V_{a,j}\rangle,$$
$$\cdots, \langle A_a, B_{m-1}, V_{m-1,j}\rangle,$$
$$\langle A_b, B_0, V_{b,j}\rangle, \langle A_b, B_1, V_{b,j}\rangle,$$
$$\cdots, \langle A_b, B_{m-1}, V_{m-1,j}\rangle$$

出力 入力の 2 つの 2 進数の加算された値を表わすオブジェクト

$$\langle A_s, B_0, V_{s,j}\rangle, \langle A_s, B_1, V_{s,j}\rangle,$$
$$\cdots, \langle A_s, B_{m-1}, V_{m-1,j}\rangle$$

Step 1 $x_j = a_j \oplus b_j, y_j = a_j \wedge b_j$ を求める．このステップは，半加算器の動作を表わしている．

Step 2 $p_j = x_j \wedge \neg y_j$ を求める．このステップは，各桁の桁上がりを求める動作を表わしている．

Step 3 以下の条件に従い，$c_j (0 \leq j \leq m-1)$ を決定する．

$$c_j = \begin{cases} 1 & (y_{j-1} = 1) \\ 1 & (k \leq j \text{ に対して } p_{j-1} = p_{j-2} = \cdots \\ & \qquad = p_{k+1} = 1 \text{ で } y_k = 1) \\ 0 & (\text{その他の場合}) \end{cases}$$

このステップでは Step 2 で求めた桁上がりが伝播する桁を求める動作を表わしている．

Step 4 $s_j = x_j \oplus c_j$ を求める．得られた s_j が各桁の加算後の論理値である．

上記の，2 つの m ビットの 2 進数の加算を実行する P システム Π_{add} は，定数回の進化規則の適用によりアルゴリズムが終了するので，以下のような定理が得られる．

定理 6 2 つの m ビット 2 進数を表すオブジェクトを入力とし，加算を行う P システム Π_{add} は，$O(m)$ 種類のオブジェクト，$O(1)$ 個の膜，及び，$O(m^2)$ 個の進化規則を用いて，$O(1)$ ステップで実行可能である． □

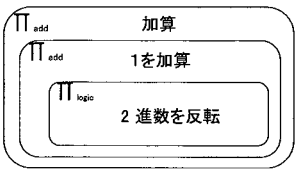

図 2: P システム Π_{sub}

また，減算を実行する P システム Π_{sub} は加算アルゴリズム，及び，論理演算アルゴリズムを組み合わせることにより，アルゴリズムを構築できる．2 進数 a, b に対して減算 $a-b$ を実行する方法を示す．

まず，2 指数 b の 2 の補数を求める．この 2 の補数は，2 進数 b に対して論理演算と加算を実行することにより計算することができる．次に，b の 2 の補数と a を加算する．この加算により得られた値が $a-b$ の減算の値となる．

上記の手順は，本稿で提案した P システム Π_{logic}，及び，Π_{add} を利用し実現することができる．図 2 に減算を実行する P システム Π_{sub} の膜構造を示す．

上記の手順により，以下の定理が得られる．

定理 7 2 つの m ビット 2 進数を表すオブジェクトを入力とし，2 つの値の減算を行う P システム Π_{sub} は，$O(m)$ 種類のオブジェクト，$O(1)$ 個の膜，及び，$O(m^2)$ 個の進化規則を用いて，$O(1)$ ステップで実行可能である． □

また，n 個の m ビットの 2 進数を表わすオブジェクトに対して $\frac{n}{2}$ 対の加算，または，減算を実行するアルゴリズムは，上記で説明した P システム Π_{add}，または，Π_{sub} において，使用するオブジェクト，及び，進化規則の数を $\frac{n}{2}$ 種類準備することにより実行できる．よって，以下の定理が得られる．

定理 8 n 個の m ビット 2 進数を表わすオブジェクトを入力とする $O(n)$ 対の加算，及び，減算は，P システムにおいて，$O(mn)$ 種類のオブジェクト，$O(1)$ 個の膜，及び，$O(m^2n)$ 個の進化規則を用いて，$O(1)$ ステップで実行可能である． □

7 ソート

本節では，n 個の m ビットの 2 進数をソートする P システムの概要を示す．まず最初に，2 つの m ビットの 2 進数の比較交換操作を実行する P システム Π_{CE} の概要を示し，次に，その P システム Π_{CE} を用いて，奇遇転換ソートを実行する P システム Π_{OE} の概要を示す．なお，紙面の関係上，P システムの詳細については省略する．

まず最初に，比較交換操作を実行する P システム Π_{CE} の概要を示す．

Step 1 2 つの m ビットの 2 進数の減算を行う．

Step 2 減算の結果によって，交換操作を行う．

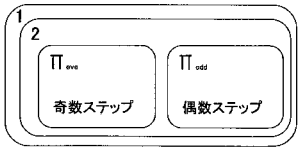

図 3: P システム Π_{OE}

2 つの m ビットの 2 進数の比較交換操作を実行する P システム Π_{CE} は，前述の減算を行う P システム Π_{sub} を用いることにより，定数回の進化規則の適用によりアルゴリズムが終了するので，以下のような定理が得られる．

定理 9 2 つの m ビットの 2 進数の比較交換操作を実行する P システム Π_{CE} は，$O(m)$ 種類のオブジェクト，$O(1)$ 個の膜，及び，$O(m^2)$ 個の進化規則を用いて，$O(1)$ ステップで実行可能である． □

また，n 個の m ビットの 2 進数を表わすオブジェクトに対して $\frac{n}{2}$ 対の比較交換操作を実行するアルゴリズムは，上記で説明した P システム Π_{CE} において，使用するオブジェクト，及び，進化規則の数を $O(n)$ 種類準備することにより実行できる．よって，以下の定理が得られる．

定理 10 n 個の m ビットの 2 進数を表わすオブジェクトを入力とする $O(n)$ 対の比較交換操作は，P システムにおいて，$O(mn)$ 種類のオブジェクト，$O(1)$ 個の膜，及び，$O(m^2n)$ 個の進化規則を用いて，$O(1)$ ステップで実行可能である． □

次に，ソートを行う P システム Π_{OE} の概要について説明する．まず，n 個の m ビットの 2 進数に対して $O(n)$ 対の比較交換操作を実行する P システム Π_{odd}，及び，Π_{even} を以下のように定義する．

$$\Pi_{odd} = (O_{odd}, H_{odd}, \mu_{odd}, \omega_{odd}, R_{odd}, i_{odd-in}, i_{odd-out})$$

$$\Pi_{even} = (O_{even}, H_{even}, \mu_{even}, \omega_{even}, R_{even}, i_{even-in}, i_{even-out})$$

これらの P システムは n 個の m ビットの 2 進数を入力とし，$O(n)$ 対の比較交換操作を実行する．ただし，奇偶転換ソートにおいて，Π_{odd} は偶数ステップに対応する比較交換操作を実行し，Π_{even} は奇数ステップに対応する比較交換操作を実行する．奇偶転換ソートを実行する P システム Π_{OE} は，図 3 に示す様に上記の 2 つの P システムをサブシステムに持つ計算モデルである．

以下に n 個の m ビットの 2 進数の奇偶転換ソートを実行するアルゴリズムの概要を示す．

入力 n 個の m ビットの 2 進数を表したオブジェクト $x_0, x_1, \cdots, x_{n-1}$

出力 比較交換されたオブジェクト $x_0, x_1, \cdots, x_{n-1}$

以下の Step 1, Step 2 を，$\frac{n}{2}$ 回繰り返す．

Step 1 n 個の m ビットの 2 進数に対して $\frac{n}{2}$ 対の x_{2l}，及び，x_{2l+1} の比較交換操作を行う．

Step 2 n 個の m ビットの 2 進数に対して $\frac{n}{2}$ 対の x_{2l+1}，及び，x_{2l+2} の比較交換操作を行う．

n 個の m ビットの 2 進数の奇偶転換ソートを実行する P システム Π_{OE} は，定数回の進化規則を適用するサブシステム Π_{odd}, Π_{even} を $O(n)$ 回適用することによりアルゴリズムを終了するので，以下のような定理が得られる．

定理 11 n 個の m ビットの 2 進数を表わすオブジェクトを入力とする奇偶転換ソートは，P システムにおいて，$O(mn)$ 種類のオブジェクト，$O(1)$ 個の膜，及び，$O(m^2n)$ 個の進化規則を用いて，$O(n)$ ステップで実行可能である． □

8 まとめ

本研究では，膜計算を用いて論理演算，多入力の論理演算，加減算，比較交換操作，及び，奇偶転換ソートを実行するアルゴリズムを示した．

今後の課題としては，乗算，及び，除算アルゴリズムの提案や基本演算を用いた NP 困難問題の解法がある．また，アルゴリズムで使用されるオブジェクト，膜，及び，進化規則の数の削減などが挙げられる．

参考文献

[1] N. Haberman. Parallel neighbor-sort(or the glory of the induction principle). *Technical Report AD-759 248, National Technical Information Service, US Department of Commerce, 5285 Por Royal Road, Springfield VA 22151*, 1972.

[2] A. Leporati and C. Zandron. P systems with input in binary form. *International Journal of Foundations of Computer Science*, 17:127–146, 2006.

[3] L. Pan and A. Alhazov. Solving HPP and SAT by P systems with active membranes and separationrules. *Acta Informatica*, 43(2):131–145, 2006.

[4] G. Păun. Computing with membranes. *Journal of Computer and System Sciences*, 61(1):108–143, 2000.

[5] G. Păun. P system with active membranes: Attacking NP-complete problems. *Journal of Automata, Languages and Combinatorics*, 6(1):75–95, 2001.

[6] C. Zandron, G. Rozenberg, and G. Mauri. Solving NP-complete problems using P systems with active membranes. *Proceedings of the Second International Conference on Uncoventional Models of Computation*, pages 289–301, 2000.