A-003

DNA 計算における乗算および除算アルゴリズム
Procedures for multiplication and division in DNA computing

深川 宏樹 † Hiroki Fukagawa
藤原 暁宏 † Akihiro Fujiwara

1 はじめに

近年新しい計算パラダイムの 1 つである DNA 計算が注目を集めている．DNA 計算では，DNA の持つ超並列性を利用して計算困難な問題に対して多項式時間で解を求めるアルゴリズムや，基本演算として利用される論理演算や算術演算を行うアルゴリズムが提案されている．本論文では，DNA を用いて表現された 2 つの m ビットの 2 進数に対して積を計算する乗算アルゴリズム，および，DNA を用いて表現された 2 つの m ビットの 2 進数に対して除算を行い，商と余りを求める除算アルゴリズムを提案する．提案する乗算アルゴリズム，および，除算アルゴリズムは，どちらも $O(m^2)$ 種類の DNA を用いることにより $O(\log m)$ ステップで実行することができる．

2 準備

2.1 DNA 計算モデル

本研究では，RDNA モデル [2] に基づき，*Merge*, *Copy*, *Detect*, *Separation*, *Selection*, *Cleavage*, *Annealing*, *Denaturation*, *Empty* という 9 つの基本操作を用いる．これらの DNA に対する操作は生化学的操作の組み合わせにより実行できるので，その計算量を $O(1)$ ステップと定義する．

また，文献 [1] において，$O(mn)$ 種類の一本鎖 DNA を用いることにより n 個の m ビットの 2 進数を表現する方法が定義されている．この方法は，n 個の 2 進数を 1 ビット毎に一本鎖 DNA で表現するというアイデアに基づいており，この 1 ビットを表す一本鎖 DNA をメモリ鎖と呼ぶ．本研究ではこのメモリ鎖を用いて，DNA により入力される値を 2 進数で表現するものとする．

2.2 DNA による既知の演算

n 個の m ビットの 2 進数を $O(mn)$ 種類の一本鎖 DNA で表現した場合，文献 [1] において任意の個数のメモリ鎖に値を割り当てるアルゴリズム，任意の個数のメモリ鎖の対に対して論理演算を実行するアルゴリズム，および，加算を実行するアルゴリズムについて，以下のような補題が示されている．

補題 1 任意の個数のメモリ鎖への値の割り当ては，$O(1)$ 種類の DNA を用いることにより $O(1)$ ステップで実行可能である． □

補題 2 n 個の m ビットの 2 進数における $O(n)$ 個の対に対する論理演算は，$O(mn)$ 種類の DNA を用いることにより $O(1)$ ステップで実行可能である．

補題 3 n 個の m ビットの 2 進数における $O(n)$ 個の対に対する加算は，$O(mn)$ 種類の DNA を用いることにより $O(1)$ ステップで実行可能である． □

また，文献 [3] において任意の個数のメモリ鎖の中から最大値を求めるアルゴリズムについて，以下のような補題も示されている．

補題 4 n 個の m ビットの 2 進数の集合中の最大値の計算は $O(mn)$ 種類の DNA を用いることにより $O(\log n)$ ステップで実行可能である． □

3 乗算アルゴリズムの概要

今回提案する乗算アルゴリズムは，ビットシフトと加算の操作の組み合わせにより構成されている．この乗算アルゴリズムの入力は，前述のメモリ鎖を用いた 2 つの m ビットの 2 進数 X, Y として表す．X は掛けられる数を表し，Y は掛ける数を表す．この 2 つの入力から積を求めるために，本アルゴリズムでは乗算を筆算で計算する方法を基にしている．2 つの m ビットの 2 進数 X,Y の筆算では，Y の $i(0 \leq i \leq m-1)$ 番目の値により加算する値を決定する．i 番目の値が 1 ならば加算する値を X を i ビット左シフトした値とし，値が 0 ならば加算する値を 0 とする．この値をすべての i について加算することにより，X と Y の積を求めることができる．

乗算の積を求めるアルゴリズムの概要を以下に示す．

Step 1 2 進数で表現された X を $0 \sim m-1$ ビット左シフトした $2m$ ビットのメモリ鎖 m 個を並列に生成する．この操作は補題 1 のメモリ鎖への値の割当アルゴリズム，および，補題 2 の論理演算アルゴリズムを用いることにより，$O(\log m)$ ステップで実行可能である．

Step 2 2 進数で表現された Y を，値が 1 であるビットの集合と，値が 0 であるビットの集合に分ける．この操作は DNA に対する基本操作を用いることにより $O(1)$ ステップで実行可能である．

Step 3 Step 2 で求めた値が 1 であるビットの集合から，各ビットのビット番号を並列に求め，ビット番号分だけ X を左シフトした一本鎖 DNA を試験管 T_{pre_answer} に抜き出す．次に，Step 2 で求めた値が 0 であるビットの集合から，各ビットのビット番号を並列に求め，$2m$ ビット全ての値が 0 である一本鎖 DNA を試験管 T_{pre_answer} に抜き出す．この操作は DNA に対する基本操作を用いることにより $O(1)$ ステップで実行可能である．

Step 4 試験管 T_{pre_answer} に保存されている m 種類の一本鎖 DNA を並列に加算することで積を求める．この操作は m 種類の一本鎖 DNA に対して $\log m$ 回の加算を実行するので，補題 3 の加算アルゴリズム

†九州工業大学情報工学部電子情報工学科

を用いることにより $O(\log m)$ ステップで実行可能である．

上記のアルゴリズムにより，以下の定理が得られる．

定理 1 2 つの m ビットの 2 進数を乗算し $2m$ ビットの積を得る操作は，$O(m^2)$ 種類の DNA を用いることにより，$O(\log m)$ ステップで実行可能である． □

4 除算アルゴリズムの概要

今回提案する除算アルゴリズム入力は，前述のメモリ鎖を用いた 2 つの m ビットの 2 進数 X，Y として表す．X は割られる数を表し，Y は割る数を表す．この 2 つの入力から商と余りを求めるために，加減算と乗算を用いて関数の近似値を求めるニュートン法により Y の逆数 $\frac{1}{Y}$ の近似値を求め，それを X に乗算することによって商を，また，得られた商と Y の積を X から減算することによって余りを求める．ここで，ニュートン法とは，関数 $f(x)$ を近似値 x_n のまわりで一次近似し，解を求める方法である．入力された Y の逆数 $\frac{1}{Y}$ をニュートン法で求めるには，関数 $f(x) = x - \frac{1}{Y}$ において，$f(x) = 0$ となる x を適当な初期値 x_0 から始めて近似値を求めればよい．ただし，今回提案する除算アルゴリズムでは，Y の逆数 $\frac{1}{Y}$ の精度を上げるために 1 次収束の漸化式ではなく，5 次収束の漸化式 (1)，(2) を用いる．

$$h_i = 1 - Y \times x_i \tag{1}$$

$$x_{i+1} = x_i \times (1 + (1 + h_i^2) \times (h_i + h_i^2)) \tag{2}$$

5 次収束の漸化式では i 番目の近似値に比べ，$(i+1)$ 番目の近似値の方が 5 乗で精度が良くなっている．また，ニュートン法では，通常漸化式の繰り返し回数を決めるために，非常に小さな正の値 ε を用いる．しかし，今回提案する除算アルゴリズムでは，この 5 次収束の漸化式を 3 回繰り返して得られる近似値を逆数 $\frac{1}{Y}$ として扱う．ここで，逆数 $\frac{1}{Y}$ は 1 より小さい小数の値となるため，メモリ鎖で小数の値を表現する必要がある．そのため，入力された m ビットの値の他に，図 1 のような，ビット番号 $-1 \sim -5m$ である $5m$ ビットの DNA 鎖を小数点以下のビットを表すために用いる．また，5 次収束の漸化式を用いて近似値を求める際，i 番目の近似値から $(i+1)$ 番目の近似値を求めると有効ビット数が 5 倍になる．ゆえに，漸化式の繰り返しにおいて，有効ビット数が極端に増加するのを防ぐために，ビット番号が $-5m$ より小さいメモリ鎖とビット番号が $m-1$ より大きいメモリ鎖を切り取り，常に整数部分 m ビットと小数部分 $5m$ ビットの計 $6m$ ビットに保つようにする．

除算の商と余りを求めるアルゴリズムの概要を以下に示す．

Step 1 2 進数で表現された m ビットの X と Y の大小を比較し，Y が大きければ商を 0，余りを Y として出力する．この操作は DNA に対する基本操作と補題 3 の加算を応用した減算を用いて $O(1)$ ステップで実行可能である．

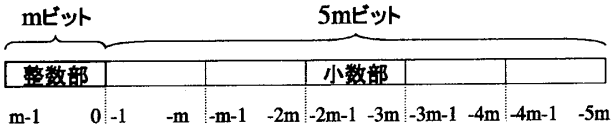

図 1: 小数の表現方法

Step 2 Y の逆数 $\frac{1}{Y}$ をニュートン法を用いて求める．この操作は DNA に対する基本操作と補題 3 の加算，および，定理 1 の乗算を用いて $O(\log m)$ ステップで実行可能である．

Step 3 求めた Y の逆数 $\frac{1}{Y}$ を X に乗算し，その整数部分の値を商とする．この操作は DNA に対する基本操作と定理 1 の乗算を用いて $O(\log m)$ ステップで実行可能である．

Step 4 Step 3 で求めた商を Y に乗算し，その値を X から減算した値を余りとする．この操作は DNA に対する基本操作と補題 3 の加算を応用した減算，および，定理 1 の乗算を用いて $O(\log m)$ ステップで実行可能である．

上記のアルゴリズムにより，以下の定理が得られる．

定理 2 2 つの m ビットの 2 進数を除算し m ビットの商と余りを求める操作は，$O(m^2)$ 種類の DNA を用いることにより，$O(\log m)$ ステップで実行可能である． □

5 まとめ

本論文では DNA 計算を用いて乗算と除算を行うアルゴリズムを提案した．両アルゴリズムは $O(m^2)$ 種類の DNA を用いることにより，$O(\log m)$ ステップで実行可能である．ただし，今回提案したアルゴリズムはあくまでも理論に基づくものであり，実際の DNA を使用した実験は行っていない．したがって，アルゴリズム中で使用する DNA 鎖の生成，基本操作におけるエラー率，除算アルゴリズムで商を求める際に発生する誤差等を考慮した精度の高いアルゴリズムの提案を行うことが今後の研究課題である．

参考文献

[1] A. Fujiwara, K. Matsumoto, W. Chen. Procedures for Logic and Arithmetic operations with DNA Molecules. *International Journal of Foundations of Computer Science*, Vol. 15,No.3 pp.461-474, 2004.

[2] J.H.Reif. Parallel biomolecular computation: models and simulations. *Algorithmica*, Vol. 25, No. 2-3, pp. 142–175, 1995.

[3] S. Kamio, A. Takahara and A. Fujiwara. Procedures for computing the maximum with DNA strands. *Proceedings of the International Conference on Parallel and Distributed Processing Symposium, 2003.*