

8. 四則演算と冪乗

- 2 進法で表された二つの自然数の加算を考える. k ビット (2 進法で k 桁) の自然数の加算は, 次の操作を k 回繰り返すことで行われる.

– 足されるビット x , 足すビット y , 下の桁からの繰上ビット z に対して,

$$x + y + z = 2c + r$$

によって, 足した結果ビット r , 上の桁への繰上ビット c を定める. ここで, $x, y, z, r, c \in \{0, 1\}$ である.

このビット演算を単位に求めた計算量をビット演算量 (bit complexity) という. したがって, 高々 k ビットの二つの自然数の加算のビット演算量は高々 k である.

- 高々 k ビットの二つの自然数の減算は, 高々 k のビット演算量でできる (ただし, ビット演算を適切に拡張するものとする).
- k ビットの自然数と l ビットの自然数の乗算は, 筆算と同様に行うと, 高々 kl のビット演算量でできる. ここで, 桁をずらす操作の計算量は小さいので無視した.
- k ビットの自然数を l ビットの自然数で割って商と剰余を求める計算は, 筆算と同様に行うと, 高々 kl のビット演算量でできる.
- x を乗法の定義された代数系 (正確には半群) の要素, n を自然数として, x^n の計算を考える. 素朴な方法は, 順に x^2, x^3, \dots, x^n と, x による乗算を $n - 1$ 回繰り返す方法である. 乗算回数がより少ない方法を以下で述べる. n の 2 進展開を $n = (n_k \dots n_1 n_0)_2$ とする. ただし, $n_k = 1$ とする.

$$x_0 = x, \quad x_i = x_{i-1}^2 = x^{2^i} \quad (i = 1, 2, \dots, k)$$

と定める. この計算は k 回の乗算でできる. このとき,

$$x^n = \prod_{\substack{n_i=1 \\ 0 \leq i \leq k}} x_i$$

によって x^n が計算できる.

$$\nu(n) = \#\{i \mid n_i = 1, 0 \leq i \leq k\}$$

とおくと, x^n の計算に必要な乗算の回数は, $k + \nu(n) - 1$ 回である. この方法を繰り返し二乗法という. (繰り返し二乗法には, 乗算の順序がこれと異なるものもある.)

注意. 自然数の乗算アルゴリズムには, 筆算よりも高速な方法が知られている. 例えば, 高速フーリエ変換 (FFT) を用いることで, 高々 k ビットの二つの自然数の積を $O(k \log k \log \log k)$ のビット演算量で計算することが出来る (Schönhage-Strassen, 1971). また, 理論的にこれより少し高速な乗算アルゴリズムも最近得られた (Fürer, 2007). 除算についても, ニュートン法によって乗算と同程度の計算量で計算できる.

問題

解答に際して, その問題より前にある問題の結果を用いてもよい.

四則演算のビット演算量として, 前ページで述べたものを使うものとする.

8-1 から 8-3 までは, 2 進法のまま計算すること.

8-1. 2 進法で表された二つの整数 10110 と 11101 の積を筆算で求めよ.

8-2. 2 進法で表された整数 11010101 を 1010 で割った商と剰余を筆算で求めよ.

8-3. 2 進法で表された整数 11111001 を 1101 で割った商と剰余を筆算で求めよ.

8-4. 自然数 n を 2 進法で表したときの桁数は $\lfloor \log_2 n \rfloor + 1$ であることを示せ.

8-5. (a) 素朴な方法で x^{56} を計算したときの乗算の回数を求めよ.

(b) 繰り返し二乗法で x^{56} を計算したときの乗算の回数を求めよ.

8-6. k を自然数, a, b を $2k$ ビットの自然数とする. $a = a_1 2^k + a_0, b = b_1 2^k + b_0$ を満たす高々 k ビットの非負整数 a_0, a_1, b_0, b_1 を取る.

$$t = (a_1 + a_0)(b_1 + b_0), \quad c_2 = a_1 b_1, \quad c_0 = a_0 b_0, \quad c_1 = t - c_2 - c_0 \quad (1)$$

とおくと,

$$ab = c_2 2^{2k} + c_1 2^k + c_0 \quad (2)$$

である. この方法で積 ab を計算したときのビット演算量を $C(k)$ とすると, k によらない定数 c が存在して, $C(k) \leq 3k^2 + ck$ であることを示せ. ただし, (1) における乗算は前ページの方法で計算し, (2) における $2^{2k}, 2^k$ による乗算の計算量は桁をずらすだけなので無視するものとする.

8-7. 8-6 の方法を再帰的に用いることで n ビットの二つの自然数の積を計算できる. このときのビット演算量の最大値を $K(n)$ とすると, n によらない正の定数 α, β が存在して,

$$K(n) \leq \alpha \quad (1 \leq n \leq 3), \quad K(n) \leq 3K\left(\left\lceil \frac{n}{2} \right\rceil + 1\right) + \beta n \quad (n \geq 4)$$

となることが示せる. これを用いて, $K(n) = O(n^{\log_2 3})$ であることを示せ. (この方法をカラツバ法という.)