アルゴリズムとデータ構造

分割統治法とソート

森 立平
mori@c.titech.ac.jp

2018 年 6 月 22 日

今日のメッセージ

- **分割統治法 = 「漸化式作ってそれをプログラムにするだけ」**
- **ソートは分割統治法で解ける**
- **分割統治法の時間計算量は漸化式を立てることによって見積もる**

今日の目標

- いろいろなソートアルゴリズムを習得する

1 分割統治法

「アルゴリズム $\approx$ 漸化式」に従って漸化式をそのままアルゴリズムにしてしまう方法を「分割統治法 (divide and conquer)」という。ユークリッドの互除法や二分探索は分割統治法の特別な場合となる (漸化式の右辺に一つしか、今計算しようとしている関数が登場しないので「decrease and conquer」と呼ばれることもあるようだ)。分割統治法が使える代表的な問題にソート問題がある。

2 ソート問題

与えられた数列 $a_1, a_2, \dots, a_n$ を小さい順に並び変える操作のことをソートと呼ぶ。

入力： $a_1, a_2, \dots, a_n$
出力： $a_{i_1}, a_{i_2}, \dots, a_{i_n}$ ここで $i_1, \dots, i_n \in \{1, 2, \dots, n\}$ は互いに相異なり、$a_{i_j} \le a_{i_{j+1}}$ を満たす

このソートを計算するために分割統治法を使うことができる。だが分割統治法を考える前に、素朴な方法でどれくらいの時間計算量があるかを考えてみよう。

3 選択ソート

$$\mathrm{Ssort}(A) = \begin{cases} [\,], & \text{if } |A| = 0 \\ [\min(A)] \circ \mathrm{Ssort}(A \setminus \min(A)), & \text{otherwise.} \end{cases}$$

ここで $\circ$ は配列の連結とする。

```
void Ssort(int A[], int n){
  int i, j, min;
  for(i = 0; i < n; i++){
    min = i;
    for(j = i+1; j < n; j++){
      if(A[j] < A[min]){
        min = j;
      }
    }
    int z = A[i];
    A[i] = A[min];
    A[min] = z;
  }
}
```

4　挿入ソート

$$\mathrm{Isort}(A) = \begin{cases} [\,], & \text{if } |A| = 0 \\ \mathrm{insert}(\mathrm{last}(A), \mathrm{Isort}(A - \mathrm{last}(A))), & \text{otherwise.} \end{cases}$$

ここで $\mathrm{insert}(a, B)$ は、整数 a をソート済み配列 B の適切な位置に挿入することで得られる配列である。

```
void Isort(int A[], int n){
  int i, j, k, z;
  for(i = 1; i < n; i++){
    for(j = 0; j < i && A[j] < A[i]; j++) ; //二分探索で置き換え可能
    z = A[i];
    for(k = i; k > j; k--) A[k] = A[k-1];
    A[j] = z;
  }
}
```

5　マージソート

マージソートは挿入ソートと良く似ているが、配列を半分のサイズに分割する。

$$\mathrm{Msort}([a_1, \ldots, a_n]) = \begin{cases} [\,], & \text{if } n = 0 \\ \mathrm{merge}(\mathrm{Msort}([a_1, \ldots, a_{\lfloor n/2 \rfloor}]), \mathrm{Msort}([a_{\lfloor n/2 \rfloor + 1}, \ldots, a_n)]), & \text{otherwise.} \end{cases}$$

ここで $\mathrm{merge}(A, B)$ は 2 つのソート済みの配列 A と B について、$A \circ B$ をソートした配列である。

```
int B[N];

void Msort(int A[], int n){
  int i, j, k;
  if(n <= 1) return;
  Msort(A, n/2);
  Msort(A+n/2, n - n/2);
  i = j = k = 0;
  while(i < n/2 && j < n - n/2){
    if(A[i] < A[n/2 + j]) B[k++] = A[i++];
```

```
    else B[k++] = A[n/2 + j++];
  }
  while(i < n/2) B[k++] = A[i++];
  while(j < n - n/2) B[k++] = A[n/2 + j++];
  for(i = 0; i < n; i++) A[i] = B[i];
}
```

6 時間計算量の見積り

分割統治法の時間計算量は漸化式を立てることによって見積もる。簡単のために、n を 2 の冪と仮定するとマージソートの時間計算量 $\chi(n)$ は

$$\chi(n) = 2\chi(n/2) + cn$$

を満たす。よってこの漸化式を解くと

$$\frac{\chi(n)}{n} = \frac{\chi(n/2)}{n/2} + c = \chi(1) + c\log n$$

となり $\chi(n) = O(n\log n)$ であることが分かる。

7 比較に基づくソートの下界

T 回要素の比較をして、その結果だけを使って要素の置換をしてソート問題を解いたとしよう。その場合、$2^T \geq n!$ が成り立つ。$n! \geq \left(\frac{n}{e}\right)^n$ より、$T \geq n\log n - n\log e$ が得られる。よって、比較に基づくソートでは $O(n\log n)$ という時間計算量を改善することはできない。