KYOTO UNIVERSITY

アルゴリズムとデータ構造⑤
～ 順序統計量・動的計画法 ～

鹿島久嗣

DEPARTMENT OF INTELLIGENCE SCIENCE AND TECHNOLOGY

順序統計量：大きい方からk番目の要素は線形時間で発見可能

- 順序統計量：大きい方からk番目の要素
- ソートを使えば$O(n \log n)$
- 工夫すれば$O(n)$で可能
  - 平均的に$O(n)$で見つける方法
  - 最悪ケースで$O(n)$で見つける方法

平均$O(n)$の順序統計量アルゴリズム：クイックソートと同じ考え方で可能

- $q \leftarrow \mathrm{Partition}(A, p, r)$を実行した結果：
  1. $k \leq q$ であれば、求める要素は$A[p\colon q]$にある
  2. $k > q$であれば、求める要素は$A[q+1\colon r]$にある

  –再帰的にPartitionを呼ぶことで範囲を限定していく
- 平均的には問題サイズは半々になっていく：

$$T(n) = T\left(\frac{n}{2}\right) + O(n) = O(n)$$

クイックソートでは$2T\left(\frac{n}{2}\right)$

分割

最悪$O(n)$の順序統計量アルゴリズム：うまく「だいたい真ん中」をとってくる

- $\mathrm{Order}(A, k)$：全要素Aの大きい方からk番目の要素を見つける

1. A を5個ずつのグループに分け、それぞれをソートして、中央値（3番目の値）を見つけ、これらを集めてTとする
2. Tの中央値mをみつける $\mathrm{Order}(A, \lfloor n/10 \rfloor)$
3. Aをmより大きいもの（S_1）、同じもの（S_2）、小さいもの（S_3）に分割する
4. $k \leq |S_1|$ならば$\mathrm{Order}(S_1, k)$、$|S_1| \leq k \leq |S_1| + |S_2|$ならば$m$は目的の要素、$k > |S_1| + |S_2|$ならば$\mathrm{Order}\left(S_3, k - (|S_1| + |S_2|)\right)$

最悪$O(n)$の順序統計量アルゴリズム：計算量の漸化式

- $T(n) = O(n) + T\left(\left\lfloor \frac{n}{5} \right\rfloor\right) + T\left(\left\lceil \frac{3}{4}n \right\rceil\right) = O(n)$
  - ステップ4の分岐でS_1が選ばれたとする
  - 中央値より必ず大きい要素が少なくとも$\frac{1}{4}n$個ある
  - したがって中央値より小さい要素は最大$\frac{3}{4}n$個

最悪$O(n)$の順序統計量アルゴリズム：計算量の導出

- 定理：$s_1 + s_2 + \cdots + s_d < 1$として

$$T(n) = \begin{cases} c & (n \leq n_0) \\ T(s_1 n) + T(s_2 n) + \cdots + T(s_d n) + c'n & (n > n_0) \end{cases}$$

とするとき、$T(n) \leq \frac{cn}{1-(s_1+s_2+\cdots+s_d)}$

- 今回のケースでは$s_1 = \frac{1}{5}, s_2 = \frac{3}{4}$であり、☝の定理を使うと

$T(n) = O(n)$

動的計画法：問題を再帰的に分割しボトムアップに解く

- 動的計画法と分割統治法はともに問題を再帰的に分割
  - 分割統治法：トップダウン
  - 動的計画法：ボトムアップ
- 動的計画法の流れ
  1. 問題の構造を再帰的に捉える
  2. 解を再帰的に構成する
  3. ボトムアップで解を計算する

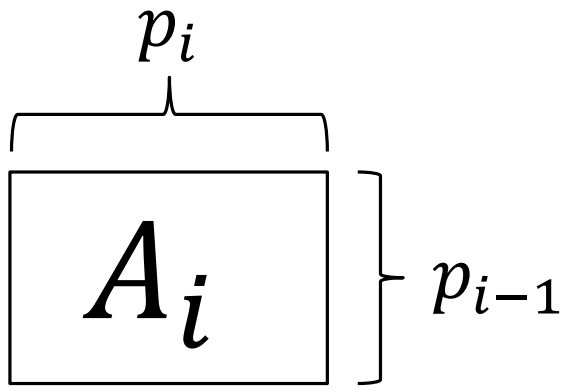
（1, 2 は）分割統治法と同じ

動的計画法のポイント：解の使いまわしによる効率化

- 分割された問題が重複している場合に差が生じる
  - トップダウンでは同じ問題を何度も解くことになる
  - ボトムアップでは解の使いまわしが可能
  - 両者に指数的な差が生じうる
- 逆にいえば、部分問題が重複していることが動的計画法のカギ

動的計画法の例：複数の行列積の計算

- 入力： 行列 $A_1, A_2, \ldots, A_n$
- 出力： 積 $A_1 A_2 \cdots A_n (= A_{1,\ldots,n})$
- 掛け算できる前提
  - $A_i A_{i+1}$の計算は$O(p_{i-1}\, p_i\, p_{i+1})$かかる
- $A_1 : 10 \times 100, A_2 : 100 \times 5, A_1 : 5 \times 50$ とすると
  $\big((A_1 A_2) A_3\big) = 7500, \big(A_1 (A_2 A_3)\big) = 75000$



再帰式の構成：観察

- $A_1 A_2 \cdots A_n (= A_{1,\ldots,n})$を$k$番目で分割するとする
- $A_{1,\ldots,k}$と$A_{k+1,\ldots,n}$を別々に計算して最後に統合
  $O(p_0\ p_k\ p_n)$
- もっとも計算コストの小さい解でk番目の分割が最後にくるとすると、$A_{1,\ldots,k}$と$A_{k+1,\ldots,n}$の分割もコストが最小のはず
  - そうでなければ、コスト最小のものに置き換えれば全体のコストが下がるはず

再帰的な解構成：
行列積をより小さな行列積の積として計算

- $A_{i,\ldots,j}$を計算する最小のコストを $m[i,j]$ とする
- 再帰式：

$$m[i,j] = \begin{cases} 0 & (i = j) \\ \min\limits_{i \le k < j} m[i,k] + m[k+1,j] + p_{i-1}\, p_k\, p_j & (i \neq j) \end{cases}$$

- トップダウン計算（分割統治）だと指数的な計算量

$T(n) = \sum_{k=1}^{n-1}(T(k) + T(n-k) + c) = 2\sum_{k=1}^{n-1} T(k) + cn$

動的計画法の計算量：ボトムアップ計算により多項式時間に

- 再帰式の適用により最小の$m[1, n]$が求まる
- 一方、ボトムアップだと：
  1. $i = j$ の場合について計算（全部ゼロ）
  2. $i = j$-1 の場合について計算
  3. $i = j$-2 の場合について計算
  4. …
  - 上記がnステップ、それぞれで$O(n)$個の再帰式評価、それぞれの評価に$O(n)$ 必要なので、全部で$O(n^3)$の計算量
- バックトラック：実際の掛け算の順番を得るには各再帰式での最良のkを記憶しておく

最長共通部分系列問題：２つの系列に共通に含まれる最長の部分系列をみつける

- 系列$X = (x_1, x_2, \ldots, x_m)$の部分系列(subsequence)とは$X$からいくつかの要素を取り除いたもの
- 2つの系列XとYに対して、系列Zが両方の部分系列のとき、これを共通部分系列とよぶ
- 最長部分系列(LCS; Longest Common Sequence)問題
  - 入力：2つの系列
    $$X = (x_1, x_2, \ldots, x_m), Y = (y_1, y_2, \ldots, y_n)$$
  - 出力： XとYのLCS

最長共通部分系列問題：観察

- $Z = (z_1, z_2, \ldots, z_k)$を$X$と$Y$の任意のLCSとすると

① $x_m = y_n$ のとき、$x_m = y_n = z_k$で、
また、$Z_{k-1} = (z_1, z_2, \ldots, z_{k-1})$は$X_{m-1}$と$Y_{n-1}$のLCS

  - つまり$x_m = y_n = z_k$としてLCSを伸ばせる

② $x_m \neq y_n$ かつ $z_k \neq x_m$ならば、ZはX_{m-1}とYのLCS

③ $x_m \neq y_n$ かつ $z_k \neq y_n$ならば、ZはXとY_{n-1}のLCS

再帰的な解構成：動的計画法により$O(mn)$

- X_iとY_jとのLCSの長さを$c[i,j]$とする
- $$c[i,j] = \begin{cases} 0 & (i = 0 \text{ または } j = 0) \\ c[i-1,j-1]+1 & (x_i = y_j) \leftarrow ① \\ \max\{c[i,j-1], c[i-1,j]\} & (x_i \neq y_j) \leftarrow ②③ \end{cases}$$
- LCSはバックトラックで構成できる
  - 注：LCSは複数ありうる（最適な経路は複数ありうる）