I482F　実践的アルゴリズム特論
10,11回目：乱択アルゴリズム

上原隆平
(uehara@jaist.ac.jp)

乱択アルゴリズム(Randomized Algorithm)

- 乱数を用いたアルゴリズム
  - 乱数を用いることで計算の高速化/単純化を目指すアルゴリズム
  - ラスベガス型
    - いつでも正しい答えを返すことが保証されている。
    - 計算時間は何らかの確率分布に従う
  - モンテカルロ型
    - ときどき間違える。例えば　Yes/No タイプの問題であれば
      - two-sided error; Yes を No という確率も No を Yes という確率も0ではない
      - one-sided error; 上記のどちらかのみ 0 でない
    - 普通は誤答を与える確率を小さくできる

乱択アルゴリズム(Randomized Algorithm)

- 乱数を用いたアルゴリズム
  - ラスベガス型：いつでも正しい解答を出力する
  - モンテカルロ型：一定の割合で誤答を出力する
  - 演習問題1：
    - アルゴリズムAは1回の実行時間が $t(n)$ 時間であり、そのとき3/4の確率で正解を出力する。これを3回実行して多数決をとる。このときの実行時間と正解を出力する確率を求めよ。
  - レポート問題1：
    - アルゴリズムBは1回の実行時間が $t_1(n)$ 時間である。このとき確率 p で正解を出力する。出力が正しいかどうかは別のアルゴリズム C で $t_2(n)$ 時間で確認することができる。正解が得られるまで B, C を繰り返し実行するアルゴリズムの実行時間と正解を出力する確率を求めよ。ここから何が言えるか。

確率解析の例(1): QuickSortの解析

- ソーティング問題
  Input: n 個のデータを記録した配列 a[n]
  Output: 以下の条件を満たす配列 a[n]
  　　a[1]<a[2]<…<a[n]
  ★話を単純にするため、a[i]=a[j], i≠j を満たすペアはないと仮定
- QuickSort は実用上、最速と言われることが多い
  - 典型的な分割統治法に基づくアルゴリズム
  - 都合良く分割されると $O(n \log n)$ 時間で計算が終わる
  - いつでも最悪の場合だと $O(n^2)$ かかる
  - …理論的な解析と速度保証はできるのか？

確率解析の例(1): QuickSortの解析

- QuickSortのおさらい
  - qsort(a,1,n) を呼び出す
  - qsort(a, i, j) が呼び出されると、
    - pivot a[m] を（ランダムに）選ぶ
    - a を a[m] を基準に「前半」と「後半」に分ける。つまり
      i≦ i' < m なら a[i']<a[m]
      m< j' < j なら a[j']>a[m]
      を満たすように並べ替える
    - qsort(a, i, i'), a[m], qsort(a, j', j) がソート結果

ラスベガスタイプのアルゴリズム

[余談]
毎回 $O(j-i)$ 時間かければちょうど中央の値を見つけることもできる。[9.3節]

- QuickSort は実用上、最速と言われることが多いが、、、
  - a[m] がいつでも a[i]..a[j] の中央の値だと
    $T(n) \leqq 2T(n/2) + (c+1)\,n$
    が成立するので、$T(n) = O(n \log n)$ を得る。
  - a[m] がいつでも a[i] や a[j] だと
    $T(n) \leqq T(1)+T(n-1)+(c+1)n$
    が成立するので、$T(n) = O(n^2)$ を得る。

平均的な場合はどうなのか？

確率解析の例(1): QuickSortの解析

- QuickSort は実用上、最速と言われることが多いが、、、
  - 平均的には、a[i] ... a[j] の値が一様に選ばれると仮定する。
    - つまり k 番目のものを pivot にする確率は $1/(j-i+1)$

[定理10.1]
上記の仮定のもとでの QuickSort の実行時間の期待値の上界は　$2n\,H(n) \sim 2n \log n$

オーバーヘッドの少なさから、確かに速いと言えそう

  - 記法
    - a[1]…a[n] の中で k 番目に来るべき要素を s_k と書く。
    - 指示変数（indicator variable）X_{ij} を以下のように定義する

$$X_{ij} = \begin{cases} 0 & s_i \text{ と } s_j \text{ がアルゴリズム中で比較されないとき} \\ 1 & s_i \text{ と } s_j \text{ がアルゴリズム中で比較されるとき} \end{cases}$$

    - QuickSort の実行時間~要素の比較回数= $\sum_{i=1}^{n} \sum_{j>i} X_{ij}$

確率解析の例(1): QuickSortの解析

[定理10.1] 仮定のもとでの QuickSort の実行時間の期待値の上界は $2n\,H(n) \sim 2n \log n$ （期待値の線形性による）

- QuickSort の実行時間の期待値= $E[\sum_{i=1}^{n}\sum_{j>i} X_{ij}] = \sum_{i=1}^{n}\sum_{j>i} E[X_{ij}]$
- 「$p_{ij} = s_i$ と s_j が比較される確率」と定義すると、
  $E[X_{ij}] = p_{ij} \times 1 + (1 - p_{ij}) \times 0 = p_{ij}$
  よって p_{ij} の値を考える
- s_i と s_j はどんなときに比較されるのか？
  1. どちらかが pivot に選ばれている
  2. それまでの計算過程で、別々の qsort にわけられていない
     ⇔ s_i と s_j の間の要素が、まだ pivot として選ばれていない

確率解析の例(1): QuickSortの解析

[定理10.1] 仮定のもとでの QuickSort の実行時間の期待値の上界は $2n\,H(n) \sim 2n \log n$

- s_i と s_j はどんなときに比較されるのか？
  1. どちらかが pivot に選ばれている
  2. それまでの計算過程で、別々の qsort にわけられていない
     ⇔ s_i と s_j の間の要素が、まだ pivot として選ばれていない
  3. $s_i, s_{i+1}, s_{i+2}, \ldots, s_{j-1}, s_j$ が pivot に選ばれる順番は、すべて等確率！
  4. よってこれらの中で s_i か s_j が最初の pivot になる確率: $\frac{2}{j-i+1}$
- よって QuickSort の実行時間の期待値=

$$E[\sum_{i=1}^{n}\sum_{j>i} X_{ij}] = \sum_{i=1}^{n}\sum_{j>i} E[X_{ij}] = \sum_{i=1}^{n}\sum_{j>i} p_{ij} = \sum_{i=1}^{n}\sum_{j>i} \frac{2}{j-i+1}$$
$$= \sum_{i=1}^{n}\sum_{k=2}^{n-i+1} \frac{2}{k} \le 2\sum_{i=1}^{n}\sum_{k=1}^{n} \frac{1}{k} = 2nH(n)$$

確率解析の基本ツール(1)：マルコフの不等式

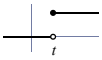
定理10.2(マルコフの不等式)
非負の値をとる任意の確率変数を Y とする。
すると任意の正の実数 t に対して次が成立する: $\Pr[Y \ge t] \le \frac{E[Y]}{t}$
これは次と同値である: $\Pr[Y \ge kE[Y]] \le \frac{1}{k}$

- マルコフの不等式の意味とは…
  「期待値の k 倍以上になる確率は $1/k$ 以下である」
  - QuickSort の実行時間が $4n \log n$ 以上になる確率は ½ 以下
  - 10種類のクーポンを全部集めるために 69 回以上買う確率は 1/3 以下
  …
- Yが非負であることと期待値 $E[Y]$ しかわからないときは、これは改善できない。分散もわかると、もうちょっと改善できる。

確率解析の基本ツール(1)：マルコフの不等式

定理10.2(マルコフの不等式)
非負の値をとる任意の確率変数を Y とする。
すると任意の正の実数 t に対して次が成立する: $\Pr[Y \ge t] \le \frac{E[Y]}{t}$
これは次と同値である: $\Pr[Y \ge kE[Y]] \le \frac{1}{k}$

[証明] 関数 $f(y)$ を次のように定義する: $f(y) = \begin{cases} 0 & y<t \\ 1 & y \ge t \end{cases}$



すると $\Pr[Y \ge t] = E[f(Y)]$ である。
ここですべての y に対して $f(y) \le y/t$ なので、

$$\Pr[Y \ge t] = E[f(y)] \le E[Y/t] = \frac{E[Y]}{t}$$

となる。

確率解析の基本ツール(1)：マルコフの不等式

定理10.2(マルコフの不等式)
非負の値をとる任意の確率変数を Y とする。
すると任意の正の実数 t に対して次が成立する: $\Pr[Y \ge t] \le \frac{E[Y]}{t}$
これは次と同値である: $\Pr[Y \ge kE[Y]] \le \frac{1}{k}$

演習問題2:
正整数 k を一つ固定する。このとき非負の値をとる確率変数 X で以下の条件を満たすのはどのような確率変数だろうか。

$$\Pr[X \ge kE[X]] = \frac{1}{k}$$

確率解析の基本ツール(2)：チェビシェフの不等式

定理10.3(チェビシェフの不等式)
期待値 μ_x と標準偏差 σ_x をもつ確率変数を X とする。
すると任意の正の実数 t に対して次が成立する: $\Pr[|X - \mu_x| \ge t\sigma_x] \le \frac{1}{t^2}$

- 復習：標準偏差とは…
  - 定義: 確率変数Xが値 $x_1, x_2, \ldots, x_n$ をそれぞれ確率 $p_1, p_2, \ldots, p_n$ でとるとする。このとき期待値（平均値）μ_xと分散σ_x^2は次で定義される:
    $\mu_x = \sum_{i=1}^{n} p_i x_i$
    $\sigma_x^2 = \sum_{i=1}^{n} p_i (x_i - \mu_x)^2$
  - 分散の正の平方根σ_xが標準偏差
- 標準偏差（分散）がわかっている確率変数に対しては、こちらの方がマルコフの不等式よりもずっと強力である。

確率解析の基本ツール(2)：チェビシェフの不等式

定理10.3(チェビシェフの不等式)
期待値 μ_x と標準偏差 σ_x をもつ確率変数を X とする。
すると任意の正の実数 t に対して次が成立する: $\Pr[|X-\mu_x| \geq t\sigma_x] \leq \frac{1}{t^2}$

[証明] まず、明らかに次が成立する。

$$\Pr[|X-\mu_x| \geq t\sigma_x] = \Pr[(X-\mu_x)^2 \geq t^2\sigma_x^2]$$

確率変数 Y を $Y=(X-\mu_x)^2$ とおくと、Y の期待値は σ_x^2 である。
Y と t^2 に対してマルコフの不等式を適用すると、

$$\Pr[Y \geq t^2 E[Y]] \leq \frac{1}{t^2}$$

より以下を得る。

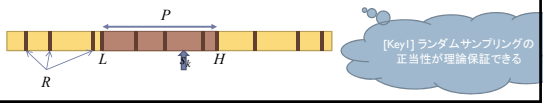
$$\Pr[|X-\mu_x| \geq t\sigma_x] = \Pr[(X-\mu_x)^2 \geq t^2\sigma_x^2] \leq \frac{1}{t^2}$$

確率解析の例(2): 乱択の解析

- 選択問題:
  Input: n 個のデータを記録した配列 a[n] と整数 k
  Output: a[n] の中で k 番目に小さい値 s_k
  - k=1 や k=n (あるいはその前後)なら$O(n)$で簡単に求められる
  - $O(n \log n)$ 時間かけてよいならソートして探せばよい
  - $3n$ 回の比較で求められるアルゴリズムが存在するが、かなり複雑
    - 例えば以下の文献を参照
      - M. Blum, R.W. Floyd, V. Pratt, R. Rivest, and R. Tarjan, "Time bounds for selection", J. Comput. System Sci. 7(1973) 448-461
  - LazySelect: 単純なラスベガスタイプのアルゴリズムで、ほぼ $2n$ 回の比較で動作する確率的アルゴリズム

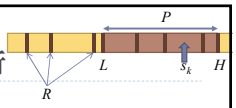
確率解析の例(2): 乱択の解析

- 選択問題:
  Input: n 個のデータを記録した配列 a[n] と整数 k
  Output: a[n] の中で k 番目に小さい値 s_k
  ★話を単純にするため、a[i]=a[j], i≠j を満たすペアはないと仮定
  ★以下では $n^{1/4} \leqq k \leqq n-n^{1/4}$ の場合を考える。(それ以外は、より単純)
- LazySelect: 単純なラスベガスタイプのアルゴリズムの基本アイデア
  1. 適当な数のデータ(R)をランダムサンプリングする
  2. s_k が存在する範囲(P)をRを使って絞り込む(失敗したらやり直し)
  3. 絞り込んだ範囲Pをソートして、その中から s_k を見つけ出す



[Key1] ランダムサンプリングの正当性が理論保証できる

確率解析の例(2): 乱択の解析



- LazySelect:
  Input: n 個のデータを記録した配列 a[n] と整数 k
  Output: a[n] の中で k 番目に小さい値 s_k
  1. 集合R:={a[]から$n^{3/4}$個の要素を復元方式で独立にランダムに選ぶ};
     // 復元方式…同じ要素が複数回選ばれても気にしない
  2. Rを最適な方法でソートする; // $O(n^{3/4} \log n) = o(n)$ 回の比較でOK
  3. $x=kn^{-1/4}$ とし、l, h を以下の通り決める; // s_k はだいたい x の近辺にある
     $l = \lfloor x-\sqrt{n} \rfloor$　　$h = \lceil x+\sqrt{n} \rceil$
  4. L := Rのl番目の要素, H:=Rのh番目の要素とする
  5. L と H がs_k[]の中で何番目に大きいかを調べておく; // 7 で使う
  6. P:={$y \in$a[] | $L \leqq y \leqq H$}
  7. 「$s_k \in$P かつ |P| $\leqq 4n^{3/4}+2$」を満たしていないときは、1からやりなおし;
  8. Pを最適な方法でソートして s_k を見つける.

確率解析の例(2): 乱択の解析

[定理10.2] LazySelect は
1. 確率1-O($n^{-1/4}$)でステップ7のチェックをくぐり抜けて
2. よって $2n$+o(n) 回の比較で無事 s_k を見つけることができる

[注意] 証明を単純にするため、細かい定数は最適化してあるわけではない。例えば非復元方式にするだけでも性能は実用上の性能はアップする。

[定理10.2](2)の証明:

$O(n^{3/4})=o(n)$	1.	集合R:={a[]から$n^{3/4}$個の要素を復元方式で独立にランダムに選ぶ};
$O(n^{3/4} \log n)=o(n)$	2.	Rを最適な方法でソートする;
$O(1)$	3.	$x=kn^{-1/4}$ とし、l, h を以下の通り決める;
$O(1)$	4.	L := Rのl番目の要素, H:=Rのh番目の要素とする
$2n$回の比較	5.	L と H がs_k[]の中で何番目に大きいかを調べておく; // 7 で使う
ステップ5と同時	6.	P:={$y \in$a[]	$L \leqq y \leqq H$}
$O(1)$	7.	「$s_k \in$P かつ |P| $\leqq 4n^{3/4}+2$」を満たしていないときは、1からやりなおし;
$O(\|P\| \log \|P\|) = o(n)$	8.	Pを最適な方法でソートして s_k を見つける.

確率解析の例(2): 乱択の解析

[定理10.2] LazySelect は
1. 確率1-O($n^{-1/4}$)でステップ7のチェックをくぐり抜けて
2. よって $2n$+o(n) 回の比較で無事 s_k を見つけることができる

[定理10.2](1)の証明: アルゴリズムのステップ7をみると、

7. 「$s_k \in$P かつ |P| $\leqq 4n^{3/4}+2$」を満たしていないときは、1からやりなおし;

1. s_k が P に入っていて、かつ
2. |P| が大きすぎなければよい。

(1)-1: s_k が P に入らない場合とは…
L=R[l] が s_k よりも大きいか、
H=R[h] が s_k よりも小さい場合

2. $x=kn^{-1/4}$ とし、l, h は $l = \lfloor x-\sqrt{n} \rfloor$　$h = \lceil x+\sqrt{n} \rceil$
4. L := Rのl番目の要素, H:=Rのh番目の要素
5. L と H がs_k[]の中で何番目に大きいか？
6. P:={$y \in$a[] | $L \leqq y \leqq H$}

⇔ Rがs_k 未満の要素をl 個未満しか含まないか、Rがs_k 未満の要素をh 個以上含む場合

確率解析の例(2): 乱択の解析

[定理10.2] LazySelect は
1. 確率1-O($n^{-1/4}$)でステップ7のチェックをくぐり抜けて…

2. $x=kn^{-1/4}$ とし、l, h は $l=\lfloor x-\sqrt{n} \rfloor$ $h=\lceil x+\sqrt{n} \rceil$
4. L := Rのl番目の要素, H:=Rのh番目の要素
5. L と H がa[]の中で何番目に大きいか？
6. P:={y∈a[] | $L \leqq y \leqq H$}

[定理10.2](1)-1の証明:

(1)-1: s_k が P に入らない場合とは…

(a) Rがs_k 未満の要素をl 個以下しか含まないか、
(b) Rがs_k 未満の要素をh個以上含む場合

(a)を解析するために指標変数 X_i を次のように定義する:

$$X_i = \begin{cases} 0 & i \text{ 番目に選んだ要素が } s_k \text{ 以上} \\ 1 & i \text{ 番目に選んだ要素が } s_k \text{ 未満} \end{cases}$$

すると次を得る: $\Pr[X_i=1]=k/n, \Pr[X_i=0]=1-k/n$

また確率変数 $X=\sum_{i=1}^{n^{3/4}} X_i$ は R の中の s_k 未満の要素数を与える。

さらにそれぞれの確率変数 X_i は独立である。

確率解析の例(2): 乱択の解析

[定理10.2] LazySelect は
1. 確率1-O($n^{-1/4}$)でステップ7のチェックをくぐり抜けて…

[定理10.2](1)-1(a)の証明: (a) Rがs_k 未満の要素をl 個以下しか含まない場合

指標変数 $X_i = \begin{cases} 0 & i \text{ 番目に選んだ要素が } s_k \text{ 以上} \\ 1 & i \text{ 番目に選んだ要素が } s_k \text{ 未満} \end{cases}$

Rの中の s_k 未満の要素の数を表す確率変数 $X=\sum_{i=1}^{n^{3/4}} X_i$
それぞれの確率変数 X_i は独立。
固定された n, k に対して$\Pr[X_i=1]=k/n$ は一定なので、
これは通常のベルヌーイ試行列。
よって平均値μ_Xと分散σ_X^2について以下を得る。

$$\begin{cases} \mu_X = \frac{k}{n} n^{3/4} = kn^{-1/4} \\ \sigma_X^2 = n^{3/4}\left(\frac{k}{n}\right)\left(1-\frac{k}{n}\right) \le \frac{n^{3/4}}{4} \end{cases}$$

確率解析の例(2): 乱択の解析

[定理10.2] LazySelect は
1. 確率1-O($n^{-1/4}$)でステップ7のチェックをくぐり抜けて…

[定理10.2](1)-1(a)の証明: (a) Rがs_k 未満の要素をl 個以下しか含まない場合

$$\begin{cases} \mu_X = \frac{k}{n} n^{3/4} = kn^{-1/4} \\ \sigma_X \le \frac{n^{3/8}}{2} \end{cases}$$

に対してチェビシェフの不等式を使うと以下を得る。

$$\Pr[|X-\mu_X| \ge \sqrt{n}] \le \Pr[|X-\mu_X| \ge 2n^{1/8}\sigma_X] = O(n^{-1/4})$$

(b) Rがs_k 未満の要素をh個以上含む場合
に対しても同様に示すことができる。

確率解析の例(2): 乱択の解析

[定理10.2] LazySelect は
1. 確率1-O($n^{-1/4}$)でステップ7のチェックをくぐり抜けて…

[定理10.2](1)-2: |P| は大きすぎない
(1)-2: |P| が大きすぎない場合とは…

Pが少なくとも $4n^{3/4}+2$ 個の要素を含む場合

⇔ l が十分小さい
hが十分大きい

本質的に(1)-1 と同じ解析が使えて
|P|が小さすぎる場合の起きる確率は $O(n^{-1/4})$ であることが示せる。