# レポートの解答と解説

(2010年9月15日、於金沢大学)

上原隆平
uehara@jaist.ac.jp
http://www.jaist.ac.jp/~uehara

## 問題1：モンテカルロアルゴリズムの繰り返し

- 2個のアルゴリズム：
  - アルゴリズムB: 実行時間が $t_1(n)$, 正解を出す確率 $p$
  - アルゴリズムC: 実行時間が $t_2(n)$, 正解かどうかをチェックする
  - アルゴリズムH: 正解が得られるまで {B→C} を繰り返す
    - {B→C}を1回実行して停止する場合；確率 $p$, 実行時間 $t_1(n)+t_2(n)$
    - {B→C}を2回実行して停止する場合; 確率 $(1-p)p$, 実行時間 $2(t_1(n)+t_2(n))$
    - …
    - {B→C}を $i$ 回実行して停止する場合; 確率 $(1-p)^{i-1} p$, 実行時間 $i\,(t_1(n)+t_2(n))$

    - したがってアルゴリズムHは正解を出す確率は1であり、
      実行時間の期待値$E[H]$は以下の通り。
      $E[H]=(1\cdot p+2(1-p)p+3(1-p)^2 p+\cdots+i(1-p)^i p+\ldots)(t_1(n)+t_2(n))$
      等比級数の解析を使えば、 $E[H]=\frac{t_1(n)+t_2(n)}{p}$ を得る。

      よってアルゴリズムHはラスベガスタイプのアルゴリズム。

## 問題2：非復元式ランダム生成の正当性

- 入力：配列 a[1], a[2], …, a[n];

1. for i=1,2,…,n do
   2. 1 から $n$-$i$+1 までのランダムな整数を一つ生成して、これを $r$ とする;
   3. 「まだ出力していない」aの中で $r$ 番目のものを出力する;
   4. ステップ3で出力した要素に「出力したマーク」をつけて今後選ばれないようにする;
5. end.

- a[i] が $j$ 番目に出力される事象を $X_{ij}$ とし、それが起きる確率を $p_{ij}$ とする。
  題意より、$p_{ij}=1/n$ であることを示せばよい。
- $X_{ij}$ は以下の事象がすべて起きたとき：
  - $j'=1,2,\ldots,j-1$ 番目に $i$ が選ばれない確率： $\frac{n-j'}{n-j'+1}$
  - $j$ 番目に $i$ が選ばれる確率： $\frac{1}{n-j+1}$
- よって $p_{ij}=\frac{n-1}{n}\frac{n-2}{n-1}\frac{n-3}{n-2}\cdots\frac{n-j+1}{n-j+2}\frac{1}{n-j+1}=\frac{1}{n}$ となる。

宝くじは
先に買っても
後で買っても
同じ！

## 問題3：非復元式ランダム生成の効率のよい実装

- 入力：配列 a[1], a[2], …, a[n];

1. for i=1,2,…,n do
   2. 1 から $n$-$i$+1 までのランダムな整数を一つ生成して、これを $r$ とする;
   3. 「まだ出力していない」aの中で $r$ 番目のものを出力する;
   4. ステップ3で出力した要素に「出力したマーク」をつけて今後選ばれないようにする;
5. end.

- ステップ3,4の実装方法例と実行時間を示す。
- 例1: ステップ4で配列に直接∞などの使わない値を書き込む。
  ステップ3では前から sweep して $r$ 番目の要素を探す。
  このときアルゴリズム全体の実行時間の期待値は $O(n^2)$
- 例2: 配列a[]を双方向リストで実現し、ステップ3で出力した要素はステップ4で削除する。
  順序を与える木を使うと、アルゴリズム全体の実行時間は $O(n \log n)$
- 応用問題: 線形時間で実行できるだろうか。

Yes!!

## 問題3：非復元式ランダム生成の効率のよい実装

- 入力：配列 a[1], a[2], …, a[n];

1. for i=1,2,…,n do
   2. 1 から $n$-$i$+1 までのランダムな整数を一つ生成して、これを $r$ とする;
   3. a[$r$] を出力する;
   4. a[$r$] := a[$n$-$i$+1] ;
5. end.

- いつでも a[1] … a[ $n$-$i$+1] に有効なデータが入っている!!
- 配列は壊れるが、単純かつ明らかに線形時間アルゴリズム。