The Incorrectness of a Shuffle Algorithm

DONG Yuxuan <https://www.dyx.name>

11 Jan 2023 (+0800)

Intuition fails frequently while designing stochastic algorithms. This text provides an example. A shuffle algorithm inspired by the merge sort was presented. It seems correct intuitively, but can be proved wrong.

The Algorithm

The input array is divided into two parts as equal length as possible. After two parts are recursively shuffled, they're merged to one array. The merge procedure is similar to the one in merge sort. But the decision of which element should be merged is based on the output of a random number generator, instead of the comparison between two elements. The C implementation is the following.

```
#define N 65536
int *mshuf(int *in, int n)
{
        static int a[N];
        int m, *l, *r, *p;

        if (n < 2)
                return in;
        m = n/2;
        l = mshuf(in, m);
        r = mshuf(in+m, n-m);
        p = a;
        while (l<in+m && r < in+n)
                if (rand()%2)
                        *p++ = *l++;
                else
                        *p++ = *r++;
        while (l<in+m)
                *p++ = *l++;
        while (r<in+n)
                *p++ = *r++;
        return memcpy(in, a, n*sizoef(*a));
}
```

The Algorithm Is Wrong

For $in = \{0, 1, 2, 3, \ldots, n-1\}$, n is a power of 2, we define event S as $a = \{0, 1, 2, \ldots, n-1\}$ after `mshuf()` returned. Assume `mshuf()` is correct, we should have:

$$Pr\{S\} = \frac{1}{n!}$$

We will prove that it's false. To make $a = \{0, 1, 2, \ldots, n-1\}$, after the recursive `mshuf()` calls returned, we must have $l = \{0, 1, \ldots, m-1\}$ and $r = \{m, m+1, \ldots, n-1\}$. Denote this event as D. By assuming the correctness of `mshuf()`, we have:

$$Pr\{D\} = (\frac{1}{m!})^2$$

After D occurred, we must ensure $a[i] = i$ in the merge procedure. We define event A_i as $a[i] = i$ under the condition that D occurred. By the multiplication rule, we have:

$$\begin{aligned} & Pr\{A_0 A_1 \dots A_{n-1}\} \\ =& Pr\{A_0\} Pr\{A_1|A_0\} Pr\{A_2|A_0 A_1\} \dots Pr\{A_{n-1}|A_0 A_1 \dots A_{n-2}\} \end{aligned}$$

When decide the value of `a[i]`, if $i < m$, we randomly chose a value from `l[]` or `r[]`. If $i \geq m$, we can only take the value from `r[]` because all values in `l[]` are merged. Thus we have:

$$Pr\{A_i|A_0 A_1 \dots A_{i-1}\} = \begin{cases} \frac{1}{2}, & i \in [0, m) \\ 1, & i \in [m, n) \end{cases}$$

Thus we get:

$$Pr\{A_0 A_1 \dots A_{n-1}\} = \frac{1}{2^m}$$

Then we get:

$$\begin{aligned} & Pr\{S\} \\ =& Pr\{D A_0 A_1 \dots A_{n-1}\} \\ =& (\frac{1}{m!})^2 \frac{1}{2^m} \\ \neq& \frac{1}{n!} \end{aligned}$$

Thus `mshuf()` is incorrect.