Subject: Re: Randomize array order
Posted by David Streutker on Fri, 27 Jul 2007 14:54:55 GMT
View Forum Message <> Reply to Message

On Jul 27, 6:03 am, Allan Whiteford
<allan.rem...@phys.remove.strath.ac.remove.uk> wrote:
> Conor wrote:
>> Hi everyone!
>
>> Anyone know an efficient way to randomize an array (I have a
>> sorted array that I want unsorted). Initially, I tried something like
>> this:
>
>> array = findgen(1000000)
>> unsort = array[sort(randomu(seed,1000000))]
>
>> It works, but sorting on a million elements is rather slow. Anyone
>> know a faster way?
>
> Slightly different point and probably a bit academic:
>
> If you have a million elements then you have 1000000! (i.e. one million
> factorial) different ways to re-order the data. However, your seed is a
> 4 byte integer which can only take 2^32 different values.
>
> Some messing about suggests that:
>
> 1000000! =~ 10^5568636
>
> which means there are ~ 10^5568636 different ways to re-arrange your
> elements as opposed to the 4 x 10^9 values your seed can take.
>
> Thus, using any of the algorithms suggested you're only going to sample
>
> 10^-5568625 %
>
> of the possible values. This is a really small number. It means that no
> matter how hard you try and how many times you do things you'll never be
> able to access anything but a tiny number of the possibilities without
> doing multiple shufflings - I think it's something like 618737
> sub-shufflings (i.e. 5568636 / 9) but that could be wrong. However, that
> requires producing 618737 seeds per major-shuffle (and you can't use a
> generator based on a 4 byte seed to produce these seeds).
>
> But, since you're only going to be running the code 1000-10,000 times
> (which is much smaller than 4e9) I guess everything will be ok. I don't
> know if anyone has studied possible correlations of results as a
> function of the very small number of seeds (compared to the data),

> whatever random number generator is used and the shuffling method.
> Presumably they have and presumably everything is ok. Does anyone know?
>
> Thanks,
>
> Allan

I'm not sure that I agree. Where in any of our algorithms are we unable to access a (theoretically) possible outcome? As long as we are able to randomly select any element of the array in each step, it should work, right? (I.e., as long as the input array has fewer than 2^32 elements.) In your analysis, shouldn't we be using (2^32)^n for the maximum possible number of randomly generated combinations, where n is the number of steps/elements?

Also, in the Knuth method, the final element may or may not be swapped, depending on whether it is randomly selected for one of the previous swaps.

-David
