

Re: Infinite number of people toss a coin infinite times

Source: <http://sci.tech-archive.net/Archive/sci.math/2004-12/1524.html>

From: Stephen Harris (cyberguard1048-usenet_at_yahoo.com)

Date: 11/25/04

Date: Thu, 25 Nov 2004 11:13:48 GMT

"Mark Nudelman" <markn@greenwoodsoftware.com> wrote in message news:%TAod.658966\$8_6.48009@attbi_s04...

> "HERC777" <herc777@hotmail.com> wrote in message
> news:1101187321.351063.132020@z14g2000cwz.googlegroups.com...
>> If infinite people repeatedly toss a coin, don't you think your diag
>> HHHHTHTHTH.. will have been done already? THINK about it this time,
>> all sequences of log(oo) length have been done, where oo is the number
>> of people.
>
> I don't know exactly what you mean by log(oo) (obviously infinity doesn't
> have a well-defined logarithm), but you seem to be saying that all
> possible
> infinite sequences of {H,T} will be generated by the infinite number of
> coin
> tossers. Could you provide a proof of that?
>
> In fact, your coin-toss example seems rather ill-chosen, since I think the
> best you could offer is a probabilistic argument. What if every one of
> those people throws heads on every toss? Then only one sequence has been
> generated, and any sequence containing any tails is not in the list. I
> think you want better control over what your list contains.
>
> It's hard to understand how you can claim that the diagonal sequence has
> already been generated, when it's clear from the construction of the
> sequence that it is different from every single sequence that's been
> generated by the coin tossers.
>
> --Mark
>

I think one can think of this as a branching algorithm.

The first flip is heads or tails and keep creating all possible outcome.

The second flip is HH or TT or HT or TH

The third flip can produce HHH,TTT,HTH,THT,HHT,TTH

The fourth flip can produce HHHH,HHHT,TTTT,TTTH,HHTT,HTHT,HTTH

sci.math: Re: Infinite number of people toss a coin infinite times

HHTH, TTHT, THTH, THHT, THHH
and so on

This procedure is going to produce every possible sequence, one finite step at a time, and I guess you have to bring in the idea of a completed infinity to project that this procedure generates all possible sequences and all of these sequences could have been produced by flipping a coin randomly.

So one of these generated sequences, which could also occur randomly will be HHH... all the way out and now consider it a completed infinity in the sense it can be named or indexed. So these "completed" infinities of all the sequences can be put in one-to-one correspondence with the counting numbers 1,2,3... which will serve as library for the index assignment.

So all the infinite coin toss sequences are now represented by 1,2,3,...

Now to randomize 1,2,3...oo this set needs to be randomized. Which means changing the order of 1,2,3.. to express all the permutations with which this set could be re-ordered such as 2,1,3... however no numbered index needs or can be duplicated. They just appear in a different order and the number of juxtapositions will grow towards infinity if you look at this as a process.

Now think of this infinite number of juxtapositional re-orderings as each being represented by an infinite number of marbles in an infinite volumed urn. Each marble in the urn is re-indexed with a fraction of a Cauchy sequence. An oracle randomly removes one of the marbles from the urn which represents a random unique ordering of one of the random coin-toss sequences. The oracle is programmed to detect the initial indexed symbol of inscribed on a marble in the urn. It is programmed not to select any marble which begins with the same symbol that the oracle removed in its last extraction. This will bias the selection yet eliminate the possibility of removing the identical sequence (say all HHHs) an infinite number of times, but maintain the random selection of an alternate infinite random sequence. The odds against the oracle alternately selecting an identical indexed marble an infinite number of times from the urn have a probability of one.

HTH
Stephen