Re: Help with a recursive equation

Source: http://sci.tech−archive.net/Archive/sci.math/2007−06/msg05565.html

- *From*: KP <silverphoenix.kp@xxxxxxxxx>
- *Date*: Sat, 30 Jun 2007 14:16:21 −0700

On Jun 30, 9:25 pm, "mensana...@xxxxxxxxxxx" <mensana...@xxxxxxx>
wrote:

> On Jun 30, 9:17?am, "mensana...@xxxxxxxxxxx" <mensana...@xxxxxxx>
> wrote:
>
> > On Jun 30, 3:43?am, KP <silverphoenix...@xxxxxxxxx> wrote:
> >
> > > Hi,
> >
> > > I derived this recursive equation myself when I was trying to
> > > come
> > > up with a general equation for the sum of all binary numbers
>
> I interpreted that as "count" for some reason.
>
> > having "n" digits with m 1's in them.
> >
> > > Let S(n,m) denote the sum I wish to find.
> >
> > > My recursion is
> >
> > This isn't a recursion.

You haven't given a terminating condition,
S() will be called an infinite number of times.

Nevertheles, the above still applies

$$S(n,m) = S(n-1,m) + (2^{(n-1)})* C(n-1,m-1) + S(n-1,m-1)$$

where ^ denotes "raised to the power of"

Could somebody give me any pointers or links that could help in
solving this?

And why recursion? Don't you know the answer is C(n,m)?

Nevertheless, you CAN do it recursively, but not
as shown.

First, lose the last term, + S(n−1,m−1).
This makes no sense, why do you care about
how many 3, 2 or 1 bit counts there are if
you're looking for 4?

Second, what are you trying to accomplish
here: + (2^(n−1))* C(n−1,m−1)? If you're
going to recursively count C(n−1,m−1)
instead of directly calculating C(n,m),
ok. But why the multiplication by 2^(n−1)?

Sorry, the above applies if you're counting.
For summing, the original formula appears
to be correct.

I'll have to change the program

− to check lower bound of m also
− put back the mising terms

```
# Python
import gmpy

def S(n,m):
? ? # terminate recursion if n or m ==0
? ? # also, no point in asking how
? ? # many counts if n not as large
? ? # as m, we already know it's 0
if n==0 or m==0 or n<m:
return 0
s = S(n−1,m) + gmpy.comb(n−1,m−1)*2**(n−1) + S(n−1,m−1)
? ? return s

s = S(8,4)
print
print """ok, here's our answer, calculted recursively"""
print s

print
print """now, check by actually summing
the 8 bit numbers that have 4 1−bits (popcount)"""

correct = []
for i in xrange(2**8):
p = gmpy.popcount(i)
if p == 4:
correct.append(i)
print sum(correct)

## ok, here's our answer, calculated recursively
## 8925
##
## now, check by actually summing
## the 8 bit numbers that have 4 1−bits (popcount)
## 8925
```

That looks better.

Thanks in advance,

KP

Hi,
Thanks for the effort.
I was looking for any pointers to the closed form solution of the recursion.
Any pointers to methods or links would suffice if not the actual answer.

Regards,
KP

.