

Re: Combinatorics question

Source: <http://sci.tech-archive.net/Archive/sci.math/2008-04/msg00834.html>

- *From:* Mariano Suárez-Alvarez <mariano.suarezalvarez@xxxxxxxxxx>
 - *Date:* Sat, 5 Apr 2008 15:27:29 -0700 (PDT)
-

On Apr 5, 6:17 pm, Mensanator <mensana...@xxxxxxxx> wrote:

On Apr 5, 1:52pm, Ketakop...@xxxxxxxx wrote:

Hi all, I have a problem regarding combinatorics. Let's see if I'm capable of explaining myself correctly:

I have S distinct elements, from which I will pick four. So far, the number of groups I can make up this way is $C(S,4)$ or $S! / \{4!(S - 4)!\}$.

Now, from those $C(S,4)$ possibilities, I want to discard those which share 3 elements, well just one of the two. Let's better put an example:

I have the letters a,b,c,d,e,f (6 elements), and I can group them together in groups of four in $C(6,4)=15$ ways:

abcd
abce
abcf
abde
abdf
abef
acde
acdf
acef
adef

Re: Combinatorics question

bcde
bcdf
bcef
bdef
cdef

Now, I have to start to discard: "abce" is the first one to discard, since "abc" already appeared on the first combination, and same with "abcf". I also reject "abde" as "abd" appeared on the first combination, "abdf" too. I'll maintain "abef" since it has only two elements repeated with the first combination. Doing this for all combinations, I end up with only three: "abcd", "abef" and "cdef".

So, I'd like to know if there's an easy way to know how many valid combinations there are, and which ones they are. I'm aware that depending on which one I start to compare from, I will end up with different solutions, but I don't care about that. Any formula, algorithm or help is much appreciated. Thanks in advance,

Start by mapping your 6 letters to a binary number whose bit position (if 1) represents a letter. Thus,

111111 -> fedcba
011001 -> ed a

So, taken 4 at a time means only those binary numbers who have a pop-count (1 bits) of exactly 4:

001111 -> dcba
101011 -> f d ba

Now, for any candidate number to be added to the list, it must have a Hamming Distance (number of different bit positions) greater than 2 with respect to all previous numbers added to the list.

HD is 0 if four bits match
HD is 2 if three bits match
HD is 4 if two bits match
HD is 6 if one bit matches
HD is 8 if no bits match

```
# Python
import gmpy
s = 'abcdef'
L = len(s)
p = 4
```

Re: Combinatorics question

```
ham_target = (p-2)
last = 2**L
the_combos = []
for i in xrange(last):
    if gmpy.popcount(i) == p:
        ok = True
        for j in the_combos:
            h = gmpy.hamdist(i,j)
            if h<=ham_target:
                ok = False
        if ok:
            the_combos.append(i)
            print L,'letters taken',p,'at a time'
            for k in the_combos:
                for n,m in enumerate(reversed(gmpy.digits(k,2).zfill(6))):
                    if m=='1':
                        print s[n],
                        print
```

```
## 6 letters taken 4 at a time
## a b c d
## a b e f
## c d e f
##
## 10 letters taken 4 at a time
## a b c d
## a b e f
## c d e f
## a c e g
## b d e g
## b c f g
## a d f g
## b c e h
## a d e h
## a c f h
## b d f h
## a b g h
## c d g h
## e f g h
## a b i j
## c d i j
## e f i j
## g h i j
```

Keta

It's a bit better (asymtotocally much better!)

Re: Combinatorics question

to just iterate through 4-bit numbers than filter them out:

```
def subsets(n, i, min = 1):
    if i == 0:
        yield 0
    elif n - min + 1 >= i:
        for m in xrange(min, n + 1):
            for s in subsets(n, i - 1, m + 1):
                yield (2 << m) + s

for s in subsets(10, 4):
    print s
-- m
.
```