

Re: Looking for an algorithm to accomplish the following

Re: Looking for an algorithm to accomplish the following

Source: <http://sci.tech-archive.net/Archive/sci.math/2005-08/msg06008.html>

- *From:* "mensanator@xxxxxxxxxxxx" <mensanator@xxxxxxx>
 - *Date:* 30 Aug 2005 17:52:14 -0700
-

fadi wrote:

> Hello all,
>
> Lets say there are N number of sets and each set has the same number of
> attributes as follows:
>
> Set1: W=10, X=15, Y=5, Z=25
> Set2: W=10, X=5, Y=10, Z=20
> Set3: W=10, X=10, Y=20, Z=0
> Set4: W=10, X=30, Y=10, Z=5
>
> Now, to find the sets that if summed, you would get the following values:
>
> Result: W= 20, X=15, Y=30, Z=20
>
> Answer would be Set2 + Set3 of course, but is there an algorithm that I can
> follow that would find this answer for me?

Sure, there's an algorithm. Here's one in Python:

```
from gmpy import *

#
# create a set of sets
#
# indexed by 0, so the first 10 is sets[0][0], the 30 is sets[3][1]

sets = [[10,15,5,25],[10,5,10,20],[10,10,20,0],[10,30,10,5]]

#
# assuming that any combination of sets can be summed, simple binary
# numbers can represent the combinations since a set either is part
# of the sum or it isn't. so there are 2**4 combinations.
#

# t will be the binary number representing the current combination
# t = 5 is 101 in binary, so bits 0 and 2 are set so it is
```

Re: Looking for an algorithm to accomplish the following

Re: Looking for an algorithm to accomplish the following

```
# sets[0] and sets[2] that form the sum

for t in range(2**len(sets)):
# initialize the total to 0
total = [0,0,0,0]

# for each combination, test whether bit b is 1 or 0
for b in range(len(sets)):
# if b is 1, add that set to the running total
if getbit(t,b)==1:
total[0] += sets[b][0]
total[1] += sets[b][1]
total[2] += sets[b][2]
total[3] += sets[b][3]

# now check to see if the total is what we're looking for
if total==[20,15,30,20]:
# found it!
print "%4s" % (digits(t,2)),
print '--->',total
else:
print "%4s" % (digits(t,2)),
print ' ',total
```

```
"""
```

```
0 [0, 0, 0, 0]
1 [10, 15, 5, 25]
10 [10, 5, 10, 20]
11 [20, 20, 15, 45]
100 [10, 10, 20, 0]
101 [20, 25, 25, 25]
110 ---> [20, 15, 30, 20]
111 [30, 30, 35, 45]
1000 [10, 30, 10, 5]
1001 [20, 45, 15, 30]
1010 [20, 35, 20, 25]
1011 [30, 50, 25, 50]
1100 [20, 40, 30, 5]
1101 [30, 55, 35, 30]
1110 [30, 45, 40, 25]
1111 [40, 60, 45, 50]
"""
```

Solution found at t=110, so the answer is sets[1] + sets[2].

>

> It is really more complicated than this where the desired Result can be off
> by certain factor

Re: Looking for an algorithm to accomplish the following

Re: Looking for an algorithm to accomplish the following

Then you simply have to expand

if total==[20,15,30,20]:

into something a bit more complicated.

> and would have condition as to which can be added..etc,

For example, if the sum has to combine only two or three sets,
we could simply test if the popcount (# of 1 bits in t) is
2 or 3.

> but an algorithm would be a starting point for a software I am writing

> hopefully.

>

> Thanks!

.

• ***Follow-Ups:***

◆ ***Re: Looking for an algorithm to accomplish the following***

◇ *From:* fadi

• ***References:***

◆ ***Looking for an algorithm to accomplish the following***

◇ *From:* fadi

• Prev by Date: ***Re: number sequence***

• Next by Date: ***recondite herculean style puzzle***

• Previous by thread: ***Looking for an algorithm to accomplish the following***

• Next by thread: ***Re: Looking for an algorithm to accomplish the following***

• Index(es):

◆ ***Date***

◆ ***Thread***