## henryzz
2018-01-11 13:10

**The Square-Sum problem**

[url]https://www.youtube.com/watch?v=G1m7goLCJDY[/url]
[url]https://www.youtube.com/watch?v=7_ph5djCCnM[/url]

How far can you prove upto 299 seems fairly easy to beat?
Are cubes possible?

## science_man_88
2018-01-11 13:42

[QUOTE=henryzz;477276][url]https://www.youtube.com/watch?v=G1m7goLCJDY[/url]
[url]https://www.youtube.com/watch?v=7_ph5djCCnM[/url]

How far can you prove upto 299 seems fairly easy to beat?
Are cubes possible?[/QUOTE]

Well we can at least use combinatorial arguments to figure out how many times at least 1 square gets used etc. 300 is less than half of 625 so there are only 23 squares to sum up to for the numbers under it. That means, by pigeonhole principle either all of them show up equally ( which low ones can't) or at least one shows up 14 times as a sum in any Hamiltonian path for 300.

## science_man_88
2018-01-11 15:26

[QUOTE=science_man_88;477280]Well we can at least use combinatorial arguments to figure out how many times at least 1 square gets used etc. 300 is less than half of 625 so there are only 23 squares to sum up to for the numbers under it. That means, by pigeonhole principle either all of them show up equally ( which low ones can't) or at least one shows up 14 times as a sum in any Hamiltonian path for 300.[/QUOTE]

Doh should say at least 14 times. You can use the same thing for cubes for numbers under 63 there are only 4 cubes they could sum up to. As 62 mod 4 = 2 and 62/4>15 there are at least 2 cubes with at least 16 pairings for cube sums. This still doesn't include low ones not having enough for equal splitting. If we knew if odd or Even for the powers we can relate that to the edges between pairs of opposite parity or same parity etc.

## science_man_88
2018-01-12 08:31

[QUOTE=science_man_88;477280]Well we can at least use combinatorial arguments to figure out how many times at least 1 square gets used etc. 300 is less than half of 625 so there are only 23 squares to sum up to for the numbers under it. That means, by pigeonhole principle either all of them show up equally ( which low ones can't) or at least one shows up 14 times as a sum in any Hamiltonian path for 300.[/QUOTE]

In fact most must show up 14 times as the first 4 summable squares total 24 sums out of the 52 needed of them leaving 28 to be made up of the remaining 19 squares pushing most up (in fact , 9 squares must appear at least 15 times each)

## henryzz
2018-01-12 13:21

1 Attachment(s)
I have attached a graph of summing upto a cube complete to 124. No solution yet. Unless 108 is an end the solution is at least 235. It is at least all connected.

## science_man_88
2018-01-12 13:37

[QUOTE=henryzz;477360]I have attached a graph of summing upto a cube complete to 124. No solution yet. Unless 108 is an end the solution is at least 235. It is at least all connected.[/QUOTE]

Nice work did my pm clue help ? There is still one other rule in my head, but it's more about number of possible connections versus actual connections.

## henryzz

2018-01-12 14:33

[QUOTE=science_man_88;477363]Nice work did my pm clue help ? There is still one other rule in my head, but it's more about number of possible connections versus actual connections.[/QUOTE]

Are you saying that it can't work for odd powers?

## science_man_88

2018-01-12 14:44

[QUOTE=henryzz;477371]Are you saying that it can't work for odd powers?[/QUOTE]

My clue by pm ,was that the endpoint sum parity is the same parity as the number of sums to the odd based powers. In Matt's first video we find endpoints 18 and 22 these sum to an even number, therefore there must be an even number of sums that sum to the odd squares ( aka the pairings of opposite parity, count 12) . The point about number of possible connections total, is simply that numbers that are half of an even based power have less connections, 18 only had 1 because it couldn't pair with itself and there are only two squares between it and twice it one that can't be made.

## R. Gerbicz

2018-01-12 17:58

[QUOTE=henryzz;477276]
How far can you prove upto 299 seems fairly easy to beat?
[/QUOTE]

Nice problem, easily outperformed that in one day of work.

There exists a solution for n=15,16,17,23 and for all 25<=n<=1048576. (the last checked is n=2^20).
To prove more we give a Hamiltonian cycle for 32<=n<=1048576.

Download the "proof" at my drive [url]https://drive.google.com/file/d/1S9Efu4oTr32nNLQTVCnBa8xtny2-8IF8/view?usp=sharing[/url]
(8.9 MB compressed zip).

There are only two cases in the algorithm:

S: we give simply a valid sequence with length=n.

F: we use the previous sequence with length=n-1, and the two indexes i,j after the F symbol
that is: S=a(1),...,a(n-1) sequence
flip the terms between i to j and insert n to the i-th place (the indexes starts with one).
We have binomial(n-1,2) choices for i,j, and roughly n^(-3/2) chance that this will be a good sequence,
because we see only 3 new terms in the modified sequence:
a(i-1)+n, n+a(j) and a(i)+a(j+1)
hence the probability that we can't find a solution for n is roughly
(1-n^(-3/2))^(n^2)~exp(-sqrt(n)) so we have not only good probability for a continuation, but
this serie converges, so likely this will give a solution for each (large) n value.

So for example
if seq=[2,4,1,5,6,3] and
we see n=7: F 2,4 then seq'=[2,7,5,1,4,6,3]

To give a Hamiltonian cycle we use only 1<i<j<n-1, because with this the first and the last term of the sequence won't change, so if it is a Hamiltonian path then also a cycle.

And this happens in practice the largest n index for that we needed 'S' is at n=6109,
The given solution is a Hamiltonian cycle for all 32<=n<=2^20.

Computed these solutions in only 15 minutes.

ps. there are much more such sequence transformations, but using only these we see only a few S,
so needed to find a sequence from scratch in a few n cases.
Double checking the file with brute force is still possible.

## CRGreathouse
2018-01-12 18:03

[QUOTE=henryzz;477360]I have attached a graph of summing upto a cube complete to 124. No solution yet. Unless 108 is an end the solution is at least 235. It is at least all connected.[/QUOTE]

296 is the first number that I can't easily prove impossible, if I'm understanding correctly.

## science_man_88
2018-01-12 19:42

I think factoring is actually an answer to this in a sense. The sum of powers is 2*T_n - endpoint sum. Modulo or factoring may have implications on any given case.

## science_man_88
2018-01-13 11:01

For anyone still stuck on what I mean, in the cases of n not being 1 mod 3 and endpoints that are additive inverses mod 3 we get the following:

In squares/ even powers, bases that are 1 or 2 mod 3 show up a multiple of 3 times between the base types.

In cases raised to odd exponents the two base types must be equally mixed to fit such endpoint pairs.

This set of circumstances may not be so uncommon T_n divides by 3, 2/3 times and of the reduced cases of endpoints without ordering, 1/3 of cases are such endpoint cases.

## henryzz
2018-01-14 11:13

[url]https://ibb.co/jMDpCR[/url]
Above is the image of 296. 256 has to be an end. Any other issues?

## science_man_88
2018-01-14 11:28

[QUOTE=henryzz;477522][url]https://ibb.co/jMDpCR[/url]
Above is the image of 296. 256 has to be an end. Any other issues?[/QUOTE]

That I can't tell you if the other end is odd or Even because I can barely make out the graph.

## science_man_88
2018-01-14 12:30

[QUOTE=science_man_88;477523]That I can't tell you if the other end is odd or Even because I can barely make out the graph.[/QUOTE]

Edit: might be able to use screen magnification.

## henryzz
2018-01-14 16:19

You can download and zoom in. Sorry the graph program I am using doesn't cope very well.

## science_man_88
2018-01-14 17:43

[QUOTE=henryzz;477544]You can download and zoom in. Sorry the graph program I am using doesn't cope very well.
[/QUOTE]

Decided to make a quick PARI/GP script, I ( not the script) counted 98 even odd links I believe, if so that implies an even number of odd bases, and hence an even sum of squares, hence both endpoints should be even ( except the fact I didn't discard possibly unused paths to come to that conclusion). As to mod 3 I haven't worked it out.

[CODE]my(b=vector(148,i,select(r->sqrtn(r+2*(i-1)+1,3)==sqrtnint(r+2*(i-1)+1,3),2*[1..148])));b[/CODE]

## science_man_88

2018-01-16 11:02

Correction( post 12),
Even powers: bases 1 mod 3 and 2 mod 3 show up -a mod 3 times together, a being the mod 3 of the endpoints.

Odd powers: base 1 mod 3 shows up -a mod 3 times more than base 2 mod 3, or base 2 mod 3 shows up a mod 3 more times than base 1 mod 3.

Composite powers: must follow rules for factors as powers.

## R. Gerbicz

2018-01-17 15:15

Returning to the original square-sum problem,
a non trivial result: the sequence is infinite, for n=(71*25^k-1)/2 there is a solution for every k>=0 integer.

For given "a" sequence that is a solution for n.
let us define T(c)=25*a[1]+c,25*a[2]-c,25*a[3]+c,25*a[4]-c,...,25*a[n]+(-1)^(n+1)*c
and R(a)=a[n],a[n-1],..,a[2],a[1] the reversing of a[].

Then obviously if abs(c)<=12 then in T(c) and in R(T(c)) we see different integers,
and if we consider all these sequences for c=-12...,12 then it will give the
permutation of [13,25*n+12]. Moreover in these sequences the sum of adjacent terms
is square, because we see (25*a(k)+-c)+(25*a(k+1)-+c)=25*(a(k)+a(k+1)), and here
a(k)+a(k+1) was square.
What is left to use the integers from [1,12] to glue these 25 sequences so that
we see squaresums at the sequence endpoints (and in the constructed new pairs).
And this is possible:

if n is odd, a(1)=1 and a(n)=3, then
b=1,T(-1),T(1),R(T(-7)),T(6),T(-6),R(T(0)),11,R(T(-5)),5,4,12,T(-12),T(12),\
R(T(7)),T(-8),R(T(2)),T(-3),9,7,T(4),T(-4),10,6,T(5),R(T(-11)),2,T(-2),8,T(3),\
R(T(-9)),R(T(9)),T(-10),T(10),T(11),R(T(8)),3

is a good sequence, and this is a Hamiltonian cycle, constructed in such a
way, that b(1)=1 and b(n)=3.
So we can use induction for the new sequence.

We need only to find a good sequence, an example for n=35:
v=[1,8,28,21,4,32,17,19,6,30,34,15,10,26,23,13,12,24,25,11,5,20,29,35,14,2,7,18,31,33,16,9,27,22,3];

This gives a solution for n=25*35+12=887, what gives a solution for n=25*887+12=22187 etc.
An explicit formula for the length z(n)=(71*25^n-1)/2.

trivial code to get b from a.
[CODE]
```
F(a,b,c,ty)={local(n,i,v);if(ty==0,v=[c],n=length(b);v=25*b+c*vector(n,i,(-1)^(i+1));
if(ty==-1,v=Vecrev(v)));return(concat(a,v))}

fun_odd(a)={local(r,w,n);
n=length(a);
if(n%2==0,print("Not implemented even n.");return());
if(a[1]!=1||a[n]!=3,print("Invalid input.");return());
w=a;r=[];
r=F(r,w,1,0);
r=F(r,w,-1,1);
r=F(r,w,1,1);
r=F(r,w,-7,-1);
r=F(r,w,6,1);
r=F(r,w,-6,1);
r=F(r,w,0,-1);
r=F(r,w,11,0);
```

```
r=F(r,w,-5,-1);
r=F(r,w,5,0);
r=F(r,w,4,0);
r=F(r,w,12,0);
r=F(r,w,-12,1);
r=F(r,w,12,1);
r=F(r,w,7,-1);
r=F(r,w,-8,1);
r=F(r,w,2,-1);
r=F(r,w,-3,1);
r=F(r,w,9,0);
r=F(r,w,7,0);
r=F(r,w,4,1);
r=F(r,w,-4,1);
r=F(r,w,10,0);
r=F(r,w,6,0);
r=F(r,w,5,1);
r=F(r,w,-11,-1);
r=F(r,w,2,0);
r=F(r,w,-2,1);
r=F(r,w,8,0);
r=F(r,w,3,1);
r=F(r,w,-9,-1);
r=F(r,w,9,-1);
r=F(r,w,-10,1);
r=F(r,w,10,1);
r=F(r,w,11,1);
r=F(r,w,8,-1);
r=F(r,w,3,0);
return(r)}

v=[1,8,28,21,4,32,17,19,6,30,34,15,10,26,23,13,12,24,25,11,5,20,29,35,14,2,7,18,31,33,16,9,27,22,3];
fun_odd(v)
[/CODE]
```

You can easily find a similar rule for even n value.
It could be possible to extend this idea to prove that every n>=25 is in the sequence,
I haven't reached this, for example a nice blocking subproblem was this:
for n>1 there is no such sequence where a(i)-i is even for all i.

## henryzz

2018-01-17 16:06

If I understand correctly you gave an example based upon a known solution for n=35. This led to solutions for n=(71*25^k-1)/2.
Given a solution for n=37 you would be able to generate solutions for (75*25^k-1)/2 in the same way.

It also looks like 25 should be replaceable with any odd square. I am not quite certain why 9 wouldn't qualify for this. I need to give this more thought. I suppose 25 relies on there being a solution that connects the 25 sequences together. This wouldn't happen for 9. Is this guaranteed to work for any odd square >=25?

## R. Gerbicz

2018-01-17 16:26

[QUOTE=henryzz;477791]If I understand correctly you gave an example based upon a known solution for n=35. This led to solutions for n=(71*25^k-1)/2.
Given a solution for n=37 you would be able to generate solutions for (75*25^k-1)/2 in the same way. [/QUOTE]

Correct, but the goal was to prove that n>=25 is in the sequence. Since it wasn't reached I've posted the above weaker result.

[QUOTE=henryzz;477791]
It also looks like 25 should be replaceable with any odd square. I am not quite certain why 9 wouldn't qualify for this. I need to give this more thought. I suppose 25 relies on there being a solution that connects the 25 sequences together. This wouldn't happen for 9. Is this guaranteed to work for any odd square >=25?[/QUOTE]

Yes, you need odd square, otherwise c=k^2/2 gives the same residue as -c.

Using only odd squares and the above construction is not enough, since
sum(k>1,1/k^2)<1, so the density is smaller than 1, not every integer will be covered.

The main following idea (from me) was to use two sequences a0 and a1, one for length of n,
and one for length (n+1), with this you could build a solution for length=e^2*n+res for
all res=[(e^2-1)/2,(3*e^2-1)/2) if you can give the base cases for all [N..e^2*N)
then with induction there is a solution for all n>=N. The main problem is that to use induction
you need the same parity of the position of k in a0() and in a1(), and it is hard
to maintain this in the induction.
Maybe one could reach say 25<=n<=81*2^20 *easily* using e=9, but in the induction it will
be collapsing.

## R. Gerbicz

2018-01-21 11:09

There is a solution for every n>=25 !
Giving a Hamiltonian cycle for all n>=32, see my linked code:
[url]https://drive.google.com/file/d/1XpYRdJR00CldIfssjDW1clkh-HdPA708/view[/url]
somewhat large file ( 5.7 MB ), but the actual code is pretty short,
using a constant memory, and fast algorithm in O(n*log(n)) time.
[it could be faster but using O(n) memory]
n=10000000 is solved in roughly 5 seconds

examples:
[CODE]
```
gerbicz@gerbicz:~/gmp-6.1.2$ gcc -o squares squares.c -lm
gerbicz@gerbicz:~/gmp-6.1.2$ ./squares -n 10000 -out "seq.txt"
Computed the sequence for n=10000 in 0 sec.
gerbicz@gerbicz:~/gmp-6.1.2$ ./squares -n 100 -screen
1 80 64 36 85 84 16 48 73 71 50 94 75 69 100 96 4 5 76 24 97 72 49 95 74 47 17 19 45 99 22 59 62 82 39 25 11 70
51 30 91 53 28 93 7 57 43 78 66 34 87 13 68 32 89 55 9 27 54 90 31 18 46 98 2 14 86 35 65 79 42 58 63 81 40 41
23 26 38 83 61 60 21 15 10 6 3 33 67 77 92 52 29 20 44 37 12 88 56 8
Computed the sequence for n=100 in 0 sec.
gerbicz@gerbicz:~/gmp-6.1.2$
```
[/CODE]
note: for n<=2032 we are actually using a precomputed table, for larger n values
we are using a recursion.

The above ideas were quite good, call seq0 and seq1 a nice pair of sequence iff
length(seq0)=n and length(seq1)=n+1 and for all k<=n it is true that
if k=seq0[p]=seq1[q] then p-q is even (in other word p+q is even), so we see the same
terms in the same position's parity.

Using such a pair of sequence I'm giving a nice pair of sequence for every
49*n+res, where res=24..72, this a complete residue system for mod=49, and
with a larger precomputed table I'll give a solution for every n=41..2032.
This completes the proof for n>=41, for smaller n values using another
lookup table.

The construction is similar to the above strategy, just using two(!) sequences,
one with length=n and one with length=n+1, lets define

T(c,0)=49*seq0[1]+c,49*seq0[2]-c,..,49*seq0[n]+(-1)^(n+1)*c
T(c,1)=49*seq1[1]+c,49*seq1[2]-c,..,49*seq1[n+1]+(-1)^(n+2)*c

so T(c,1) is longer by one. For each c=-24..24 use exactly one of them
or its reversed sequence, use the remaining [1,24] integers to glue/attach
them at the sequence endpoints, so we are seeing only square pairsums.
It was a somewhat harder problem as above, because we needed
two glues: one for N=49*n+res and one for N=49*n+res+1 so that the resulted
pair of sequence is still nice, to make sure the induction will work.
Observe that it is determined what we should choose T(c,0) or T(c,1) for each c
(so here you have no real choice), because these contain the same integers, and we
know the extra term in T(c,1), why(?), because if in seq1 the (n+1) is in
position k, then k==n+1 mod 2.

Suprisingly my code found these glues in roughly 1 second, and
the basic sequences in 1 hour (could be maybe somewhat faster) using one thread.

In code data[][][] contains the glues, the very large S[][]
the basic sequences for small n values.

Note that in all sequences a[1]=1 and for even n: a[n]=8, for odd n: a[n]=3,
and we maintain this property also, so it'll be a Hamiltonian cycle.

Furthermore, the code checks (in several ways) that the output sequence
is good, for example checking that sum(i=1,n,a[i]^3)=(n*(n+1)/2)^2 mod 2^64 is true,
and the square tests are done for all consecutive pair sums.

If we'd need just Hamiltonian path then we can use 25*n+res in the recursion:
use a[1]=1 and for even n: a[n]=2, for odd n: a[n]=3. (this results smaller tables).
And for 9*n+res there is absolutely nothing in our plate.

## LaurV

2018-01-21 21:16

[QUOTE=R. Gerbicz;478035]There is a solution for every n>=25 ![/QUOTE]
:tu: Very nice! Man, you are good!

## henryzz

2018-01-22 03:20

Very nice. Maybe someone should let numberphile know.
Also I believe papers have been published on less worthwhile subjects.

## MattcAnderson

2018-01-22 19:50

I posted to the numberphile youtube page that this problem has been solved.

Regards,
Matt

## science_man_88

2018-01-22 20:07

[QUOTE=MattcAnderson;478144]I posted to the numberphile youtube page that this problem has been solved.

Regards,
Matt[/QUOTE]

I posted a link to this thread on the video page near the start of it as well.

## R. Gerbicz

2018-01-23 15:59

[QUOTE=henryzz;478090]Also I believe papers have been published on less worthwhile subjects.[/QUOTE]

I've thought the same in the recent days, maybe I'll come up with a paper.

## Auto Felix

2018-03-16 13:12

How do you know that all sequences start with 1, and end with 8 (for even) or 3 (for odd)? Is there a proof?

## R. Gerbicz

2018-03-17 03:05

[QUOTE=Auto Felix;482534]How do you know that all sequences start with 1, and end with 8 (for even) or 3 (for odd)? Is there a proof?[/QUOTE]

Thanks for your interest!
I"ve constructed all basic sequences that holds this, and then by induction we still maintain this. Note that all integers in [1,24] is free, because T(c,0) and T(c,1) doesn't use them. So we can use these small (1,3,8) integers at the

endpoints.

You could ask why we haven't used a shifted and reversed representation, so the sequences starts with 1,3 and 1,8; because in that case we don't know the last term of seq0 and seq1, so we can't glue the sequences. Or why we haven't used the constant 3 at the end for every sequence, because in that case 3=seq0(n)=seq1(n+1), but (n+1-n)=1 is odd so the parity position condition wouldn't be true. There are some traps here.

## R. Gerbicz

2021-11-02 04:51

[QUOTE=R. Gerbicz;478035]
The above ideas were quite good, call seq0 and seq1 a nice pair of sequence iff
length(seq0)=n and length(seq1)=n+1 and for all k<=n it is true that
if k=seq0[p]=seq1[q] then p-q is even (in other word p+q is even), so we see the same
terms in the same position's parity.
...
Observe that it is determined what we should choose T(c,0) or T(c,1) for each c
(so here you have no real choice), because these contain the same integers, and we
know the extra term in T(c,1), why(?), because if in seq1 the (n+1) is in
position k, then k==n+1 mod 2.
[/QUOTE]

There was a request to explain this more, took me some time to understand some tricks.

First why parity is so important here. It is trivial: for each c=-24..24 we choose T(c,0) or T(c,1), exactly one of them. Say c=6,-6 if you choose T(6,1) and T(-6,1) then we are good, covering the 6 and -6 residues mod 49, the same happens for choosing T(6,0) and T(-6,0). But why isn't this broken if we'd select T(6,0) and T(-6,1), these contains for all x<=n the 49*x+-6 numbers, if you choose 49*x+6 in T(6,0) and 49*x+6 in T(-6,1) then the sequence is broken, the same integer appears twice. The trick is that in seq0 and seq1 all x<=n numbers appear in the same parity position, this forces that if one seq you'll choose 49*x+6 then in the other it'll be 49*x-6.

Let res=47 so we want the solution for N=49*n+47, when we know the good sequence pair for n and n+1. Just for the example remain at c=6,-6 the extra n+1 term in seq1 is in the parity position of n+1 mod 2, because with seq0 these contain the 1..n integers in the same parity position, so the largest two terms in T(6,1) and T(-6,1) are:
y0=49*(n+1)+(-1)^(n+1)*6 and
y1=49*(n+1)+(-1)^(n+1)*(-6).
First notice that all terms except y0,y1 in these two seq are at most 49*n+24, so not larger than N=49*n+47 and this is true in general, because we recurse for res>=24. What is really not explained, but in the code (quite hidden), that you need to know also the parity of n to know these two terms: say n is even then y0=49*n+43 and y1=49*n+55, but N=49*n+47, so here y0 should be selected but not y1 and this is forced, so you have to select T(6,1) and T(-6,0).

What is also important: you have to maintain the parity condition for N (to make a working induction), how do you ensure that this will hold? When you glue the sequences you're using 1 integer or n or n+1 integers in a block, if you know the parity of n you'll know the parity between subsequences, and a smaller trick: for a subsequence it is enough to see that the first term goes to the same parity position.

ps. in some places we used the reverse of the T(c,) subsequence to give more breath to the method.

## arbooker

2021-11-12 08:34

[QUOTE=R. Gerbicz;592273]There was a request to explain this more, took me some time to understand some tricks.
[/QUOTE]You really should write this up for publication, if only to provide a reference point and a proof that's more than a quick description on an internet forum. You're welcome to send it to me for consideration at JNT, and there are loads of other good options. (The American Mathematical Monthly comes to mind, given that the problem has broad interest and the proof is also very accessible.)

## R. Gerbicz

2021-11-15 09:12

[QUOTE=arbooker;592995]You really should write this up for publication, if only to provide a reference point and a proof that's more than a quick description on an internet forum. You're welcome to send it to me for consideration at JNT, and there are loads of other good options. (The American Mathematical Monthly comes to mind, given that the problem has broad interest and the proof is also very accessible.)[/QUOTE]

Thanks for the idea, then I'll try the AMM first, need to write down the solution.
It is a somewhat recreational Maths problem, but worth a paper.

## LaurV

2022-08-08 04:52

Good Job from HexagonVideos. Give that guy a high five!

[YOUTUBE]-vxW42R47bc[/YOUTUBE]

## Angel

2022-08-08 13:11

[QUOTE=LaurV;610952]Good Job from HexagonVideos. Give that guy a high five!

[YOUTUBE]-vxW42R47bc[/YOUTUBE][/QUOTE]

Yeah, I came to know this problem thanks to that video! Then I started thinking about it and came up with some interesting variations, the main of which is pretty meta, you can check it out in [URL="https://mersenneforum.org/showthread.php?t=27993"]this post I made[/URL]; if you have any ideas of a possible attempt to solve it please post it there.

## hoppi

2022-08-08 13:30

I have found this site after watching that video. And he talks about this thread in the video. But I know about this problem since 2014 as per the link: [URL]https://rosecode.neocities.org/problem.html?id=115[/URL]

## oreotheory

2022-11-30 00:45

I can't help wondering about the square sum problem where three (or more) adjacent numbers have to add up to a square. I'm sure many of the same techniques Gerbicz used could be applied there...

## Nabi744

2023-08-11 09:45

I have a question of this algorithm you made. Can you tell me how to find the glues(data) and basic sequences(S) when C is 25?

## LaurV

2023-08-17 01:52

[QUOTE=oreotheory;618723]I can't help wondering about the square sum problem where three (or more) adjacent numbers have to add up to a square. I'm sure many of the same techniques Gerbicz used could be applied there...
[/QUOTE]

I am sure this is necroposting, but let's try, you want a+(a+1)+(a+2) to be a square? This is 3(a+1) and it will work for any number which is 3 times a square minus 1. :razz:

## Azruine

2023-09-21 18:46

1 Attachment(s)
[QUOTE=R. Gerbicz;593153]Thanks for the idea, then I'll try the AMM first, need to write down the solution.
It is a somewhat recreational Maths problem, but worth a paper.[/QUOTE]

I have a question about your method.

How can we find glues [B]with good position[/B]? Can I just check position of 1~24? or Should I check all numbers, i.e. 1~24, 25~73 and corresponding numbers?

I found glues which is 1, 3 at odd index, and 8 at even index. and they are just messed up with 2 or more iteration since 4 was at different index :(

I attached my code, which finds glues for 25 but seems like not working...

## R. Gerbicz

2023-09-22 19:17

[QUOTE=Azruine;638843]I have a question about your method.

How can we find glues [B]with good position[/B]? Can I just check position of 1~24? or Should I check all numbers, i.e. 1~24, 25~73 and corresponding numbers?

I found glues which is 1, 3 at odd index, and 8 at even index. and they are just messed up with 2 or more iteration since 4 was at different index :([/QUOTE]

Even if your method is good with the gluing you still need that the base pair of sequences are good, so:
For every x<=n it is in the same parity position for the two sequences.

Track also how many numbers you have already placed, need only the parity. When you place a single number, or n or n+1 numbers you will know how the parity is changing (for this you need only parity of n).
With the above parity you can assure that you are not placing down the same remainder residue twice, just see the first number of the placed sequence, since with the parity trick every number will be good or wrong.

About search: it was a bactracking algorithm, basically without any tricks, but since it was run in 1 second there was no need to optimize it. Even this is true, to find the base sequences.

## HermannSW

2023-09-23 08:13

I looked into 1st video until problem statement.
Sparse graph, so I implemented recursive search in Python as good enough.

Then read Robert's posting about N>15 and changed script.
Became slow above 30 with 784 different solutions:
(every solution sequence is solution as well when reversed, so total number of solutions is even)
[CODE]$ time python nsum.py 25 | wc --lines
average vertex degree: 2.56
20

real 0m0.035s
user 0m0.026s
sys 0m0.011s
$ time python nsum.py 30 | wc --lines
average vertex degree: 2.8
40

real 0m0.402s
user 0m0.397s
sys 0m0.007s
$ time python nsum.py 32 | wc --lines
average vertex degree: 2.875
784

real 0m1.213s
user 0m1.207s
sys 0m0.007s
stammw:python$
[/CODE]

Added stop argument to make script stop after 1st solution;
[CODE]$ time python nsum.py 32 stop
average vertex degree: 2.875
[1, 8, 28, 21, 4, 32, 17, 19, 30, 6, 3, 13, 12, 24, 25, 11, 5, 31, 18, 7, 29, 20, 16, 9, 27, 22, 14, 2, 23, 26, 10, 15]

real 0m0.029s
user 0m0.024s
sys 0m0.004s
$ time python nsum.py 40 stop
average vertex degree: 3.25
[1, 3, 6, 10, 39, 25, 24, 40, 9, 16, 33, 31, 18, 7, 2, 23, 26, 38, 11, 5, 20, 29, 35, 14, 22, 27, 37, 12, 13, 36, 28, 8, 17, 19, 30, 34, 15, 21, 4, 32]

real 0m0.156s

user 0m0.149s
sys 0m0.006s
$ time python nsum.py 45 stop
average vertex degree: 3.5555555555555554
[1, 3, 6, 10, 15, 21, 28, 8, 17, 32, 4, 5, 11, 14, 35, 29, 20, 44, 37, 12, 13, 36, 45, 19, 30, 34, 2, 7, 18, 31, 33, 16, 9, 27, 22, 42, 39, 25, 24, 40, 41, 23, 26, 38, 43]

real 0m1.338s
user 0m1.332s
sys 0m0.004s
$[/CODE]

So with increasing average vertex degree and no sophisticated depth first search Python script is good for up to 45.
Pylinted and "black" formatted 50 line Python script:
[URL="https://gist.github.com/Hermann-SW/19d429787d7f8e4cf43cd04e8345868e"]https://gist.github.com/Hermann-SW/19d429787d7f8e4cf43cd04e8345868e[/URL]
[CODE]$ python nsum.py
average vertex degree: 2.0
[8, 1, 15, 10, 6, 3, 13, 12, 4, 5, 11, 14, 2, 7, 9]
[9, 7, 2, 14, 11, 5, 4, 12, 13, 3, 6, 10, 15, 1, 8]
$ python nsum.py 23
average vertex degree: 2.4347826086956523
[2, 23, 13, 12, 4, 21, 15, 10, 6, 19, 17, 8, 1, 3, 22, 14, 11, 5, 20, 16, 9, 7, 18]
[9, 16, 20, 5, 11, 14, 22, 3, 1, 8, 17, 19, 6, 10, 15, 21, 4, 12, 13, 23, 2, 7, 18]
[18, 7, 2, 23, 13, 12, 4, 21, 15, 10, 6, 19, 17, 8, 1, 3, 22, 14, 11, 5, 20, 16, 9]
[18, 7, 9, 16, 20, 5, 11, 14, 2, 23, 13, 12, 4, 21, 15, 10, 6, 19, 17, 8, 1, 3, 22]
[18, 7, 9, 16, 20, 5, 11, 14, 22, 3, 1, 8, 17, 19, 6, 10, 15, 21, 4, 12, 13, 23, 2]
[22, 3, 1, 8, 17, 19, 6, 10, 15, 21, 4, 12, 13, 23, 2, 14, 11, 5, 20, 16, 9, 7, 18]
$[/CODE]

## HermannSW

2023-09-24 06:54

[QUOTE=henryzz;477360]I have attached a graph of summing upto a cube complete to 124. No solution yet. Unless 108 is an end the solution is at least 235. It is at least all connected.[/QUOTE]
[URL="https://mersenneforum.org/attachment.php?attachmentid=17531&d=1515784794"]https://mersenneforum.org/attachment.php?attachmentid=17531&d=1515784794[/URL]
The attached graph does not have a solution.
Because we are searching a hamilton path, visitng each vertex exactly once.
All connected graphs having a hamilton path have at most two vertices of degree 1.
And you graph has a 2-digit number of vertices of degree 1, so has no hamilton path.

I modified my code from yesterday in several ways.
cube seems to be too sparse, so I tried tetrahedral numbers n*(n+1)*(n+2)/6
[url]https://oeis.org/A000292[/url]
but found no solution.

Then I tried triangular numbers n*(n+1)/2
[url]https://oeis.org/A000217[/url]
and found many. But triangular numbers are more dense than squares and not less.

So finally I tried n^2+c and those have solutions (I tested for up to c=10).
Starting at higher numbers for higher c.
So not much sparser than square, but a little:
[CODE]$ for((i=1;i<=50;++i)); do echo -n $i" "; python nsum2.py $i stop; done
1 average vertex degree: 0.0
2 average vertex degree: 0.0
3 average vertex degree: 0.0
4 average vertex degree: 0.0
5 average vertex degree: 0.0
6 average vertex degree: 0.0
7 average vertex degree: 0.0
8 average vertex degree: 0.0
9 average vertex degree: 0.0
10 average vertex degree: 0.0
11 average vertex degree: 0.0
12 average vertex degree: 0.0

```
13 average vertex degree: 0.0
14 average vertex degree: 0.0
15 average vertex degree: 0.0
16 average vertex degree: 0.0
17 average vertex degree: 0.0
18 average vertex degree: 0.1111111111111111
19 average vertex degree: 0.21052631578947367
20 average vertex degree: 0.3
21 average vertex degree: 0.38095238095238093
22 average vertex degree: 0.45454545454545453
23 average vertex degree: 0.5217391304347826
24 average vertex degree: 0.6666666666666666
25 average vertex degree: 0.8
26 average vertex degree: 0.9230769230769231
27 average vertex degree: 1.037037037037037
28 average vertex degree: 1.1428571428571428
29 average vertex degree: 1.2413793103448276
30 average vertex degree: 1.4
31 average vertex degree: 1.5483870967741935
32 average vertex degree: 1.6875
33 average vertex degree: 1.8181818181818181
34 average vertex degree: 1.9411764705882353
35 average vertex degree: 2.0
36 average vertex degree: 2.0555555555555554
37 average vertex degree: 2.108108108108108
38 average vertex degree: 2.210526315789474
39 average vertex degree: 2.3076923076923075
40 average vertex degree: 2.4
41 average vertex degree: 2.4878048780487805
42 average vertex degree: 2.5714285714285716
43 average vertex degree: 2.6511627906976742
44 average vertex degree: 2.727272727272727
45 average vertex degree: 2.8
[1, 34, 12, 23, 36, 10, 25, 21, 38, 8, 27, 19, 40, 6, 29, 45, 14, 32, 3, 43, 16, 30, 5, 41, 18, 17, 42, 4, 31, 28, 7, 39,
35, 11, 24, 22, 13, 33, 2, 44, 15, 20, 26, 9, 37]
46 average vertex degree: 2.869565217391304
[1, 34, 12, 23, 36, 10, 25, 21, 14, 32, 3, 43, 16, 30, 5, 41, 18, 17, 42, 4, 31, 28, 7, 39, 35, 11, 24, 22, 37, 9, 26, 20,
15, 44, 2, 33, 13, 46, 45, 29, 6, 40, 19, 27, 8, 38]
47 average vertex degree: 2.9361702127659575
[1, 34, 12, 23, 36, 10, 25, 21, 38, 8, 27, 19, 40, 6, 29, 17, 42, 4, 31, 15, 20, 26, 9, 37, 22, 24, 11, 35, 39, 7, 28, 18,
41, 5, 30, 16, 43, 3, 32, 14, 45, 46, 13, 33, 2, 44, 47]
48 average vertex degree: 3.0
[1, 34, 12, 23, 36, 10, 25, 21, 14, 32, 3, 43, 48, 11, 24, 35, 39, 7, 28, 18, 17, 42, 4, 31, 15, 20, 26, 9, 37, 22, 13, 46,
45, 29, 6, 40, 19, 16, 30, 5, 41, 33, 2, 44, 47, 27, 8, 38]
49 average vertex degree: 3.061224489795918
[1, 34, 12, 23, 36, 10, 25, 21, 38, 8, 27, 47, 44, 2, 33, 41, 5, 30, 16, 19, 40, 6, 29, 17, 18, 28, 7, 39, 35, 24, 11, 48,
43, 3, 32, 14, 45, 46, 13, 22, 37, 9, 26, 20, 15, 31, 4, 42, 49]
50 average vertex degree: 3.12
[1, 34, 12, 23, 36, 10, 25, 21, 38, 8, 27, 47, 44, 2, 33, 13, 46, 45, 14, 32, 3, 43, 48, 11, 35, 39, 7, 28, 18, 17, 29, 6,
40, 19, 16, 30, 5, 41, 50, 24, 22, 37, 9, 26, 20, 15, 31, 4, 42, 49]
$[/CODE]
```

So solutions with c=10 for n=45..50.
I have not found a function significantly sparser than squares allowing for a solution.

Testing for n^2+c is small diff to Python code from previous posting, allowing for your experiments:

```
[CODE]$ diff nsum.py nsum2.py
13c13
< def is_square(i):
---
> def is_square2(i):
15c15,16
< return i == isqrt(i) ** 2
---
> j = isqrt(i)
> return i == j ** 2 + 10
21c22
< if is_square(sq):
```

---
> if is_square2(sq):
$[/CODE]

P.S:
The is_tetra() function I used:
[CODE]from math import floor

def is_tetra(i):
j = floor((6*i) ** (1/3.0))
return i == j * (j+1) * (j+2) / 6
[/CODE]

## Azruine

2023-10-02 20:08

1 Attachment(s)
[QUOTE=R. Gerbicz;638883]Even if your method is good with the gluing you still need that the base pair of sequences are good, so:
For every x<=n it is in the same parity position for the two sequences.

Track also how many numbers you have already placed, need only the parity. When you place a single number, or n or n+1 numbers you will know how the parity is changing (for this you need only parity of n).
With the above parity you can assure that you are not placing down the same remainder residue twice, just see the first number of the placed sequence, since with the parity trick every number will be good or wrong.

About search: it was a bactracking algorithm, basically without any tricks, but since it was run in 1 second there was no need to optimize it. Even this is true, to find the base sequences.[/QUOTE]

Thank you for your kind answer!

I have one more question: How did you find glues? I'm currently using backtracking since I need to find them only one time, but it takes forever I guess.

tbh, I'm currently working on this problem becuase this problem is in the problem solving website I'm using, and they only accept code less than 512KB, or 524288bytes.

The goal is find out hamiltonian path and I've almost complete the work.
Found good pairs of sequences from 41 to 1036, which starts with 1 and end with 2(even) or 3(odd), and 1037 = 25*41+12. Also found recursive formula for expending. For n = 1E7, only takes 0.6 sec now.

But size of the code exceeded 1MB, I have to find good sequence pairs less than 41 to reduce precomputed sequences.

So I find some good? sequences below 41:
33-34
1,8,28,21,15,10,26,23,13,3,6,30,19,17,32,4,12,24,25,11,5,20,29,7,18,31,33,16,9,27,22,14,2,
1,8,28,21,15,34,2,14,22,27,9,16,33,31,18,7,29,20,5,11,25,24,12,4,32,17,19,30,6,10,26,23,13,3,

35-36
1,3,13,23,26,10,6,19,30,34,15,21,28,8,17,32,4,12,24,25,11,5,20,29,35,14,22,27,9,16,33,31,18,7,2,
1,8,17,32,4,5,11,25,24,12,13,36,28,21,15,34,30,19,6,10,26,23,2,14,35,29,20,16,33,31,18,7,9,27,22,3,

36-37
1,8,17,32,4,5,11,25,24,12,13,36,28,21,15,34,30,19,6,10,26,23,2,14,35,29,20,16,33,31,18,7,9,27,22,3,
1,8,17,32,4,5,11,25,24,12,37,27,9,7,18,31,33,16,20,29,35,14,22,3,13,36,28,21,15,34,30,19,6,10,26,23,2,

39-40
1,3,22,14,35,29,20,5,11,38,26,23,13,36,28,8,17,32,4,21,15,34,30,19,6,10,39,25,24,12,37,27,9,16,33,31,18,7,2,
1,8,17,32,4,5,11,38,26,23,2,14,35,29,20,16,33,31,18,7,9,40,24,25,39,10,6,19,30,34,15,21,28,36,13,12,37,27,22,3,

40-41
1,8,17,32,4,5,11,38,26,23,2,14,35,29,20,16,33,31,18,7,9,40,24,25,39,10,6,19,30,34,15,21,28,36,13,12,37,27,22,3,
1,3,6,19,30,34,15,10,39,25,24,12,37,27,22,14,35,29,20,5,11,38,26,23,13,36,28,21,4,32,17,8,41,40,9,16,33,31,18,7,2,

The pairs have same parity, and end with 3 for even and 2 for odd numbers.
Yes, I need another recursive formula ;( And backtracking algorithm just stuck at res = 12 for 15 hours. I'm pretty sure about the code, because I found recursive formula for 41~ using the same code.
I don't know whether the recursive formula is exist or not for odd end 2, even end 3...

So, back to the original question: How did you find glues in 1 second?
Really need help :(

edit: Find mistake in my code, initializing step for pair sequence was wrong. This should happen while I fixed initialization step. I was kinda idiot. Now running fixed code and hope this would help...

## R. Gerbicz

2023-10-03 19:16

[QUOTE=Azruine;639644]Thank you for your kind answer!

I have one more question: How did you find glues? I'm currently using backtracking since I need to find them only one time, but it takes forever I guess.

tbh, I'm currently working on this problem becuase this problem is in the problem solving website I'm using, and they only accept code less than 512KB, or 524288bytes.

...
So, back to the original question: How did you find glues in 1 second?
Really need help :(
[/QUOTE]

Checked my original post, it says that only the glues are found in 1 sec, the all base sequences only in one hour. That really makes sense (without major tricks).

First: only the glue table is pretty small, so calculating only once is fine and writing into the code.
For gluing you could try always place down first sequence, so place only number if you can not place sequence.
Or after fixed number of iterations just completely restart the algorithm from a random (good) initial placing. Notice that to find a glue is almost a Hamiltonian path problem with not that many points. It should not take that long time.

For Hamiltonian path the 5^2=25 is enough for modulus, so we don't need the larger 49 modulus (for Hamiltonian cycle).

As for search when you need sequence pair generate first S0, and then S1. On each position try the numbers in decreasing order, I think this gives faster times, since then at the end you have smaller numbers, so larger probability to get squares.

Another optimization: check quickly if you can not get Hamiltonian path: say if you still have not put x and for all k you have already put the x+-k^2 (and this is not in the last position), then you can not place down x, so you can backtrack. Do this quickly. Don't know if I have written this in my code.

Another idea: calculate the base sequence only if your program needs for a given input. Of course if you'd need the same pair of sequence then just lookup for the calculated pairs.

## solesiesonic

2023-10-08 20:54

1 Attachment(s)
[QUOTE=R. Gerbicz;639691]Checked my original post, it says that only the glues are found in 1 sec, the all base sequences only in one hour. That really makes sense (without major tricks).

First: only the glue table is pretty small, so calculating only once is fine and writing into the code.
For gluing you could try always place down first sequence, so place only number if you can not place sequence.
Or after fixed number of iterations just completely restart the algorithm from a random (good) initial placing. Notice that to find a glue is almost a Hamiltonian path problem with not that many points. It should not take that long time.

For Hamiltonian path the 5^2=25 is enough for modulus, so we don't need the larger 49 modulus (for Hamiltonian cycle).

As for search when you need sequence pair generate first S0, and then S1. On each position try the numbers in decreasing order, I think this gives faster times, since then at the end you have smaller numbers, so larger probability to get squares.

Another optimization: check quickly if you can not get Hamiltonian path: say if you still have not put x and for all k you have already put the x+-k^2 (and this is not in the last position), then you can not place down x, so you can backtrack. Do this quickly. Don't know if I have written this in my code.

Another idea: calculate the base sequence only if your program needs for a given input. Of course if you'd need the same pair of sequence then just lookup for the calculated pairs.[/QUOTE]

I have a question about the method you used to find the base sequence.

I'm trying to use 25 * n + res, so I need to find 1036 nice pairs (a [1] = 1 and for even n: a [n] = 2, for odd n: a [n] = 3).

I have applied backtracking algorithm, but my code takes over an hour to find a nice pair of only 70 lengths.
(It seems that the slowness in my code is likely due to the sparse graph with few edges when generating seq1 after obtaining seq0)

This speed is much slower compared to your achievement of finding 2000 nice pairs in just one hour with a single thread.

Can you tell me if you used any special heuristic techniques?

## R. Gerbicz

2023-10-11 20:51

You can do the steps in the backtracking method in only O(L) steps where L is the number of not placed numbers, that is update the graph and revert the update if you need it.

Triviality: the degree should be at least 2 for the non-visited numbers (except for the last, fixed number), since every inner number has two neighbors. Previously I have written 1...

Also: if you have found S0, then on S1 you know the parity positions of each number, that means also that not every two numbers can be neighbor, only if their parity position is different. This basically halves the degrees if you have not used it.

## R. Gerbicz

2023-10-14 20:06

There is an even faster and easier method to get the Hamiltonian path/cycle for the sequence pairs.
Known technique:
Consider the complete(!) graph, and let the edge's cost to be 1 if the difference is square number, otherwise zero. On this graph any maximum cost Hamiltonian path has cost of n-1 (if the sequence pair S0/S1 exists).
Just apply any existing idea to find a Hamiltonian path, say the Lin-Kernighan heuristic ([url]https://en.wikipedia.org/wiki/Lin%E2%80%93Kernighan_heuristic[/url]), the 2-opt is enough ( [url]https://en.wikipedia.org/wiki/2-opt[/url] ).

Start from any (random) path, where the first and last vertex is fixed. And do 2-opts, make the change if the cost would not decrease (so make the change also when it remains the same). Stop when you reach the n-1 as cost.

For two opts you can not use the first and last vertex.
And for the second sequence, you should also maintain the parity positions, so fewer 2-opts are valid.

ps. 2-opts should be enough here. You could restart the search from another random n points for S0/S1 if you'd find nothing (for smallish n). In the past when coded this for other problem it was really efficient (using also larger 4-6.. opts).

## Younghwan Lee

2024-01-10 12:53

Very nice! What a good content.

I have some question. From your C code, I see that data shows the sequence such that makes length n to 49n+24~72.

But, I think there are some problems.

For example,
{{49,24,0,0},{1,0,0},{-7,-1,0},{-10,1,0},{10,-1,0},{22,0,0},{3,0,0},{-5,-1,0},{-12,1,0},{12,-1,0},{20,0,0},
{16,0,0},{9,0,0},{7,0,0},{2,0,0},{-6,-1,0},{21,0,0},{11,1,0},{-11,-1,0},{13,1,0},{-13,-1,0},{-21,1,0},{22,1,0},
{-22,-1,0},{24,1,0},{-17,1,0},{17,-1,0},{15,0,0},{7,-1,0},{-24,1,0},{19,1,0},{-19,-1,0},{-15,1,0},{15,-1,0},
{17,0,0},{-2,1,0},{6,0,0},{19,0,0},{-4,1,0},{4,-1,0},{11,0,0},{5,0,0},{4,0,0},{12,0,0},{24,0,0},{8,1,0},
{-8,-1,0},{-9,1,0},{9,-1,0},{23,0,0},{13,0,0},{5,-1,0},{10,0,0},{2,-1,0},{21,1,0},{-20,1,0},{20,-1,0},{-23,-1,0},
{6,1,0},{14,0,0},{1,1,0},{-1,-1,0},{3,1,0},{-3,-1,0},{18,0,0},{14,1,0},{-14,-1,0},{16,1,0},{-16,-1,0},{18,1,0},

{-18,-1,0},{23,-1,0},{0,1,0},{8,0,0}}
makes n to 49n+24, for even n.

Almost of the pairs sum ups as square number. But, for example {3,0,0}+{-5,-1,0} and {13,0,0}+{5,-1,0} does not makes square number.
I think there is some problems for {?,0,0}+{?,-1,?} which means reverse something after just one number(1~24).

Is there anything I'm misunderstanding?
Sorry for my bad English.

## Younghwan Lee

2024-01-11 02:14

[QUOTE=R. Gerbicz;478035]There is a solution for every n>=25 !
Giving a Hamiltonian cycle for all n>=32, see my linked code:[/QUOTE]

I mean the C code from this thread.

All times are UTC -6. The time now is 10:39.