Week 3

Week 3

SUBFORUMS

All

Assignment: Programming Assignment 3

Problem: Assign Frequencies to the Cells of a GSM Network

Alexander S. Kulikov Instructor Week 3 · 4 years ago

Please use this thread to discuss Assign Frequencies to the Cells of a GSM Network problem (make sure to review forum rules before posting).

0 Upvotes Reply Follow this discussion

Earliest **Top** **Most Recent**

Ahmad Bashar Eter · 3 years ago

I've been thinking the whole day about a problem in my implementation.

I'm getting WA on #3 test case which have 300 vertex. My implementation is based on this paper http://research.ijcaonline.org/encc/number1/encc004.pdf as follows:

I've defined 3 literal for each vertex formal ***Xuc*** where 1<=u<=n is a vertex and 1<=c<=3 is a color.

Then for each vertex ***u*** I've add clause of **exactly_one_of** each color of it.

Next for each color ***c*** I've added clauses that define **at_most_one** for each vertex and its adjacent is true.

The problem is that the function **at_most_one** is quadratic in the input size. It works by adding a clause for each pair of literal and with this if we have 500 vertex and one of them is connected to all other vertex then we will get 500*500/2 clause (T_T). So what could I do about this. Are there any hints for good or o(n) output **at_most_one** implementation?

Thanks in advance.

3 Upvotes Hide 3 Replies

Ahmad Bashar Eter · 3 years ago

I've solve it that was too easy lol.

One hint in order to not fall in my mistake. Don't view the constraint of adjacent vertices as a constraint for each set of adjacent vertices for each vertex. View it as a constraint on each edge to have its vertices had different color.

Good job! (Max time used: 0.05/5.00, max memory used: 9023488/1073741824.)

3 Upvotes

Daniel Cheang · 2 years ago

Thank you for the hint!!! I have passed as well. I initially did what you mentioned in the original post. And of course it failed at case #3 too.

As you said, there is no need to make an adjacency list and use **exactly_one_of** on the columns. Focus on the edges instead. So simple, yet effective.

0 Upvotes

БА **Буров Арсений** · a year ago

An easy Fix for this problem is to connect only the first vertex to the second one. That way we will only consider an edge (v1, v2) while (v2, v1) will not exist in our graph.

It won't cause any trouble since imposing a restriction on (v1, v2) is enough.

0 Upvotes

Reply

Reply

JW **JICHEN WU** · 3 years ago · Edited

Can someone help me on this problem? @instructor

I have been stuck on test case 6 for a while. The grader always ouput me a bunch of number(Warning, long feedback: only the beginning and the end of the feedback message is shown, and the middle was replaced by " ... ". Failed case #6/26: (Wrong answer). then said correct output is unsatisfactory.

Is there something wrong with my method? I cannot figure out why I cannot pass the grader

0 Upvotes Hide 2 Replies

JW **JICHEN WU** · 3 years ago

just solve the question.

This case edges does not include all vertex so some vertex are isolated to others.

A hint for people who might have the same problem. Loop through vertex first then loop through edges. My mistakes was to loop through edges and made a set to record if the vertex appear before. If it does not appear, then add constraint but this method cannot loop through all vertexes.

1 Upvote

RM **Rafael Marino** · 3 years ago

Thanks for posting this Jichen. I has making the same mistake.

0 Upvotes

Reply

Reply

J. Andrew Howe, PhD · 3 years ago

Anyone here using python3, I suggest not using the

```
1  os.system("minisat -verb=0 tmp.cnf tmp.sat")
```

call from the sudoku solver. At least on my Ubuntu system, that forced

```
1  WARNING: for repeatability, setting FPU to use double precision
2  SATISFIABLE
```

to be output also, even when I set the minisat option -verb=0. Instead, call minisat from python like this:

```
1  from subprocess import DEVNULL, STDOUT, check_call, call
2  call(['minisat','tmp.cnf','tmp.sat'], stdout=DEVNULL, stderr=STDOUT)
```

Passed with: Good job! (Max time used: 0.10/5.00, max memory used: 8900608/1073741824.)

0 Upvotes Reply

Jeremy Chan · 3 years ago

I spent unnecessary time on figuring out why my program could not even passes the first test case. I was using minisat to test and it does say SATISFIABLE but not the grader. Turned out I can only output ONE clause per line.

If your formula is (x1 OR x2) AND (x3 OR x4), you should output two lines as

```
1  1 2 0 2 3 4
```

0 Upvotes Reply

ZL **Zhaorui Liu** · 4 years ago

I am failing the case when n=300 and m=1000, the variable naming I use is 3*(j-1)+i. I don't know what's wrong with my method because I am able to pass assignment 2 with similar scheme.

0 Upvotes Reply

J **Julian** · 4 years ago

One of the constraints in the problem description is inaccurate.

$1 \leq n \leq 500; 1 \leq m \leq 1000$ where n and m are number of vertices and edges, respectively.

It's stated that no vertex is connected to itself by an edge. By the constraint on m, there's at least 1 edge. So it must be the case that $2 \leq n \leq 500;$

0 Upvotes Hide 1 Reply

Alexander S. Kulikov Instructor · 4 years ago

Julian, this just means that there is no graph with a single vertex in the tests (because, as you noticed correctly, such a graph has to contain at least one edge, but it is a self-loop then and this is forbidden by the problem statement).

3 Upvotes

Reply

Reply

Kush Shukla · 4 years ago

I have read all the comments here, but still stuck at test # 3. Here is my approach :

1. Boolean variable x(i,j) 1<=i<=n, 1<=j<=3

2. Exactly 1 true in rows => exactly_one_of(x(1,1), x(1,2), x(1,3)) similarly for other vertices.

This makes the clauses, (x(1,1) V x(1,2) V x(1,3)) (-x(1,1) V -x(1,2)) (-x(1,2) V -x(1,3)) (-x(1,3) V -x(1,1))

3. vertices connected by an edge must have different colors=> If node l is connected m, then clause would be (-l V -m)

4. Based on the above comments used variable scheme i + (3*j), where i is the vertex number and j is the color number.

For the Case-1, I am getting 21 clauses with 9 variables, and this passes, but not sure why is it not working for test case # 3 with 300 nodes and 1000 edges.

Please help

0 Upvotes Hide 1 Reply

DT **Daniel Trebbien** · 4 years ago

Hello Kush,

The variable numbering scheme, i + (3*j), is not correct. For example, x(1, 2) and x(4, 1) both map to 7.

Daniel

1 Upvote

Reply

Reply

Avi Tevet · 4 years ago

Hi, I'm having a problem where minisat produces the correct result, and the grader thinks the program produces an incorrect result. When I replace the first line with "p cnf 9 21" and pass the output to minisat, minisat produces "SATISFIABLE." Note that in the real output, there are clauses where the "..." appears. So I'm not sure what I'm doing wrong here.

```
Failed case #1/26: (Wrong answer)

Input:
3 3
1 2
2 3
1 3

Your output:
21 9
11 12 13 0
...
-13 -33 0

Correct output:
SATISFIABLE
```

1 Upvote Hide 4 Replies

Avi Tevet · 4 years ago

By changing the naming scheme to the one that Luis used, I was able to pass all tests.

0 Upvotes

MA **michael samy abdallah** · 4 years ago

i have the same problem. would you give me a hint ?

0 Upvotes

André Garcia · 3 years ago · Edited

I also have trouble, it says my output is incorrect in the first test case. I tested it locally using minisat and it says it is SATISFIABLE. But it fails on coursera.

Should i output 'SATISFIABLE' and 'UNSATISFIABLE' instead of the clauses in cnf form?

I need help, please.

3 Upvotes

刘 **刘嘉熹** · 2 years ago

That's the problem of the name of the variables. My name method is the same as yours, 11 12 13, 21 22 23, 31 32 33. So the SAT solver will find the maximum value of the variables, which is 33, but actually the number

is 9. So there will be something wrong. Just rename them start from 1 consistently, just like 1, 2, 3 ... 8, 9. and print the result as the pdf says. Needless to use console to run minisat by ourselves.

Spent hours on this. Hope will help anyone in the future.

Reply 0 Upvotes

Reply

SS **Sabina Sloman** · 4 years ago

Hi,

I'm totally stuck on test case #6. I've tried my best to implement the algorithm as described and as discussed on this thread.

My version of minisat calculates the number of variables as the max number passed, not as the number of different variables actually passed. For example, for this input:

2 1

1 1000

I calculate the number of variables as 6 (3 * 2), but minisat tells me it's 1004 (the number assigned to the largest variable). Because of this, I put in some assertions in my code to to check my input data is continuous so it's consistent with what I think minisat expects. But I'm getting assertion errors when submitting to the grader.

Is it a correct assumption that for n variables, there will be edges that include nodes in the range 1 to n, and would my algorithm fail if it outputted the number of variables incorrectly?

Thanks, everyone!

0 Upvotes Reply

Pradyumn Agrawal · 4 years ago

@Instructor Sir, please suggest me what the mistake am I making in my algo because I have stress tested the algo a lot and till now I am not getting any bug.

I have got stuck in the problem and it's over than a week and till now all attempts proved unsuccessful. @Instructor and anyone else please help me.

Thanking You in advance.

I used the approach of the following research paper to understand the concept :

http://research.ijcaonline.org/encc/number1/encc004.pdf

0 Upvotes Hide 2 Replies

Pradyumn Agrawal · 4 years ago

@Instructor Sir, please suggest me what the mistake am I making in my algo because I have stress tested the algo a lot and till now I am not getting any bug.

I have got stuck in the problem and it's over than a week and till now all attempts proved unsuccessful. @Instructor and anyone else please help me.

Thanking You in advance.

I used the approach of the following research paper to understand the concept :

http://research.ijcaonline.org/encc/number1/encc004.pdf

0 Upvotes

Alexander S. Kulikov Instructor · 4 years ago

Pradyumn, could you describe your approach to solve the problem and the way you've tested your solution?

Also, have you read other posts at this thread?

0 Upvotes

Reply

Reply

Alexander S. Kulikov Instructor · 4 years ago

Kyrylo, the second approach make sense, of course, so I must admit that I don't know why it doesn't pass. Do you always use ExactlyOneOf for three literals? (If you call it for k literals, it will produce about 2^k clauses.)

1 Upvote Hide 3 Replies

Kyrylo Kolodiazhnyi · 4 years ago

I just hardcoded formula for three literals, and used it for constraint "each vertex has to be colored by one of the three colors", for adjacent vertices I used another formula. My program failed 8th test when I used second expression. For tests I created myself program with second expression works correctly. May be it's possible to share 8th tests source to debug the problem?

0 Upvotes

Alexander S. Kulikov Instructor · 4 years ago

What is the maximum number of clauses in your reduction?

0 Upvotes

Kyrylo Kolodiazhnyi · 4 years ago

if n is the number of vertices and m is the number of edges then total number of clauses in my program will be = n*5+m*3. So for 8th test with 500 variables and 1000 edges number of clauses = 5500. And this number is **greater then limit** in the task. I've missed this moment. Also grader showed that the correct answer is different from the my one, and not that limit exceeded. Thanks.

0 Upvotes

Reply

Reply

Kyrylo Kolodiazhnyi · 4 years ago

I have a question - for 3 variable A,B,C we can make two expressions for implementation of ExactlyOneOf function:

1.(A|B|C)&(~A|~B)&(~A|~C)&(~B|~C) - as from lectures

2.(A|B|C)&(A|~B|~C)&(~A|B|~C)&(~A|~B|C)&(~A|~B|~C) - I made from the truth table

They have equivalent truth tables for all possible values of A, B, C but I was able to pass task only with the first one. Could someone say me where is the difference or error?

2 Upvotes Reply

Christine Miyachi · 4 years ago

I am able to write an algorithm that color codes the cells correctly but I don't know how to convert that into a CNF. It seems almost too easy - if I can find a sequence of color codes, then I just create a CNF that is satisfiable, otherwise I create one that is not? Regardless of the number of variables? I'm sure I have something wrong here (but I haven't tried it on the grader yet).

1 Upvote Hide 2 Replies

Kota Mori · 4 years ago

If you have your own algorithm to figure out the cell coloring, then you can use that. If coloring is possible, then print the answer to the sample question 1, otherwise print the answer to the sample question 2.

0 Upvotes

Luis Alberto Herrera Gomez · 4 years ago

You have to convert the 3-coloring problem to its equivalent SAT problem like the instructor did with the sudoku problem(there are many hints in the previous answers). Then you can use a minisat solver to test your solutions.

0 Upvotes

Reply

Reply

AE **Aaron Elmquist** · 4 years ago

I have a question on the SAT solver. Do the variables used need to be sequential?

I ask this because I am considering using the cantor function and its inverse to create and decode unique id's for combinations of location and color.

If this approach works with the solver, would I return the number of variables created, or the maximum value of the ids?

I suppose this is the same as just multiplying by 10 and adding a color id of 1, 2, or 3, as I would always be missing digits 0 and 4 .. 9.

0 Upvotes Hide 1 Reply

Alexander S. Kulikov Instructor · 4 years ago

Aaron, no, they don't have to be sequential.

0 Upvotes

Reply

Reply

KB **Kyle Banks** · 4 years ago

So how are you supposed to stress test these assignments? I can pass them into picosat just fine, but what am I supposed to validate against?

I have a program that passes the first 2, but since the 3rd has so many inputs, I can't even see that one to try and troubleshoot it.

My basic plan is create a set of 3 variables for each nodes with 'only one of clauses', then for each color and each adj node, create a set of 'only one of' clauses.

Am I overcomplicating things?

0 Upvotes Hide 5 Replies

DT **Daniel Trebbien** · 4 years ago

Hello Kyle,

I am not sure what you mean by "for each color and each adj node, create a set of 'only one of' clauses."

Suppose we denote the 3 boolean variables for each node as the hint suggests: x_{ij}, where i is the node number and j is the color, $1 \leq j \leq 3$.

Now suppose that nodes i and p are adjacent (they are connected by an edge). It would **not** be correct to add $\mathrm{ExactlyOneOf}(x_{i1}, x_{p1})$, $\mathrm{ExactlyOneOf}(x_{i2}, x_{p2})$, and $\mathrm{ExactlyOneOf}(x_{i3}, x_{p3})$ as clauses; because, for example, if a valid solution colors node i with color 1 and node p with color 2, then $\mathrm{ExactlyOneOf}(x_{i3}, x_{p3})$ is false.

Daniel

0 Upvotes

KB **Kyle Banks** · 4 years ago

yeah, I changed it to just create negations for the adjacent nodes and it finally passed.

0 Upvotes

Vivekanand Ganapathy Nagarajan · 4 years ago

As Daniel pointed out, I am also having difficulty understanding the second part of the statement regarding vertices connected by an edge have to have different color. In this case, we cannot use ExactlyOne, since there is a possible assignment of none of the colors being used.

Any insights, Kyle Banks.

Thanks,

0 Upvotes

DT **Daniel Trebbien** · 4 years ago

Here's a hint, Vivekenand: x_{i1} and x_{p1} cannot both be true (because in that case, adjacent vertices i and p would both be colored with color 1 and we can't have that). Another way of phrasing this is that one or both of x_{i1}, x_{p1} must be false: $(\bar{x}_{i1} \vee \bar{x}_{p1})$

Daniel

2 Upvotes

Vivekanand Ganapathy Nagarajan · 4 years ago

Thank you so much for the hint Daniel.

My solution passed the grader. I was going in circles thinking about how adjacent vertices must either be true for one or false for both if they both are of the same color.

0 Upvotes

Reply

Reply

Reply

Reply