Yash Avinash Patole

Course: CS600

Homework 8

CWID: 10460520

Chapter 17

Exercise 17.8.8

Answer:

To convert a clause with fewer than three literals, we shall add free variables until the clause has three literals.

And if the phrase contains more than three literals, we will divide it into two or more pieces so that each clause contains three literals while adding one or more free variables and the satisfiability of the original and new clauses is equivalent.

And if the clause has fewer than three literals, we will add a free variable to make it a clause of three literals and create an additional clause such that the satisfiability of the product of clause is solely dependent on the satisfiability of the original clause.

So, in the given CNF formula B, the first clause is (x_1) having one literals, so we add two free variables y_1 and y_2 to make it,

$(x_1 \vee y_1 \vee y_2) \wedge (x_1 \vee y_1 \vee \bar{y_2}) \wedge (x_1 \vee \bar{y_1} \vee y_2) \wedge (x_1 \vee \bar{y_1} \vee \bar{y_2})$.

So, if (x_1) is satisfiable then,

$(x_1 \vee y_1 \vee y_2) \wedge (x_1 \vee y_1 \vee \bar{y_2}) \wedge (x_1 \vee \bar{y_1} \vee y_2) \wedge (x_1 \vee \bar{y_1} \vee \bar{y_2})$ is satisfiable and vice-versa.

B = (x1) · (x2 + x3 + x5 + x6) · (x1 + x4) · (x3 + x5)

Let us break B and apply local replacements for each Bi in B B1 = (x1) = (x1 + ~x7 + ~x8) ·(x1 + ~x7 + x8)·(x1 + x7 + ~x8)·(x1 + x7 + x8)

B2 = (x2 + x3 + x5 + x6) = (x2 + x3 + x9) · (~x9 + x5 + x10) ·(~x10 + x6 + x11)

B3 = (x1 + x4) = (x1 + x4 + x11) · (x1 + x4 + ~x11)

B4 = (x3 + x5) = (x3 + x5 + x12) · (x3 + x5 + ~x12)

Therefore B = (x1 + ~x7 + ~x8) · (x1 + ~x7 + x8) · (x1 + x7 + ~x8) ·(x1 + x7 + x8)· (x2 + x3 + x9)·(~x9 + x5 + x10)·(~x10 + x6 + x11)·(x1 + x4 + x11)· (x1 + x4 + ~x11)·(x3 + x5 + x12)·(x3 + x5 + ~x12.

Exercise 17.8.12

Professor Amongus has just designed an algorithm that can take any graph G with n vertices and determine in O(nk) time whether G contains a clique of size k. Does Professor Amongus deserve the Turing Award for having just shown that P=NP? Why or why not?

Answer:

Indeed, he surely deserves that award if he has addressed the P = NP issue, as it is anything from fair. Many analysts are following it, and if he is the one who emerges by demonstrating that arrangement, it will be a tremendous accomplishment for us all.

A proof that P = NP can have amazing real-world consequences if it prompts productive strategies for dealing with some of the major issues in NP. It is also possible that a proof would not lead directly to effective strategies, for example, if the verification is ineffective or the span of the bouncing polynomial is too large to be proficient eventually. Because different NP complete issues are important in many fields, the outcomes, both positive and negative, emerge.

Cryptographic hashing as the issue of finding a pre-picture that hashes to a given esteem must be troublesome to be helpful, and in a perfect world ought to require exponential time. Be that as it may, if P=NP, at that point finding a pre-image M can be done in polynomial time, through decrease to SAT.

So also, Stephen Cook says it would change science by enabling a PC to locate a formal confirmation of any hypothesis which has a proof of a sensible length, since formal evidence can without much of a stretch be perceived in polynomial time. Model issues may well incorporate the majority of the CMI prize issues.

Exercise 17.8.28

Answer:

HYPER-COMMUNITY is in NP because a nondeterministic machine could simply guess k web pages and check that they are all connected to one another. Also, hyper-community is in NP, because a non-deterministic machine uncomplicated theory "k" website pages and ensure that they are all associated with one other.

We reduce from the autonomous set to show that hyper-networks are NP-hard. Assume we have Graph G with n vertices, and we need to find an independent collection of size k. We build G' on the same n vertices as G, where (v, w) is an edge in G' if and only if it is not an edge in G. Because we only need to iterate over all pairs of vertices, this reduction obviously takes polynomial time.

Now if there is a set of “k” mutually connected vertices in G', then they must form an independent set in G. Conversely, if there is an independent set of size k in G, then those k vertices must all be connected in G'.

Due to the fact that Independent Set decreases to Hyper-Community, Hyper-Community must be NP-Complete.

Exercise 17.8.35

Answer:

The collection of decision problems that can be solved in polynomial time on a non-deterministic Turing machine and whose answers can be verified in polynomial time are referred to as NP-complete questions.

A decision problem C is NP-complete if:

- C is in NP, and
- Every problem in NP is reducible to C in polynomial time.

By demonstrating that a candidate solution to C can be verified in polynomial time, C can be demonstrated to be in NP.

To begin, we must verify that it is an NP problem. To do so, we must have a polynomial-time verifier.

We can easily determine if the companies are non-competitive or belong to a pair of competing companies from the previous year in polynomial time. So, it is proven in NP.

We take a problem that has already been demonstrated to be NP Hard. We must demonstrate that any instance of an NP hard problem can be reduced to an instance of a DR drama problem. Inviting all firms is a difficult challenge, and not inviting non-competing companies is an example of inviting all companies (if we place restrictions in our algorithm).

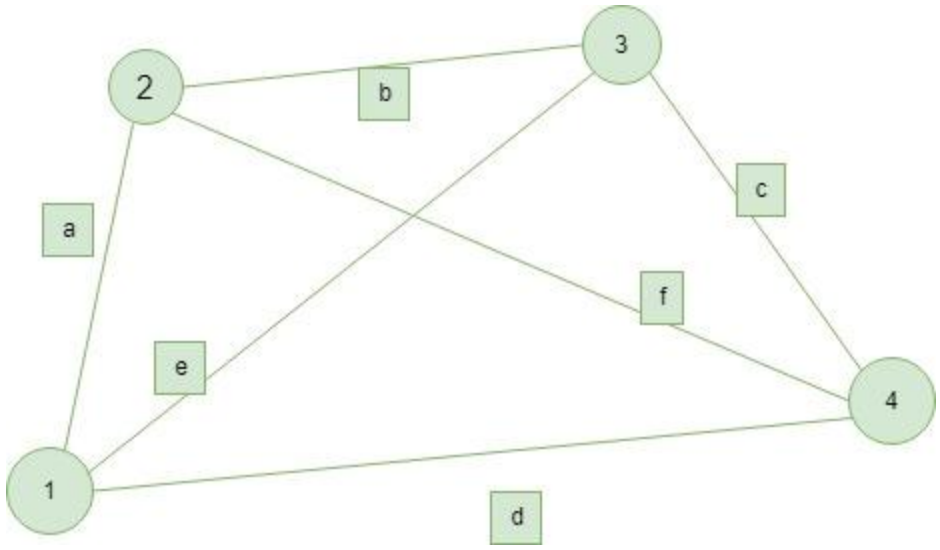
By demonstrating that a candidate solution to C can be verified in polynomial time, C can be demonstrated to be in NP. To begin, we must verify that it is an NP problem. To do so, we must have a polynomial-time verifier. We can easily determine if the companies belong to a non-competitive pair or a pair of competing companies from the previous year in polynomial time. So, it is proven in NP. Dr. Drama's problem may be reduced to a 3SAT problem in polynomial time by applying the equations (xi + yi1 + yi2 + yi3 +...+ yin), where xi is the corporation and yi is its opponent. Since, this problem is in NP and can be reduced to a problem C in polynomial time, Dr. Drama’s problem is NP complete

Chapter 18

Exercise 18.6.19

In the Euclidean traveling salesperson problem, cities are points in the plane and the distance between two cities is the Euclidean distance between the points for these cities, that is, the length of the straight line joining these points. Show that an optimal solution to the Euclidean TSP is a simple polygon, that is, a connected sequence of line segments such that no two ever cross.

Answer:



The Euclidean TSP will be as following also, according to triangle law-

a + b <= e – (1)

b + c <= f – (2)

c + d <= e – (3)

a + d <= f – (4)

Therefore, consider the TSP paths 1 – 2 – 3 – 4 – 2 and the path 2 – 3 – 1 – 4 – 2.

According to optimal path solution we will select path 2 – 3 – 1 – 4 – 2

Where edges e and f intersect only if

b + e + f + d <= a + b + c + d

e + f <= a + c

a + 2b + c <= e + f <= a + c by adding (1) and (2)

a + 2b + c <= a + c

2b <= 0

b <= 0

Which is only feasible if 2 and 3 are the same vertices or if it fails the Euclidean distance assumption that distance >= 0, both of which are not possible. As a result, a simple polygon, that is, a connected sequence of line segments such that no two ever cross, is an optimal solution to the Euclidean TSP.

Exercise 18.6.26

Answer:

Algorithm GreedyMinTruck (W, M, n):

Input: Set W of boxes, such that each box i ∈ W has a positive weight wi ; positive maximum total weight M, that is the limit of each truck can carry; and number of boxes.

Output: Minimum number of trucks t such that no truck carries more than M pounds

sum ← 0

for each box i ∈ W do

sum ← sum + wi

return sum / M + 1

Considering an example where n = 4

W = 100, 200, 300, 400 and M = 400

Therefore, 100+200+300+400/400

=1000/400+1

=3+1

=4

Now, where n = 3 while W = 200,400,400 and M = 500

Therefore, 200+400+400/500

2+1

=3

so that (200), (400), and (400) can be packed optimally in three trucks and 3 2 * 3

As a result, the algorithm always provides a number of trucks that is fewer than double the number of ideal vehicles.