MATH 333 Discrete Mathematics
Chapter 10 Computational Complexity

Jianan Lin

linj21@rpi.edu

May 24, 2022

NPC

P problem: We can solve it in polynomial time. For example, sorting.

NP problem: We can test the result in polynomial time. For example, SAT.

NPC problem: The hardest problem in NP.

If we find a polynomial algorithm for some NPC problem, then P = NP. Although most scientists don't believe.

We introduce some famous NPC problems.

NPC

SAT problem: Given a sentence, can we find the values of each variable to let it be true?

Example:

$$(p \vee q \vee a \vee b) \wedge (p \vee q \vee \neg a \vee \neg b)$$

Let's try some examples.

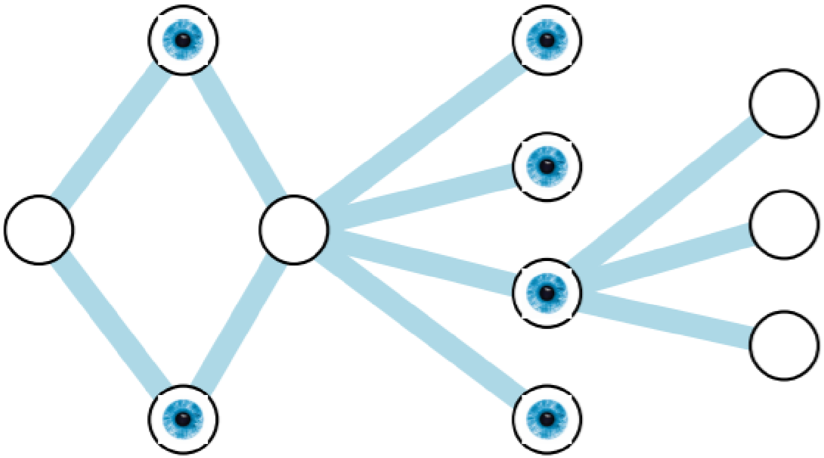
NPC

3-SAT problem: SAT problem with each clause 3 variables.

Example:

$$(p \lor q \lor r) \land (p \lor q \lor \neg r) \land (\neg p \lor q \lor r) \land (p \lor \neg q \lor \neg r)$$

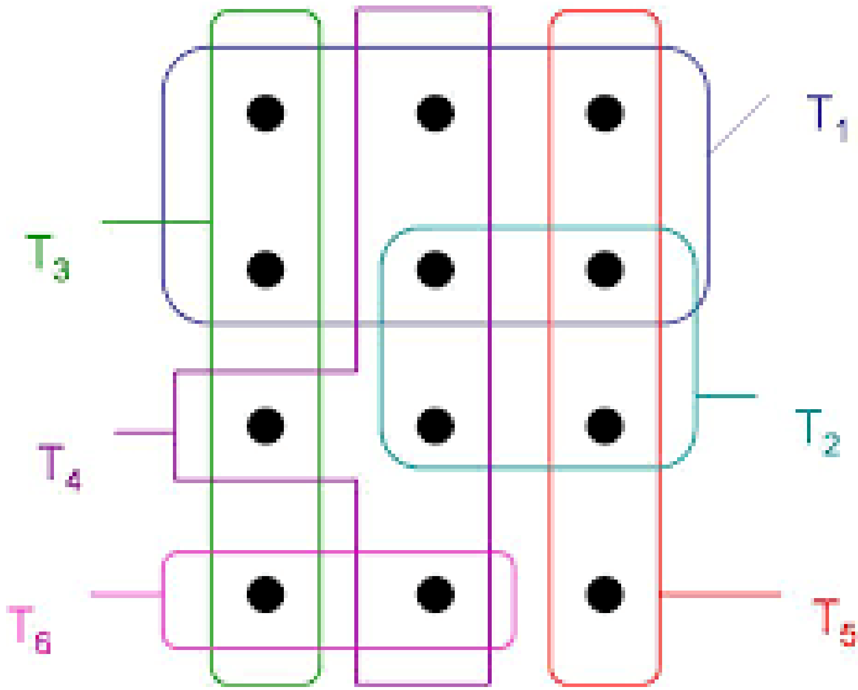
Vertex Cover: In an undirected graph, find some set S of vertices so that, each edge must have at least one end in S. Given k, can we find such a set with $|S| = k$?



A vertex cover of size 6

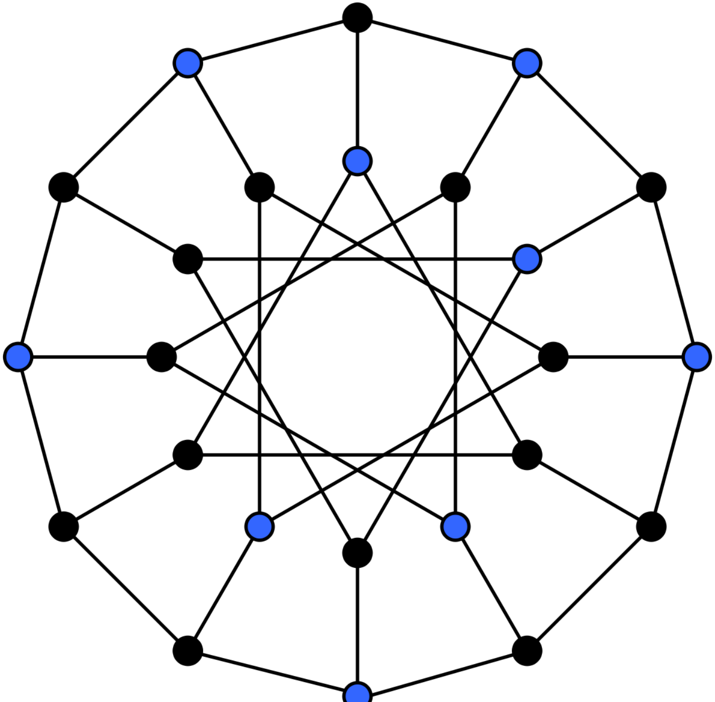
NPC

Set Cover: There are n sets and the union of all is U. Given k, can we find k sets so that the union of them is U?

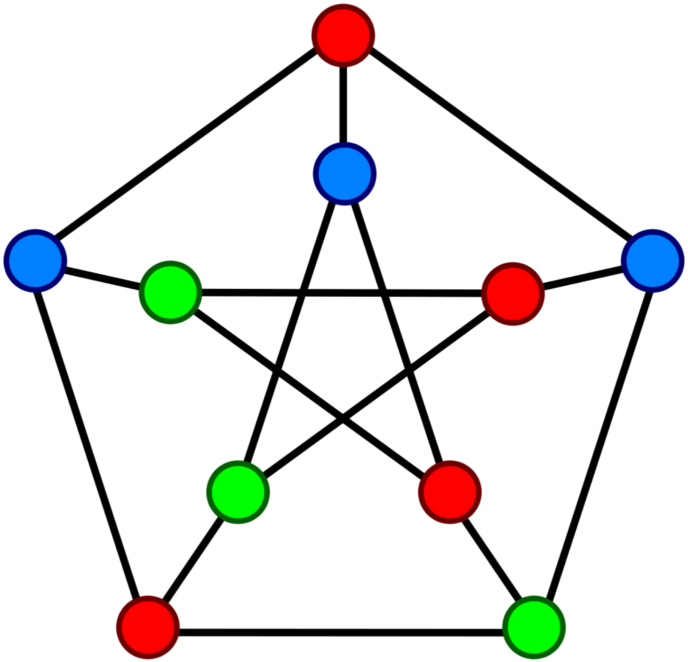


NPC

Independent Set: In an undirected graph, given k, can we find a set S of vertices so that no pair of vertices are neighbor/adjacent with $|S| = k$?



3-Coloring: In an undirected graph, can we paint it with 3 colors so that no adjacent vertices have the same color?



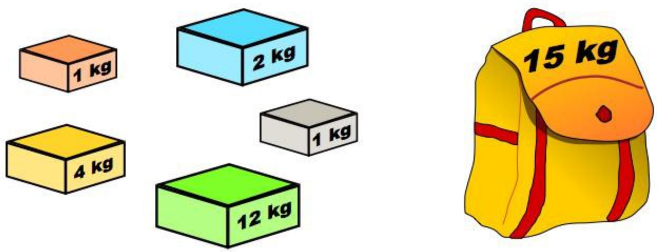
NPC

Subset Sum: Given a set of numbers and a target number T, can we find a subset so that the sum is T?

Example: Given $\{1, 3, 5, 7, 9\}$ and $t = 4$.

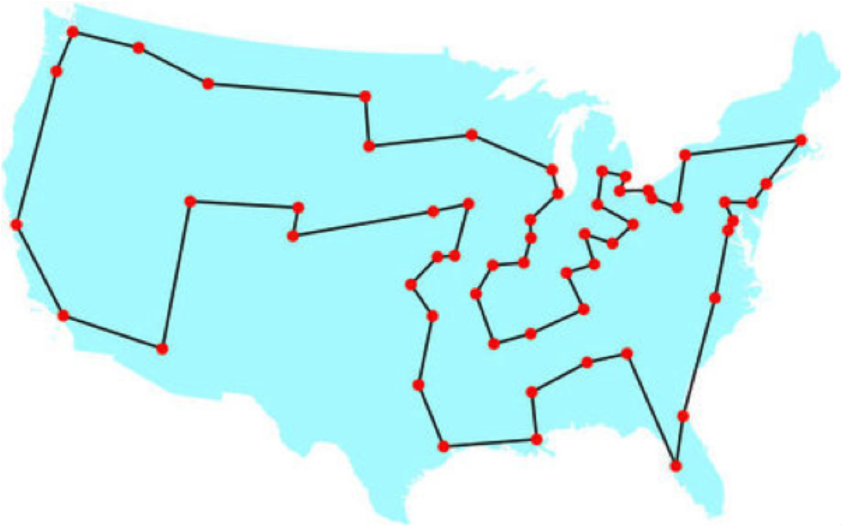
NPC

Knapsack Problem: Given some things with weight and a package with capacity c. What's the largest value we can have?



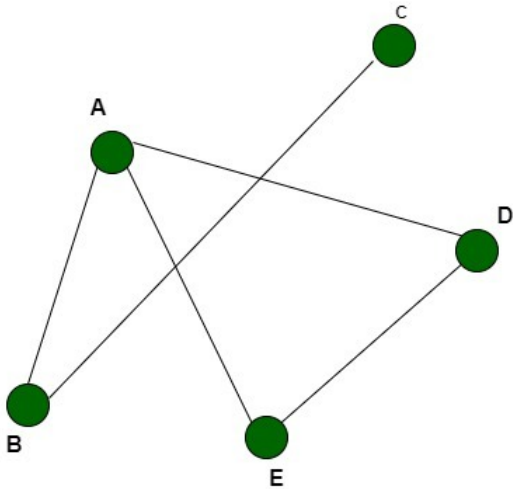
NPC

Travelling Salesman Problem/TSP: Given a graph, what is the minimum cost (sum of weight of edge) to travel all the vertices and return the start?



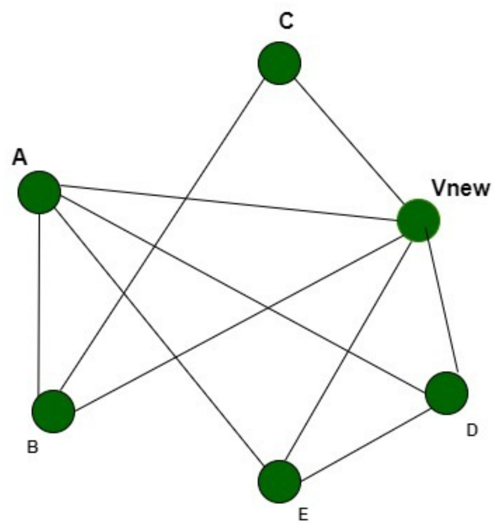
NPC

Hamiltonian Cycle and Path: Given a graph, can we find a cycle/path travelling all the vertices. Cycle and path are different but similar problems.



G ≅

Hamiltonian Parh DEABC



G' ≅

Hamiltonian Cycle DEABCVnewD

NPC

How to prove a problem is NPC?

First, prove it is NP (test in poly time). Second, prove it is at least as hard as a know NPC problem.

What is A at least as hard as B?

We have two ways.

1. B is a special case of A.

2. for any B, we can construct A in poly time. This means, B cannot be harder than A.

Problem

We define the density of a graph $G = (V, E)$:

$$d = \frac{2|E|}{|V| \cdot (|V| - 1)}$$

We have a problem: given any k and graph, can we find a sub-graph with k vertices and with density d?

Prove this is NPC.

Hint: Recall independent set. What's the relationship between them?

Problem

We define the density of a graph $G = (V, E)$:

$$d = \frac{2|E|}{|V| \cdot (|V| - 1)}$$

We have a problem: given any k and graph, can we find a sub-graph with k vertices and with density d?

Prove this is NPC.

Hint: Recall independent set. What's the relationship between them?

Answer: Let $d = 0$. Then this problem becomes the independent set problem.

Problem

We change the setting of SAT problem. This time we forbid $\neg p, \neg q$ and only use p, q..... So we can write a sentence as follows:

$$(a \vee b) \wedge (a \vee c) \wedge ...$$

Obviously, if we let all the variables be true then the sentence is true. So we set a upper bound k: no more than k variables are true. Can this be solved?

Prove that this problem is NPC.

Hint: Use vertex cover problem to construct this. Let the length of each clause be 2, i.e. $a \vee b$.

Problem

We change the setting of SAT problem. This time we forbid $\neg p, \neg q$ and only use p, q..... So we can write a sentence as follows:

$$(a \vee b) \wedge (a \vee c) \wedge ...$$

Obviously, if we let all the variables be true then the sentence is true. So we set a upper bound k: no more than k variables are true. Can this be solved?

Prove that this problem is NPC.

Hint: Use vertex cover problem to construct this. Let the length of each clause be 2, i.e. $a \vee b$.

Answer: Each vertex is a variable, each edge is a clause. So we can construct this problem from vertex cover.

Problem

We have a problem: Given an undirected graph $G = (V, E)$ and some paths $P_i = a \rightarrow b \rightarrow ....$ Can we have k paths so that no vertex is used more than once in paths.

Prove this problem is NPC.

Hint: Use independent set problem to construct this.

Problem

We have a problem: Given an undirected graph $G = (V, E)$ and some paths $P_i = a \to b \to ...$. Can we have k paths so that no vertex is used more than once in paths.

Prove this problem is NPC.

Hint: Use independent set problem to construct this.

Answer: Each vertex in independent set is a path in this. Each edge in independent set is a vertex in this. So no vertex is used in more than one path, is the same as, no edge is used twice in vertices.

PS: This is path selection problem. You can search different methods to prove this.

Problem

We have a problem: Given an undirected graph. We give each vertex a value, and this may be positive or negative or 0. Can we find a path so that: 1. No vertex is used more than once, 2. The sum of value in this path is 0?

Prove that this problem is NPC.

Hint: Consider subset sum problem.

Problem

We have a problem: Given an undirected graph. We give each vertex a value, and this may be positive or negative or 0. Can we find a path so that: 1. No vertex is used more than once, 2. The sum of value in this path is 0?

Prove that this problem is NPC.

Hint: Consider subset sum problem.

Answer: Given a subset sum problem, we let each vertex denote a number, and each two vertices have an edge. So let the target number be 0. They are the same.

Problem

We have a problem: Given n positive numbers $x_1, ..., x_n$. Given k and B, can we divide the number into k sets S_i so that the following formula holds:

$$\sum_{i=1}^{k} \left(\sum_{j \in S_i} x_j \right)^2 \leq B$$

Prove this is NPC.

Hint: We use a new problem which is NPC: Given a set of number, whether we can divide it into 2 parts so that the sum are equal.

Problem

We have a problem: Given n positive numbers $x_1, ..., x_n$. Given k and B, can we divide the number into k sets S_i so that the following formula holds:

$$\sum_{i=1}^{k} \left(\sum_{j \in S_i} x_j \right)^2 \leq B$$

Prove this is NPC.

Hint: We use a new problem which is NPC: Given a set of number, whether we can divide it into 2 parts so that the sum are equal.

Answer: Given such a problem with $a_1, ...a_n$. Let W denote the sum. Let $k = 2, B = W^2/2$ and $x_i = a_i$. So the inequality holds if and only if we can divide them into two parts with the same sum. Because each set has sum $W/2$ and the square is $W^2/4$.

Problem

We mentioned this problem above: Given n numbers $x_1, ..., x_n$, can we divide this into two parts with equal sum?

Prove this is NPC.

Hint: Use subset-sum problem.

Problem

We mentioned this problem above: Given n numbers $x_1, ..., x_n$, can we divide this into two parts with equal sum?

Prove this is NPC.

Hint: Use subset-sum problem.

Answer: In subset-sum problem, given $s_1, ..., s_n$ and target t, we assume W is the sum of s_i. Let $X = \{s_1, ..., s_n, W - 2t\}$. In this way, we can find some s_i with sum equal to t if and only if, we can divide X into two parts with equal sum.

Suppose $A = \{s_i ...\}$ and $\mathrm{sum}(A) = t$. Then the sum of $A + \{W - 2t\}$ is $W - t$, the half of X. This is called number partitioning problem.

Problem

There are n goods, m trucks and each truck has k containers. For the following questions, give a polynomial algorithm or prove it is NPC.

(a). Suppose each good can be delivered by some trucks. Is there an algorithm to judge whether we can deliver all the goods (i.e., no truck is overloaded) and (if yes) give the specific plan? Hint: Network flow

(b). Suppose each good can be delivered by any truck but some goods cannot be delivered by the same truck. Is there an algorithm to judge and give the specific plan? Hint: 3-coloring

Problem

(a). Use network flow, we can solve it in polynomial time. Construct a graph with a source node s, truck a_i and good b_j, sink node t. We connect (s, a_i) with capacity (k, and b_j, t) with capacity 1. We connect a_i and b_j with capacity 1 if truck i can carry j. Then we can satisfy the requirement if and only if the max $s - t$ flow is n.

(b). We use 3-coloring problem with graph $G = (V, E)$ to construct this problem with 3 trucks. Each truck denotes a color. Each vertex is a good and each edge means the two goods cannot be delivered by the same truck. So they are the same.

PS: there will not be so hard problems in the exam. Don't worry.