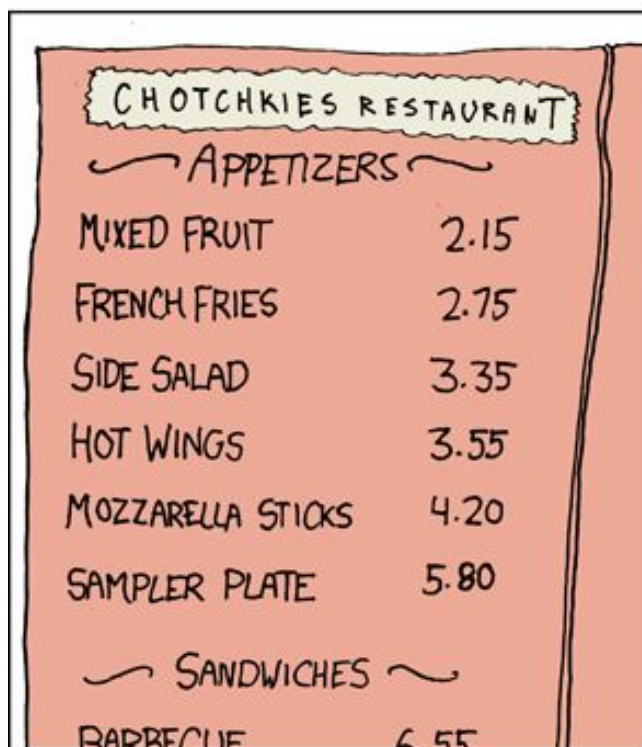


# Even More NP-Complete Problems

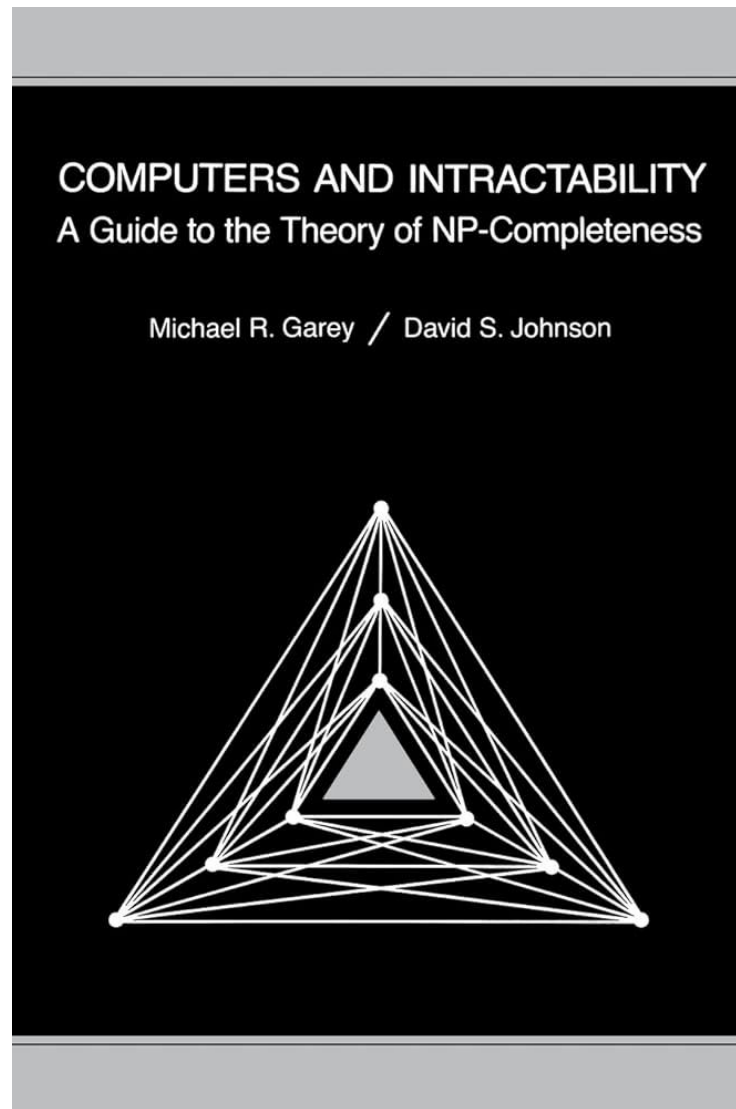
MY HOBBY:  
EMBEDDING NP-COMPLETE PROBLEMS IN RESTAURANT ORDERS



CHOTCHKIES RESTAURANT	
~ APPETIZERS ~	
MIXED FRUIT	2.15
FRENCH FRIES	2.75
SIDE SALAD	3.35
HOT WINGS	3.55
MOZZARELLA STICKS	4.20
SAMPLER PLATE	5.80
~ SANDWICHES ~	
BARBECUE	6.55



\*this is not actually the knapsack problem



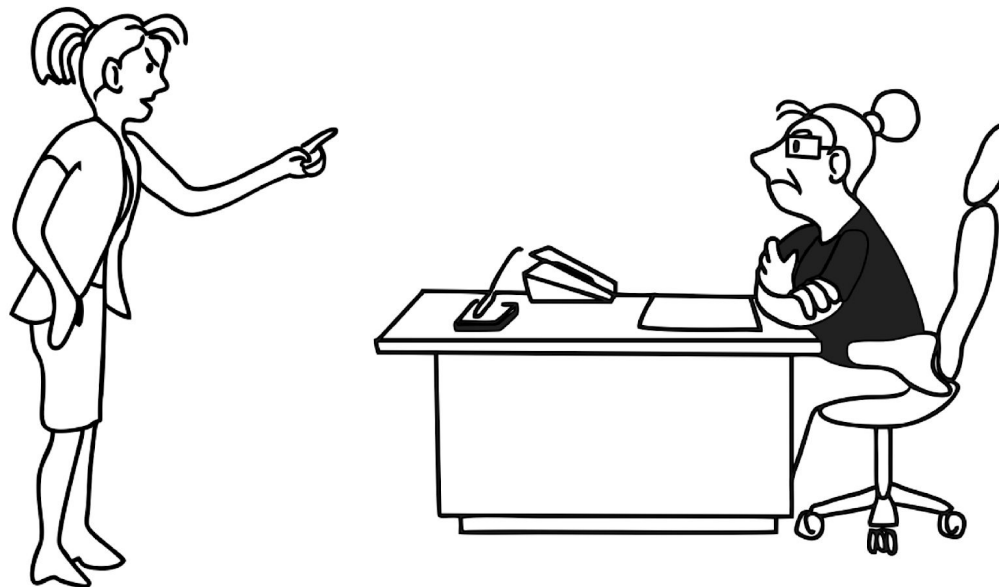
- Contains hundreds of NP-complete problems (1979)
- the most cited reference in the CS literature (as of 2006)
  - also contains the following comic...

# NP-Completeness



**Sorry! I can't find an efficient algorithm.**

# NP-Completeness



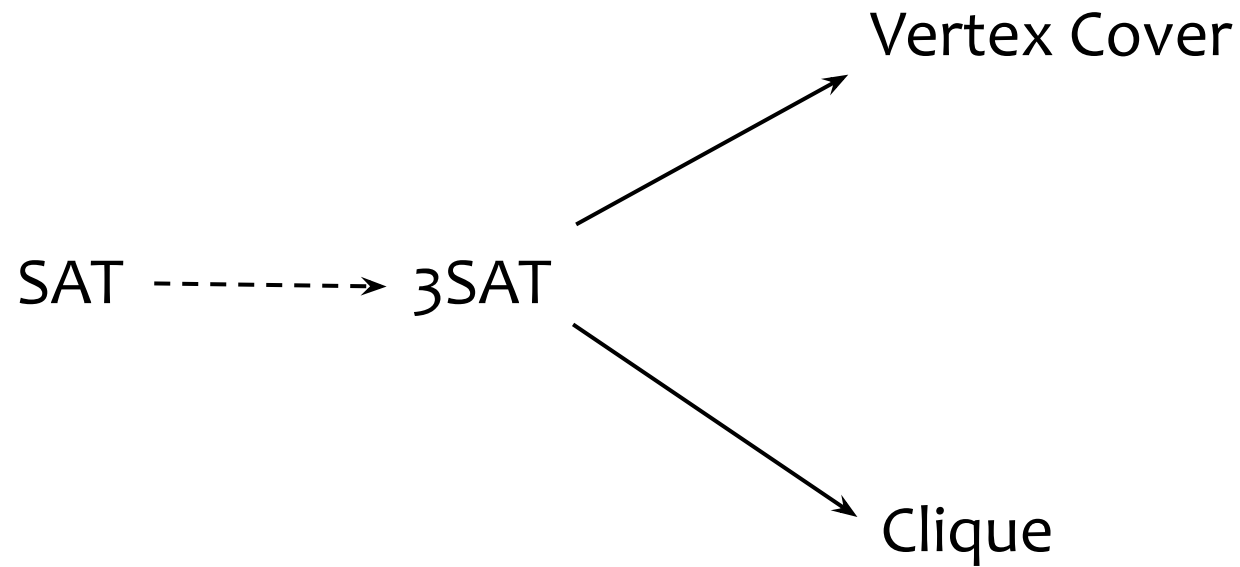
**I can't find an efficient algorithm, because no such algorithm is possible!**

# NP-Completeness



**I can't find an efficient algorithm, but neither can all these famous people!**

So far...



# Set Cover (SC)

(Contractor Problem)



## Problem Setup:

- $n$  workers, each worker has a set of skills
- Goal: hire a team of workers that together have every skill.

## Formally:

Given a collection  $U$  of elements (skills) and  $n$  subsets  $S_1, \dots, S_n \subseteq U$  (skills of each worker), a **set cover** is a group of  $S_i$ 's whose union is  $U$ .

**Set cover decision problem:** Given collection  $U$ , subsets  $S_1, \dots, S_n \subseteq U$ , and a budget  $k$ , does there exist a set cover of size  $k$  or less?

$$U = \{1, 2, \dots, 7\}$$

$$S_1 = \{1, 2, 3\}$$

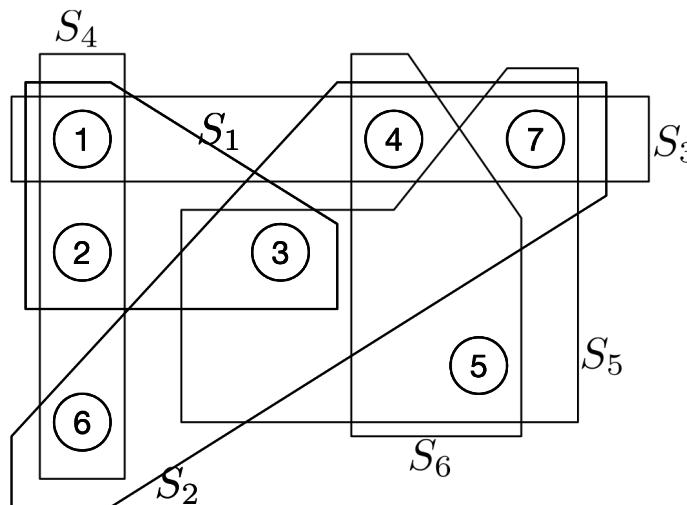
$$S_2 = \{3, 4, 6, 7\}$$

$$S_3 = \{1, 4, 7\}$$

$$S_4 = \{1, 2, 6\}$$

$$S_5 = \{3, 5, 7\}$$

$$S_6 = \{4, 5\}$$



, 3

We will show  
 $VC \leq_p SC$

# General template for proving that a problem B is NP-hard

0. Pick any NP-hard problem **A** to reduce from. **Goal:** show  $A \leq_p B$ .
1. Describe the reduction. I.e. how to take an arbitrary instance of **A** and convert it into a specific instance of **B**.
2. Explain why the reduction takes polynomial time.
3. Prove that the answer to the original instance of **A** is “yes” if and only if the answer to the constructed instance of **B** is “yes”:
  - a. Prove that **if** the answer to the original instance of **A** is “yes” **then** the answer to our constructed instance of **B** is “yes”.
  - b. Prove that **if** the answer to our constructed instance of **B** is “yes” **then** the answer to the original instance of **A** is “yes”.

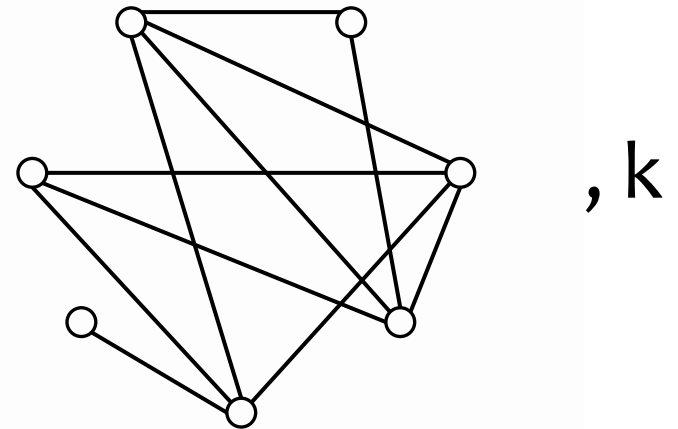
\*To show that problem **B** is NP-complete, also need to show  $B \in \text{NP}$



# Step 1: Describe the reduction

Given an arbitrary VC instance

VC: can we circle  $\leq k$  vertices so that every edge has at least one circled endpoint?



Construct a (carefully crafted)  
instance of SC

edges from VC instance

$U = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$

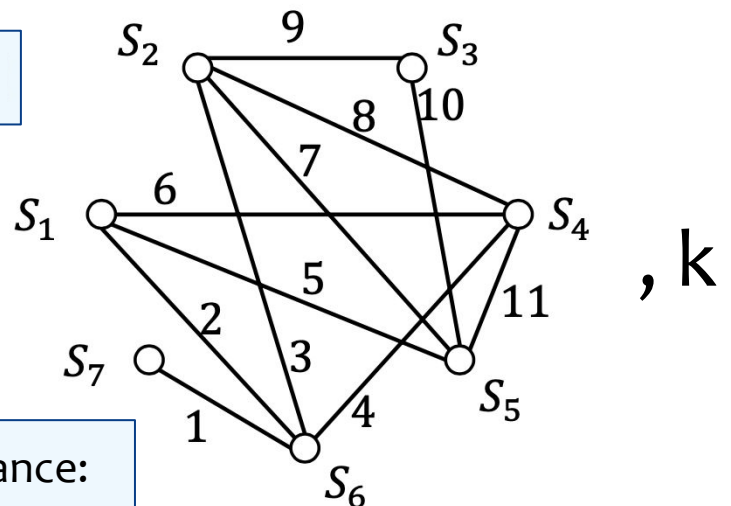
$S_1 = \{2, 5, 6\}, S_2 = \{3, 7, 8, 9\},$

$S_3 = \{9, 10\}, S_4 = \{4, 6, 8, 11\},$

$S_5 = \{5, 7, 10, 11\}, S_6 = \{1, 2, 3, 4\},$

$S_7 = \{1\}$

for each vertex  $v$  from VC instance:  
a set containing all incident edges

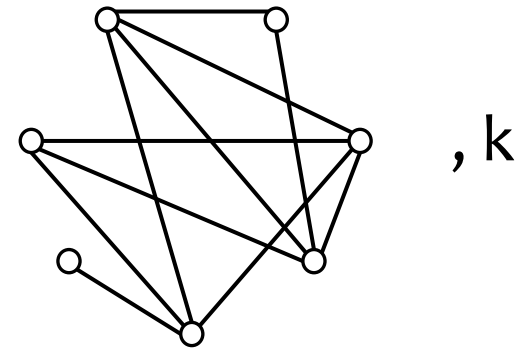


Step 2: Explain why the reduction runs in polynomial time

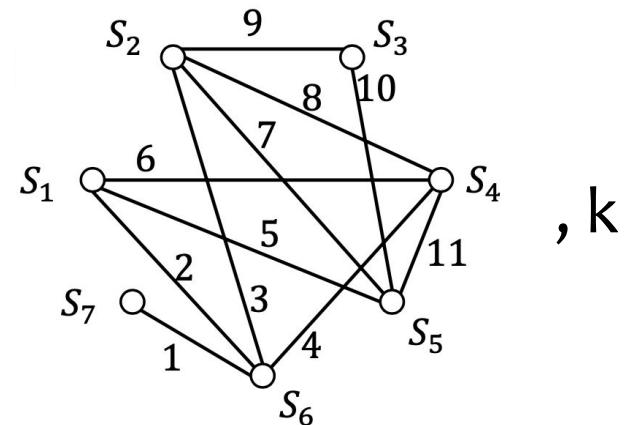
# Step 3a: VC “yes” $\Rightarrow$ SC “yes”

Suppose there is a VC of size  $\leq k$  in our original VC instance.

Goal: Use it to find an SC of size  $\leq k$  in our constructed SC instance.



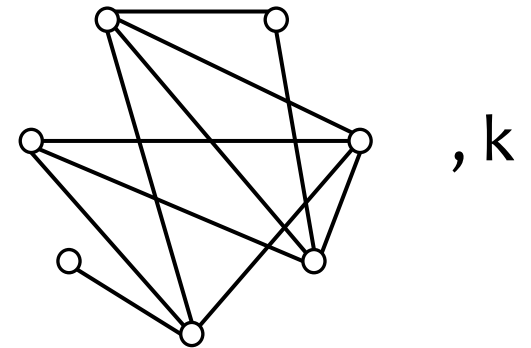
$U = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$   
 $S_1 = \{2, 5, 6\}, S_2 = \{3, 7, 8, 9\},$   
 $S_3 = \{9, 10\}, S_4 = \{4, 6, 8, 11\},$   
 $S_5 = \{5, 7, 10, 11\}, S_6 = \{1, 2, 3, 4\},$   
 $S_7 = \{1\}$



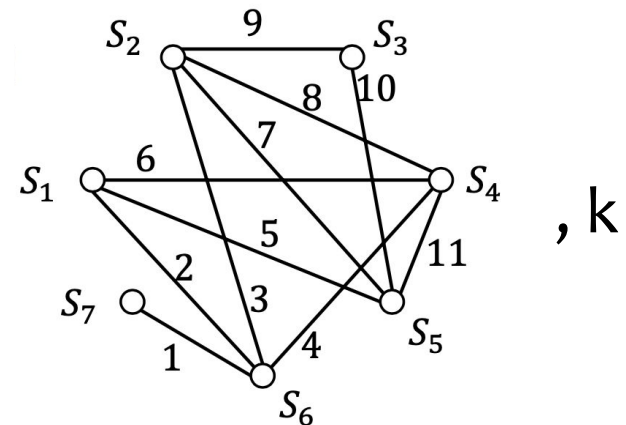
# Step 3b: SC “yes” $\Rightarrow$ VC “yes”

Suppose there is an SC of size  $\leq k$  in our constructed SC instance.

Goal: Use it to find a VC of size  $\leq k$  in the original VC instance.



$U = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$   
 $S_1 = \{2, 5, 6\}, S_2 = \{3, 7, 8, 9\},$   
 $S_3 = \{9, 10\}, S_4 = \{4, 6, 8, 11\},$   
 $S_5 = \{5, 7, 10, 11\}, S_6 = \{1, 2, 3, 4\},$   
 $S_7 = \{1\}$



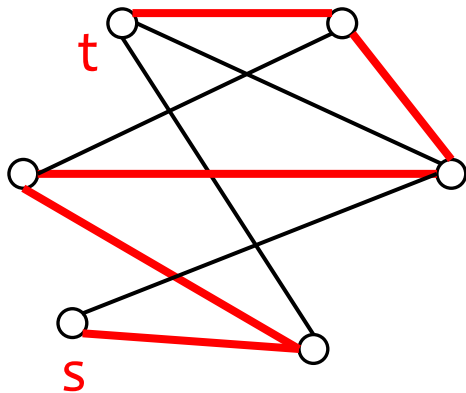
# Hamiltonian Path and Hamiltonian Cycle

A **Hamiltonian Path** in an undirected graph from a vertex **s** to a vertex **t** is a path that starts at **s**, ends at **t**, and visits every vertex **exactly** once.

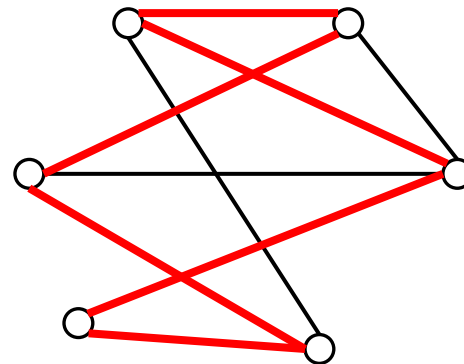
- Decision Problem: Given a graph and **s**, **t**, is there a Hamiltonian Path from **s** to **t**?

A **Hamiltonian Cycle** in an undirected graph is a cycle that visits every vertex **exactly** once.

- Decision Problem: Given a graph, does it have a Hamiltonian Cycle?

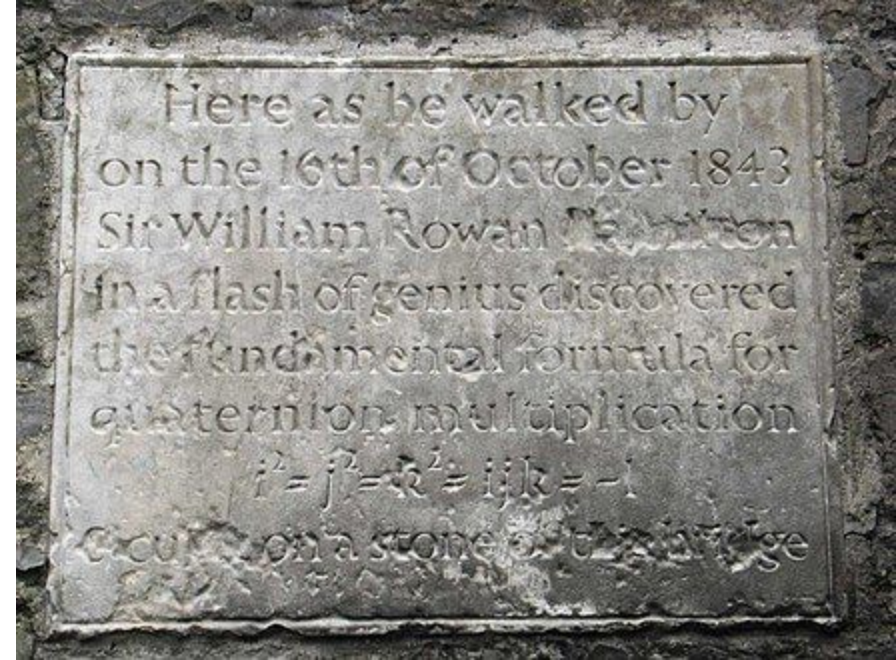


A Hamiltonian Path



A Hamiltonian Cycle

# William Rowan Hamilton



Posed the problem of whether a dodecahedron has a Hamiltonian Cycle.

Not the first to consider Hamiltonian cycles: In the 9th century in India, the poet Rudrata studied knight's tours in chess.





# Hamiltonian Path and Hamiltonian Cycle

A **Hamiltonian Path** in an undirected graph from a vertex **s** to a vertex **t** is a path that starts at **s**, ends at **t**, and visits every vertex **exactly** once.

- Decision Problem: Given a graph and  $s, t$ , is there a Hamiltonian Path from  $s$  to  $t$ ?

A **Hamiltonian Cycle** in an undirected graph is a cycle that visits every vertex **exactly** once.

- Decision Problem: Given a graph, does it have a Hamiltonian Cycle?

“Find the Longest Path”

Billy Joel parody:

<https://www.youtube.com/watch?v=a3wwogwEszo>



# Hamiltonian Path and Hamiltonian Cycle

A **Hamiltonian Path** in an undirected graph from a vertex **s** to a vertex **t** is a path that starts at **s**, ends at **t**, and visits every vertex **exactly** once.

- Decision Problem: Given a graph and **s**, **t**, is there a Hamiltonian Path from **s** to **t**?

A **Hamiltonian Cycle** in an undirected graph is a cycle that visits every vertex **exactly** once.

- Decision Problem: Given a graph, does it have a Hamiltonian Cycle?

**Hamiltonian Path** (HP) is NP-Complete  
(we won't prove)

**Hamiltonian Cycle** (HC) is NP-Complete

**We will prove HC is NP-hard by showing  $HP \leq_p HC$**

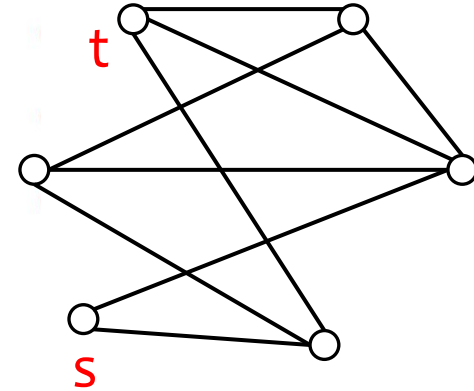
Different from an Eulerian Cycle which visits every **edge** exactly once

- Solvable in polynomial time by Euler's Theorem: "A graph has an Eulerian cycle if and only if every vertex has even degree."



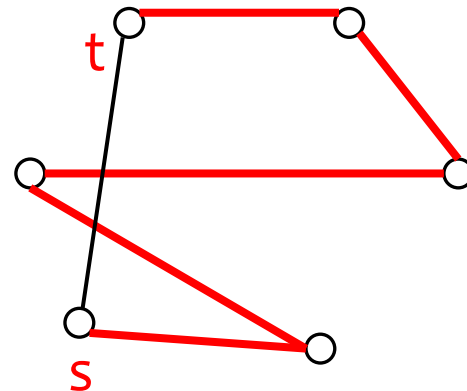
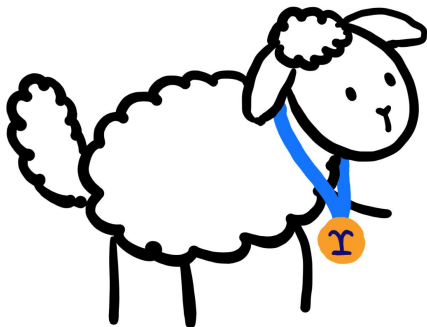
# Step 1: Describe the reduction

Given an arbitrary HP instance



Construct a (carefully crafted)  
instance of HC

Include every edge of  
the Hamiltonian Path  
plus the edge  $(s,t)$  to  
form a Hamiltonian  
Cycle!



Something  
is fishy...

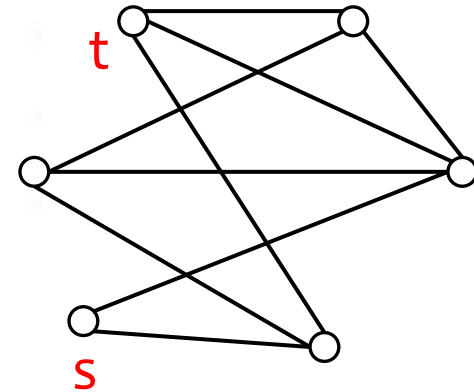


# Step 1: Describe the reduction

Given an arbitrary HP instance



Construct a (carefully crafted)  
instance of HC



Step 2: Explain why the reduction runs in polynomial time.

## Step 3a: HP “yes” $\Rightarrow$ HC “yes”

Suppose there is an HP in our original instance.

Goal: Use it to find an HC in our constructed instance.

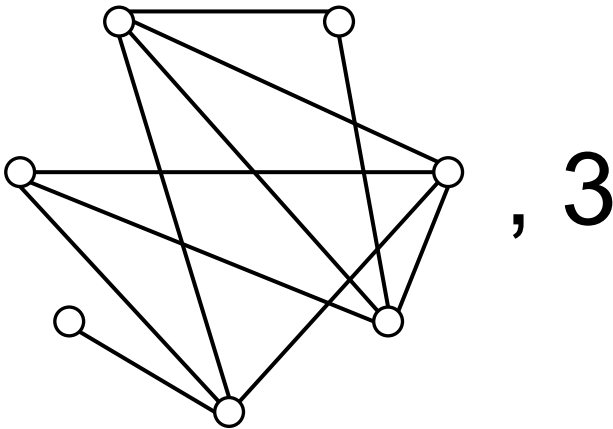
## Step 3b: HC “yes” $\Rightarrow$ HP “yes”

Suppose there is an HC in our constructed instance.

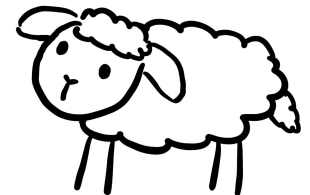
Goal: Use it to find an HP in the original instance.

# Independent Set (IS) Problem

- Given a graph, an **independent set** is a set **S** of vertices so that every pair of vertices in **S** does NOT have an edge between them.
- Independent Set decision problem:** Given a graph **G** and a budget **k**, does **G** have an independent set of size **k** or more?

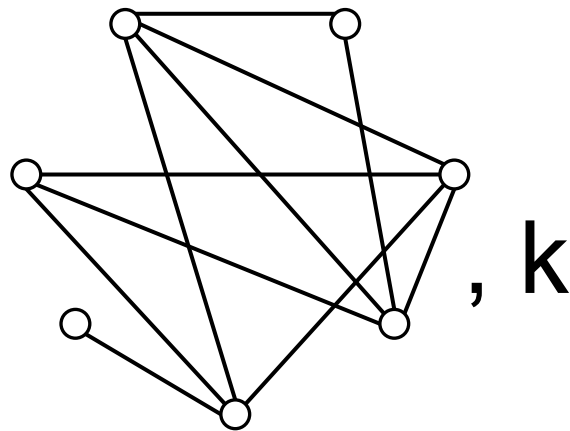
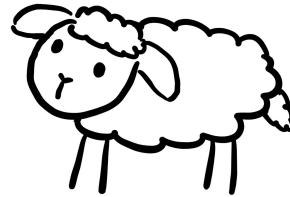


IS is the “opposite” of Clique. We can use this observation to build a reduction  $\text{Clique} \leq_p \text{IS}$ . See if you can find it...



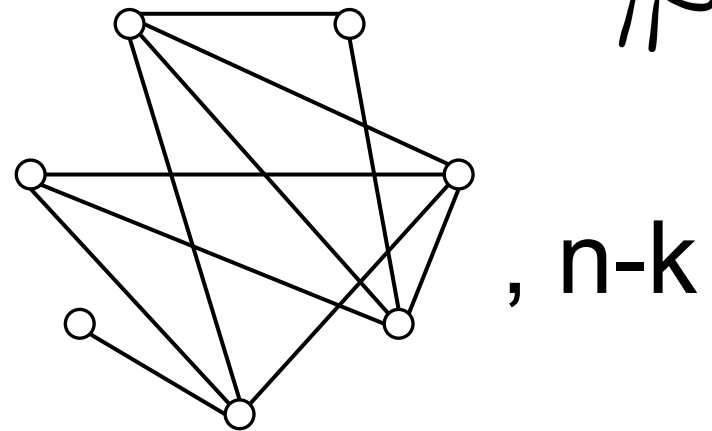
# IS is NP-hard: $VC \leq_p IS$

This reduction illustrates the principle that the budget can change.



Original VC instance

Same graph



Constructed IS instance

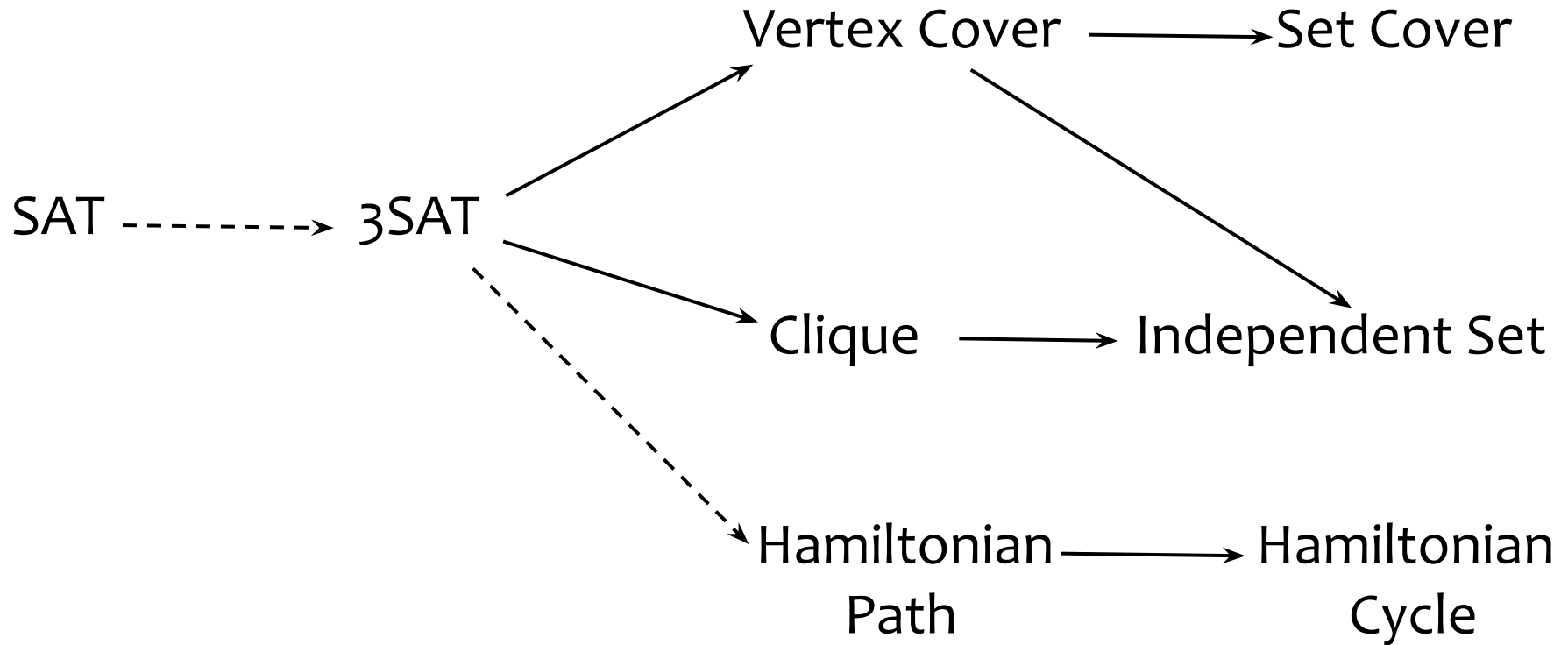
Goal: Show that an  $n$ -vertex graph  $G$  has a VC of size  $\leq k$  if and only if  $G$  has an IS of size  $\geq n-k$ .

Suppose  $G$  has a VC  $\mathbf{S}$  of size  $\leq k$ ...

Suppose  $G$  has an IS  $\mathbf{S}'$  of size  $\geq n-k$ ...

# A Web of NP-Hard Problems

(all of these are also in NP, and therefore NP-Complete)





# NP-Completeness is Everywhere

- **Constraint Satisfaction:** SAT, 3SAT
- **Routing:** Longest Path, Hamiltonian Path, Traveling Salesperson
- **Covering Problems:** Vertex Cover, Set Cover
- **Coloring Problem:** 3-Coloring a Graph
- **Scheduling Problems**
- **Social Networks:** Clique, Maximum Cut
- **Arithmetic Problems:** Subset Sum, Knapsack
- **Games:** Sudoku, Battleship, Super Mario, Pokémon

... ONE ALGORITHM WOULD SOLVE THEM ALL!

"About 20 diverse scientific disciplines were unsuccessfully struggling with some of their internal questions and came to recognize their intrinsic complexity when realizing that these questions are, in some form, NP-complete"

- Theory of Computing: a Scientific Perspective (Oded Goldreich, Avi Wigderson 1996)