Reduction of NP Problems & Property-Based Testing

Chenhao Zhang

CS396 Spring 2023
Northwestern
Plan of the week

- NP Problem & Reduction (Today)
- Examples, Reduction in Karp -- Wednesday
- Lab, Assignment 4 -- Friday
Many problems have efficient algorithms

Minimum Spanning Tree

Shortest path
Many problems have efficient algorithms

*Minimum* Spanning Tree

*Shortest* path
version with Yes/No answer

Has Spanning Tree w/ Cost <=15?

Has S-T path w/ Cost <=5?
version with Yes/No answer – decision problem

Has Spanning Tree w/ Cost <=15 ?

Has S-T path w/ Cost <=5 ?
version with Yes/No answer – decision problem

Has Spanning Tree w/ Cost $\leq 15$?

$$1 + 5 + 3 + 4 + 2 = 15$$

Has S-T path w/ Cost $\leq 5$?

$$1 + 4 = 5$$
version with Yes/No answer – *decision problem*

Has Spanning Tree w/ Cost $\leq 15$?

![Graph with spanning tree and cost calculation](image1)

1 + 5 + 3 + 4 + 2 = 15

Yes

Has S-T path w/ Cost $\leq 5$?

![Graph with S-T path and cost calculation](image2)

1 + 4 = 5
version with Yes/No answer – decision problem

Has Spanning Tree w/ Cost $\leq 15$?

Has S-T path w/ Cost $\leq 5$?

Yes

$1+5+3+4+2=15$

Yes

$1+4=5$
Yes-Instance has a certificate, i.e., proof of yes

Has Spanning Tree w/ Cost $\leq 15$ ?

Has S-T path w/ Cost $\leq 5$ ?

$1 + 5 + 3 + 4 + 2 = 15$

$1 + 4 = 5$

Yes
No-Instance has no *certificate*, proof of yes

Has Spanning Tree w/ Cost $\leq 14$?

\[1+5+3+4+2=15 > 14\]

Has S-T path w/ Cost $\leq 4$?

\[1+4=5 > 4\]
No-Instance has no *certificate*, proof of yes

Has Spanning Tree w/ Cost $\leq 14$?

```
1 + 5 + 3 + 4 + 2 = 15 > 14
```

Has S-T path w/ Cost $\leq 4$?

```
1 + 4 = 5 > 4
```
No-Instance has no *certificate*, proof of yes

Has Spanning Tree w/ Cost $\leq 14$?

\[
\begin{align*}
3 & \quad 6 & \quad 1 & \quad 5 \\
6 & \quad 5 & \quad 4 & \quad 2 \\
5 & \quad 6 & \quad 6 & \quad 6
\end{align*}
\]

\[1+5+3+4+2=15 > 14\]

... ...

Has S-T path w/ Cost $\leq 4$?

\[
\begin{align*}
6 & \quad 1 & \quad 5 \\
3 & \quad 5 & \quad 6 \\
4 & \quad 5 & \quad 2
\end{align*}
\]

\[1+4=5 > 4\]
Has Spanning Tree w/ Cost $\leq 14$?

Has S-T path w/ Cost $\leq 4$?

$1+5+4+2=12 \leq 14$

$1+4=5 > 4$

No-Instance has no *certificate*, proof of yes
No-Instance has no *certificate*, proof of yes

Has Spanning Tree w/ Cost $\leq 14$?

Has S-T path w/ Cost $\leq 4$?

1+5+4+2=12 $\leq$ 14

4=4 $\leq$ 4
No-Instance has no *certificate*, proof of yes

Has Spanning Tree w/ Cost $\leq 14$?

$$1+5+4+2=12 \leq 14$$

Has S-T path w/ Cost $\leq 4$?

$$4=4 \leq 4$$
Has Spanning Tree w/ Cost $\leq 14$?

1+5+4+2=12 $\leq 14$

......

Has S-T path w/ Cost $\leq 4$?

4=4 $\leq 4$
There are also many other problems...

Can we get all by buying only 2 bundles?

SET-COVER
There are also many other problems...

Can we get all by buying only 2 bundles?

SET-COVER
There are also many other problems...

Can we watch all roads by setting only 2 sentry points?

**Vertex-Cover**
There are also many other problems...

Can we watch all roads by setting only 2 sentry points?

**Vertex-Cover**
There are also many other problems...

Can we watch all roads by setting only 2 sentry points?

VERTEX-COVER
There are also many other problems...

Is there a cycle that visits all vertices?
There are also many other problems...

Is there a cycle that visits all vertices?

Hamiltonian-Cycle
Q: What do they have in common?
Q: What do they have in common?

A: Validity of certificate EASY to check! (can be done in polynomial-time)
Q: What do they have in common?

A: Validity of certificate EASY to check!
(can be done in polynomial-time)

\[ O(n) \quad O(n^2) \]
Q: What do they have in common?

A: Validity of certificate EASY to check!
(can be done in polynomial-time)

\[ O(n) \quad O(n^2) \quad O(n^{10^{10}}) \]
Q: What do they have in common?

A: Validity of certificate EASY to check!
(can be done in polynomial-time)

\[ O(n) \quad O(n^2) \quad O(n^{10^{10}}) \quad O(1.01^n) \]
Q: What do they have in common?

A: Validity of certificate EASY to check! (can be done in polynomial-time)

**NP-Problems**

(Non-deterministic Polynomial-time)
Q: Any difference?
“Easy”

Minimum-Spanning-Tree

“Hard”

Set-Cover

Shortest-Path

Vertex-Cover

Hamiltonian-Cycle
Q: Any difference?

A: It is generally believed that:

“Hard” problems have NO efficient algorithms
Q: Any difference?

A: It is **generally believed** that:
   “Hard” problems have NO efficient algorithms

   But there’s no proof for it yet...
Q: Any difference?

A: It is **generally believed** that:
   "Hard" problems have **NO efficient algorithms**

   But there’s no proof for it yet...
How do you prove that an NP-problem is “Hard”?

Design an efficient algorithm for problem N!
How do you prove that an NP-problem is “Hard”? 

Design an efficient algorithm for problem N!

But... problem N is “Hard”
If N could be solved, a known hard problem $H$ could be also solved.
How do you prove that an NP-problem is “Hard”?

“reduction”

If N could be solved, a known hard problem $H$ could be also solved.
One-Call Reduction

Instance of Problem H

Reduction

Instance of Problem N
One-Call Reduction – Correctness Property

H is the problem known to be hard

Yes

h certificate

\exists \ c^h

Instance of Problem H

Reduction

n is the new problem

Yes

n certificate

\exists \ c^n

Instance of Problem N
One-Call Reduction – Correctness Property

Instance of Problem H

Reduction

Instance of Problem N
One-Call Reduction

$k=2$

SET-COVER

VERTEX-COVER
One-Call Reduction

VERTEX-COVER

SET-COVER
One-Call Reduction

**K**=2

**Vertex-Cover**

**Set-Cover**
One-Call Reduction

Vertex-Cover

Set-Cover
One-Call Reduction

**Vertex-Cover**

**Set-Cover**

$k=1$
One-Call Reduction

Suppose there is an algorithm for $N$.
One-Call Reduction

Reduction from H to N

Algorithm for N
One-Call Reduction

Reduction from H to N

Algorithm for N

Algorithm for H
One-Call Reduction

Reduction from H to N

Algorithm for N

Algorithm for H
One-Call Reduction

Algorithm for H

Reduction from H to N

Algorithm for N

$h$ Yes
One-Call Reduction

Algorithm for H

Reduction from H to N

Algorithm for N

$h\rightarrow n\rightarrow$
One-Call Reduction

Reduction from $H$ to $N$

Algorithm for $N$

Algorithm for $H$
One-Call Reduction

Reduction from H to N

Algorithm for N

Algorithm for H
One-Call Reduction

Algorithm for H

Reduction from H to N

Algorithm for N

No
Reduction and Justifications of Correctness

Reduction from \( H \) to \( N \)

Algorithm for \( N \)
Call this part “instance construction” from now on
Instance Construction

VERTEX-COVER

SET-COVER
Instance Construction

Instance Construction

Vertex-Cover

Set-Cover
Instance Construction

\[ k=2 \]

**VERTEX-COVER**

**SET-COVER**

\[ k=2 \]
Justifying N Yes => H Yes

Instance Construction

\[ h \] to [Algorithm for N]

Yes

\[ c^h \] h certificate

\[ c^n \] n certificate
Justifying N Yes => H Yes

Instance Construction

Algorithm for N

Backward Certificate Construction

h certificate

n certificate
Backward Certificate Construction

Instance Construction

Vertex-Cover

Set-Cover

$k=2$
Backward Certificate Construction

Instance Construction

Backward Certificate Construction

k=2

1 2 3 4

k=2

 VERTEX-COVER

SET-COVER
Backward Certificate Construction

Instance Construction

Backward Certificate Construction

k=2

VERTEX-COVER

SET-COVER
Backward Certificate Construction

Instance Construction

Backward Certificate Construction

k=2

VERTEX-COVER

SET-COVER
Backward Certificate Construction

1 → Instance Construction → Backward Certificate Construction

k=2

Vertex-Cover

Set-Cover
Justifying N No => H No

Instance Construction

Algorithm for N

h

n

h-certificate

n-certificate
Justifying N \( \text{No} \Rightarrow \text{H No} \)

- \( h \)
- \( \text{Instance Construction} \)
- \( n \)
- Algorithm for N

\( \not \in C^x \) ↔ \( \not \in C^y \)
Justifying $N \ No \Rightarrow H \ No$

* we are in a classical world

\[
\exists \ C^x \iff \exists \ C^y
\]
Justifying N No => H No

\[ h \rightarrow \text{Instance Construction} \rightarrow n \rightarrow \text{Algorithm for N} \rightarrow \text{Yes} \]

\[ h \rightarrow \text{Forward Certificate Construction} \rightarrow c^n \rightarrow \text{n certificate} \]

\[ c^h \rightarrow \text{h certificate} \]
Forward Certificate Construction

\begin{align*}
\text{Vertex-Cover} & : k=2 \\
\text{Set-Cover} & : k=2
\end{align*}
Forward Certificate Construction

**Vertex-Cover**

**Forward Certificate Construction**

**Set-Cover**
Forward Certificate Construction

Vertex-Cover

Set-Cover
Forward Certificate Construction

**Instance Construction**

k=2

**Forward Certificate Construction**

k=2

**VERTEX-COVER**

**SET-COVER**