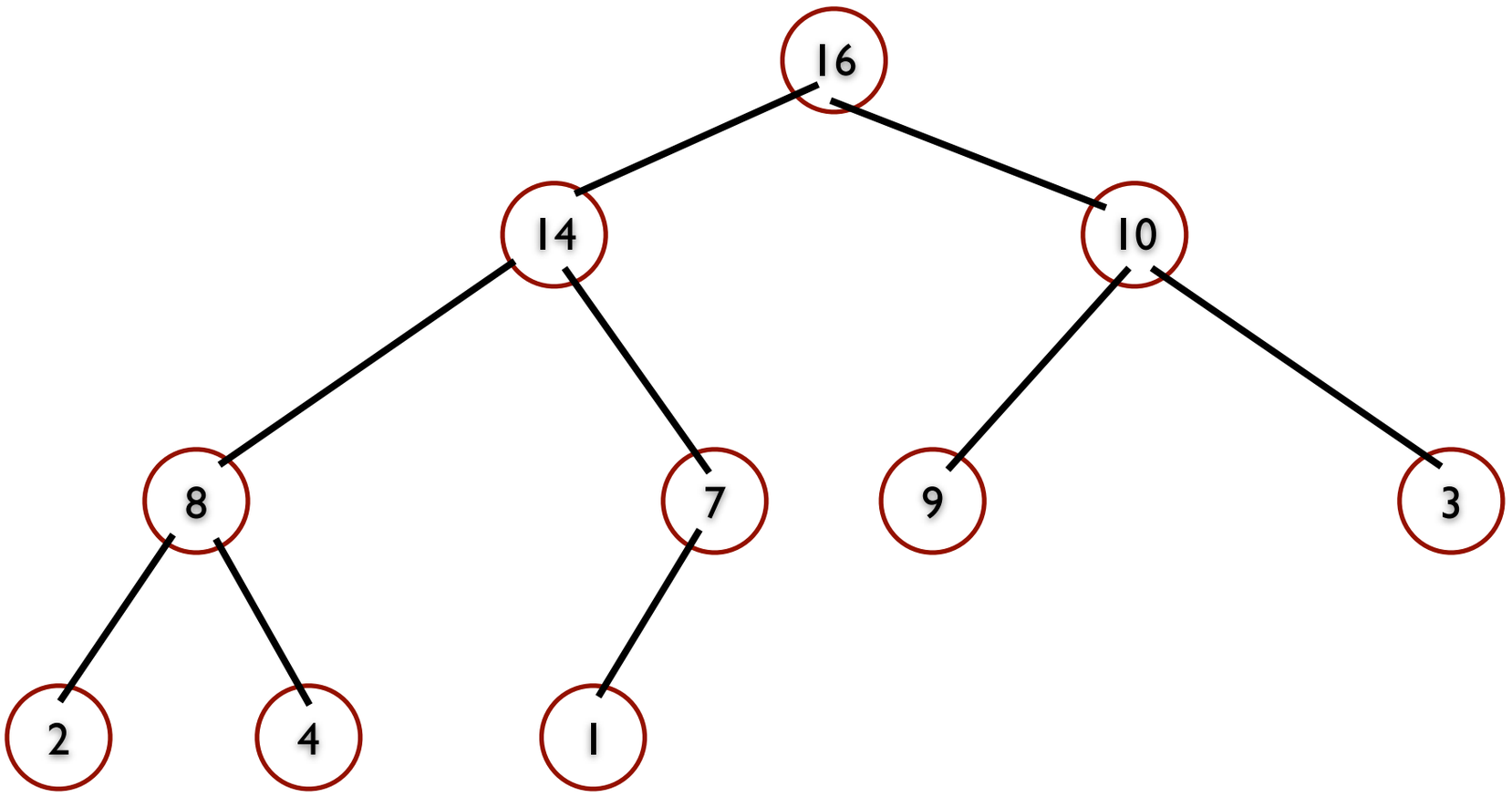
CSc106 Lab 6 Fall 2012

Heaps: A quick review
Complexity: P vs NP

Heaps

- A nearly complete binary tree
- Satisfies a heap property:
  - Max heaps: every node (except the root) is less than or equal to its parent
  - Min heaps: every node (except the root) is greater than or equal to its parent

A Max Heap



Complexity

Kinds of Problems

- Optimization
  - Find the smallest/largest...
- Decision
  - Is there a solution of size k?
- Can convert from Optimization to Decision
  - ask if there's a solution for a specific measure

Fitch's

- Optimization Version:
  - Find a most parsimonious tree...
- Decision Version:
  - Is there a tree of size at most k?

Fitch's

- Optimization Version:

- Decision Version:

MST

- Optimization Version:

- Decision Version:

MST

- Optimization Version:
  - Find the minimum-cost spanning tree
- Decision Version:
  - Is there a spanning tree of cost less than or equal *k*?

Hardness

- Complexity Classes:
  - P
  - NP
  - NP-hard
  - NP-complete
  - lots of others....

The Class P

- How do we know a problem is in P?

The Class P

- How do we know a problem is in P?
  - It has an algorithm
  - That runs in at most polynomial time

The Class P

- What problems have we seen in the Class P?

The Class P

- What problems have we seen in the Class P?
  - MST
  - Fitch's
  - Quicksort
  - Selection Sort
  - ...

The Class NP

- How do we know a problem is in NP?
  - It has an algorithm
  - We can check an answer in polynomial time

The Class NP

- What problems have we seen in the Class NP?
  - MST, Fitch's, Quicksort, etc.
  - Large Parsimony Problem
  - Travelling Salesperson Problem
  -

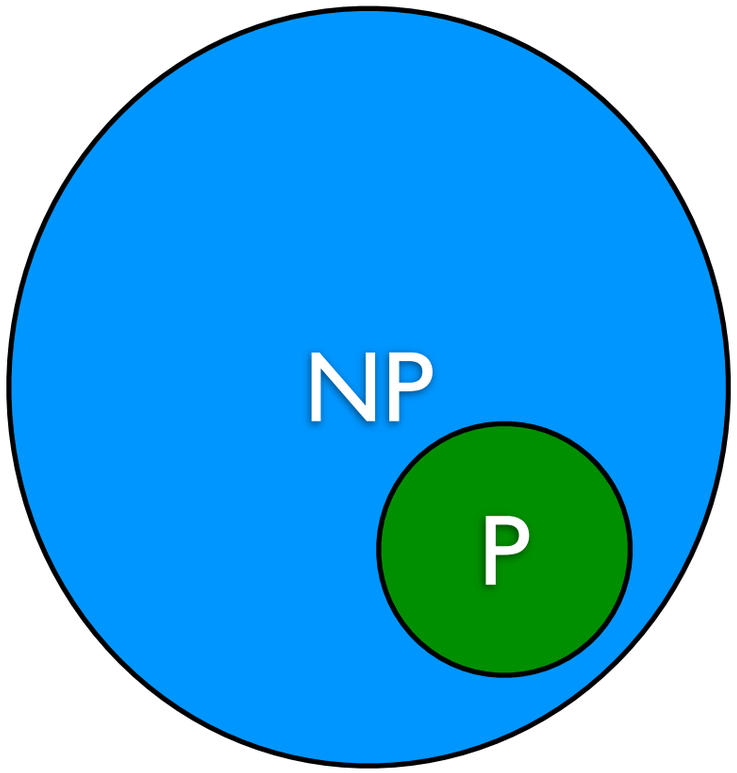
P and NP

- How are they related??
- P is in NP
  - Why?

P and NP

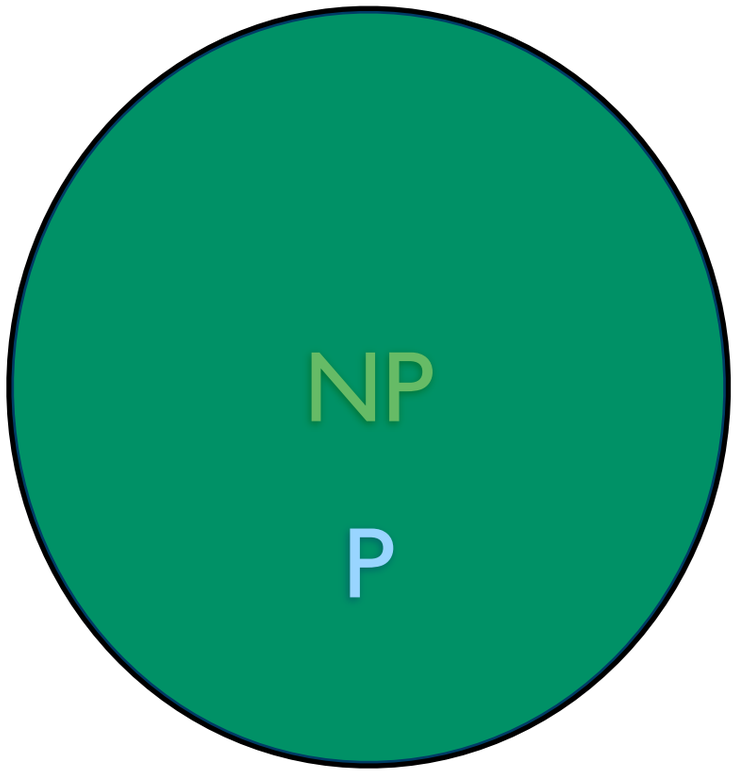
- How are they related??
- P is in NP
  - Why?
- What does that look like?

P and NP: two possibilities



$P \neq NP$

P and NP:
two possibilities



P = NP

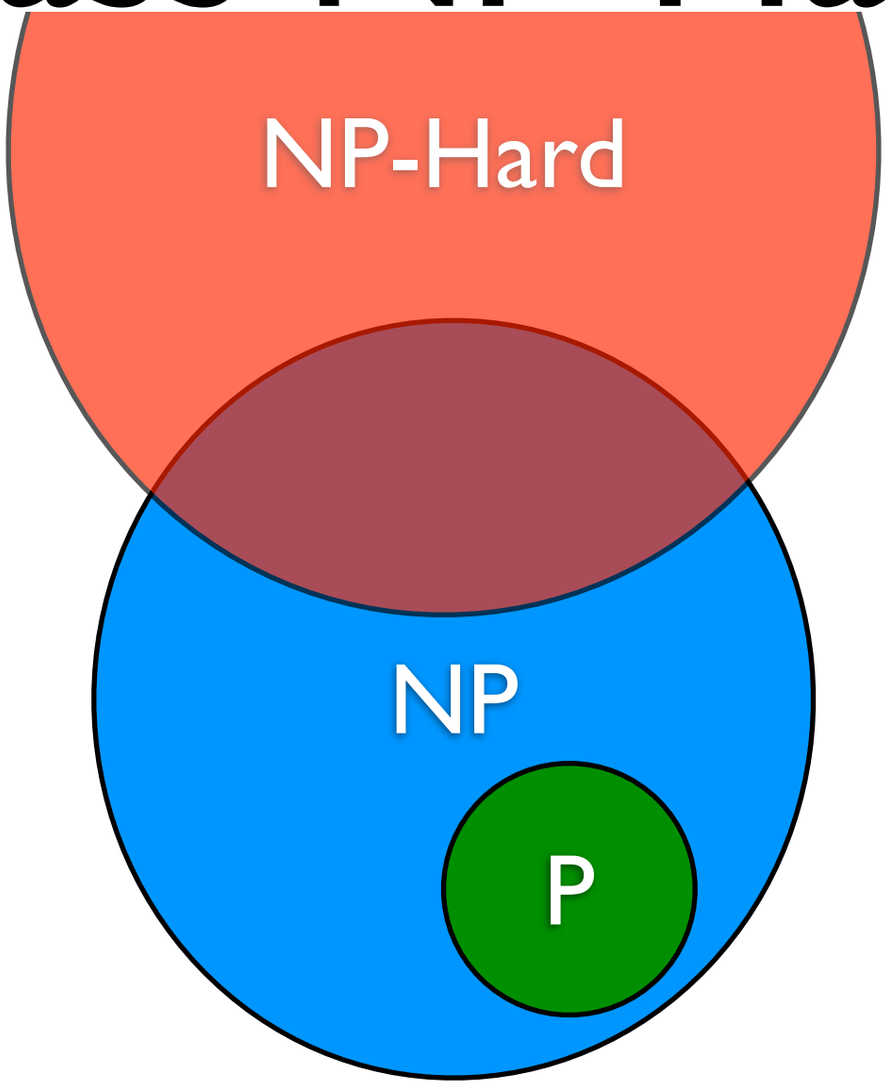
NP-Hard and NP-Complete

- Where do they fit in???

Class NP-Hard

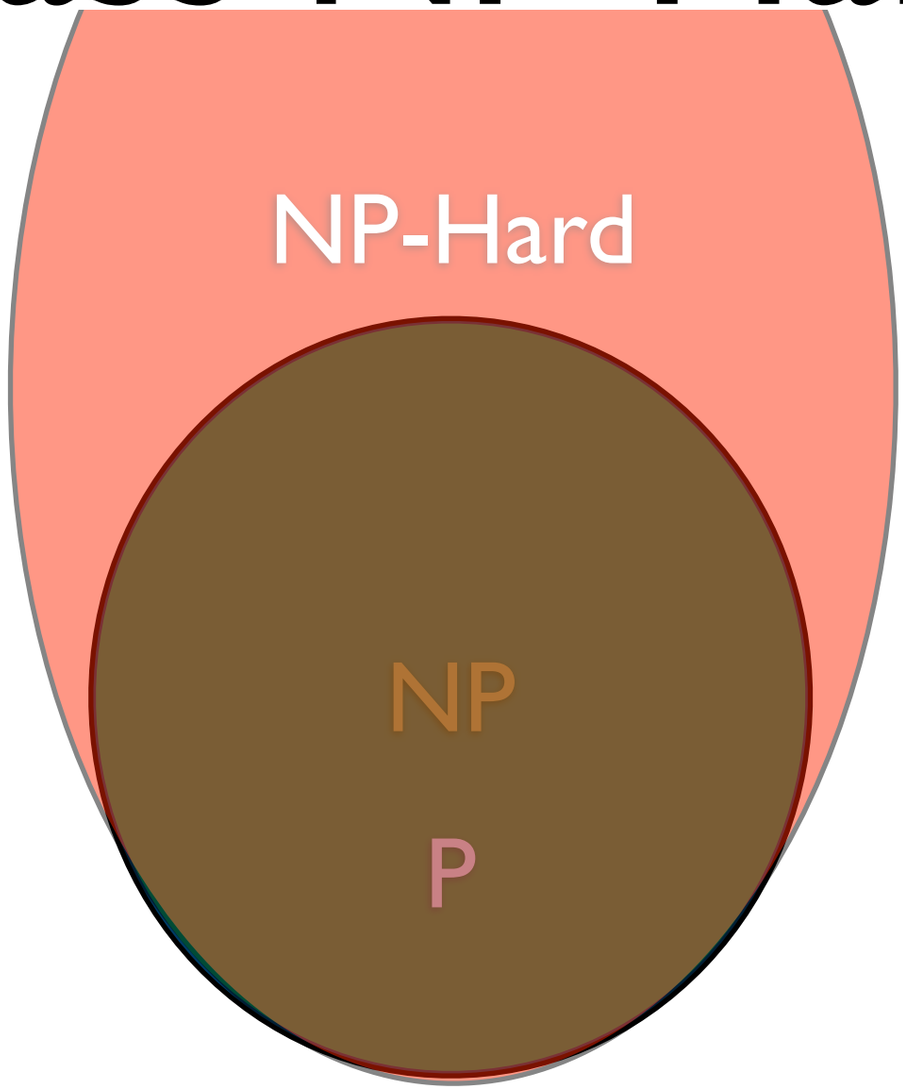
- Problems that are at least as hard as any other in NP
- No necessarily in NP!
  - For example: Halting Problem
    - undecidable (cannot check correctness)

Class NP-Hard



$$P \neq NP$$

Class NP-Hard

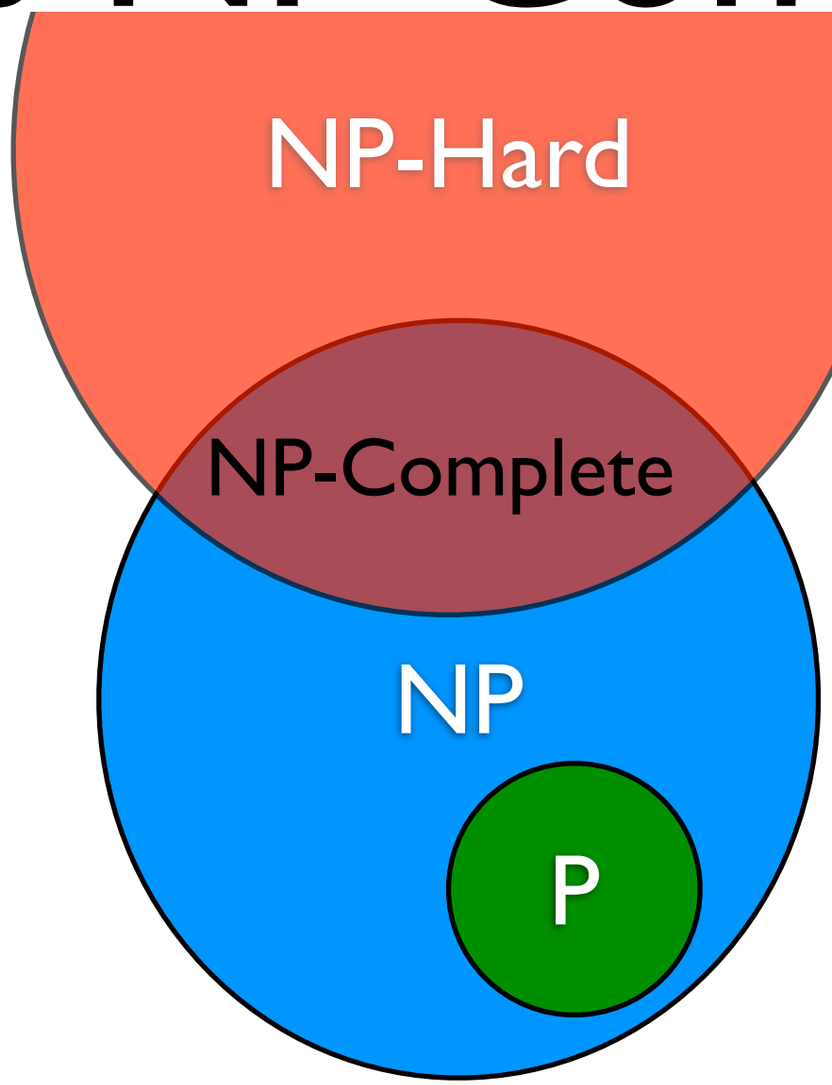


P = NP

Class NP-Complete

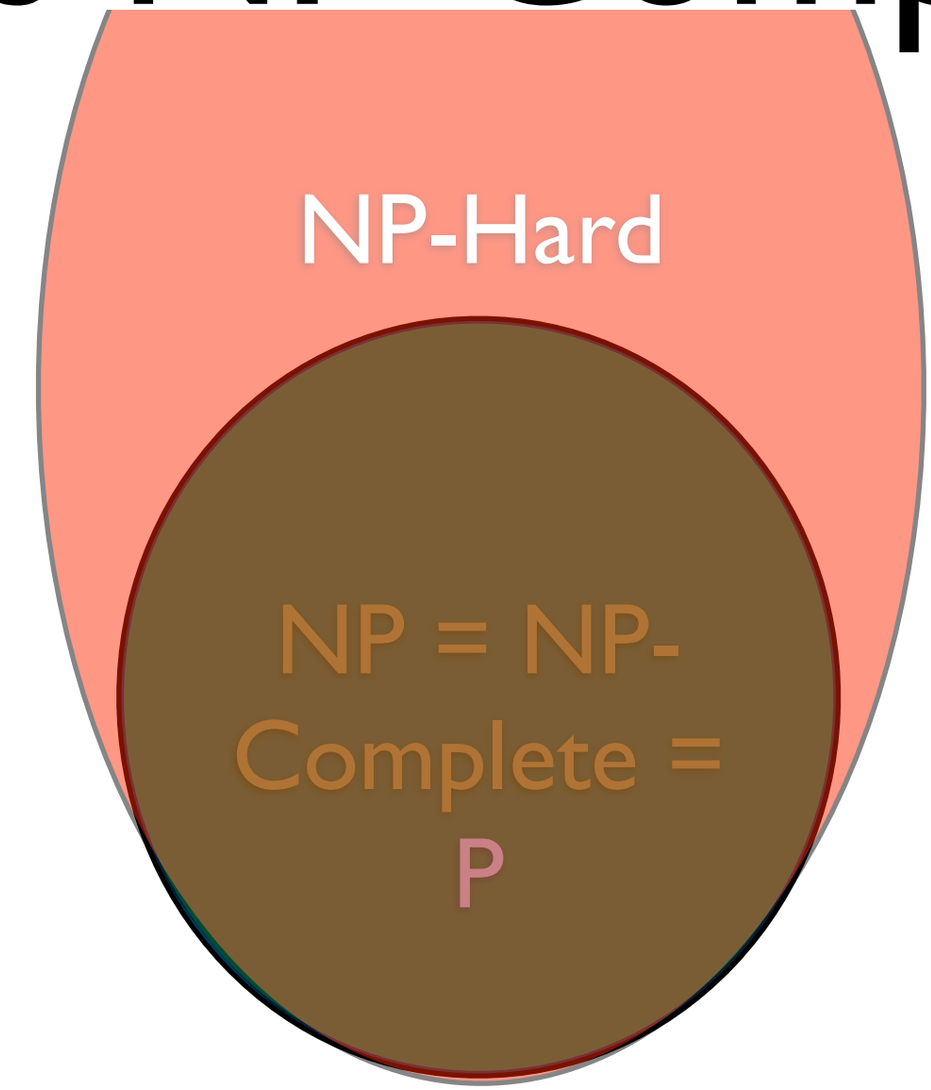
- Problems that are at least as hard as any other in NP
- **And* in NP*

Class NP-Complete



$P \neq NP$

Class NP-Complete



P = NP

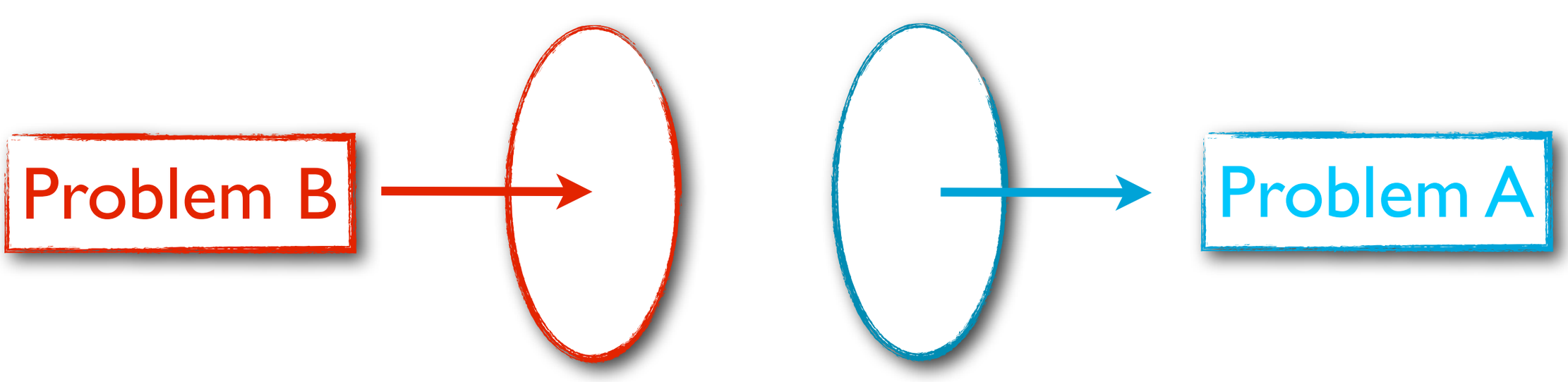
NP-Complete

- How to show Problem A is NP-Complete:

1. Show it is in NP
2. Reduction: convert a hard problem, B, into A
3. Show that if B is solvable, so is A and vice versa
4. Show this transformation is poly-time

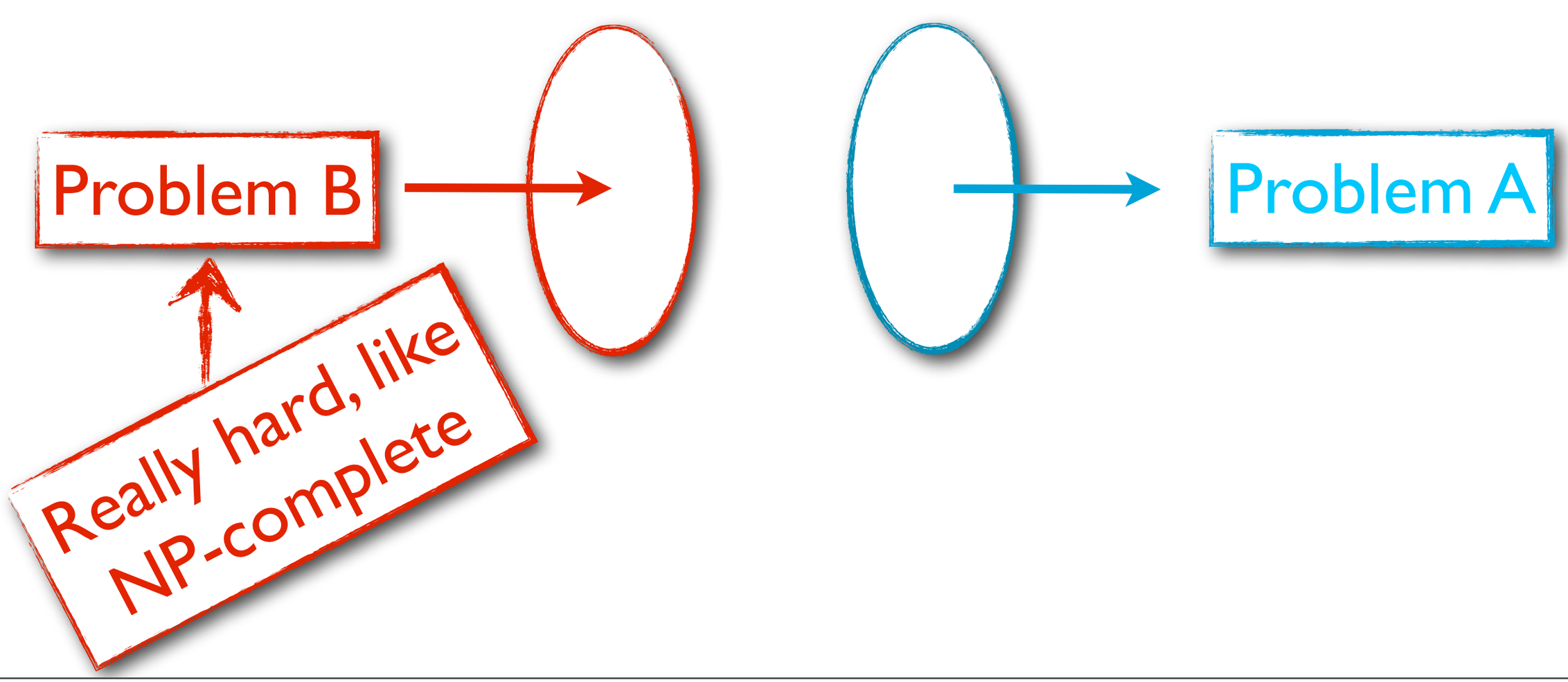
NP-Complete

- How to show Problem A is NP-Complete:
  - Show it is in NP
  - Reduction: convert a hard problem, B, into A



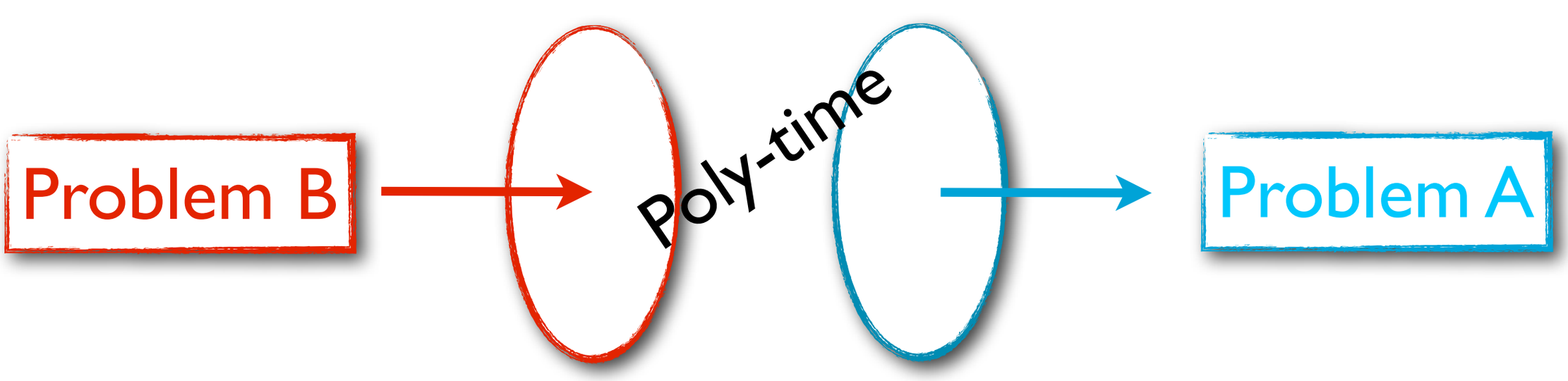
NP-Complete

- How to show Problem A is NP-Complete:
  - Show it is in NP
  - Reduction: convert a hard problem, B, into A



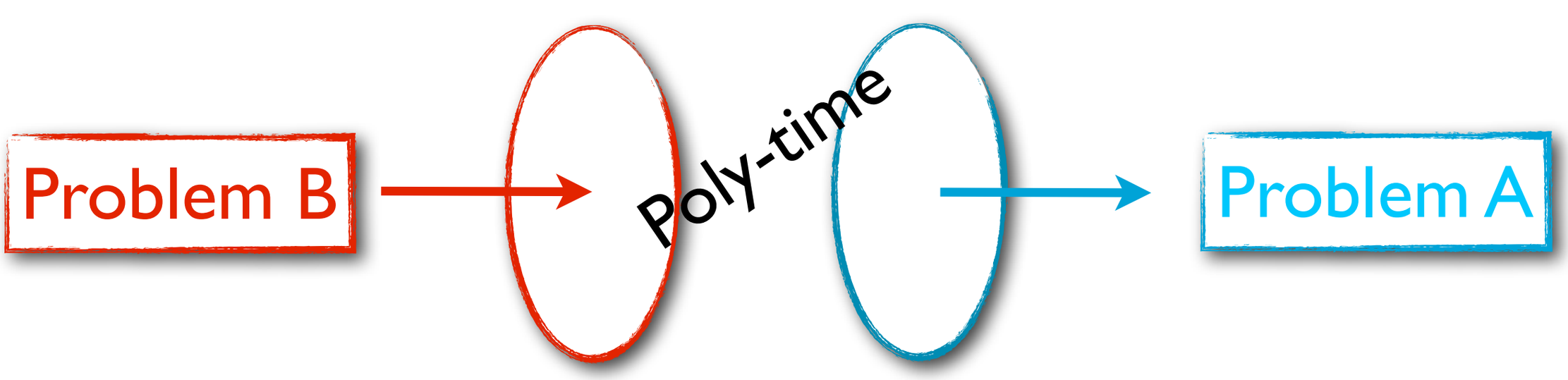
NP-Complete

- How to show Problem A is NP-Complete:
  - Show it is in NP
  - Reduction: convert a hard problem into A



NP-Complete

- If Problem B is NP-Complete:
  - NO KNOWN POLY-TIME ALGORITHM



NP-Complete

- We can use Problem A to solve Problem B, so if A isn't hard, we could solve solve B!