Outline

- Polynomial time computability
- Reductions
- Optimization, decision and verification versions of computational problems
- Classes P, NP
Polynomial time computability

- Class of polynomial functions
  - $f(x) = O(x^k)$
  - Closed over product
    - E.g., product of polynomials is also a polynomial
    - This does not apply to the class of cubic functions
Polynomial time computability

- **Problem**
  - Solving a linear equation
    - E. g., given $a$, $b$, find the value of $x$ s. t. $ax + b = 0$

- **Solver**
  - Given the linear equation problem (LEP), a linear equation solver (LES) is defined as a black box with input $(a_l, b_l)$ and output $x_l$
Polynomial time computability

- **Problem**
  - Solving a quadratic equation
    - E.g., given $a$, $b$, $c$, find the value of $x$ s.t. $ax^2 + bx + c = 0$

- **Solver**
  - Given the quadratic equation problem (QEP), a quadratic equation solver (QES) is defined as a black box with input $(a_q, b_q, c_q)$ and output $(x_{q,1}, x_{q,2})$
Reduction

- **Problem**
  - Solving a linear equation

- **Solver**
  - A black box that reduces a LEP to a QEP and then uses a QES to solve the original LEP

![Diagram showing the process of reducing LEP to QEP and then solving with QES]
Optimization and decision versions

- Shortest Path Problem-Optimization Version
  - Given a graph $G = (V, E)$ and two vertices $s$ and $t$, find the minimum cost path between $s$ and $t$

- Solver
  - Given the shortest path problem optimization version (SPP-OV), a SPP-OV solver (SPP-OVS) is defined as a black box with input $(G, s, t)$ and output $(s-t \text{ path})$

*Example: Dijkstra's Algorithm*
Optimization and decision versions

- **Shortest Path Problem-Decision Version**
  - Given a graph $G = (V, E)$, two vertices $s$ and $t$ and a cost $x$, say whether there exists an $s$-$t$ path of cost $\leq x$

- **Solver**
  - Given the shortest path problem decision version (SPP-DV), an SPP-DV solver (SPP-DVS) is defined as a black box with input $(G, s, t, x)$ and output (Yes/No)
Reductions between OV and DV

- Solving SPP-DV with SPP-OVS is feasible
- Solving SPP-OV with SPP-DVS looks more complicated but it's also feasible
Shortest Path Problem-Verification Version

- Given a graph $G = (V, E)$, two vertices $s$ and $t$, a cost $x$ and an $s$-$t$ path (certificate), say whether the $s$-$t$ path is of cost $\leq x$

Solver

- Given the shortest path problem verification version (SPP-VV), an SPP-VV solver (SPP-VVS) is defined as a black box with input $(G, s, t, x, s$-$t$ path) and output (Yes/No)
Optimization and decision versions are equally difficult

- Pay attention in developing an OV/DVS adapter such that it has polynomial time complexity and thus its composition with a polynomial time algorithm is still polynomial

Verification version is usually easy, this being independent of OV and DV complexity

- E.g., Longest Path Problem OV and DV haven't polynomial time algos, while LPP VV is clearly polynomial time

- This applies to NP problems
Classes P and NP

- **Class P**
  - The set/class of decision version problems that can be solved in polynomial time

- **Class NP**
  - The set/class of verification version problems that can be solved in polynomial time

- $P \subseteq NP$ easy to observe from definitions
  - Turing machine is needed for formal proof

- $P = NP$ is an open issue