CS F364
Design & Analysis of Algorithms

Complexity – Optimization Problems

Intractable Optimization Problems

- Complexity Classes and Reduction
- Hardness of Optimization Problems
  - Examples
- Versions of Optimization Problems
  - Relative Complexity

CLASSES PO AND NPO

- PO
  - An optimization problem π is in PO if there exists an algorithm that solves π in polynomial time
- NPO:
  - An optimization problem **π = (I, F, m, goal)** belongs to the class NPO if
    - membership in **I** is *decidable in polynomial time*
    - there exists a polynomial **q** such that
      - given **x** in **I** and for all **y** in **F(x):**
      - **|y| <= q(|x|)** and
      - **y in F(x)** is *decidable in polynomial time*.
    - m is *computable in poly time*.
  - Example:
    - Vertex Cover is in NPO

RECALL: REDUCTIONS

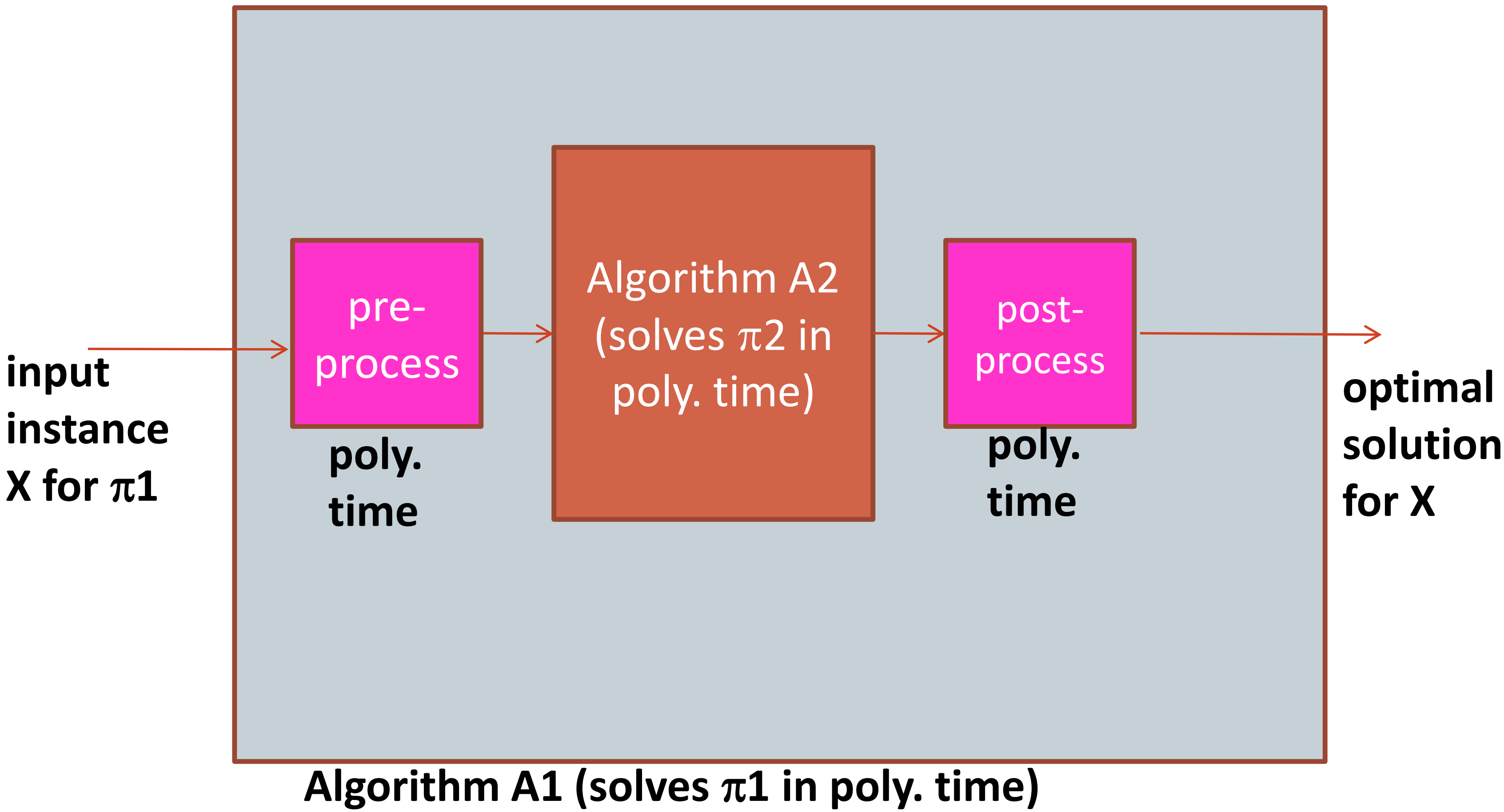
- Recall:
  - We say π_1 ***(polynomially) reduces to*** π_2
    - if there is a polynomial-time computable function **$f : I(\pi_1) \dashrightarrow I(\pi_2)$** such that
    - for every **$x \in I(\pi_1)$,**
      - $\pi_1(x) = 1$ if and only if $\pi_2(f(x)) = 1$
- We use $\pi_1 \precsim \pi_2$
  - to denote that π_1 ***(polynomially) reduces to*** π_2

RECALL: REDUCTIONS AND ORACLES

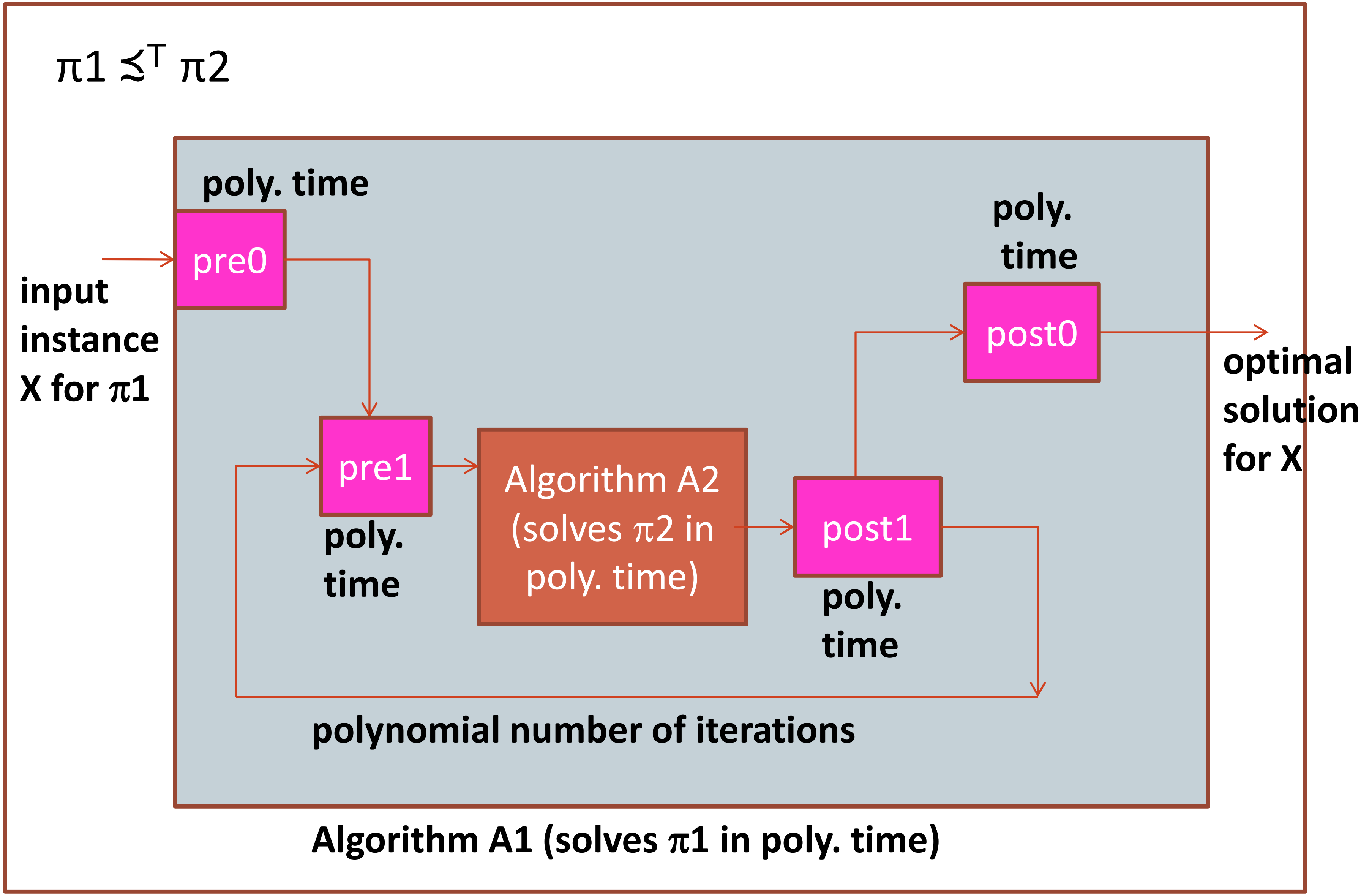
- Another way of understanding $\pi_1 \precsim \pi_2$
  - If there is an algorithm $\mathbf{A_2}$ to (efficiently) solve $\boldsymbol{\pi_2}$
  - then one can use $\mathbf{A_2}$ construct an (efficient) algorithm $\mathbf{A_1}$ to solve $\boldsymbol{\pi_1}$
- Alternatively,
  - If we can use a poly-time algorithm $\mathbf{A_2}$ that solves $\boldsymbol{\pi_2}$ to construct a poly-time algorithm $\mathbf{A_1}$ that solves $\boldsymbol{\pi_1}$,
  - then $\boldsymbol{\pi_1 \precsim \pi_2}$
- Question: Can we quantify "use"?
  - Karp Reduction: Single query to Oracle
  - Turing Reduction: Polynomial Number of queries to Oracle

Karp reduction (or 1-1 Reduction)

$\pi 1 \precsim^{K} \pi 2$



TURING REDUCTION



HARDNESS OF OPTIMIZATION PROBLEMS

- An optimization problem π in $\mathcal{NPO}$ is said to be $\mathcal{NP}$-hard if for all problems π' in $\mathcal{NP}$:
  - π' $\precsim$ π
- Note on reduction:
  - We will typically assume Turing reductions as a Karp reduction is a special case of a Turing reduction.

OPTIMIZATION PROBLEMS - FORMULATIONS

- An optimization problem π = **<I, F, m, goal>** can be formulated in different ways:
  - **Construction Version (π_c)** :
    - Given an input instance x, find the ***optimal solution*** **OPT(x)**
  - **Evaluation Version (π_e)** :
    - Given an input instance **x**, find ***the optimal measure*** i.e. measure of the optimal solution **m*(x) = m(x,OPT(x))**
  - **Decision Version (π_d) :**
    - Given an input instance **x**, and a threshold value **k** decide whether ***the optimal measure is bounded by k***
      - i.e. Is **m*(x) <= k** ? (if **goal** is *min*)
      - Is **m*(x) >= k** ? (if **goal** is *max*)

Optimization Problems – Formulations : Example

- Example: TSP
  - TSP_c :
    - Given a weighted, complete graph G, find a minimum weight tour.
  - TSP_e :
    - Given a weighted, complete graph G, find the weight of a minimum weight tour.
  - TSP_d :
    - Given a weighted, complete graph G, and a number k, find whether the weight of a minimum weight tour is less than k.

OPTIMIZATION PROBLEMS – FORMULATIONS – RELATIVE COMPLEXITIES

- Claim:
  - Given an optimization problem π
    - the construction version (π_c) is at least as hard as the evaluation Version (π_e)
    - which in turn is at least as hard as the decision version (π_d)
  - i.e.
    - $\pi_d \precsim \pi_e \precsim \pi_c$
- Observation:
  - If the decision version of a problem is $\mathcal{NP}$-hard, then so is the construction version!

Optimization Problems: Complexity of Forms

- Theorem 1: For any problem π in NPO, $\pi_e \precsim_{poly} \pi_d$
- Proof:
  - By definition of NPO : for any x in I_π , and for any y in F(x): $m(x,y) <= 2^{p(|x|)}$ for some polynomial p
  - Using bisection (i.e. binary search over the range of m): π_e can be solved by at most p(|x|) queries to the oracle for π_d.

5/4/2016 Sundar B. CSIS, BITS, Pilani

Optimization Problems: Complexity of Forms

- Theorem 2: For any problem π in $\mathcal{NPO}$, if π_d is NP-complete, then $\pi_c \precsim_{poly} \pi_d$
- Proof:
  - We construct (see Lemma in next slide) a NPO problem π' such that
    - $\pi_c \precsim_{poly} \pi'_c$ and
    - $\pi'_c \precsim_{poly} \pi'_e$
  - and we know
    - $\pi'_e \precsim_{poly} \pi'_d$ (by Theorem 1 – see previous slide) and
    - $\pi'_d \precsim_{poly} \pi_d$ (since π_d is NP-complete)
  - Thus we can conclude
    - $\pi_c \precsim_{poly} \pi_d$ (by transitivity of reduction)

Optimization Problems: Complexity of Forms

- Lemma: For any problem π in NPO, there exists a problem π' in NPO such that $\pi_c \precsim_{poly} \pi'_e$
- Proof: (assume π is a max. problem).
  - Let p be a polynomial such that, for any x in I_π and for any y in SOL(x), $|y| <= p(|x|)$; and let $\lambda(y)$ be the rank of y in SOL(x), assuming lexicographic ordering.
  - Define a problem π' s.t. π' is the same as π (i.e. $I_\pi = I_{\pi'}$ and $SOL_\pi = SOL_{\pi'}$) except for the measure:
    - $m_{\pi'}(x) = 2^{p(|x|)+1} * m_\pi(x) + \lambda(y)$
    - This implies that
      - all feasible solutions for an instance of π' have unique measures and therefore the optimal solution for a given instance is unique.
      - and that given an instance if the solution is optimal for π' it is also optimal for π (i.e. $\pi_c \precsim_{poly} \pi'_c$).
    - The optimal solution for an instance x of π'_c can be constructed given the optimal measure:
      - Because $m^*_{\pi'}(x)$ mod $2^{p(|x|)+1}$ is $\lambda(y)$, the position – in lexicographic order – of the optimal solution in $SOL_\pi(x)$ (i.e. $\pi'_c \precsim_{poly} \pi'_e$).

5/4/2016 Sundar B. CSIS, BITS, Pilani

Optimization Problems: Complexity of Forms

- Theorem 3:
  - For any problem π in NPO, $\pi_c \precsim_{poly} \pi_e$
- Proof:
  - $\pi_c \precsim_{poly} \pi_d$ by Theorem 2, and
  - $\pi_d \precsim_{poly} \pi_e$ by definition.
  - So,
    - $\pi_c \precsim_{poly} \pi_e$ (by transitivity of reduction)

Optimization Problems – Decision Versions - Formulations

- Given an optimization problem **π = <I, F, m, goal>** its decision Version (π_d) can be formulated in two ways :
  - Formulation 1:
    - Given an input instance **x**, and a threshold value **k** decide whether the optimal measure is bounded by **k**
      - i.e. Is **m*(x) <= k** ? (if **goal** is *min*)
  - Formulation 2:
    - Given an input instance **x**, and a threshold value **k**, decide whether there exists a feasible solution whose measure is bounded by **k**
      - i.e. Is there a **y ϵ F(x) s.t. m(x,y) <= k** ? (if **goal** is *min*)
- Exercise:
  - Prove that both these formulations are equivalent!

OPTIMIZATION PROBLEMS – FORMULATIONS [5]

- Example: TSP_d
  - Formulation 1 :
    - Given a weighted, complete graph G, and a number k, find whether the weight of a minimum weight tour is less than k.
  - Formulation 2:
    - Given a weighted, complete graph G, a number k, find whether there is a tour of weight less than k.

$\mathcal{NP}$-HARD OPTIMIZATION PROBLEMS - EXAMPLES

- The following optimization problems are $\mathcal{NP}$-hard :
  - Min Vertex Cover
  - TSP
  - 0,1 Knapsack
- because their decision versions are $\mathcal{NP}$-hard.