ShanghaiTech University

CS240 Algorithm Design and Analysis

Fall 2024

Problem Set 3

Due: 23:59, Nov. 26, 2024

1. Submit your solutions to the course Gradescope.
2. If you want to submit a handwritten version, scan it clearly.
3. Late homeworks submitted within 24 hours of the due date will be marked down 25%. Homeworks submitted more than 24 hours after the due date will not be accepted unless there is a valid reason, such as a medical or family emergency.
4. You are required to follow ShanghaiTech's academic honesty policies. You are allowed to discuss problems with other students, but you must write up your solutions by yourselves. You are not allowed to copy materials from other students or from online or published resources. Violating academic honesty can result in serious penalties.

Notice: When proving that a problem A is NP-complete, you need to strictly follow the below steps and explicitly state each part.

- Prove that A is in NP.
- Choose an NP-complete problem B. For any instance b of B, construct an instance a of problem A in polynomial time.
- Prove that b has a solution if and only if a has a solution.

Problem 1:

Triple-SAT: Let Triple-SAT denote the following decision problem: given a Boolean formula ϕ, decide whether ϕ has at least three distinct satisfying assignments. Prove that Triple-SAT is NP-complete.

Solution:

Theorem. Triple-SAT is NP-complete.

Lemma. Triple-SAT is in NP.

Proof: For any input formula ϕ, we need only guess three distinct assignments and verify that they satisfy ϕ, which is polynomial time.

Reduction. SAT $\leq_{\text{p}}$ Triple-SAT.

Let ϕ_{SAT} denote the input Boolean formula to a SAT problem and suppose that the set of variables in ϕ_{SAT} are $X = \{x_1, \ldots, x_n\}$. We construct a Triple-SAT problem with a Boolean formula $\phi_{\text{Triple-SAT}}$ over a new variable set $X' = \{x_1, \ldots, x_n, y, z\}$.

- $X' = \{x_1, \ldots, x_n, y, z\}$.
- $\phi_{\text{Triple-SAT}} = \phi_{\text{SAT}}$.

Lemma. $\phi_{\text{Triple-SAT}}$ is satisfiable iff ϕ_{SAT} is satisfiable.

Proof:

$\leftarrow$

If ϕ_{SAT} is satisfiable, then we can augment any particular assignment by adding any of the 4 possible pairs of values for $\{y, z\}$ to give at least four satisfying assignments overall.

$\rightarrow$

Assume that $\phi_{\text{Triple-SAT}}$ has at least three satisfying assignments. Since y and z are the two newly added variables, if ϕ_{SAT} is unsatisfiable, then no matter how we set the values of y and z, $\phi_{\text{Triple-SAT}}$ cannot be satisfied. Therefore, if $\phi_{\text{Triple-SAT}}$ has at least three satisfying assignments, it implies that ϕ_{SAT} must be satisfiable.

Problem 2:

Given a set $F = \{f_1, f_2, \ldots, f_n\}$ and a system of subsets $\{S_1, S_2, \ldots, S_m\}$, where each $S_i \subseteq F$ satisfies $|S_i| \geq 2$. The objective is to assign each element in F one of two labels, "red" or "blue", so that for each subset S_i, there is at least one element labeled "red" and at least one element labeled "blue".

The task is to prove that determining whether such a labeling scheme exists to satisfy the above condition is an NP-Complete problem.

Solution:

To show that this problem is NP-Complete, we proceed in three steps.

Step 1: Prove the problem is in NP.

We prove this by providing a polynomial-time certifier. Given the input $F = \{f_1, f_2, \ldots, f_n\}$, $S_1, S_2, \ldots, S_m$, and a certificate: a coloring scheme, the certifier checks for each S_i whether it contains at least one element labeled "red" and at least one labeled "blue". This verification takes $O(mn)$ time, which is polynomial.

Step 2: Reduce 3-SAT to this problem.

Given an instance of 3-SAT with k variables $x_1, x_2, \ldots, x_k$ and p clauses $c_1, c_2, \ldots, c_p$, we construct an instance of this coloring problem as follows:

1. Set $n = 2k + 1$, with $f_i = x_i$ for $1 \leq i \leq k$, $f_i = \overline{x}_{i-k}$ for $k < i \leq 2k$, and $f_n = f_{2k+1} = B$, where $\overline{x}_i$ is the negation of x_i and B is a special element that will always be colored "blue".

2. Set $m = p + k$, and define:

$$S_i = \{x \mid x \text{ is a literal in } c_i\} \cup \{B\}, \quad \forall 1 \leq i \leq p$$

$$S_i = \{x_{i-p}, \overline{x}_{i-p}\}, \quad \forall p < i \leq p + k$$

3. Map a true assignment of a literal to "red" coloring of the corresponding element, and a false assignment to "blue" coloring. B is always colored blue.

This construction takes polynomial time since it requires iterating over variables and clauses.

Step 3: Prove that the answers for the two instances are equivalent.

(i) If the 3-SAT instance is satisfiable, then each clause has at least one true literal. Thus, for $1 \leq i \leq p$, S_i contains at least one red element and

at least one blue element (since $B \in S_i$). For $p < i \leq p + k$, $S_i = \{x_{i-p}, \overline{x}_{i-p}\}$ trivially satisfies the property, as x_{i-p} and $\overline{x}_{i-p}$ cannot both be true simultaneously.

(ii) Conversely, if there is a valid coloring for the instance of this problem, then each S_i contains at least one red element. Consequently, each clause in the 3-SAT instance has at least one literal assigned to true, thus satisfying the 3-SAT instance. The subsets $S_i = \{x_{i-p}, \overline{x}_{i-p}\}$ ensure that a variable and its negation are not both true simultaneously.

Since the problem is both in NP and can be reduced from 3-SAT (an NP-Complete problem), we conclude that this problem is NP-Complete.

Problem 3:

Given a finite set of items I, where each item $i \in I$ has a positive integer size $s(i) \in Z^+$, an integer bin capacity B, and an integer K. Is there a way to partition the set I into K disjoint subsets such that the sum of the sizes of items in each subset I_1, I_2, ..., I_i is less than or equal to B? Prove this problem is NP-complete.

Solution:

To prove that this problem is NP-complete, we typically follow two steps:

1. **Prove that the problem is in NP**.
2. **Prove that the problem is NP-hard** (usually by reducing a known NP-complete problem to this problem).

1. Proving the problem is in NP

To show that this problem is in NP, we need to demonstrate that we can verify a given solution in polynomial time:

- Given a solution (a partition of the item set I into $\{I_1, I_2, \ldots, I_K\}$), we can check the sum of the sizes in each subset I_j to confirm that it is less than or equal to B.
- This check can be completed in polynomial time by iterating through each subset and calculating the sum.

Thus, the problem is in NP.

2. Proving the problem is NP-hard

To prove that this problem is NP-hard, we can reduce it from the **3-Partition Problem**, which is a known NP-complete problem. The 3-Partition Problem is defined as follows:

- Given a set A containing $3m$ elements, each with a positive integer size, the goal is to partition A into m disjoint subsets such that each subset has an equal sum.

This problem is similar to the bin-packing problem, as we are also seeking a partition with a capacity constraint.

Reduction from the 3-Partition Problem

1. **Construct the Instance**: Suppose we have an instance of the 3-Partition Problem where the set A contains $3m$ elements, and the goal is to partition it into m subsets, each with a sum equal to a target T.

2. **Set Parameters**: For the bin-packing problem, we set the bin capacity $B = T$, the item set $I = A$, and the number of bins $K = m$.

3. **Equivalence**: The 3-Partition Problem requires partitioning the set A into m subsets, each with a total size of T, which is equivalent to finding m bins in the bin-packing problem where the sum of item sizes in each bin does not exceed $B = T$.

4. **Conclusion**: If we can solve this bin-packing problem in polynomial time, we could also solve the 3-Partition Problem, proving that this problem is at least as hard as the 3-Partition Problem, meaning it is NP-hard.

Since this problem is in NP and we have reduced a known NP-complete problem (3-Partition Problem) to it, we conclude that this bin-packing problem is **NP-complete**.