CS/ECE 374 A (Spring 2022)
Homework 11 Solutions

Problem 11.1: Consider the following problem COLORFUL-WALK (which is an extension of Problem 8.2(b) from HW8):

> *Input*: a directed graph $G = (V, E)$ with n vertices and m edges, a vertex $s \in V$, and a color $c(e) \in \{1, \ldots, k\}$ for each edge $e \in E$, and a number ℓ.
>
> *Output*: "yes" iff there exists a walk that starts at s and encounters at least ℓ distinct colors.

Consider the SET-COVER problem:

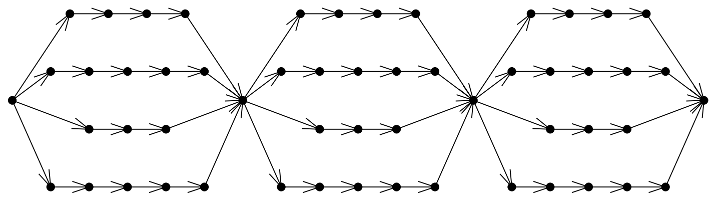
> *Input*: a set X of N elements, a collection of M subsets $\mathcal{S} = \{S_1, \ldots, S_M\}$ where each $S_i \subseteq X$, and a number K.
>
> *Output*: "yes" iff there exists a subcollection $\mathcal{T} \subseteq \mathcal{S}$ of K subsets, such that the union of the subsets in $\mathcal{T}$ includes all elements of X.

Describe a polynomial-time reduction from SET-COVER to COLORFUL-WALK. Follow these steps:

- Given an input to SET-COVER (i.e., X, $\mathcal{S}$, and K), describe a construction of an input to COLORFUL-WALK (i.e., G, s, $c(\cdot)$, and ℓ). Check that your construction takes polynomial time.
- Prove that the input to SET-COVER has a "yes" answer *if and only if* your input to COLORFUL-WALK has a "yes" answer.

Since SET-COVER is a well-known NP-hard problem, it would then follow that COLORFUL-WALK is NP-hard.

(Hint: for your construction of the directed graph G, try something like the picture below, with appropriate colors assigned to edges. Remember that in the actual description of your construction, you are given X, $\mathcal{S}$, and K; you are *not* given $\mathcal{T}$, which may or may not exist, since the goal is to determine whether the answer is "yes" or "no".)



Solution: We describe a polynomial-time reduction from Set-Cover to Colorful-Walk.

The Reduction. We are given an input to Set-Cover, i.e., a set X of N elements, a collection of M subsets $\mathcal{S} = \{S_1, \ldots, S_M\}$ where each $S_i \subseteq X$, and a number K. We want to construct an input to Colorful-Walk, i.e., a directed graph G, a vertex s, and a coloring $c(\cdot)$ of the edges, and a number ℓ.

Here is our construction. Without loss of generality (by relabeling), say $X = \{1, \ldots, N\}$.

- We define $k = N$ (i.e., the colors are X).
- For each $j \in \{1, \ldots, K+1\}$, we create a vertex s_j in G.
- For each $j \in \{1, \ldots, K\}$ and $i \in \{1, \ldots, M\}$, we create a new path $\pi_{i,j}$ from s_j to s_{j+1} in G. This path $\pi_{i,j}$ has length $|S_i|$ and the colors of the edges of $\pi_{i,j}$ are precisely the set S_i. (The order in which the colors appear along the path would not matter.)
- We define $s = s_1$.
- We define $\ell = N$.

(Note that the resulting graph G is a DAG, with source $s = s_1$ and sink s_{K+1}.)

Clearly, this construction can be carried out in polynomial time.

Correctness. We need to prove the following statement: there exists a subcollection $\mathcal{T} \subseteq \mathcal{S}$ of K subsets such that the union of the subsets in $\mathcal{T}$ includes all elements of X $\iff$ there exists a walk in G that starts at s and encounters at least ℓ distinct colors.

Proof of ($\Longrightarrow$): Suppose we have a subcollection $\mathcal{T} = \{S_{i_1}, \ldots, S_{i_K}\}$ of K subsets such that the union of the subsets in $\mathcal{T}$ includes all elements of X. We define a walk from $s = s_1$, where for each $j \in \{1, \ldots, K\}$, we go from s_j to s_{j+1} via the path $\pi_{i_j,j}$. The colors along this walk is precisely $S_{i_1} \cup \cdots \cup S_{i_K}$, which is all of X. So, the walk encounters all $\ell = N$ colors.

Proof of ($\Longleftarrow$): Suppose we have a walk τ in G that starts at $s = s_1$ and encounters at least ℓ distinct colors. We define a subcollection $\mathcal{T}$ as follows. For each $j \in \{1, \ldots, K\}$ and $i \in \{1, \ldots, M\}$, if the walk τ uses (all or some) edges from the path $\pi_{i,j}$, we add the subset S_i to the collection $\mathcal{T}$. Note that for each j, the walk τ can only visit at most one path from $\{\pi_{1,j}, \ldots, \pi_{M,j}\}$, so at most one subset S_i is added to $\mathcal{T}$ per j. Thus, the total number of subsets in $\mathcal{T}$ is at most K. We can make the number exactly K by adding extra subsets. Since the walk uses at least ℓ distinct colors, the union of the subsets in $\mathcal{T}$ must contain at least ℓ distinct colors. Since $\ell = N$, this means that the union of the subsets in $\mathcal{T}$ is all of X.

We conclude that Colorful-Walk is NP-hard.

Problem 11.2: By now, everyone has heard of Wordle, the online word-guessing game. Here, you will consider a tougher variant of the game, where the word length n is a variable, all strings of length n from a fixed alphabet are legal words (no dictionary needed!), and after each guess, you are only told whether there is a green (matching) position, but not the number of green (nor yellow) positions, nor where they are exactly. You will show that in this version of the game, just deciding whether there exists a solution after a series of guesses is already hard (so forget about the question of how to generate a good next guess!).

If you didn't follow the above paragraph, don't worry—here is the precise definition of our problem: For two strings y and z of length n, first define match(y, z) to be true if there exists a position k such that the k-th symbol of y is equal to the k-th symbol of z, and false otherwise. The statement of the Tough-Wordle problem is as follows:

> *Input*: a finite alphabet Σ, a list of m strings $y_1, \ldots, y_m \in \Sigma^n$, and m Boolean values $b_1, \ldots, b_m \in \{\text{true}, \text{false}\}$.
>
> *Output*: "yes" iff there exists a string $z \in \Sigma^n$ such that match$(y_i, z) = b_i$ for each $i = 1, \ldots, m$.

(Example: on the input with $\Sigma = \{A, B, C\}$, $y_1 = AAAA$, $b_1 = \text{true}$, $y_2 = BBCC$, $b_2 = \text{true}$, $y_3 = BCAB$, $b_3 = \text{false}$, the answer is "yes", e.g., by choosing $z = ABCC$.)

Describe a polynomial-time reduction from 3SAT to Tough-Wordle. Follow these steps:

- Given an input to 3SAT (i.e., a Boolean formula F in 3CNF form), describe a construction of an input to Tough-Wordle (i.e., an alphabet Σ, and a list of strings and Boolean values). Check that your construction takes polynomial time.

- Prove that F is satisfiable *if and only if* your input to Tough-Wordle has a "yes" answer.

Since 3SAT is NP-hard, it would then follow that Tough-Wordle is NP-hard.

(Hint: in your construction, an alphabet of size 3 (0, 1, and an extra symbol) would be sufficient. Suppose F has n variables and m clauses. Intuitively, the string z (if exists) should correspond to a satisfying assignment of F (if exists). For each clause C_i, construct a string y_i of length n (and a corresponding Boolean value b_i) to somehow make sure that z contains at least one true literal of C_i. Also, construct an extra string to somehow make sure that z uses only 0s and 1s and not the extra symbol... Remember that in the actual description of your construction, all you are given is F; you are *not* given a satisfying assignment, which may or may not exist, since the goal is to determine whether the answer is "yes" or "no".)

Solution: We describe a polynomial-time reduction from 3SAT to Tough-Wordle.

The Reduction. We are given an input to 3SAT, i.e., a Boolean formula F in 3CNF form, say, with n variables $x_1, \ldots, x_n$ and m clauses $C_1, \ldots, C_m$. We want to construct an input to Tough-Wordle, i.e., an alphabet Σ, a list of strings $y_1, \ldots, y_{m'}$, and a list of Boolean values $b_1, \ldots, b_{m'}$.

Here is our construction. We choose $\Sigma = \{0, 1, 2\}$. For each clause C_i ($i \in \{1, \ldots, m\}$), we define a string $y_i = y_{i1} y_{i2} \cdots y_{in}$ of length n, with

$$y_{ij} = \begin{cases} 1 & \text{if } x_j \text{ is a literal in the clause } C_i \\ 0 & \text{if } \overline{x_j} \text{ is a literal in the clause } C_i \\ 2 & \text{otherwise.} \end{cases}$$

(As the clause C_i has length three, the string y_i consists mostly of 2's, except in three places.) We set the corresponding Boolean value b_i = true.

To finish the construction, we define one additional string $y_{m+1} = 22\cdots 2$ consisting of n 2's and let b_{m+1} = false. (The total number of strings is thus $m' = m + 1$.)

Clearly, this construction can be carried out in polynomial time.

Correctness. We need to prove the following statement: F is satisfiable $\Longleftrightarrow$ there exists a string $z \in \{0, 1, 2\}^n$ such that match$(y_i, z) = b_i$ for each $i = 1, \dots, m + 1$.

Proof of ($\Longrightarrow$): Suppose F is satisfied by some truth assignment $\mathcal{A}$ of the variables $x_1, \dots, x_n$.

Define a string $z = z_1 \cdots z_n$ of length n, where z_j is set to 1 if x_j is assigned to true in $\mathcal{A}$, and 0 otherwise.

We claim that match$(y_i, z) = b_i$ for every $i \in \{1, \dots, m + 1\}$. To see this, take a clause C_i ($i \in \{1, \dots, m\}$). Since C_i is satisfied by $\mathcal{A}$, some literal in C_i must be assigned to true. If this literal is x_j, then both $z_j = 1$ and $y_{ij} = 1$; if this literal is $\overline{x_j}$, then both $z_j = 0$ and $y_{ij} = 0$. In either case, match(y_i, z) = true. To finish, observe that match(y_{m+1}, z) = false, since z does not contain any 2's.

Proof of ($\Longleftarrow$): Suppose $z \in \{0, 1, 2\}^n$ is a string such that match$(y_i, z) = b_i$ for each $i \in \{1, \dots, m + 1\}$. In particular, match$(y_{m+1}, z)$ = false, implying that z consists of only 0's and 1's. Define an assignment $\mathcal{A}$ where x_j is set to true if $z_j = 1$, and false otherwise.

We claim that F is satisfied by $\mathcal{A}$. To see this, take a clause C_i ($i \in \{1, \dots, m\}$). Since match(y_i, z) = true, we must have $y_{ij} = z_j$ for some position j. If $z_j = 1$, then x_j is set to true in $\mathcal{A}$ and appears in C_i; if $z_j = 0$, then $\overline{x_j}$ is set to true in $\mathcal{A}$ and appears in C_i. In either case, the clause C_i is satisfied by $\mathcal{A}$.

We conclude that TOUGH-WORDLE is NP-hard.