**15-455: UCT** **K. Sutner**

Assignment 5 Due: **February 25**, 2022.

## 1. Paths and Trees (20)

**Background**

We know that Hamiltonian Cycle is $\mathbb{NP}$-complete. A very closely related problem is

| | |
|---|---|
| Problem: | **Hamiltonian Path** |
| Instance: | A ugraph $G = \langle V, E \rangle$. |
| Question: | Does $G$ have a Hamiltonian path? |

Recall that a spanning tree of a ugraph $G = \langle V, E \rangle$ is a subgraph $T = \langle V, E_0 \rangle$ such that $T$ is a tree (connected, acyclic). Consider the following instances: the graph $G$ together with either a bound $k$, $1 \leq k \leq n = |V|$ (for 1, 2, 3), or a vertex set $L \subseteq V$ (for 4, 5). The questions are as follows: Is there a spanning tree $T$ of $G$ such that

1. $T$ has $k$ leaves.
2. $T$ has at most $k$ leaves.
3. The nodes of $T$ have degree at most $k$.
4. The set of leaves of $T$ is $L$.
5. There are no leaves of $T$ outside $L$.

**Task**

A. Show that Hamiltonian Path is $\mathbb{NP}$-complete.

B. Show that the five spanning tree problems from above are $\mathbb{NP}$-complete.

**Extra Credit:** Show that problem (3) is hard for any *fixed* $k \geq 2$.

## 2. Tame Turing Machines (40)

**Background**

In a nondeterministic Turing machine the transitions are given by a relation

$$\delta \subseteq Q \times \Sigma \times Q \times \Sigma \times \{0, \pm 1\}$$

It is often convenient to clean things up a bit and insist that that the branching factor of all computation trees is 2 everywhere. Also, we want all branches to be of the same length. We call such a machine tame. You can think of the transition relation in a tame machine as the union of two transition functions

$$\delta_i : Q \times \Sigma \to Q \times \Sigma \times \{0, \pm 1\}$$

This is particularly helpful when dealing with probabilistic machines.

**Task**

A. Show that any nondeterministic polynomial time Turing machine $\mathcal{M}$ there is a tame polynomial time Turing machine $\mathcal{M}'$ that accepts the same language.

B. Roughly how much of a slow-down should one expect? Using multiple tapes is fine.

C. Explain how to modify the proof of Cook-Levin given in class to handle the case when the $\mathbb{NP}$-set $A$ is given by a nondeterministic Turing machine rather than a projection.

**Comment** For part (C), just explain where the argument differs from the one given in class.

## 3. Tilings (40)

**Background**

Informally, a Tiling Problem consists of a collection of square tiles with colored edges. We want to place the tiles on an infinite chess board in a way that the colors of adjacent edges match, filling up the whole board in the process. The tiles cannot be rotated, only shifted; we assume an unlimited supply of tiles of each type. For the infinite chessboard the problem of whether a set of tiles admits a tiling is undecidable; the proof rests on encodings of a computation of a Turing machine and is quite messy.

We are here interested in finite versions of the problem: we are given the tiles plus a board of size $n \times n$. Note: $n$ is in unary here! Technically, the tiles are given as $\langle T, C\rangle$ where $C$ is a finite set of colors and $T \subseteq C^4$ where $t = (a, b, c, d) \in T$ means that the north/east/south/west edge of tile $t$ is colored $a$, $b$, $c$ and $d$, respectively. We write $N(t)$, $E(t)$, $S(t)$ and $W(t)$ for the colors of a tile $t$.

Suppose $S = [n] \times [n]$ is the $n \times n$ grid. A tiling of $S$ is an admissible placement of tiles, more precisely, a map $\pi : S \to T$ such that

$$\begin{aligned} N(\pi(i,j)) &= S(\pi(i,j+1)) \qquad 1 \le i \le n, 1 \le j < n, \\ E(\pi(i,j)) &= W(\pi(i+1,j)) \qquad 1 \le i < n, 1 \le j \le n. \end{aligned}$$

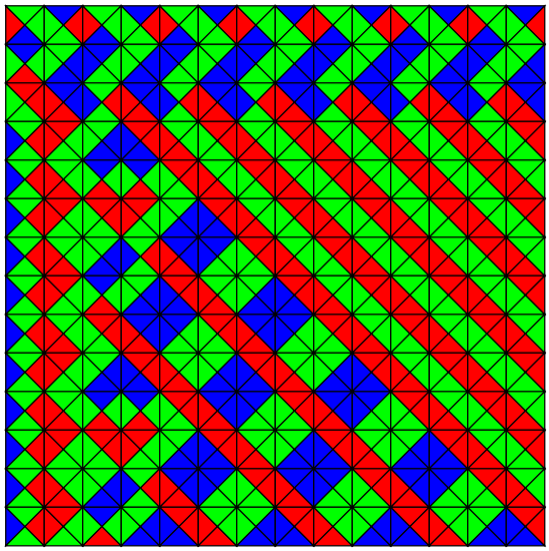
We will consider a version of the problem where we are given two tiles $t_{\mathrm{SW}}$ and $t_{\mathrm{NE}}$ that are supposed to be placed in the south-west and north-east corner of the grid, respectively. An instance of our tiling problem thus consists of the tiles $\langle T, C\rangle$, the grid size $n$, coded as $0^n$, and two tiles $t_0$ and $t_1$. Let's call this pointed square tiling (PST).

**Task**

A. Show that pointed square tiling problem is in $\mathbb{NP}$.

B. Show that pointed square tiling problem is $\mathbb{NP}$-complete.

**Comment** Here is an example of what one of these tilings might look like.



Hardness of tiling problems is a well-established topic, see e.g. R. Berger (1966), "The Undecidability of the Domino Problem". The author uses a tile set of insane cardinality 20,426.