Ros Roger, Alexandre

May 12th, 2023

20. Puzzle. Consider the $n \times n$ sliding puzzle which consists of a frame with $n \times n$ with n^2 tiles. $n^2 - 1$ tiles hold numbers from 1 to $n^2 - 1$. The frame has one empty tile and this enables the others to move horizontally and vertically. The puzzle is solved if all numbers are row sorted. Show that the problem of deciding whether a sliding puzzle can be solved in $\leq k$ moves, parameterized by k, belongs to `FPT`.

Solution:

Let Ω represent an arbitrary state (tile configuration) of the sliding puzzle. For every state Ω, let $\texttt{Next}(\Omega) = \{\omega \mid \Omega \xrightarrow{1} \omega\} \setminus \{\Omega\}$ be the set of states accessible by exactly one move. Let $\Omega \xrightarrow{k} \omega$ denote the fact that we can reach ω, starting from Ω, with k moves or less.

Our algorithm can be described by the following observations, with inputs Ω (initial state), k (maximum moves allowed) and Γ (final state, we may assume there is only one final state):

If $k = 1$, then the problem boils down to computing $\texttt{Next}(\Omega)$ and checking whether $\Gamma \in \texttt{Next}(\Omega)$ or $\Gamma = \Omega$, which can be accomplished in polynomial time w.r.t. the size of Ω. More precisely, in $O(n^2)$ time, for we only need to find the empty tile and yield the configurations that arise by moving an adjacent tile to the empty slot. Note that, since a slot can be adjacent to at most four other slots, $2 \leq \#\texttt{Next}(\Omega) \leq 4$.

If $k \neq 1$, then the input $\langle \Omega, k, \Gamma \rangle$ is accepted by our algorithm if, and only if,

$$\Omega = \Gamma \;\; \textbf{or} \;\; \exists \omega \in \texttt{Next}(\Omega) \text{ s.t. } \langle \omega, k-1, \Gamma \rangle \text{ is accepted by our algorithm}$$

This is because $\Omega \xrightarrow{k} \Gamma \iff \Omega = \Gamma \;\vee\; \exists \omega \in \texttt{Next}(\Omega) \text{ s.t. } \omega \xrightarrow{k-1} \Gamma$.

This is a recursive definition to our algorithm. In this last case, the additional costs (excluding the recursive calls to our algorithm) are also polynomial w.r.t. the size of the board, since we only need to compute $\texttt{Next}(\Omega)$.

The time complexity of our algorithm can be expressed by the following two equations:

$$T(1) = O(n^2) \tag{1}$$

$$T(k) = 4\,T(k-1) + O(n^2) \tag{2}$$

Solving the equations above, our algorithm has a time complexity bounded by $O(4^k \times n^2)$, and belongs to `FPT`, since k is parameterized.

■