Individual Homework 8

(1a) On input $s$, we remember whether the first symbol is an "a" or a "b" [we do this by having a subset of states to which we transition to upon reading an "a," and a disjoint subset of states upon reading a "b"]. We then accept or reject upon reading the last symbol [on a Turing machine this means that we move right until reaching a blank symbol, $\_\_$, and then move left one cell].
\( \Sigma = \{ a, b \}, \quad \Gamma = \{ a, b, \uparrow \} \)

\( \delta \) is described above

\( q_0, q_{\text{accept}}, q_{\text{reject}} \) are \( q_0, q_{\text{accept}}, q_{\text{reject}} \) in \( Q \)

(this is a good convention to use)

[There is a lot of flexibility: for example, when transitioning to \( q_{\text{accept}} \), the R/L is irrelevant.]
(1c)  
$q_0 a a b$

\[ \text{[You can add any finite number of } \_\text{'s to the right of these configuration descriptions, if you like.]} \]

\[
q_1 a a b
\]
\[
a a q_1 a b
\]
\[
a a b q_1 a
\]
\[
a a q_2 e a b
\]
\[
a a b q_3 e
\]

To see the role of the \_ (blank) at the end of the input, you could also write

\[
q_0 a a b _
\]
\[
q_1 a a b _
\]
\[
a a q_1 a b _
\]
\[
a a b q_1 a _
\]
\[
a a q_2 e a b _
\]
\[
a a b q_3 e _
\]
(1a) If \( S = \sigma_1 \ldots \sigma_n \) with \( \sigma_i \in \{a,b\} \)

we begin:

step/config 1: \( q_0 \sigma_1 \ldots \sigma_n \)

step/config 2: \( \sigma_1 q \sigma_2 \ldots \sigma_n \)

\[
\begin{cases}
\sigma = q_1 a \\
\text{or } q_1 b
\end{cases}
\]

step/config \( n \): \( \sigma_1 \ldots \sigma_{n-1} q \sigma_n \)

step/config \( n+1 \): \( \sigma_1 \ldots \sigma_n q_L \)

step/config \( n+2 \): \( \sigma_1 \ldots q' \sigma_n q_L \) \( q' = q_{\text{end}, a} \)

step/config \( n+3 \): \( \ldots q_{\text{accept/\text{reject}} \ldots} \) or \( q_{\text{end}, b} \)

Total # of steps = \( n+3 - 1 = n+2 \)

\[
\begin{cases}
\text{It is OK to also say } n+3 \text{ steps,} \\
\text{since the last step is step # } n+3.
\end{cases}
\]
Group Homework 8

(1) We can recognize \( L \) by running a universal Turing machine on \( \langle M, \emptyset \rangle \), accepting \( \langle M, \emptyset \rangle \) if \( M \) accepts \( \emptyset \).

To prove that \( L \) is undecidable, assume that some Turing machine, \( P \), decides \( L \). Then we claim that we can decide

\[
\text{ACCEPTANCE} = \{ \langle M, w \rangle \mid w \text{ accepts} \}.
\]

Indeed, given \( \langle M, w \rangle \), we can build a Turing machine \( P' \) that constructs \( M' \) such that

1. \( M' \) erases its input
2. \( M' \) writes \( w \) on its input tape, and
3. \( M' \) simulates \( M \) with these new symbols on its input tape,
4. Having built \( M' \), \( P' \) now gives \( P \)
\( \langle M' \rangle \) as input, and therefore checks if \( \langle M' \rangle \in \mathbb{L} \).

We have \( M' \in \mathbb{L} \iff \langle M, \omega \rangle \in \text{ACCEPTANCE} \), and hence \( T' \) decides ACCEPTANCE.

This contradicts the fact that ACCEPTANCE is undecidable.

Since \( L \) is recognizable but undecidable, the complement of \( L \) is unrecognizable.

NOTE: Problem (1) and (2) are essentially the same in the context of Turing machines or in the context of Python programs.
(2) Given \( \langle M, q \rangle \) we can tell what is

\[ \Sigma = \{1, \ldots, n, \Sigma\}, \quad n \in \mathbb{N}. \]

Now write \( \Sigma^* \) in a list:

\[ \omega_1, \omega_2, \ldots \]

For \( k = 1, 2, \ldots \), consider an algorithm that

\[ \begin{cases} 
\text{simulates } \langle M \rangle \text{ for } k \text{ steps on each} \\
\text{of } \omega_1, \ldots, \omega_k, \text{ and we halt and} \\
\text{accept if any configuration enters} \\
\text{state } q 
\end{cases} \]

If state \( q \) of \( Q \) is reached by some input, then this algorithm will detect this for some value of \( k \), and therefore accept \( M \).
If not, then this algorithm will never halt. Hence this algorithm recognizes \( L \).

Let us show that \( L \) is undecidable:
Say that \( L \) is decided by a \( T, M, P \).

Let us show that we can decide \( \text{ACCEPTANCE} \):
Let \( P' \) be the algorithm that builds \( M' \) as in Problem 1, and gives \( \langle M', q_{\text{accept}} \rangle \) as input to \( L \). Then \( \langle M', q_{\text{accept}} \rangle \in L \) iff \( M \) accepts \( w \). Hence this shows that \( \text{ACCEPTANCE} \) is undecidable, which is impossible.
Hence $L$ is undecidable.

Since $L$ is recognizable but undecidable, the complement of $L$ is unrecognizable.
(3a) \( \langle G \rangle \) contains the symbols \#n_1\#n_2 for each edge \( \{N_1, N_2\} \in E \), hence at 4 symbols per edge. Hence the length of \( \langle G \rangle \), \( n \), is at least \( 4|E| \). So \( n \geq 4|E| \) so \( |E| \leq n/4 \).

(3b) If \( G \in \text{HALF-CLIQUE} \), then for some \( |V'| \geq |V|/2 = N/2 \), \( G \) has at least \( \binom{|V'|}{2} = \frac{|V'|(|V'|-1)}{2} \geq \frac{(N/2)(N/2-1)}{2} = \frac{N(N-2)}{8} \), and hence \( \langle G \rangle \) is of length \( \geq \frac{N(N-2)}{8} \).
(3c) Consider all subsets \( E' \subset E \), of which there are \( \leq 2^{n/4} \). For each \( E' \), we can form the subset, \( V' \), of vertices that are endpoints of the edges in \( E' \).

\[
\begin{align*}
\text{For example, if } E' \text{ is } \{1,2\}, \{7,8\}, \{1,7\}, \\
\text{then } V' \text{ is } \{1,2,7,8\}
\end{align*}
\]

Then we can check if
\[
\begin{align*}
(1) & \quad |V'| \geq |V|/2, \quad \text{and} \\
(2) & \quad \text{if } E' \text{ consists of all pairs of elements of } V'.
\end{align*}
\]

For any \( E' \), the time needed to
find $V'$ and check (1), (2) is polynomial in $|E'|$. Since $|E'| \leq |E| \leq n/4$, these operations require time polynomial in $n/4$.

The time it takes to generate all $E' \subset E$ plus this time per each $E'$ is therefore

$$\text{polynomial}(n) \cdot 2^{n/4}.$$