Question 1

Let $A = \{w \in \{0,1\}^* |\ w$ has the same number of 0s and 1s$\}$.

Required: Describe a Turing Machine M whose accepted set is A and halts on all inputs.

Part 1: Defining TM M

We will define a TM M for A, and then show that M is a decider for A.

Let $\Sigma = \{0,1\}$ be the alphabet.

Let $\Gamma = \Sigma \cup \{\dot{x} : x \in \Sigma\} \cup \{X, \dot{X}, \sqcup\} = \{0, 1, \dot{0}, \dot{1}, X, \dot{X}, \sqcup\}$ be the tape alphabet.

The rest of our Turing Machine M, including the transitions, will be given by an ordinary English description.

$M =$ "On input w:

1. If $w = \epsilon$, accept. Otherwise, go to Step 2.

2. Mark the first symbol of w with its dotted version. The dotted symbol will serve to mark the leftmost cell of the tape. Any symbol that replaces this leftmost cell will also be a dotted symbol.

3. Move the head to the leftmost cell of the tape. Scan right until the first 0 or $\dot{0}$ is read. Replace this 0 or $\dot{0}$ with an X or $\dot{X}$ respectively and go to Step 4. If a blank symbol $\sqcup$ is read before reading a 0 or $\dot{0}$, go to step 5.

4. Move the head to the leftmost cell of the tape. Scan right until the first 1 or $\dot{1}$ is read. Replace this 1 or $\dot{1}$ with an X or $\dot{X}$ respectively and go to Step 3. If a blank symbol $\sqcup$ is read before a 1 or $\dot{1}$, reject.

5. Move the head to the leftmost cell of the tape. Scan right and check if each cell has an X or $\dot{X}$ until a blank $\sqcup$ is reached. If every cell read had an X or $\dot{X}$, accept. Otherwise, reject."

Part 2: Showing M is a decider for A

To show that M is indeed a decider for A, we will show that for every $w \in A$, M accepts, and for every $w \notin A$, M rejects.

Case 1: Let $w \in A$. If $w = \epsilon$, then M will correctly accept. If $w \neq \epsilon$, then $|w| > 0$. Each time M replaces a 0 with an X (or their dotted versions), M attempts to replace a 1 with an X (or their dotted versions). Since w has an equal number of 0s and 1s, the entire string

w will be replaced with Xs in pairs, and will correctly accept in Step 5.

From hereafter, we will omit discussions involving dots for readability, since the dot only serves to mark the leftmost cell of the tape.

Case 2: Let $w \notin A$. Since $w \notin A$, we have that $w \neq \epsilon$. Hence, $|w| > 0$. Since w does not have an equal number of 0s and 1s, either there are more 0s than 1s, or more 1s than 0s. If there are more 0s than 1s, eventually we will replace a 0 with an X in Step 3, but we will not have a corresponding 1 to replace with an X in Step 4, and hence we will correctly reject on Step 4. If there are more 1s then 0s, once we reach a blank in Step 3, we will move onto Step 5 which will identify that there are still some 1s left on the tape, and hence correctly reject.

Since there only finitely many 0s and 1s on the tape, either all the 0s will eventually be changed to Xs or all the 1s will eventually be changed to Xs, or both. Hence, M will always accept or reject. Hence, M will always halt and is a decider.

Therefore, we have constructed a decider M for our set A, as required.

Question 2

Required: Show that if a set A is Turing recognizable, then so is A^*

Proof. Let A be a Turing recognizable set. Hence, there exists a TM M that recognizes A.

Now we will define a non-deterministic TM M' that recognizes A^*.

Notice that for any non-empty input w, we can partition w into 1 or more nonempty strings $w = w_1w_2, ..., w_k$. Since w is of finite length, there is only a finite number of possible partitions of w into 1 or more nonempty strings. We will implicitly use this fact in our non-determinism.

So consider the following nondeterministic TM M' defined as follows.

$M' =$ "On input w:

1. If $w = \epsilon$, accept.

2. If $w \neq \epsilon$, then for each partition $w = w_1w_2, ..., w_k$ of w into 1 or more nonempty strings chosen non-deterministically: run M on input w_i for each $i \in \{1, ..., k\}$ in parallel. If M accepts w_i for all $i \in \{1, ..., k\}$, then M' accepts w."

To show that M' recognizes A^*, consider an arbitrary input $w \in A$. If $w = \epsilon \in A^*$, then M' accepts. If $w \neq \epsilon$, then since $w \in A^*$, we know that w has the form $w = w_1, w_2, ..., w_k$ where $w_i \in A$ for all $i \in \{1, ..., k\}$. Since this is a partition of w into 1 or more strings, we know that M' will evaluate this particular partition non-deterministically. Since M recognizes A, we have that M accepts w_i for all $i \in \{1, ..., k\}$. Hence, M' will accept w.

And if $w \notin A^*$, then there is no valid partition of w into strings of A. Hence, M' will not be able to non-deterministically find an accepting partition. Hence, M' will not accept.

Therefore, M' is a non-deterministic TM that recognizes A^*.

By Corollary 3.18, we know that a language is recognizable if and only if some non-deterministic TM recognizes it. Hence, A^* is recognizable, completing the proof, as required. □

Question 3

Show: Every decidable language is accepted by a TM with left reset which halts on all inputs.

Proof. Let L be a decidable language. Hence, there exists some TM M that decides L. We will construct a TM with left reset denoted M' which decides L.

Let $M = \langle Q, \Sigma, \Gamma, \sigma, q_0, q_{accept}, q_{reject} \rangle$.

Define $M' = \langle Q', \Sigma, \Gamma', \sigma', q_0, q_{accept}, q_{reject} \rangle$ where $\Gamma' = \Gamma \cup \{\dot{a} : a \in \Gamma\}$ and $\sigma' : Q' \times \Gamma' \to Q' \times \Gamma' \times \{R, RESET\}$ is a transition function that simulates the transition function $\sigma : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$. We will describe σ' shortly.

Note that we are going to define M' so that is simulates M.

For any right transition in M's transition function σ, define σ' to have the same transition. Now we need to simulate the left transitions of σ in σ' using only $RESET$ and R and we're done.

Consider a left transition in σ. If the head is at the leftmost cell of the tape, then a left transition here will just be $RESET$ (i.e. effectively no head movement). So assume that the head is currently NOT at the leftmost cell of the tape. Consider the following.

Simulation of Left-Transition:

1. Replace the symbol in the current cell with its dotted version and $RESET$ to the leftmost cell.

2. Replace the symbol in the leftmost cell with its dotted version. Move the head one cell to the right.

3. If this cell's symbol has a dot, then the previous cell was the desired cell. Replace the current cell's symbol with its undotted version, $RESET$, and move right until you reach a dot. This is the cell to the left of the original cell. Replace the dotted symbol in this cell with its undotted version. This completes the simulation of a left transition. Otherwise, ignore this step and move to step 4.

4. If this cell's symbol does not have a dot, then replace the symbol with its dotted version. Then $RESET$ and move right until a cell with a dotted symbol is reached. Replace the symbol in this cell with its undotted version. Then $RESET$ again and move right until you reach a cell with a dotted symbol. Move the head one cell to the right. Go back to step 3.

The above simulation works by dotting the current cell and dotting the leftmost cell. And the simulation proceeds by moving the dot on the leftmost cell closer and closer until it is

one cell to the left of the original cell. After erasing the dots, we complete our simulated left transition.

Hence, we are now able to simulate left transitions using just $RESET$ and R. Combining this with our earlier work, our Left-Reset TM M' can now simulate our ordinary TM M.

We know that M decides L. Therefore, M' also decides L since M' precisely simulates M. Hence, M' halts on all inputs.

Therefore, we have shown that any decidable language is accepted by a TM with left reset which halts on all inputs, as required. $\square$

Question 4

Required: Prove that every infinite Turing recognizable language has an infinite subset that is decidable.

Proof. Let A be an infinite Turing recognizable language.

By Theorem 3.21, we know that a language is Turing recognizable if and only if some enumerator enumerates it.

Since A is Turing recognizable, we know that there is some enumerator E that enumerates A.

Convention: We consider $0 \in \mathbb{N}$.

Now, consider the output that E prints. Let w_0 be the first string in A that E prints. We have that the length of w_0 is $|w_0| = j$ for some $j \in \mathbb{N}$.

We know that there are only a finite number of strings in A of length less than or equal to j. Since A is infinite, E must eventually output some string w_1 such that $|w_1| > |w_0| = j$. We will continue this process to get an infinite set $B \subseteq A$ such that $B = \{w_n : n \in \mathbb{N}\}$ where $|w_{i+1}| > |w_i|$ for all $i \in \mathbb{N}$. We will define $B \subseteq A$ as follows.

Condition 1: $w_0 \in B$ where w_0 is the first string printed by E.

Condition 2: For each $n \in \mathbb{N}$, if $w_n \in B$, then $w_{n+1} \in B$ where w_{n+1} is the first string printed by E after printing w_n such that $|w_{n+1}| > |w_n|$.

Condition 3: B contains no other strings.

We know that B is well-defined because for each $i \in \mathbb{N}$, there is only a finite number of strings of length less than or equal to i. Since A is infinite, E must always eventually print a string of length greater than i.

Now we will show that there is an enumerator E' that enumerates B. Let E' be defined as follows.

$E' =$ "ignore input w:

1. Run the enumerator E. Call the first string that E prints w_0. Let E' also print w_0.

2. For each string w_n that E and E' had both already printed out, let w_{n+1} be the **first** string that E prints after w_n such that $|w_{n+1}| > |w_n|$. Let E' also print w_{n+1}."

Notice, E' prints out all strings in $B = \{w_n : n \in \mathbb{N}\}$ in index order where $|w_n| < |w_{n+1}|$ for each $n \in \mathbb{N}$. Hence, E' enumerates B in Standard String Order (SSO) as defined below.

Def: An ordering of strings is in Standard String Order (SSO) if they are first ordered by length, and strings of the same length are ordered by some alphabetical ordering.

Note: Standard String Order (SSO) is also known as lexicographical ordering. In Tutorials, we used the term Standard String Order (SSO).

In Tutorials, we proved the following Lemma. We state only one direction of the Lemma since that is all we need here. The converse also holds, which we also proved in Tutorials.

Tutorial Lemma: Let L be a language. If there is an enumerator F that enumerates L in Standard String Order (SSO), then L is decidable.

We have shown that our infinite language $B \subseteq A$ has some enumerator E' that enumerates B. And E' prints the strings of B in increasing order by length. And every string in B has different lengths. Hence, E' prints the strings in B in SSO. By our **Tutorial Lemma**, we have that B is decidable.

So we have shown that our infinite Turing-recognizable language A, has an infinite subset B that is decidable. This is what we needed to show, as required. □

Extra: Proof of Tutorial Lemma

On Piazza, it was indicated that we did not need to prove claims that were proven in Tutorials and Lectures. But I wrote the proof below before reading that post. The proof below is optional to the grader.

Please see the next page if you would like to see the proof of the **Tutorial Lemma**.

Tutorial Lemma: Let L be a language. If there is an enumerator F that enumerates L in Standard String Order (SSO), then L is decidable.

To prove our lemma, let L be a language with an enumerator F that enumerates L in SSO.

If L were a finite language, then the Lemma trivially holds since we know from the textbook that every finite language is decidable. So let's assume L is infinite.

Consider the following decider D for L which we assume is an infinite language.

$D =$ "on input w:

1. Run F.

2. For each string s that F prints out: compare s with w. If $s = w$, then accept. If $s > w$ (in SSO), then reject."

To see that D is indeed a decider for L, if L contains w, then F will eventually print w since F prints in SSO, and there are only finitely many strings with length less than or equal to w printed before w. Hence, D will correctly accept. And if L does not contain w, then F will never print w, and F will eventually print a string s such that $s > w$ in SSO. Hence, in this case D will correctly reject. So in every input w, D will eventually accept or reject. Hence, D is a decider for L. This completes the proof of the **Lemma**.