Name: SOLUTIONS

Problem 1: True/False. Correct =2, incorrect = -2, no answer = 0.

(T) F Each context-free language is decidable
Given a context-free grammar and a word w, there is an algorithm to decide whether the grammar generates w.

T (F) There is an undecidable language L whose complement $\bar{L}$ is context-free
If $\bar{L}$ is context-free, then $\bar{L}$ is decidable, so $\bar{\bar{L}} = L$ must also be decidable.

(T) F If L and $\bar{L}$ are Turing-recognizable then L is decidable
Given an input w, simulate the TM for L and the TM for $\bar{L}$ on w until one of them accepts. If the TM for L accepts, accept. If the TM for $\bar{L}$ accepts, reject.

T (F) If L is Turing recognizable then so is $\bar{L}$
From the previous question, we know that if L and $\bar{L}$ are Turing-recognizable, then L is decidable. So if this were true, every Turing-recognizable language would be decidable. But we know this is not the case (e.g. A_{TM} is recognizable but not decidable.)

(T) F If L is decidable then $\bar{L}$ is decidable
To decide $\bar{L}$ do the following. On input w, simulate the decider for L on input w. If that decider accepts, reject. If that decider rejects, accept. (We know it will always halt.)

(T) F If L and L' are Turing-recognizable then so is $L \cap L'$
To recognize $L \cap L'$, do the following. On input w, run the TM M_L for L on w. If it accepts, run the TM $M_{L'}$ for L' on w. If it also accepts, then accept.

Problem 2: Prove or disprove: *the following language is decidable:*

$$L = \{\langle M, w\rangle : \text{TM } M\text{, on input } w\text{, makes an odd number of transitions.}\}$$

L is not decidable. To prove this, we will show that if L is decidable, then $HALT_{TM}$ is decidable. Since we know $HALT_{TM}$ is not decidable, we conclude that L is not either.

Assume that L is decidable, and let M_L be a TM that decides it. Construct the following TM M_H to decide $HALT_{TM}$:

M_H on input $\langle M, w\rangle$:

1. Construct a TM M' by modifying M so that M' makes an extra transition at the start. (Do this by adding a new start state s', and make each transition from s' go to the old start state without changing the tape.) *Note that M halts on w iff $\langle M', w\rangle \in L$ or $\langle M, w\rangle \in L$.* 2. Use M_L to decide whether each of $\langle M\rangle$ and $\langle M'\rangle$ are in L. 3. If either is, ACCEPT, else REJECT.

claim: *(assuming M_L decides L) M_H decides $HALT_{TM}$.*
proof of claim:

Suppose M halts on w. Then either M or M' make an odd number of transitions on input w. So M_H accepts $\langle M, w\rangle$. Thus, M_H accepts every instance of $HALT_{TM}$.

On the other hand, suppose M does not halt on w. Then neither does M'. So (by the construction of M_H) M_H rejects $\langle M, w\rangle$. Thus, M_H rejects every string not in $HALT_{TM}$.

This proves the claim.

We have shown that, if M_L is decidable, so is $HALT_{TM}$. Since $HALT_{TM}$ is not decidable, M_L cannot be either.

Problem 3: Prove or disprove: *the following language is decidable:*

$$L = \{\langle D\rangle : \text{DFA } D\text{, on some input } w\text{, visits each one of its states.}\}$$

L is decidable.

Proof 1:

1. A DFA has an input that makes it visit all its states if and only if the DFA, considered as an directed graph, has a path p that starts at the start vertex (state) and visits all the nodes.

2. If there is such a path p, and the DFA has n states, then there is such a path p' of length at most $n^2 + n$. (To see this, consider tracing the path p from the start vertex. Every time you trace a cycle, if that cycle does not visit a vertex that hasn't yet been visited, then delete the cycle from p. When you are done, the path p minus the deleted cycles form a path p' that starts at the start vertex and visits all the vertices. Furthermore, p' has length at most $n^2 + n$: every cycle in p' visits some vertex that wasn't visited before, so there are at most n cycles, and each cycle has length at most n.)

3. So here is an algorithm that decides L:

1. Consider each path p of length $n^2 + n$ or less in the graph of the DFA. 2. If any such path visits all states of the DFA, then accept, else reject.

Proof 2:

The following TM decides L:

1. For each state s of D, construct a DFA D_s by modifying D so that s is the only accept state, and every transition from s leads back to s again. That is, construct D_s so that $L(D_s)$ contains those strings that cause the DFA to enter state s at some point. 2. Construct a DFA D' such that $L(D') = \cap_s L(D_s)$, using the standard construction for taking the intersection of regular languages. (Here the intersection is taken over all states s.) Then $L(D')$ is the set of all strings that cause D to enter *every* state at least once. 3. Test whether $L(D') = \emptyset$ using the algorithm from the book (look for any path from the start state to an accept state). 4. If $L(D') = \emptyset$, then REJECT, else ACCEPT.

Problem 4: Let $\{0,1\}^\infty$ denote the set of countably infinite sequences of zeros and ones. For example, $000000000\ldots$, $11111111\ldots$, and $010101010\ldots$ are members of the set.

Prove or disprove: *The set $S = \{0,1\}^\infty$ is countably infinite.*

The set S is uncountable (that is, not countable).

Consider any 1-1 mapping f from the natural numbers to the set S. (So $f(i)$ is a string in S coresponding to i.)

Consider the string x where the ith bit of x is $1 - f(i)_i$ — where $f(i)_i$ is the ith character of the string $f(i)$. This string is in S, yet it differs from each string $f(i)$.

Thus, any such mapping f leaves out at least one string in S. Thus, S cannot be countable.