B501 Assignment 5
Enrique Areyan

Due Date: Friday, March 30, 2012
Due Time: 11:00pm

1. (10 points) Show that decidable languages are closed under union.

   Solution: Let languages L_1 and L_2 be decidable. We need to show that $L_1 \cup L_2$ is also decidable, i.e., that a decider U can be built to decide $L_1 \cup L_2$. We assume that there is a decider U_1 for L_1 and a decider U_2 for L_2. The proof is by construction. Here is the construction for U:

   Machine U on input x:
   1. Run U_1 on input x.
   2. If U_1 accepts, *accept*.
   3. If U_1 rejects x, then run machine U_2 on input x.
   5. If U_2 accepts, *accept*; if U_2 rejects, *rejects*.

   Machine U accepts only every input that is accepted by either U_1 or U_2 and thus $L(U) = L(U_1) \cup L(U_2) = L_1 \cup L_2$. Moreover, machine U always halts on any input making it a decider.

2. (10 points) Show that decidable languages are closed under intersection.

   Solution: Let languages L_1 and L_2 be decidable. We need to show that $L_1 \cap L_2$ is also decidable, i.e., that a decider U can be built to decide $L_1 \cap L_2$. We assume there is a decider U_1 for L_1 and a decider U_2 for L_2. The proof is by construction. Here is the construction for U:

   Machine U on input x:
   1. Run U_1 on input x.
   2. If U_1 rejects, *rejects*.
   3. If U_1 accepts x, then run machine U_2 on input x.
   5. If U_2 accepts, *accept*; if U_2 rejects, *rejects*.

   Machine U accepts only every input that is accepted by both U_1 and U_2 and thus $L(U) = L(U_1) \cap L(U_2) = L_1 \cap L_2$. Moreover, machine U always halts on any input making it a decider.

3. (10 points) Show that Turing-recognizable languages are closed under union.

   Solution: Let languages L_1 and L_2 be Turing-recognizable. We need to show that $L_1 \cup L_2$ is also Turing-recognizable, i.e., that a TM U can be built to decide $L_1 \cup L_2$. We assume that there is a TM U_1 that recognizes

L_1 and a TM U_2 that recognizes L_2 and try to build U as follows:

Machine U is similar to the Universal Turing Machine Ω, but we need to expand it to work on the description of both machines U_1 and U_2 in parallel. Therefore, machine U on input $< U_1, U_2, w >$ does the following:

Machine U will have 6 tapes. The first 3 tapes will be used for machine U_1 (just as in Ω), and the last 3 for U_2. In the very first tape there will be a description of machine U_1; the second tape keeps track of the current state of machine U_1 and the third tape has a copy of input w and keeps track of the configurations. The other three tapes contain this same information but for U_2.

Machine U will use the descriptions of U_1 and U_2 to operate both machines simultaneously according to each description. Machine U will accept, reject or loop on the input if, operating according to the descriptions of U_1 and U_2, it occurs at any point in time that the content of states' tapes (tape number 2 or 4), is as follow:

Tape 2	Tape 4	U
accept	accept	*accept*
accept	reject	*accept*
reject	accept	*accept*
reject	reject	*reject*
accept	loop	*accept*
reject	loop	*loop*
loop	accept	*accept*
loop	reject	*loop*
loop	loop	*loop*

In short, it suffices that one of the machines accepts the input for U to accept. Note that loop is not actually a state, only a short hand to express that the machine never stops. Because U accepts w whenever U_1 or U_2 accepts, it follows that $L(U) = L(U_1) \cup L(U_2) = L_1 \cup L_2$. Note that machine U may loop and thus is not a decider. Therefore, U is a TM that recognizes $L_1 \cup L_2$ and thus, Turing-recognizable languages are closed under union.

4. (10 points) Show that Turing-recognizable languages are closed under intersection.

   Solution: The proof here is the same as the proof for the union above, but changing the table as follow:

Tape 2	Tape 4	U
accept	accept	*accept*
accept	reject	*reject*
reject	accept	*reject*
reject	reject	*reject*
accept	loop	*loop*
reject	loop	*reject*
loop	accept	*loop*
loop	reject	*reject*
loop	loop	*loop*

In short, the only way for U to accept input w is if both U_1 and U_2 to accept it. Note that loop is not actually a state, only a short hand to express that the machine never stops. Because U accepts w whenever U_1 and U_2 accepts, it follows that $L(U) = L(U_1) \cap L(U_2) = L_1 \cap L_2$. Note that machine U may loop and thus is not a decider. Therefore, U is a TM that recognizes $L_1 \cap L_2$ and thus, Turing-recognizable languages are closed under intersection.

5. (10 points) Define a language that is neither Turing-recognizable nor co-Turing-recognizable.

   Solution:

   $EQ_{TM} = \{< M_1, M_2 > | M_1 \text{ and } M_2 \text{ are Turing Machines and } L(M_1) = L(M_2)\}$

6. (10 points) Let A and B be two disjoint languages. Say that language C *separates* A and B if $A \subseteq C$ and $B \subseteq \overline{C}$. Show that any two disjoint co-Turing-recognizable languages are separable by some decidable language.

   Solution: Preliminaries: Let A and B be two co-turing recognizable languages. By definition, there exists turing machines that recognizes the complement of these languages, let us call these: TM $M_{\bar{A}}$ and $M_{\bar{B}}$. Note that $L(M_{\bar{A}}) = \bar{A}$ and $L(M_{\bar{B}}) = \bar{B}$. Also assume that $A \cap B = \emptyset \iff \bar{A} \cup \bar{B} = \Sigma^*$

   Proof: The proof is by construction, i.e., I will construct a machine M, which is a decider and separates A and B. Such machine works as follow:

   $M =$ "on input w :
   1. Run both machines $M_{\bar{A}}$ and $M_{\bar{B}}$ in parallel on input w.
   2. If $M_{\bar{B}}$ accepts w, *accept*.
   3. If $M_{\bar{A}}$ accepts w, *reject*."

   From the preliminaries we know that any string in Σ^* is either on $\bar{A}$ or $\bar{B}$. Machine M uses recognizers for $\bar{A}$ and $\bar{B}$ and thus, will halt on every input, i.e., it is a decider.

Moreover, because A and B are disjoints, any string in A is in $\bar{B}$, i.e., $A \subseteq \bar{B}$. Every string in $\bar{B}$ is accepted by M, and thus, $A \subseteq L(M)$. Conversely, any string in B is in $\bar{A}$, i.e., $B \subseteq \bar{A}$, but every string in $\bar{A}$ is rejected by M; thus, B is not in the language of M.

It follows that $L(M) = C$ is a decider that separates A and B.