Tutorial 7: Turing machines all around

---

1. **Turing machines and halting programs**

   Let $\Sigma = \{a, b\}$ and $L \subseteq \Sigma^*$ be a language over Σ.

   - Define PREFIX(L) $:= \{w \mid \exists y \in \Sigma^*$ such that $wy \in L\}$. Thus, PREFIX(L) is the the set of all prefixes of all words in L.
   - Define HALF(L) $:= \{w \mid \exists y \in \Sigma^*$ such that $|w| = |y|$ and $wy \in L\}$. Thus, HALF(L) is the set of all first-halves of words of even length in L.

   (a) Your best friend has constructed a Turing machine M, such that its halting language $H(M) = L$. Show how to build a Turing machine that has PREFIX(L) as its halting language. Your Turing machine can use M as a sub-machine.

   (b) Your second best friend has constructed a Turing machine M' that enumerates all and only the words in L on an output tape. Show how to build an enumerator for HALF(L) using M.

   [Hint: Try to construct a simple pseudo-code (with loops, if statements, assignments, jump statements etc.) that achieves the desired task. Then think about whether every construct in this pseudo-code can be converted into a (possibly multi-tape) Turing machine, all of which can then be "strung" together to give you an overall Turing machine that achieves the desired task.]

2. **Closure properties of languages accepted by Turing machines**

Recall from our discussion in class that a language L is *recursively enumerable* if there exists a Turing machine M such that its halting language $H(M)$ equals L. Recall also that we discussed that for every such language, we can build another Turing machine that takes no input, but enumerates all and only the strings in L on an output tape.

Finally, recall from our class discussions that language L is *recursive* if there is a Turing machine M' with two designated states q_Y and q_N such that (a) $H(M') = \Sigma^*$, (b) for all $w \in L$, the machine M' halts in state q_Y when it is started with w on its tape, and (c) for all $w \notin L$, the machine M' halts in state q_N when it is started with w. In other words, M' halts on all inputs, but it halts in q_Y for every string in L, and halts in q_N for every string not in L.

Let RE be the class of all recursively enumerable languages, and let R be the class of recursive languages.

(a) Is RE closed under Kleene closure? Is R closed under Kleene closure?

(b) A homomorphism is defined by a mapping $h : \Sigma \to \Sigma^*$. With abuse of notation, we use h to also denote a mapping from strings to strings as follows: $h(\varepsilon) = \varepsilon$ and $h(a_1.a_2.\ldots.a_k) = h(a_1).h(a_2).\ldots.h(a_k)$, where each $a_i \in \Sigma$ and $a_1.a_2.\ldots.a_k \in \Sigma^*$.

Is RE closed under homomorphisms? Is R closed under homomorphisms?

*[***Food for thought:** *Define the inverse homomorphism of a language* $L \subseteq \Sigma^*$ *as* $h^{-1}(L) = \{w \mid w \in \Sigma^*, h(w) \in L)\}$. *Is* RE *closed under inverse homomorphism? What about* R*?]*

3. **Let's decide (if we can)**

   Find whether the following problems are decidable.

   - Given a Turing machine M, a state q and a string w, does M when started on the string w reach the state q?
   - Given a Turing Machine M, does it halt on all strings in less than 100 transitions?
   - Given two Turing machines M_1 and M_2, do they accept the same language?

4. **Take-away problem: Enumerating recursive languages**

   Show that every infinite recursively enumerable language has an infinite subset that is a recursive language.

5. **Take-away problem: Neither r.e. nor co-r.e.**

For purposes of this question, let M_i denote the Turing machine encoded by the binary representation of the natural number i, and let $H(M_i)$ denote the language accepted by M_i by halting. Let $L = \{1^i 0^j \mid i, j \in \mathbf{N}, H(M_i) \text{ is a strict subset of } H(M_j)\}$. In other words, $1^i 0^j \in L$ iff every word in $H(M_i)$ is also in $H(M_j)$, but there is at least one word in $H(M_j)$ that is not in $H(M_i)$.

1. Show that L is not recursively enumerable.
2. Show that L is not co-recursively enumerable either.

6. **Take-away problem: Almost undecidable, but not quite!**

Let $\langle M \rangle$ denote the encoding of Turing Machine M as a string over $\Sigma = \{0, 1\}$, as studied in class. Let w be a string in $\{0, 1\}^*$ and let $L = \{(\langle M \rangle, w) \mid M$ when run with w as the initial string on its tape visits less than $|w|$ tape squares$\}$.

(a) Show that L is decidable.

(b) Give an upper bound of the time complexity (i.e. count of steps taken by a halting Turing machine) of deciding if a given $(\langle M \rangle, w)$ pair is in L. Your complexity expression should be a function of $|\langle M \rangle|$ and $|w|$.