School of Computing and Information Systems
COMP30026 Models of Computation
Week 11: Turing-Decidable and Turing-Recognisable Languages

Exercises

T11.1 Here is how we can see that the class of decidable languages is closed under intersection. Let A and B be decidable languages, let M_A be a decider for A and M_B a decider for B. We construct a decider for $A \cap B$ as a Turing machine which implements this routine:

> On input w:
> (1) Run M_A on input w and reject if M_A rejects.
> (2) Run M_B on input w and reject if M_B rejects; else accept.

Show that the class of decidable languages is closed under union.

Continued in P11.1, P11.2 & P11.3

T11.2 The class of Turing-recognisable languages is closed under intersection, and the construction we gave in the previous question, for decidable languages, can equally be used to prove this. (Convince yourself that the argument is still right, even though we now have no guarantee that M_A and M_B always terminate.)

The class of Turing-recognisable languages is also closed under union, but we can't argue that the same way, by constructing a Turing machine which effectively runs M_A and M_B, one after the other. (Why not?)

Show that the class of Turing-recognisable languages is closed under union.

T11.3 Show that the language $ALL_{DFA} = \{\langle A \rangle \mid A \text{ is a DFA and } L(A) = \Sigma^*\}$ is decidable.

T11.4 Let M be a Turing machine with alphabet Σ and $w \in \Sigma^*$ a string. Define a language A containing only the single string s, where

$$s = \begin{cases} \mathtt{1} & \text{if } M \text{ halts on input } w \\ \mathtt{0} & \text{if } M \text{ does not halt on input } w \end{cases}$$

Is A decidable? Why or why not?

Homework problems

P11.1 Show that the class of decidable languages is closed under complement. Why can't we use the same argument to show that the class of Turing recognisable languages is closed under complement?

P11.2 Show that the class of decidable languages is closed under concatenation.

P11.3 Show that the class of decidable languages is closed under Kleene star.

P11.4 Show that the problem of whether the language of a DFA is empty, is decidable. That is, show that the language

$$E_{DFA} = \{\langle D\rangle \mid D \text{ is a DFA and } L(D) = \emptyset\}$$

is decidable. *Hint:* write pseudocode for an algorithm which analyses the graph of the DFA, and argue that your algorithm will not run forever on any input DFA $\langle D\rangle$.

P11.5 Show that the problem of whether the language of a CFG is empty, is decidable. That is, show that the language

$$E_{CFG} = \{\langle G\rangle \mid G \text{ is a CFG and } L(G) = \emptyset\}$$

is decidable. *Hint:* write pseudocode for an algorithm which analyses the rules of the CFG, and argue that your algorithm will not run forever on any input CFG $\langle G\rangle$.

P11.6 Consider the alphabet $\Sigma = \{0, 1\}$. The set Σ^* consists of all the *finite* bit strings, and the set, while infinite, turns out to be countable. (At first this may seem obvious, since we can use the function $binary : \mathbb{N} \to \Sigma^*$ defined by

$$binary(n) = \text{ the binary representation of } n$$

as enumerator; however, that is not a surjective function, because the legitimate use of leading zeros means there is no unique binary representation of n. For example, both 101 and 00101 denote 5. Instead the idea is to list all binary strings of length 0, then those of length 1, then those of length 2, and so on: $\epsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100, 101, \ldots$.)

Now consider instead the set $\mathcal{B}$ of *infinite* bit strings. Show that $\mathcal{B}$ is much larger than Σ^*. More specifically, use diagonalisation to show that $\mathcal{B}$ is not countable.

P11.7 (a) Let A be a DFA with n states. Show that $L(A) = \emptyset$ iff A does not accept any string of length at most $n - 1$.

(b) Argue that the following "brute force" algorithm decides E_{DFA}, and that E_{DFA} is thus decidable:

> On input DFA A with n states:
> (1) For each string w of length at most $n - 1$:
> (2) Run A on input w and reject if A accepts.
> (3) Otherwise, accept.

P11.8 Let G be a context-free grammar in Chomsky normal form and let $w \in L(G)$. Show that every derivation of w in G has $2|w| - 1$ steps. (Or equivalently, that every parse tree of w in G has $2|w| - 1$ internal nodes.)