School of Computing and Information Systems
COMP30026 Models of Computation Tutorial Week 12

Content: reducibility, mapping reducibility

Practice Problems

P12.1 Show that $A_{DFA} \leq_m A_{CFG} \leq_m A_{TM}$.

Solution:
Intuitively, this should be correct as all regular languages are context-free, and all context-free languages are T-recognisable.
In order to actually prove it, we first give a mapping from A_{DFA} to A_{CFG}, $f_1(\langle D, w\rangle) = \langle G, w\rangle$ where $L(G) = L(D)$. Given a DFA, D, we construct an equivalent CFG, G, through the following steps. First, let the terminals of G be the alphabet of D, and add a variable Q_i to G for each state q_i in D. If the starting state for D is q_0, then the starting variable for G is Q_0. For each transition from state q_n to state q_m given symbol v in D, add rule $Q_n \to vQ_m$ to G. The final step is to, for each accept state q_a of D, add the rule $Q_a \to \varepsilon$ to G. Each of these steps is computable, and CFG G has the same language as DFA D, thus $A_{DFA} \leq_m A_{CFG}$.

Now to prove $A_{CFG} \leq_m A_{TM}$, let $f_2(\langle G, w\rangle) = \langle D_{Gw}, \epsilon\rangle$ where D_{Gw} ignores its input, then runs D_{A-CFG} on $\langle G, w\rangle$. Recall that D_{A-CFG} is the decider for A_{CFG} given in lecture 19. This mapping is computable, so $A_{CFG} \leq_m A_{TM}$. Therefore we have proven $A_{DFA} \leq_m A_{CFG} \leq_m A_{TM}$ as required.

P12.2 Earlier, we gave a mapping reduction from an undecidable problem ALL_{CFG} to EQ_{CFG} and concluded that EQ_{CFG} is undecidable. Show that the language

$$\overline{EQ_{CFG}} = \{\langle G, H\rangle \mid G \text{ and } H \text{ are grammars and } L(G) \neq L(H)\}$$

is Turing-recognisable. Conclude that EQ_{CFG} is in fact not Turing-recognisable.

Solution:
Recall from lectures that A_{CFG} is decidable. Let M be a decider for A_{CFG}. We construct a recogniser R for $\overline{EQ_{CFG}}$ as follows: For each string s in Σ^* in lexicographic order, if $M(\langle G, s\rangle) \neq M(\langle H, s\rangle)$, accept. If the two grammars are different, then there exists a string on which they disagree so R will eventually accept. This is a general reduction from $\overline{EQ_{CFG}}$ to A_{CFG}, but since R is a recogniser not a decider, it only proves $\overline{EQ_{CFG}}$ is T-recognisable rather than decidable.

We know from lectures that if A and $\overline{A}$ are both T-recognisable, then A is decidable. Since we now know that $\overline{EQ_{CFG}}$ is T-recognisable, if EQ_{CFG} is T-recognisable then it should also be decidable. However, we proved earlier that EQ_{CFG} is undecidable, and so it cannot be T-recognisable.

P12.3 Let $T = \{\langle M\rangle \mid M \text{ is a TM that accepts } w^R \text{ whenever it accepts } w\}$. Show that T is undecidable.

Solution:
Recall that A_{TM} is undecidable. Let Turing machine M_{wR} behave as follows:
"On input x, if $x \neq w$ and $x \neq w^R$, reject. If $x = w^R$, accept. If $x = w$, simulate M on w".
If M accepts w, then $L(M_{wr}) = \{w, w^R\}$ and so $M_{wR} \in T$. If M does not accept w, then $L(M_{wR}) = \{w^R\}$ and so $M_{wR} \notin T$. Then $f(\langle M, w\rangle) = \langle M_{wR}\rangle$ is a computable mapping from A_{TM} to T, and so $A_{TM} \leq_m T$. Since A_{TM} is undecidable, T must also be undecidable.

P12.4 Show that if A is Turing-recognisable and $\overline{A} \leq_m A$, then A is decidable.

Solution: Note that the earlier version of this question had a typo where it gave $A \leq_m \overline{A}$ instead of $\overline{A} \leq_m A$. We know from lectures that if a problem A and $\overline{A}$ are both T-recognisable, then A is decidable. Since $\overline{A} \leq_m A$ and A is T-recognisable, then $\overline{A}$ is T-recognisable. Thus, A is decidable.

Student Learning Survey

- What was good? What can be improved?
- Mention tutor/lecturer name specifically
- Mid-Sem Survey Feedback not yet received
- We take feedback seriously. Based on student feedback, we have
  - Removed Haskell
  - Included more resources for discrete math and informal proofs
  - Submitted a handbook change proposal to increase tutorials from 1-hr to 2-hrs
  - Added weekly quizzes to help you catch misunderstandings/misconceptions early
- William:
  - Added illustrations, examples and Python equivalents to make arguments more concrete
  - Added motivation
  - Alternative proof of undecidability

https://unimelb.bluera.com/unimelb/