University of Victoria

CSC 320 - Spring 2023

Foundations of Computer Science

Tutorial 08

Teaching Team

Learning Outcomes:

- Construct high level Turing Machines.
- Prove that a language is decidable by construction.

Interesting Article:

"Non-Erasing Turing Machines: A New Frontier Between A Decidable Halting Problem and Universality" [1]
"Strongly Universal Quantum Turing Machines and Invariance of Kolmogorov Complexity" [2]

March 14th, 2023

Question 01

Prove that the following language is decidable by constructing (high level) Turing Machines which decide the language.

$$A_{DFA} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$$

Let T be the decider for A_{DFA}. ← our claim!

T = "On input $\langle B, w\rangle$:

1. Simulate w on B.
2. If end in an accept state, accept.
   If end in a non-accept state, reject."

We can now use a proof of correctness to prove our decider T for A_{DFA}.

<u>Proof of Correctness</u>

Take any string w.

$\Longrightarrow$ a) Suppose B accepts w. Then w ends on an accept state, ↙ by definition!
so T accepts $\langle B, w\rangle$.

$\Longleftarrow$ b) Suppose T accepts $\langle B, w\rangle$. Then w ended on an accept state,
so B accepts w.

$\Longrightarrow$ c) Suppose B does not accept w.
Write the proof yourself!

<u>Note</u>: It is important to go both ways!

$\Longleftarrow$ d) Suppose T rejects $\langle B, w\rangle$.
Write the proof yourself!

Furthermore, since B is a DFA, we will never run into an infinite loop when simulating it, so T will always halt. ▲

Question 02

Prove that the following language is decidable by constructing (high level) Turing Machines which decide the language.

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

Let $\mathcal{E}$ be the decider for $\mathcal{E}_{DFA}$

$\mathcal{E}$ = " On input A :

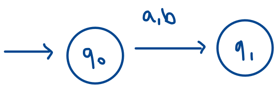
1. Mark the start state.
2. Mark each state with an incoming transition from a marked state.
3. Repeat step 2 until no new states are marked.
4. If an accept state is marked, reject.
   If no accept states are market, accept."

Proof of Correctness

We note that accept state(s) will only be marked if and only if (iff) we can reach the accept state(s) by following transitions from th start state (and no accept is marked iff we cannot reach any accept state from th start state).

Furthermore, th language of a DFA is empty iff we cannot reach any accept state from th start (and the language of the DFA is non-empty iff we can reach an accept state).

Idea



, where $\mathcal{E}_{DFA} = \{\{q_0, q_1\}, \{a,b\}, \delta, q_0, \{\}\}$

DFA $\mathcal{E}_{DFA}$

Symbol for therefore ⟶ $\therefore$ If N is a DFA, then...

$L(N) = \emptyset$ iff $\mathcal{E}$ accepts.

$L(N) \neq \emptyset$ iff $\mathcal{E}$ rejects.

Lastly, $\mathcal{E}$ always halts since N is a DFA and there are a finite amount of states. ∎

Question 03

Prove that the following language is decidable by constructing (high level) Turing Machines which decide the language.

$$EQ_{DFA} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}$$

Consider the following:

We need to show both cases!

If $L(A) = L(B)$, then

- $(L(A) \cap \overline{L(B)}) = \emptyset$ (i.e., everything in $L(A)$ must be in $L(B)$)
- $(\overline{L(A)} \cap L(B)) = \emptyset$ (i.e., everything in $L(B)$ must be in $L(A)$)

So,

$$(L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B)) = \emptyset$$

Let P be the decider for EQ_{DFA}.

P = "On input $\langle A, B\rangle$:

1. Construct DFA C such that $L(C) = (L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B))$

   Note: We know C can be constructed since regular languages are closed under intersection, union, and complement.

2. Run E on input $\langle C\rangle$.

3. If E accepts, accept

   If E rejects, reject." ∎

Remember that $L(C) = \emptyset$ iff $L(A) = L(B)$.

Thus, E accepts iff $L(A) = L(B)$.

Thus, P accepts iff $L(A) = L(B)$.

Bonus

$A_{TM} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$

Let S recognize A_{TM}.

S = "On input $\langle M, w\rangle$:

1. Simulate M on input w.

2. If M reaches an accept state, accept.
   If M reaches a reject state, reject." ■

Bonus: Reductions in English

Problem A: $L(A) = \{ J \mid J$ is a person and J moves to Germany$\}$

Problem B: $L(B) = \{ F \mid F$ is a TM and F accepts only cats$\}$

We will reduce A to B!

Note: Obviously there is no real relationship between A and B. But often in reductions, you still need to find a way to relate two seemingly unrelated problems, so this is valid.

We create a turing machine C.

C = "On input x:

1. If x is a cat, accept.
2. If x is not a cat:
   a) If P moves to Germany, reject.
   b) If P does not move to Germany, accept."

What is the language of C if P moves to Germany?

if P doesn't move to Germany?

We can use this to solve $L(A) = \{ P \mid P$ moves to Germany$\}$!

TM A = "On input P.

1. Construct TM C.

   C = "On input x:
   1. If x is a cat, accept.
   2. If x is not a cat:
      a) If P moves to Germany, reject.
      b) If P does not move to Germany, accept."

2. Run F on input ⟨C⟩.
   a) If F accepts, accept.
   b) If F rejects, reject. "

Resources

[1] M. Margenstern, “Non-erasing turing machines: A new frontier between a decidable halting problem and universality,” English, in *LATIN ’95: Theoretical Informatics*, G. Goos, J. Hartmanis, J. van Leeuwen, R. Baeza-Yates, E. Goles, and P. V. Poblete, Eds., vol. 911, Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 386–397, ISBN: 978-3-540-59175-7. DOI: 10.1007/3-540-59175-3_104. [Online]. Available: http://link.springer.com/10.1007/3-540-59175-3_104.

[2] M. Muller, “Strongly universal quantum turing machines and invariance of kolmogorov complexity,” English, *IEEE Transactions on Information Theory*, vol. 54, no. 2, pp. 763–780, 2008, ISSN: 0018-9448. DOI: 10.1109/TIT.2007.913263. [Online]. Available: http://ieeexplore.ieee.org/document/4439860/.