CS Theory (Fall '25) Assigned: Nov 2, 2025

Homework 4 Solutions

Instructors: *William Pires and Toniann Pitassi* Due: Nov 13, 2025 at 11am

0 Exercises

Here are some **highly** recommended exercises. You should not turn in your solutions for these, and we will not grade them.

(a) Prove that for every infinite set S, the following are equivalent:

(i) There exists a function $g : \mathbb{N} \to S$ that is onto (i.e., g is surjective).

(ii) There exists a function $f : S \to \mathbb{N}$ that is one-to-one (e.g., f is injective).

Solution.

Proof. (i) $\implies$ (ii): Assume there exists a function $g : \mathbb{N} \to S$ that is onto. Define the function $f : S \to \mathbb{N}$ so that for $s \in S$,

$$f(s) = \min g^{-1}(s),$$

where g^{-1} is the inverse image. In other words, f maps $s \in S$ to the smallest natural number $n \in \mathbb{N}$ such that $g(n) = s$. The preimage on any $s \in S$ is nonempty because g is surjective, plus a nonempty set of natural numbers will always have a minimum by the well ordering principle. Therefore f is well-defined. Furthermore, observe that $g(f(s)) = s$ for all $s \in S$ by definition. Now to show that f is injective, consider any $s_1, s_2 \in S$ such that $f(s_1) = f(s_2)$. This implies that $g(f(s_1)) = g(f(s_2))$. But $g(f(s_1)) = s_1$ and $g(f(s_2)) = s_2$ as noted earlier, so $s_1 = s_2$.

(ii) $\implies$ (i): Assume there exists a function $f : S \to \mathbb{N}$ that is one-to-one. Since f is injective, its inverse image on any natural number n must have size at most 1. Otherwise, if $f^{-1}(n)$ contains distinct elements $s_1, s_2 \in S$, then $f(s_1) = f(s_2) = n$ but $s_1 \neq s_2$, a contradiction. In fact, the inverse image of any injective function must be empty or a single point set. Therefore we can define the function $g : \mathbb{N} \to S$ as follows: for each $n \in \mathbb{N}$

$$g(n) = \begin{cases} s & \text{if } f^{-1}(n) = \{s\} \\ s_0 & \text{if } f^{-1}(n) = \emptyset \end{cases}$$

where s_0 is some fixed element we choose from S. To show g is surjective, consider any $s \in S$. Since f maps $s \in S$ to a natural number, $f(s) = n$ for some $n \in \mathbb{N}$. Therefore $f^{-1}(n) = \{s\}$ and $g(n) = s$. □

(b) Are the following languages (i) decidable or (ii) undecidable. Prove your answer.

- $L = \{\langle M \rangle \mid M \text{ is a TM and for every even length } \ell, \text{ there is a string } w, |w| = \ell, \text{ s.t. } M \text{ halts on } w\}$

Solution.

L is undecidable.

Proof. We show a Turing reduction $A_{TM} \leq_t L$, in class we've shown that A_{TM} is undecidable so this proves L is not decidable. Let N be a decider for L. We can construct a decider for A_{TM} as follows:

Algorithm 1 A decider D for A_{TM}

Input: $\langle M, w \rangle$ where M is a TM and w is a string

1: (Note: Technically the input string is not always a valid encoding, but we can fix something not in L and return it as output for this case. *For the purpose of this course, you can always assume the input has the right encoding.*)
2: Consider the TM M' defined as follows:
"On input x,
1. If x has odd length, *accept* x.
2. If x has even length:
3. Run M on input w. ▷ If M loops on w, we get "stuck" here and M' loops on x
4. If M accepted w, accept x.
5. If M rejected w, loop forever. " ▷ This means M' doesn't halt on x.
3: Run N on $\langle M' \rangle$. ▷ N always halts because it *decides* L
4: If N accepted, accept $\langle M, w \rangle$.
5: If N rejected, reject $\langle M, w \rangle$.

[a] We now show D is a decider for A_{TM}. We first observe the following: For any string x with $|x|$ being even, we can see that:

- If M loops on w, then M' loops on w (on line 3).
- If M rejects on w, then M' loops on w (on line 5).
- If M accepts w, then M' halts and accepts x (on line 4).

So, if M accepts w, M' halts on every string of even length. If M doesn't accept w, then M' loops on all strings of even length. So we have that $\langle M' \rangle \in L$ if and only if M accepts w.

(1) If $\langle M, w \rangle \in A_{TM}$, then M accepts w, so $\langle M' \rangle \in L$. So N accepts $\langle M' \rangle$ on line 3, and D accepts $\langle M, w \rangle$ on line 4.

(2) If $\langle M, w \rangle \notin A_{TM}$, then M doesn't accept w, so $\langle M' \rangle \notin L$. So N rejects $\langle M' \rangle$ on line 3, and D rejects $\langle M, w \rangle$ on line 5.

Therefore, D always halts and accepts $\langle M, w \rangle$ iff $\langle M, w \rangle \in A_{TM}$. So D decides A_{TM}, meaning $A_{TM} \leq_t L$. □

[a] Small note: to compute a description of M' from the description of M and w we simply needs to add extra states to M and to check the length of the input x. So step 1 always halts

- $L = \{\langle M \rangle \mid$ M is a TM and $L(M)$ is finite and has size divisible by 3 $\}$

Solution.

L is undecidable.

Proof. We show a Turing reduction $A_{TM} \leq_t L$, in class we've shown that A_{TM} is undecidable so this proves L is not decidable. Let N be a decider for L. We can construct a decider for A_{TM} as follows:

Algorithm 2 A decider D for A_{TM}

Input: $\langle M, w \rangle$ where M is a TM and w is a string

---

1: (Note: Technically the input string is not always a valid encoding, but we can fix something not in L and return it as output for this case.)
2: Build a TM M' as follows:
"On input x,
1. If x is not ϵ, 0, nor 1, *reject* x.
2. Else if x is ϵ or 0, *accept* x.
3. Else: ▷ this means $x = 1$
4. Run M on input w.
5. If M accepted w, accept x.
6. If M rejected w, reject x. "
3: Run N on $\langle M' \rangle$. ▷ N always halts because it *decides* L
4: If N accepted, accept $\langle M, w \rangle$.
5: If N rejected, reject $\langle M, w \rangle$.

We now show D is a decider for A_{TM}. We first observe the following:

- M' always accepts ϵ and 0 (on line 2).
- M' accepts 1 if and only if M accepts w (on line 5).
- M' rejects every other strings (on line 1).

So if M accepts w we have $L(M') = \{\epsilon, 0, 1\}$ and if M doesn't accept w we have $L(M') = \{\epsilon, 0\}$. By looking at the two cases for $|L(M')|$ we have $\langle M' \rangle \in L$ if and only M accepts w.

(1) If $\langle M, w \rangle \in A_{TM}$, then M accepts w, so $\langle M' \rangle \in L$. So N accepts $\langle M' \rangle$ on line 3, and D accepts $\langle M, w \rangle$ on line 4.

(2) If $\langle M, w \rangle \notin A_{TM}$, then M doesn't accept w, so $\langle M' \rangle \notin L$. So N rejects $\langle M' \rangle$ on line 3, and D rejects $\langle M, w \rangle$ on line 5.

Therefore, D always halts and accepts $\langle M, w \rangle$ iff $\langle M, w \rangle \in A_{TM}$. So D decides A_{TM}, meaning $A_{TM} \leq_t L$. □

(c) Prove that L_{pug} (as defined in the bonus problem) is undecidable.

Solution.

Proof. We show a Turing reduction $Halt_{TM} \leq_t L$, in class we've shown that $Halt_{TM}$ is undecidable so this proves L is not decidable. Let N be a decider for L_{pug}. Then we can construct a decider for $Halt_{TM}$ as follows:

Algorithm 3 A decider D for A_{TM}

Input: $\langle M, w \rangle$ where M is a TM and w is a string

1: (Note: assume that we first checked if the input string is a valid encoding, and rejected if it was not one)
2: Build a TM M' as follows:
"On input x,
1. If x is the string 000111000111 or 000000000000, *reject* x.
2. If x is the string 010101010101 or 111000111000, loop forever.
3. Else:
4. Run M on input w. ▷ if M doesn't halt on w, M' is "stuck" here and it loops on x
5. If M halted on input w, accept x.
3: Run N on $\langle M' \rangle$. ▷ N always halts because it *decides* L_{pug}
4: If N accepted, accept $\langle M, w \rangle$.
5: If N rejected, reject $\langle M, w \rangle$.

We now show D is a decider for $Halt_{TM}$. We first observe the following:

- M' always rejects 000111000111 and 00000000000 (on line 1).
- M' always loops on 10101010101 and 111000111000 (on line 2).
- If M halts on w, then M' accepts 11111111111 and 101010101010 (on line 5). If M doesn't halt on w then M' loops on 11111111111 and 101010101010 (on line 4).

So if M halts on w, then M' is pugnacious so $\langle M' \rangle \in L_{pug}$. And if M doesn't halt on w, then M' isn't pugnacious (since it doesn't accept 11111111111) so $\langle M' \rangle \notin L_{pug}$.

(1) If $\langle M, w \rangle \in Halt_{TM}$, then M halts on w, so $\langle M' \rangle \in L_{pug}$. So N accepts $\langle M' \rangle$ on line 3, and D accepts $\langle M, w \rangle$ on line 4.

(2) If $\langle M, w \rangle \notin Halt_{TM}$, then M doesn't halt on w, so $\langle M' \rangle \notin L$. So N rejects $\langle M' \rangle$ on line 3, and D rejects $\langle M, w \rangle$ on line 5.

Therefore, D always halts and accepts $\langle M, w \rangle$ iff $\langle M, w \rangle \in Halt_{TM}$. So D decides A_{TM}, meaning $Halt_{TM} \leq_t L$. □