Handout 3 Solutions: TMs (Decidable and Recognizable)

Gianna Baker, Alice Lin, Charlie Mei

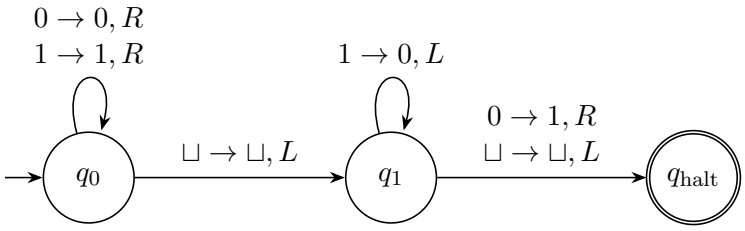
updated 5 November 2026

Solutions to Practice Problems

Problem 1: Understanding the TM

Problem 1: Understanding the TM

Consider the input-output TM $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{halt}})$ where $Q = \{q_0, q_1, q_{\text{halt}}\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$ drawn bellow.



(a) List the complete sequence of configurations on input 100.

(b) What are the outputs of M on inputs 10011 and 11?

(c) What function does M compute?

Solution:

(a) Listing the configurations:
On input 100, the machine begins in state q_0 and moves right, leaving each symbol unchanged. When it reaches the blank symbol at the end of the input, it moves left and enters state q_1.
In state q_1, it reads the final 0, changes it to a 1, moves right, and halts.
The complete sequence of configurations is:

$$\begin{aligned} q_0100 &\to 1q_000 \\ &\to 10q_00 \\ &\to 100q_0\sqcup \\ &\to 10q_10\sqcup \\ &\to 101q_{\text{halt}}\sqcup \end{aligned}$$

Recall that the state is written immediately before the symbol currently being read by the tape head. The final tape contains 101.

(b) Finding the outputs:
The machine effectively performs binary addition by 1. Starting from the rightmost bit, it changes each trailing 1 into a 0 until it encounters a 0, which it changes into a 1 before halting. Be careful with edge cases: if the machine reaches the leftmost tape cell while carrying, the head cannot move further left and may perform an additional transition on the symbol it just wrote.

- Input 10011: The machine changes the two trailing 1s into 0s, then changes the preceding 0 into a 1:

$$10011 \to 10010 \to 10000 \to 10100$$

  Output: 10100.

- Input 11: The machine changes both 1s into 0s, then encounters the left boundary and changes the first 0 back into a 1:

$$11 \to 10 \to 00 \to 10$$

  Output: 10.

(c) Identifying the function:
When the string isn't of the form $11\ldots1$ the TM computes binary increment (i.e. we turn x into $x+1$). However, when the string is made of 1 only, the TM will output the binary representation of $(x+1)/2$. In other words, it adds 1 to the input interpreted as a binary number and if the length of the output would have been increased, we divide by 2.
For example:

$$10011_2 + 1 = 10100_2$$

and

$$11_2 + 1 = 100_2.$$

Since the machine preserves the original two-bit length, the output in the second example is 10.
The empty string remains unchanged.

Problem 2: Finding the Language of a TM

Problem 2(a): Counting Zeros and Ones

Problem: Describe $L(M)$. What happens on the inputs $\epsilon, 001, 010, 1001, 10100, 1110$?
Solution:
The machine accepts if and only if:

- The input does not contain the substring 00, and
- The number of 0's is greater than the number of 1's.

Therefore, the language is:

$$\boxed{L(M) = \left\{ w \in \{0,1\}^* \;\middle|\; \begin{array}{l} w \text{ does not contain 00, and} \\ \#_0(w) > \#_1(w) \end{array} \right\}}$$

where $\#_0(w)$ and $\#_1(w)$ denote the number of 0's and 1's in w, respectively.

Evaluating the given inputs:

Input	#0's	#1's	Outcome	Reason
ϵ	0	0	Loop	Equal counts
001	2	1	Reject	Contains 00
010	2	1	Accept	No 00, more zeros
1001	2	2	Reject	Contains 00
10100	3	2	Reject	Contains 00
1110	1	3	Reject	More ones

Problem 2(b): Length Divisibility

Problem: Describe $L(N)$. What happens on the inputs $\epsilon, 0, 01, 101, 0101, 10101, 101010$?
Solution:
The machine accepts if the input is empty or its length is divisible by 3.
For nonempty inputs:

- If $|w|$ is divisible by 3, the machine accepts.
- Otherwise, if $|w|$ is even, the machine loops forever.
- Otherwise, the machine rejects.

Since the empty string has length 0, which is divisible by 3, we can express the language as:

$$\boxed{L(N) = \{w \in \{0,1\}^* \mid |w| \bmod 3 = 0\}}$$

Evaluating the given inputs:

Input	Length	Outcome	Reason
ϵ	0	Accept	Empty string
0	1	Reject	Odd, not divisible by 3
01	2	Loop	Even, not divisible by 3
101	3	Accept	Divisible by 3
0101	4	Loop	Even, not divisible by 3
10101	5	Reject	Odd, not divisible by 3
101010	6	Accept	Divisible by 3

Important distinction: The order of the algorithm matters. The machine checks whether the length is divisible by 3 before checking whether it is even.
Therefore, an input of length 6 is accepted, not looped on, even though 6 is also even.

Problem 3: Proving Recognizability

Problem 3(a): $A\mathcal{3}_{TM}$

Problem: Prove that

$$A\mathcal{3}_{TM} = \left\{ \langle M, w_1, w_2, w_3 \rangle \;\middle|\; \begin{array}{l} M \text{ is a TM, } w_1, w_2, w_3 \text{ are strings, and} \\ M \text{ accepts } w_1, w_2, \text{ and } w_3 \end{array} \right\}$$

is TM-recognizable.

Proof: Here's a TM recognizing $A\mathcal{3}_{TM}$:
R: "On input x:
if $x \neq \langle M, w_1, w_2, w_3 \rangle$, reject. (1)
else:
Run M on w_1; if M rejects, reject. (2)
Run M on w_2; if M rejects, reject. (3)
Run M on w_3; if M rejects, reject; if M accepts, accept." (4)

Analysis:

- $x \in A\mathcal{3}_{TM} \Rightarrow x = \langle M, w_1, w_2, w_3 \rangle$ where M is a TM and M accepts w_1, w_2, and w_3
  $\Rightarrow$ lines (2) and (3) finish without rejecting, and M accepts w_3 in line (4) $\Rightarrow$ R accepts x in line (4).
- $x \notin A\mathcal{3}_{TM} \Rightarrow$ either
  (a) $x \neq \langle M, w_1, w_2, w_3 \rangle \Rightarrow$ R rejects x in line (1), or
  (b) M does not accept some w_j. If M rejects $w_j \Rightarrow$ R rejects x; if M runs forever on $w_j \Rightarrow$ R runs forever.

So R recognizes $A\mathcal{3}_{TM}$, and $A\mathcal{3}_{TM}$ is TM-recognizable. ■

Problem 3(b): $MOD3_{TM}$

Problem: Prove that

$$MOD3_{TM} = \left\{ \langle M \rangle \;\middle|\; \begin{array}{l} M \text{ is a TM, } \exists\, w \in \Sigma^* \text{ s.t.} \\ |w| \bmod 3 = 0, M \text{ accepts } w \end{array} \right\}$$

is TM-recognizable.

Proof: Here's a TM recognizing $MOD3_{TM}$:
R: "On input x:

if $x \neq \langle M \rangle$, reject. (1)
else:
for $i = 3, 6, 9, \ldots$ (2)
for all $s \in \Sigma^*$ s.t. $|s| \in \{0, 3, 6, \ldots, i\}$: (3)
Run M on s for i steps. (4)
if M accepts, accept." (5)

Analysis:

- $x \in MOD3_{TM} \Rightarrow x = \langle M \rangle$ where M is a TM, and there is a string w such that
  (a) $|w| \bmod 3 = 0 \Rightarrow w$ is one of the strings in line (3) once $i \geq |w|$, and
  (b) M accepts w in k (finite) steps $\Rightarrow$ in the first iteration with $i \geq \max(|w|, k)$, M accepts w in line (4) $\Rightarrow$ R accepts x in line (5).
- $x \notin MOD3_{TM} \Rightarrow$ either
  (a) $x \neq \langle M \rangle \Rightarrow R$ rejects x in line (1), or
  (b) M does not accept any string whose length is a multiple of 3. Line (5) never accepts $\Rightarrow$ R runs forever.

So R recognizes $MOD3_{TM}$, and $MOD3_{TM}$ is TM-recognizable. ■

Problem 3(c): $AA_{TM,TM}$

Problem: Prove that

$$AA_{TM,TM} = \left\{ \langle M_1, M_2 \rangle \;\middle|\; \begin{array}{l} M_1, M_2 \text{ are TMs, } \exists\, w \in \Sigma^* \text{ s.t.} \\ M_1 \text{ accepts } w, M_2 \text{ accepts } w \end{array} \right\}$$

is TM-recognizable.

Proof: Here's a TM recognizing $AA_{TM,TM}$:

R: "On input x:

if $x \neq \langle M_1, M_2 \rangle$, reject. (1)

else:

for $i = 1, 2, 3, \ldots$ (2)

for all $s \in \Sigma^*$ s.t. $|s| \leq i$: (3)

Run M_1 on s for i steps. (4)

Run M_2 on s for i steps. (5)

if both accepted, accept." (6)

Analysis:

- $x \in AA_{TM,TM} \Rightarrow x = \langle M_1, M_2 \rangle$ where M_1, M_2 are TMs, and there is a string w such that
  (a) M_1 accepts w in k_1 (finite) steps, and
  (b) M_2 accepts w in k_2 (finite) steps $\Rightarrow$ in iteration $i = \max(|w|, k_1, k_2)$, M_1 accepts w in line (4) and M_2 accepts w in line (5) $\Rightarrow$ R accepts x in line (6).
- $x \notin AA_{TM,TM} \Rightarrow$ either
  (a) $x \neq \langle M_1, M_2 \rangle \Rightarrow$ R rejects x in line (1), or
  (b) there is no string accepted by both M_1 and M_2, so for every s, M_1 or M_2 does not accept s. Line (6) never accepts $\Rightarrow$ R runs forever.

So R recognizes $AA_{TM,TM}$, and $AA_{TM,TM}$ is TM-recognizable. ■

Problem 4(a): L_2

Problem: Prove that

$$L_2 = \{\langle D \rangle \mid D \text{ is a DFA and } L(D) = \Sigma^*\}$$

is TM-decidable.

Proof: Here's a TM deciding L_2:
R: "On input x:

if $x \neq \langle D \rangle$ where D is a DFA, reject. (1)
else:
  Add the start state q_0 to a queue, and mark q_0 as visited. (2)
  while the queue is not empty: (3)
    pop the first item q from the queue. (4)
    if q is not an accepting state, reject. (5)
    for each $a \in \Sigma$: (6)
      · if $\delta(q, a)$ is not visited, mark it as visited and add it to the queue. (7)
  accept." (8)

Analysis: The states that are ever added to the queue are exactly the states reachable from q_0.

- $x \in L_2 \Rightarrow x = \langle D \rangle$ where D is a DFA and $L(D) = \Sigma^*$
  $\Rightarrow$ every state q that can be reached from the start state is accepting $\Rightarrow$ for all string $w \in \Sigma^*$ ends in some reachable state q
  $\Rightarrow$ line (5) never rejects, and the queue eventually becomes empty $\Rightarrow$ R accepts x in line (8).
- $x \notin L_2 \Rightarrow$ either
  (a) x is not the encoding of a DFA $\Rightarrow$ R rejects x in line (1), or
  (b) $L(D) \neq \Sigma^*$, so some $w \notin L(D) \Rightarrow D$ on w ends in a non-accepting state q, which is reachable from the start state, so q is added to the queue and later removed in line (4) $\Rightarrow$ R rejects x in line (5).
- R **halts on every input:** each state is marked and added to the queue at most once, so the loop in line (3) runs at most $|Q|$ times, and each iteration looks at $|\Sigma|$ transitions in line (6).

So R decides L_2, and L_2 is TM-decidable. ■

Problem 4(a) another solution: L_2

Problem: Prove that

$$L_2 = \{\langle D \rangle \mid D \text{ is a DFA and } L(D) = \Sigma^*\}$$

is TM-decidable.

Proof: We've seen in class that

$$\text{EQ}_{DFA} = \{\langle D, D' \rangle \mid \ D, D' \text{ are DFAs and } L(D) = L(D')\}$$

is decidable. So let M be a decider for EQ_{DFA}. We now build another TM deciding L_2:
R: "On input x:

if $x \neq \langle D \rangle$ where D is a DFA, reject. (1)
else:
Let $\langle D' \rangle$ be the description of a DFA with $L(D') = \Sigma^*$. (D' exists since Σ^* is regular) (3)
Run M on $\langle D, D' \rangle$. (M is a decider so this always halts) (4)
If M accepted: accept. (5)
If M rejected: reject. (6)

Analysis:

- $x \in L_2 \Rightarrow x = \langle D \rangle$ where D is a DFA and $L(D) = \Sigma^*$
  $\Rightarrow$ when we run M on $\langle D, D' \rangle$ it will accept on line (3) $\Rightarrow$ R accepts x in line (5).
- $x \notin L_2 \Rightarrow$ either
  (a) x is not the encoding of a DFA $\Rightarrow$ R rejects x in line (1), or
  (b) $L(D) \neq \Sigma^*$, so M rejects on line (4) $\Rightarrow$ R rejects x in line (6).

So R decides L_2, and L_2 is TM-decidable. ■

Problem 4(b): L

Problem: Prove that

$$L = \left\{ \langle M, w, k \rangle \;\middle|\; \begin{array}{l} M \text{ is a Turing machine, } w \in \Sigma^*, \text{ and } k \geq |w|, \\ M \text{ visits at most } k \text{ distinct tape cells} \\ \text{when run on input } w \end{array} \right\}$$

is TM-decidable.

Idea: A standard TM has its head point to the leftmost tape cell and move at most one tape cell at a time. Therefore, if the head stays in the first k cells then we know Turing Machine did not visit more than k distinct tape cells. While the head is within the first k cells, the configuration at any point is determined by the state q ($|Q|$ choices), the contents of the first k cells ($|\Gamma|^k$ choices), and the position of the head (k choices). So there are at most $C = |Q| \cdot |\Gamma|^k \cdot k$ such configurations.

Proof: Here's a TM deciding L:
R: "On input x:

if $x \neq \langle M, w, k \rangle$, or $k < |w|$, or $k = 0$, reject. (1)
else:
  Compute $C = |Q| \cdot |\Gamma|^k \cdot k$. (2)
  for $t = 1, 2, \ldots, C$: (3)
    Run M on w for one more step. (4)
    if M's head reaches cell $k+1$, reject. (5)
    else if M halts, accept. (6)
  accept." (7)

Analysis:

- $x \in L \Rightarrow x = \langle M, w, k \rangle$ and M never reaches cell $k+1$ when run on w, so line (5) never rejects. Then either
  (a) M halts within the first C steps $\Rightarrow$ R accepts x in line (6), or
  (b) M runs for all C steps without halting. All of these configurations stay inside the first k cells, and there are at most C such configurations, so by the pigeonhole principle some configuration repeats twice during the for loop (there are $C+1$ configurations encountered, including the initial configuration, but at most C possible configurations). Since M is deterministic, from then on it repeats the same computation forever, always inside the first k cells $\Rightarrow$ the loop in line (3) finishes and R accepts x in line (7).

  the head never reaches cell $k+1$, so line (5) never rejects $\Rightarrow$ either M halts within C steps and R accepts x in line (6), or the loop finishes and R accepts x in line (7).
- $x \notin L \Rightarrow$ either
  (a) $x \neq \langle M, w, k \rangle$, $k < |w| \Rightarrow$ R rejects x in line (1), or
  (b) M's head reaches cell $k+1 \Rightarrow$ M visits more than k distinct cells
  $\Rightarrow$ R rejects x in line (5).
- R **halts on every input:** C is a finite number computed from $\langle M \rangle$ and k, and each iteration of line (3) simulates one step of M, so R stops after at most C iterations.

So R decides L, and L is TM-decidable. ■

Problem 4(c): *CYCLE*

Problem: Prove that

$$CYCLE = \{\langle G, k\rangle \mid G \text{ is a directed graph that contains at least one cycle of size at least } k\}$$

is TM-decidable. (Here $G = (V, E)$ is a directed graph, and a cycle of size i is a sequence of i distinct vertices $v_1, \ldots, v_i$ with edges $(v_1, v_2), \ldots, (v_{i-1}, v_i), (v_i, v_1)$.)

Proof: Here's a TM deciding *CYCLE*:
R: "On input x:

```
if x ≠ ⟨G, k⟩ where G is a directed graph and k is a positive integer, reject.   (1)
else:
    for i = k, k + 1, ..., |V|:                                                  (2)
        for every sequence (v_1, ..., v_i) of i distinct vertices of G:          (3)
            if (v_1, v_2), ..., (v_{i-1}, v_i), (v_i, v_1) are all edges of G, accept.   (4)
    reject."                                                                     (5)
```

Analysis:

- $x \in CYCLE \Rightarrow x = \langle G, k\rangle$ and G has a cycle $v_1, \ldots, v_i$ of size $i \geq k$
  $\Rightarrow i \leq |V|$ (the vertices are distinct), so iteration i of line (2) happens; the sequence $(v_1, \ldots, v_i)$ is tried in line (3), and all its edges are in $G \Rightarrow$ R accepts x in line (4).
- $x \notin CYCLE \Rightarrow$ either
  (a) $x \neq \langle G, k\rangle \Rightarrow$ R rejects x in line (1), or
  (b) G has no cycle of size at least k, so for every $i \geq k$, every sequence fails the check in line (4) (if $k > |V|$, line (2) does not run at all) $\Rightarrow$ R rejects x in line (5).
- R **halts on every input:** line (2) runs for at most $|V|$ values of i, for each i there are at most $|V|^i$ sequences in line (3), and each check in line (4) looks up i edges, so R reaches line (4) or line (5) after finitely many steps.

So R decides *CYCLE*, and *CYCLE* is TM-decidable. ■