CS 374 HW 9 Problem 2

quddus2, Aldo Sanjoto, Hieu Huynh

TOTAL POINTS

91.75 / 100

QUESTION 1

1 2A **16.75 / 25**

- **0 pts** Correct
- **25 pts** Undecideable
- **25 pts** No justification of answer / Incorrect algorithm
- **25 pts** No proof of correctness on provided algorithm
- **18.75 pts** IDK
- ✓ **6.25 pts** **No / Incorrect use of powerset construction (subset construction)**
- **6.25 pts** No / Incorrect use of XOR of languages of DFAs
- **6.25 pts** No mention of path to accepting state
- **2 pts** "Powerset construction (subset construction)" not mentioned
- ✓ **2 pts** **"Product construction" not mentioned**
- **6.25 pts** DFA minimization incorrect
- **6.25 pts** DFA equality incorrect / unclear

You need to create the DFA D directly based on M and N. Providing the language relation isn't enough (we were looking for product construction).

QUESTION 2

2 2B **25 / 25**

- ✓ **0 pts** **Correct**
- **18.75 pts** IDK
- **25 pts** Claimed language was decidable
- **25 pts** Incorrect algorithm

QUESTION 3

3 2C **25 / 25**

- ✓ **0 pts** **Correct**
- **25 pts** Claimed Undecidable
- **25 pts** Incorrect Algorithm
- **18.75 pts** IDK
- **10 pts** iff condition between being finite and having loops not mentioned

QUESTION 4

4 2D **25 / 25**

- ✓ **0 pts** **Correct**
- **18.75 pts** IDK
- **25 pts** Claimed Decidable
- **25 pts** Incorrect Algorithm

Q2)

a) L_1 is decidable

Decider (M, N) {

- Construct DFA D accepts the language:

$$L(D) = (L(M) \cap \overline{L(N)}) \cup (\overline{L(M)} \cap L(N));$$

- // We need to check if L(D) is empty or not
- Construct directed graph G from DFA D with each vertex is a state in D and each edge is a transition from 1 state to another state. We skip transition $\delta(q, *) = q$ to avoid self-loop. Also, we donot include more than 1 edge from 1 vertex to another vertex to avoid multigraph.
- Run DFS on G to find all nodes that can be reach from "start" vertex, which is the vertex correspond to the start state.
- Check all vertices that can be reach from "start" vertex

If there is any vertex corresponding to "accept state"
return False;

If there is no vertex corresponding to "accept" state
return True;

}

1 2A **16.75 / 25**

- **0 pts** Correct
- **25 pts** Undecideable
- **25 pts** No justification of answer / Incorrect algorithm
- **25 pts** No proof of correctness on provided algorithm
- **18.75 pts** IDK
- ✓ **6.25 pts** **No / Incorrect use of powerset construction (subset construction)**
- **6.25 pts** No / Incorrect use of XOR of languages of DFAs
- **6.25 pts** No mention of path to accepting state
- **2 pts** "Powerset construction (subset construction)" not mentioned
- ✓ **2 pts** **"Product construction" not mentioned**
- **6.25 pts** DFA minimization incorrect
- **6.25 pts** DFA equality incorrect / unclear

You need to create the DFA D directly based on M and N. Providing the language relation isn't enough (we were looking for product construction).

b)

L_2 is Undeciable

- Suppose there is an algorithm Decide Equal that correctly decides the language L_2. Then we can build the decider for E_{TM} as follows:

> Decide Empty (<M>):
>
> - Include the constant code for a DFA D that reject all its input. We denote string encoding D as <D>
> - Run Decide Equal on <M, D>
>
> If Decide Equal accepts, then accepts
> If Decide Equal rejects, then reject

Prove correctness:

1) Suppose M is empty:
   ⇒ Run Decide Equal will accept because $L(M) = L(D) = \emptyset$
   ⇒ Decide Empty correctly accept M.

2) Suppose M is not empty
   ⇒ Run Decide Equal will reject because $L(M) \neq L(D)$
   ⇒ Decide Empty correctly rejects M.

We know that E_{TM} is undeciable (from lecture slide)

⇒ L_2 is undecidable!

2 2B 25 / 25

✓ - 0 pts Correct

- 18.75 pts IDK

- 25 pts Claimed language was decidable

- 25 pts Incorrect algorithm

c) L_3 is Decidable.

Decider (M) {

- Construct a directed graph G from the DFA M. Each vertex in G is a state in D. Each edge in G is a transition from 1 state to a different state.
  - Note: . We only add edge from 1 vertex to another vertex once to avoid Multigraph
    . We do not add edge corresponding to self-loop transition to avoid self-loop in grap G.
  - If a state in M has a self-loop transition, we mark the vertex correspond to that state te be a BAD state
- We run DFS on G starting at the "start" vertex to find all Cycles that can be reached from the "start" vertex. We mark states in those Cycles to be BAD states
- Run DFS to find all vertices that can be reach from "start" vertex
- For each v ∈ reach ("start" vertex) {

  If (v is a Bad State) {

  Run DFS on G to find all vertices that can be reach from v.

  If any vertex in reach(v) is a vertex that corresponding to ACCEPT state in D

  return FALSE;

  } }

Return True;

}

To find all cycles can be reached from u & mark states in those cycles to be BAD states, we modify DFS:

```
DFS(u) {
    Mark all vertices as unvisited
    Mark all vertices as NOT BAD STATE
    Mark u as visited; prev(u) ← NULL;
    For v ∈ Out(u) do {
        If (v is not visited) {
            prev(v) = u;
        }
        Else {
            temp ← u
            While (temp ≠ v) {
                Mark temp as BAD state;
                temp ← prev(temp);
            }
        }
    }
}
```

3 2C 25 / 25

✓ - 0 pts Correct

- 25 pts Claimed Undecidable
- 25 pts Incorrect Algorithm
- 18.75 pts IDK
- 10 pts iff condition between being finite and having loops not mentioned

2D) L_4 is Undeciable

Suppose there is an algorithm DecideFinite that correctly decides the language L_4. Then we can solve halting problem as follows:

```
Decide Halt(<M, w>) {
    Encode the following Turing Machine M'

        M'(x):
            run M on input w
            return True

    If(DecideFinite(<M'>) == True)
        return False;
    Else
        return True;
}
```

- Suppose M halts on input w
  - $\Rightarrow$ M' accepts every input string x
  - $\Rightarrow$ Decide Finite reject the encoding $\langle M' \rangle$
  - $\Rightarrow$ Decide Halt correctly accept the encoding $\langle M, w \rangle$
- Suppose M not halt on input w
  - $\Rightarrow$ M' diverges on every input string x
  - $\Rightarrow$ M' accept 0 strings
  - $\Rightarrow$ Decide Finite accept the encoding $\langle M' \rangle$
  - $\Rightarrow$ DecideHalt correctly reject the encoding $\langle M, w \rangle$.

Halt is undeciable. Therefore, L_4 is undeciable.

4 2D 25 / 25

✓ - 0 pts Correct

- 18.75 pts IDK

- 25 pts Claimed Decidable

- 25 pts Incorrect Algorithm