ST0270 Formal Languages and Compilers
Assignment 1

Sergio Ramírez Rico

February 10, 2025

1 Deadline

See course website.

2 Preliminaries

Let $M = (Q, \Sigma, \delta, s, F)$ be a deterministic finite automata (DFA).

1. **Inaccessible states.** A state $q \in Q$ is said to be **inaccessible** if there is no string $x \in \Sigma^*$ such that $\hat{\delta}(s, x) = q$.
2. **Equivalent states.** A pair of states $p, q \in Q$ is said to be **equivalent** if and only if $(\forall x \in \Sigma^*)(\hat{\delta}(p, x) \in F \equiv \hat{\delta}(q, x) \in F)$. That is, any string x that from p allows us to access a final state, must also allows us to reach a final state from q, and vice versa.
3. We say that two states can be **collapsed** if they are equivalent.

3 Assignment

The assignment is to implement, in any programming language, the minimization algorithm presented in Kozen 1997, Lecture 14.

Given a DFA with no inaccessible states, the algorithm returns the states that are equivalent. Therefore, such states can be collapsed and we shall obtain a minimized automaton.

The minimization algorithm in Lecture 14 is based on the construction given in Lecture 13. It is not necessary to read Lecture 13 to implement the algorithm, but if you choose to read it, you can gain a high level understanding of the minimization procedure.

4 Input/output

Your program should fulfill the following specifications.

Input

A *case* is a DFA M with no inaccessible states.

You may assume states are denoted by natural numbers and the initial state (s) is always the number 0 (zero). Alphabets are formed with letters of the latin alphabet (with 26 letters). In ASCII code, characters from 97 to 122.

The input of the program is as follows.

1. A line with a number $c > 0$ indicating how many cases you will receive.
2. For each case, in a single line a number $n > 0$ denoting of states of M.
3. Then, a single line with the alphabet of M. Symbols are separated by blank spaces.
4. Then, the final states of M separated by blank spaces.
5. Finally, n lines, one for each state. Each line is a row of the table that represents M.

   Assume that the symbols of the alphabet appear in the table in the same order as they were given in step 3. For instance, if the automaton is

		a	b
$\rightarrow$	0	1	2
$\leftarrow$	1	3	4
	2	4	3
	3	5	5
$\leftarrow$	4	5	5
$\leftarrow$	5	5	5

   those n lines are

```
0 1 2
1 3 4
2 4 3
3 5 5
4 5 5
5 5 5
```

Output

For each case, print the pairs of states that are equivalent in lexicographical order. All the pairs in a single line separated by blank spaces.

5 Assignment Submission

You should follow the instructions below for this assignment.

1. Submissions only accepted via the GitHub Classroom link available on course website.
2. Follow the input/output instructions. Do not print extra lines.

3. A `README.md` file (Markdown format) in English is required. It must contain the following information:

   - Student's full name.
   - Student's class number.
   - Versions of the operating system, programming language, and tools used in your implementation.
   - Detailed instructions for running your implementation.
   - Explanation of the algorithm.

4. Do not include unnecessary files or directories in the repository.

References

Kozen, Dexter C. (1997). *Automata and Computability*. 1st. Berlin, Heidelberg: Springer-Verlag. ISBN: 0387949070. DOI: https://doi.org/10.1007/978-1-4612-1844-9.

Example I/O

Input

```
4
6
a b
1 2 5
0 1 2
1 3 4
2 4 3
3 5 5
4 5 5
5 5 5
6
a b
3 4 5
0 1 2
1 3 4
2 4 3
3 5 5
4 5 5
5 5 5
6
a
1 4
0 1
1 2
2 3
3 4
4 5
5 0
4
a b
0 1
0 1 2
1 1 2
2 3 1
3 3 3
```

Output

```
(1, 2) (3, 4)
(1, 2) (3, 4) (3, 5) (4, 5)
(0, 3) (1, 4) (2, 5)
(0, 1)
```