**DFA Minimization**

**Theory of Computing Phase 1**

**Supervised by**

**Dr. Zeinab Abdelhalim**

**Eng. Ahmed Gamal**

**Bahaa Shamoon Atia Ehab Yahia Zakaria Basma Mahfouz**

**202001572 202001294 202000920**

**Adham Waleed Fouad**

**202000857**

***Table of Contents:***

**1. What is DFA Minimization**

**2. The Methods used for DFA Minimization**

**3. Minimization of DFA Using Equivalence Theorem**

**4. code**

**4.1. Inputs Format**

**4.2. Outputs Format**

**1. What is DFA minimization**

DFA (Deterministic Finite Automaton) minimization is the process of reducing the number of states in a deterministic finite automaton while preserving its essential behavior. The goal is to create an equivalent DFA with the minimum possible number of states that still recognize the same language as the original DFA. Minimizing a DFA simplifies its structure, making it more manageable and efficient for various applications.

The primary motivation for DFA minimization is to optimize automata for better performance in terms of memory usage, computational speed, and ease of analysis. The process involves identifying and merging equivalent states, where two states are considered equivalent if they lead to the same outcomes for any given input string. By merging these equivalent states, redundant information is eliminated, resulting in a more compact representation of the language accepted by the automaton.

**2. Methods used for DFA minimization**

There are two popular methods for minimizing a DFA.

A diagram of a method

Description automatically generated

We will use the Equivalence Theorem to Minimize DFA

**3. Minimization of DFA Using Equivalence Theorem**

There are 6 Essential Steps for this method.

Step-01:

* Eliminate all the dead states and inaccessible states from the given DFA (if any)

Step-02:

* Draw a state transition table for the given DFA.

Step-03:

Now, start applying the equivalence theorem.

* Take a counter variable k and initialize it with value 0.
* Divide Q (set of states) into two sets such that one set contains all the non-final states and the other set contains all the final states.
* This partition is called P0.

Step-04:

* Increment k by 1.
* Find Pk by partitioning the different sets of Pk-1 .
* In each set of Pk-1 , consider all the possible pair of states within each set and if the two states are distinguishable, partition the set into different sets in P

Step-05:

* Repeat step-04 until no change in partition occurs.
* In other words, when you find Pk = Pk-1, stop.

Step-06:

* All those states which belong to the same set are equivalent.
* The equivalent states are merged to form a single state in the minimal DFA.

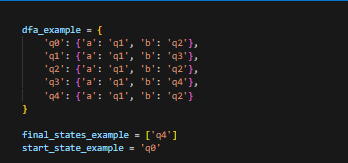
**Number of states in Minimal DFA** **= Number of sets in Pk**

**4. Code**

The code was written with Python Jupyter

**4.1 input Format**

We will insert the state transition table, The name of the starting state and the ending states

****

**4.1 Output Format**

There are 2 output formats. Graphical – text)

**Graphical:**

We draw the DFA before and after Minimization**.** We used “graphviz” library to draw the DFA.

**Before After**

A diagram of a circle with arrows

Description automatically generated A diagram of a red circle with black arrows and black lines

Description automatically generated

**Text:**

We printed the state transition tables.

**A screenshot of a computer

Description automatically generated**

**→** Starting State

\* Final State

**Snip Shot of the code:**

A screen shot of a computer program

Description automatically generated

**References**

1. [*https://www.youtube.com/watch?v=0XaGAkY09Wc*](https://www.youtube.com/watch?v=0XaGAkY09Wc)
2. [*https://www.gatevidyalay.com/minimization-of-dfa-minimize-dfa-example/*](https://www.gatevidyalay.com/minimization-of-dfa-minimize-dfa-example/)