**Homework 1 Questions**

# **Problem 1** (5 points) answer the following questions:

## If has elements and has elements, how many elements are in ?

* + There are elements in

*Assume elements in*

*Assume elements in*

*The length of is , therefore*

*The length of is , therefore*

*Listing the elements shows the length of is*

*Calculating also shows a length of*

## If is a set with elements, how many elements are in the power set of ?

* + The power set is of

*Assume elements in*

*The length of is 2, therefore*

*Listing the elements shows the length of is*

*Calculating also shows a length of*

## Let is a relation on the set . What is (Reflexive and transitive closure)?

*Elements in*

*Find Reflexive Closure of :*

*Reflexive Closure Formula*

*Subtitute Values*

*Combine Elements (Union)*

*Find Transitive Closure of :*

*Transitive Property*

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

*Find Missing Pairs*

*Include Transitive Pairs*

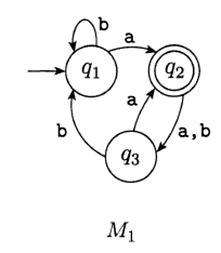
*Find Reflexive and Transitive Closure of :*

*Reflexive and Transitive Closure Formula*

*Subtitute Values*

*Combine Elements*

# **Problem 2** (5 points) Answer the following questions about the state diagrams of a DFA



## What is the start state?

## What is the set of accept states?

## What sequence of states does the machine go through on input ?

## Does the machine accept the string ?

* + No, because string finishes in , which is not an accept state.

## Does the machine accept the string ?

* + No, because string finishes in , which is not an accept state.

# **Problem 3** (10 points) Each of the following languages is the intersection of two simpler languages. In each part, construct DFAs for the simpler languages, then combine them using the construction discussed in footnote 3 (page 46) to give the state diagram of a DFA for the language given. In all parts, .

Diagram

Description automatically generated

Figure 1: DFA with Test Input

Diagram

Description automatically generated

Figure 2: DFA with Test Input

### 

*The Transition Function*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *State* | *Input* | *A* | *B* | *Transition* |
|  |  |  |  |  |
|  |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |

Diagram

Description automatically generated

Figure 3: DFA with Test Input

Diagram

Description automatically generated with medium confidence

Figure 4: DFA with Test Input

Diagram

Description automatically generated

Figure 5: DFA with Test Input

*The Transition Function*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *State* | *Input* | *A* | *B* | *Transition* |
|  |  |  |  |  |
|  |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
|  |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |
| , |  |  |  | , |

Diagram

Description automatically generated

Figure 6: DFA with Test Input

# **Problem 4** (10 points) 1.6 Give state diagrams of DFAs recognizing the following languages. In all parts, the alphabet is {0,1}.

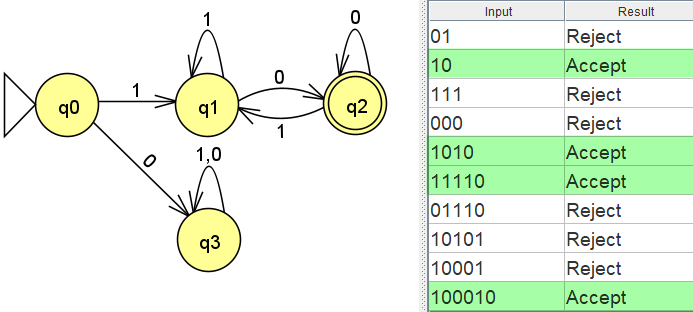


Figure 7: DFA with Test Input

Diagram

Description automatically generated

Figure 8: DFA with Test Input

# **Problem 5** (10 points) 1.7 Give state diagrams of NFAs with the specified number of states recognizing each of the following languages. In all parts, the alphabet is {0,1}.

## The language with three states

Diagram

Description automatically generated

Figure 9: NFA with Test Input

* *Since NFAs accept input if any sequence of the possible choices leads to a final state, we can loop all 1’s and 0’s until the final two 0’s appear (aka we do not need to account for every possible appearance of a 1 since NFAs automatically considers that possibility)*

# **Problem 6** (10 points) 1.16 Use the construction given in Theorem 1.39 to convert the following nondeterministic finite automata to equivalent deterministic finite automata **(show the process).**

Diagram

Description automatically generated

## Determine -*closure* for each state

-*closureself-state -reachable states* -*closureFormula*

*Name new state*

|  |  |  |  |
| --- | --- | --- | --- |
|  | self | -reachable |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Obtain transition for any New States Found

### Calculate Transition for

*State*

-*closure Transition Formula*

-*closure Substitute Values*

-*closure Expand, Showing Union*

-*closure Find Transition for each*

*Find -closure for each*

*Solve for*

*Name new state*

-*closure Substitute Values*

-*closure Expand, Showing Union*

-*closure Find Transition for each*

*Solve for*

### Calculate Transition for

*State*

-*closure Substitute Values*

-*closure Expand, Showing Union*

-*closure Find Transition for each*

*Find -closure for each*

*Solve for*

*State B*

-*closure Substitute Values*

-*closure Expand, Showing Union*

-*closure Find Transition for each*

-*closure Remove Empty Sets*

*Find -closure for each , Solve for*

*Name new state*

### Calculate Transition for

*State*

-*closure Substitute Values*

-*closure Expand, Showing Union*

-*closure Find Transition for each*

*Find -closure for each*

*Solve for*

*State A*

-*closure Substitute Values*

-*closure Expand, Showing Union*

-*closure Find Transition for each*

-*closure Remove Empty Sets*

*Find -closure for each , Solve for*

*State C*

## Determine Final State(s) for DFA

### Identify Final State in Original NFA

Diagram

Description automatically generated

Figure 10: The Final State in the original NFA is 2

### Identify any DFA states which contain the NFA Final State

|  |  |
| --- | --- |
| DFA States | |
|  |  |
|  |  |
|  |  |

Figure 11: States A, B, C will be Final States in the DFA

## Converted DFA:

Diagram

Description automatically generated

Figure 12: DFA with Test Input

*By comparing the results of which strings are accepted/rejected, we can see that the DFA above is equivalent to the provided NFA:*

Diagram

Description automatically generated

Figure 13: Provided NFA with Test Input

# **Problem 7** (20 points) 1.18 Give regular expressions generating the following languages. In all parts the alphabet is {0, 1}



# **Problem 8** (10 points) 1.19 Use the procedure described in Lemma 1.55 to convert the following regular expressions to nondeterministic finite automata.



Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated



Chart

Description automatically generated

Chart

Description automatically generated

Figure 14: -NFA of

*Additionally, I have converted this -NFA to NFA, as shown below*

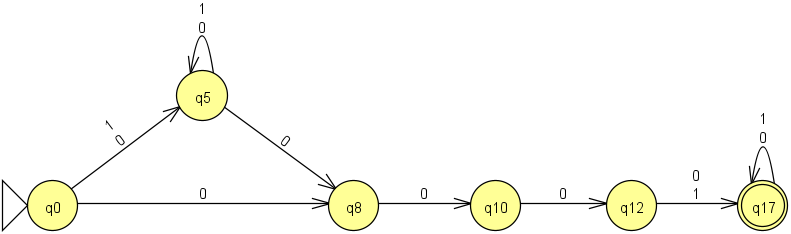


Figure 15: NFA of

# **Problem 9** (20 points) A syntactically valid email address is made up of a username followed by ‘@’ followed by a list of at least two domain names separated by ‘.’. Assume that user and domain names are made up of letters [a-z] and digits [0-9], and the main domain name, i.e., the last domain name, contains two or three characters.

|  |  |
| --- | --- |
| Valid | Invalid |
| abc@dsu.edu | a.b.ab |
| abc@pluto.dsu.edu | ab@ab |
| [11@123.com](mailto:11@123.com) | ab@ab.abcd |

## (5 points) Use JFLAP design a finite automaton to recognize valid mail address. Enclose your FA’s JFLAP file (.jff) in your homework submission.

Diagram

Description automatically generated

Figure 16: -NFA for Email Address Strings

Diagram

Description automatically generated

Figure 17: NFA for Email Address Strings

Diagram, schematic

Description automatically generated

Figure 18: DFA for Email Address Strings

*JFLAP* .jff *files are also attached in homework submission as* HW1-Q9eNFA[Final].jff, HW1-Q9DFA[Final].jff

## (5 points) Test the above six testing cases using JFLAP Multiple Run function. You can include more testing cases if you want. Include a screenshot of your testing results in your homework submission as below:

Chart

Description automatically generated

Figure 19: Test Input for Email ε-NFA

Chart

Description automatically generated

Figure 20: Test Input for Email NFA

Chart

Description automatically generated

Figure 21: Test Input for Email DFA

## (10 points) Use a programming language at your choice to implement the FA designed in Step 1. Submit your source code and also write a readme file to show how to compile your program, how to run your program, and some testing results you have. **(Note: The program should implement the FA based on your design and simulate the way how FA works. There is no credit for the program if the implementation is based on the use of regular expressions.)**

### Will be included in HW folder