COMP 2049 (AE2LAC) Languages and Computation

Coursework: Automata, Regular Languages, and Context-Free Languages

Spring 2020

Release date: Tuesday, May 5th, 2020

***Deadline*: Wednesday, May 27th, 2020, 16:00**
Cut-off Date: Thursday, May 28th, 2020, 16:00

Total Mark: 100
Weight: 25% of the module mark
How to submit: Via Moodle

1 Finite languages

Task 1.1. *Assume that $L \subseteq \Sigma_*$ is a language over the alphabet Σ. Prove that, if L is finite, then L is a regular language.*

- If L is a finite language, then it contains a finite number of strings $a_0, a_1, \ldots a_n$
- The language $\{a_i\}$ consisting of a single literal string a_i is regular.
- The union of a finite number of regular languages is also regular.
- Therefore, L=$\{a_0\} \cup \{a_1\} \cup \cdots \cup \{a_n\}$ is regular.

2 Infinite-state automaton (ISA)

Task 2.1. *Assume that $\Sigma = \{a, b\}$ and $L \subseteq \Sigma_*$ is an* ***arbitrary*** *language over Σ. Prove that there exists an ISA M such that $L(M) = L$.*

Note: The proof must be clear, complete, and no longer than half a page.

- An arbitrary language has its necessary and sufficient conditions as any subset of {a, b}*
- So for any language under subset of {a, b}* it contains finite number of words, it is finite language
- Previously we prove that for any given finite language it contains a finite number of stings and these single literal string is also regular . The union of such string is also regular. Thus we prove it is regular language.
- While we can see from the given information "if L is a regular language, then there exists an ISA that accepts L."
- Thus the assumption has been proven.

3 Context-free languages (CFLs)

For each $n \geq 1$, let $b(n)$ denote the binary representation of n, and let $\mathrm{revb}(n)$ denote the reverse of $b(n)$. Some examples of these strings are provided for various values of n in the following table:

n in decimal	5	6	12	13	43	44
$b(n)$	101	110	1100	1101	101011	101100
revb(n)	101	011	0011	1011	110101	001101

Consider the alphabet $\Sigma = \{0, 1, .\}$, and define the language $L_r \subseteq \Sigma*$ as follows:

$$L_r := \{\mathrm{revb}(n) \;.\; b(n+1) \mid n \geq 1\}$$

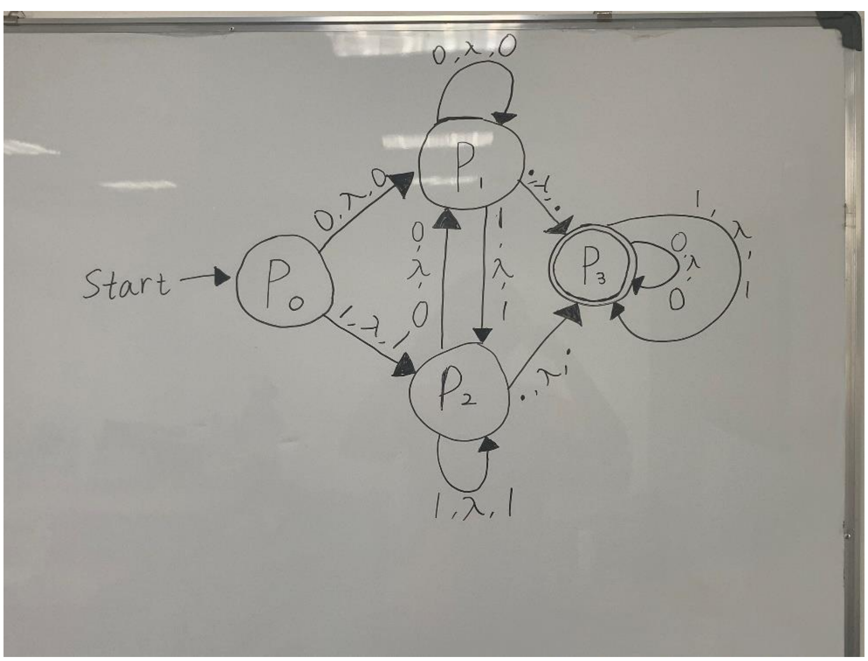
$$L_r = \{1 \;.\; 10, 01 \;.\; 11, 11 \;.\; 100, 001 \;.\; 101, \cdots\}$$

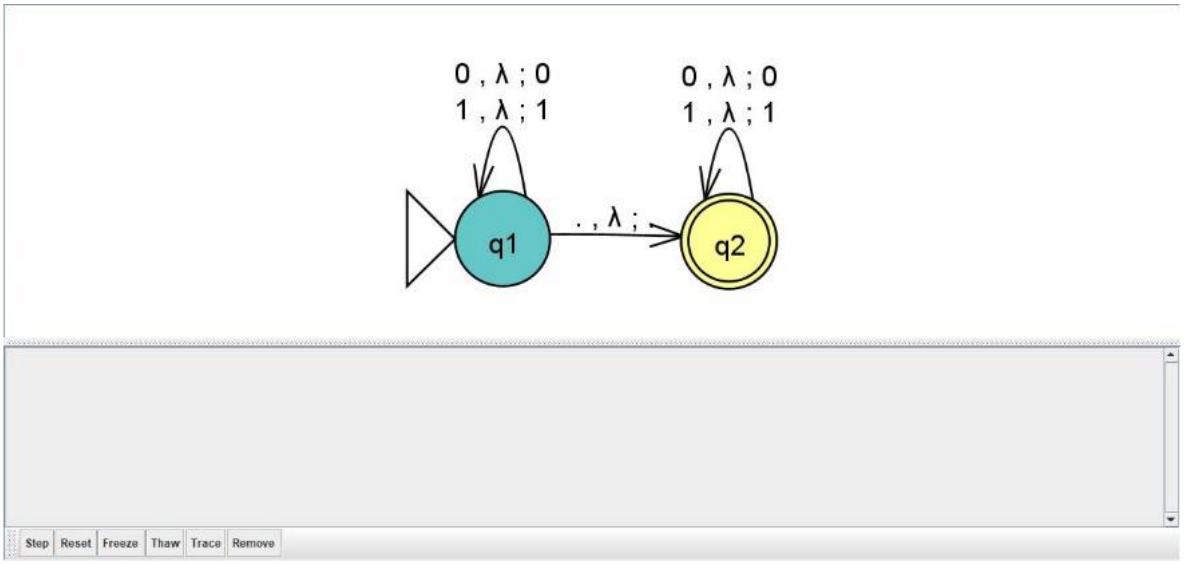
Task 3.1. *Show that L_r is a context-free language by constructing a pushdown automaton (PDA) that accepts L_r. In particular, you are required to:*

(a) Provide a clear description of the main idea of the design of the PDA and what the stack is used for.

So this PDA is used to determine if this input string is legal for acceptance, or it will be rejected. After reaching the end of string if the automaton is entering the final state then it is belonging to L_r , if it doesn't entered the final state it is rejected. The stack is used to keep track of what is read in and help with the transition. If I were to establish this automaton I would first put what is read in into the stack and make the corresponding transitions.

(b) Draw a transition diagram for the constructed PDA.





Task 3.2. *Design a context free grammar G for which $L_r = L(G)$. Specifically, you are required to:*

(a) Write down all the production rules clearly.

$S \rightarrow SS$

$S \rightarrow 1S$

$S \rightarrow 0S$

$S \rightarrow .S$

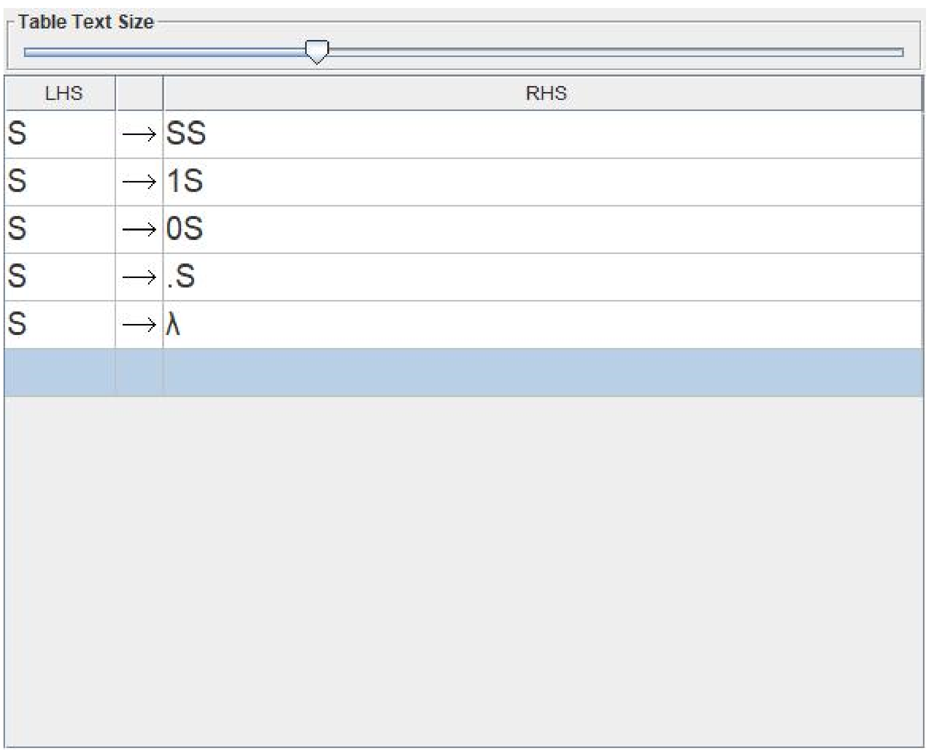
$S \rightarrow \lambda$

(b) Give a verbal account of the main points of your answer, such as, what each variable is used for. This should not take more than a paragraph.

$G = (S, \{0,1,.\}, R, S)$

A context-free grammar G is defined by the 4-tuple $G = (V, \sum, R, S)$
The first parameter V is a finite set which contains a lot of variables , in this case , we just take single variable S
The second parameter is sigma which is all the possible alphabet of the input string
The members of R are called the (rewrite) rules or productions of the grammar. (also commonly symbolized by a P)
The last parameter S is the start variable (or start symbol), used to represent the whole sentence (or program). It must be an element of V. Here it is just S.

(c) Implement the grammar in JFLAP.



LHS		RHS
S	→	SS
S	→	1S
S	→	0S
S	→	.S
S	→	λ

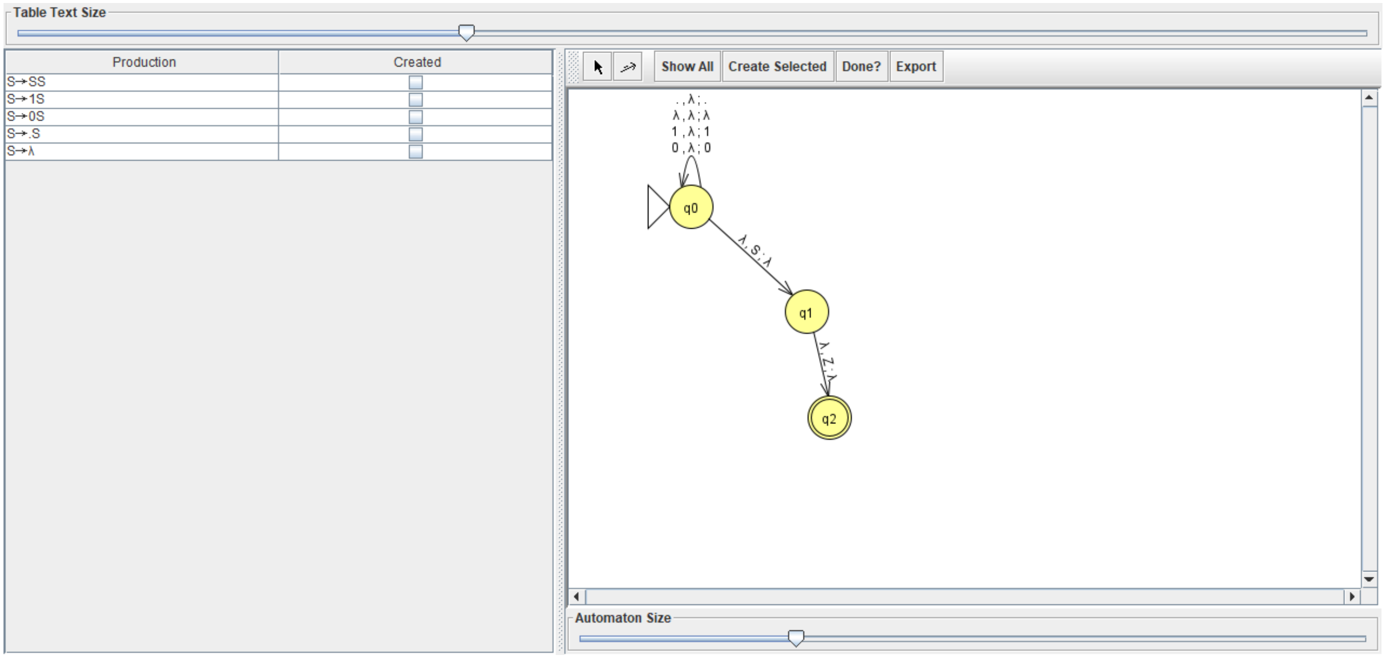


Table Text Size
Production
Created
S→SS
S→1S
S→0S
S→.S
S→λ
Show All
Create Selected
Done?
Export
. , λ ; .
λ , λ ; λ
1 , λ ; 1
0 , λ ; 0
q0
λ , S ; λ
q1
λ , Z ; λ
q2
Automaton Size

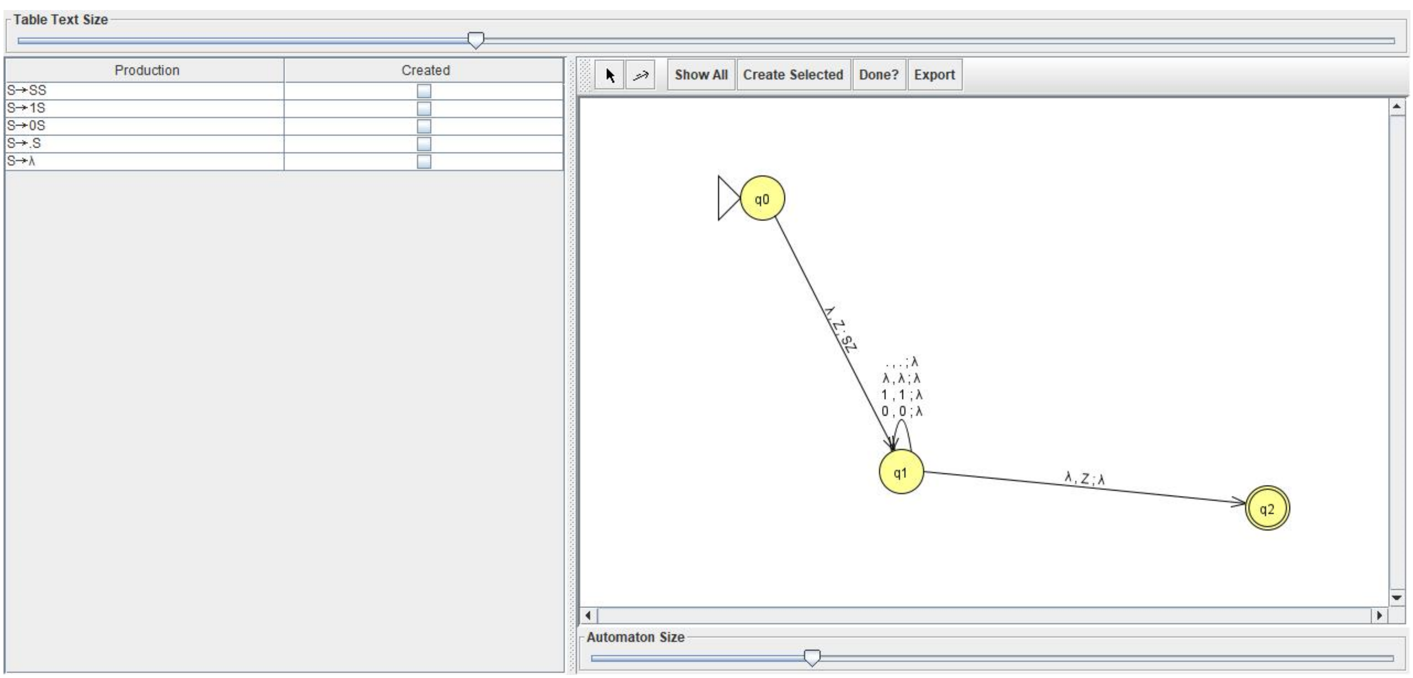


Table Text Size
Production
Created
S→SS
S→1S
S→0S
S→.S
S→λ
Show All
Create Selected
Done?
Export
q0
λ , Z ; SZ
. , . ; λ
λ , λ ; λ
1 , 1 ; λ
0 , 0 ; λ
q1
λ , Z ; λ
q2
Automaton Size