Theory of Computation – Construction of Regular Expressions from Deterministic Finite Automata

We know that for every regular expression there exist a deterministic finite automaton. So we can say that regular languages, regular expressions and finite automata are all different representation of the same thing. Earlier we learnt how to convert a regular expression into a finite automaton; in this tutorial we will learn how to convert a given finite automaton to a regular expression. For that we must first learn about Arden's Theorem.

Arden's Theorem

Let P and Q be two regular expressions over alphabet Σ. If P does not contain null string, then

R = Q + RP

has a unique solution that is R = QP*

Proof:

Put the value of R in the R.H.S.

$R = Q + (Q + RP)P = Q + QP + RP^2$

When we put the value of R again and again we get the following equation

$R = Q + QP + QP^2 + QP^3$.....

$R = Q(1 + P + P^2 + P^3 + ...$

$R = Q(\epsilon + P + P^2 + P^3 + ...$

Technology Courses

- SAN I - Technology Training
- SAN II - Administration Training
- Linux Administration
- Advanced C Programming
- Linux Debugging Techniques
- Linux System Programming
- Linux IPCs

The second part of the product on the L.H.S can be replaced with the kleen closure. So the equation becomes
R = QP*

Using Arden's Theorem to find Regular Expression of a Deterministic Finite automata

1. For getting the regular expression for the automata we first create equations of the given form for all the states
   $q_1 = q_1w_{11} + q_2w_{21} + \ldots + q_nw_{n1} + \epsilon$ (q_1 is the initial state)
   $q_2 = q_1w_{12} + q_2w_{22} + \ldots + q_nw_{n2}$
   . .
   . .
   . .
   $q_n = q_1w_{1n} + q_2w_{2n} + \ldots + q_nw_{nn}$
   w_{ij} is the regular expression representing the set of labels of edges from q_i to q_j
   Note: for parallel edges there will be that many expressions for that state in the expression.
2. Then we solve these equations to get the equation for q_i in terms of w_{ij} and that expression is the required solution, where q_i is a final state.

Assumptions made while forming the regular expression

1. The transition diagram should not have ϵ – transitions
2. It must have only a single initial state

Let us see an example to demonstrate this method

Example: Draw a FA that accepts strings containing exactly 1 over alphabet {0, 1} and write a regular expression for the same.
View Answer

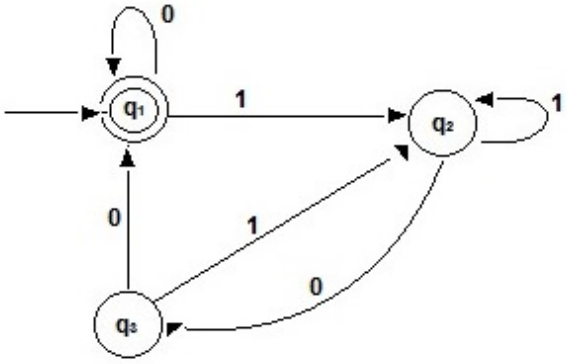
Note: We need not consider a trap state in our equations as it does not contribute to the regular expression

Given below are few questions for you to practice.

Question1: Construct a regular expression corresponding to the automata given below

Programming
- Linux Network Programming
- Linux Multithreading
- Linux Kernel Programming
- Linux Interr
- Linux Basic
- Linux Block Device Drivers
- Linux Network Device Drivers
- Linux PCI Device Drivers
- Linux USB Device Drivers
- Linux Video Device Drivers
- Linux Audio Device Drivers
- Linux I2C Device Drivers

READ MORE:
Artificial Intelligence Questic Agent Architecture and Onlir



View Answer

Question 2: Derive a regular expression for the language containing strings ending in 1 but not containing substring 00.

View Answer

Sanfoundry Global Education & Learning Series – 100+ Theory of Computation Tutorials.

If you wish to look at all Tutorials and their examples, go to Theory of Computation Tutorials.

Theory of Computation – Regular Expressions and Regular Languages

Theory of Computation – Transition Graph and Transition Table For a Finite Automata

Theory of Computation – Language of a DFA and Extended Transition Function

Best Reference Books – Integer Programming: Theory and Computations

C++ Program to Construct an Expression Tree for a Given Prefix Expression

Next / Around the World

C MCQs
C++ MCQs
C# MCQs
Java MCQs
Linux MCQs
SAN MCQs
Javascript
PHP MCQs
Python MCQs
OS MCQs
COA MCQs
DBMS MCQs
SE MCQs
AI MCQs
LISP MCQs

C Programs
C++ Algorithm
Java Algorithm
Android Progra

Theory of Computation - Designing and Programming a DFA

Theory of Computation - Transition Graph and Transition Table For a Finite Autom...

Computer Science and Engineering Internships in Theory of Computation

Theory of Computation - Regular Grammars

Manish Bhojasia, a technology veteran with 17+ years @ Cisco & Wipro, is Founder and CTO at Sanfoundry. He is Linux Kernel Developer and SAN Architect and is passionate about competency developments in these areas. He lives in Bangalore and delivers focused training sessions to IT professionals in Linux Kernel, Linux Debugging, Linux Device Drivers, Linux Networking, Linux Storage & Cluster Administration, Advanced C Programming, SAN Storage Technologies, SCSI Internals and Storage Protocols such as

iSCSI & Fiber Channel. Stay connected with him below.

Follow 115

Manish Bhojasia

Follow

502 followers

Sanfoundry

Follow +1

+ 842

Follow Sanfoundry @ Twitter

Connect with Manish @ LinkedIn

Join Sanfoundry Jobs @ LinkedIn

Join Linux Programming & Quizzing

Join C/C++ Programming & Quizzing

Join SAN Programming & Quizzing

Join Computer Science Programming

Join Computer Science Programming

Join C/C++ Programming & Quizzing

Join C/C++ Programming & Quizzing

Join Linux Programming & Quizzing

Sanfoundry is **No. 1** choice for **Deep Hands-ON** Trainings in **SAN, Linux & C, Kernel & Device Driver Programming**. Our Founder has trained employees of almost all Top Companies in India. Here are few of them: VMware, Citrix, Oracle, Motorola, Ericsson, Aricent, HP, Intuit, Microsoft, Cisco, SAP Labs, Siemens, Symantec, Redhat, Chelsio, Cavium Networks, ST Microelectronics, Samsung, LG-Soft, Wipro, TCS, HCL, IBM, Accenture, HSBC, Northwest Bank, Mphasis, Tata Elxsi, Tata Communications, Mindtree, Cognizant, mid size IT companies and many Startups. Students from top Universities and colleges such as NIT Trichy, BITS Pilani, University of California, Irvine, University of Texas, Austin & PESIT Bangalore have benefited a lot from these courses as well. The assignments and real time projects for our courses are of extremely high quality with excellent learning curve.

Register for **Expert Level Training Classes** by our Founder & CTO. Alternatively, call us for your **Corporate Training** or **College Training** needs.

↑