CSE_105_Midterm1_Review_Doc

- **Everything up to Monday (Chapter 1)**
- **MIDTERM 2: everything after midterm 1**
- **Terms**

Vocabulary review

From CSE20,etc. Sipser p. 14

- { a,b,c,d,e } The **set** whose elements are a,b,c,d,e
- | ababab | = 6 The **length** of the string ababab is 6
- | { a,b,c,d,e } | = 5 The **size** of the set {a,b,c,d,e} is 5

New vocabulary

Sipser p. 14

- { a,b }* The set of finite strings over the symbols a, b
  - Includes empty string ε
  - Includes a, aa, aaa
  - Includes b, bb, bbb
  - Includes ab, ababab, aaaaaaabbb
  - Does **not** include infinite sequences of a's and b's
  - Has **infinitely many** different elements

- **Alphabet** Nonempty set of symbols
- **String** over alphabet Σ, Element of Σ*
- **Language** over alphabet Σ, Subset of Σ*

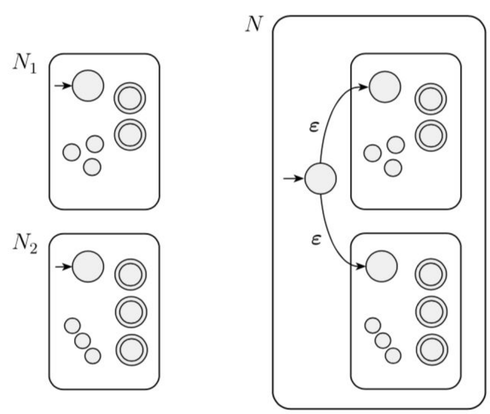
- **1.1. Finite Automata**
  - Move from states to states, depending on the input received.
  - 5-tuple expression - **formal definition**
    - Q - a **finite** set of states
    - ∑ - a **finite** set of alphabet
    - **δ** - transition function

- $q_0 \in Q$ - start state
- $F \subseteq Q$ - set of accept states
- Regular operations - closed
    - **Union** -
        - M_1 = (Q1, sigma, delta1, q1, F1); M_2 = (Q2, sigma, delta2, q2, F2).
        - M = (Q, sigma, epsilon, q0, F)
        - 1. Q = Q1 X Q2
        - 2. Delta((r1, r2), a) = (Delta1(r1, a), Delta2(r2, a))
        - 3. Q0 = (q1, q2)
        - 4. F = (r1, r2) where r1 belongs to F1 or r2 belongs to F2
    - **Concatenation** -
    - **Star** -
    - Proof by construction machines to recognize them.
- A language is called a regular language if some finite automaton recognizes it
- **Only has one unique next state**
- **Given the current state, we know what the next state will be**
- **The number of outgoing arrows must be |∑|**

- **1.2 Nondeterminism**
    - **DFA vs NFA**
        - **Every DFA is a NFA**
        - Every states of DFA has **exactly one transition arrow** for each symbol in the alphabet. NFA may have **n arrows** for each symbol, where **n ≥ 0**.
        - DFA has arrows only on alphabet but NFA might have arrow labeled with **ε**
        - **Every NFA can be converted to some DFA**
        - **Every DFA is a NFA**
    - NFA Computation
        - NFA splits to follow all possibilities in parallel, and if **any one of** the copies of machine is in an accept state at the end of input, NFA accepts.
        - When empty is string is encountered, one copy follow empty string arrow and one stay at current state.
    - **Formal definition of NFA**
        - Q is a finite set of states
        - ∑ is a finite alphabet
        - **δ:** Q X Sigma(empty string) -> P(Q)
        - q_0 belongs to Q: start state
        - F is subset of Q: set of accept states.
    - Equivalence of NFAs and DFAs
        - Two machines are equivalent if they **recognize the same language**
        - Every NFA has an equivalent DFA -> convert NFA to DFA

- NFA has k states, then DFA has 2^k states (number of subsets **but not necessary**).
- If a language is recognized by an NFA, then it is recognized by some DFA. Construct the DFA M as following
    - Q' = P(Q)
    - δ'(R, a) = union of all sets of states original transition function takes to = E(δ(r, a)) ---> 包括empty string
    - q_0' = E{q_0}
    - F' = {R ∈ Q'| R contains an accept state of N}
    - **Consider Ɛ arrows**
        - **We define E(R) to the collection of states that can be reached from members of R by going along Ɛ arrows**
- A language is regular if and only if some NFA recognizes it.
    - Two way
- **Given the current state, there could be multiple next states**

- **The class of regular languages is closed under the regular operations**
    - Union



    - Concatenation

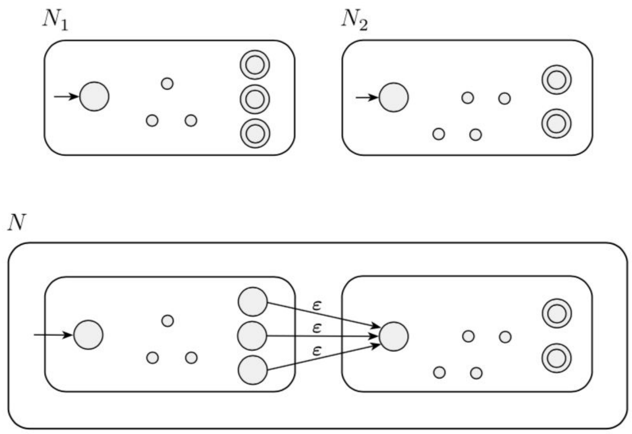


FIGURE 1.48
Construction of N to recognize $A_1 \circ A_2$

- Star operation

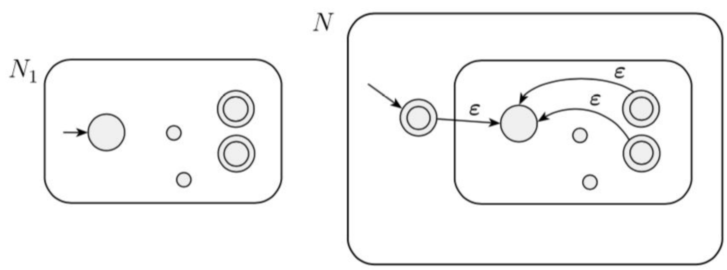


FIGURE 1.50
Construction of N to recognize A^*

- **1.3. Regular Expressions**
  - Formal Definition: R is a regular expression if R is: (inductive definition)
    - a for some a in the alphabet
    - Empty string
    - Ø the language that doesn't contain any string
    - (R1 U R2)
    - (R1 0 R2)
    - (R1*)
    - Note: R+ has all strings that are 1 or more concatenation of strings from R.
  - Equivalence with Finite Automata
    - **A language if regular if and only if some regular expression describes it(exactly recognized by NFA, exactly recognized by DFA)**

- Lemma 1: If language is described by a regular expression, it's regular. (Proof referred to textbook 67)
- Lemma 2: if language is regular, it's described by a regular expression. (Proof refer to text. 68. GNFA??)

- **1.4. Non-regular Expression**

  - **Pumping Lemma**
    If A is a regular language, then there is a number p (the pumping length) where if s is any string in a of length at least p, then s may be divided into three pieces, s = xyz, satisfying the following conditions:
    1. For each i >= 0, x y^i z is an element of A
    2. |y| > 0, and
    3. |xy| ≤ p.
    - Proof idea: pigeonhole principle - sequence contains a repeated state.

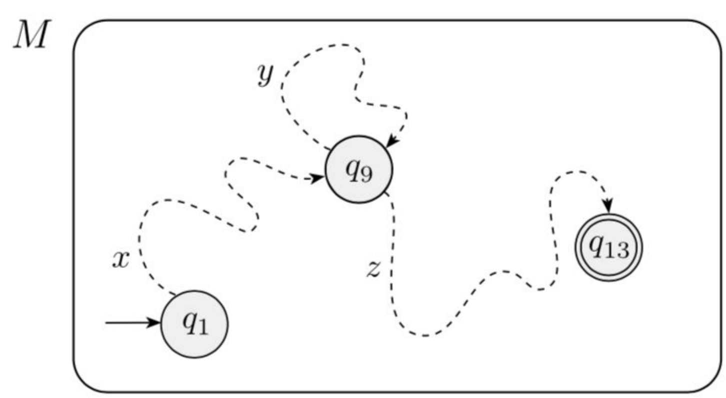


FIGURE 1.72
Example showing how the strings x, y, and z affect M

  - Proving non-regularity
    - Assume B is regular and use pumping lemma. Find a string s in B that has length p or greater in B can be pumped. Then demonstrate that s cannot be pumped by considering all ways of dividing s into x, y, z.

✓ Which of the following sets are countably infinite? (select all that apply) 1/1

- [ ] The set of all languages over {0,1}
- [x] The set of all regular languages over {0,1} ✓
- [x] The set of all strings over {0,1} ✓
- [ ] The set {0,1}
- [x] The set of all DFAs over {0,1} (whose states are labelled by integers) ✓
- [x] The set of all regular expressions over {0,1} ✓