State Machines

University of Massachusetts Amherst
ECE 242 – Data Structures and Algorithms
Lecture 24

Regular expressions

• Method to describe patterns of text
  – Character-by-character processing
  – Special operators
    • | (alternatives)
    • . (arbitrary character)
    • * (zero or more repetitions)
    • + (one or more repetitions)
    • () (precedence)
    • ...
Regular expressions

• Examples
  – abcd
    • abcd matches; aabcd does not match
  – a*bcd
    • aabcd matches; bcd matches; cd does not match
  – (ab|bb)cd
    • abcd matches; bbcd matches; abbbcd does not match
  – (ab|bb)*cd
    • abbbcd matches; bbabcbd matches; cd matches; ababccd matches; abbb does not match

Use of regular expressions

• Compiler
  – Interpreting characters in program
  – Regular expressions for numbers, keywords, etc.
  – Example tool: flex

• Networking
  – Checking network traffic for attacks
  – Regular expressions for attack patterns
  – Example tool: snort database
Limits of regular expressions

• Regular expression match patterns from “regular language”
• Regular expression cannot describe patterns from more complex language
  – What can you not describe with a regular expression?

• Equal number of opening and closing parentheses
• Grammatically correct English language
State machine

- Regular expressions can be matched with a “state machine” (or “finite automaton”)
- SM is special case of directed graph
  - Node represents state
  - Edge represents transition (based on input)
- State machine can be constructed for any regular expression

State machine examples

- Example 1: ac+ | bd

- Example 2: a(b* | c)d
Deterministic vs. non-deterministic

• What is the problem with (ab)*ac?

– Non-deterministic transition on a

• Non-deterministic state machines
  – A bit more complex to implement
  – We do not consider them here
  – There exist algorithms to convert from NFA to DFA
Implementing a state machine

- Vertex with multiple outgoing edges
  - Need class to represent edge
  - Need linked list to store edges

- Matching operation
  - Start at start node
  - Follow edge that matches character
  - At end, check if accepting state
  - If no edge or no accepting state, then no match

Edge class

```java
public class Edge {
    public char character;
    public Vertex destination;

    public Edge(char c, Vertex dest) {
        destination = dest;
        character = c;
    }
}
```
**Vertex class**

```java
public class Vertex {
    public int number;
    Queue<Edge> edgeList;
    boolean isAcceptingState;

    public Vertex (int n) {
        number = n;
        edgeList = new LinkedList<Edge>();
        isAcceptingState = false;
    }
}
```

**Matching method**

- Following edge in vertex:
  ```java
  public Vertex followEdge(char c) {
      Iterator<Edge> i = edgeList.iterator();
      while (i.hasNext()) {
          Edge e = i.next();
          if (e.character == c) {
              return e.destination;
          }
      }
      return null;
  }
  ```
### Matching method

```java
public void match(String s) {
    char[] characters = s.toCharArray();
    Vertex current = start;

    System.out.print("trying to match \"s\": ");
    for (int i=0; i<characters.length; i++) {
        if (current==null) {
            System.out.println("no match");
            return;
        }
        System.out.print(current.number + " ");
        current = current.followEdge(characters[i]);
    }
    if (current==null) {
        System.out.println("no match");
        return;
    }
    ... 
```

### Matching example

- **Graph:**

- **Matching**
  - abc
  - bbabc
  - baab
  - baabcc
  - abcdbbbabc
  - abcd
  - e
Matching example

• Program output:

  trying to match abc: 1 2 3 4 match
  trying to match bbabc: 1 1 1 2 3 4 match
  trying to match baab: 1 1 2 5 6 match
  trying to match baabcc: 1 1 2 5 6 6 6 match
  trying to match abcdddbabc: 1 2 3 4 1 1 1 1 2 3 4 match
  trying to match abcd: 1 2 3 4 1 no match
  trying to match e: 1 no match

Next Steps

• Project deadline has been extended to Monday, Nov 4 11:00 p.m.
• Lecture Monday
• Review for exam
  – Wednesday in class
  – Thursday in discussions