

Regular Expressions

Pattern matching

Substring Search : find a single thing in text

Pattern Matching : find one of a specified set of things in text

Regular Expression - notation to specify a set of things

Operations

- 1) concatenation
- 2) or |
- 3) closure * (0 or more occurrences)
- 4) parenthesis ()
- 5) wildcard .
- 6) character class [A-Za-z] [a-z]
- 7) at least 1 occurrences +

8) exactly k $\{k\}$

* Regular expressions are surprisingly expressive

- substring search

. * SPB. *

- social security numbers

$[0-9]\{3\} - [0-9]\{2\} - [0-9]\{4\}$
 ↖ $\underbrace{\hspace{1.5cm}}_{k \text{ digits}}$

- email

$[a-z]^+ @ ([a-z]^+ \backslash .)^+ (edu / com)$

* RE important in the theory of computation

XI FA

- RE (concise way to describe a string)

- DFA (Discrete Finite Automaton) - machine to recognize whether a given string is in a given set

Kleene's theorem

- for any DFA, there exists a RE that describes the same set of strings
- for any RE, there exists a DFA that recognizes the same set of strings

Pattern matching implementation

Basic Plan

- build a DFA from RE
- simulate DFA with text as input

Bad news

- basic plan is infeasible; DFA may have exponential # of states

Solution - we use NFA instead

NFA = Non-deterministic finite automaton

Regular Expression matching NFA

- RE enclosed in parentheses

- one state per RE character (start=0, accept=M) \nearrow epsilon

- Red ϵ -transition (change state, but don't scan text)

- Black match transition (change state and scan to next text char)

- accept if any sequence of transitions ends in accept state
 after scanning all text characters

* systematically consider all possible transition sequences

NFA Simulation

Representation

- state names - integers from 0 to M
- match transitions - keep regular expression in array $re[]$

- ϵ transitions \Rightarrow store in digraph G
- * maintain set of all possible states that NFA could be in after reading in the first i text characters

How to perform reachability?

- Read next input character
 - find states reachable by match transitions
 - find states reachable by ϵ -transitions
- When no more input characters:
 - accept if any state reachable is an accept state
 - reject otherwise

Digraph reachability

- find all vertices reachable from a given source or set of vertices

Solution : run DFS from each source

without unmarking vertices

* runs in time proportional to $E+V$

* worst case it takes time proportional to N^4

NFA Construction

- concatenation \rightarrow match transitions
- parentheses \rightarrow ϵ -transition edge from parentheses to next state
- closure \rightarrow add 3 ϵ -transition edges for each $*$ operator
 - \exists 2 ϵ -transition edges for each $/$ operator

Challenges - remember left parentheses to implement closure and \exists ; remember $/$ to implement \exists .

Solution

STACK

(push
/ push
) pop

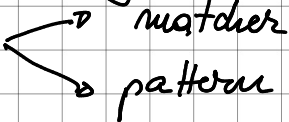
Regular Expression Applications

◦ step \Rightarrow build an NFA

◦ use grep to solve crossword puzzles

◦ RegEx built in programming languages

◦ Java string library `input.matches(re)`

• Java Util 

```
graph LR; A[Java Util] --> B[matcher]; A --> C[pattern]
```

◦ evolution \rightarrow writing a compiler