T-(538|725)-MALV, Natural Language Processing
Regular expressions

Hrafn Loftsson[1] Hannes Högni Vilhjálmsson[1]

[1]School of Computer Science, Reykjavik University

September 2010

Outline

Outline

Strings

An alphabet

- A finite set of symbols or characters.
- Example: {0,1} is the binary alphabet.

A string

- A string s from the alphabet Σ is a finite sequence of characters drawn from Σ.
- $|s|$ denotes the length of s.
- ϵ denotes the empty string; its length is 0.

Strings

Concatenation and multiplication

- If x and y are strings then their concatenation xy is a string obtained by concatenating y to x.
- $s\epsilon = \epsilon s = s$
- $s^0 = \epsilon$, $s^1 = s$, $s^2 = ss$,
- $s^i = ss^{i-1}$, $i > 0$

A language

Definition

- A set of strings.
- Example: $\emptyset$, $\{\epsilon\}$, {ab,ba}, {011,101,111}.

Concatenation of languages

- Concatenation of two languages A and B is the set of all strings that are formed by concatenating the strings of B at the end of the strings of A.
- We can use exponentation to denote concatenation:
- $A^0 = \{\epsilon\}$, $A^1 = A$, $A^2 = AA$, $A^3 = AAA \ldots$

Operations on languages

- $L \cup M = \{s \mid s \in L \text{ or } s \in M\}$
- $LM = \{st \mid s \in L \text{ and } t \in M\}$
- Kleene closure: 0 or more concatenations of L
    - $L^* = \cup_{i=0}^{\infty} L^i$
- Positive closure: 1 or more concatenations of L
    - $L^+ = \cup_{i=1}^{\infty} L^i$

Examples of languages

$L = \{A, B, \ldots, Z, a, b, \ldots, z\}$ and $D = \{0, 1, \ldots, 9\}$.
What languages (set of strings) are:

- $L \cup D$
- LD
- L^4
- L^*
- $L(L \cup D)^*$
- D^+

Outline

Regular expressions (regex)

- A language used to describe a set of strings.
- Very powerful devices to describe patterns to search for in texts.
  - When a particular string is in the set described by a regex, we say that the regex *matches* the string.
- Each regular expression r denotes a language $L(r)$.
- Are composed of ordinary text characters (e.g. *abc*) and metacharacters, e.g "*" and "+".
- Complex regex can be constructed from simple regex using special rules.

Regular expressions

For an alphabet Σ:

1. ϵ is a regex denoting $\{\epsilon\}$.
2. If $a \in \Sigma$, then a is a regex denoting $\{a\}$.
3. Let us assume r and s are regex denoting the languages $L(r)$ and $L(s)$. Then:
   - $(r)|(s)$ is a regex denoting $L(r) \cup L(s)$.
   - $(r)(s)$ is regex denoting $L(r)L(s)$.
   - $(r)^*$ is a regex denoting $(L(r))^*$.
   - (r) is a regex denoting $L(r)$.

Regular expressions

Operator precedence:

- * has the highest precedence.
- Concatenation next highest.
- | has the lowest precedence.
- Accordingly: $(a)|((b)^*(c)) = a|b^*c$

Examples of regular expressions

Which languages denote the regular expressions:

- $a|b$
- $(a|b)(a|b)$
- a^*
- $a|b^*c$

More about regular expressions

Other characters having a special meaning

In many tools which support regex the following characters have a special meaning:

- ? + . {n}
- See descriptions in table 2.9 on page 37

More about regular expressions

Character classes

- A list of characters between square brackets matches any character contained in the list.
- The regex $[abc]$ means one occurrence of either a, or b or c $(a|b|c)$.

Complement and range

- [^a] means any character that is not an a.
- [a-zA-Z] means $a, b, \ldots, z, A, B, \ldots, Z$.

Longest match

Ambiguity

- String matching can be ambiguous.
- For example, the string $s =$"aabbc" and the regex a^+b^*
- This regex matches the following substrings of s: *a*, *aa*, *ab*, *aab*, *abb*, *aabb*

Disambiguation – two rules

Most tools which support regex:

- They match as early as they can in a string.
- They match as many characters as they can.

- Thus, a^+b^* matches *aabb*, the **longest match**.

Regex and finite-state automata

- A regex can be converted automatically to an NFA (non-deterministic FSA).
  - The method is, for example, discussed in the *Compiler* course.
- An FSA can accept the set of strings which a particular regex stands for.

Various tools and programming languages

- grep/egrep (Unix/Linux tool)
  - grep 'ab*c' myFile
  - Prints all the lines from the file *myFile* containing the strings *ac*, *abc*, *abbc*, *abbbc*, etc.
  - In Windows you can install *Cygwin* `http://www.cygwin.com/` which is a Linux-like environment for Windows.
- Support for regular expressions is in various contemporary languages, e.g. Perl, Python, Java, C#.
- Tutorial in Java: `http://java.sun.com/docs/books/tutorial/essential/regex/`