Regular expressions

ian.mcloughlin@gmit.ie

Regular expressions

- Regular expressions are strings that represent patterns of text.
- The strings can contain special characters.
- Brackets can be used to group characters together.
- Regular expressions are used to search other strings for patterns.

Examples

$a.b.c^*$

An a followed by a b followed by zero or more c's.

$a|b.c$

An a, or a b followed by a c.

$(a|b).c$

An a or a b, followed by a c.

$0.0.(0|1)^*$

All strings of 0's and 1' that begin with two zeros.

Special characters

. means *concatenate*. So, $a.b$ means an a followed by a b.

| means *or*. So, $a|b$ means an a or a b.

* means *zero of more times*. So, a^* means zero or more a's.

Precedence

1. Always apply * first.
2. Apply . after * but before |.
3. Apply | last.
4. Treat bracketed groups as individual characters.

Infix and postfix

It is sometimes convenient to re-write expressions in postfix. This applies to lots of different expressions, not just regular expressions.

Example
The infix mathematical expression "$(3 + 4) \times 5$" is "$3\ 4\ +5\ \times$" in postfix.

Example
The infix regular expression "$a.(b.b)^*.a$" is "$abb.^*.a.$" in postfix. Note we often omit the . in infix notation: "$a(bb)^*a$" but can't in postfix. However, the brackets aren't needed in postfix.

Executing regular expressions against strings

$$r = \text{“}(0^*1^*|1^*0^*)\text{”}$$

$$s = \text{“}00000000011111111111111111\text{”}$$

- Regular expressions are *executed* against strings.
- This means an algorithm determines if the string (s) matches the pattern as defined by the regular expression (r).
- We can ask two related questions: does the whole string match, or does a substring of it match?
- We can write algorithms to answer either question.