RePeg

Edman Paes Anjos   Srgio Queiroz de Medeiros

July 8, 2012

A pattern matching tool that translates regular expressions to identical Parsing Expression Grammars (PEGs). We implement the regular expression's semantics using PEGs. , so that the user doe.

1 The RePeg Library

RePeg is a library for pattern matching in the Lua programming language. It uses most of the traditional and well-known PCRL syntax of regular expressions, therefore it does not requires the user any new knowledge, providing a familiar environment. In this text you can find a reference manual for the library.

The table 1 describes the syntax recognized by RePeg for regular expressions. Here the `a` or `b` represent a single character; `s` represents a string of characters; `p` represents a pattern; `num` represents a number (`[0-9]+`).

2 Functions

2.1 `RePeg.match (pattern, subject)`

`pattern` → a string describing a regular expression
`subject` → the string of characters to be matched against the `pattern`

Matches directly a `pattern` to a string, returning the portion of the `subject` succesfully matched.

Syntax	Description
`(?: p )`	grouping
`( p )`	capture
`.`	any character
`‘’`	empty string
`‘s’`	literal string
`$`	end of input
`\z`	end of line or end of input
`\Z`	end of input preceded or not by end of line
`[a-b]`	character range
`p1 / p2`	choice
`p1 p2`	concatenation
`?= p`	and predicate
`?! p`	not predicate
`p ?`	optional match
`p *`	zero or more repetitions
`p +`	one or more repetitions
`p *+`	possessive repetition
`p *?`	lazy repetition
`p { num }`	exactly `num` repetitions
`p { num , }`	`num` repetitions or more
`p { num1 , num2 }`	between `num1` and `num2` repetitions, inclusive

2.2 `RePeg.find (pattern, subject)`

`pattern` → a string describing a regular expression
`subject` → the string of characters to be matched against the `pattern`

Seeks for the first substring of the `subject` that can be matched by the given `pattern`. If it matches more than one substring, return the largest.

3 Usage Examples

3.1 A simple program

The following code specifies a running Lua program

```
repeg = require 'repeg'
```

```
-- find the first number in a string
string = "this string has 29 characters"
print(repeg.find("[0-9]+", string)) --> {28}
print(repeg.match(".*? ([0-9]+)", string)); --> {28}
```