CM0081 Formal Languages and Automata
Regular Expression in Haskell: An Introduction

Andrés Sicard-Ramírez

Universidad EAFIT

Semester 2024-1

Preliminaries

Conventions

- The number and page numbers assigned to chapters, examples, exercises, figures, quotes, sections and theorems on these slides correspond to the numbers assigned in the textbook [Hopcroft, Motwani and Ullman (1979) 2007].
- The natural numbers include the zero, that is, $\mathbb{N} = \{0, 1, 2, \dots\}$.
- The power set of a set A, that is, the set of its subsets, is denoted by $\mathcal{P}A$.

Introduction

- There are various libraries for handling regular expressions in HASKELL.
- POSIX (Portable Operating System Interface) is a family of standards specified for maintaining compatibility between operating systems.

Notation for Regular Expressions

POSIX	Textbook
ab	ab
$a\|b$	$a + b$
$a*$	a^*
(a)	(a)
$a+$	aa^*
$a?$	$a + \varepsilon$
$[abc]$	$a + b + c$
$.$	Any symbol

Demo

Examples

We shall use GHC 9.6.2, the libraries REGEX-POSIX 0.96.0.1 and REGEX-TDFA 1.3.2.1[†] and we shall see some examples from [O'Sullivan, Goerzen and Stewart 2008, Ch. 8]

[†]Hackage: `https://hackage.haskell.org/package/regex-posix` and `https://hackage.haskell.org/package/regex-tdfa`, respectively.

Other Libraries

From the description of REGEX-BASE 0.94.0.2:[†]

> *This package does not provide the ability to do regular expression matching. Instead, it provides the type classes that constitute the abstract API that is implemented by regex-* backends such as:*
>
> - REGEX-POSIX
> - REGEX-PARSEC
> - REGEX-DFA
> - REGEX-TDFA
> - REGEX-PCRE

[†] https://hackage.haskell.org/package/regex-base.

Other Libraries

From the description of REGEX-POSIX 0.96.0.1:[†]

> *Benchmarking shows the default regex library on many platforms is very inefficient. You might increase performace by an order of magnitude by obtaining* LIBPCRE *and* REGEX-PCRE *or* LIBTRE *and* REGEX-TRE. *If you do not need the captured substrings then you can also get great performance from* REGEX-DFA. *If you do need the capture substrings then you may be able to use* REGEX-PARSEC *to improve performance.*

[†] https://hackage.haskell.org/package/regex-posix-0.96.0.1/docs/Text-Regex-Posix.html.

References

- Hopcroft, J. E., Motwani, R. and Ullman, J. D. [1979] (2007). Introduction to Automata Theory, Languages, and Computation. 3rd ed. Pearson Education (cit. on p. 2).
- O'Sullivan, B., Goerzen, J. and Stewart, D. (2008). Real World Haskell. O'Really Media, Inc. (cit. on p. 5).