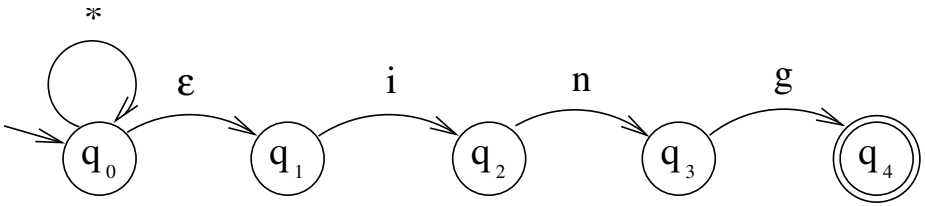
Natural Language Processing
CSCI 4152/6509 — Lecture 4
Regular Expressions and Perl

Instructors: Vlado Keselj
Time and date: 16:05 – 17:25, 16-Sep-2024
Location: Carleton Tupper Building Theatre C

Previous Lecture

- **Part II: Stream-based Text Processing**
- Finite state automata
  - Deterministic Finite Automaton (DFA)
  - Non-deterministic Finite Automaton (NFA)
- Review of Deterministic Finite Automata (DFA)
- Non-deterministic Finite Automata (NFA)
- Implementing NFA, NFA-to-DFA translation



(to finish)

Finite Automata in NLP

- Useful in data preprocessing, cleaning, transformation and similar low-level operations on text
- Useful in preprocessing and data preparation
- Efficient and easy to implement
- Regular Expressions are equivalent to automata
- Used in Morphology, Named Entity Recognition, and some other NLP sub-areas

Regular Expressions

- Review of regular expressions (for some of you, it was covered in earlier courses as well)
- Used as patterns to match parts of text
- Equivalent to automata, although this may not be obvious
- Provide a compact, algebraic-like way of writing patterns
- Example: `/Submit (the )?file [A-Za-z.-]+/`

Some References on Regular Expressions

You can find many references on Regular Expressions, including:

- Chapter 2 of the textbook [JM]
- Perl "Camel book" or many resources on Internet
- On timberlea server: '`man perlre`' and '`man perlretut`'
- The same effect: '`perldoc perlre`' and '`perldoc perlretut`'
- Or on the web: `http://perldoc.perl.org/perlre.html` and `http://perldoc.perl.org/perlretut.html`

A Historical View on Regular Expressions

- Research by Stephen Kleene: regular sets, and the name of regular sets and regular expressions (1951),
- Implementation in QED by Ken Thompson (1968),
- Open-source implementation by Henry Spencer (1986),
- Use in Perl by Larry Wall (1987),
- Perl-style Regular Expressions in many modern programming languages.

Example Regular Expressions

- Literal: `/woodchuck/` `/Buttercup/`
- Character class: `/./` (any character),
  `/[wW]oodchuck/`, `/[abc]/`, `/[12345]/`
(any of the characters)
- Range of characters: `/[0-9]/`, `/[3-7]/`, `/[a-z]/`,
`/[A-Za-z0-9_-]/`
- Excluded characters and repetition: `/[^()]+/`
- Grouping and disjunction: `/(Jan|Feb) \d?\d/`
- Note: `\d` is same as `[0-9]`
- Another character class: `\w` is same as `[0-9A-Za-z_]`
('word' characters)
- Opposite: `\W` same as `[^0-9A-Za-z_]`

RegEx Examples: Anchors, Grouping, Iteration

```
/^This is a/   # use of anchor
/This^or^that/ # not an anchor
/woodchucks?/
/\bcolou?r\b/        # anchor \b
/is a sentence\.$/  # end of string anchor

# Grouping and iteration:
/This sentence goes on(, and on)*\.$/
/cat|dog/            # disjunction (alternation)
/The (cat|dog) ate the food\./
```

Introduction to Perl

- Created in 1987 by Larry Wall
- Interpreted, but relatively efficient
- Convenient for string processing, system admin, CGIs, etc.
- Convenient use of Regular Expressions
- Larry Wall: Natural Language Principles in Perl
- Perl is introduced in lab in more details

Why Perl?

- We only cover Perl with regular expressions and basic text processing
- Perl is very convenient for these types of tasks
- Perl uses very direct way of using regular expressions
- Perl is still used a lot in text processing, NLP, bioinformatic string processing, etc.
- Perl-style regular expressions are very important in any programming languages for NLP

Testing Code (as shown in labs)

- Login to timberlea
- Use plain editor, e.g., emacs
- Develop and test program
- Submit assignments
- You can use your own computer, but code must run on timberlea

Regular Expressions in Perl

- We will learn more about Regular Expressions by using Perl in simple text processing.
- Let us start with a program to count lines in a text

Perl Program for Counting Lines

```
#!/usr/bin/perl
# program:  lines-count.pl

while (<>) {
   ++$count;
}

print "$count\n";
```

Regular Expressions in Perl

- Perl provides an easy use of Regular Expressions
- Consider the regular expression: `/pro...ing/`
- Run the following commands on timberlea:

```
cp ~prof6509/public/linux.words .
grep proc...ing linux.words
```

- Output includes 'processing', and more:

```
coprocessing
food-processing
microprocessing
misproceeding
multiprocessing
...
```

Note About File 'linux.words' and Others

- Some helpful files can be found on timberlea in: ~prof6509/public/
- or, on the web at: http://web.cs.dal.ca/~vlado/csci6509/misc/
- For example:

```
linux.words
wordlist.txt
Natural-Language-Principles-in-Perl-Larry-Wall.pdf
TomSawyer.txt
```

Perl Regular Expressions: 'proc...ing' Example

- Similar functionality as grep:

```
#!/usr/bin/perl
# run as: ./re-proc-ing.pl linux.words

while ($r = <>) {
  if ($r =~ /proc...ing/) {
    print $r;
  }
}
```