UNIX/Linux environment

Regular expressions and find

Stefano Quer - Pietro Laface
Dipartimento di Automatica e Informatica
Politecnico di Torino

Regular expressions

- Introduced in 1956 by the mathematician Stephen Cole Kleene in the automaton and formal language domain
- Used from the seventies in the UNIX environment
  - Editors (vi, emacs, etc.)
  - Shell commandi di shell (find, grep, etc.)
  - Scripting languages (sed, awk, perl, python, etc.)

Regular expressions

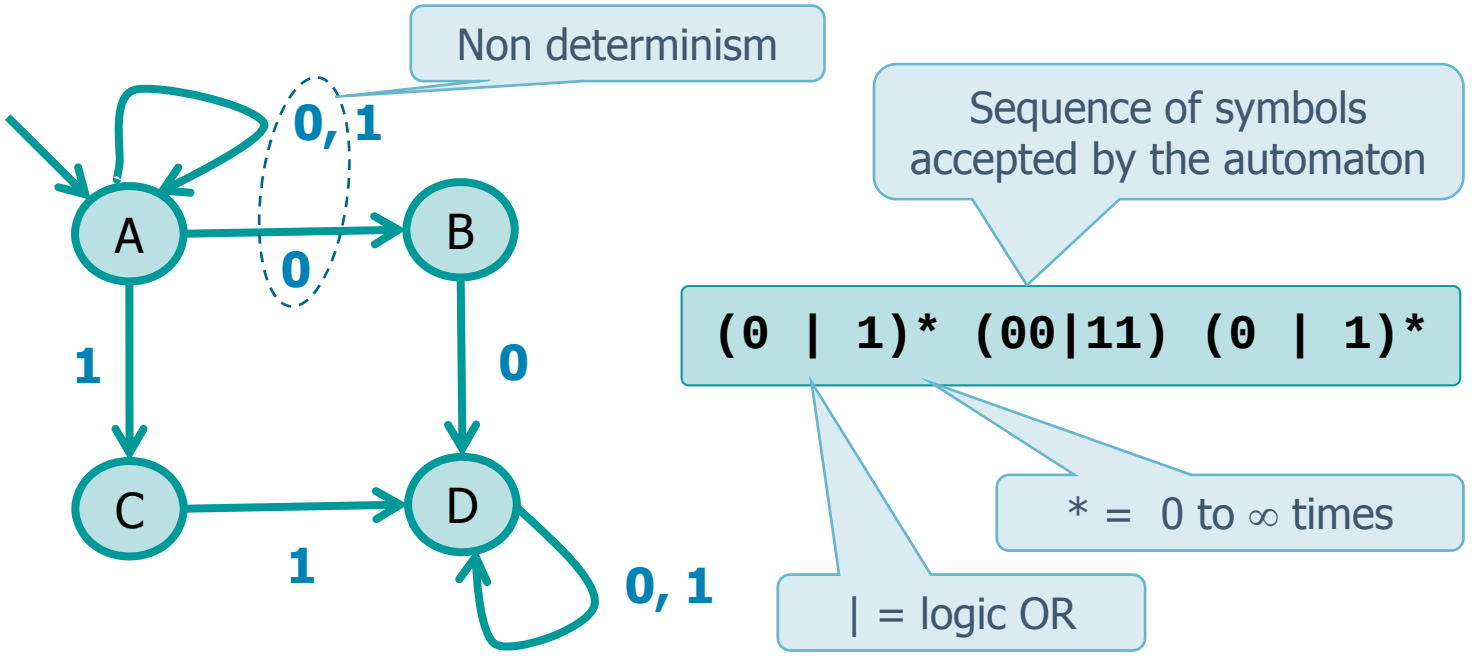
- A standard defined for POSIX in 1992
- Several versions exist, using similar but different formalisms
  - BRE, Basic Regular Expression
  - ERE, Extended Regular Expression
  - PCRE, Perl Compatible Regular Expression
    - C Library of Regular Expressions (Hazel, 1997)
    - More flexible than POSIX version
    - De-facto standard with Perl 5
- In this course
  - We will use simple regular expressions in shell scripting find, grep, awk

Regular expressions

- A regular expression (or **pattern**) is an expression that specifies a set of strings
  - Compact operators are used to represent complex sequences of characters
    - Example
      - a | b* represents the set of strings {a, ϕ, b, bb, bbb, bbbb, …}
- Expressions are useful to find if a **match** exists between objects
  - Directories or file names, lines of fields of a file, strings or sub-strings, etc.

Regular expressions and automata

- A regular expression corresponds to a Non Deterministic Automaton (NFA)



Definitions

- Literal
  - Any character (or character sequence)
    - ind in w**ind**ows, **ind**ifferent, etc.
- Metacharacter
  - One or more characters having special meaning
    - * indicates 0 to ∞ preceding symbols, e.g., b* = {ϕ, b, bb, bbb, ...}
- Escape sequence
  - Allows using literally a metacharacter
    - Character '.' must be given as '\.'

Metacharacters

Operator	Meaning
[...]	Specifies a list or range of symbols
(...)	Manages operator precedence Groups sub-expressions Allows reference to previous expressions (backward reference)
|	Logical OR

Basic RE: \[...\] and \(...\)

Anchors	Meaning
\<	Beginning of word
\>	End of word
^	Beginning of line
$	End of line

Special characters	Meaning
\+ \? \.	Characters '+', '?', '.'
\n	New line
\t	TAB

Meta-characters

Characters	Meaning
c	Any symbol c (excluding special ones)
.	Any character (excluding '\n')
\c	Any control character
\s	A space or TAB
\d	A digit
\D	Not digit
\w	Any character 0-9, A-Z, a-z
\W	Any character excluding 0-9, A-Z, a-z

Some are not recognized by some commands

Meta-characters

Quantifiers And ranges	Meaning
*	[0, ∞] times
+	[1, ∞] times
?	[0, 1] times
$[c_1c_2c_3]$	Any character in parenthesis
$[c_1\text{-}c_5]$	Any character in range
$[\hat{}c_1\text{-}c_5]$	Any character not in range
{n}	Exactly n times
$\{n_1,n_2\}$	n_1 to n_2 times

Superset

grep command:
Allows also $\{n_1,\}$ or $\{,n_2\}$

Examples

Regular expression	Meaning
ABCDEF	String "ABCDEF"
a*b	Any number of 'a' followed by a single 'b'
ab?	a or ab
a{5,15}	5 to 15 repetitions of 'a'
(fred){3,9}	3 to 9 repetitions of string "fred"
.+	Any, non empty, sequence
myfunc.*(.*)	A function with name beginning by "myfunc"
^ABC.*	A line beginning by "ABC"
.*h$	A line ending by "h"
hello\>	Word ending by "hello"
a+b+	One or more 'a' followed by one or more 'b'

Examples

Regular expression	Meaning
.*b.*3	Matches string "./fbar3"
[a-zA-Z0-9]	A letter or a digit
A|b	A or b
\w{8}	A 8 character word
((4\.[0-2])| (2\.[0-2]))	Numbers 4.0, 4.1, 4.2 or 2.0, 2.1, 2.2
(.)\1	Two times the same character (ex. "aa")
(.)(.).\2\1	Any 5 character palindrome string (e.g., radar, civic, 12321, etc.)

Backward Reference

find

- Allows
  - Searching and listing the file, directories or links that match a given criterion
  - Possibly executes a shell command on every listed file
- Notice that
  - Find outputs the relative path of the matching files, not their basenames

find

- Format
  - find directory options actions
  - Visits the **directory** subtree
  - Outputs the list of pathnames satisfying the **options**
  - Possibly performs the **actions** on every file of the list

find directory

- Specifies the search directory tree
  - .
  - /usr/bin
  - ./subDirA/subDirB

find options

Option	Meaning
-name pattern	Searches the files matching the pattern. **-iname** is case insensitive
-regex expr	Specifies a regular expression that matches the found relative path **-iregex** is case insensitive
-regextype type	Indicates the type of regular expressions used: posix-basic, posix-egrep, posix-extended, etc. (**regextype** must precede **regex**)

find options

Option	Meaning
-atime [+,-]n -ctime [+,-]n -mtime [+,-]n	Last access, status or modification time n=1 specifies from 0 to 24 hours back n value with sign : + means ≤, - means ≥
-size [+,-]n[bckwMG]	File dimension Sign + means ≥, - means ≤ Next character indicates the size : b blocks (of 512 bytes), c bytes, etc.
-type type	File type **f** per file regular (i.e., text files, executable, etc.), **d** for directories, **p** for pipes, **l** for symbolic links, **s** for sockets

find options

Opzione	Significato
-user name -group name	File owner identifier File group identifier
-readable -writable -executable	File access permissions
-mindepth n -maxdepth n -quit	Search limited to a subtree section: **mindepth** and **maxdepth** indicate the minimum and maximum depth from **directory** (-maxdepth 1 means search only on **directory**) Quit the search after the first match

Examples: find & options

```
find . -name "*.c"
find . -regex "*.c"
find . -regex ".*\.c"
find /usr/bin -iname "a.*"
```

File with `".c"` extension

Wrong: `"*.c"` is not a RE

Correct equivalent RE

All 'a' or 'A' files that do have an extension

All files with dimension >500 bytes

All readable files nel in the current directory (./) with name beginning by ab,aab, aaab, and any extension

```
find . -size +500c

find . -readable \
  -regex "\./a+b.*\.*
```

Examples: find & options

All files with extension **exe** in directory **/home/usr** form level 2 to 4 (included)

```
find /home/usr/ \
  -mindepth 2 -maxdepth 4 \
  -name "*.exe"
```

find: actions

- The default action of **find** is to output the list of matching files
  - The default action is equivalent to the **print** command
    - find directory options **`-print`**
- It is possible to execute a shell command on every pathname of the matching list

find: actions

- The execution of a command is performed by means of the option **`exec`** (o **`execdir`**)
  - Format
    - **`find directory options -exec command {} \;`**
  - Where
    - The **command** is executed in the directory
      - in which the pathname has been found by using **`execdir`**
      - in the current directory if **`exec`** has been used
    - find substitutes string **`"{}"`** with the current pathname of the list
    - String **`"\;"`** terminates the command executed by find

Examples: find & actions

```
find . –name "*.old" -type f –exec rm –f {} \;
```

Removes all files **.old** found in the current directory and, recursively, in all its subdirectories

```
find / -user root –exec cat {} \;
```

Outputs the content of each file of the root filesystem belonging to user **root**

```
find . –name "*.txt" –exec head –n 2 \{} \;
```

Outputs the first two lines of each **.txt** file

Examples: find & actions

```
find /home/usr/ \
  -mindepth 2 -maxdepth 2 \
  -name "*.exe" \
  -type f \
  -exec chmod +x \{} \;
```

Adds the execution permissions of the **".exe"** files in the second level directories of "/home/usr/"