Pattern searching with Regular Expressions

Class Meeting 4

* Notes adapted by Alexey Onufriev from previous work by other members of the CS faculty at Virginia Tech

But first, let's set up your shell.

Unix Shell Environments

Class Meeting 4, Part I

* Notes adapted by Alexey Onufriev   previous work by other members of the CS faculty at Virginia Tech

Shell Characteristics

- Command-line interface between the user and the operating system
- A **program** that automatically starts on login, waits for user to type in commands
- A **command interpreter** that uses a **programming language**
- **Shell script** is a text file containing logic for shell interpretation

Environment Variables

- A set of variables the shell uses for certain operations
- Variables have a **name** and a **value**
- Current list can be displayed with the `env` command
- Dispaly value of `varname` with `echo $varname`

Environment Variable Examples

```
[cs2204@acorn bin]$ echo $PS1
[\u@\h \W]\$

[cs2204@acorn bin]$ echo $PATH
/usr/local/bin:/bin:/usr/X11R6/bin:/usr/ga
  mes:/home/courses/
      cs2204/bin
```

Searching for patterns.

- Problem: you have 10,000 files in some directory tree, and you need to find ones that contain words “pattern search” in the beginning of line.
- Problem 2: Yiou have a file with your financial records for the past 10 years. You need to find total $$ you paid for food.

What is a Regular Expression?

- A **regular expression** (RE) is a string of **characters** that specifies a set of strings
- Each of these strings is said to **match** the regular expression
- **Pattern matching** is useful in many real-world situations:
  - searching for a file on the file system
  - finding and replacing text in a file
  - extracting data elements from a database

Unix programs that use REs

- `grep` (search within files)
- `egrep` (`grep` with extended REs)
- `vi`/`emacs` (text editors)
- `awk` (pattern scanning language)
- `perl` (scripting language)

Characters vs. metacharacters

- In patterns, characters can be any character except a newline
- Metacharacters are special characters that have a special meaning
- To use metacharacters as regular (literal) characters in a pattern, “quote” them with the ‘\’ character

Using egrep

- `egrep pattern filename(s)`
- To be safe, put quotation marks around your pattern
- Examples:
  - `egrep "abc" textfile`
    - Print lines in `textfile` containing “abc”
  - `egrep -i "abc" textfile`
    - Same, but ignores case (e.g. matches “aBc”)
  - `egrep -v "abc" textfile`
    - Print lines in `textfile` **NOT** containing “abc”
  - `egrep -n "abc" textfile`
    - Same as first example, but includes line numbers
  - `egrep -c "abc" textfile`
    - Same as first example, but prints # of lines

Metacharacters

- Period (`.`): matches **any single** character
  - `a.c` matches `abc`, `adc`, `a&c`, and `a;c`
  - `u..x` matches `unix`, `uvax`, and `u3(x`
- Asterisk (`*`): matches **zero or more occurrences** of the previous RE
  - **not** the same as wildcards in the shell
  - `ab*c` matches `ac`, `abc`, `abbc`, and `abbbc`
  - `.*` matches any string

Metacharacters (cont)

- Plus (+): matches **one or more occurrences** of the preceding RE
  - `ab+c` matches `abc`, `abbc`, `abbbc`, but not `ac`
- Question Mark (?): matches **zero or one occurrences** of the preceding RE
  - `ab?c` matches `ac` or `abc`, but not `abbc`
- Logical Or (|): matches RE before | **or** RE after |
  - `abc|def` matches `abc` or `def`

Metacharacters (cont)

- Caret (^): beginning of line
  - `^D.*` matches a line beginning with `D`
- Dollar Sign ($): end of line
  - `.*d$` matches a line ending with `d`
- Backslash (\): escapes other metacharacters
  - `file\.txt` matches `file.txt`, but not `file_txt`

Metacharacters (cont)

- Square Brackets [ ]: specifies a set of characters as a list
  - any character in the set will match
  - ^ before the set negates the set
  - – specifies a character **range**
  - Examples:
    - `[fF]un` matches `fun` and `Fun`
    - `b[aeiou]g` matches `bag`, `beg`, `big`, `bog`, `bug`
    - `[A-Z].*` matches a string starting with a capital letter
    - `[^abc].*` matches any string not starting with a, b, or c

Metacharacters (cont)

- Parentheses `()`: used for grouping
  - `a(bc)*` matches a, `abc`, `abcbc`, `abcbcbc`
  - `(foot|base)ball` matches football or baseball
- Braces `{}`: specify the number of repetitions of an RE
  - `[a-z]{3}` matches three lowercase letters
  - `m.{2,4}` matches strings with `m` followed by between 2 and 4 characters

What do these mean?

- `egrep "^B.*s$" file`
- `egrep " [0-9]{3}" file`
- `egrep "num(ber)? [0-9]+" file`
- `egrep "word" file | wc -l`
- `egrep "[A-Z].*\?" file`

Practice

- Construct egrep commands that find in `file`:
  - Lines beginning with a word of at least 10 characters
  - Lines containing a SSN in standard 3-part form
  - Lines with 2 consecutive capitalized words
  - Number of lines not ending in an alphabetic character
  - Lines containing a word beginning with a vowel at the end of a sentence

egrep notes

- Remember, RE matches largest possible string
  - `--color` option illustrates the largest match
- Lots of other useful options; see the man page or your textbook

Let's practice just a bit.

- Use sample.txt and examples from 4.html