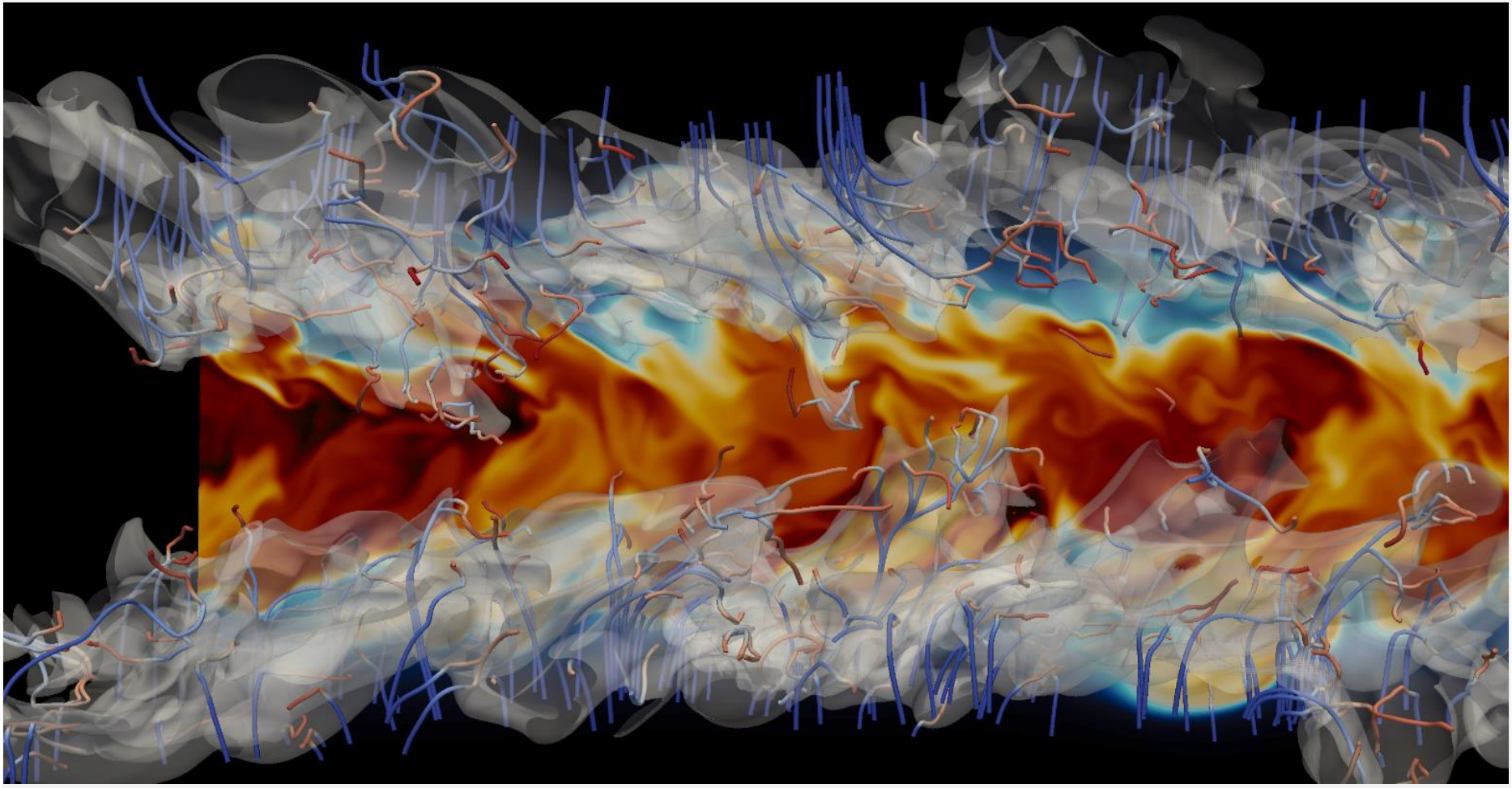
Software Tools for UNIX/Linux Systems

Part 3: find, grep, regular expressions

C. Hasse

TECHNISCHE UNIVERSITÄT DARMSTADT

STFS



Overview

1 Text-searching with grep

2 File-searching with find

3 Regular Expressions

4 Best practice

Overview

1 Text-searching with grep

2 File-searching with find

3 Regular Expressions

4 Best practice

TECHNISCHE UNIVERSITÄT DARMSTADT

What ist grep ?

- command-line text-search utility
- **g**lobal **r**egular **e**xpression **p**rint (abbr. grep)
- syntax:

  grep [options] [regexp] [filenames]
- important options:

Option	Description
-v	Select non-matching lines (vary)
-l	Print only filenames containing matches inside (list)
-n	Print linenumber with output lines
-i	Ignore case distinctions
-r	Search recursively
-c	Suppress normal output; instead print a count of matching lines for each input file.

Grep

TECHNISCHE UNIVERSITÄT DARMSTADT

```
$> cat fruitlist.txt
apple
pear
melon
pineapple
rockmelon
grape

$> grep apple fruitlist.txt
apple
pineapple

$> grep mandarine fruitlist.txt

$> grep MelOn fruitlist.txt

$> grep -i MelOn fruitlist.txt
melon
rockmelon
```

```
$> grep -v apple fruitlist.txt
pear
melon
rockmelon
grape

$> grep -n apple fruitlist.txt

1:apple
4:pineapple

$> grep -c apple fruitlist.txt
2

$> grep -l apple fruitlist.txt
fruitlist.txt
```

Grep

```
$> cat text1.txt
This is dummy file1.

$> cat text2.txt
This is dummy file2 with some more information.

$> cat text3.txt
This is dummy file3 with some more information.
Here is a second dummy line.

$> grep dummy *.txt
text1.txt:This is dummy file1.
text2.txt:This is dummy file2 with some more information.
text3.txt:This is dummy file3 with some more information.
text3.txt:Here is a second dummy line.

$> grep some *.txt
text2.txt:This is dummy file2 with some more information.
text3.txt:This is dummy file3 with some more information.
```

Overview

1 Text-searching with grep

2 File-searching with find

3 Regular Expressions

4 Best practice

- command-line utility for file-search and even more
- syntax:
  - **find** [flag] [paths] [expressions]
- important flags:

Flag	Description
-P	Symbolic links excluded (default)
-L	Symbolic links

- expressions can consist of:
  - operators
  - options
  - tests
  - actions

find

Operator (decreasing precedence)	Description
\(.\)	Grouping of expressions
\! -not	Logical NOT
. –a . . –and .	Logical AND
. –o . . –or .	Logical OR
. , .	List operator, both expressions are evaluated

Test	Description ([i] means case insensitve)
-[i]name pattern	Base of file name matches shell pattern pattern
-[i]regexp pattern	File name matches regular expression pattern
-type c	File is of type c (f for file, d for directory)
-path pattern	File name matches shell pattern pattern.

Option	Description
-mount/xdev	Don’t descend directories on other filesystems

find

Action	Description
-delete	Delete the files
-exec command	Execute command; All following arguments to find are taken to be arguments to the command until an argument consisting of '\;' is encountered. The string '{}' is replaced by the current file name being processed everywhere it occurs
-ls	List current file in ls -dils format on standard output
-print	Print the full file name on the standard output, followed by a newline. (default)
-prune	If the file is a directory, do not descend into it

find

TECHNISCHE UNIVERSITÄT DARMSTADT

$> **find . -name 'my*'**
./mydir1
./mydir1/mytext1
./mydir1/mytext2
./mydir2
./mydir2/mytext1
./mydir2/mytext2
./myfile1
./myfile2

$>**find . -name "my*" -type f**
./mydir1/mytext1
./mydir1/mytext2
./mydir2/mytext1
./mydir2/mytext2
./myfile1
./myfile2

$> **find . -name "my*" -type f -ls**
-rw-r--r-- 1 messig iec 0 29 Mai 16:45 ./mydir1/mytext1
-rw-r--r-- 1 messig iec 0 29 Mai 16:45 ./mydir1/mytext2
-rw-r--r-- 1 messig iec 0 29 Mai 16:45 ./mydir2/mytext1
-rw-r--r-- 1 messig iec 0 29 Mai 16:45 ./mydir2/mytext2
-rw-r--r-- 1 messig iec 0 29 Mai 16:45 ./myfile1
-rw-r--r-- 1 messig iec 0 29 Mai 16:45 ./myfile2

$> **find . -path ./mydir1 -prune -o -type f -name "my*" –print**
./mydir2/mytext1
./mydir2/mytext2
./myfile1
./myfile2

$> **find . \(-name "*text1*" -o -name "*file1*" \) -type f**
./mydir1/mytext1
./mydir2/mytext1
./myfile1

find

```
$> ls -la
-rwxr-xr-x  1 messig  iec  493 29 Mai 16:45 exp2.sh
-r-xr-xr-x   1 messig  iec    0  7 Jun 13:35 file1.txt
-r--rw-rw-  1 messig  iec    0  7 Jun 13:35 file2.txt
-rwxr-xr-x  1 messig  iec    0  7 Jun 13:35 script.sh

$> find . -name '*.txt' -exec chmod 644 {} \;

$> ls -la
-rwxr-xr-x  1 messig  iec  493 29 Mai 16:45 exp2.sh
-rw-r--r--    1 messig  iec    0  7 Jun 13:35 file1.txt
-rw-r--r--    1 messig  iec    0  7 Jun 13:35 file2.txt
-rwxr-xr-x  1 messig  iec    0  7 Jun 13:35 script.sh

$> find . -name "file1*" –delete

$> ls -la
-rwxr-xr-x  1 messig  iec  493 29 Mai 16:45 exp2.sh
-rw-r--r--    1 messig  iec    0  7 Jun 13:35 file2.txt
-rwxr-xr-x  1 messig  iec    0  7 Jun 13:35 script.sh
```

find

$> **cat text1.txt**
This is dummy file1.

$> **cat text2.txt**
This is dummy file2 with some more information.
Here is a second dummy line.

$> **find . -name "*.txt" -exec grep "dummy" '{}' \;**
This is dummy file1.
This is dummy file2 with some more information.
Here is a second dummy line.

$> **find . -name "*.txt" -exec grep ”some" '{}' \; -print**
This is dummy file2 with some more information.
./text2.txt

$> **find . -name "*.txt" | grep ”some”**

$> **find . -name "*.txt" | xargs grep "some"**
./text2.txt:This is dummy file2 with some more information.

Overview

1. Text-searching with grep
2. File-searching with find
3. Regular Expressions
4. Best practice

Regular expressions

TECHNISCHE UNIVERSITÄT DARMSTADT

- Wikipedia says:

  > In computing, a ***regular expression*** provides a concise and flexible means for "matching" (specifying and recognizing) strings of text, such as particular characters, words, or patterns of characters

- Basic concepts (with “\” are BRE(1), without ERE(2)):

Concept		Syntax	Example
Boolean OR	∞	|	gray|grey can match “gray” or “grey
Grouping	∞	\(\)	gray|grey and gr\(a|e\)y are equivalent
Quantification	0-1	\?	colou\?r matches "color" and "colour”.
	0-	*	ab*c matches "ac", "abc", "abbc", …
	1-	\+	ab\+c matches "abc", "abbc", …, but not "ac"

(1) BRE – Basic Regular Expressions
(2) ERE - Extended Regular Expressions

Regular expressions

TECHNISCHE UNIVERSITÄT DARMSTADT

Syntax	Description	Example
[ ]	Matches a single character that is contained	[abc] matches "a", "b", or "c"
.	Matches any single character	a.c matches "abc", etc.
[^]	Matches a single character that is not contained	[^b]at matches all strings matched by .at except "bat".
^ \<	Matches the starting position of the line/word	^[hc]at matches "hat" and "cat" only at beginning of line
$ \>	Matches the ending position of line/word	
\(\)	Defines a marked subexpression	see next section
\n	Matches what the nth marked subexpression matched, where n is a digit from 1 to 9	
\{m,n\} \{m,\} \{m\}	Matches the preceding element at least m and not more than n times. Default for n is ∞. Last syntax means exact m occurrences.	a\{2,4\} matches only "aa", "aaa" and "aaaa"

Regular expressions

TECHNISCHE UNIVERSITÄT DARMSTADT

POSIX[(1)]	ASCII	Description
[:alnum:]	[A-Za-z0-9]	Alphanumeric characters
[:alpha]	[A-Za-z]	Alphabetic characters
[:blank:]	[\t]	Space and tab
[:digit:]	[0-9]	Digits
[:lower:]	[a-z]	Lowercase letters
[:upper:]	[A-Z]	Uppercase letters
[:xdigit:]	[A-Fa-f0-9]	Hexadecimal digits

(1) POSIX - Portable Operating System Interface

Overview

1 Text-searching with grep

2 File-searching with find

3 Regular Expressions

4 Best practice

Best practice

TECHNISCHE UNIVERSITÄT DARMSTADT

```
$> cat pricelist.txt
123
123$
120
120$

$> grep '[1-9]$' pricelist.txt
123

$> grep '[1-9]\$' pricelist.txt
123$

$> grep '[0-9]\$' pricelist.txt
123$
120$

$> grep '[:numbers:]\$' pricelist.txt
123$
120$
```

```
$> cat installfiles.txt
installfile
install.file
installLinux.file
installMac.files

$> grep 'install*file' installfiles.txt
installfile

$> grep 'install.*file' installfiles.txt
installfile
install.file
installLinux.file
installMac.files

$> grep 'install\.*file' installfiles.txt
installfile
install.file

$> grep 'install.*\.file' installfiles.txt
install.file
installLinux.file
installMac.files
```

Best practice

Please note: "\" could be avoided with egrep or grep -e

```
$> cat numbers.txt
15
150
1500
15000
1500012
15000abc
21500
150150
150150150
21502
2150150

$> grep '150\?\>' numbers.txt
15
150
150150
150150150
2150150

$> grep '150\{2\}\>' numbers.txt
1500
21500
```

```
$> grep '150\{2,\}\>' numbers.txt
1500
15000
21500

$> grep '\<150\{1,2\}\>' numbers.txt
150
1500

$> grep '\<\(150\)\{1,2\}\>'
numbers.txt
150
150150

$> grep '\(2\)\(150\)\1\>'
numbers.txt
21502

$> grep '\(2\)\(150\)\2\>'
numbers.txt
2150150

$> grep '\(2\)\|\(150\)\1\>'
numbers.txt
1500012
21500
21502
2150150
```

Best practice

```
$> ls
a.txt      aa.txt    aaa.txt  exp4.sh

$> find . -name 'a*'
./a.txt
./aa.txt
./aaa.txt

$> find . -regextype grep -regex 'a.*’

$> find . -regextype grep -regex './a.*'
./a.txt
./aa.txt
./aaa.txt

$> find . -regextype grep -regex './aa\?\.txt'
./a.txt
./aa.txt

$> find . -regextype grep -regex './a\{1,2\}\.txt'
./a.txt
./aa.txt
```