Bash Script

CIRC Summer School 2015

Baowei Liu

Filename Expansion / Globbing

- Expanding filenames containing special characters
- Wild cards * ?, not include . ..
- Square brackets [set]: “-”
- Special characters: ! (other than)
- Quote special pattern character if they are to be matched literally
- Escaping backslash: protect a subsequent special character

File Manipulation

- Examine the status of a file

  -a file: True if file exists

  -s file: True if file exists and has a size greater than zero

  -f file: True if file exists and is a regular file

- Compare files

  file1 –nt file2: newer than

  file1 –ot file2: older than

Merge files

- join: merge files by a common column
- cat: merge files by rows

Arrays

- Array is a numbered list
- One-dimensional only
- Create an array with = and (), or declare –a
- Array element: ArrayName[index]
- Access elements: ${ArrayName[n]} @, *
- Array size:${#ArrayName([@])},${#ArrayName([*])}
- Initialize an array with brace expansion
- Delete array or element: unset ArrayName[n]
- Add element without key: ArrayName+=(...)

Strings and Manipulation

- Create a string
- Display a string
- Length of a string
- Substring: a Bash string just holds one element
- Compare strings:
- Concatenate of string
- Substring extraction: position starting with 0
- Substring replacement

Compare Strings

- =: [[“$a” = “$b”]], white space are important!!
- !=
- -z True if the string is null /zero-length
- -n True if the string is Not null

Substring Extraction

- ${string:position:length}
- ${string:position}

Substring Removal

- ${string#substring}
- ${string##substring}
- ${string%substring}
- ${string%%substring}

Substring Replacement

- ${string/substring/replacement}
- ${string//substring/replacement}
- ${string/#substring/replacement}
- ${string/%substring/replacement}

grep and Regular Expression

- grep: search for matches to a pattern in a file and print the matched line to stdout

  grep PATTERN file

- Regular Expression: a sequence of characters that define a search pattern, mainly for string match -- globbing pattern used for text

Regular Expression

- . : Equivalent to ? in filename expansion
- .*: any string. Equivalent to * in filename expansion
- * : zero or more times, a* will match a,aa,… but not ab
- ^: starting with, ^ab
- $: ending with, ab$

Regular Expression

- []: “[-]” “[^]”
- “\< >\” exact word

sed and Regular Expressions

- sed ‘s/abc/xyz’ File: All occurrences
- sed ‘5,10s/abc/xyz’ File: specified lines
- sed ‘0~2 s/abc/xyz/’ File: only in the even lines
- More complicated examples

sed and Regular Expressions

- Word Characters: Alphanumeric characters plus “_” [A-Za-z0-9_]
- Replace all occurrences in a line

awk

- A text-processing programming language in Linux
- awk ‘{print $1}’
- Floating number calculations

head and tail

- –n
- –c
- -f

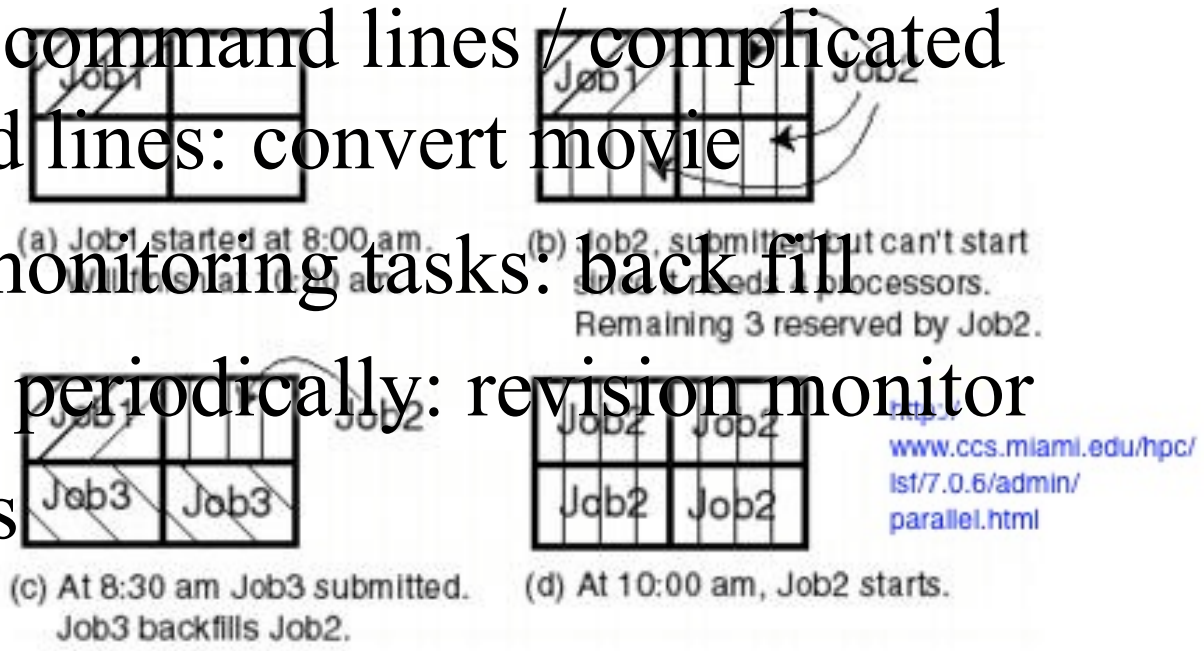
wc

- wc: print the number of bytes, words and lines in a file.
- -c
- -l
- -w

Some Examples

Scenarios & Examples Using Bash Script

- Multiple command lines / complicated command lines: convert movie
- System monitoring tasks: back fill
- Run jobs periodically: revision monitor
- Wrappers



Job Scheduler Slurm

- Slurm

  1. Free and open-source job scheduler
  2. Arbitrate resources by managing a queue of pending jobs
  3. Examples for submitting jobs to our local systems can be found on info.circ.rochester.edu

http://en.wikipedia.org/wiki/Slurm_Workload_Manager

Job Scheduler Cron

- Time-based job scheduler
- Schedule the command to run with crontab –e
- Each line of a crontab file represents a job

Cron and Crontab

- Specify the time:

  * * * * * script/command

  min hr dom m dow(0-6)

- Specify every five hour

  * */5 * * * script/command

  0 0 1 1,6 * script/command

- Non standard macros

  @yearly @reboot …

Stdin, stdout and stderr

- Stdin: standard in, data stream that is going into a process
- Stdout: data stream coming from a running process
- Stderr: data stream of error messages being generated by a process

Redirect and Pipes

- Redirect between files including the three file descriptors, stdin, stdout and stder: > >>
- Pipes takes the output of one command as the input of another command |