538 Text processing basics

Jianguo Lu, University of Windsor

September 12, 2018

Table of contents

View large files

- There are very large files

```
jlu@s2 ~/scholar/Papers $ ls -lt
total 26777184
-rwxr-xr-x 1 jlu acadperm 27419818455 Jan 24 10:03 Papers.txt
-rwxr-xr-x 1 jlu acadperm        9641 Jan 24 09:50 license.txt
```

- Look at the first screen of the file:

```
$ more Papers.txt
```

```
5A32C194        Second Order Conditioning in the Sub-cortical Nuclei of the Limbic System
   second order conditioning in the sub cortical nuclei of the limbic system       2008    2008/07/
07      10.1007/978-3-540-69134-1_19    Simulation of Adaptive Behavior sab             42B9FC1C        19596
```

- look at the last a few lines

```
jlu@s2 ~/scholar/Papers $ tail Papers.txt
```

- Count number of lines and words:

```
jlu@s2 ~/mas $ wc Papers.txt
126909021  3491831946 29292327489 Papers.txt
```

Get colums

- Get all the titles
  - Get the second column

```
cut -f2 Papers.txt > titles
```

- Get title and year
  - Get the second and 4th columns

```
cut -f2,4 Papers.txt > titleYear
```

- -f: field list
- -d: delimiters

transform to lower cases:

```
$ tr 'A-Z' 'a-z' <sigmod.txt | head -2
continuous outlier detection in data streams: an extensi
a query answering system for data with evolution relatio
```

Tools

- grep: search for a pattern (regular expression)
- sort
- uniq c (count duplicates)
- tr (translate characters)
- wc (word or line count)
- sed (edit string – replacement)
- cat (send file(s) in stream)
- echo (send text in stream)
- cut (columns in tab-separated files)
- paste (paste columns)
- head, tail, rev (reverse lines), comm , join
- shuf (shuffle lines of text)

Unix

Get access to a unix/linux/OSX system:

- Option 1: ssh

```
$ ssh jlu@cs.uwindsor.ca
```

- Option 2: if you are using a windows machine, you can install cygwin
- Use man (manual) command to see the explanation e.g.,

```
man tr
```

Exercise 1: Count words in a text

- Input: text file
- Output: list of words in the file with freq counts
- Algorithm
  - Tokenize(tr)
  - Sort (sort)
  - Count duplicates (uniq -c)

Tokenize

```
$  more hoare.txt
There are two ways of constructing a software design.
One way is to make it so simple that there are obviously no
deficiencies. And the other way is to make it so complicated
 that there are no obvious deficiencies

$ tr -sc 'A-Za-z' '\n' < hoare.txt
There
are
two
ways
of
constructing
... ...
```

First try

```
$  tr -sc 'A-Za-z' '\n' < sigmod.txt | sort | uniq -c  | head
 341 A
   1 ABS
   1 ACDN
  65 ACM
   1 ACTA
   1 ADE
   2 ADO
   1 AGILE
   1 AI
   1 AIDE
```

Why all are in uppercases?

Sort ignore cases

```
$  tr -sc 'A-Za-z' '\n' < sigmod.txt | sort -f | uniq -c  |head
 341 A
 698 a
   1 Aalborg
   1 aAqua
   1 Abe
   1 ability
   1 Abiteboul
   3 About
   9 about
   1 ABS
```

sort -f: ignore cases

sort -r: reverse order

Sort reverse order

```
$  tr -sc 'A-Za-z' '\n' < sigmod.txt | sort -r | uniq -c  | head
   1 zsu
   1 zsoyo
   1 youtopia
   1 yourself
  12 your
   1 young
  13 you
   5 yet
   1 years
   2 year
```

How to find the most common words in SIGMOD?

Counting and sorting exercises

- Find the most common words in SIGMOD
- Hint: Use sort a second time, then head

```
$tr -sc 'A-Za-z' '\n'<sigmod.txt |sort|uniq -c |sort -r|head -5
1164 for
 986 of
 938 and
 824 in
 777 data
```

Example 2: Counting Bigrams

- Bigrams = word pairs and their counts
- Useful for text analysis, e.g., in text classification.
- Algorithm:
  - tokenize by word
  - print $word_i$ and $word_{i+1}$ on the same line
  - count

```
Continuous outlier detection in data streams

Continuous      outlier
outlier         detection
detection       in
in              data
data            streams
```

Bigrams using Unix Commands

```
$ tr -sc 'A-Za-z' '\n' < sigmod.txt > sigmod.words
$ tail -n +2  sigmod.words > sigmod.nextwords
$ paste sigmod.words sigmod.nextwords > sigmod.bigrams
$ head -5 sigmod.bigrams

Continuous      outlier
outlier detection
detection       in
in      data
data    streams
```

Bigrams using Unix Commands

```
$ tr -sc 'A-Za-z' '\n' < sigmod.txt > sigmod.words
$ tail -n +2  sigmod.words > sigmod.nextwords
$ paste sigmod.words sigmod.nextwords > sigmod.bigrams
$ head -5 sigmod.bigrams

Continuous      outlier
outlier detection
detection       in
in      data
data    streams
```

tail -n 2: last two lines
tail -n +2: tail from line two onwards.

Exercises

Find the 10 most common bigrams

Find the 10 most common trigrams

```
$ sort   sigmod.bigrams |uniq -c |sort -r | head

 128 of          the
  89 in          a
  79 system      for
  73 in          the
  72 Proceedings of
  70 of          data
  62 database    systems
  58 query       processing
  56 the         ACM
  55 ACM         SIGMOD
```

grep

- Grep finds patterns specified as regular expressions
- globally search for regular expression and print

```
$ grep 'sigmod' sigmod.txt
[EMPTY]
```

Finding titles containing 'SIGMOD':

```
$grep -i 'sigmod' sigmod.txt |head
Proceedings of the 1996 ACM SIGMOD international conference on Manage
Proceedings of the 8th ACM SIGMOD workshop on Research issues in data
Proceedings of the ACM SIGMOD International Conference on Management
Proceedings of the 1976 ACM SIGMOD international conference on Manage
Proceedings of the 9th ACM SIGMOD workshop on Research issues in data
Proceedings of the Fourth SIGMOD PhD Workshop on Innovative Database
Proceedings of the ACM SIGMOD International Conference on Management
Proceedings of the 2nd SIGMOD PhD workshop on Innovative database rese
Proceedings of the 1990 ACM SIGMOD international conference on Manage
Proceedings of the 1981 ACM SIGMOD international conference on Manage

$grep -i 'sigmod' sigmod.txt | wc
      82    1006    7022
```

grep

- grep is a filter: you keep only some lines of the input
- grep 'sigmod': keep lines containing 'sigmod'
- grep '^ sigmod': lines beginning with 'sigmod'
- grep 'sigmod$': lines ending with 'sigmod'

```
$ grep -i '^sigmod' sigmod.txt |head -5
SIGMOD Contributions Award Talk
SIGMOD 10-year Test-of-Time Award: Integration of het
SIGMOD 2013 new researcher symposium
SIGMOD Jim Gray Doctoral Dissertation Award Talk
SIGMOD Jim Gray Doctoral Dissertation Award Talk
```

Join two files

join: joins two **sorted** text files based on the presence of a common field

```
join -1 2 -2 2 sigmod.freq icse.freq
of      986 4111
the     564 2386
on      395 1923
to      255 917
with    226 583
```

```
$ paste sigmod.freq icse.freq|head
1164 for        4111 of
 986 of         2927 for
 938 and        2763 software
 824 in         2551 and
 777 data       2507 a
 698 a          2386 the
 564 the        2007 in
 438 database   1923 on
 395 on         1557 based
 341 A          1003 engineering
```

Why 'for' is missing?

Join two files

```
$join -1 2 -2 2 <(sort -k 2 sigmod.freq) <(sort -k 2 icse.freq)
|sort -r -k 2|head

object          99 165
of              986 4111
distributed     97 237
and             938 2551
optimization    93 103
approach        91 415
over            90 38
```

Sort in alphabetical order by default. sort by number: -n

```
$join -1 2 -2 2<(sort -k 2 sigmod.freq)<(sort -k 2 icse.freq)
 |sort -rn -k 2|head
for      1164 2927
of       986 4111
and      938 2551
in       824 2007
data     777 371
a        698 2507
the      564 2386
database 438 57
```

shuf

- Randomly permutes (shuffles) the lines of a file
- Exercises
  - Print 10 random word tokens from sigmod.txt
  - Print 10 random word types from sigmod.txt

References

- http://web.stanford.edu/class/cs124/kwc-unix-for-poets.pdf

Exercises

- How many all uppercase words are there in this sigmod.txt file?
- how many types?
- how many tokens?
- How many 4-letter words?
- How many different words are there with no vowels

Type/token distinction: different words (types) vs. instances (tokens)