Introduction to Computational Linguistics

Section

Olga Zamaraeva

University of Washington

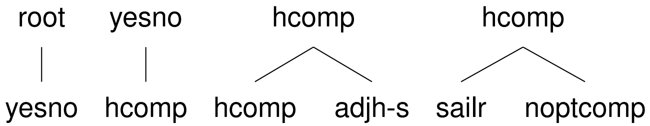
May 15, 2020

Plan for today: Preparing for Assignment 5

- Making sense of the format
- Programming concepts
- Demo

Assignment 5 files format

```
      => yesno
yesno => hcomp
hcomp => hcomp adjh_s
hcomp => sailr noptcomp
```



Assignment 5: What to count and how?

```
      => yesno
yesno => hcomp
hcomp => hcomp adjh_s
hcomp => sailr noptcomp
```

- “The rewrite possibilities for each of the following symbols...”
  - Means: what are all possible RHS for each given LHS?
  - E.g. `root` can only be “rewritten” as `yesno` (in this small fragment; not so in the real assignment!)
- “...and the relative frequency (= probability estimate) for each of the possibilities”
  - Means: For each RHS, how does its count relate to the total count of the respective LHS?

Relative frequencies

```
1        => yesno
1  yesno => hcomp
1  hcomp => hcomp adjh_s
1  hcomp => sailr noptcomp
```

- Means: For each RHS, how does its count relate to the total count of the respective LHS?
- E.g.: there's only 2 occurrences of hcomp in the fragment; P of each has to be 0.5
- What if there were 2 occurrences of hcomp $\rightarrow$ hcomp adjh_s and 1 of hcomp $\rightarrow$ sailr noptcomp?

Unix tools to sort and count things

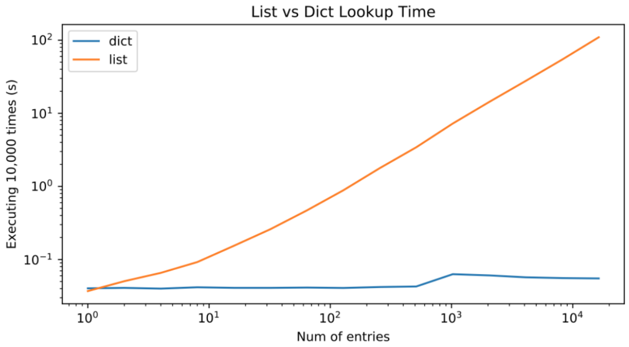
- Pro: Very fast to use when you know them
- Con: Black box! Not very illuminating in terms of programming concepts
- Recommended for: Counting how many times each rule occurs
  - Result: 3 a_det $\rightarrow$ A
  - Meaning: a_det gets “rewritten” as A 3 times in the data
  - First sort the file (and save the sorted version):
    - sort file.txt > sorted-file.txt
  - Then count:
    - uniq -c sorted-file.txt > rule-counts.txt
- Not recommended for: Computing relative frequencies
  - It’s possible but I can’t help you much! Use python :)

Programming concepts

- Reading from file
  - Using the *with open* environment
- Iterating over a list
  - For-loops
- Dict(ionaries)
  - Efficient data structure for fast access (aka hash map)
  - Values are stored under Keys and can be accessed directly by key
  - No need to iterate through the entire structure until you've found what you want

Dict (aka Hash Map)

Efficient data structure for fast access



https://stackoverflow.com/questions/43690191/why-are-dict-lookups-always-better-than-list-lookups