Computing in 571

Programming

- For standalone code, you can use anything you like
  - That runs on the department cluster

- For some exercises, we will use a Python-based toolkit

Department Cluster

- Resources on CLMS wiki
  - http://depts.washington.edu/uwcl
  - Installed corpora, software, etc.

  - patas.ling.washington.edu
  - dryas.ling.washington.edu

- If you don’t have a cluster account, request one ASAP!
  - Link to account request form on wiki
  - https://vervet.ling.washington.edu/db/accountrequest-form.php

Condor

- Distributes software processes to cluster nodes
- All homework will be tested with condor_submit
  - See documentation on CLMS wiki
    - Construction of condor scripts
    - http://depts.washington.edu/uwcl/twiki/bin/view.cgi/Main/HowToUseCondor

NLTK

- Natural Language Toolkit (NLTK)
  - Large, integrated, fairly comprehensive
    - Stemmers
    - Taggers
    - Parsers
    - Semantic analysis
    - Corpus samples, etc
  - Extensively documented
  - Pedagogically oriented
    - Implementations strive for clarity
      - Sometimes at the expense of speed/efficiency

NLTK Information

- http://www.nltk.org
  - Online book
  - Demos of software
  - HOWTOs for specific components
  - API information, etc

Python & NLTK

- NLTK is installed on cluster
  - Use python3.4+ with NLTK
    - **NOTE: This is not the default!!!**
    - May use python2.7, but some differences
- NLTK data is also installed
  - /corpora/nltk/nltk-data
- NLTK is written in Python
  - http://www.python.org; http://docs.python.org
    - Many good online intros, fairly simple

Python & NLTK

- Interactive mode allows experimentation, introspection
  - patas$ python3
  - >>> import nltk
  - >>> dir(nltk)
  - ..... AbstractLazySequence', 'AffixTagger', 'AnnotationTask', 'Assignment', 'BigramAssocMeasures', 'BigramCollocationFinder', 'BigramTagger', 'BinaryMaxentFeatureEncoding',
  - >>> help(nltk.AffixTagger)
  - ……
    - Prints properties, methods, comments,…

Turning in Homework

- Class CollectIt
  - Linked from course webpage
- Homeworks due Tuesday night
  - CollectIt time = Tuesday 23:45
- Should submit as hw#.tar
  - Where # = homework number
  - Tar file contains top-level condor scripts to run

HW #1

- Read in sentences and corresponding grammar
- Use NLTK to parse those sentences
- Goals:
  - Set up software environment for course
  - Gain basic familiarity with NLTK
  - Work with parsers and CFGs

HW #1

- Useful tools:
  - Loading data:
    - nltk.data.load**(resource_url**)
      - Reads in and processes formatted cfg/fcfg/treebank/etc
        - Returns a grammar from cfg
      - E.g. nltk.data.load("grammars/sample_grammars/toy.cfg")
        - Load nltk built-in grammar
      - nltk.data.load("file://"+path_to_my_grammar_file)
        - Load my grammar file from specified path

  - Tokenization:
    - nltk.word_tokenize(mystring)
      - Returns array of tokens in string

HW #1

- Useful tools:
  - Parsing:
    - parser = nltk.parse.EarleyChartParser(grammar)
      - Returns parser based on the grammar

    - parser.parse(token_list)
      - Returns iterable list of parses
      - for item in parser.parse(tokens):
        - print(item)
      - (S (NP (Det the) (N dog)) (VP (V chased) (NP (Det the) (N cat))))