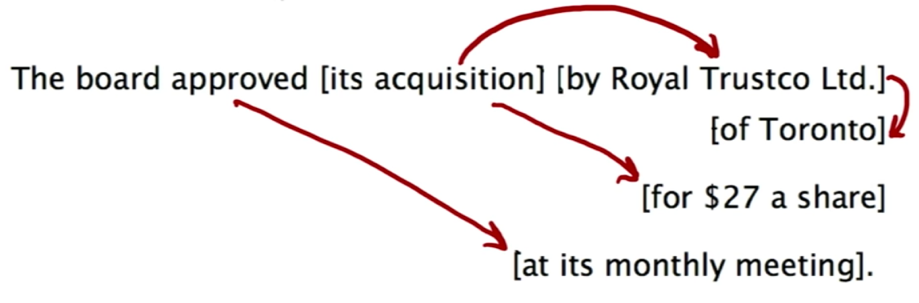
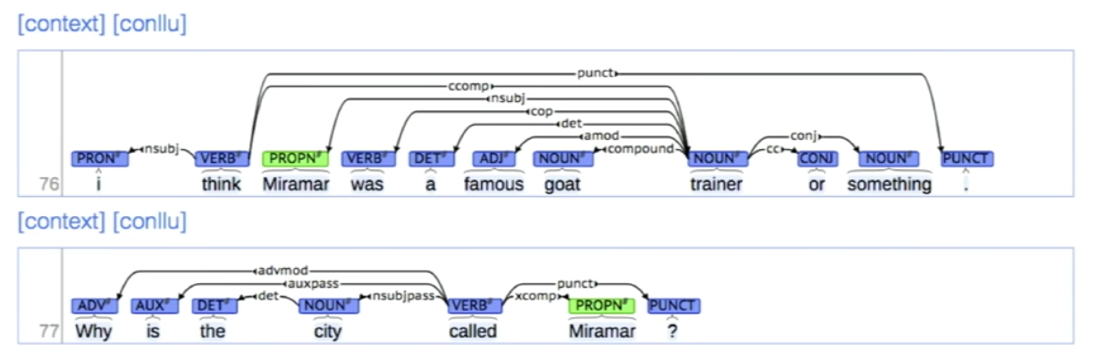
Lecture 6 | Dependency Parsing

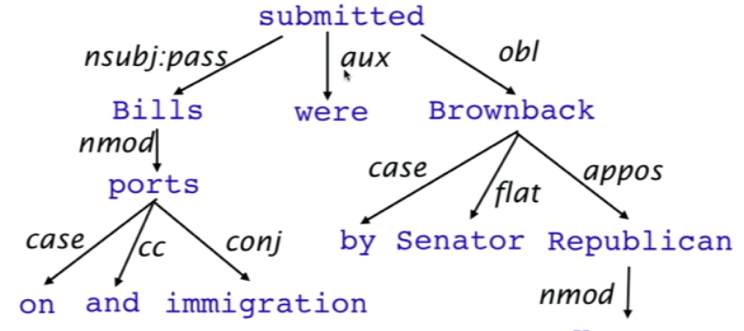
* Two views of linguistic structure
  + Constituency = phrase structure grammar = **context-free grammars**
    - Phrase structure organises words into nested constituents
  + **Dependency structure**
    - Shows which words depend on (modify or are arguments of) which other words
* A key parsing decision is how we attach various constituents:



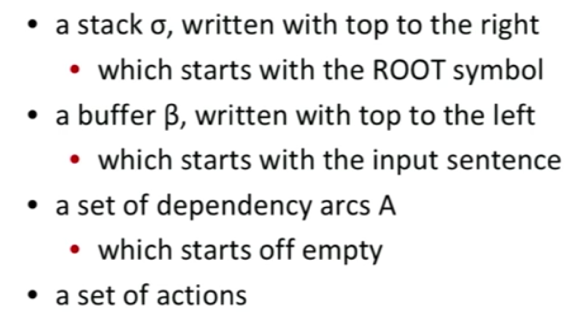
* The rise of annotated data in the form of treebanks

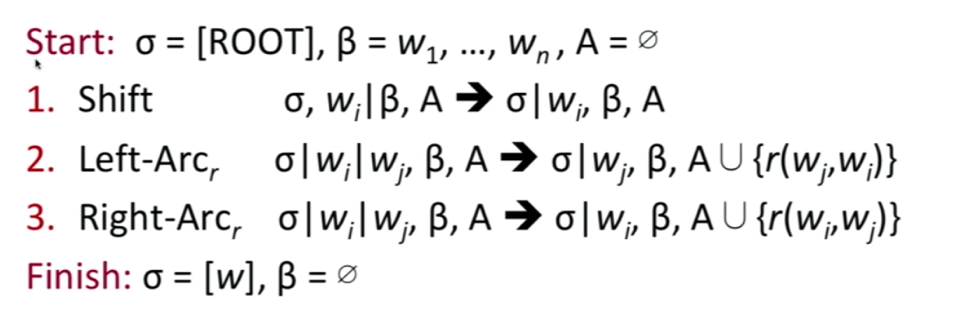


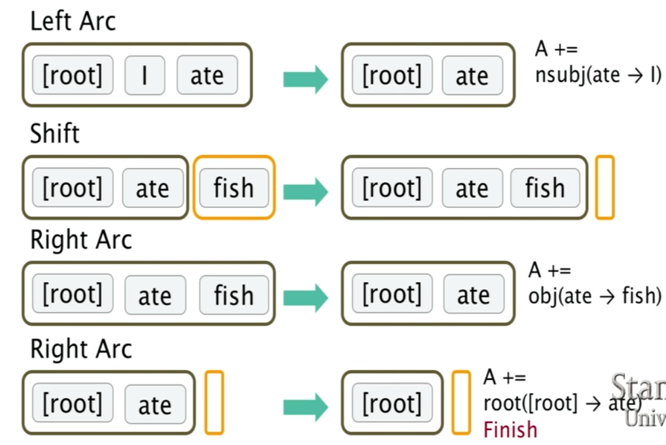
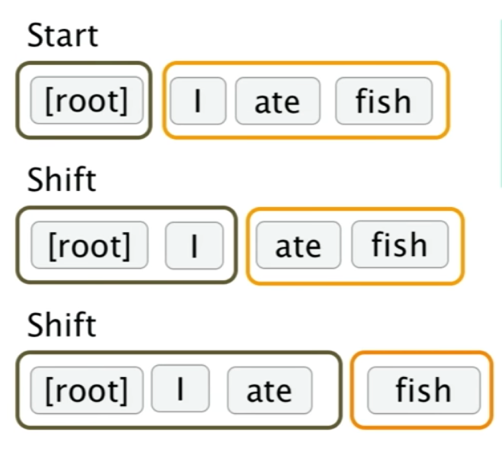
* + At the start, building a treebank seems to be a lot slower and less useful than building a grammar (generalised rule for lots of sentences)
  + But a treebank has advantages:
    - Reusability of the labour
      * Many parsers, POS taggers can be built on it
      * Valuable resource for linguistics
    - Broad coverage (not just a few intuitions)
    - Frequencies and distributional information
    - A method of evaluating systems
* What is dependency grammar and dependency structure?
  + Dependency syntax postulates that syntactic structure consists of relations between lexical items (words), normally binary asymmetric relations (arrows) called dependencies:



* + The arrows are commonly labelled with the name of the grammatical relations
  + Sources of information for dependency parsing
    - Bilexical affinities
      * Discussion 🡪 issues
    - Dependency distance
      * Mostly nearby words
    - Intervening material
      * Dependencies rarely span intervening verbs or punctuation
    - Valency of heads
      * How many dependents on which side are usual for a head?
  + How do we do dependency parsing?
    - A sentence is parsed by choosing for each word what other word (including ROOT) is it a dependent of
    - Some constraints:
      * Only one word is dependent of ROOT
      * Don’t want cycles A 🡪 B and B 🡪 A
* Methods of dependency parsing
  + Dynamic programming
    - Producing parse items with heads at the end rather than in the middle
  + Graph algorithms
    - Create a Minimum Spanning Tree for a sentence
  + Constraint satisfaction
  + **“Transition-based parsing”** or **“deterministic dependency parsing”**
    - Greedy choice of attachments guided by good machine learning classifiers
* Greedy transition-based parsing
  + A simple form of greedy discriminative dependency parser
  + The parser does a sequence of bottom up actions, roughly like ‘shift’ or ‘reduce’ in a shift-reduce parser but the ‘reduce’ actions are specialised to create dependencies with head on left or right
  + The parser has:

****

****

****

* + How do we choose which action to use?
    - Each action is predicted by a discriminative classifier over each legal move
    - Maximum of 3 untyped choices
    - Features would be top of stack word, POS
  + Evaluation of dependency parsing
    - Unlabeled accuracy score
    - Labeled accuracy score