Probabilistic CCG Parsing

Natalie Parde
UIC CS 421

Combinatory Categorial Grammars (CCGs)

- *Heavily* lexicalized approach that groups words into categories and defines ways that those categories may be combined
- Three major parts:
  - Categories
  - Lexicon
  - Rules

CCG Categories

- **Atomic elements**
  - $\mathcal{A} \subseteq \mathcal{C}$, where $\mathcal{A}$ is a set of atomic elements, and $\mathcal{C}$ is the set of categories for the grammar
  - Sentences and noun phrases
- **Single-argument functions**
  - (X/Y), (X\Y) $\in \mathcal{C}$, if $X, Y \in \mathcal{C}$
    - (X/Y): Seeks a constituent of type Y to the right, and returns X
    - (X\Y): Seeks a constituent of type Y to the left, and returns X
  - Verb phrases, more complex noun phrases, etc.

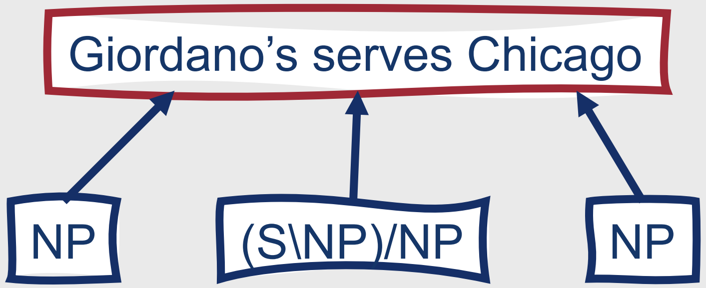
CCG Lexicon

- Assigns CCG categories to words
    - Chicago: NP
        - Atomic category
    - cancel: (S\NP)/NP
        - Functional category
        - Seeks an NP to the right, returning (S\NP), which seeks an NP to the left, returning S

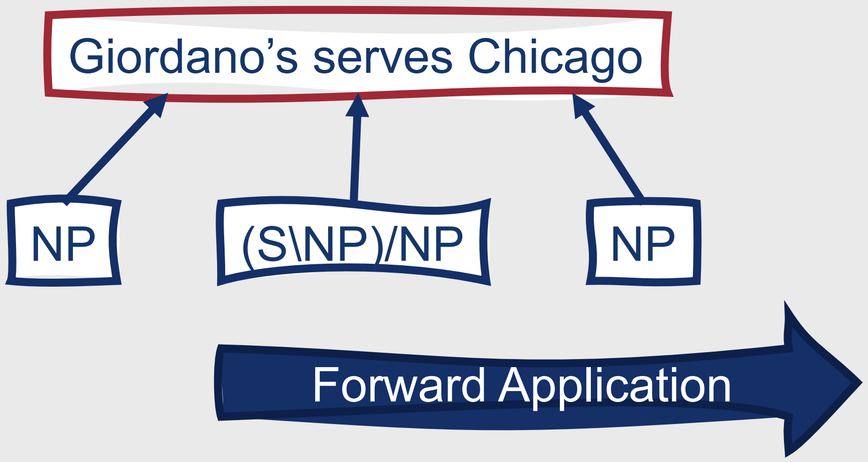
CCG Rules

- Specify how functions and their arguments may be combined
- **Forward function application:** Applies the function to its argument on the right, resulting in the specified category
  - $X/Y \; Y \Rightarrow X$
- **Backward function application:** Applies the function to its argument on the left, resulting in the specified category
  - $Y \; X\backslash Y \Rightarrow X$
- A coordination rule can also be applied
  - $X \; CONJ \; X \Rightarrow X$

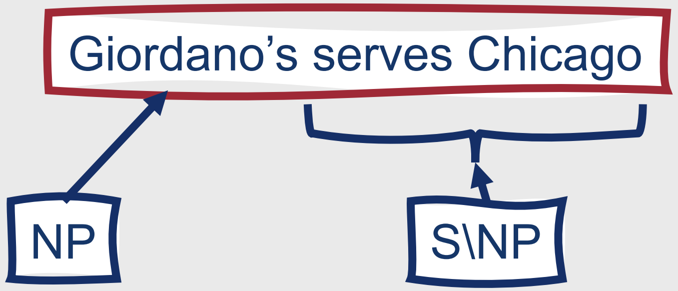
CCG Rules: Example



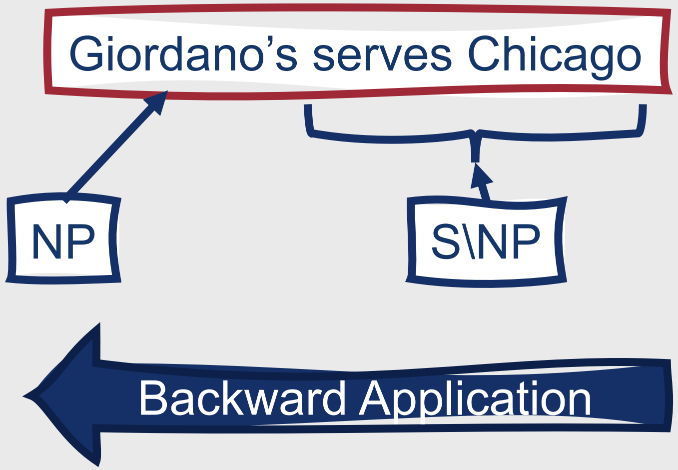
CCG Rules: Example



CCG Rules: Example



CCG Rules: Example



CCG Rules: Example

$$\underbrace{\boxed{\text{Giordano's serves Chicago}}}_{\boxed{S}}$$

CCG Operations

- **Forward composition**
  - Can be applied when, given two functions, the first seeks a constituent of type Y to the right and the second provides a constituent of type Y as its result
    - $X/Y \ Y/Z \Rightarrow X/Z$
- **Backward composition**
  - Can be applied when, given two functions, the first seeks a constituent of type Y to the left and the second provides a constituent of type Y as its result
    - $Y\backslash Z \ X\backslash Y \Rightarrow X\backslash Z$

CCG Operations

- **Type raising**
  - Converts atomic categories to functional categories, or simple functional categories to more complex functional categories
    - $X \Rightarrow T/(T\backslash X)$, where T can be any existing atomic or functional category
    - $X \Rightarrow T\backslash(T/X)$
  - Facilitates the creation of intermediate elements that do not directly map to traditional constituents in the language
- Type raising and function composition can be employed together to parse **long-range dependencies**

CCGBank

- Largest and most popular CCG treebank
- Based on the Penn Treebank
- 44,000-word lexicon with 1200+ categories

Ambiguity in CCGs

- CCG lexicons allow words to be associated with numerous categories, depending on how they interact with other words in the sentence
- This can create ambiguity when parsing!

CCG Parsing Frameworks

- Probabilistic CKY
  - Okay, but needs to be adapted a bit due to the large number of categories available for each word (otherwise, lots of unnecessary constituents would be added to the table)
  - The solution: **Supertagging**
- Supertags are also used in other CCG parsing frameworks

Supertagging

- Trained using CCG treebanks (e.g., CCGBank)
- Predict allowable category assignments (supertags) for each word in a lexicon, given an input context
- Commonly framed as a supervised sequence labeling problem

After extracting supertags, probabilistic CKY can be employed as a CCG parser.

- Another popular CCG parsing technique: **A* Algorithm**
- **A*:** Heuristic search algorithm that finds the lowest-cost path to an end state, by exploring the lowest-cost partial solution at each iteration until a full solution is identified
- Search states = edges representing completed constituents
- Cost is based on the probability of the CCG derivation
- A* results in fewer unnecessary constituents being explored than probabilistic CKY