Introduction to Computational Linguistics

Tokenization and Sentence Boundary Detection II

Jan-Philipp Söhn

jp.soehn@uni-tuebingen.de

December 19th, 2007

Incremental Linguistic Analysis

- tokenization
- morphological analysis (lemmatization)
- part-of-speech tagging
- named-entity recognition
- partial chunk parsing
- full syntactic parsing
- semantic and discourse processing

Tokenization: Just do it!

Caipirinha is the national drink of Brazil. The drink is made with cachaça (pronounced - kuh-sha-suh). It is technically a brandy and a member of the aguardente family. **There is an old adage in Brazil: "quanto pior a cachaça, melhor a caipirinha" – the worse the cachaça, the better the caipirinha. The name caipirinha is derived from the Portuguese word caipira (hick, hayseed, country bumpkin, rube, etc.) coupled with the -inha suffix** (a diminutive denoting little or small) and can be roughly translated as little hick, little hayseed, little country bumpkin, little rube, etc., etc. Poor man's drink or not, cachaça has become an integral part of Brazilian culture and its significance ranks right up there with soccer/football (futebol), carnival and samba as Brazilian national icons. The Brazilians like it sweet.... VERY sweet. **In Brazil, the very best caipirinhas are made with "limões gallegos" – what in the U.S. is often referred to as a key lime.** Can also be made by the pitcher!

Potential Tasks

- Tokenize arbitrary text
- Subtask: Recognize date expressions
- Given an inflected verb: Find a base form of verbs and their agreement features
- Given a base form of verbs and their agreement features: find the appropriate inflected form
- Morphology: derivation: English verbs + suffix *-able* (yields an adjective: desirable, printable, readable, etc.)
- Assign syntactic categories to tokens in preprocessed text
- Bracketing of syntactic chunks in arbitrary text

Formal Languages & Computation

The language perspective

1. Type 3: regular expression languages
2. Type 2: context free languages
3. Type 1: context sensitive languages
4. Type 0: recursively enumerable languages

Formal Languages & Computation

The language perspective

1. Type 3: regular expression languages
2. Type 2: context free languages
3. Type 1: context sensitive languages
4. Type 0: recursively enumerable languages

The automata perspective

1. Finite automata
2. Pushdown automata
3. Linear automata (Turing machines with finite tapes)
4. Turing machines

Form of Grammars of Type 0–3

For $i \in \{0, 1, 2, 3\}$, a grammar $\langle N, T, P, S \rangle$ of Type i, with N the set of non-terminal symbols, T the set of terminal symbols (N and T disjoint, $\Sigma = N \cup T$), P the set of productions, and S the start symbol ($S \in N$), obeys the following restrictions:

T3: Every production in P is of the form $A \longrightarrow aB$ or $A \longrightarrow \epsilon$, with $B, A \in N$, $a \in T$.

T2: Every production in P is of the form $A \longrightarrow x$, with $A \in N$ and $x \in \Sigma^*$.

T1: Every production in P is of the form $x_1 A x_2 \longrightarrow x_1 y x_2$, with $x_1, x_2 \in \Sigma^*$, $y \in \Sigma^+$, $A \in N$ and the possible exception of $C \longrightarrow \epsilon$ in case C does not occur on the righthand side of a rule in P.

T0: No restrictions.

An Example of a Type 2 Grammar

Let $\langle N, T, P, S \rangle$ be a grammar with N, T and P as given below:

- $N = \{S, NP, VP, V\}$
- $T = \{\text{John}, \text{walks}\}$
- $P = \{S \rightarrow NP\ VP, NP \rightarrow \text{John}, VP \rightarrow V, V \rightarrow \text{walks}\}$