Grammars

10-31-2011

Opening Discussion

- Minute essay comments:
  - Resubmitting assignments.
  - References for Chomsky grammars?
- IcP Solutions
- What did we talk about last class?

Context-Sensitive Grammars

- Takes surrounding characters into account:
  - $\alpha A\beta \rightarrow \alpha\gamma\beta$
- Equivalent to a linear bounded non-determinstic Turing machine.
- Not used all that much because of challenges. Needed for some elements of natural language.

Recursively Enumerable Grammars

- Allows basically any transformation.
  - $\alpha \rightarrow \beta$
- There are no bounds on what these can be.
- This is equivalent to a Turing machine. That means that you could calculate anything you want using one of these.

Regular Expressions

- One if the applications of these formal systems is the use of regular expressions to perform String operations.
- Scala has a class called scala.util.matching.Regex. You can get one of these by calling the r method on a String.
- This wraps the functionality of java.util.regex.Pattern and provides Scala style functionality and pattern matching.
- Let's look at API entries.

Details of RegEx

- findAllIn gives back a MatchIterator. It is an Iterator[String]. Call matchData to get an Iterator[Match].
- The Match class has lots of data about each match including subgroups.

For Loops and RegEx

- Remember that for-loops do pattern matches for storing values. They also skip anything that doesn't match the pattern.
- This makes them ideal when running through the results of findAllIn.

Examples of RegEx

- Let's run through some different examples of using regular expressions.
  - Decimal numbers
  - Points in 2-D or 3-D
  - Dates
  - Polynomials

Minute Essay

- Any thoughts on regular expressions?
- What do you see as limits of regular expressions?