Programming Languages (CS302 2007S)

Class 34: Declarative Langauges

Held: Monday, April 23, 2007

Summary: Today we begin our exploration of declarative languages.

Related Pages:

- EBoard.

Notes:

- When does each group want to present?
- EC for attending today's Quantitative Sociology talk.

Overview:

- Language Paradigms, Revisited.
- Abstraction, Revisited.
- Common Categories of Declarative Languages.
- Predicate Logic Languages.
- Regular Expression Languages.
- Database Languages.

Language Paradigms, Revisited

- As you know (and have learned), there are a number of key language paradigms that reflect the primary techniques of the language.
  - In *Imperative* languages, we think of algorithms and programs as sequences of instructions.
  - In *Object-Oriented* languages, we think of algorithms and programs as collections of cooperating objects.
  - In *Functional* languages, we think of algorithms and programs in terms of the definition and application of functions.
- Of course, each paradigm also includes a number of related ideas that we tend to expect.
  - The basic instructions in imperative languages focus on moving data.
  - Object oriented languages also support encapsulation, inheritance, and polymorphism
  - Functional languages tend to support a symbol data type, higher-order programming, and garbage collection.
- This week, we will consider another paradigm, the *Declarative* languages
  - A *Declcarative Language* supports programs in which the focus of algorithm design is on specifying *what* we want to compute, rather than *how* we compute it.

Abstraction, Revisited

- In many ways, declarative languages are a natural extension of the increasing emphasis on abstraction we have seen in a number of languages.
  - That is, declarative languages abstract away not just data representation and some simple things, but also control.
- As with other instances of abstraction, there are positives and negatives with this kind of abstraction.

Categories of Declarative Languages

- While the category of declarative languages is regularly cited, there do not seem to be that many declarative languages (and even fewer purely declarative languages) .
  - Different people often categorize different languages as declarative.
- Nonetheless, there are a few key subcategories of declarative languages.
- Many people consider *pure functional languages* as declarative. Why? Because, in such languages, it can be up to the implementation what order to do things. (In fact, one can even apply a function before evaluating the parameters to the function.)
  - Formally, pure functional languages represent programs written in equational logic.
  - Detour: My advisor wrote a language based on this idea.
- The *predicate logic languages* form one particularly popular branch of declarative languages.
  - Prolog is the prototypical predicate logic language.
- Many languages use *regular expressions* for a declarative sublanguage.
  - A regular expression describes something to match, but not how to match it.
- *Database languages*, most typially SQL, are sometimes cited as declarative languages.

Predicate Logic Languages

- Key ideas:
  - Write statements in predicate logic
  - Ask questions
  - The system uses the statements to compute the answer.
- For example,
  - X is the sorted version of Y if X is a permutation of Y and X is permuted.
  - More realistically, X is the sorted version of Y if X was created by repeatedly inserting each element of Y into the correct place into an empty list.
    - M is the result of inserting X into L if M is (cons X L) and X is less than the first element of L
    - M is the result of inserting X into L if M is (cons Y N), Y is the first element of L, and N is the result of inserting X into the cdr of L.
- Expressed most clearly in Prolog.

Regular Expression Languages

- Grep
- AWK
- Lots of libraries.

Database Languages

- Provide an overview of the ways to access the data.
- Underlying structure hidden from the user.
- ...