Logic Programming

*Introduction*

Michael Genesereth
Computer Science Department
Stanford University

Lecture will begin at ~1:39 PDT.
Programmed Computer System

Inputs → Interpreter / Compiler → Outputs

Data Structures

- Interpreters
- Compilers
- Data Structures
- Programmed Computer System
Specifications versus Programs

Definitions
Assumptions
Goals
Implicit Specification

Definitions
Assumptions
Goals

Inputs
Interpreter / Compiler
Outputs

Database
Data Structures

Definitions
Assumptions
Goals

Inputs
Interpreter / Compiler
Outputs

Database
Data Structures
A logic program is effectively a **runnable specification**.
Logic as a Specification Language

Language of Logic

Domain Independent
+
Highly expressive

Logic Interpreters / Compilers
Automated Reasoners capable of drawing conclusions
Can take advantage of domain-dependent reasoners
but are also capable domain-independent reasoning
Types of Logic Programming

- Database Programming (Datalog)
- Classical Logic Programming (Prolog)
- Dynamic Logic Programming (Epilog)
- Constraint Programming
- Answer Set Programming
- Inductive Logic Programming (Progol)
- Automatic Theorem Proving
- Automatic Programming
Why Logic Programming
Benefits

Efficiency
Lots of traditional programmers
Well established software engineering practices

Disadvantages

Creation, **maintenance** expensive and time-consuming
Different programs for different tasks
Difficult to explain results
Programs not comprehensible to ordinary users
Logic Programs are relatively easy to create.

Requires **little work**. The specification is the program; no need to make choices about data structures and algorithms.

Specification authors can get by with **few assumptions** about the capabilities of systems executing those programs.

**Easier to learn** logic programming than traditional programming. Think spreadsheets.

*Oddly, expert computer programmers often have more trouble with logic programming than novices.*
Adaptability

*Easy to deal with changing circumstances*
Sample Program
A person X is the grandparent of a person Z if and only if there is a person Y such that X is the parent of Y and Y is the parent of Z.

Uses
Determine whether Art is the grandparent of Cal.
Determine all of the grandchildren of Art.
Compute the grandparents of Cal.
Compute all grandparent-grandchildren pairs.
McCarthy’s Example
The building is illegal

The shadow line is 262 cm

The allowable shadow is 240 cm

The parcel is in zone R-1

The allowable shadow in R-1 is 240 cm

The building is 462 cm high

The building is 200 cm from the boundary

462 - 200 = 262

262 > 240
Why was my building plan rejected?

*Your plan is illegal because your shadow line (262 cm) exceeds the allowable shadow (240 cm).*

What is my shadow line?

*Your shadow line (262 cm) is the maximum intrusion into the yard of a side neighbor determined by a 45 degree line from the highest point of the building.*

What is the allowable shadow line?

*Your parcel is in zone R-1 and in zone R-1, the maximum shadow that can be cast on a side neighbor is 240 cm.*
Successes
Engineering

Circuit:

Applications:
Simulation
Configuration
Diagnosis
Test Generation

Description:

\[
o \leftrightarrow (x \land \neg y) \lor (\neg x \land y)
\]

\[
a \leftrightarrow z \land o
\]

\[
b \leftrightarrow x \land y
\]

\[
s \leftrightarrow (o \land \neg z) \lor (\neg o \land z)
\]

\[
c \leftrightarrow a \lor b
\]
Deductive Databases

\[ q(X) :\neg p(X,Y) \land p(X,Z) \land Y \neq Z \]

\[ g(X,Z) :\neg p(X,Y) \land p(Y,Z) \]

\[ \text{illegal} :\neg p(X,Y) \land p(Y,X) \]

Questions
Updates

Database Manager

Answers
Notifications

\[ p(a,b) \]
\[ p(b,c) \]
\[ p(a,b) \]
Interactive Web Pages (Worksheets)

Gates Information Network

Program Sheet

DEPARTMENT OF COMPUTER SCIENCE
MSCS Program Sheet (2010-11)

Name: Charles Pumell

Advisor: Primary Specialization

Student ID: Proposed degree for degree conferral: Date: 10/8/2010

GENERAL INSTRUCTIONS

Before the end of your first quarter, you should complete the following steps. Detailed instructions are included in the Guide to the MSCS Program Sheet. An online version is available at cs.stanford.edu/degreetrace/programsheets/.

- Complete this program sheet by filling in the name, number, title of each course you intend to use for your degree.
- Create a course schedule showing the year and quarter in which you intend to take each course in your program sheet.
- Meet with your advisor and secure the necessary signatures on the program sheet.

PROGRAM REQUIREMENTS

You must satisfy the requirements listed in each of the following areas; all courses taken elsewhere must be approved by your adviser and on a foundation course waiver form. Required documents for waiving a course include course descriptions, syllabi, and textbook lists. These documents can be organized here: cs.stanford.edu/degreetrace/programsheets/.

Note: If you are amending an old program sheet, enter "on file" in the approval column for courses that have already been approved.

Required:

- Logic, Automata, and Complexity (5 of CS 103)
- Probability (5 of CS 110, STATS 110, CME 100, or CS 105)
- Algorithmic Analysis (5 of CS 161)
- Computer Organization and Systems (5 of CS 107)

Principles of Computer Systems (5 of CS 110)

TOTAL UNITS USED TO SATISFY FOUNDATIONS REQUIREMENTS: 10

Note: This total may not exceed 15 units.

Change Your Scope
Business Rules and Workflow
**Computational Law** is that branch of legal informatics concerned with the mechanization of legal reasoning.

**Automated Compliance Management**
- Legal analysis of specific cases
- Planning for compliance in specific cases
- Analysis of regulations for overlap, consistency, etc.

*Portico*
General Game Playing
General Game Playing

Pelican Hunters
Non-Successes
Natural Language Processing
PTTP
means
Prolog Technology Theorem Prover
by acronymsandslang.com
Japan’s Fifth Generation Project
History
IBM 360

Figure 4. Card Codes and Graphics for 84-Character Set
Assembly Language (2GL)

Assembly Language

- mov ecx, ebx
- mov esp, edx
- mov edx, r9d
- mov rax, rdx

Assembler + Linker

Machine Language

- 100101011001
- 0100111111011
- 111010101101
- 0101010101010
Symbolic Processing Languages (3GL)
Imperative Programming Languages

Python
Java
JavaScript
C++
Ruby
Scala
Node.js
C#
Declarative Programming Languages

Declarative vs. Imperative
The main advantage we expect the advice taker to have is that its behavior will be improvable merely by making statements to it, telling it about its … environment and what is wanted from it.

- John McCarthy 1958
The potential use of computers by people to accomplish tasks can be “one-dimensionalized” into a spectrum representing the nature of the instruction that must be given the computer to do its job. Call it the **what-to-how spectrum**. At one extreme of the spectrum, the user supplies his intelligence to instruct the machine with precision exactly how to do his job step-by-step. ... At the other end of the spectrum is the user with his real problem. ... He aspires to communicate what he wants done ... without having to lay out in detail all necessary subgoals for adequate performance.

- Ed Feigenbaum 1974
Chris Date (Mr. SQL)
If code is the problem, the only possible answer is to eliminate the coding by building systems directly from their specifications.

- Val Huber, 1997
This course
<table>
<thead>
<tr>
<th>Date</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apr 4</td>
<td>Introduction</td>
</tr>
<tr>
<td></td>
<td>6 Datasets</td>
</tr>
<tr>
<td></td>
<td>11 Queries</td>
</tr>
<tr>
<td></td>
<td>13 Query Examples</td>
</tr>
<tr>
<td></td>
<td>18 Query Evaluation</td>
</tr>
<tr>
<td></td>
<td>20 Query Optimization</td>
</tr>
<tr>
<td></td>
<td>25 View Definitions</td>
</tr>
<tr>
<td></td>
<td>27 View Evaluation</td>
</tr>
<tr>
<td>May 2</td>
<td>Simple Examples</td>
</tr>
<tr>
<td></td>
<td>4 Lists, Sets, Trees</td>
</tr>
<tr>
<td>May 9</td>
<td>Operation Definitions</td>
</tr>
<tr>
<td></td>
<td>11 Model Management</td>
</tr>
<tr>
<td></td>
<td>16 Reactive Worksheets</td>
</tr>
<tr>
<td></td>
<td>18 Semantic Worksheets</td>
</tr>
<tr>
<td></td>
<td>23 Computational Law</td>
</tr>
<tr>
<td></td>
<td>25 Computable Contracts</td>
</tr>
<tr>
<td></td>
<td>30 Past Projects</td>
</tr>
<tr>
<td>Jun 1</td>
<td>Extensions and Advances</td>
</tr>
<tr>
<td></td>
<td>6 Project Reports</td>
</tr>
<tr>
<td></td>
<td>8 Project Reports</td>
</tr>
</tbody>
</table>
Sets

\{a, b, c\} \cup \{b, c, d\} = \{a, b, c, d\}

a \in \{a, b, c\}

\{a, b, c\} \subseteq \{a, b, c, d\}

Functions and Relations

f(a, b) = c

r(a, b, c)
CS 106 or equivalent
Teams

Composition

3 people each (2 or 4 okay with *good* reason)

Names:

- Pansy Division
- The Pumamen
- Team Camembert
- Mighty Bourgeoisie
- Greedy Bastards
- Red Hot Chili Peppers
- /*v*/
- X Æ A-12
- Michael Genesereth
Grades

Numerical Score
  15% for each of Assignments 1, 2, 3, 4
  40% for the Term Project

Reported Grade
  Based on numerical score (see above)
  *No* curve - independent of number of students
  Satisfactory = 70% and above

Extra Credit
  Added to score before determining Reported Grade
  Discretionary
Introduction to Logic Programming

Michael Genesereth, Stanford University
Vinay K. Chaudhri, Stanford University

“This is a book for the 21st century: presenting an elegant and innovative perspective on logic programming. Unlike other texts, it takes datasets as a fundamental notion, thereby bridging the gap between programming languages and knowledge representation languages; and it treats updates on an equal footing with datasets, leading to a sound and practical treatment of action and change.” – Bob Kowalski, Professor Emeritus, Imperial College London

“In a world where Deep Learning and Python are the talk of the day, this book is a remarkable development. It introduces the reader to the fundamentals of traditional Logic Programming and makes clear the benefits of using the technology to create runnable specifications for complex systems.” – Son Cao Tran, Professor in Computer Science, New Mexico State University

“Excellent introduction to the fundamentals of Logic Programming. The book is well-written and well-structured. Concepts are explained clearly and the gradually increasing complexity of exercises makes it so that one can understand easy notions quickly before moving on to more difficult ideas.” – George Younger, student, Stanford University

ABOUT SYNTHESIS
This volume is a printed version of a work that appears in the Synthesis Digital Library of Engineering and Computer Science. Synthesis books provide concise, original presentations of important research and development topics, published quickly, in digital and print formats.