Logic Programming

Introduction

Michael Genesereth
Computer Science Department
Stanford University

Lecture will begin at ~1:35 PDT.
Logic Programming is a style of programming based on Symbolic Logic.

Logic Program is a collection of sentences encoded in the language of Symbolic Logic.

Logic Programming Language is a specific language for writing such programs.

Logic Programming System is a computer system that manages the creation, modification, and execution of logic programs.
Imperative Programming

Inputs → Interpreter → Outputs
A triangle is a polygon with 3 sides.

\[ e = mc^2 \]
Runnable Specifications

Specification
What we believe about the application area
What we want to know or to achieve in application area

With no arbitrary decisions
With no concern for internal processing details

Runnable
Can be directly interpreted
Can be compiled into traditional programs
Runnable Specifications

A logic program is a runnable specification.
A triangle is a polygon with 3 sides.
Logic as a Specification Language

Language

Declarative language
+ Highly expressive

Other declarative languages exist but statements in most of those languages can be translated to logical form.

Interpreter

Automated Reasoners capable of drawing conclusions
Can take advantage of domain-dependent reasoners but are also capable domain-independent reasoning
Benefits
Programming Ease

Easier to create and modify than traditional programs

Programmers can get by with little or no knowledge of the capabilities of systems executing those programs.

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

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

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

*Ability to respond to changing circumstances or goals*
Ability to be used for multiple purposes

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 of Versatility
McCarthy’s Example of Versatility
Successes
Circuit:

Applications:
- Simulation
- Configuration
- Diagnosis
- Test Generation

Premises:

\[ o \iff (x \land \neg y) \lor (\neg x \land y) \]
\[ a \iff z \land o \]
\[ b \iff x \land y \]
\[ s \iff (o \land \neg z) \lor (\neg o \land z) \]
\[ c \iff a \lor b \]
Deductive Databases

\[ q(X) :\text{-} p(X,Y) \land p(X,Z) \land Y!\neq Z \]

\[ g(X,Z) :\text{-} p(X,Y) \land p(Y,Z) \]

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

\[ p(a,b) \]
\[ p(b,c) \]
\[ p(a,b) \]

Questions Updates

Database Manager

Answers Notifications
Data Integration

Integrated Search

Side-by-side Comparison

Integration Engine

Supplier 1
Supplier 2
Supplier 3
Supplier 4
Manufacturer 1
Manufacturer 2
Satisfaction Ratings
Product analysis

Marketplace Data
Constraint Satisfaction

Example: Cryptarithmetic

```
<table>
<thead>
<tr>
<th>5</th>
<th>3</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>9</td>
</tr>
</tbody>
</table>
```

- Variables: $F T U W R O$  $X_1 X_2 X_3$
- Domains: $\{0,1,2,3,4,5,6,7,8,9\}$  $\{0,1\}$
- Constraints: Alldiff $(F, T, U, W, R, O)$
  - $O + O = R + 10 \cdot X_1$
  - $X_1 + W + W = U + 10 \cdot X_2$
  - $X_2 + T + T = O + 10 \cdot X_3$
  - $X_3 = F, T \neq 0, F \neq 0$
Business Rules

ERP SYSTEM

- Financial Management
- Supply Chain Management
- Human Resource Management
- Manufacturing Resource Planning
- Customer Relationship Management

SAP
ORACLE
IBM
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.

http://logicprogramming.stanford.edu/examples/portico/demonstration.html
General Game Playing
General Game Playing

http://logicprogramming.stanford.edu/examples/nineboard/demonstration.html
Non-Successes
Natural Language Processing
Theorem Proving

PTTP
means
Prolog Technology Theorem Prover
by acronymsandslang.com
Japan’s Fifth Generation Project
History
LGP-30
Assembly Language

Programmer

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

Assembler + Linker

Machine Language

Processor

100101011001
010011111011
111010101101
0101010101010
Higher Level Languages
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
Alain Colmerauer and Bob Kowalski
This course
Types of Logic Programming:
- Database Programming
- Classical Logic Programming
- Dynamic Logic Programming
- Constraint Systems
- Answer Set Programming
- Inductive Logic Programming (i.e. Progol)

Languages:
- Datalog
- Prolog
- Epilog
- Golog
- Progol
<table>
<thead>
<tr>
<th>Mar 29</th>
<th>Introduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>Datasets</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Apr 5</th>
<th>Queries</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Examples</td>
</tr>
<tr>
<td>12</td>
<td>Query Evaluation</td>
</tr>
<tr>
<td>14</td>
<td>Query Optimization</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>May 3</th>
<th>Action Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Dynamic Systems</td>
</tr>
<tr>
<td>10</td>
<td>Database Management</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>May 12</th>
<th>Worksheets</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>May 17</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>Applications</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>May 24</th>
<th>Project Reports</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>Project Reports</td>
</tr>
<tr>
<td>31</td>
<td>Project Reports</td>
</tr>
</tbody>
</table>
Mathematical Background

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
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
- XÆA-12
- Michael Genesereth
Grades

**Numerical Score**
- 10% for each of Assignments 1, 2, 3, 4, 5
- 50% 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.
http://cs151.stanford.edu
The following syllabus lists all of the materials of the course. Note that there are interactive exercises at the ends of the chapters in the course textbook. (Click on the exercise numbers to go to the exercise pages.) These exercises are an essential part of the course, and you will benefit from tackling them. Some are easier than others, but you should attempt them all. Do the exercises! Do The Exercises!! DO THE EXERCISES!!!

**Color Code**
- Black - Lecture Slides
- Blue - Readings
- Red - Assignments
- Grey - Comment

**Introduction (Week 1)**
- Lecture 1 - Introduction
- Lecture 2 - Datasets
- Chapter 1 - Introduction
- Chapter 2 - Datasets
- Programs With Common Sense
- Logic Programming
- Assignment 1.1 - Datasets in Sierra
- Assignment 1.2 - Game State
- Assignment 1.3 - Triples
- Project