Differentiable Probabilistic Answer Set Programming
For Neurosymbolic Learning and Reasoning

Renato Lui Geh, Jonas Gonçalves, Igor Cataneo Silveira,
Denis Deratani Mauá, Fabio Gagliardi Cozman
University of São Paulo
{renatolgi,Jonasrlg,igorc,ddm}@ime.usp.br, fgcozman@usp.br

Answer Set Programming
is a powerful declarative language to describe NP-hard combinatorial problems

Example: 3-coloring a 4-node graph

% graph has 4 nodes
node(1). node(2). node(3). node(4).
% and the following edges
edge(1,2). edge(2,3). edge(3,4). edge(1,4). edge(1,3).
% graph is undirected
edge(X,Y) :- edge(Y,X).
% adjacent nodes must be colored differently
:- not conflict(X,Y), edge(X,Y), color(X,C), color(Y,C).
% a node must have at least 1 of 3 colors
color(X,red); color(X,blue); color(X,green) :- node(X).

Stable models:

Probabilistic Answer Set Programming
extends ASP with independent probabilistic choices to encode uncertain knowledge

Example: 2-colorability of random graph

% graph has 3 nodes
node(1). node(2). node(3).
% prob. 0.5 of having an (undirected) edge between two nodes
0.5::edge(X,Y) :- node(X), node(Y), X < Y.
edge(X,Y) :- edge(Y,X).
% conflict if neighbors have same color
color(X,red); color(X,blue); color(X,green) :- node(X).
% a graph is colorable if it has no conflicts
colorable :- not conflict.
% what is the prob. of a random graph being colorable P(c)?
#query colorable.

Probabilities and stable models:

<table>
<thead>
<tr>
<th>p</th>
<th>P(c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1/4</td>
</tr>
<tr>
<td>1</td>
<td>1/4</td>
</tr>
<tr>
<td>2</td>
<td>1/4</td>
</tr>
<tr>
<td>3</td>
<td>1/4</td>
</tr>
<tr>
<td>4</td>
<td>1/4</td>
</tr>
<tr>
<td>5</td>
<td>1/4</td>
</tr>
</tbody>
</table>

Semantics
tells us how to interpret programs both logically and probabilistically

Example: the barber paradox

% barber shaves every villager who does not shave themselves
shaves(X, Y) :- barber(X), villager(Y), not shaves(Y, Y).
villager(a). barber(b). 0.5::villager(b).
% does the barber shave himself?
#query shaves(b, b).

Which semantics does dPASP support?

Logic semantics
- STABLE
- PARTIAL
- L-STABLE
- D-MAP

Probabilistic semantics
- CREDAL
- MAX-ENT

Neural Answer Set Programming
associates probability annotation with the output of neural classifiers

Example: Parsing arithmetic expressions, e.g.
X + Y = f(X) + f(Y) = ?

% neural rule
?::digit(Image, {0..9}) :- data(Image).
% data loaders -- interact with Python code
data(img1) ~ test(@mnist_test), train(@mnist_train).
data(img2) ~ test(@mnist_test), train(@mnist_train).
% prob. answer set program
add(Z) :- digit(I, X), digit(J, Y), Z = X + Y.
subtract(Z) :- digit(I, X), digit(J, Y), Z = X - Y,
multiply(Z) :- digit(I, X), digit(J, Y), Z = X * Y.
% learn the program end-to-end and pass learning parameters
#learn @mnist_test, lr = 1., niter = 5, ..., batch = 1000.
% inference: what is the probability of X + Y = 14 given X = 8?
#query add(11) | digit(img1, 8).

How does dPASP work?

Experiments

How much faster is dPASP on the MNIST Add?