clp(pfd(Y)) : Constraints for Probabilistic Reasoning in Logic Programming

Nicos Angelopoulos
nicos@cs.york.ac.uk

http://www.cs.york.ac.uk/~nicos

Department of Computer Science

University of York

talk structure

- Logic Programming (LP)
- Uncertainty and LP
- Constraint LP
- clp(pfd(Y))
- clp(pfd(c))
  - Caesar's coding experiments
  - Monty Hall
- clp(pfd(bn)) -sketch

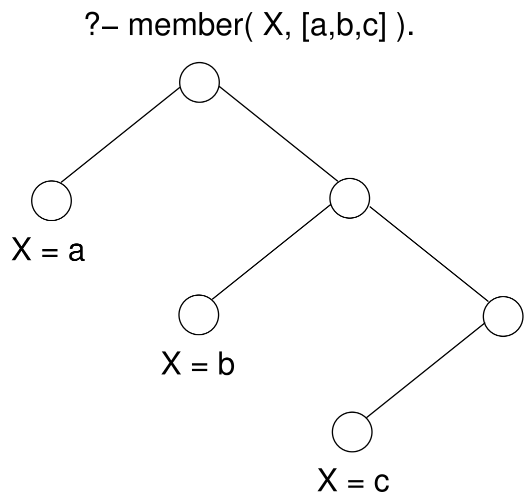
logic programming

Used in AI for crisp problem solving and for building executable models and intelligent systems.

Programs are formed from logic based rules.

```
member( H, [H|T] ).
member( El, [H|T] ) :- member( El, T ).
```

execution tree



member(H, [H|T]).

member(El, [H|T]) :- member(El, T).

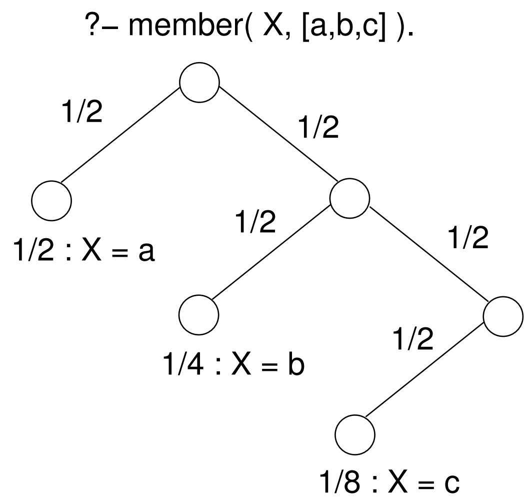
uncertainty in logic programming

Most approaches use Probability Theory but there are fundamental questions unresolved.

For example in SLP (stochastic logic programming),

```
0.5 : member( H, [H|T] ).
0.5 : member( El, [H|T] ) :- member( El, T ).
```

stochastic tree



0.5 : member(H, [H|T]).

0.5 : member(El, [H|T]) :- member(El, T).

Prism example

```
/* Declarations */
target( pmember, 2 ).
values( m(List), List ).

/* Model */
pmember( El, List ) :-
    msw( m(List), El ).

/* Utility part */
prob_pmember( El, List, Prob ):-
    length( List, Length ),
    get_uniform_param( Length, Params ),
    set_sw( m(List), Params),
    prob(pmember(El, List),Prob).
```

overloading

In these and similar formalisms both logical and statistical inference are done by a single engine.

As a result, either statistical reasoning is subordinate to logical reasoning or vice versa.

constraints in lp

Logic Programming :

- execution model is inflexible, and
- its relational nature discourages use of state information.

constraints in lp

Logic Programming :

- execution model is inflexible, and
- its relational nature discourages use of state information.

Constraints add

- specialised algorithms

constraints in lp

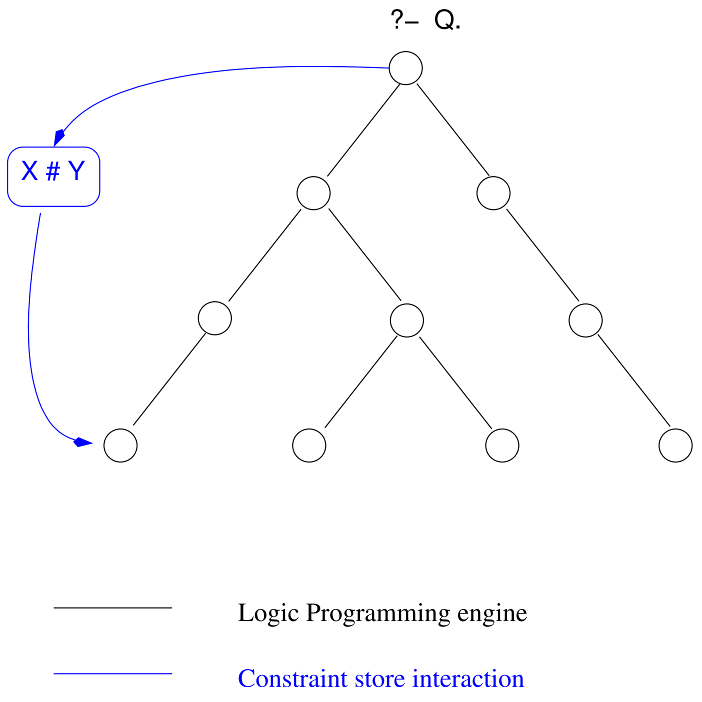
Logic Programming :

- execution model is inflexible, and
- its relational nature discourages use of state information.

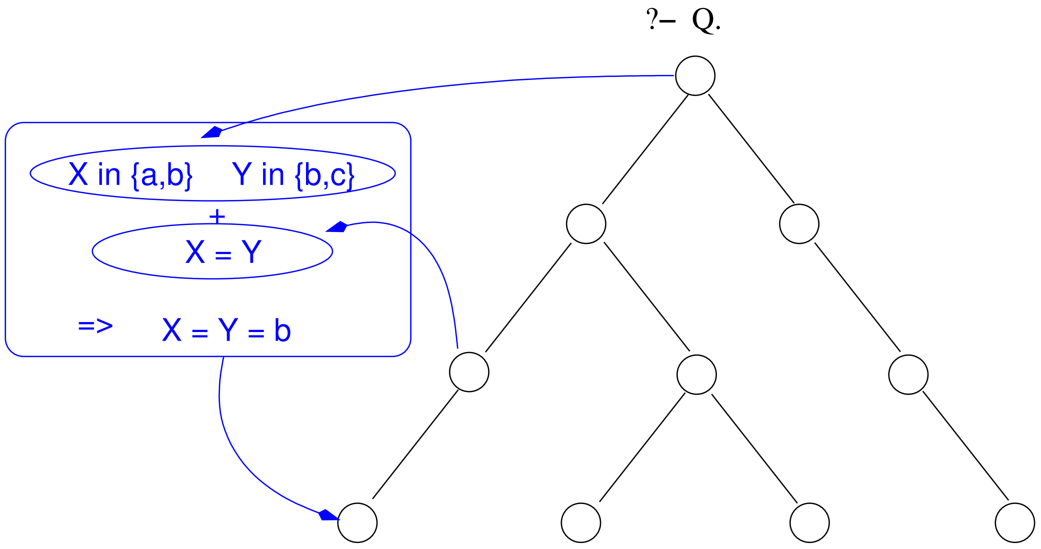
Constraints add

- specialised algorithms
- state information

constraint store



constraints inference



finite domain distributions

For discrete probabilistic models clp(pfd(Y)) extends the idea of finite domains to admit distributions.

from clp(fd)

$$X \ in \ \{a, b\} \qquad \text{(i.e. } X = a \quad \text{or} \quad X = b)$$

to clp(pfd(Y))

$$p(X = a) + p(X = b)$$

finite domain distributions

For discrete probabilistic models clp(pfd(Y)) extends the idea of finite domains to admit distributions.

from clp(fd)

$$X \ in \ \{a, b\} \qquad \text{(i.e. } X = a \quad \text{or} \quad X = b)$$

to clp(pfd(Y))

$$[\ p(X = a) + p(X = b)\ ] = 1$$

constraint based integration

Execution, assembles the probabilistic model in the store according to program and query.

Dedicated algorithms can be used for probabilistic inference on the model present in the store.

clp(pfd(Y))

For finite domain variable $V \ in \ \{e_1, \ldots, e_n\}$

and specific probabilistic inference algorithm Y, clp(pfd(Y)) assumes

$$\psi_{\mathcal{S}}(V) = \{(e_1, \pi_1), (e_2, \pi_2), \ldots, (e_n, \pi_n)\}$$

We let $p(e_i) = \pi_i$.

Given a particular store and program:

probability of a query or predicate containing probabilistic variables is equal to the sum of product of probabilities for elements that satisfy the query.

clp(pfd(Y)) example

For example, for program $\mathcal{P}_1$:

lucky(iv, hd). lucky(v, hd). lucky(vi, hd).

store $\mathcal{S}_1$ with variables D and C, with

$$\psi_{\mathcal{S}_1}(D) = \{(i, 1/6), (ii, 1/6), (iii, 1/6), (iv, 1/6), (v, 1/6), (vi, 1/6)\}$$

$$\psi_{\mathcal{S}_1}(C) = \{(hd, 1/2), (tl, 1/2)\}.$$

The probability of a lucky combination is $P_{\mathcal{S}_1}(lucky(D, C)) = 1/4$.

probability of predicates

- $\mathcal{S}$ - a constraint store.
- e - vector of finite domain elements
- E/e - E with variables replaced by e.

The probability of predicate E with respect to store $\mathcal{S}$ is

$$P_{\mathcal{S}}(E) = \sum_{\substack{\forall e \\ \mathcal{S} \vdash E/e}} P_{\mathcal{S}}(e) = \sum_{\substack{\forall e \\ \mathcal{S} \vdash E/e}} \prod_i p(e_i)$$

labelling

In clp(fd) systems labelling instantiates a group of variables to elements from their domains.

In clp(pfd(Y)) the probabilities can be used to guide labelling. For instance we have implemented

$$label(Vars, mua, Prbs, Total)$$

Selector mua approximates best-first algorithm which instantiates a group of variables to most likely combinations.

clp(pfd(c))

Probabilistic variables are declared with

$$V \sim \phi_V(Fd, Args)$$

e.g. $$Heat \sim finite\_geometric([l, m, h], [2])$$

finite geometric distribution with deterioration factor is 2.
In the absence of other information

$$\psi_{\{Heat\}}(Heat) = \{(l, 4/7)(m, 2/7), (h, 1/7)\}$$

Probability ascribing function ϕ_V and finite domain Fd are kept separately.

clp(pfd(c)) conditionals

Conditional C

$$D_1 : \pi_1 \oplus \ldots \oplus D_m : \pi_m \;|\; Q$$

Each D_i is a predicate and all should share a single probabilistic variable V. Q is a predicate not containing V, and

$$0 \leq \pi_i \leq 1, \sum_i \pi_i = 1$$

V's distribution is altered as a result of C being added to the store.

conditional different-than

Conditional different-than constraint

$$Y \mid\neq_{\pi} Z$$

Equivelant to

$$Y \neq T : \pi \oplus Y = T : (1 - \pi) \mid Z = T$$

conditional example

lucky(iv, hd). lucky(v, hd). lucky(vi, hd).

$$Coin \sim uniform([hd, tl])$$
$$Die \sim uniform([i, ii, iii, iv, v, vi])$$

```
constrained( P ) :-
Coin pin uniform( [hd,tl] ),
Die pin uniform( [i,ii,iii,iv,v,vi] ),
Die # v :: 1/3 ++ Die = v :: 2/3 \\ Coin = hd,
P is p( lucky(Die,Coin) ).
```

Querying this program

```
?- constrained(Prb)
```

$Prb = 2/5.$

Caesar's coding

Each letter is encrypted to a random letter. Words drawn from a dictionary are encrypted. Programs try to decode them. We compared a clp(fd) solution to clp(pfd(c)) .

clp(fd) no probabilistic information, labelling in lexicographical order.

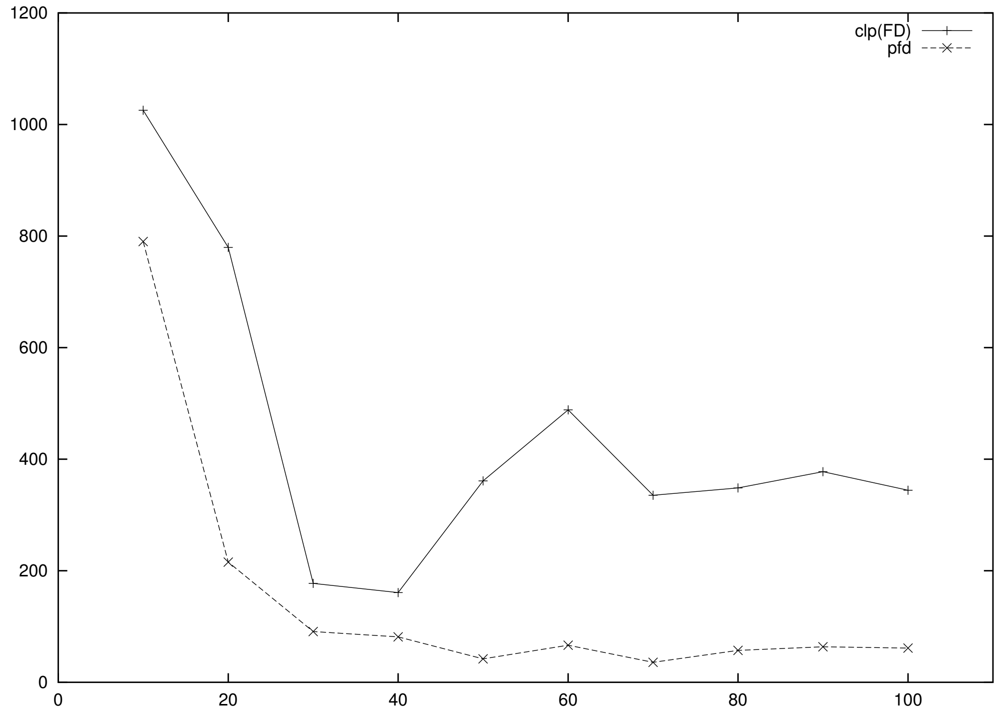
clp(pfd(c)) distributions based on frequencies, labelling using mua.

proximity functions

- X_i - variable for ith encoded letter
- D_i - dictionary letter
- freq() - frequency of letter

$$px(X_i, D_j) = \frac{1/ \mid freq(X_i) - freq(D_j) \mid}{\sum_k 1/ \mid freq(X_i) - freq(D_k) \mid}$$

execution times



clp(pfd(c)) and clp(fd) on SICStus 3.8.6

Monty Hall

Three curtains hiding a car and two goats.
Contestant chooses an initial curtain.
A close curtain opens to reveal a goat.
Contestant is asked for their final choice.

What is the best strategy ?
Stay or Switch ?

Monty Hall solution

If probability of switching is Swt,
($Swt = 0$ for strategy $Stay$ and $Swt = 1$ for $Switch$)

then probability of win is $P(\gamma) = \frac{1+Swt}{3}$.

Monty Hall in clp(pfd(c))

curtains(gamma, Swt, Prb) :-

$Gift \sim uniform([a,b,c])$,

$First \sim uniform([a,b,c])$,

$Reveal \sim uniform([a,b,c])$,

$Second \sim uniform([a,b,c])$,

$Reveal \neq Gift$, $Reveal \neq First$, $Second \neq Reveal$,

$Second$ $\Vdash$$\!\!\!\!\!\!\!\neq_{Swt}$ $First$,

Prb is $\mathbf{p}(Second{=}Gift)$.

Strategy γ Query

Querying this program

```
?- curtains(gamma, 1/2, Prb)
```

$Prb = 1/2$.

Strategy γ Query

Querying this program

```
?- curtains(gamma, 1/2, Prb)
```

$Prb = 1/2$.

```
?- curtains(gamma, 1, Prb)
```

$Prb = 2/3$.

Strategy γ Query

Querying this program

```
?- curtains(gamma, 1/2, Prb)
```

$Prb = 1/2$.

```
?- curtains(gamma, 1, Prb)
```

$Prb = 2/3$.

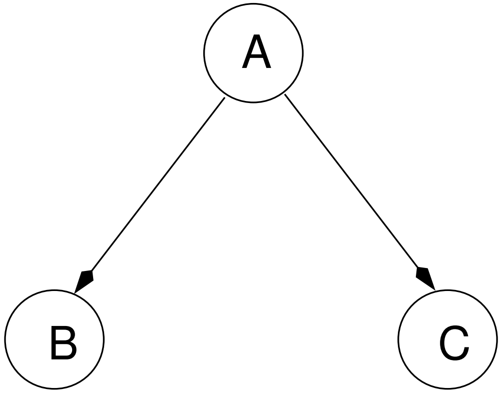
```
?- curtains(gamma, 0, Prb)
```

$Prb = 1/3$.

clp(pfd(bn))

Other discrete probabilistic inference engines can be employed. For instance Bayesian Networks representation and inference.

example BN



	A = y	A = n
B = y	0.80	0.10
B = n	0.20	0.90

	A = y	A = n
C = y	0.60	0.90
C = n	0.40	0.10

clp(pfd(bn)) program

```
example_bn( A, B, C ) :-
    cpt(A,[],[y,n]),
    cpt(B,[A],[(y,y,0.8),(y,n,0.2),
                (n,y,0.1),(n,n,0.9)]),
    cpt(C,[A],[(y,y,0.6),(y,n,0.4),
                (n,y,0.9),(n,n,0.1)]).
```

program

```
example_bn( A, B, C ) :-
    cpt(A,[],[y,n]),
    cpt(B,[A],[(y,y,0.8),(y,n,0.2),
               (n,y,0.1),(n,n,0.9)]),
    cpt(C,[A],[(y,y,0.6),(y,n,0.4),
               (n,y,0.9),(n,n,0.1)]).

?- example_bn(X,Y,Z),
   evidence(X,[(y,0.8),(n,0.2)],
   Zy is p(Z = y).
```

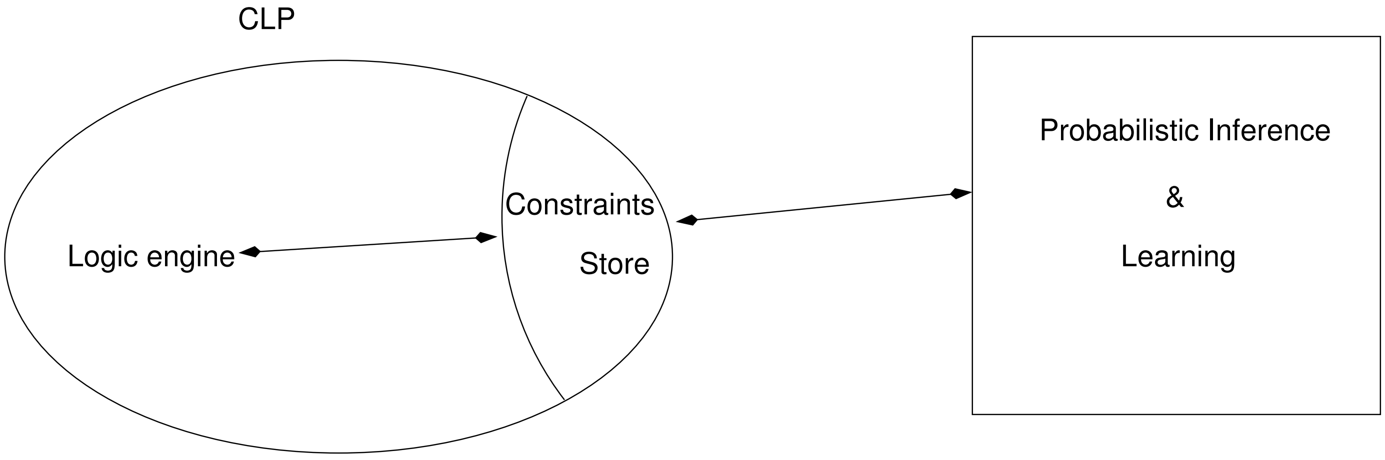
program

```
example_bn( A, B, C ) :-
    cpt(A,[],[y,n]),
    cpt(B,[A],[(y,y,0.8),(y,n,0.2),
               (n,y,0.1),(n,n,0.9)]),
    cpt(C,[A],[(y,y,0.6),(y,n,0.4),
               (n,y,0.9),(n,n,0.1)]).
```

```
?- example_bn(X,Y,Z),
   evidence(X,[(y,0.8),(n,0.2)],
   Zy is p(Z = y).
```

Zy = 0.66

current inference scheme

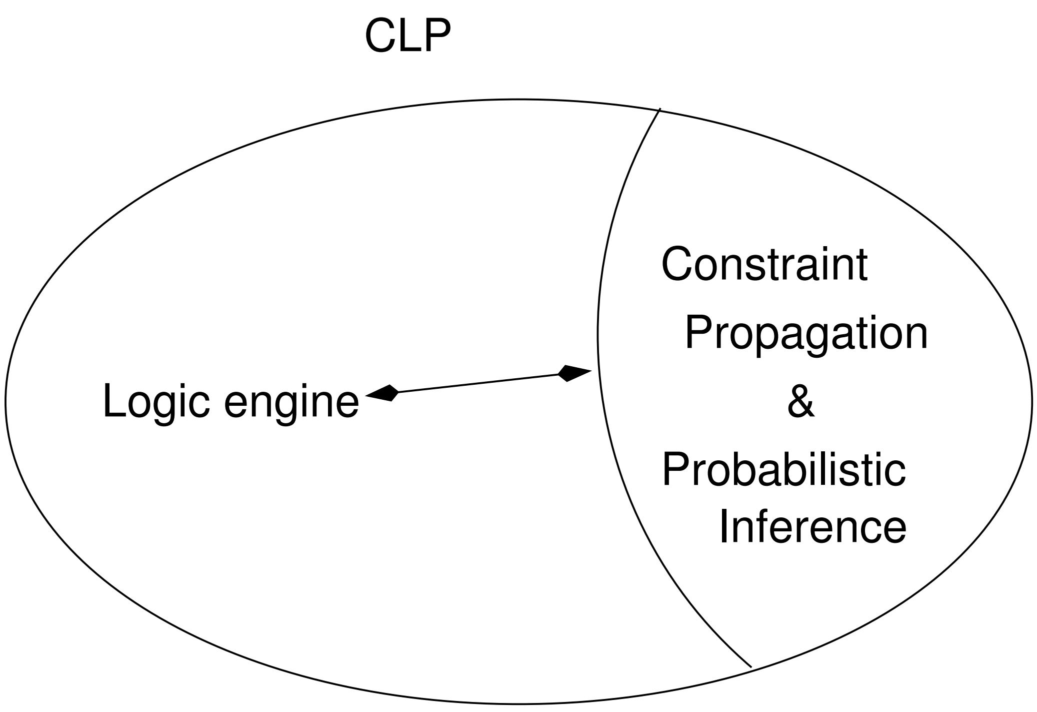


bottom line

Constraint LP based techniques can be used for frameworks that support probabilistic problem solving.

clp(pfd(Y)) can be used to take advantage of probabilistic information at an abstract level.

bottom line



CLP
Logic engine
Constraint
Propagation
&
Probabilistic
Inference