Probabilistic Logic Programming with cplint

Week 2, lecture 1: hybrid programs

Fabrizio Riguzzi

Hybrid Programs

- Up to now only discrete random variables and discrete probability distributions.
- Hybrid Probabilistic Logic Programs: some of the random variables are continuous.
- cplint allows the specification of density functions over arguments of atoms in the head of rules

Hybrid Programs

- A probability density on an argument `Var` of an atom `A` is specified with

  `A : Density :- Body.`

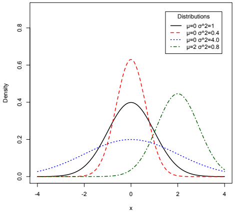
  where `Density` is a special atom
  - `uniform(Var,L,U)`: `Var` is uniformly distributed in $[L, U]$
  - `gaussian(Var,Mean,Variance)`: Gaussian distribution
  - `dirichlet(Var,Par)`: Dirichlet distribution with parameters α specified by the list `Par`
  - `gamma(Var,Shape,Scale)`: gamma distribution
  - `beta(Var,Alpha,Beta)`: beta distribution
  - + others (see the manual)

Hybrid Programs

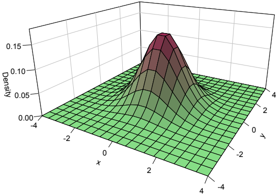
- Also discrete distribution, with either a finite or countably infinite support:
  - `discrete(Var,D)` or `finite(Var,D)`: `D` is a list of couples `Value:Prob` assigning probability `Prob` to `Value`
  - `uniform(Var,D)`: `D` is a list of values each taking the same probability (1 over the length of `D`).
  - `poisson(Var,Lambda)`: Poisson distribution

Examples

```
g(X) : gaussian(X,0,1).
```




```
g(X) : gaussian(X,[0,0],[ [1,0],[0,1] ]).
```



Inference

- If an atom encodes a continuous random variable (such as `g(X)` above), asking the probability that a ground instantiation, such as `g(0.3)`, is true is not meaningful, as the probability that a continuous random variables takes a specific value is always 0.
- In this case you are more interested in computing the distribution of `X` of a goal `g(X)`, possibly after having observed some evidence.

Gaussian Mixture Example

- `http://cplint.eu/e/gaussian_mixture.pl` defines a mixture of two Gaussians:

  ```
  heads:0.6;tails:0.4.
  g(X): gaussian(X,0, 1).
  h(X): gaussian(X,5, 2).
  mix(X) :- heads, g(X).
  mix(X) :- tails, h(X).
  ```

- The argument `X` of `mix(X)` follows a distribution that is a mixture of two Gaussian, one with mean 0 and variance 1 with probability 0.6 and one with mean 5 and variance 2 with probability 0.4.

Gaussian Mixture Example

- We can perform the query

  `mc_sample_arg(mix(X),1000,X,Values).`

- `histogram/3` draws a histogram of values

  `histogram(+List:list,-Chart:dict,+Options:list).`

- possible `Options`:
  - `min(+Min:float)` the minimum value of domain, default value the minimum in List
  - `max(+Max:float)` the maximum value of domain, default value the maximum in List
  - `nbins(+NBins:int)` the number of bins for dividing the domain, default value 40

Gaussian Mixture Example

- Probability density function of `X`, we can use

```
mc_sample_arg(mix(X),1000,X,_Values),
  histogram(_Values,Chart,[]).
```

Posterior estimation in Bayesian models Example

- The parameters of the distribution atoms can be taken from the probabilistic atom, the example
  http://cplint.eu/e/gauss_mean_est.pl

```
val(I,X) :-
  mean(M),
  val(I,M,X).
mean(M): gaussian(M,1.0, 5.0).
val(_,M,X): gaussian(X,M, 2.0).
```

- states that for an index `I` the continuous variable `X` is sampled from a Gaussian whose variance is 2 and whose mean is sampled from a Gaussian with mean 1 and variance 5.

Kalman Filter Example

- Any operation is allowed on continuous random variables. Kalman filter `http://cplint.eu/e/kalman_filter.pl`:

```
kf(N,O, T) :-
  init(S),
  kf_part(0, N, S,O,T).
kf_part(I, N, S,[V|RO], T) :-
  I < N,
  NextI is I+1,
  trans(S,I,NextS),
  emit(NextS,I,V),
  kf_part(NextI, N, NextS,RO, T).
kf_part(N, N, S, [],S).
trans(S,I,NextS) :-
  {NextS =:= E + S},
  trans_err(I,E).
emit(NextS,I,V) :-
  {NextS =:= V+X},
  obs_err(I,X).
init(S):gaussian(S,0,1).
trans_err(_,E):gaussian(E,0,2).
obs_err(_,E):gaussian(E,0,1).
```

- In case random variables are not sufficiently instantiated to exploit expressions for inferring the values of other variables, inference will return an error.

Conditional Queries

- You can also execute conditional queries over hybrid programs.
- Sampling arguments of goals representing continuous random variables and drawing a probability density of the sampled argument.
- Three cases
  1. The evidence does not contain atoms with continuous random variables (the probability of evidence is different from 0).
  2. The evidence contains atoms with continuous random variables, but its probability is not zero.
  3. The evidence contains the grounding of atoms with continuous random variables (its probability is 0).
- For the first two cases you can use the predicates `mc_rejection_sample_arg/6` and `mc_mh_sample_arg/6`.

Conditional Queries, Case 1

- Take 1000 samples of `X` in mix(X) given that heads was true using rejection sampling and Metropolis-Hastings MCMC

```
mc_rejection_sample_arg(mix(X),heads,1000,X,_V),
  histogram(_V,Chart,[]).
mc_mh_sample_arg(mix(X),heads,1000,X,_V,[lag(2)]),
  histogram(_V,Chart,[]).
```

Conditional Queries, Case 2

- Take 1000 samples of `X` in `mix(X)` given that `X>2` was true using rejection sampling and draw an histogram of the probability density of `X`

```
mc_rejection_sample_arg(mix(X),(mix(Y),Y>2),
  1000,X,_V,), histogram(_V,Chart,[]).
mc_mh_sample_arg(mix(X),(mix(Y),Y>2),1000,X,
  _Values,[lag(2)]),histogram(_Values,Chart,[]).
```

Conditional Queries, Case 3

- When you have evidence on ground atoms that have continuous values as arguments (probability of the evidence is 0), you need to use likelihood weighting
- For each sample to be taken, likelihood weighting uses a meta-interpreter to find a sample where the goal is true
- Then a different meta-interpreter is used to evaluate the evidence attaching a weight to the sample.
- Each time the meta-interpreter encounters a probabilistic choice over a continuous variable, it it was already sampled, it computes the probability density of the sampled value and multiplies the weight by it.

Posterior estimation in Bayesian models Example

- Estimating the true value of a Gaussian distributed random variable, given some observed data.
- The variance is known (2) and we suppose that the mean has a Gaussian distribution with mean 1 and variance 5.
- We take different measurement (e.g. at different times), indexed with an integer. Given that we observe 9 and 8 at indexes 1 and 2, how does the distribution of the random variable (value at index 0) changes with respect to the case of no observations?

Posterior estimation in Bayesian models Example

- Likelihood weighing

```
mc_lw_sample_arg(:Query:atom,:Evidence:atom,
  +N:int,?Arg:var,-ValList)
```

- `ValList` a list of couples `V-W` where `V` is a value of `Arg` for which `Query` succeeds and `W` is the weight computed by likelihood weighting according to `Evidence`
- Given that we observe 9 and 8 at indexes 1 and 2, what is the distribution of the random variable (value at index 0)?

```
mc_lw_sample_arg(val(0,X),(val(1,9),val(2,8)),
  1000,X,V).
```

Posterior estimation in Bayesian models Example

`density(+List:list,-Chart:dict,+Options:list)`

draws a line chart of the density of the samples in `List`

`densities(+PriorList:list,+PostList:list,-Chart:dict,`
`+Options:list)`

draws a line chart of the density of two sets of samples, usually prior and post observations.
The same options as in `histogram/3` are recognised.

```
?-  mc_sample_arg(val(0,X),1000,X,L0,[]),
    histogram(L0,Chart,[]).
?-  mc_sample_arg(val(0,X),1000,X,L0,[]),
    density(L0,Chart,[]).
? - mc_sample_arg(val(0,Y),1000,Y,_V0),
    mc_lw_sample_arg(val(0,X),(val(1,9),val(2,8)),1000,X,_V),
    densities(_V0,_V,Chart,[]).
```

Expectations

```
mc_lw_expectation(:Query:atom,Evidence:atom,
  +N:int,?Arg:var,-Exp:float)
```

- computes the expected value of `Arg` in `Query` given that `Evidence` is true.
- It takes `N` samples, weighting each according to the evidence, and returns their weighted average.

Particle Filtering

- In some cases likelihood weighting encounters numerical problems, as the weights of samples may go rapidly to very small numbers that can be rounded to 0 by floating point arithmetic.
- This happens for example for dynamic models,
- Particle filtering periodically resamples the individual samples/particles so that their weight is reset to 1.
- In particle filtering, the evidence is a list of literals. A number n of samples of the query is taken that are weighted by the likelihood of the first element of the evidence list.
- Each sample constitutes a particle and the sampled random variables are stored away.
- After weighting, n particles are resampled with replacement with a probability proportional to their weight.
- Then the next element of the evidence is considered.

Particle Filtering Example

`http://cplint.eu/e/kalman_filter.pl`

```
?-[O1,O2,O3,O4]=[-0.133, -1.183, -3.212, -4.586],
  mc_particle_sample_arg([kf_fin(1,T1),kf_fin(2,T2),
  kf_fin(3,T3),kf_fin(4,T4)],
  [kf_o(1,O1),kf_o(2,O2),kf_o(3,O3),kf_o(4,O4)],100,
  [T1,T2,T3,T4],[F1,F2,F3,F4]).
```

performs particle filtering for a Kalman filter with four observations. For each observation, the value of the state at the same time point is sampled. The list of samples is returned in `[F1,F2,F3,F4]`

```
mc_particle_sample(:Query:atom,:Evidence:list,
  +Samples:int,-Prob:float)
mc_particle_expectation(:Query:atom,Evidence:atom,
  +N:int,?Arg:var,-Exp:float)
```