PRIMO

PRobabilistic Inference MOdules

Manuel Baum, Denis John,
Lukas Kettenbach, Maximilian Koch

Bielefeld University

11. Oktober 2013

Introduction

Idea

- probabilistic inference modules for Python
- library which offers well known probabilistic (graphical) models like Bayesian or temporal networks
- variety of inference algorithms

Download/Documentation/Installation Guide

- `github.com/mbaumBielefeld/PRIMO`
- `github.com/mbaumBielefeld/PRIMO/wiki`

Structure

PRIMO/

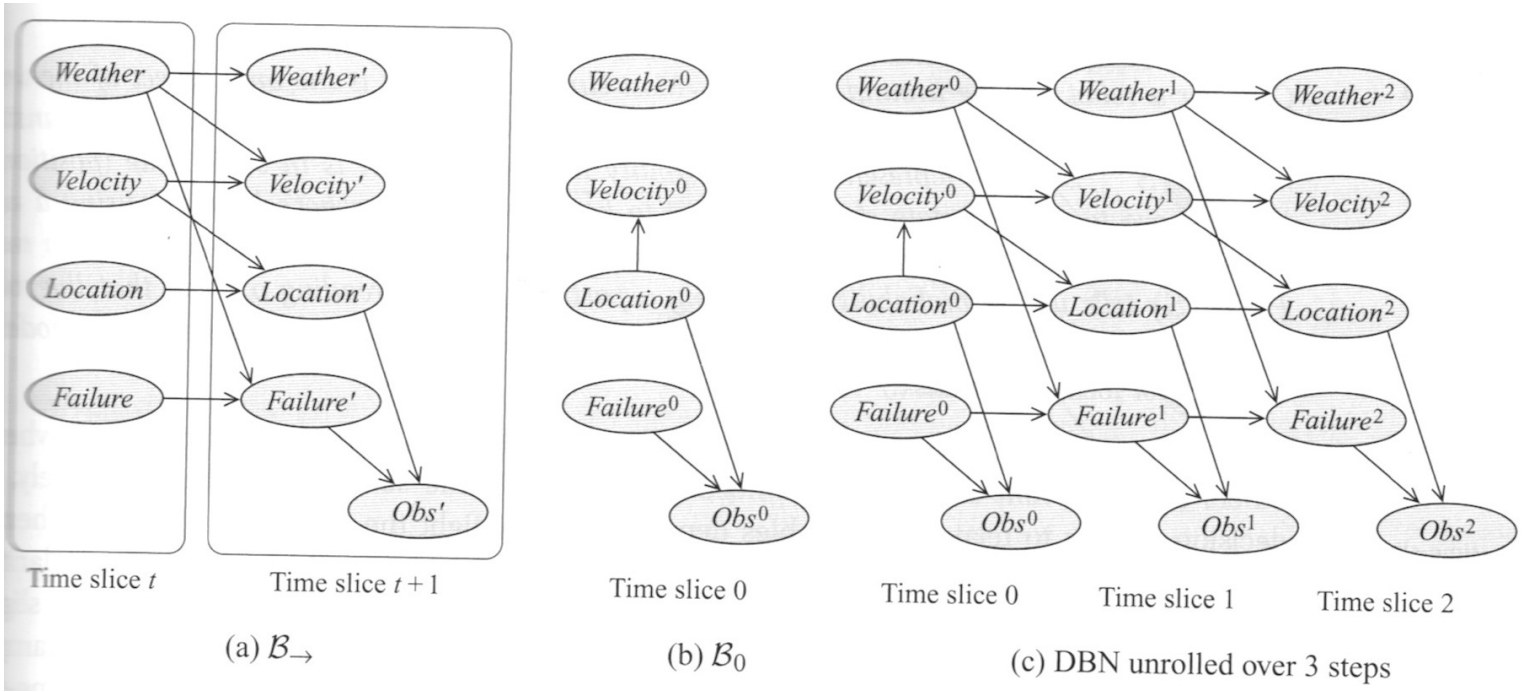
- doc/
- examples/
- primo/
  - core/ → BayesNet.py, Node.py, DynamicBayesNet.py, ...
  - decision/ → DecisionNode.py, UtilityNode.py, ...
  - reasoning/ → DiscreteNode.py, density/, MCMC.py, ...
  - tests/
  - utils/ → XMLBIF.py
- setup.py

Definition

A DBN is a pair $(B_0, B_{\rightarrow})$, where B_0 is a Bayesian network over $\chi^{(0)}$ representing the initial distribution, and $B_{\rightarrow}$ is a 2-TBN for the process. For any desired time span $T \geq 0$, the distribution over $\chi^{(0:T)}$ is defined as a unrolled Bayesian network, where, for any $i = 1, ..., n$:

- the structure and CPDs of $X_i^{(0)}$ are the same as those for X_i in B_0,
- the structure and CPDs of $X_i^{(t)}$ for $t \geq 0$ are the same as those for X_i' in $B_{\rightarrow}$.

Example


Weather
Velocity
Location
Failure
Obs'
Weather'
Velocity'
Location'
Failure'
Time slice t
Time slice $t+1$
(a) $\mathcal{B}_{\rightarrow}$
Weather^0
Velocity^0
Location^0
Failure^0
Obs^0
Time slice 0
(b) $\mathcal{B}_0$
Weather^0
Weather^1
Weather^2
Velocity^0
Velocity^1
Velocity^2
Location^0
Location^1
Location^2
Failure^0
Failure^1
Failure^2
Obs^0
Obs^1
Obs^2
Time slice 0
Time slice 1
Time slice 2
(c) DBN unrolled over 3 steps

Exact Inference

- We can use standard inference algorithms (e.g. variable elimination)
- Problem I: run inference on larger an larger networks over time
- Problem II: maintain our entire history of observations indefinitely
- Solution/workaround: use approximate inference

Approximate Inference

- We can use some kind of Likelihood Weighting
- Two modifications:
  1. run all samples together through the DBN, one slice at a time
  2. focus the set of samples on the high-probability regions of the state space
- Particle Filter:
  1. Each sample is propagated forward by sampling the next state value x_{t+1} given the current value x_t for the sample
  2. Each sample is weighted by the likelihood it assigns to the new evidence $P(e_{t+1}|x_{t+1})$
  3. The population is *resampled* to generate a new population of N samples. Each new sample is selected from the current population; the probability that a particular sample is selected is proportional to its weight.

Algorithm

Task Description

- Exact inference
- Use elimination trees
- Prior Marginal, Posterior Marginal & PoE

Algorithm 10 FE2($\mathcal{N}$, $\mathbf{Q}$, $(\mathcal{T}, \phi)$, r)

input:

$\mathcal{N}$:	Bayesian network
$\mathbf{Q}$:	some variables in network $\mathcal{N}$
$(\mathcal{T}, \phi)$:	elimination tree for the CPTs of network $\mathcal{N}$
r:	a node in tree $\mathcal{T}$ where $\mathbf{Q} \subseteq \text{vars}(r)$

output: the prior marginal Pr($\mathbf{Q}$)

main:

1: **while** tree $\mathcal{T}$ has more than one node **do**
2: remove a node $i \neq r$ having a single neighbor j from tree $\mathcal{T}$
3: $\mathbf{V} \leftarrow$ variables appearing in ϕ_i but not in remaining tree $\mathcal{T}$
4: $\phi_j \leftarrow \phi_j \sum_{\mathbf{V}} \phi_i$
5: **end while**
6: **return** project(ϕ_r, $\mathbf{Q}$)

Literature

- Modeling and Reasoning with Bayesian Networks , Adnan Darwiche

Thank you for your attention!