Bayesian Inference with Probabilistic Programs

Fritz Obermeyer

Department of Mathematics
Carnegie Mellon University

2009:04:14

Overview

Aim Explain research in Multiple Target Tracking

Aim Mention research in Programming Langauges

Theme Probabilistic Programs

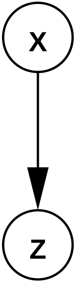
Deterministic programs

$X = 2$
$Z = X + 1$
output Z

Probabilistic programs

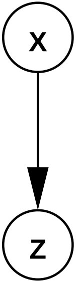
$$X \sim \mathrm{unif}(0, 1)$$
$$Z \sim \mathrm{unif}(0, X)$$
$$\textbf{output} \;\; Z$$

Probabilistic programs



$$X \sim \text{unif}(0, 1)$$
$$Z \sim \text{unif}(0, X)$$
$$\textbf{output}\ \ Z$$

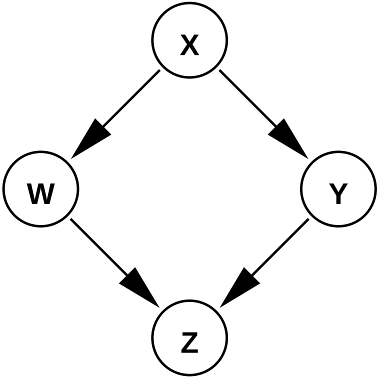
Probabilistic programs



$$X \sim \text{unif}(0, 1)$$
$$Z \sim \text{unif}(0, X)$$
$$\textbf{output} \;\; Z$$

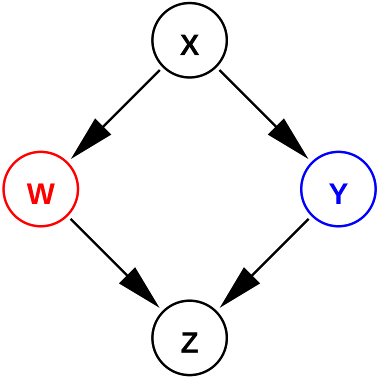
$$P(X, Z) = P(X) \;\; P(Z \mid X)$$

Bayesian networks use definitions



$$X \sim \phi_X$$
$$W \sim \phi_W(X)$$
$$Y \sim \phi_Y(X)$$
$$Z \sim \phi_Z(W, Y)$$
output Z

Bayesian networks use definitions



$X \sim \phi_X$

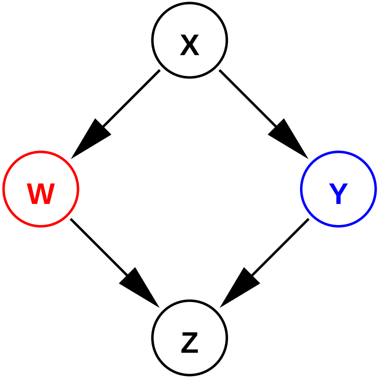
$W \sim \phi_W(X)$

$Y \sim \phi_Y(X)$

$Z \sim \phi_Z(W, Y)$

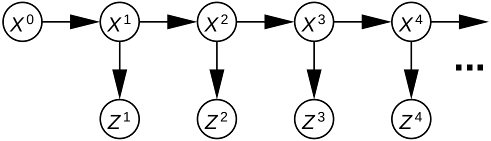
output Z

Bayesian networks use definitions



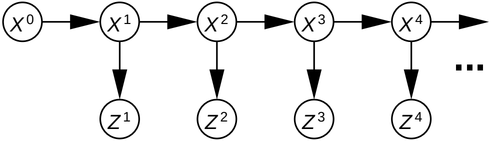
$$X \sim \phi_X$$
$$Y \sim \phi_Y(X)$$
$$W \sim \phi_W(X)$$
$$Z \sim \phi_Z(W, Y)$$
output Z

HMMs, Bayes filters use loops

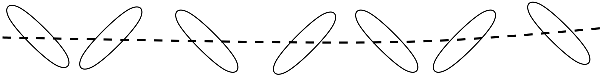


$\mathrm{X}^0 \sim \phi_{\mathrm{prior}}$
for t **in** $[1, \ldots]$:
  $\mathrm{X}^t \sim \phi_{\mathrm{trans}}(\mathrm{X}^{t-1})$
  $\mathrm{Z}^t \sim \phi_{\mathrm{obs}}(\mathrm{X}^t)$
  output Z^t

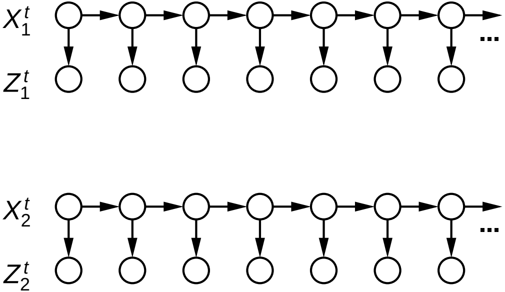
HMMs, Bayes filters use loops



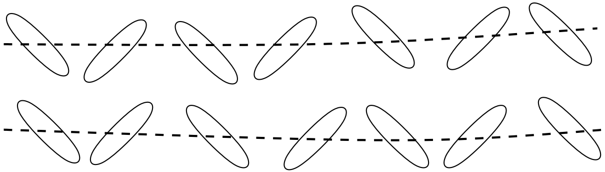
$X^0 \sim \phi_{\text{prior}}$
for t **in** $[1, \ldots]$:
  $X^t \sim \phi_{\text{trans}}(X^{t-1})$
  $Z^t \sim \phi_{\text{obs}}(X^t)$
  output Z^t



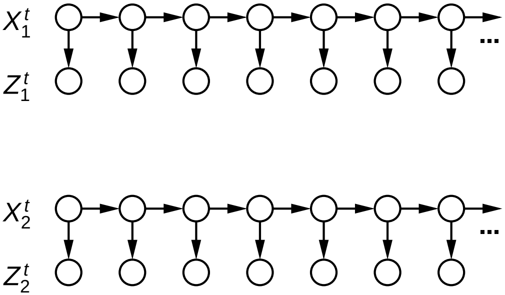
Multiple Target Tracking



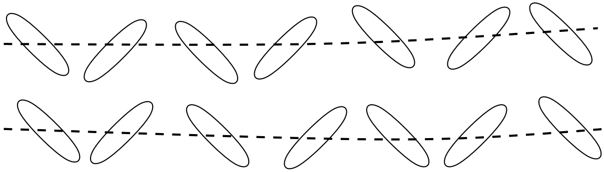
$X_1^0 \sim \phi_{\text{prior}}$
$X_2^0 \sim \phi_{\text{prior}}$
for t **in** $[1, \ldots]$:
    $X_1^t \sim \phi_{\text{trans}}(X_1^{t-1})$
    $X_2^t \sim \phi_{\text{trans}}(X_2^{t-1})$
    $Z_1^t \sim \phi_{\text{obs}}(X_1^t)$
    $Z_2^t \sim \phi_{\text{obs}}(X_2^t)$
    output Z_1^t, Z_2^t



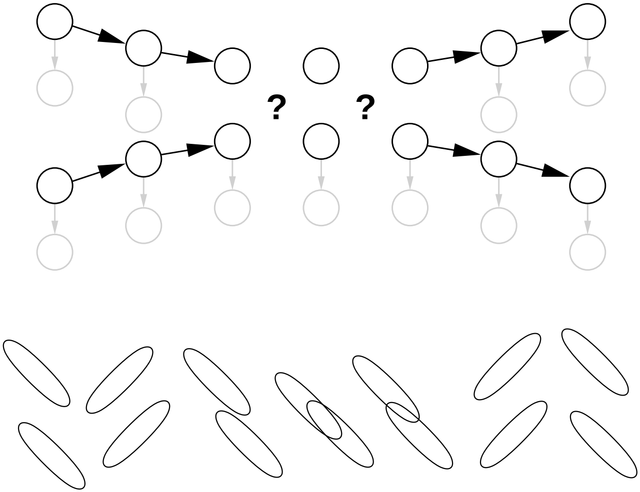
Multiple Target Tracking



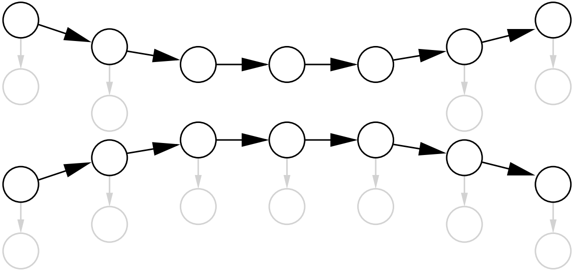
$X_1^0 \sim \phi_{\text{prior}}$
$X_2^0 \sim \phi_{\text{prior}}$
for t **in** $[1, \ldots]$:
  $X_1^t \sim \phi_{\text{trans}}(X_1^{t-1})$
  $X_2^t \sim \phi_{\text{trans}}(X_2^{t-1})$
  $Z_1^t \sim \phi_{\text{obs}}(X_1^t)$
  $Z_2^t \sim \phi_{\text{obs}}(X_2^t)$
  output $Z_?^t, Z_?^t$

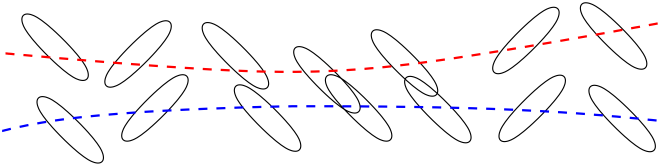


...but sometimes association is ambiguous

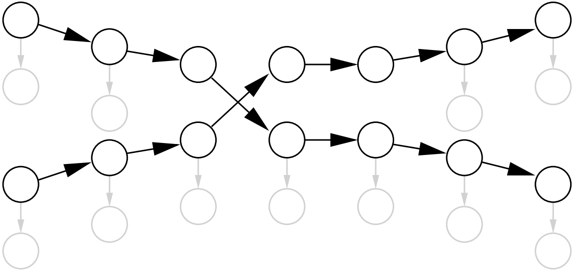


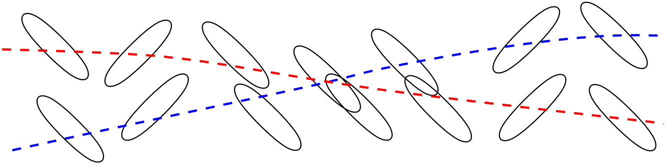
...but sometimes association is ambiguous



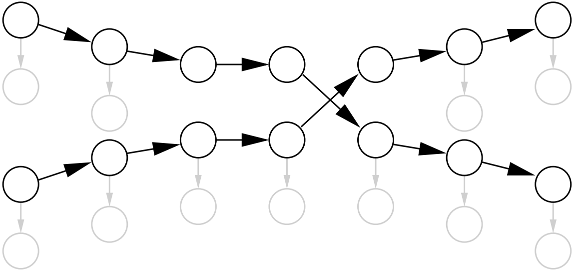


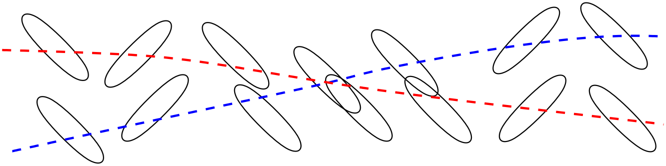
...but sometimes association is ambiguous



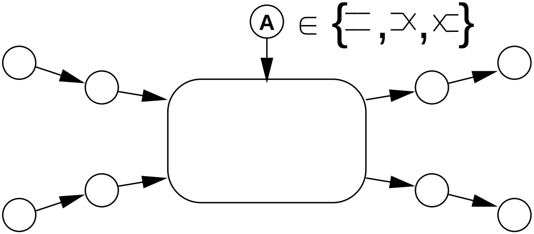


...but sometimes association is ambiguous





Multiple Hypothesis Tracking



...

for t **in** [1, ...]:

  $A \sim \phi_{\text{assoc}}(\ldots)$

  branch on A:

    case $=$:

      $X_1^t \sim \phi_{\text{trans}}(X_1^{t-1})$

      $X_2^t \sim \phi_{\text{trans}}(X_2^{t-1})$

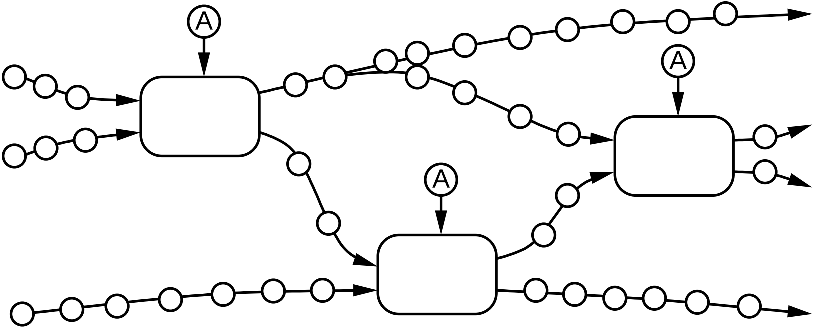
    case $\times$:

      $X_1^t \sim \phi_{\text{trans}}(X_2^{t-1})$

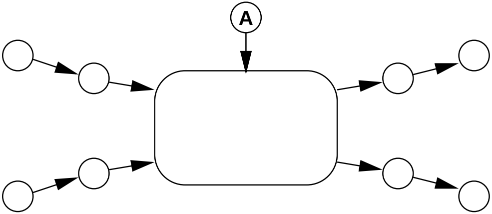
      $X_2^t \sim \phi_{\text{trans}}(X_1^{t-1})$

...

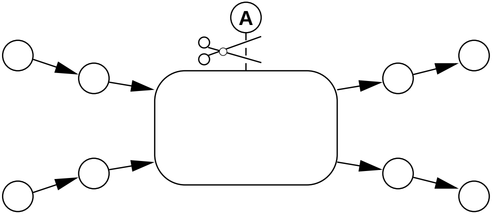
Long-term MHT is too expensive



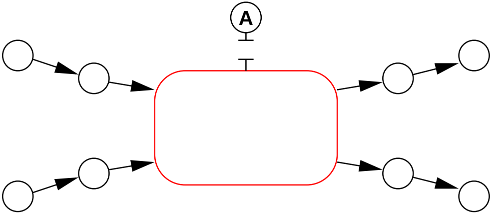
Bayes Net Tracking Database



Bayes Net Tracking Database



Bayes Net Tracking Database



$\ldots$

for t **in** $[1, \ldots]$:

  if X_1^t is close to X_2^t:

    $\underline{X}^t \sim \phi_{\text{mix}}(\underline{X}^{t-1})$

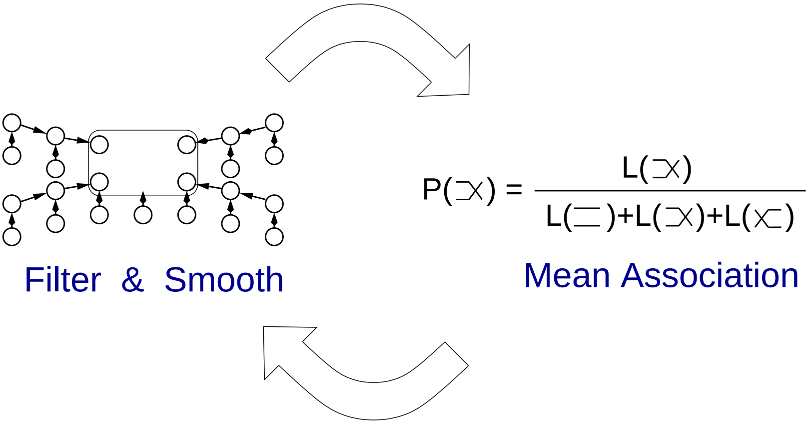
  else:

    $X_1^t \sim \phi_{\text{trans}}(X_1^{t-1})$

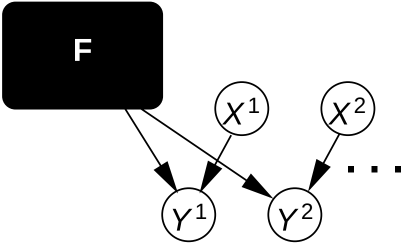
    $X_2^t \sim \phi_{\text{trans}}(X_2^{t-1})$

  $\ldots$

Iteratively compute state+association



What about more general programs?



$F \sim \phi_{\text{program}}$
for i **in** $[1, \ldots]$:
  input X^i
  $Y^i \sim F(X^i)$
  output Y^i

What is a random program?

What is a random program?

$$\begin{array}{rcl} \text{prog} & ::= & \text{S} \\ & | & \text{K} \\ & | & \text{prog(prog)} \end{array}$$

What is a random program?

$$\begin{aligned}\text{prog} \quad ::= \quad & \text{S} \\ | \quad & \text{K} \\ | \quad & \text{prog(prog)}\end{aligned}$$

$$\begin{aligned}\phi_{\text{prog}} = & \ 1/4 \ (\textbf{return} \ \ \text{S}) \\ & + 1/4 \ (\textbf{return} \ \ \text{K}) \\ & + 1/2 \ (\ \ \text{F} \sim \phi_{\text{prog}} \\ & \qquad\quad \text{X} \sim \phi_{\text{prog}} \\ & \qquad\quad \textbf{return} \ \ \text{F(X)} \ \ )\end{aligned}$$

Research in programming languages

Developed tools for coarse equivalence

Research in programming languages

Developed tools for coarse equivalence

Simulated types in untyped languages

Research in programming languages

Developed tools for coarse equivalence

Simulated types in untyped languages

Implemented an equational theorem prover

Research in programming languages

Developed tools for coarse equivalence

Simulated types in untyped languages

Implemented an equational theorem prover

Finds first 10,000 programs-mod-equivalence

Research in programming languages

Developed tools for coarse equivalence

Simulated types in untyped languages

Implemented an equational theorem prover

- Finds first 10,000 programs-mod-equivalence
- Generates conjectures from evidence

Research in programming languages

- Developed tools for coarse equivalence
- Simulated types in untyped languages
- Implemented an equational theorem prover
    - Finds first 10,000 programs-mod-equivalence
    - Generates conjectures from evidence
    - Optimizes language to fit examples

Questions?