Artificial Intelligence

Lecture 8：Bayesian Network II

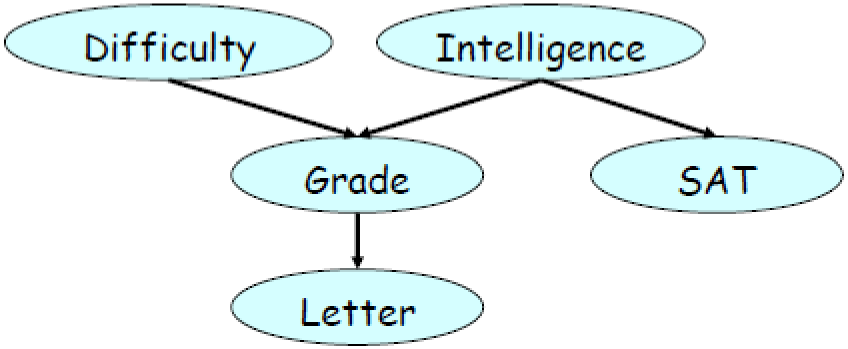
Xiaojin Gong

2021-04-26

Credits: AI Course in Berkeley

Review

- Bayesian Network
  - Representation
    - Joint Probability
    - Conditional Independence
  - Inference
    - Enumeration
    - Variable Elimination
    - Rejection Sampling
    - Likelihood Weighting
    - Gibbs Sampling



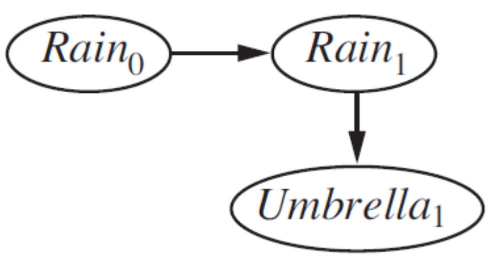
Outline

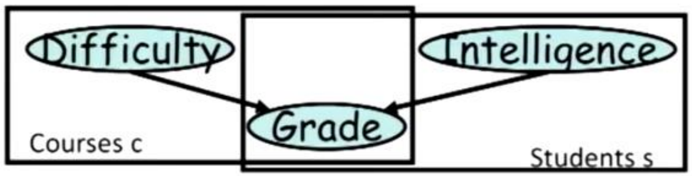
- Bayesian network
    - Representation
        - Template models
    - Inference
    - Applications

Template Models

- Temporal Modeling
  - State-observation model
  - Hidden Markov model
  - Dynamic Bayesian network
- Object-relational models
  - Plate models

Repetition

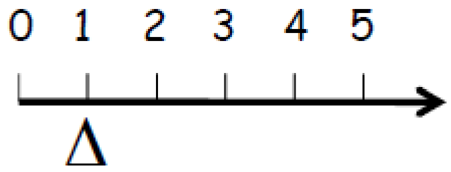




Temporal Models

- Distributions over Time
  - Discretize the timeline into a set of time slices
  - X_t: the set of state variables at time $t\Delta$
  - $X_{t:t'} = \{X_t, \dots, X_{t'}\} \quad (t \leq t')$

$$P(X_{0:T}) = P(X_0) \prod_{t=0}^{T-1} P(X_{t+1} | X_{0:t})$$



Markov Chains / Markov Processes

- Markov Assumption:
  - The current state depends on only a finite fixed number of previous states.
  - Such systems are called Markovian.
- First-order Markov chains
  - A dynamic system over the template variable X satisfies the Markov assumption if, for all $t \geq 0$,

$$(X_{t+1} \perp X_{0:t-1} | X_t)$$

  - The distribution over time:

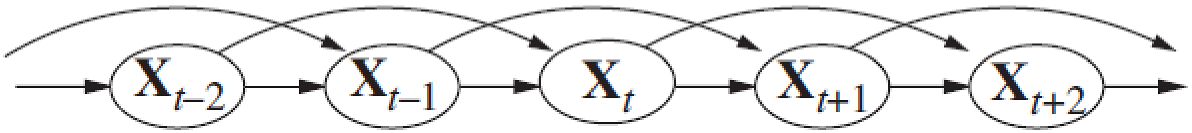
$$P(X_{0:T}) = P(X_0) \prod_{t=0}^{T-1} P(X_{t+1} | X_t)$$

- A Markovian dynamic system is stationary (time invariance) if $P(X_{t+1}|X_t)$ is the same for all t.

Markov Chains

- First-order Markov chain
  - $(X_{t+1} \perp X_{0:t-1} | X_t)$

- Second-order Markov chain
  - $(X_{t+1} \perp X_{0:t-2} | X_t, X_{t-1})$



State-Observation Models

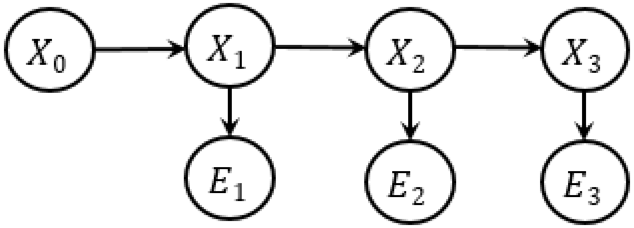
- Two independence assumptions
  - The state variables evolve in a Markovian way

$$(X_{t+1} \perp X_{0:t-1} | X_t)$$

  - The observation variables at time t conditionally independent of the entire state sequence given the state variables at time t:

$$(E_t \perp X_{0:t-1}, X_{t+1:\infty} | X_t)$$

$$(E_t \perp E_{1:t-1}, E_{t+1:\infty} | X_t)$$



State-Observation Models

- The transition model:

$$\mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t \mid \mathbf{X}_{t-1})$$

- The sensor/observation model:

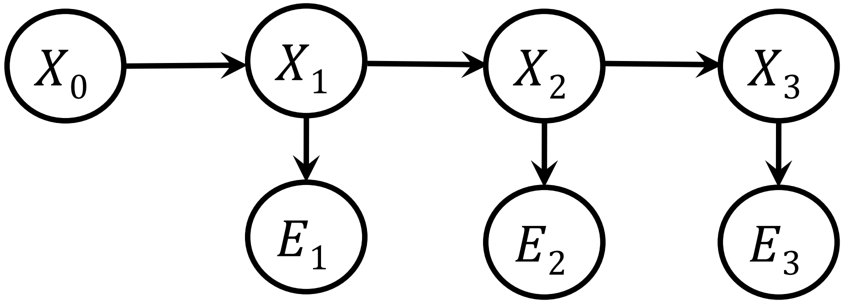
$$\mathbf{P}(\mathbf{E}_t \mid \mathbf{X}_{0:t}, \mathbf{E}_{0:t-1}) = \mathbf{P}(\mathbf{E}_t \mid \mathbf{X}_t)$$

- The initial state model: $P(X_0)$
- Joint distribution:

$$\mathbf{P}(\mathbf{X}_{0:t}, \mathbf{E}_{1:t}) = \mathbf{P}(\mathbf{X}_0) \prod_{i=1}^{t} \mathbf{P}(\mathbf{X}_i \mid \mathbf{X}_{i-1}) \, \mathbf{P}(\mathbf{E}_i \mid \mathbf{X}_i)$$

Hidden Markov Models

- HMM is a state-observation model in which the state is described by a single discrete random variable.



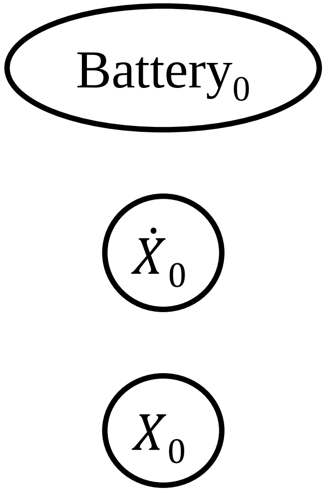
- Applications:
  - Speech recognition
  - Part-of-speech tagging

Dynamic Bayesian Networks

- Dynamic Bayesian Network (DBN) represents a temporal probability model in which each slice can have any number of state variables and evidence variables
- A DBN contains:
  - The prior distribution over the state variables $P(X_0)$
  - The transition model $P(X_{t+1}|X_t)$
  - The sensor model $P(E_t|X_t)$

Dynamic Bayesian Networks

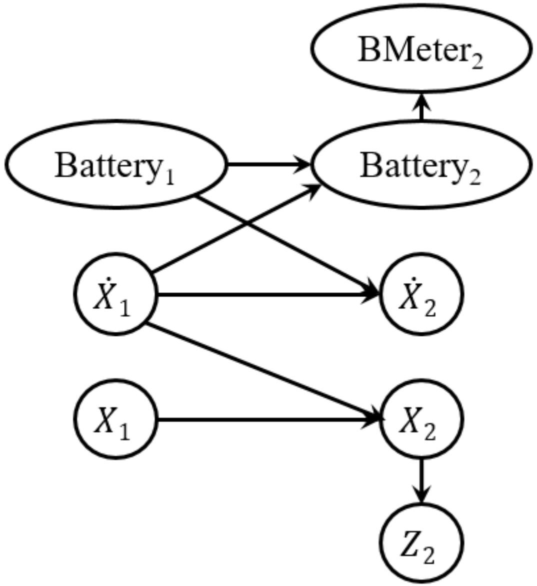
- An initial state distribution $P(X_0)$
  - Eg. Monitoring a battery-powered robot moving in the X-Y plane



Dynamic Bayesian Networks

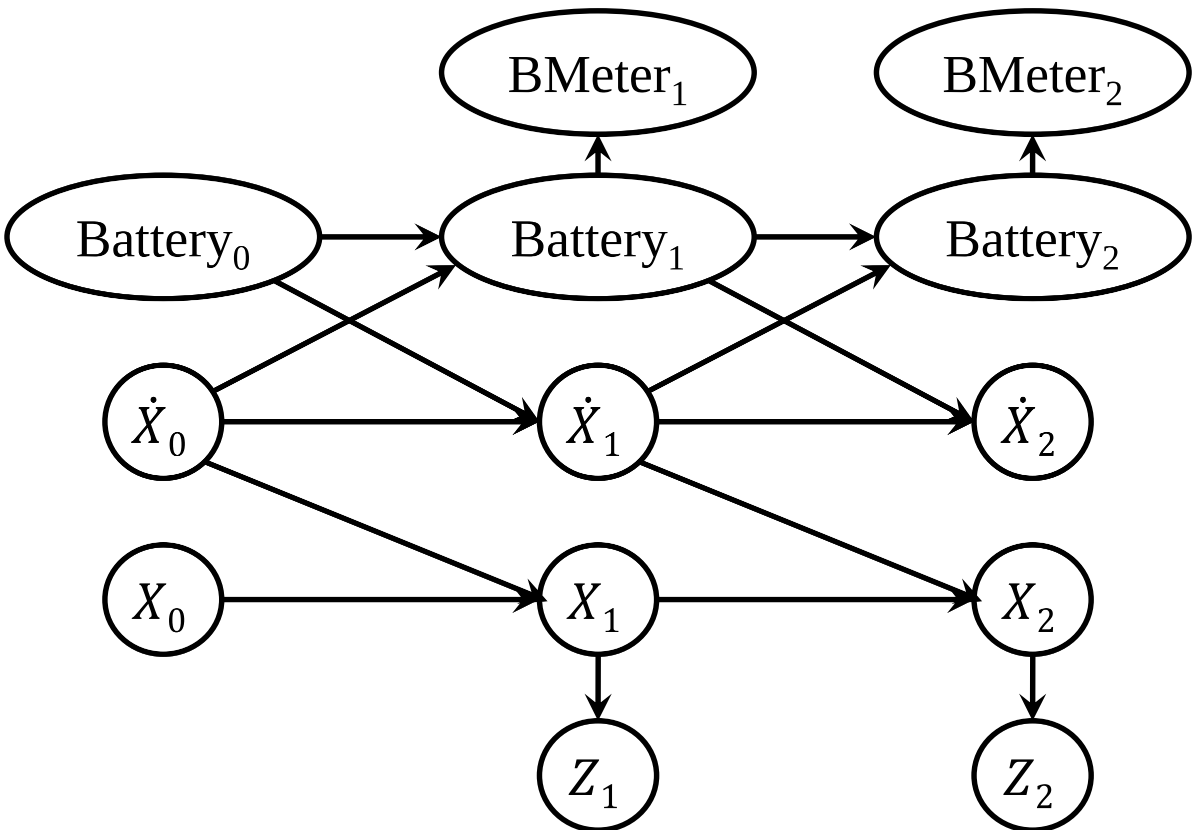
- A transition model $P(X_{t+1}|X_t)$
  - A 2-time-slice Bayesian Network (2TBN) over $X^1, \cdots, X^n$ is specified as a BN fragment s.t.
    - The nodes include $X_{t+1}^1, \cdots, X_{t+1}^n$ and a subset of $X_t^1, \cdots, X_t^n$.
    - Only the nodes $X_{t+1}^1, \cdots, X_{t+1}^n$ have parents and a CPD
  - The 2TBN defines a conditional distribution

$$P(X_{t+1}|X_t) = \prod P(X_{t+1}^i|Pa(X_{t+1}^i))$$



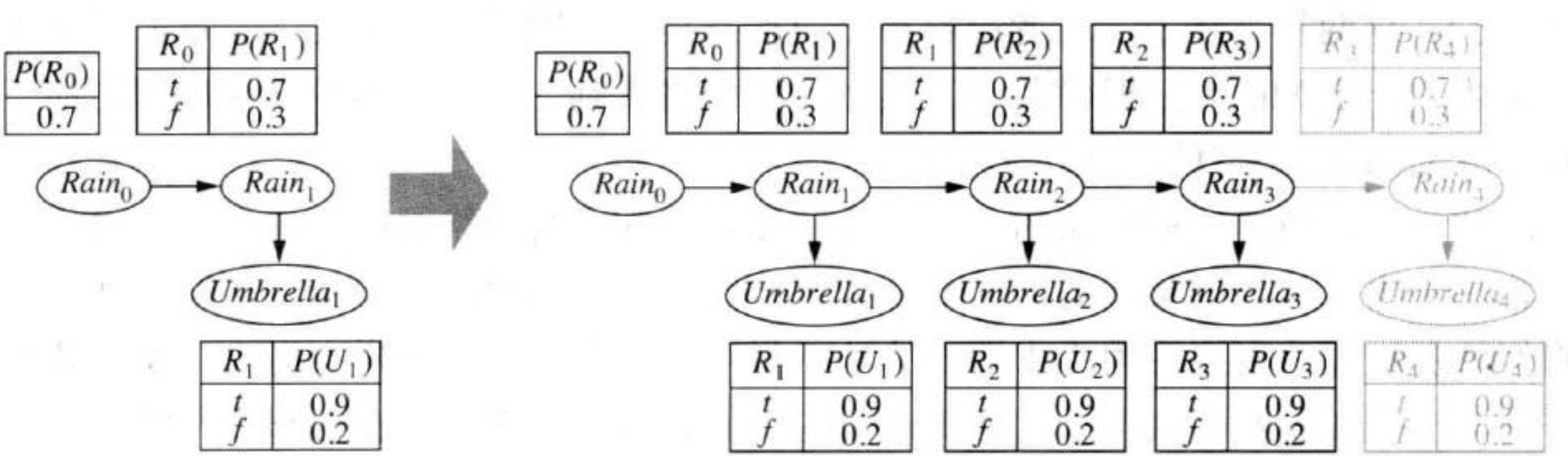
Dynamic Bayesian Networks

- A dynamic Bayesian network (DBN) over $X^1, \cdots, X^n$ is defined by
  - A Bayesian network BN_0 over $X_0^1, \cdots, X_0^n$
  - A 2TBN $BN_{\rightarrow}$ over $X^1, \cdots, X^n$



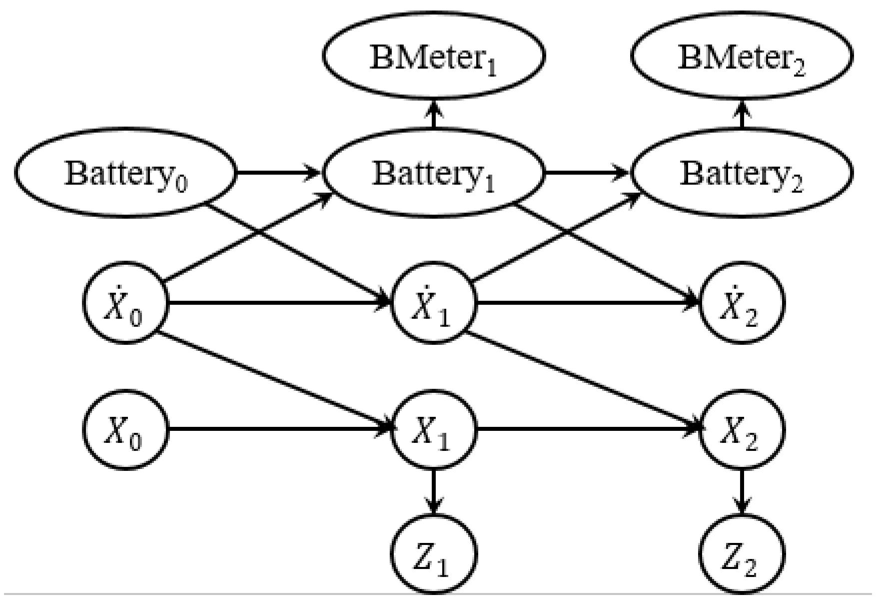
Dynamic Bayesian Networks

- A dynamic Bayesian network (DBN) over $X^1, \cdots, X^n$ is defined by
  - A Bayesian network BN_0 over $X_0^1, \cdots, X_0^n$
  - A 2TBN $BN_{\rightarrow}$ over $X^1, \cdots, X^n$



DBN vs. HMM

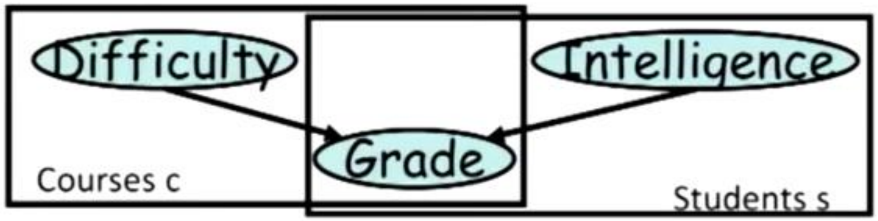
- Every HMM can be represented as DBN with a single state variable and a single evidence variable.
- Every discrete variable DBN can be represented as an HMM by combining all the state variables into a single state variable

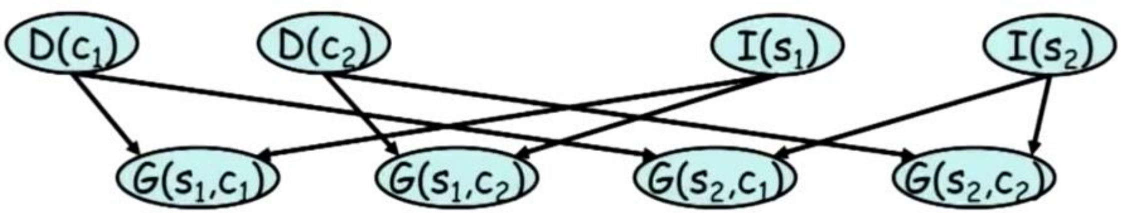


DBN is sparser.

Plate Models

- Plate models are the simplest object-relational models.





Inference in Temporal Models

- Filtering: $P(X_t|e_{1:t})$
  - Computing the belief state (the posterior distribution over the most recent state) given all evidence to date.
  - Also called state estimation
- Prediction: $P(X_{t+k}|e_{1:t}) \quad k > 0$
  - Computing the posterior distribution over the future state given all evidence to date.
- Smoothing: $P(X_k|e_{1:t}) \quad 0 \leq k < t$
  - Better estimate of past states, essential for learning
- Most likely explanation: $\mathrm{argmax}_{X_{1:t}} P(X_{1:t}|e_{1:t})$
  - Speech recognition
- Learning:
  - Learn the transition and sensor models from observations

Filtering

- Compute $P(X_t|e_{1:t})$
- Recursive estimation
  - Recursively perform prediction and update

$$\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t+1}) = \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t}, \mathbf{e}_{t+1})$$

$$= \alpha \, \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1}, \mathbf{e}_{1:t}) \, \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t})$$

$$= \alpha \, \underbrace{\mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1})}_{\text{Update}} \, \underbrace{\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t})}_{\text{Prediction}}$$

$$\boxed{\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t+1})} = \alpha \, \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1}) \sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{x}_t, \mathbf{e}_{1:t}) P(\mathbf{x}_t \mid \mathbf{e}_{1:t})$$

$$= \alpha \, \underbrace{\mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1})}_{\text{The sensor model}} \sum_{\mathbf{x}_t} \underbrace{\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{x}_t)}_{\text{The transition model}} \boxed{P(\mathbf{x}_t \mid \mathbf{e}_{1:t})}$$

Filtering

- View as message passing

$$\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t+1}) = \alpha \, \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1}) \sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{x}_t) P(\mathbf{x}_t \mid \mathbf{e}_{1:t})$$

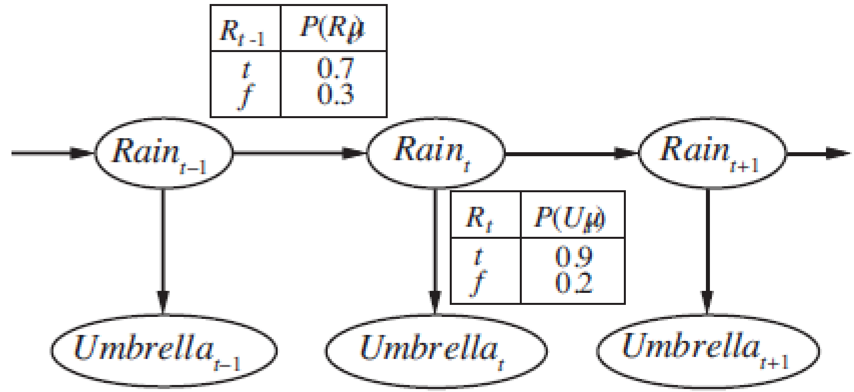
  - Consider $\mathbf{P}(\mathbf{X}_t \mid \mathbf{e}_{1:t})$ as a message $\mathbf{f}_{1:t}$
    - Propagated forward along the sequence
    - Modified by each transition
    - Updated by each new observation

$$\mathbf{f}_{1:t+1} = \alpha \, \textsc{Forward}(\mathbf{f}_{1:t}, \mathbf{e}_{t+1})$$

  - Constant time and space for each update

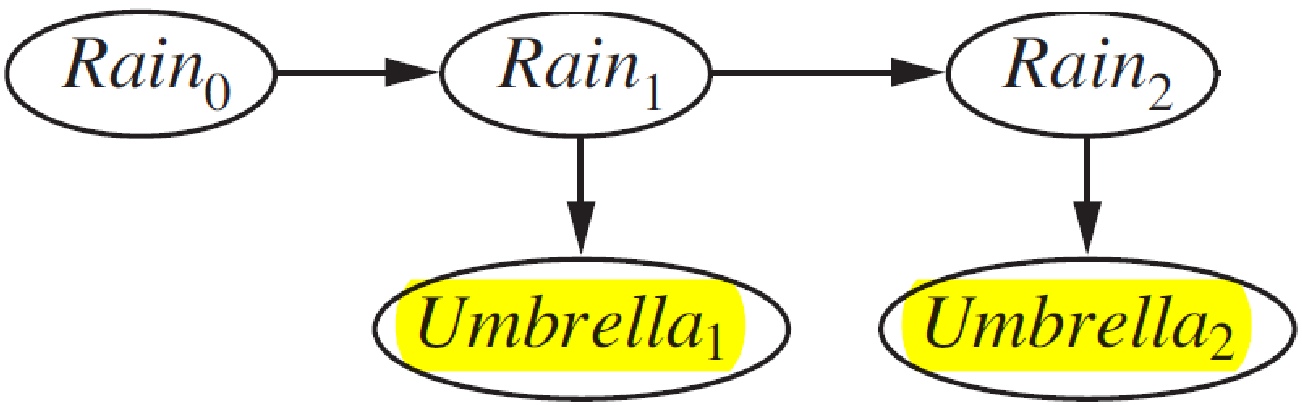
Filtering

- Example:
  - Compute $\mathbf{P}(R_2 \mid u_{1:2})$



$$\mathbf{P}(R_2 \mid u_1, u_2) = \alpha\, \mathbf{P}(u_2 \mid R_2)\mathbf{P}(R_2 \mid u_1) = \alpha\, \langle 0.9, 0.2\rangle \langle 0.627, 0.373\rangle$$

$$\mathbf{P}(R_0) = \langle 0.5, 0.5\rangle \qquad = \alpha\, \langle 0.565, 0.075\rangle \approx \langle 0.883, 0.117\rangle\ .$$



true *true*

$$\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t+1}) = \alpha\, \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1})\, \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t})$$

$$= \alpha\, \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1}) \sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{x}_t) P(\mathbf{x}_t \mid \mathbf{e}_{1:t})$$

Prediction

- Compute $P(X_{t+k}|e_{1:t})$
- Viewed as filtering without the addition of new evidence.
- Recursively computed by

$$\boxed{\mathbf{P}(\mathbf{X}_{t+k+1} \mid \mathbf{e}_{1:t})} = \sum_{\mathbf{x}_{t+k}} \underbrace{\mathbf{P}(\mathbf{X}_{t+k+1} \mid \mathbf{x}_{t+k})}_{\text{The transition model}} \boxed{P(\mathbf{x}_{t+k} \mid \mathbf{e}_{1:t})}$$

Smoothing

- Compute $P(X_k | e_{1:t})$ $0 \le k < t$
- Smoothing provides a better estimate of the state than was available at the time
- Learning requires smoothing, rather than filtering
- Forward-backward algorithm

$$
\begin{aligned}
\mathbf{P}(\mathbf{X}_k \mid \mathbf{e}_{1:t}) &= \mathbf{P}(\mathbf{X}_k \mid \mathbf{e}_{1:k}, \mathbf{e}_{k+1:t}) \\
&= \alpha\, \mathbf{P}(\mathbf{X}_k \mid \mathbf{e}_{1:k}) \mathbf{P}(\mathbf{e}_{k+1:t} \mid \mathbf{X}_k, \mathbf{e}_{1:k}) \\
&= \alpha\, \underbrace{\mathbf{P}(\mathbf{X}_k \mid \mathbf{e}_{1:k})}_{\substack{\text{Filtering} \\ \text{Forward}}} \underbrace{\mathbf{P}(\mathbf{e}_{k+1:t} \mid \mathbf{X}_k)}_{\text{Backward}} \\
&= \alpha\, \mathbf{f}_{1:k} \times \mathbf{b}_{k+1:t}
\end{aligned}
$$

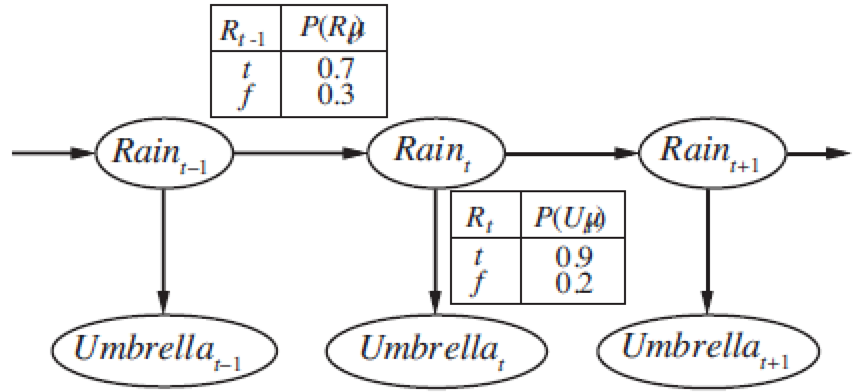
Smoothing

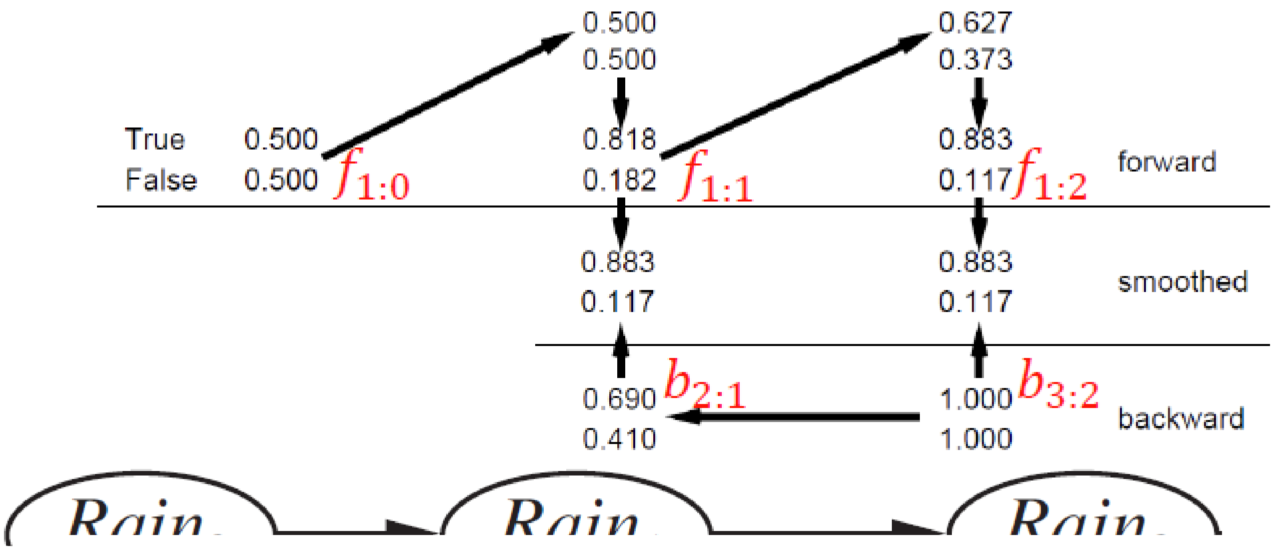
- $$\begin{aligned}\mathbf{P}(\mathbf{X}_k \mid \mathbf{e}_{1:t}) &= \alpha\,\mathbf{P}(\mathbf{X}_k \mid \mathbf{e}_{1:k})\mathbf{P}(\mathbf{e}_{k+1:t} \mid \mathbf{X}_k) \\ &= \alpha\,\mathbf{f}_{1:k} \times \mathbf{b}_{k+1:t}\end{aligned}$$
- Backward message: $\mathbf{b}_{k+1:t} = \mathbf{P}(\mathbf{e}_{k+1:t} \mid \mathbf{X}_k) = \textsc{Backward}(\mathbf{b}_{k+2:t}, \mathbf{e}_{k+1})$

$$\begin{aligned}\boxed{\mathbf{P}(\mathbf{e}_{k+1:t} \mid \mathbf{X}_k)} &= \sum_{\mathbf{x}_{k+1}} \mathbf{P}(\mathbf{e}_{k+1:t} \mid \mathbf{X}_k, \mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1} \mid \mathbf{X}_k) \\ &= \sum_{\mathbf{x}_{k+1}} P(\mathbf{e}_{k+1:t} \mid \mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1} \mid \mathbf{X}_k) \\ &= \sum_{\mathbf{x}_{k+1}} P(\mathbf{e}_{k+1}, \mathbf{e}_{k+2:t} \mid \mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1} \mid \mathbf{X}_k) \\ &= \sum_{\mathbf{x}_{k+1}} \underbrace{P(\mathbf{e}_{k+1} \mid \mathbf{x}_{k+1})}_{\text{Sensor model}}\boxed{P(\mathbf{e}_{k+2:t} \mid \mathbf{x}_{k+1})}\underbrace{\mathbf{P}(\mathbf{x}_{k+1} \mid \mathbf{X}_k)}_{\text{Transition model}}\end{aligned}$$

Smoothing

- Example:
  - Compute $\mathbf{P}(R_1 \mid u_1, u_2)$





$$\mathbf{P}(R_1 \mid u_1, u_2) = \alpha \, \langle 0.818, 0.182 \rangle \times \langle 0.69, 0.41 \rangle \approx \langle 0.883, 0.117 \rangle$$

true *true*

$$\mathbf{P}(\mathbf{X}_k \mid \mathbf{e}_{1:t}) \;=\; \alpha \, \mathbf{P}(\mathbf{X}_k \mid \mathbf{e}_{1:k}) \mathbf{P}(\mathbf{e}_{k+1:t} \mid \mathbf{X}_k) \;=\; \alpha \, \mathbf{f}_{1:k} \times \mathbf{b}_{k+1:t}$$

The Most Likely Sequence

- $\text{argmax}_{X_{1:t}} P(X_{1:t}|e_{1:t})$
- Viterbi algorithm

$$\boxed{\max_{\mathbf{x}_1 \ldots \mathbf{x}_t} \mathbf{P}(\mathbf{x}_1, \ldots, \mathbf{x}_t, \mathbf{X}_{t+1} \mid \mathbf{e}_{1:t+1})}$$

$$= \alpha \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1}) \max_{\mathbf{x}_t} \left(\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{x}_t) \boxed{\max_{\mathbf{x}_1 \ldots \mathbf{x}_{t-1}} P(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{x}_t \mid \mathbf{e}_{1:t})} \right)$$

- Identical to filtering if:
  - Replace the forward message $\mathbf{f}_{1:t} = \mathbf{P}(\mathbf{X}_t \mid \mathbf{e}_{1:t})$ by

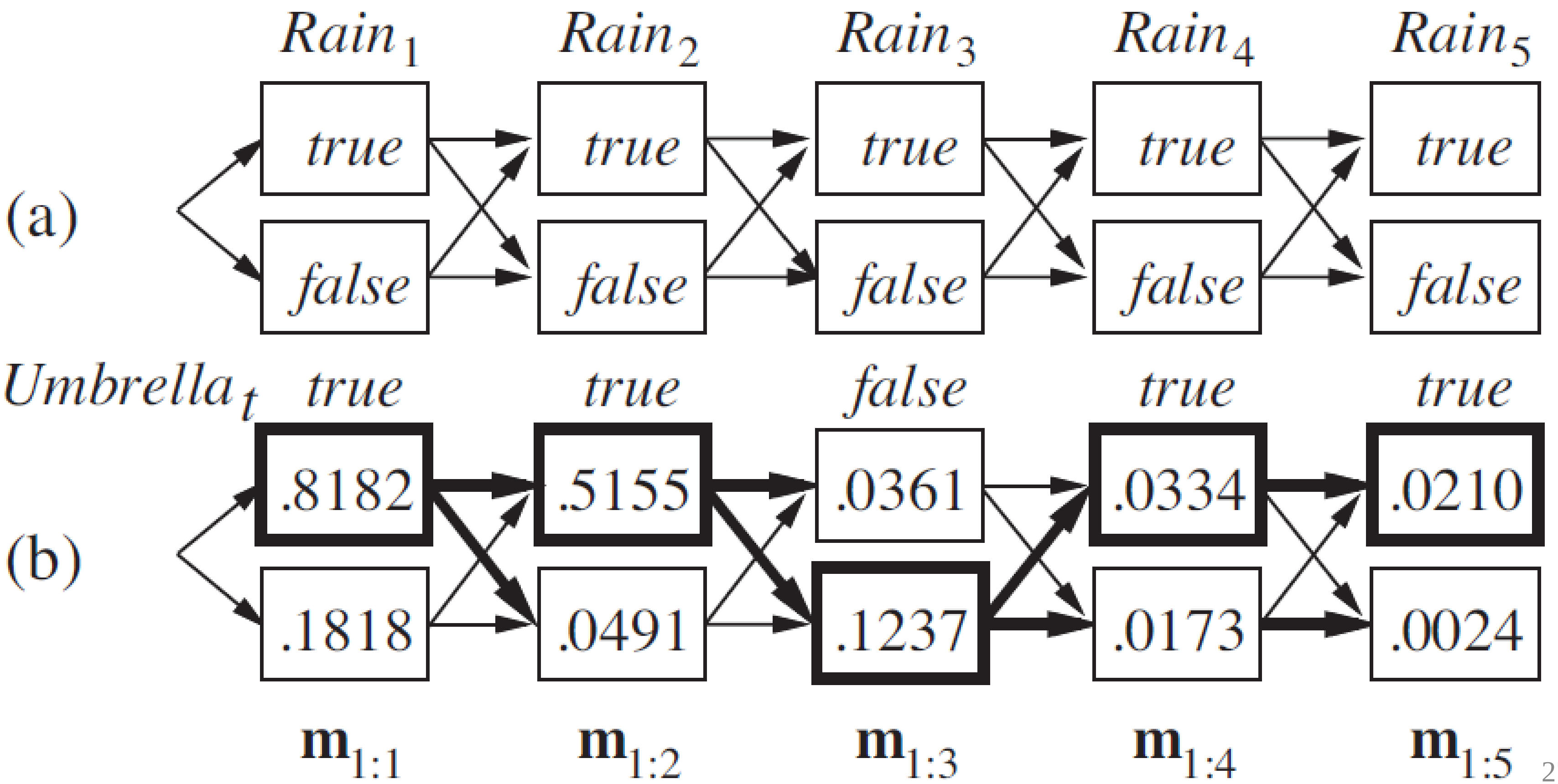
$$\mathbf{m}_{1:t} = \max_{\mathbf{x}_1 \ldots \mathbf{x}_{t-1}} \mathbf{P}(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{X}_t \mid \mathbf{e}_{1:t})$$

  - Replace the summation over $\mathbf{x}_t$ by the maximization over $\mathbf{x}_t$

$$\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t+1}) = \alpha \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1}) \sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{x}_t) P(\mathbf{x}_t \mid \mathbf{e}_{1:t})$$

The Most Likely Sequence

- Example
  - Given the umbrella sequence [*true*, *true*, *false*, *true*, *true*]
  - Find the weather sequence most likely to explain this

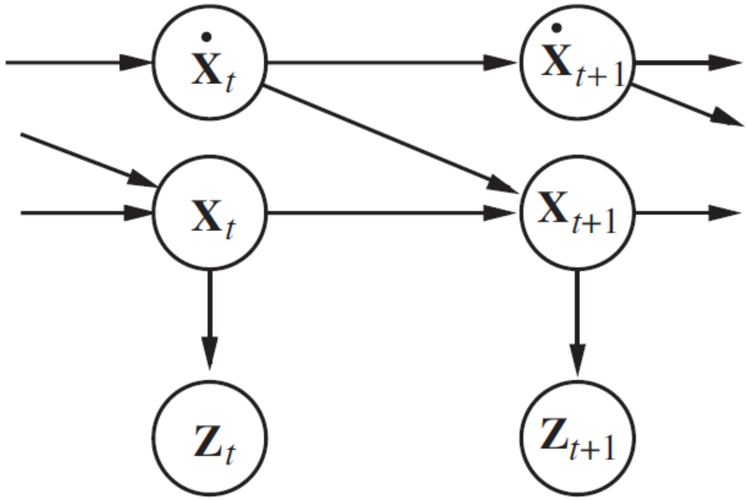


Kalman Filtering

- A dynamic Bayesian network
  - Transition model
  - Sensor model

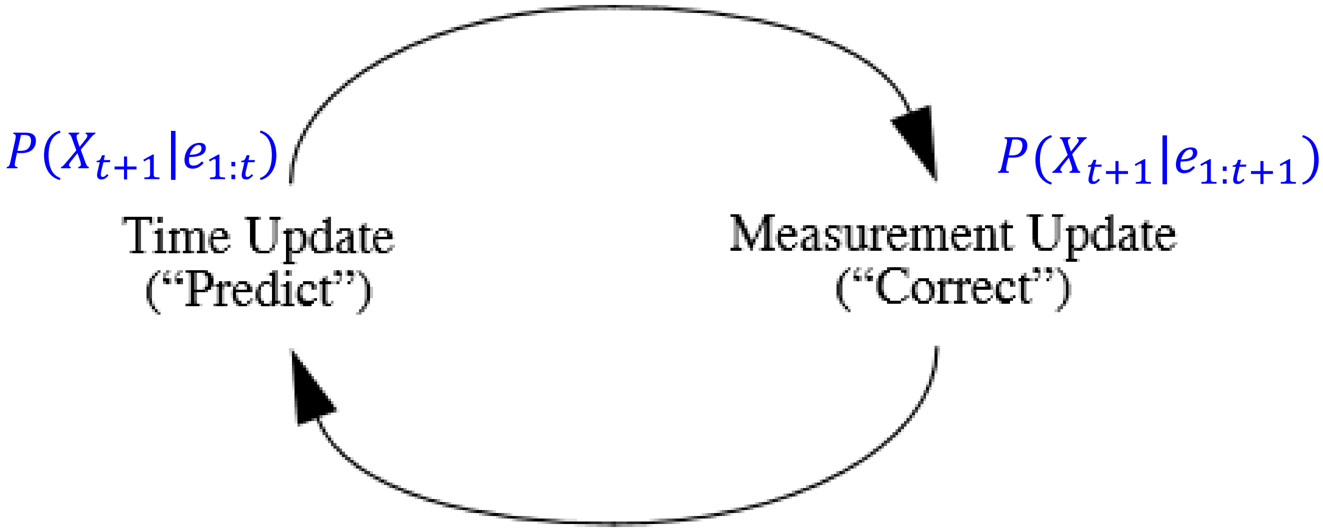
Linear Gaussian Distributions

  - The next state is a linear function of the current state, plus some Gaussian noise



Kalman Filtering

- Inference



$$\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t+1}) = \alpha \, \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1}) \sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{x}_t) P(\mathbf{x}_t \mid \mathbf{e}_{1:t})$$

Kalman Filtering

- Inference
  - Predict: (time update $X_t \rightarrow X_{t+1}$)

$$P(X_{t+1}|e_{1:t}) = \int P(X_{t+1}|x_t)P(x_t|e_{1:t})dx_t$$

  - Update: (measurement update $e_{t+1} \rightarrow X_{t+1}$)

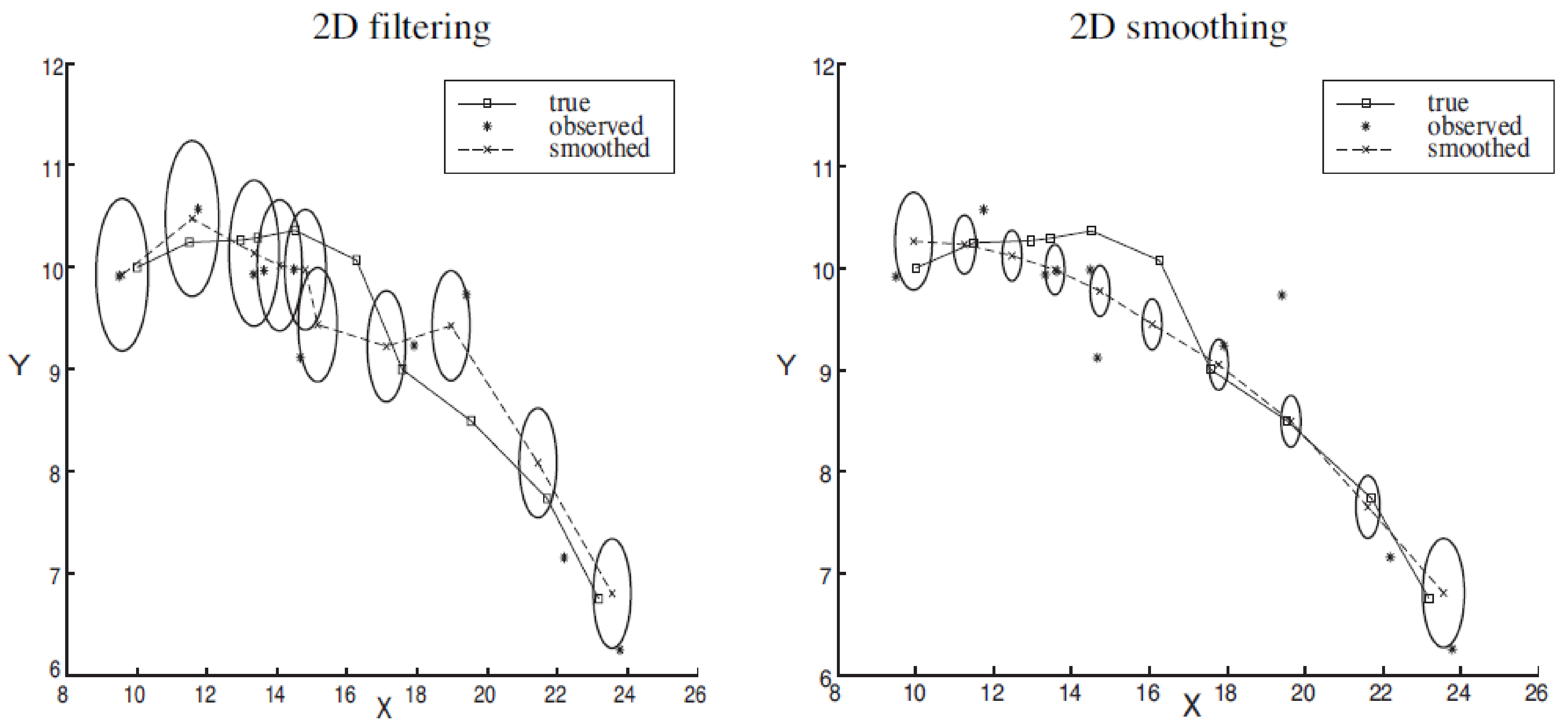
$$P(X_{t+1}|e_{1:t+1}) = \alpha P(e_{t+1}|X_{t+1})P(X_{t+1}|e_{1:t})$$

$$\begin{aligned}\mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t+1}) &= \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t}, \mathbf{e}_{t+1}) \\ &= \alpha\, \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1}, \mathbf{e}_{1:t})\, \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t}) \\ &= \alpha\, \mathbf{P}(\mathbf{e}_{t+1} \mid \mathbf{X}_{t+1})\, \mathbf{P}(\mathbf{X}_{t+1} \mid \mathbf{e}_{1:t})\end{aligned}$$

  - Start with a Gaussian prior $f_{1:0} = P(X_0) = N(\mu_0, \Sigma_0)$, filtering with a linear Gaussian model produces a Gaussian state distribution for all time.

Kalman Filtering

- Example:
  - A robot moving in a plane with constant but noisy velocity, where only position is observed.



Kalman Filtering

- Linear Gaussian assumption is too strong in real application
- Extensions:
  - Extended Kalman filter
  - Unscented Kalman filter

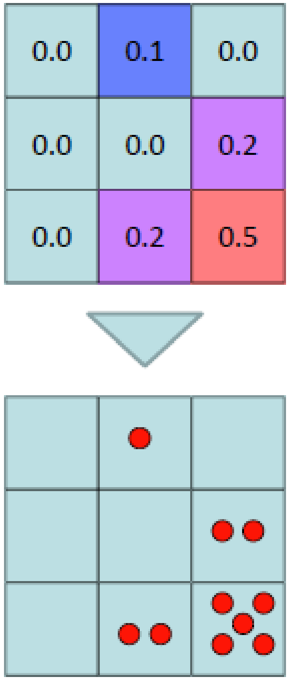
Inference in DBNs

- Exact inference
  - Variable Elimination

- Approximate inference
  - Particle filter

Particle Filtering

- Problem for filtering:
  - The state distributions grow without bound over time. That is, |X| may be too big to store.
- Solution: approximate inference
  - Use a set of samples as the forward message, i.e. use the samples as approximate representation of the current state distribution
  - Focus the set of samples on the high-probability regions of the state space, throw away low-probability samples.



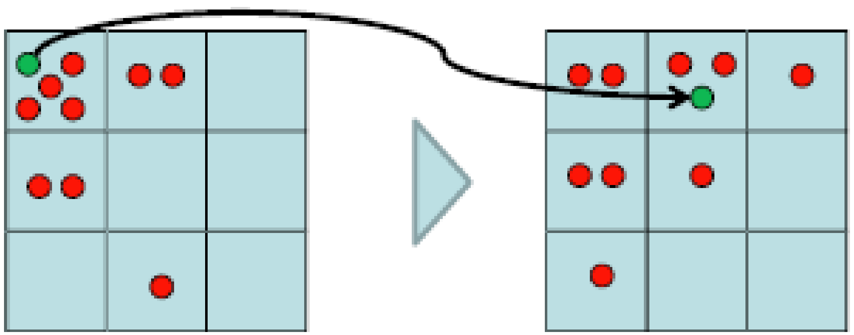
Particle Filtering

- Track samples of X, not all values
- Samples are called particles
- P(x) approximated by number of particles with value x
- More particles, more accuracy

Particle Filtering

- Step 1: Transition (time update)
  - Each particle is moved by sampling its next position from the transition model

$$x_{t+1} = sample(P(X_{t+1}|x_t))$$ ← Forward sampling

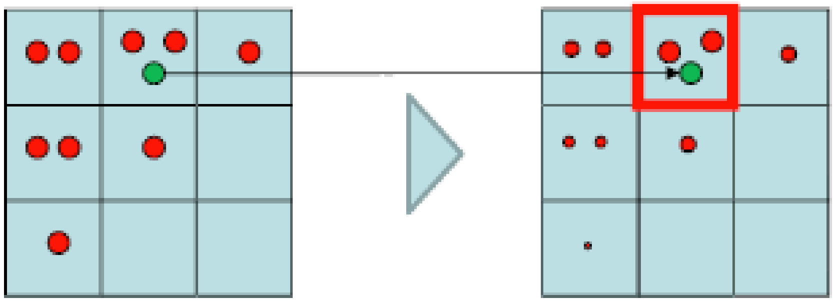


Particle Filtering

- Step 2: Weighting (Measurement update)
  - Fix sample observation (evidence)
  - Down weight samples based on the evidence ⟸ Likelihood weighting

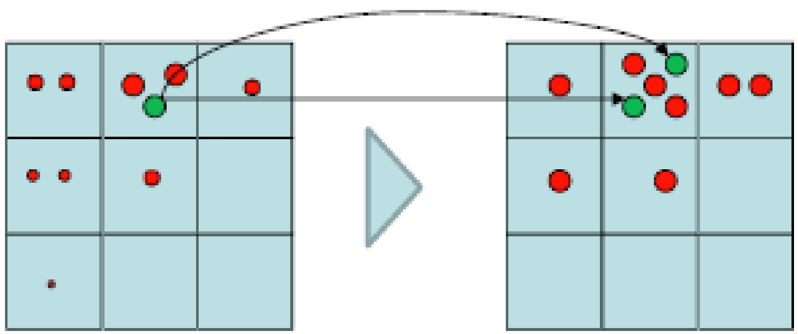
$$w(x_{t+1}) = P(e_{t+1}|x_{t+1}))$$

  - The probabilities don't sum to one since all have been down weighted (in fact they now sum to (N times) an approximation of P(e)



Particle Filtering

- Step 3: Resampling
  - Rather than tracking weighted samples, we resample from our weighted sample distribution
  - This is equivalent to renormalizing the distribution



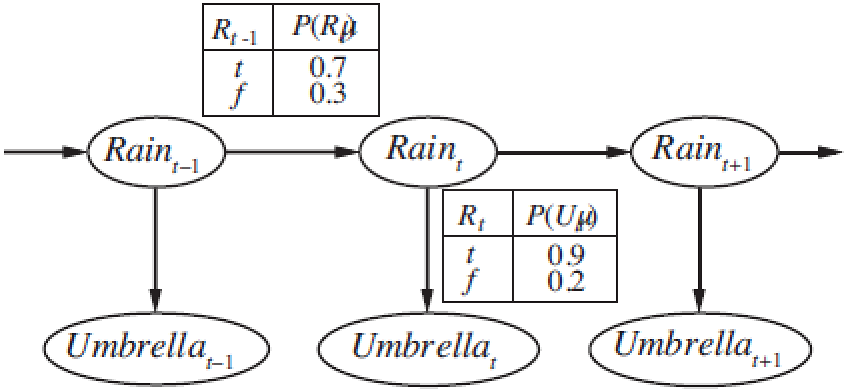
Particle Filtering

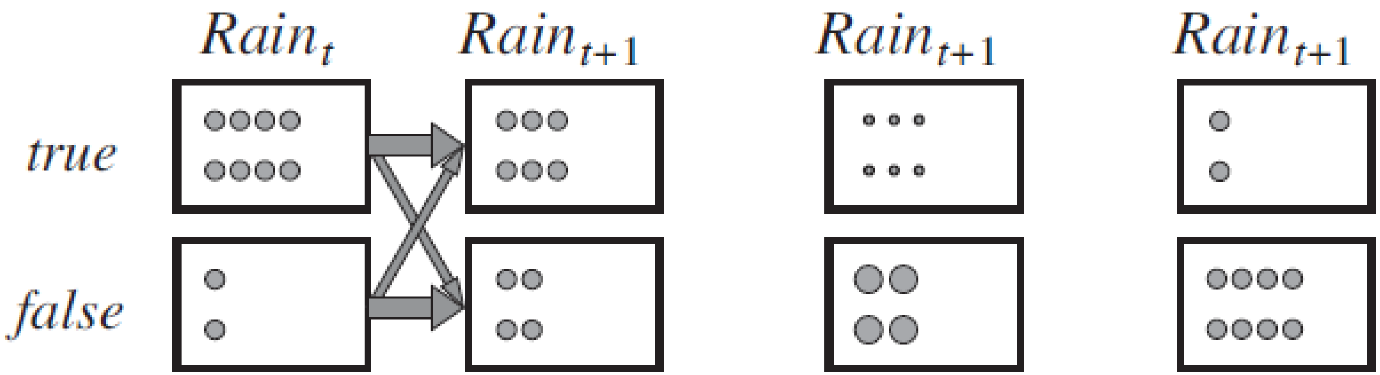
function PARTICLE-FILTERING($\mathbf{e}$, N, dbn) **returns** a set of samples for the next time step
  inputs: $\mathbf{e}$, the new incoming evidence
    N, the number of samples to be maintained
    dbn, a DBN with prior $\mathbf{P}(\mathbf{X}_0)$, transition model $\mathbf{P}(\mathbf{X}_1|\mathbf{X}_0)$, sensor model $\mathbf{P}(\mathbf{E}_1|\mathbf{X}_1)$
  persistent: S, a vector of samples of size N, initially generated from $\mathbf{P}(\mathbf{X}_0)$
  local variables: W, a vector of weights of size N

  for i = 1 to N **do**
    $S[i] \leftarrow$ sample from $\mathbf{P}(\mathbf{X}_1 \mid \mathbf{X}_0 = S[i])$ /* step 1 */
    $W[i] \leftarrow \mathbf{P}(\mathbf{e} \mid \mathbf{X}_1 = S[i])$ /* step 2 */
  $S \leftarrow$ WEIGHTED-SAMPLE-WITH-REPLACEMENT(N, S, W) /* step 3 */
  return S

Particle Filtering

- Example:

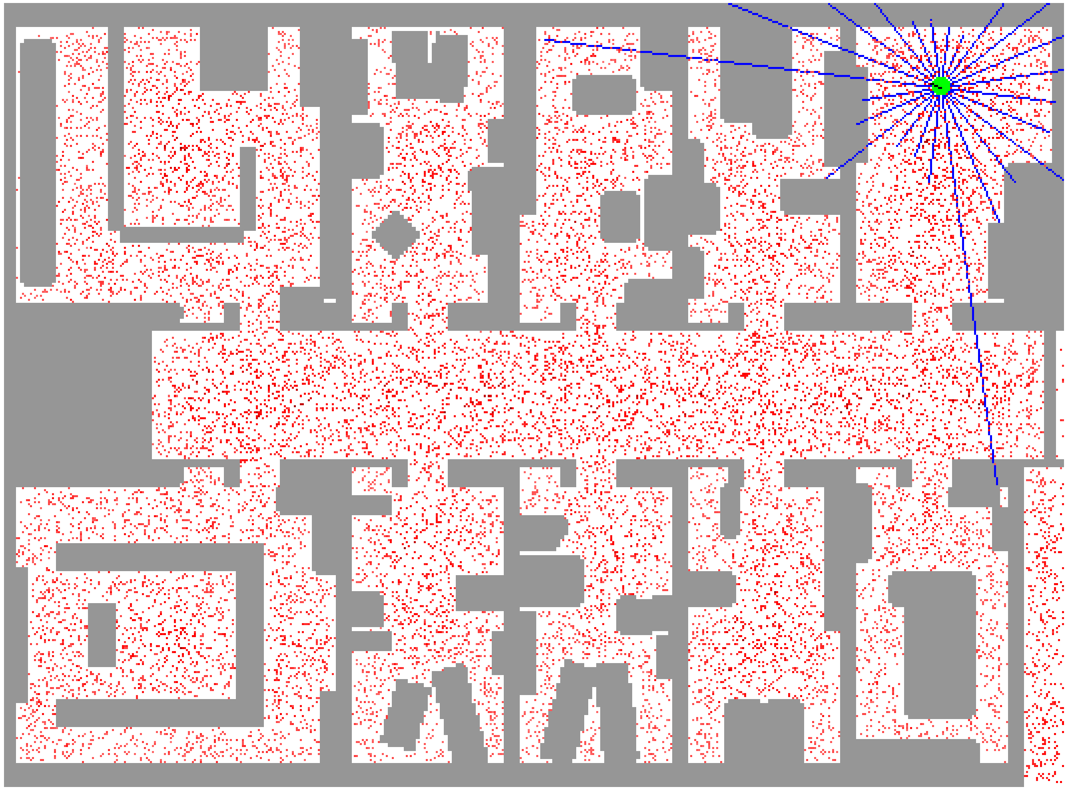




(a) Propagate (b) Weight (c) Resample

Particle Filtering

- Example: Robot Localization

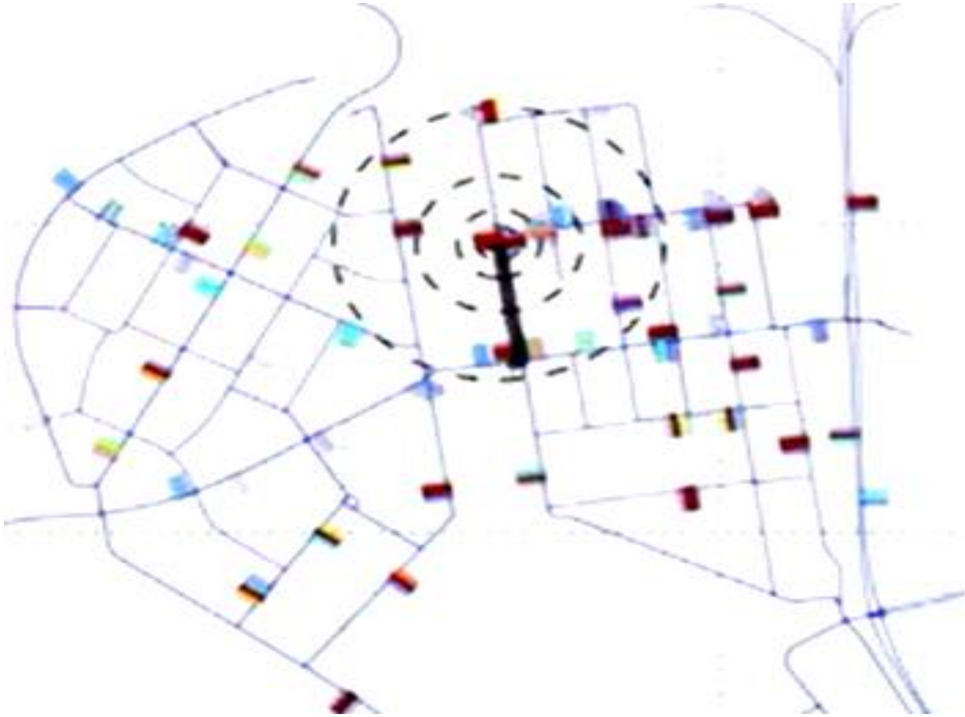


http://www.cs.washington.edu/research/rse-lab/projects/mcl

D. Fox, S. Thrun, W. Burgard, and F. Dellaert, Particle Filters for Mobile Robot Localization

Particle Filtering

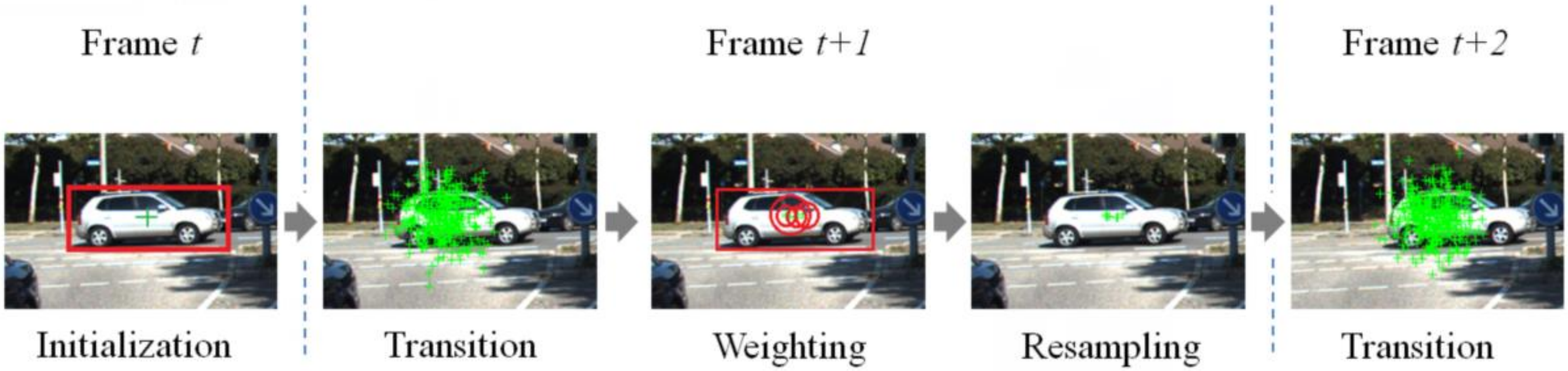
- Example: Map-Based Probabilistic Visual Self-Localization



http://www.cs.toronto.edu/~mbrubake/projects/map/
M. A. Brubaker, A. Geiger and R. Urtasun, Map-Based Probabilistic Visual Self-Localization, TPAMI 2016

Particle Filtering

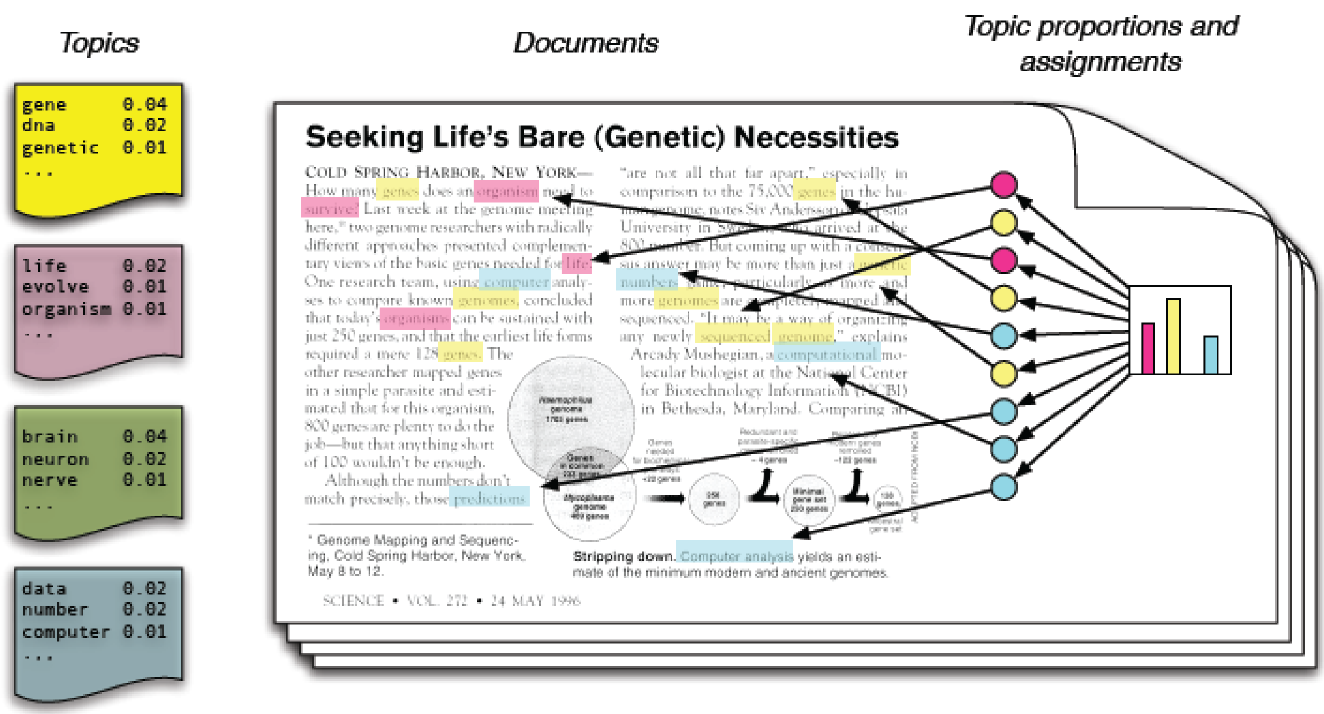
- Example: Object Tracking



[2] S. Song, Z. Xiang, and J. Liu, Object tracking with 3D LIDAR via multi-task sparse learning, ICMA 2015.
[1] X. Mei and H. Ling, Robust Visual Tracking using l1 Minimization, ICCV 2009

Latent Dirichlet Allocation

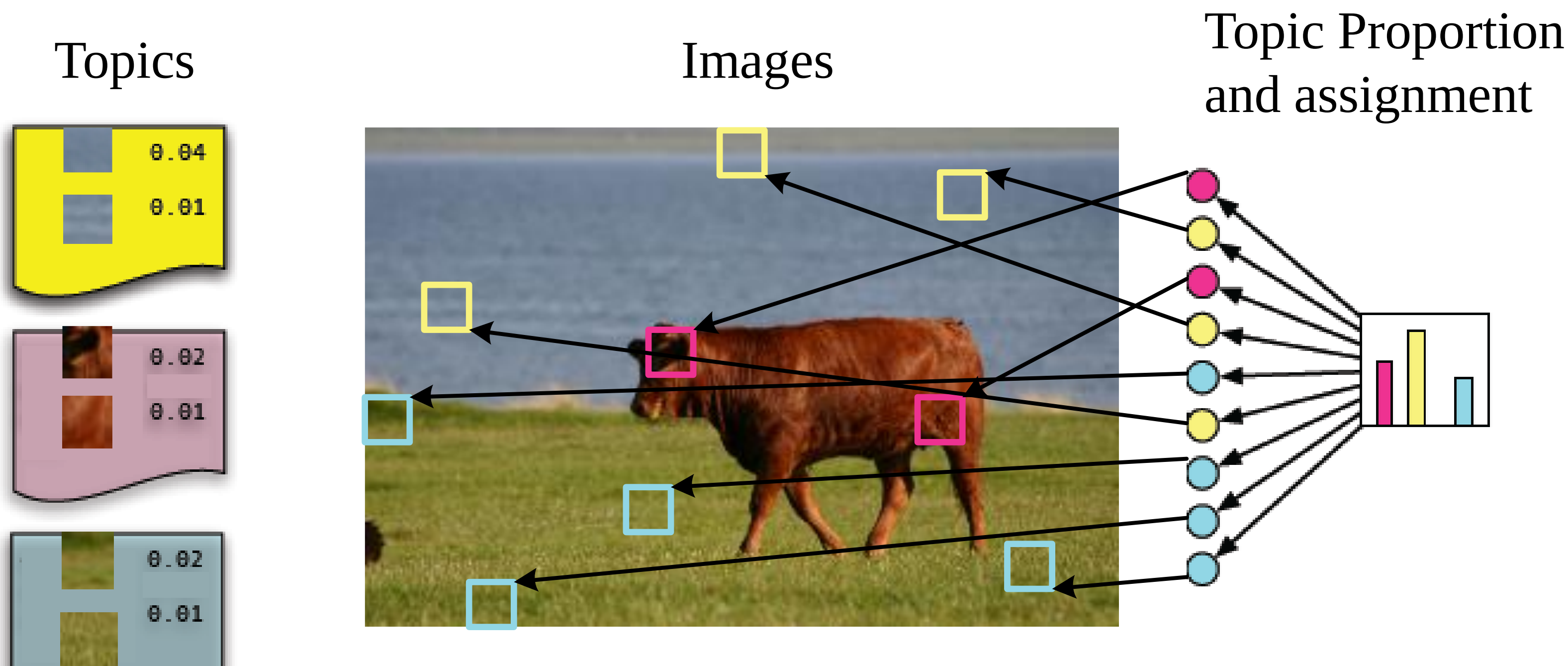
- Example: Information retrieval
  - Each topic is a distribution over words
  - Each document is a random mixture of topics
  - Each word is draw from one of those topics



https://www.cs.princeton.edu/~blei/topicmodeling.html

Latent Dirichlet Allocation

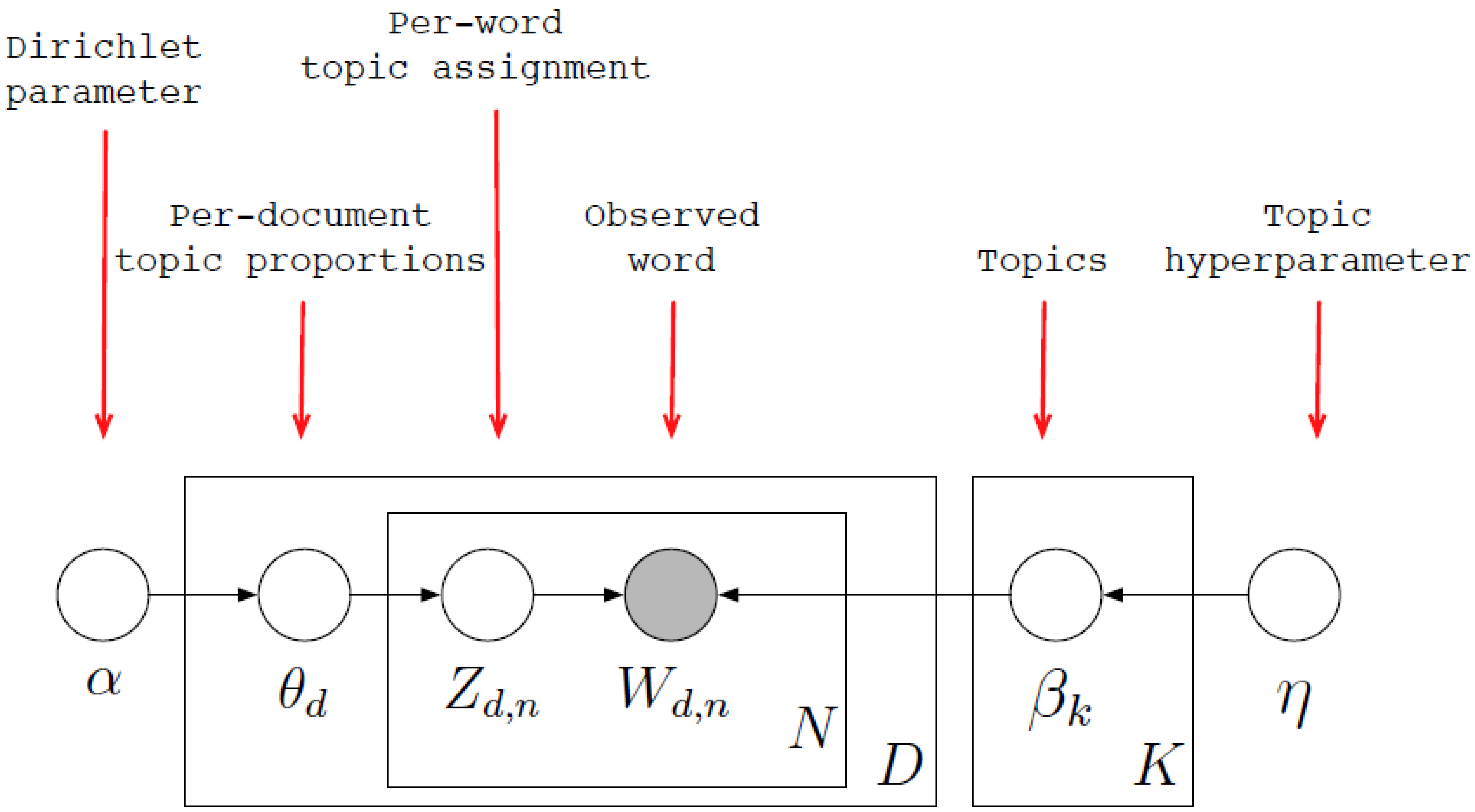
- Example: Image Parsing
  - Each topic is a distribution over visual words
  - Each image is a random mixture of topics
  - Each visual word is draw from one of those topics



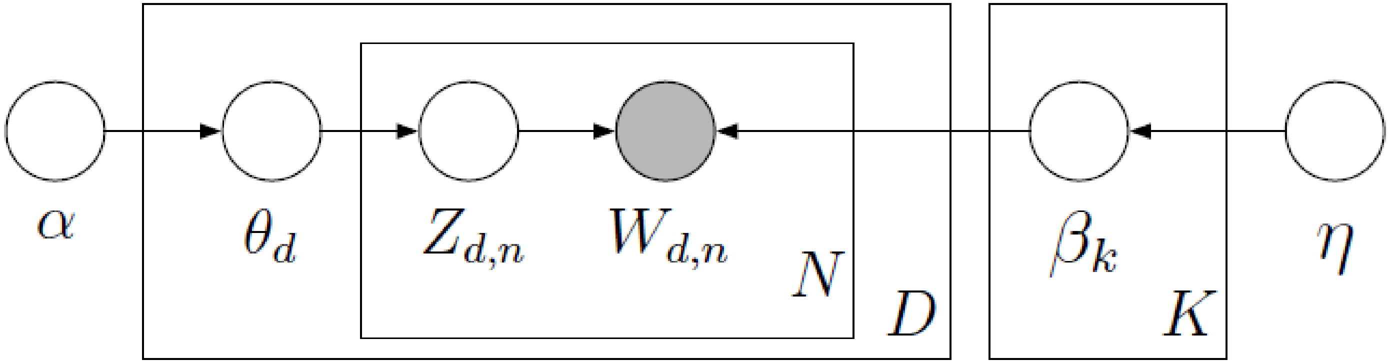
L. Li, R. Socher, and L. Fei-Fei, Towards Total Scene Understanding: Classification, Annotation and Segmentation in an Automatic Framework, CVPR 2009

Latent Dirichlet Allocation

- LDA Modeling



Latent Dirichlet Allocation

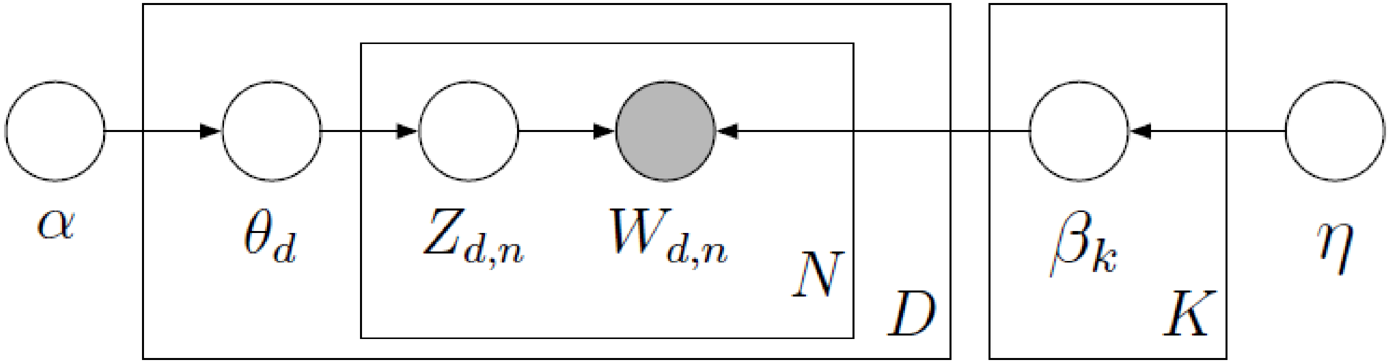


- Joint probability

$$P(Z, W, \theta, \beta, \alpha, \eta)$$
$$= P(\alpha)P(\eta)\prod_{d=1}^{D} P(\theta_d|\alpha)\prod_{n=1}^{N} P(Z_{d,n}|\theta_d)\prod_{k=1}^{K} P(\beta_k|\eta)P(W_{d,n}|\beta_k, Z_{d,n})$$

Latent Dirichlet Allocation

- Inference:

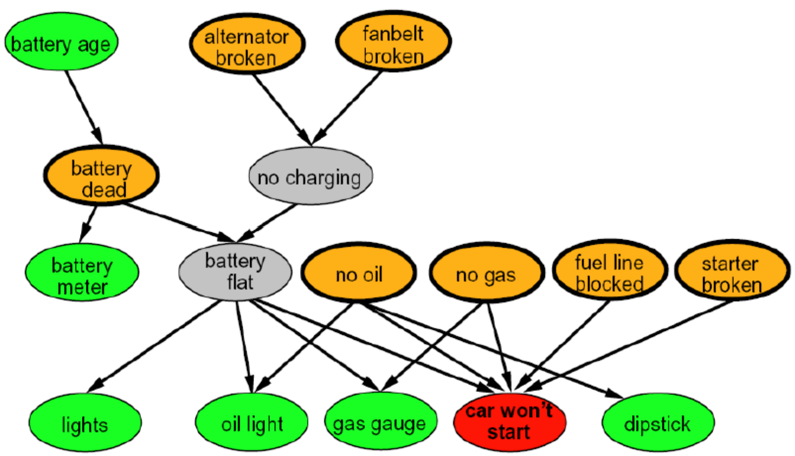


  - Maximize the posterior probability $P(Z, \theta, \beta | W, \alpha, \eta)$

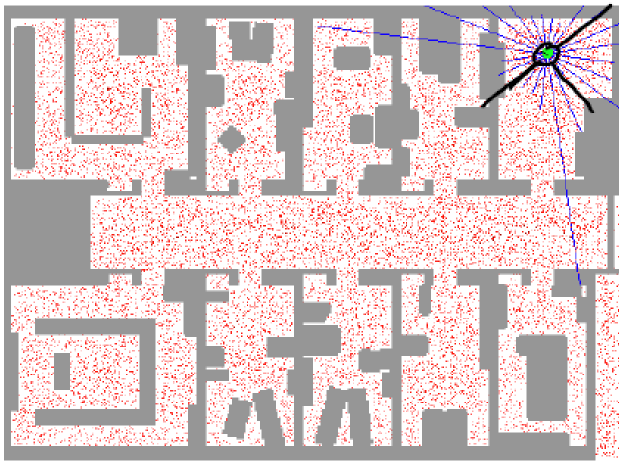
  - Collapsed Gibbs sampling

Bayesian Network Applications

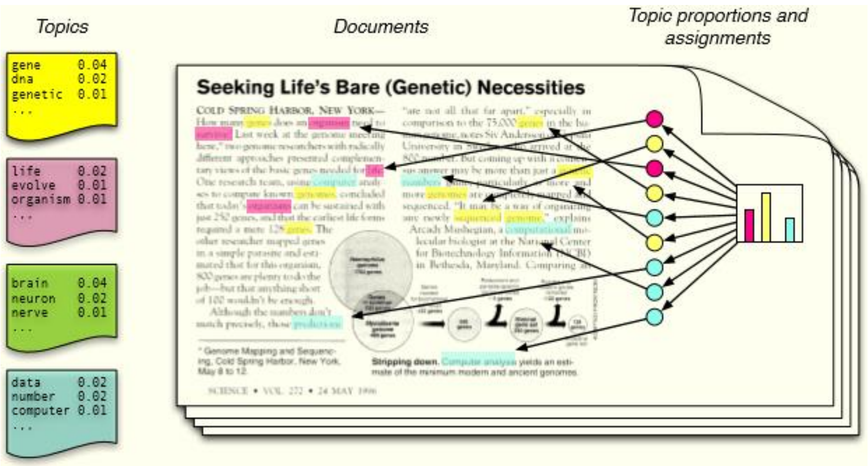
- Car Diagnosis



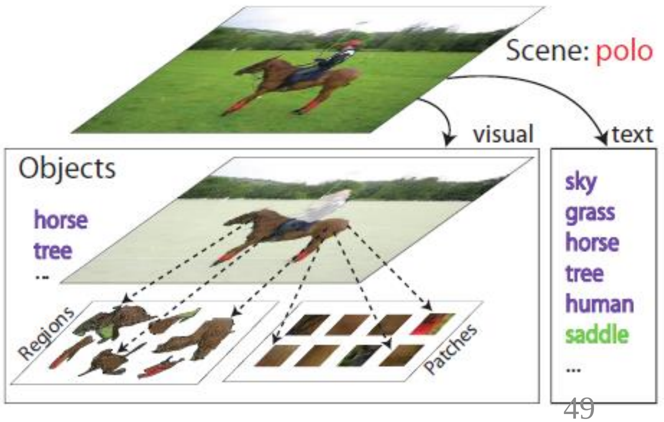
- Robot Localization



- Information Retrieval



- Scene Parsing



Assignments

- Reading assignment:
  - Ch. 15

- Homework 4
- Project 2