

# COMP90051 Statistical Machine Learning

Semester 2, 2018

Lecturer: Ben Rubinstein

*20. Just for fun.*

Hidden Markov Models.

Message passing



THE UNIVERSITY OF  
MELBOURNE

# Looking back...

- Representation of joint distributions
- Conditional/marginal independence
  - \* Directed vs undirected
- Probabilistic inference
  - \* Computing other distributions from joint
- Statistical inference
  - \* Learn parameters from (missing) data
- Today: putting these all into practice...



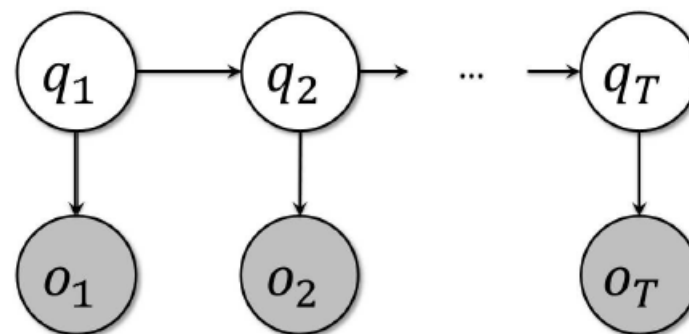
# Hidden Markov Models

*Model of choice for sequential data. A form of clustering (or dimensionality reduction) for discrete time series.*

# HMM Formulation

- Formulated as directed PGM

\* therefore joint expressed as



$$P(\mathbf{o}, \mathbf{q}) = P(q_1)P(o_1|q_1) \prod_{i=2}^T P(q_i|q_{i-1})P(o_i|q_i)$$

\* **bold** variables are shorthand for vector of  $T$  values

- Parameters (for *homogenous* HMM)

$$\begin{array}{ll} A = \{a_{ij}\} & \text{transition probability matrix; } \forall i : \sum_j a_{ij} = 1 \\ B = \{b_i(o_k)\} & \text{output probability matrix; } \forall i : \sum_k b_i(o_k) = 1 \\ \Pi = \{\pi_i\} & \text{the initial state distribution; } \sum_i \pi_i = 1 \end{array}$$

# Fundamental HMM Tasks

HMM Task	PGM Task
<b>Evaluation.</b> Given an HMM $\mu$ and observation sequence $\mathbf{o}$ , determine likelihood $\Pr(\mathbf{o} \mu)$	Probabilistic inference
<b>Decoding.</b> Given an HMM $\mu$ and observation sequence $\mathbf{o}$ , determine most probable hidden state sequence $\mathbf{q}$	MAP point estimate
<b>Learning.</b> Given an observation sequence $\mathbf{o}$ and set of states, learn parameters $A, B, \Pi$	Statistical inference

# “Evaluation” a.k.a. marginalisation

- Compute prob. of observations  $\mathbf{o}$  by summing out  $\mathbf{q}$

$$\begin{aligned} P(\mathbf{o}|\mu) &= \sum_{\mathbf{q}} P(\mathbf{o}, \mathbf{q}|\mu) \\ &= \sum_{q_1} \sum_{q_2} \dots \sum_{q_T} P(q_1)P(o_1|q_1)P(q_2|q_1)P(o_2|q_2) \dots P(q_T|q_{T-1})P(o_T|q_T) \end{aligned}$$

- Make this more efficient by moving the sums

$$P(\mathbf{o}|\mu) = \sum_{q_1} P(q_1)P(o_1|q_1) \sum_{q_2} P(q_2|q_1)P(o_2|q_2) \dots \sum_{q_T} P(q_T|q_{T-1})P(o_T|q_T)$$

- Déjà vu? Maybe we could do var. elimination...

# Elimination = Backward Algorithm

$$P(\mathbf{o}|\mu) = \sum_{q_1} P(q_1)P(o_1|q_1) \sum_{q_2} P(q_2|q_1)P(o_2|q_2) \dots \sum_{q_T} P(q_T|q_{T-1})P(o_T|q_T)$$

Eliminate  $q_T$

$$m_{T \rightarrow T-1}(q_{T-1})$$

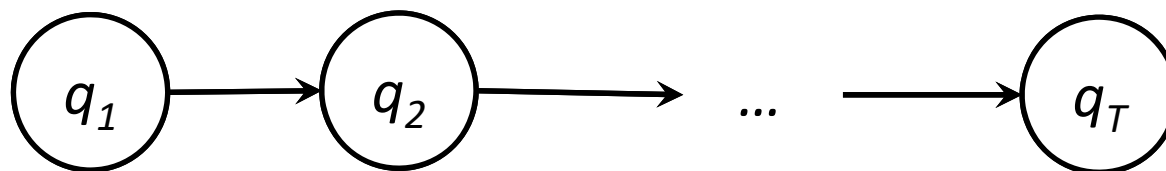
...

Eliminate  $q_2$

$$m_{2 \rightarrow 1}(q_1)$$

“Eliminate”  $q_1$

$$P(\mathbf{o}|\mu) = \sum_{q_1} P(q_1)P(o_1|q_1)m_{2 \rightarrow 1}(q_1)$$



# Elimination = Forward Algorithm

$$P(\mathbf{o}|\mu) = \sum_{q_T} P(o_T|q_T) \sum_{q_{T-1}} P(q_T|q_{T-1}) P(o_T|q_T) \dots \sum_{q_1} P(q_2|q_1) P(q_1) P(o_1|q_1)$$

Eliminate  $q_1$

...

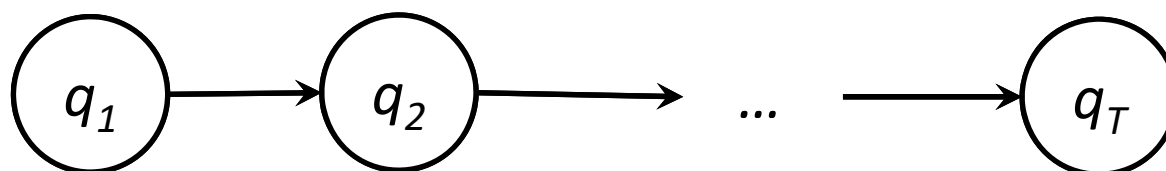
Eliminate  $q_{T-1}$

“Eliminate”  $q_T$

$m_{1 \rightarrow 2}(q_2)$

$m_{T-1 \rightarrow T}(q_T)$

$$P(\mathbf{o}|\mu) = \sum_{q_1} P(o_T|q_T) m_{T-1 \rightarrow T}(q_T)$$





# Forward-Backward

- Both algorithms are just *variable elimination* using different orderings
  - \*  $q_T \dots q_1 \rightarrow$  backward algorithm
  - \*  $q_1 \dots q_T \rightarrow$  forward algorithm
  - \* both have time complexity  $O(Tl^2)$  where  $l$  is the label set
- Can use either to compute  $P(\mathbf{o})$ 
  - \* but even better, can use the  $m$  values to compute marginals (and pairwise marginals over  $q_i, q_{i+1}$ )

$$P(q_i | \mathbf{o}) = \frac{1}{P(\mathbf{o})} \underset{\text{forward}}{m_{i-1 \rightarrow i}(q_i)} P(o_i | q_i) \underset{\text{backward}}{m_{i+1 \rightarrow i}(q_i)}$$

# Statistical Inference (Learning)

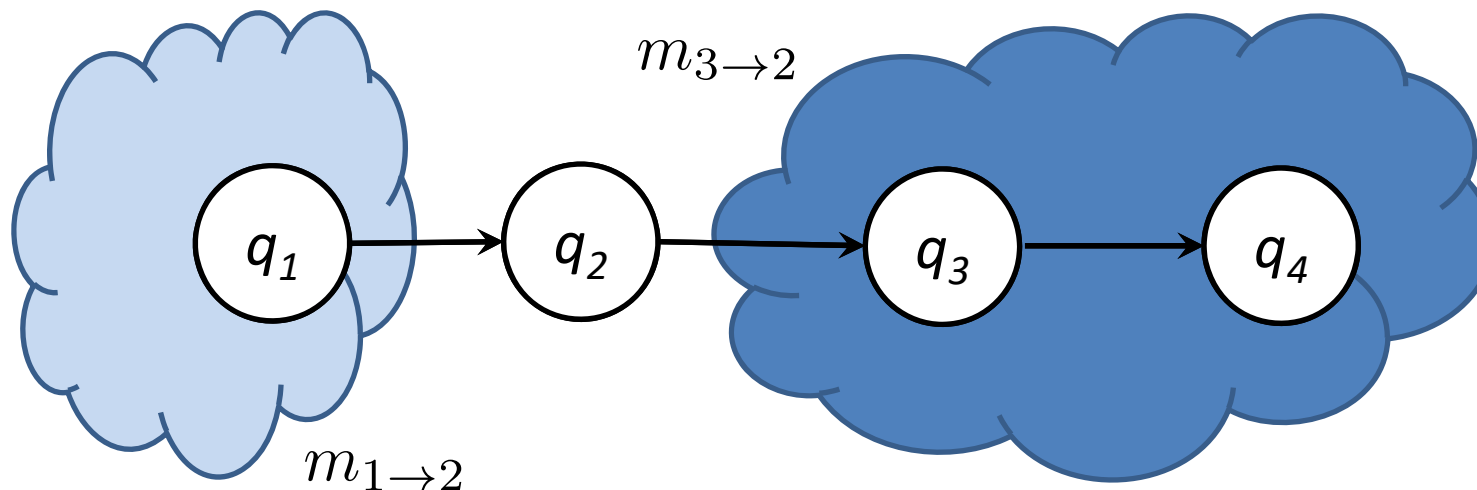
- Learn parameters  $\boldsymbol{\mu} = (A, B, \boldsymbol{\pi})$ , given observation sequence  $\mathbf{o}$
- Called “**Baum Welch**” algorithm which uses **EM** to approximate MLE,  $\operatorname{argmax}_{\boldsymbol{\mu}} P(\mathbf{o} | \boldsymbol{\mu})$ :
  1. initialise  $\boldsymbol{\mu}^1$ , let  $j=1$
  2. compute expected marginal distributions  $P(q_t | \mathbf{o}, \boldsymbol{\mu}^j)$  for all  $t$ ; and  $P(q_{t-1}, q_t | \mathbf{o}, \boldsymbol{\mu}^j)$  for  $t=2..T$
  3. fit model  $\boldsymbol{\mu}^{j+1}$  based on expectations
  4. repeat from step 2, with  $j=j+1$
- Expectations (2) computed using forward-backward

# Message Passing

*Sum-product algorithm for efficiently computing marginal distributions over trees. An extension of variable elimination algorithm.*

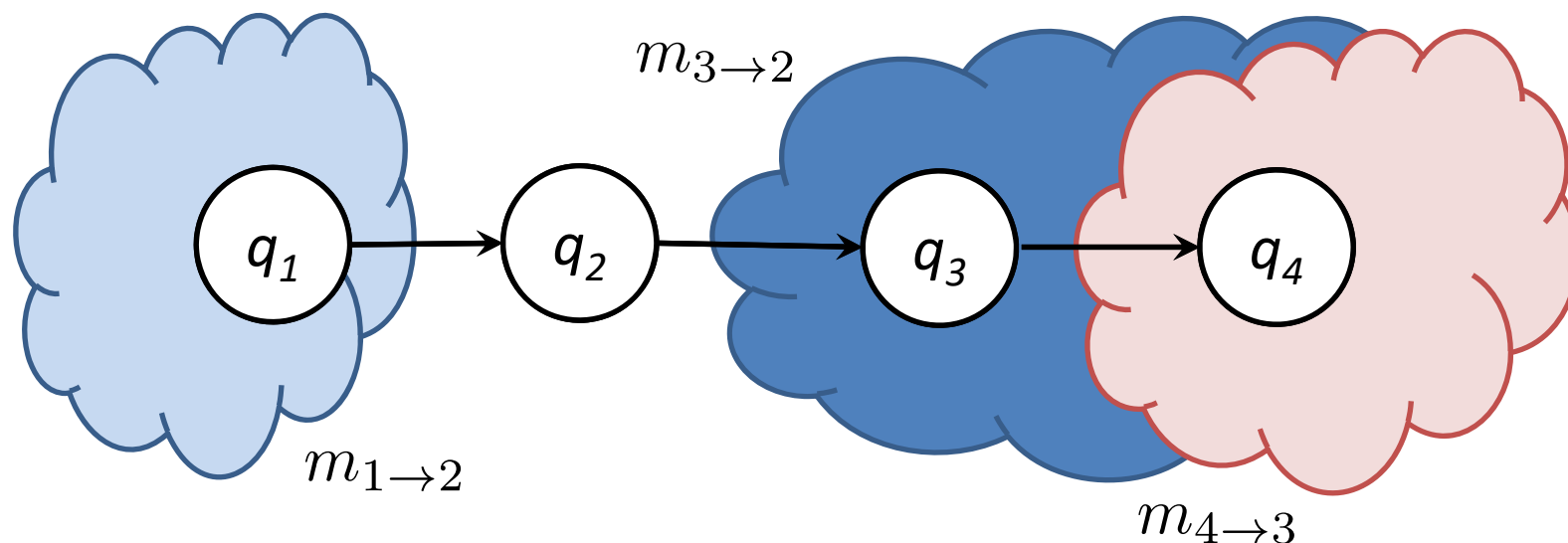
# Inference as message passing

- Each  $m$  can be considered as a **message** which summarises the effect of the rest of the graph on the current node marginal.
  - \* *Inference = passing messages between all nodes*



# Inference as message passing

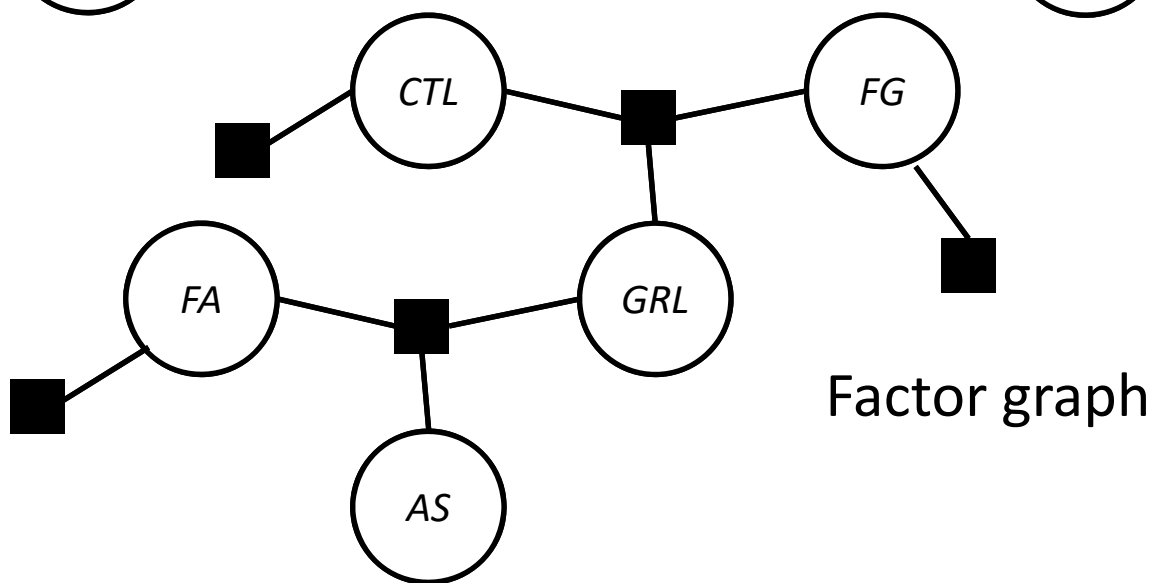
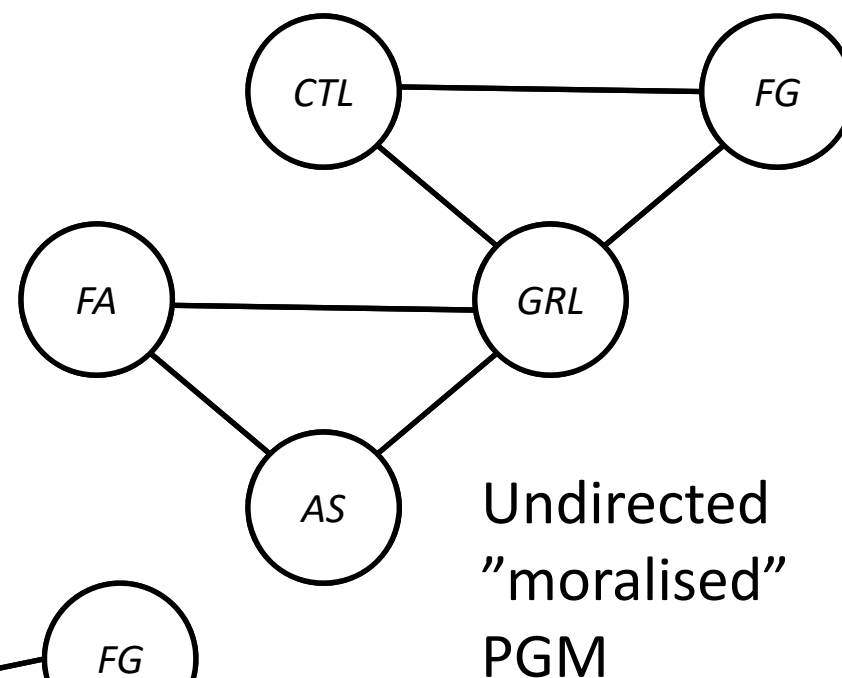
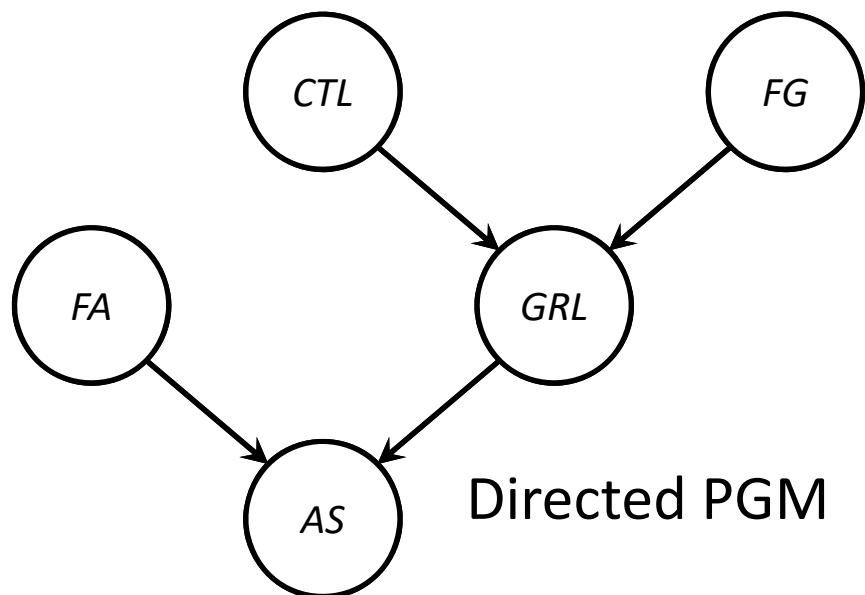
- Messages vector valued, i.e., function of target label
- Messages defined recursively: left to right, or right to left



# Sum-product algorithm

- Application of message passing to more general graphs
  - \* applies to chains, trees and poly-trees (directed PGMs with  $>1$  parent)
  - \* 'sum-product' derives from:
    - **product** = product of incoming messages
    - **sum** = summing out the effect of RV(s) *aka* elimination
- Algorithm supports other operations (semi-rings)
  - \* e.g., max-product, swapping **sum** for **max**
  - \* **Viterbi** algorithm is the max-product variant of the forward algorithm for HMMs, solves the  $\operatorname{argmax}_{\mathbf{q}} P(\mathbf{q}|\mathbf{o})$

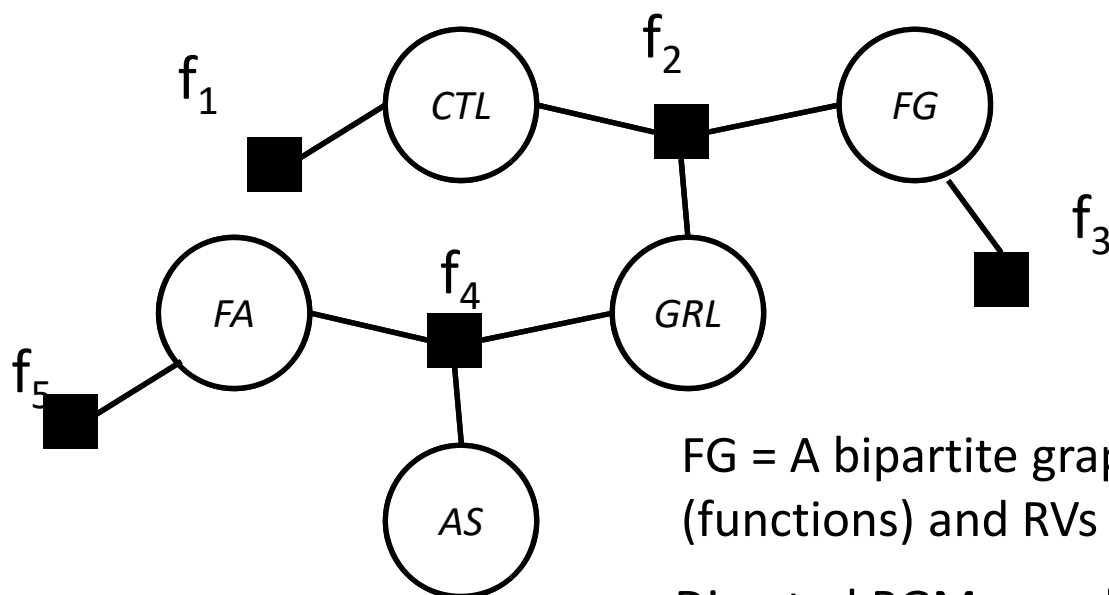
# Application to Directed PGMS



# Factor graphs

$$f_1(CTL) = P(CTL)$$

$$f_2(CTL, GRL, FG) = P(GRL|CTL, FG)$$

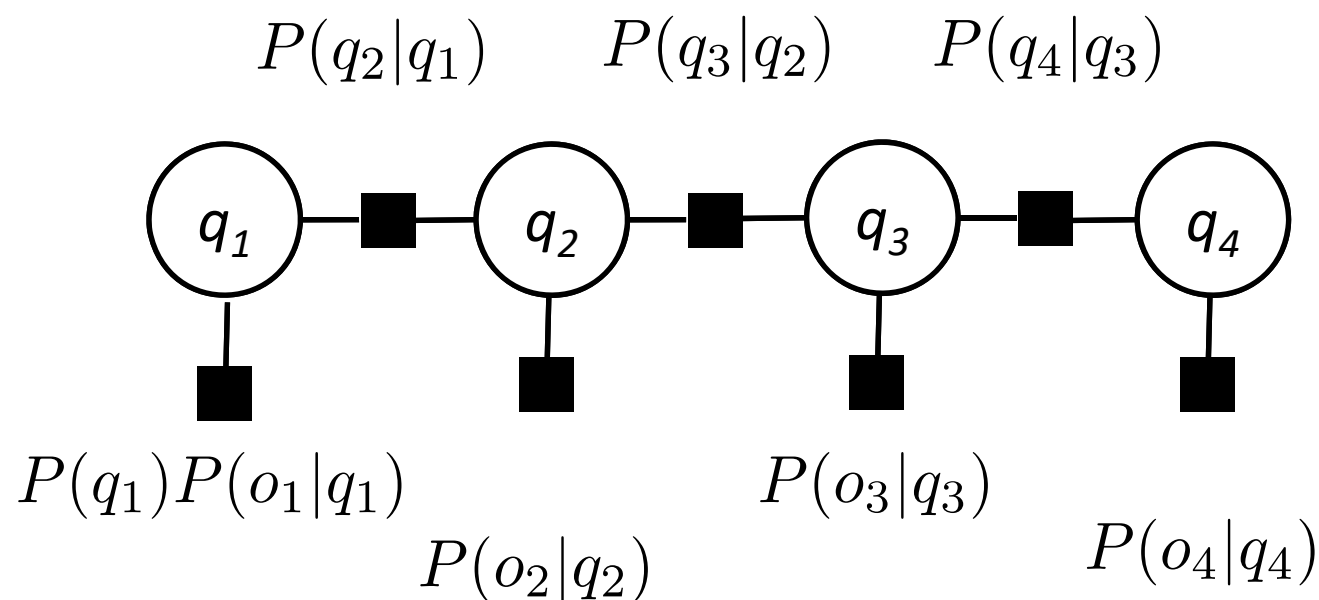


FG = A bipartite graph, with factors (functions) and RVs

Directed PGMs result in tree-structured FG



# Factor graph for the HMM



Effect of observed nodes incorporated into unary factors

# Sum-Product over Factor Graphs

- Two types of messages :
  - \* between factors and RVs; and between RVs and factors
  - \* summarise a complete sub-graph

- E.g.,

$$m_{f_2 \rightarrow GRL}(GRL) = \sum_{CTL} \sum_{FG} f_2(GRL, CTL, FG) m_{CTL \rightarrow f_2}(CTL) m_{FG \rightarrow f_2}(FG)$$

- Structure inference as “gather-and-distribute”
  - \* gather messages from leaves of tree towards root
  - \* then propagate message back down from root to leaves

# Summary

- HMMs as example PGMs
  - \* formulation as PGM
  - \* independence assumptions
  - \* probabilistic inference using forward-backward
  - \* statistical inference using expectation maximisation
- Message passing: general inference method for U-PGMs
  - \* sum-product & max-product
  - \* factor graphs