Universität
Münster

The ABC of Forgetting in PGMs: Aspects, Benefits, and Challenges

FILMRISS
Forgetting In Lifted Models and Reasoning In Search of Solutions
(Version 2.0)

Tanya Braun, Marcel Gehrke, Kai Sauerwald, Jan Speller

Probabilistic Graphical Models (PGMs)

- Set of factors F over subsets of random variables based on (conditional) independences among them
- Represent a full joint probability distribution P_F as a normalised product of the factors

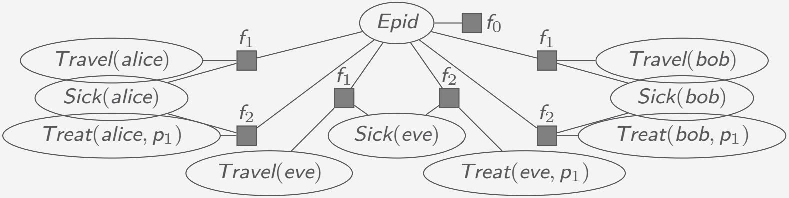


Figure: Example factor graph

Forgetting in PGMs: Objectives & Use Cases

Objectives

- Exploration of forgetting concept in marginalisation
- Long-term goal: Complexity reduction in temporal case

Use cases

- Privacy: Forgetting variables that are no longer acceptable to store without having to relearn a model
- Mental models: Emulate human forgetting
- Text analysis / digital humanities: Forgetting of characters over time / long texts

Forgetting in PGMs: One Possible Interpretation

- Idea: Forgetting as deleting parts of a model F
  - $\texttt{forget}(\delta, F)$ returns a model F', in which δ is “somehow” forgotten
  - Components to forget (δ): Random variable, factorisation (factor), edge

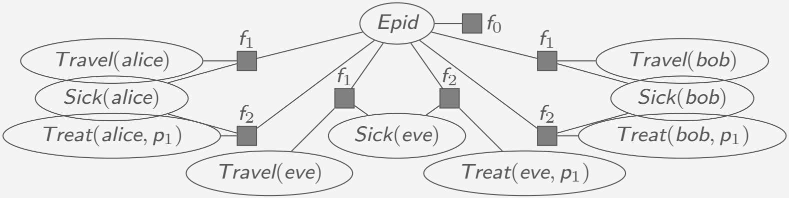


Figure: Example factor graph

Possible Properties

- *Information preserving*: Queries return the same result as before, i.e.,
$$P_F(\boldsymbol{R} \mid \boldsymbol{e}) = P_{\texttt{forget}(\delta,F)}(\boldsymbol{R} \mid \boldsymbol{e})$$
- *Reversible*: Original model can be restored from the one after forgetting, i.e.,
$$\texttt{remember}(\delta', \texttt{forget}(\delta, F)) = F$$
- *Order-independent*: The order of forgetting does not matter, i.e.,
$$\texttt{forget}(\delta_2, \texttt{forget}(\delta_1, F)) = \texttt{forget}(\delta_1, \texttt{forget}(\delta_2, F))$$
- *Idempotent*: Forgetting something twice does not change the model further, i.e.,
$$\texttt{forget}(\delta, F) = \texttt{forget}(\delta, \text{forget}(\delta, F))$$

Marginalisation as Forgetting

- Use variable elimination operators to implement forgetting in a model F
  - *Multiply* two factors $f' = \phi'(\boldsymbol{R}'), f'' = \phi''(\boldsymbol{R}'')$: Multiply the values for all possible values $\boldsymbol{r} \in \mathrm{ran}(\boldsymbol{R}' \cup \boldsymbol{R}'')$ projected onto $\boldsymbol{R}', \boldsymbol{R}''$, i.e., $\mathtt{multiply}(f', f'') = f$ with
$$\forall \boldsymbol{r} \in \mathrm{ran}(\boldsymbol{R}' \cup \boldsymbol{R}'') : \phi(\boldsymbol{r}) = \phi'(\pi_{\boldsymbol{R}'}(\boldsymbol{r})) \cdot \phi''(\pi_{\boldsymbol{R}''}(\boldsymbol{r}))$$
    - Postcondition: $P_{F \setminus \{f', f''\} \cup \{\mathtt{multiply}(f', f'')\}} = P_F$
  - *Sum out* a variable R from a factor $f = \phi R, \boldsymbol{R}$: Sum over the range values of R for all possible values $\boldsymbol{r}$ of the remaining variables $\boldsymbol{R}$, i.e., $\mathtt{sum-out}(R, f) = f' = \phi'(\boldsymbol{R})$ with
$$\forall \boldsymbol{r} \in \mathrm{ran}(\boldsymbol{R}) : \phi'(\boldsymbol{r}) = \sum_{r \in \mathrm{ran}(R)} \phi(r, \boldsymbol{r})$$
    - Precondition: f is the only factor containing R, i.e., $\forall f' \in F, f' \neq f, R \notin rv(f')$ (otherwise, multiply all factors containing R first)
    - Postcondition: $P_{F \setminus \{f\} \cup \{\mathtt{sum-out}(R, f)\}} = \sum_{r \in \mathrm{ran}(R)} P_F$

Forgetting Random Variables and Factors

Forget factor f
$$\triangleq$$
Multiply f with another factor f'

1. $f \leftarrow$ Multiply f, f' using `multiply`
   How to choose f'?

- Factor with largest overlap
- Smallest size of resulting factor

Forget random variable R
$$\triangleq$$
Sum out R

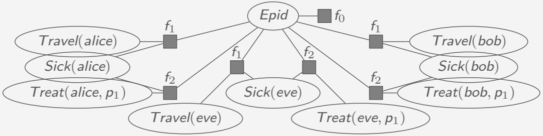
1. $f \leftarrow$ Multiply factors f_i that contain R using `multiply`
2. Sum out R in f using `sum-out`

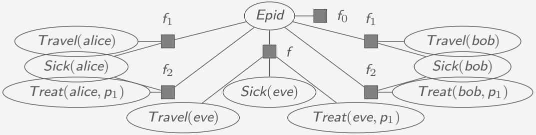
Forgetting edges is not possible with this framework.

Forgetting Random Variables and Factors: Example

Forgetting $f_1 = \phi_1(E, Tl.e, S.e)$: Multiply f_1 with $f_2 = \phi_2(E, Tt.e, S.e)$

$$\phi''(E, S.e, Tl.e, Tt.e) \leftarrow \phi_1(E, S.e, Tl.e) \cdot \phi_2(E, S.e, Tt.e)$$

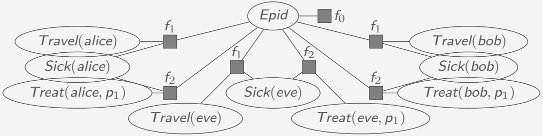




Forgetting Random Variables and Factors: Example

Forgetting $f_1 = \phi_1(E, Tl.e, S.e)$: Multiply f_1 with $f_2 = \phi_2(E, Tt.e, S.e)$

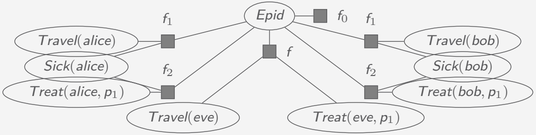
$$\phi''(E, S.e, Tl.e, Tt.e) \leftarrow \phi_1(E, S.e, Tl.e) \cdot \phi_2(E, S.e, Tt.e)$$



Forgetting *Sick.eve*: Sum out *Sick.eve*

$$\phi(E, S.e, Tl.e, Tt.e) \leftarrow \phi_1(E, S.e, Tl.e) \cdot \phi_2(E, S.e, Tt.e)$$

$$\phi(E, Tl.e, Tt.e) \leftarrow \sum_{s \in ran(S.e)} \phi(E, S.e = s, Tl.e, Tt.e)$$



Effects of Forgetting as Marginalisation

- Runtime complexity for an operation in a model F

  $$O(r^w) \text{ with } r = \max_{R \in rv(F)} |ran(R)| \text{ and } w \text{ the largest interim result}$$

  - w bounded from below by the maximum number of inputs in any factor: $\max_{f \in F} |rv(f)|$
  - w bounded above by the number of variables in F: $n = |rv(F)|$
  - Optimal elimination order leads to a minimum w (tree width)
- Forgetting a factor in F can increase w: increasing the bound from below
- Forgetting a variable in F can increase w: forcing a non-optimal elimination order

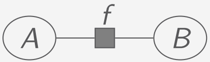
Properties of Forgetting as Marginalisation

- *Information preserving*: ✓ [by pre / postconditions of operators]
- *Reversible*: only with additional information
  - Graph: Inputs of original factors need to be stored (linear in w)
  - Potentials: Actual numbers necessary (exponential in w)
- *Order-independent*: ✓ [by commutativity of multiplication / addition; but complexity might change!]
- *Idempotent*: (✓) [not really possible to forget δ twice as it is no longer accessible in F]

What about Edges?

- Forgetting an edge means eliminating a variable from a single factor f
  $\rightarrow$ Not possible with above operators
- For all values $\boldsymbol{r}$ to the remaining variables $\boldsymbol{R}$, there is a set of potentials that the different values r of R map to
  $\rightarrow$ How do we arrive at a single value?
  - Sum? Max? Average?

  $\rightarrow$ Never information preserving, apart from sum if R only occurs in f

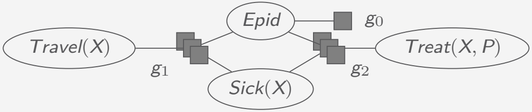


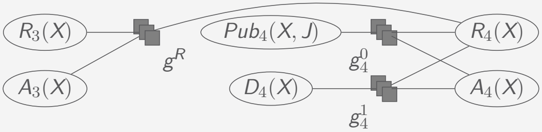
Forget edge $(B, f) \rightarrow$ Remove B from f

A	B	ϕ	ϕ'
true	*true*	φ_1	$\varphi_1 \circ \varphi_2$
true	*false*	φ_2	
false	*true*	φ_3	$\varphi_3 \circ \varphi_4$
false	*false*	φ_4	

Extensions to Marginalisation as Forgetting

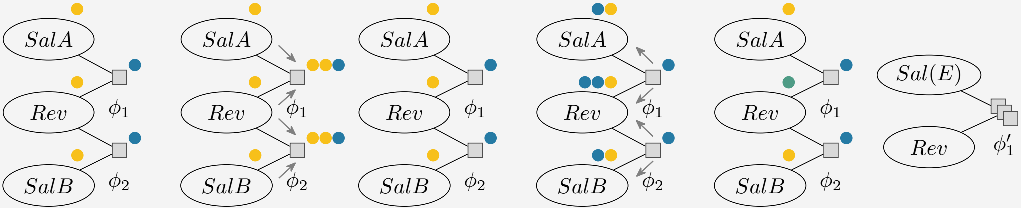
- First-order PGMs
  - Lifted marginalisation as forgetting in a similar vein
  - Includes forgetting sets of indistinguishable objects
- Temporal PGMs
  - Forgetting evidence over time (using TAMe by Gehrke et al., 2020)
  - Forgetting factors / edges between time steps
    - Markov-k to Markov-$k-1$ assumption $\rightarrow$ decreases complexity





Other Interpretations of Forgetting in PGMs

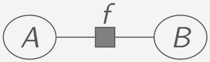
- Abstraction $\rightarrow$ ϵ-*Advanced Colour Passing* by Luttermann et al. (2024)
  - Combine ϵ-equivalent factors to abstract away from details



  - Moves from propositional model to first-order model

Other Interpretations of Forgetting in PGMs

- Potentials (certainty about them) $\rightarrow$ Variant of *Credal nets*
  - Use intervals to denote forgetting specific potentials



A	B	ϕ	ϕ'
true	*true*	φ_1	$\varphi_1 \pm \varepsilon$
true	*false*	φ_2	$\varphi_2 \pm \varepsilon$
false	*true*	φ_3	$\varphi_3 \pm \varepsilon$
false	*false*	φ_4	$\varphi_4 \pm \varepsilon$

Summary

- Use cases in privacy, mental modelling, text analysis
- Properties: information-preserving, reversible, order-independent, idempotent
- Marginalisation as forgetting
  - Sum-out, multiply operators to realise forgetting of variables, factors
  - Forgetting can negatively impact complexity
  - Forgetting edges not obvious how to handle
- Extensions: first-order models, temporal models
- Other interpretations: abstraction, forgetting potentials

*This work was partially funded by
the Ministry of Culture and Science of the German State of North Rhine-Westphalia.*