# Restricting the Maximum Number of Actions for Decision Support under Uncertainty

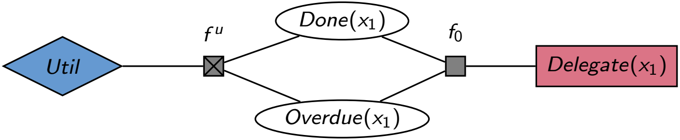
Marcel Gehrke[1], Tanya Braun[1], Simon Polovina[2]

[1] Institute of Information Systems, University of Lübeck
[2] Conceptual Structures Research Group, Sheffield Hellam University

September 19, 2020

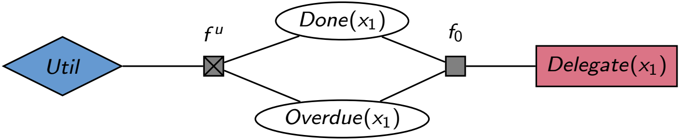
# Decision Making in Probabilistic Graphical Models

Decision Factor Graph $F = \{f_i\}_{i=1}^n$ with Utilities and Actions



# Decision Making in Probabilistic Graphical Models

Decision Factor Graph $F = \{f_i\}_{i=1}^n$ with Utilities and Actions



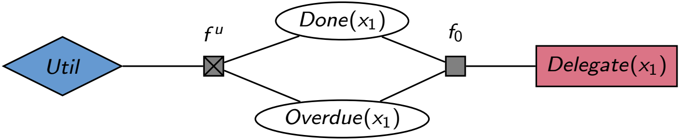

Maximum expected utility (MEU) query
given evidence $\mathbf{e}$ over all possible action assignments $\mathbf{a}$

$$MEU(F, \mathbf{e}) = (\mathbf{a}^*, EU(F, \mathbf{e}, \mathbf{a}^*)) \qquad a^* = \arg\max_{\mathbf{a}} EU(F, \mathbf{e}, \mathbf{a})$$

$EU$ = expected utility of $\mathbf{a}$ in $F$ given $\mathbf{e}$

# Decision Making in Probabilistic Graphical Models

Decision Factor Graph $F = \{f_i\}_{i=1}^n$ with Utilities and Actions




Maximum expected utility (MEU) query
given evidence **e** over all possible action assignments **a**

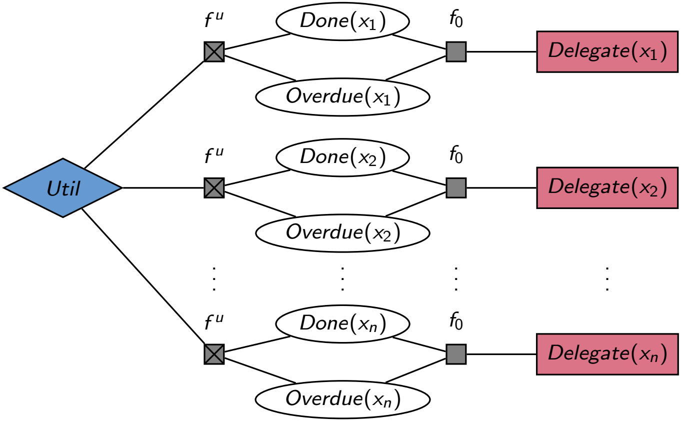
$MEU(F, \mathbf{e}) = (\mathbf{a}^*, EU(F, \mathbf{e}, \mathbf{a}^*))$ $\quad a^* = \arg\max_{\mathbf{a}} EU(F, \mathbf{e}, \mathbf{a})$

$EU$ = expected utility of **a** in $F$ given **e**

**Exponential in the number of actions!**

# Relational Domains

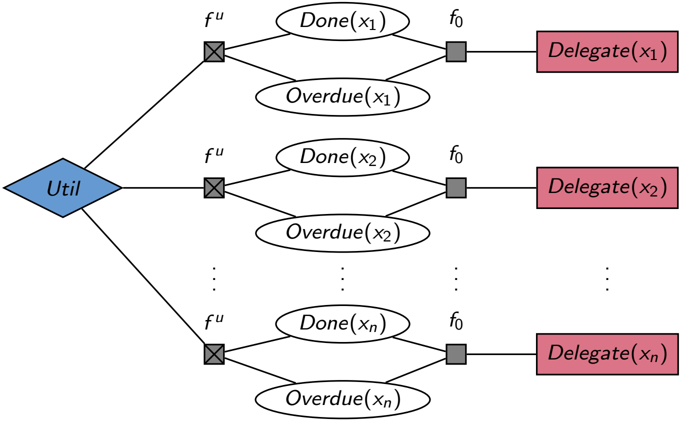
## Indistinguishable Constants




With $n = 100$: $2^{100}$ possible assignments

# Relational Domains

## Indistinguishable Constants

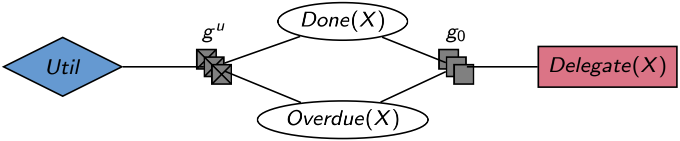



With $n = 100$: $2^{100}$ possible assignments

**Treat identically until evidence makes them distinguishable**

# Groups in Probabilistic Relational Models

Decision Parfactor Graph $G = \{g_i\}_{i=1}^n$ with *Parameterised* Utilities and Actions




Evidence **e** makes constants distinguishable, e.g., with $|dom(X)| = 100$

$$Overdue(X_1) = true, dom(X_1) = \{x_1, x_2, \dots, x_{10}\}$$

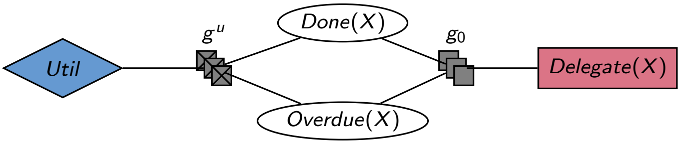
$$Overdue(X_2) = false, dom(X_2) = \{x_{11}, x_{12}, \dots, x_{20}\}$$

$$Overdue(X_3) \text{ N/A}, dom(X_3) = \{x_{21}, x_{12}, \dots, x_{100}\}$$

**Constants still indistinguishable within group**

# Decision Making in Probabilistic Relational Models

Decision Parfactor Graph $G = \{g_i\}_{i=1}^{n}$ with *Parameterised* Utilities and Actions



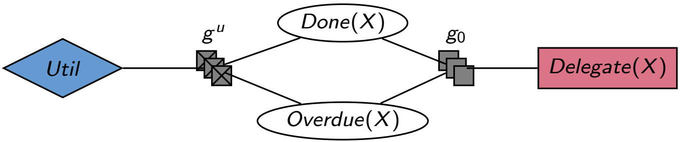

Maximum expected utility (MEU) query
given evidence **e** over all possible action assignments **a**

$MEU(G, \mathbf{e}) = (\mathbf{a}^*, EU(G, \mathbf{e}, \mathbf{a}^*))$ $\quad$ $a^* = \arg\max_{\mathbf{a}} EU(G, \mathbf{e}, \mathbf{a})$

$EU$ = expected utility of **a** in $G$ given **e**

# Decision Making in Probabilistic Relational Models

Decision Parfactor Graph $G = \{g_i\}_{i=1}^n$ with *Parameterised* Utilities and Actions




Maximum expected utility (MEU) query
given evidence **e** over all possible action assignments **a**

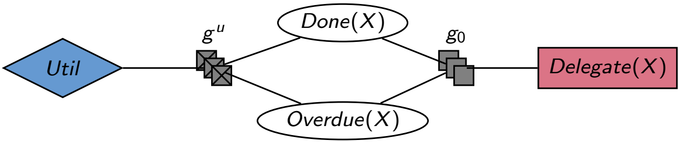
$MEU(G, \mathbf{e}) = (\mathbf{a}^*, EU(G, \mathbf{e}, \mathbf{a}^*))$ $\quad a^* = \arg\max_{\mathbf{a}} EU(G, \mathbf{e}, \mathbf{a})$

$EU$ = expected utility of **a** in $G$ given **e**

**Exponential in the number of groups in domains!**

# Decision Making in Probabilistic Relational Models

Decision Parfactor Graph $G = \{g_i\}_{i=1}^{n}$ with *Parameterised* Utilities and Actions




**Instead of $2^{100}$ possible assignments**

$$Overdue(X_1) = true, dom(X_1) = \{x_1, x_2, \dots, x_{10}\}$$

$$Overdue(X_2) = false, dom(X_2) = \{x_{11}, x_{12}, \dots, x_{20}\}$$

$$Overdue(X_3)\ \text{N/A}, dom(X_3) = \{x_{21}, x_{12}, \dots, x_{100}\}$$

**$\rightarrow 3$ groups: $2^3$ possible assignments**

# Conference Contribution

## Problem

- Constraints on resources may render action assignments invalid.
- Error-prone to check each assignment with complex restrictions.

**Iterating over too many assignments or not enough.**

# Conference Contribution

## Problem

- Constraints on resources may render action assignments invalid.
- Error-prone to check each assignment with complex restrictions.

**Iterating over too many assignments or not enough.**

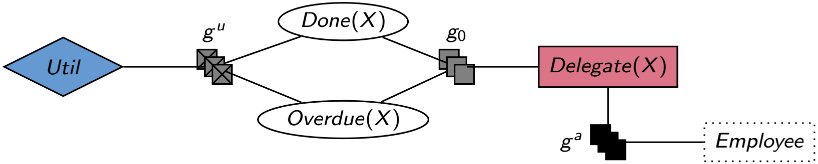
## Approach: ReLiA

- Build a graph out of the resource restrictions
- Solve a max-flow problem in the graph
- Return all max-flows with maxed out capacities

**Iterate only over the necessary assignments.**

# ReLiA: Restricting Lifted Assignments

## Resources and Action Parfactors




Action parfactors indicate

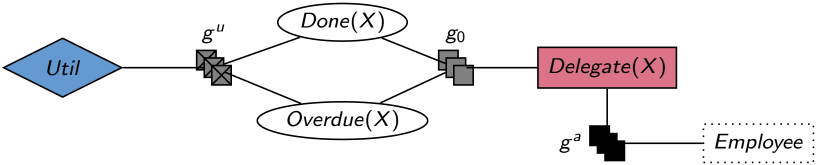
- how many resource units are needed per individual action and
- how many times an action can be assigned.

E.g., 15 employees that carry out tasks

Employees needed for one task: 1    Times task executable: 20

# ReLiA: Restricting Lifted Assignments

Resource Graph and Max-flow Problem

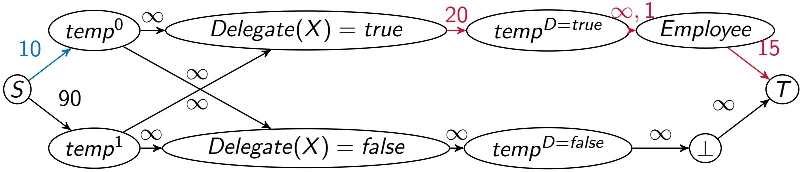



15 employees

Employees needed for one task: 1

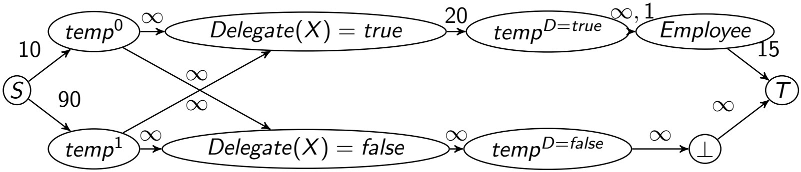
Times task executable: 20

$Overdue(X_1) = true, dom(X_1) = \{x_1, x_2, \ldots, x_{10}\}$, rest N/A



# ReLiA: Restricting Lifted Assignments

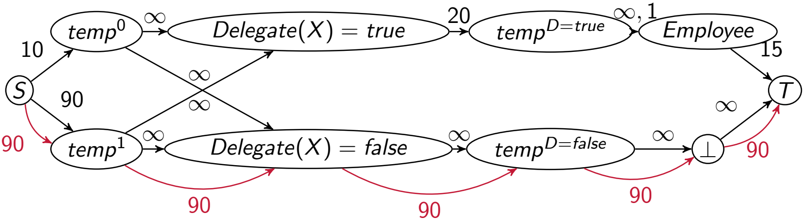
## Resource Graph and Max-flow Problem




Max-flow problem provides how many actions are executable, need all flows with max-flow when sending max capacities, e.g.:

# ReLiA: Restricting Lifted Assignments

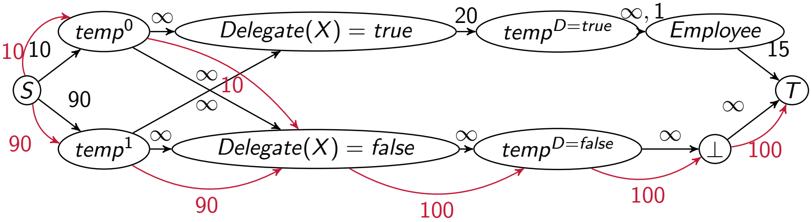
## Resource Graph and Max-flow Problem




Max-flow problem provides how many actions are executable, need all flows with max-flow when sending max capacities, e.g.:

# ReLiA: Restricting Lifted Assignments

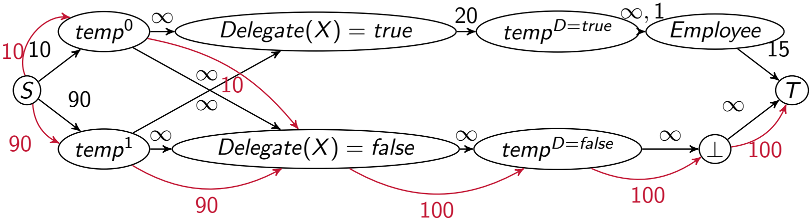
## Resource Graph and Max-flow Problem




Max-flow problem provides how many actions are executable, need all flows with max-flow when sending max capacities, e.g.:

# ReLiA: Restricting Lifted Assignments

## Resource Graph and Max-flow Problem

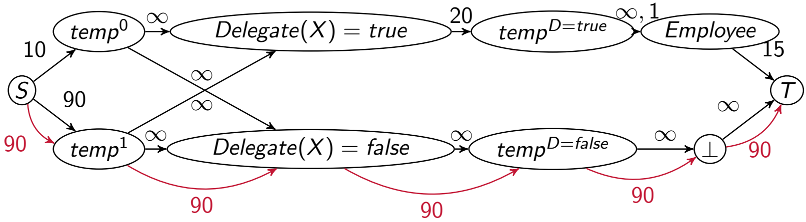



Max-flow problem provides how many actions are executable, need all flows with max-flow when sending max capacities, e.g.:

$\forall x \in dom(X_1, X_2) : Delegate(x) = false$

# ReLiA: Restricting Lifted Assignments

Resource Graph and Max-flow Problem

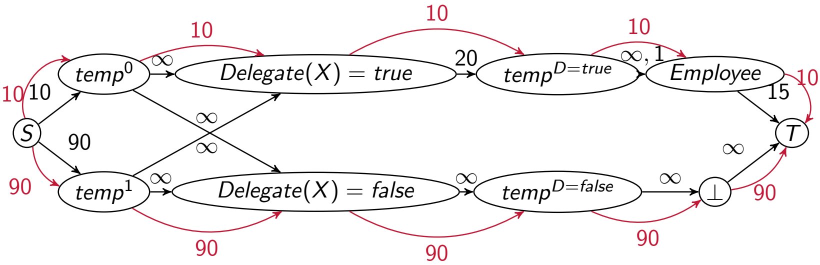



Max-flow problem provides how many actions are executable, need all flows with max-flow when sending max capacities, e.g.:

$$\forall x \in dom(X_1, X_2) : Delegate(x) = false$$

# ReLiA: Restricting Lifted Assignments

## Resource Graph and Max-flow Problem

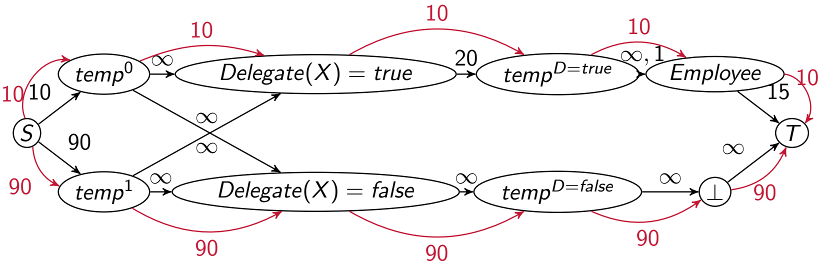



Max-flow problem provides how many actions are executable, need all flows with max-flow when sending max capacities, e.g.:

$$\forall x \in dom(X_1, X_2) : Delegate(x) = false$$

# ReLiA: Restricting Lifted Assignments

## Resource Graph and Max-flow Problem



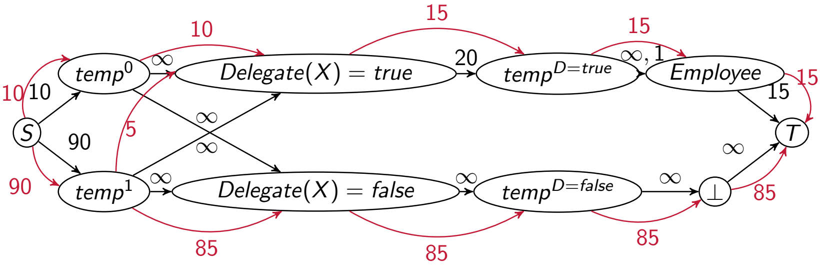

Max-flow problem provides how many actions are executable, need all flows with max-flow when sending max capacities, e.g.:

$$\forall x \in dom(X_1, X_2) : Delegate(x) = false$$

$$\forall x \in dom(X_1) : D(x) = true, \forall x \in dom(X_2) : D(x) = false$$

# ReLiA: Restricting Lifted Assignments

## Resource Graph and Max-flow Problem



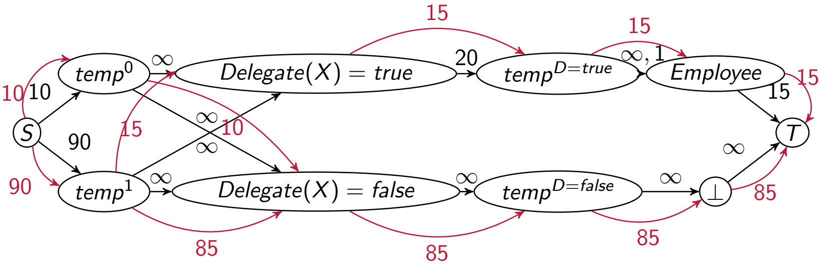

$$\forall x \in dom(X_1, X_2) : Delegate(x) = false$$

$$\forall x \in dom(X_1) : D(x) = true, \forall x \in dom(X_2) : D(x) = false$$

$$\forall x \in dom(X_1, Y), Y = \{5 \text{ of } X_2\} : D(x) = true, \forall x \in dom(X_2) \setminus Y : D(x) = false$$

# ReLiA: Restricting Lifted Assignments

## Resource Graph and Max-flow Problem




$$\forall x \in dom(X_1, X_2) : Delegate(x) = false$$

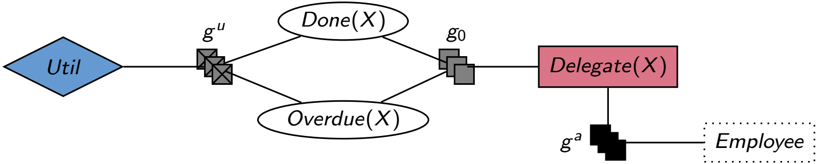
$$\forall x \in dom(X_1) : D(x) = true, \forall x \in dom(X_2) : D(x) = false$$

$$\forall x \in dom(X_1, Y), Y = \{5 \text{ of } X_2\} : D(x) = true, \forall x \in dom(X_2) \setminus Y : D(x) = false$$

$$\forall x \in dom(Z), Z = \{15 \text{ of } X_2\} : D(x) = true, \forall x \in dom(X_1, X_2) \setminus Z : D(x) = false$$

# Decision Making in Probabilistic Relational Models

Decision Parfactor Graph $G = \{g_i\}_{i=1}^{n}$ with *Parameterised* Utilities and Actions




15 employees

Employees needed for one task: 1

Times task executable: 20

$$Overdue(X_1) = true, dom(X_1) = \{x_1, x_2, \ldots, x_{10}\}$$

$\forall x \in dom(X_1, X_2) : Delegate(x) = false$

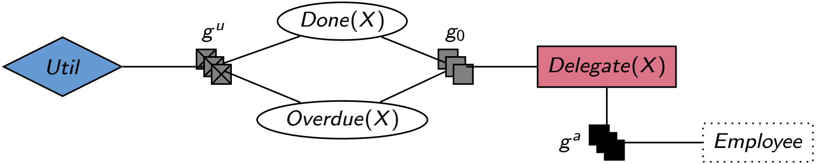
$\forall x \in dom(X_1) : D(x) = true, \forall x \in dom(X_2) : D(x) = false$

$\forall x \in dom(X_1, Y), Y = \{5 \text{ of } X_2\} : D(x) = true, \forall x \in dom(X_2) \setminus Y : D(x) = false$

$\forall x \in dom(Z), Z = \{15 \text{ of } X_2\} : D(x) = true, \forall x \in dom(X_1, X_2) \setminus Z : D(x) = false$

# Decision Making in Probabilistic Relational Models

Decision Parfactor Graph $G = \{g_i\}_{i=1}^n$ with *Parameterised* Utilities and Actions




$$MEU(G, \mathbf{e}) = (\mathbf{a}^*, EU(G, \mathbf{e}, \mathbf{a}^*)) \qquad a^* = \arg\max_{\mathbf{a}} EU(G, \mathbf{e}, \mathbf{a})$$

$$\mathbf{e} = \{Overdue(X_1) = true\}, dom(X_1) = \{x_1, x_2, \dots, x_{10}\}$$

$$\forall x \in dom(X_1, X_2) : Delegate(x) = false$$

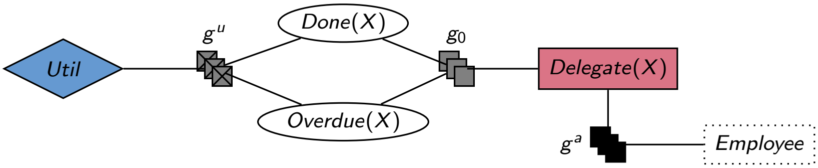
$$\forall x \in dom(X_1) : D(x) = true, \forall x \in dom(X_2) : D(x) = false$$

$$\forall x \in dom(X_1, Y), Y = \{5 \text{ of } X_2\} : D(x) = true, \forall x \in dom(X_2) \setminus Y : D(x) = false$$

$$\forall x \in dom(Z), Z = \{15 \text{ of } X_2\} : D(x) = true, \forall x \in dom(X_1, X_2) \setminus Z : D(x) = false$$

# Decision Making in Probabilistic Relational Models

Decision Parfactor Graph $G = \{g_i\}_{i=1}^{n}$ with *Parameterised* Utilities and Actions




$$MEU(G, \mathbf{e}) = (\mathbf{a}^*, EU(G, \mathbf{e}, \mathbf{a}^*)) \qquad a^* = \arg\max_{\mathbf{a}} EU(G, \mathbf{e}, \mathbf{a})$$

$$\mathbf{e} = \{Overdue(X_1) = true\}, dom(X_1) = \{x_1, x_2, \dots, x_{10}\}$$

$$\forall x \in dom(X_1, X_2) : Delegate(x) = false$$

$$\forall x \in dom(X_1) : D(x) = true, \forall x \in dom(X_2) : D(x) = false$$

$$\forall x \in dom(X_1, Y), Y = \{5 \text{ of } X_2\} : D(x) = true, \forall x \in dom(X_2) \setminus Y : D(x) = false$$

$$\forall x \in dom(Z), Z = \{15 \text{ of } X_2\} : D(x) = true, \forall x \in dom(X_1, X_2) \setminus Z : D(x) = false$$

# Conference Contribution

## Problem

- Constraints on resources may render action assignments invalid.
- Error-prone to check each assignment with complex restrictions.

**Iterating over too many assignments or not enough.**

## Approach: ReLiA

- Build a graph out of the resource restrictions
- Solve a max-flow problem in the graph
- Return all max-flows with maxed out capacities

**Iterate only over the necessary assignments.**