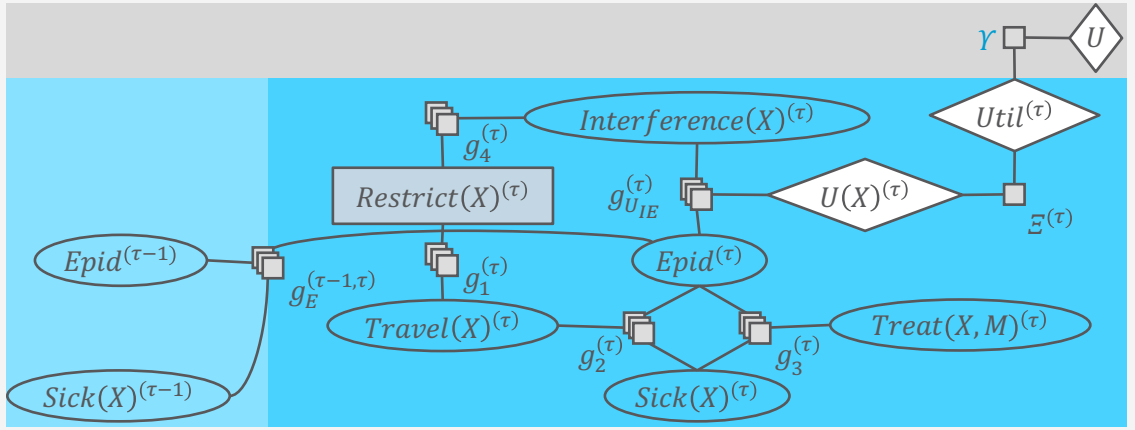
WWU
MÜNSTER

Lifted Decision Making

Statistical Relational Artificial Intelligence (StaRAI)

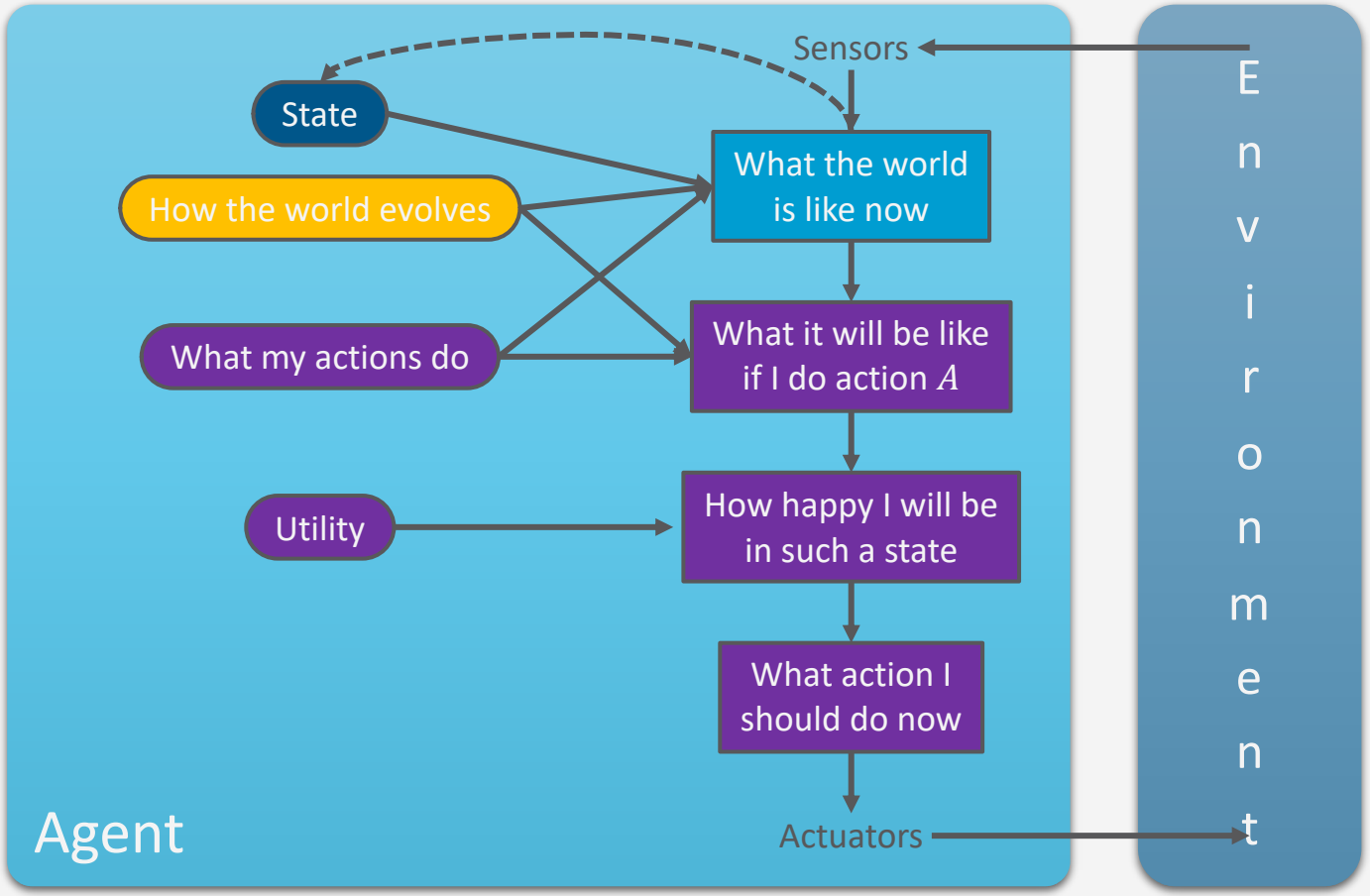


Tanya Braun

Contents

Contents in this Lecture Related to *Utility-based Agents*

- Further topics
  3. (Episodic) PRMs
  4. Lifted inference (in episodic PRMs)
  5. Lifted learning (of episodic PRMs)
  6. Lifted sequential PRMs and inference
  7. Lifted decision making
  8. Continuous space and lifting



Setting

- Agent can perform actions in an environment
  - Episodic, i.e., not sequential, environment
    - Next episode does not depend on the previous episode

    Or sequential environment
  - Non-deterministic environment
    - Outcomes of actions not unique
    - Associated with probabilities
      → probabilistic model
  - Partially observable
    - Latent, i.e., not observable, random variables
- Agent has preferences over states / action outcomes
  - Encoded in utility or utility function → Utility theory
- "Decision theory
  = Utility theory + Probability theory"
  - Model the world with a probabilistic model
  - Model preferences with a utility (function)
  - Find action that leads to the maximum expected utility, also called decision making

Outline: 7. Lifted Decision Making

A. Utility theory

- Preferences, maximum expected utility (MEU) principle
- Utility function, multi-attribute utility theory

B. Static decision making

- Modelling, semantics, inference tasks
- Inference algorithm: LVE for MEU as an example

C. Sequential decision making

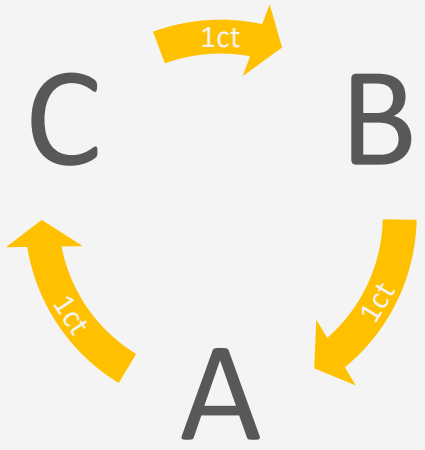
- Modelling, semantics, sequential MEU problem
- Inference algorithm: LDJT for MEU as an example
- Acting

Preferences

- An agent chooses among prizes (A, B, etc.) and lotteries, i.e., situations with uncertain prizes
  - Outcome of a nondeterministic action is a lottery
- Lottery $L = [p, A; (1 - p), B]$
  - A and B can be lotteries again
  - Prizes are special lotteries: $[1, R; 0, \text{not } R]$
  - More than two outcomes:
    - $L = [p_1, S_1; p_2, S_2; \cdots; p_M, S_M], \sum_{i=1}^{M} p_i = 1$
- Notation
  - $A \succ B$ A preferred to B
  - $A \sim B$ indifference between A and B
  - $A \succsim B$ B not preferred to A

Rational Preferences

- Idea: preferences of a rational agent must obey constraints
  - As prerequisite for reasonable preference relations
- Rational preferences → behaviour describable as maximisation of expected utility
- Violating constraints leads to self-evident irrationality
  - Example
    - An agent with intransitive preferences can be induced to give away all its money
      - If $B \succ C$, then an agent who has C would pay (say) 1 cent to get B
      - If $A \succ B$, then an agent who has B would pay (say) 1 cent to get A
      - If $C \succ A$, then an agent who has A would pay (say) 1 cent to get C

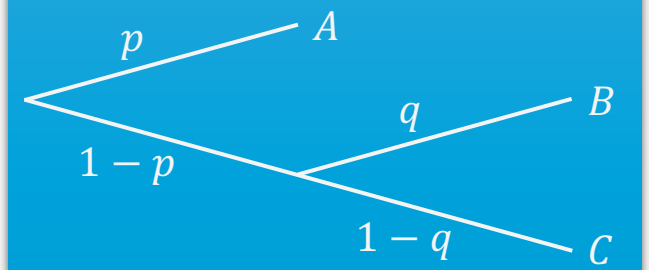


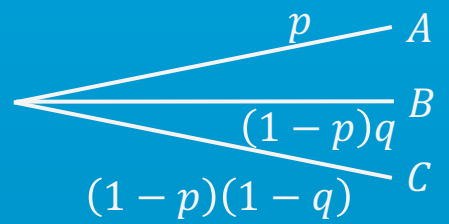
Axioms of Utility Theory

1. Orderability
   - $(A \succ B) \vee (A \prec B) \vee (A \sim B)$
     - $\{\prec, \succ, \sim\}$ jointly exhaustive, pairwise disjoint
2. Transitivity
   - $(A \succ B) \wedge (B \succ C) \Rightarrow (A \succ C)$
3. Continuity
   - $A \succ B \succ C \Rightarrow \exists p\, [p, A;\ 1-p, C] \sim B$
4. Substitutability
   - $A \sim B \Rightarrow [p, A; 1-p, C] \sim [p, B; 1-p, C]$
     - Also holds if replacing $\sim$ with $\succ$
5. Monotonicity
   - $A \succ B \Rightarrow (p \geq q \Leftrightarrow [p, A; 1-p, B] \succsim [q, A; 1-q, B])$
6. Decomposability
   - $[p, A;\ 1-p, [q, B;\ 1-q, C]] \sim [p, A;\ (1-p)q, B; (1-p)(1-q), C]$

Decomposability:
There is no fun in gambling.

Equivalent lotteries:





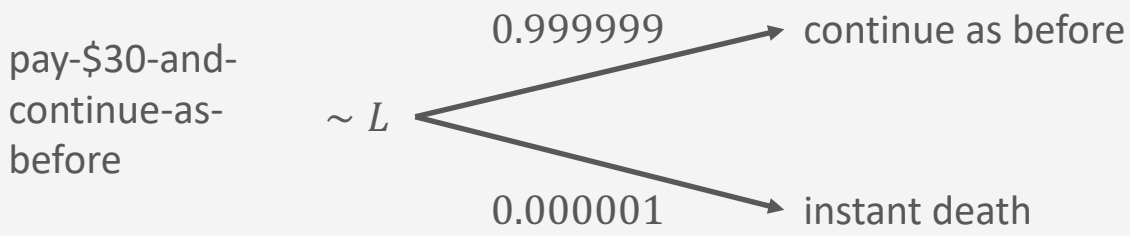
And Then There Was Utility

- Theorem (Ramsey, 1931; von Neumann and Morgenstern, 1944):
  - Given preferences satisfying the constraints, there exists a real-valued function U such that
$$U(A) \geq U(B) \Leftrightarrow A \gtrsim B$$
    - Existence of a utility function
  - Expected utility of a lottery:
$$U([p_1, S_1; \ldots; p_M, S_M]) = \sum_{i=1}^{M} p_i U(S_i)$$
- MEU principle
  - Choose the action that maximises expected utility

Utilities

- Utilities map states to real numbers.
  Which numbers?

- Standard approach to assessment of human utilities:
  - Compare a given state A to a standard lottery L_p that has
    - "best possible outcome" $\top$ with probability p
    - "worst possible catastrophe" $\bot$ with probability $(1-p)$
  - Adjust lottery probability p until $A \sim L_p$



Utility Scales

- Normalised utilities: $u_\top = 1.0, u_\bot = 0.0$
  - Utility of lottery $L \sim$ (pay-\$30-and-continue-as-before): $U(L) = u_\top \cdot 0.999999 + u_\bot \cdot 0.000001 = 0.999999$
- Micromorts: one-millionth chance of death
  - Useful for Russian roulette, paying to reduce product risks, etc.
  - Example for low risk
    - Drive a car for 370km ≈ 1 micromort → lifespan of a car: 150,000km ≈ 400 micromorts
    - Studies showed that many people appear to be willing to pay US\$10,000 for a safer car that halves the risk of death → US\$50/micromort
- QALYs: quality-adjusted life years
  - Useful for medical decisions involving substantial risk
- In planning: task becomes minimisation of cost instead of maximisation of utility

Utility Scales

- Behaviour is invariant w.r.t. positive linear transformation
$$U'(r) = k_1 U(r) + k_2$$
  - No unique utility function; $U'(r)$ and $U(r)$ yield same behaviour
- With deterministic prizes only (no lottery choices), only ordinal utility can be determined, i.e., total order on prizes
  - Ordinal utility function also called value function
  - Provides a ranking of alternatives (states), but not a meaningful metric scale (numbers do not matter)
- Note:
  An agent can be entirely rational (consistent with MEU) without ever representing or manipulating utilities and probabilities
  - E.g., a lookup table for perfect tic-tac-toe

Multi-attribute Utility Theory

- A given state may have multiple utilities
  - ...because of multiple evaluation criteria
  - ...because of multiple agents (interested parties) with different utility functions

- There are:
  - Cases in which decisions can be made *without* combining the attribute values into a single utility value
    - Strict dominance
    - Not this lecture
  - Cases in which the utilities of attribute combinations can be specified very concisely
    - This lecture!

Preference Structure

- To specify the complete utility function $U(r_1, \dots, r_M)$, we need d^M values in the worst case
  - M attributes
  - each attribute with d distinct possible values
  - Worst case meaning: Agent's preferences have no regularity at all
- Supposition in multi-attribute utility theory
  - Preferences of typical agents have much more structure
- Approach
  - Identify regularities in the preference behaviour
  - Use so-called representation theorems to show that an agent with a certain kind of preference structure has a utility function

$$U(r_1, \dots, r_M) = \Xi[f_1(r_1), \dots, f_M(r_M)]$$

    - where Ξ is hopefully a simple function such as *addition*

Preference Independence

- R_1 and R_2 preferentially independent (PI) of R_3 iff
  - Preference between $\langle r_1, r_2, r_3 \rangle$ and $\langle r_1', r_2', r_3 \rangle$ does not depend on r_3
  - E.g., $\langle Noise, Cost, Safety \rangle$
    - $\langle 20{,}000\ suffer, \$4.6\ billion, 0.06\ deaths/month \rangle$
    - $\langle 70{,}000\ suffer, \$4.2\ billion, 0.06\ deaths/month \rangle$

- Theorem (Leontief, 1947)
  - If every pair of attributes is PI of its complement, then every subset of attributes is PI of its complement
    - Called mutual PI (MPI)

Preference Independence

- Theorem (Debreu, 1960):
  - MPI ⇒ ∃ *additive* value function

$$V(r_1, \dots, r_M) = \sum_{i=1}^{M} V_i(r_i)$$

  - Hence assess M single-attribute functions
    - Decomposition of V into a set of summands (additive semantics)

    similar to
    - Decomposition of $P_{\boldsymbol{R}}$ into a set of factors (multiplicative semantics)
  - Often a good approximation
  - Example:

$$V(Noise, Cost, Deaths) = -Noise \cdot 10^4 - Cost - Deaths \cdot 10^{12}$$

Interim Summary

- Preferences
  - Preferences of a rational agent must obey constraints
- Utilities
  - Rational preferences = describable as maximisation of expected utility
  - Utility axioms
  - MEU principle
- Multi-attribute utility theory
  - Preference structure
  - (Mutual) preferential independence

Outline: 7. Lifted Decision Making

A. Utility theory
- Preferences, maximum expected utility (MEU) principle
- Utility function, multi-attribute utility theory

B. Static decision making
- Modelling, semantics, inference tasks
- Inference algorithm: LVE for MEU as an example

C. Sequential decision making
- Modelling, semantics, sequential MEU problem
- Inference algorithm: LDJT for MEU as an example
- Acting

Decision Networks/Models

- Extend a PGM to handle actions and utilities
  - Decision variables
  - Utility variables
- Also called influence diagrams
- Given a decision model, use an inference method of one's choosing to find actions that lead to the highest expected utility

- Also allows to perform so-called *Value of Information* calculations
  - Is it worth it to spend resources on getting more information (in the form of evidence)?

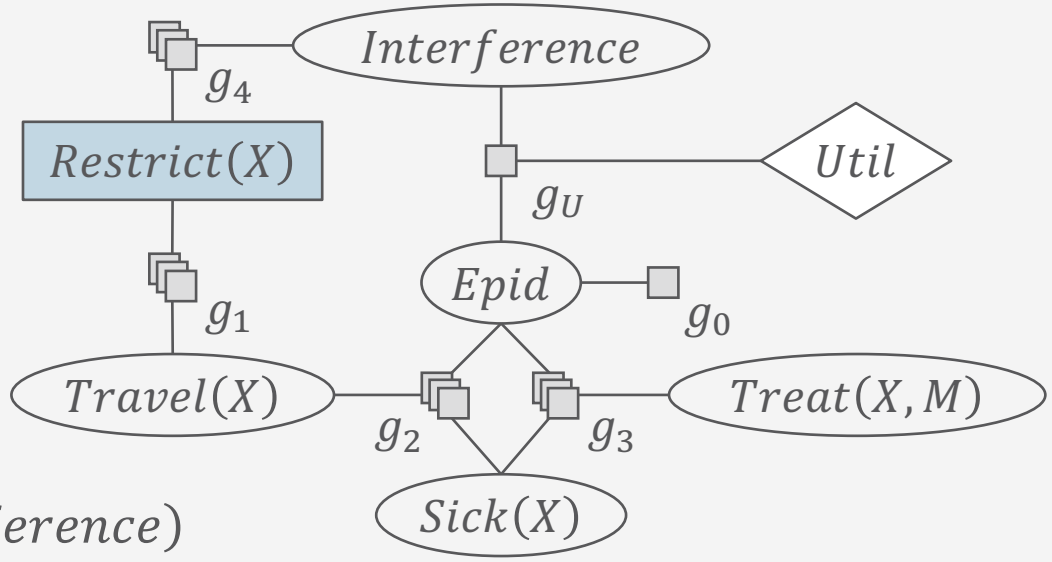
Ronald A. Howard: Information Value Theory. In: *IEEE Transactions on Systems Science and Cybernetics*, 1966.
Ronald A. Howard, James E. Matheson: Influence Diagrams. In: *Readings on the Principles and Applications of Decision Analysis*, 1984.

Decision PRVs

- Decision PRV D
  - Range $\mathrm{ran}(D) = \{d_i\}_{i=1}^{K}$ set of possible actions
    - Actions d_i mutually exclusive (consistent with range definition)
    - Always have to get a value assigned
      - Cannot not make a decision!
  - Depicted by a rectangle in a graphical representation
  - E.g., travel restrictions for people X: $Restrict(X)$
    - Range values: $ban, free$
- Set of decision PRVs $\boldsymbol{D}$ in a model, i.e., $\boldsymbol{R} = \boldsymbol{D} \cup \boldsymbol{V}$
  - $\boldsymbol{D}$ can occur as arguments to any parfactor
  - Example:
    - $\phi_1\big(Restrict(X), Travel(X)\big), \phi_4(Restrict(X), Interference)$

$R(X)$	I	ϕ_4
$free$	$false$	1
$free$	$true$	0
ban	$false$	0
ban	$true$	1

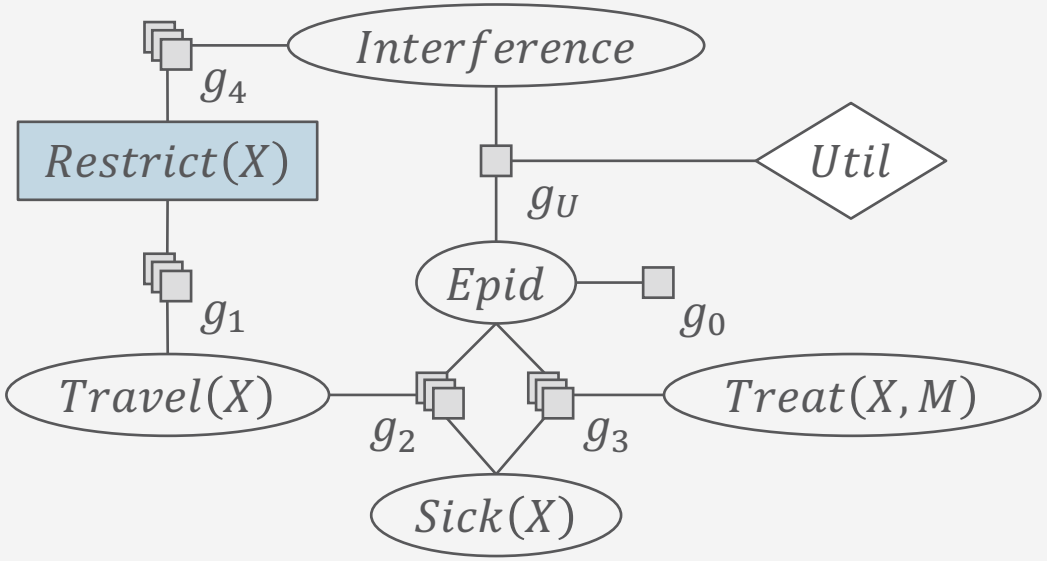
$R(X)$	$Tl(X)$	ϕ_1
$free$	$false$	1
$free$	$true$	1
ban	$false$	1
ban	$true$	0



Utility PRVs & Utility Parfactors

- Utility PRV U
  - Range $\mathrm{ran}(U) = \mathbb{R}$
  - Output variable, i.e., gets assigned a value by utility function
  - Depicted by a diamond in a graphical representation
- Utility parfactor $\phi_U(\mathcal{A})_{|C}$
  - Arguments $\mathcal{A}$ a sequence of (decision) PRVs
  - U a utility PRV
  - Function $\phi_U{:}\times_{i=1}^{l}\mathrm{ran}(R_i) \mapsto \mathrm{ran}(U)$
    - Tabular representation, additive function, …
      - Tabular example $\phi_{Util}(Interference, Epid)$
      - Example from slide 18 additive:
        $V(N, C, D) = -Noise \cdot 10^4 - Cost - Deaths \cdot 10^{12}$

I	E	$Util$
$false$	$false$	10
$false$	$true$	-10
$true$	$false$	-20
$true$	$true$	2

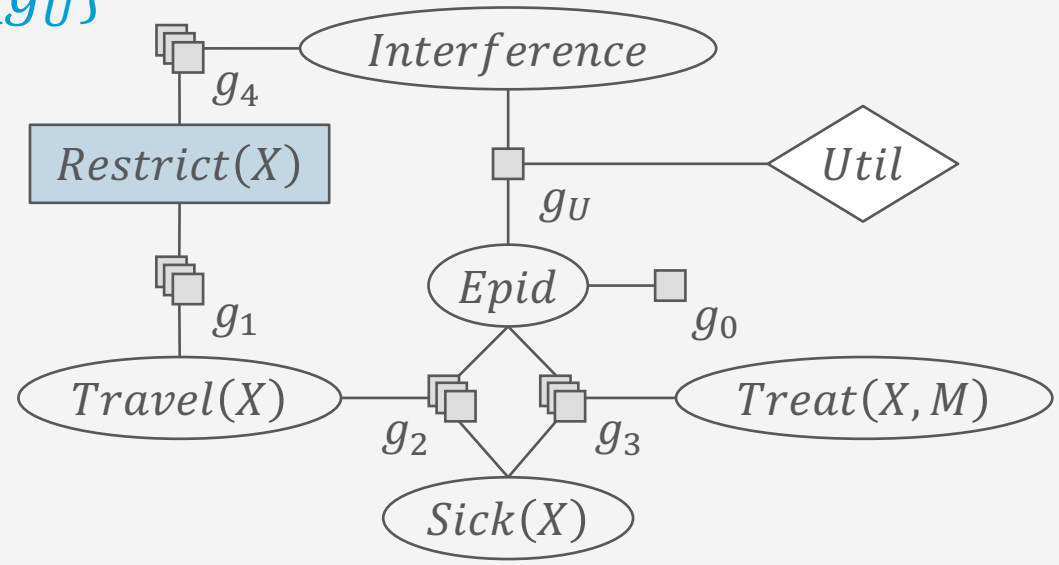


Parfactor-based Decision Model

- Decision model = Parfactor model that allows decision PRVs in the arguments of its parfactors as well as utility parfactors
  - For ease of exposition, we start with models with a utility *factor* mapping to a utility *variable*
  - Formally,

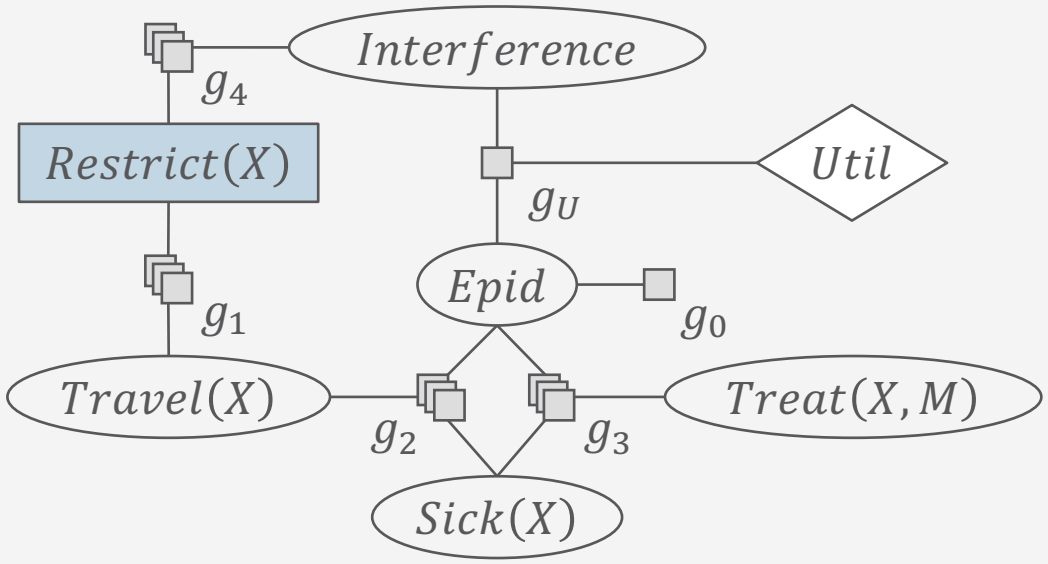
$$G = \{g_i\}_{i=1}^{n} \cup \{g_U\}$$

    - g_i parfactors with (decision) PRVs as arguments
    - g_U utility factor mapping to a utility variable U
      - $\mathrm{rv}(g_U) = \emptyset$ for now
  - E.g.,
    - $G = \{g_0, g_1, g_2, g_3, g_4, g_U\}$
      - T constraints



Decision Model: Action Assignments

- Let $\boldsymbol{D} = \{D_1, \ldots, D_k\}_{|C}$ be the set of decision PRVs in G with a constraint C for the logical variables in $\boldsymbol{D}$
- Then, $\boldsymbol{d}$ is a compound event for $\boldsymbol{D}$ that assigns each decision PRV D_i a range value d_i
  - Refer to $\boldsymbol{d}$ as an action assignment
- E.g., without evidence in G ($\boldsymbol{e} = \emptyset$, $\top$ constraints)
  - Action $Restrict(X)$ with range $\{ban, free\}$
    - $\boldsymbol{d}_1 = \{ban\}$
    - $\boldsymbol{d}_2 = \{free\}$
  - Given another action D with range $\{d', d'', d'''\}$
    - $\boldsymbol{d}_1 = \{ban, d'\}$
    - $\boldsymbol{d}_2 = \{ban, d''\}$
    - $\boldsymbol{d}_3 = \{ban, d'''\}$
    - $\boldsymbol{d}_4 = \{free, d'\}$
    - $\boldsymbol{d}_5 = \{free, d''\}$
    - $\boldsymbol{d}_6 = \{free, d'''\}$

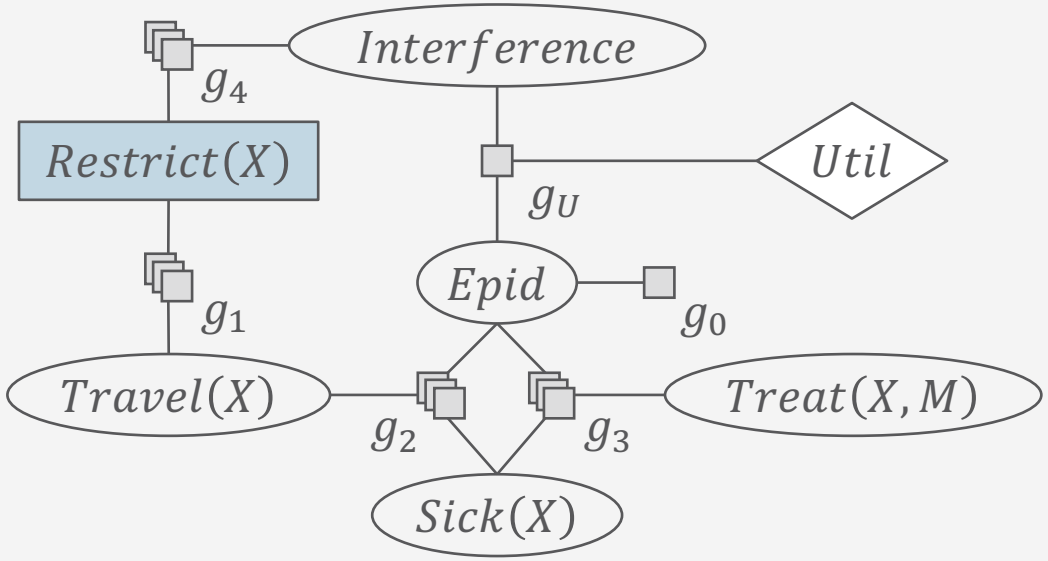


Decision Model: Setting Decisions

- Given a decision model G and an action assignment $\boldsymbol{d}$
- Let $G[\boldsymbol{d}]$ refer to G with $\boldsymbol{d}$ set, i.e.,
$$G[\boldsymbol{d}] = \text{absorb}(G, \boldsymbol{d})$$
  - In each g with decision PRV D_i,
    - Drop the lines where $D_i \neq d_i$ and the column of D_i
- E.g., set $\boldsymbol{d}_1 = \{ban\}$ in $G = \{g_0, g_1, g_2, g_3, g_4, g_U\}$
  - $\boldsymbol{e} = \emptyset$
  - Absorb $\boldsymbol{d}_1$ in g_1
  - $G[\boldsymbol{d}_1] = \{g_0, g_1', g_2, g_3, g_4', g_U\}$
    - $g_1' = \phi_1'(Travel(X))$
    - $g_4' = \phi_4'(Interference)$

$R(X)$	I	ϕ_4
~~free~~	~~false~~	~~1~~
~~free~~	~~true~~	~~0~~
ban	false	0
ban	true	1

$R(X)$	$Tl(X)$	ϕ_1
~~free~~	~~false~~	~~1~~
~~free~~	~~true~~	~~1~~
ban	false	1
ban	true	0



Decision Model: Semantics

- Semantics of decision model $G = \{g_i\}_{i=1}^{n} \cup \{g_U\}$
  - Given an action assignment $\boldsymbol{d}$ for the *grounded* set of decision PRVs $\boldsymbol{D} = \{D_1, \dots, D_k\}_{|C}$ occurring in G
  - Then, the semantics is given by grounding and building a full joint distribution for the non-utility parfactors

$$P_G[\boldsymbol{d}] = \frac{1}{Z} \prod_{f \in \mathrm{gr}(G[\boldsymbol{d}] \setminus \{g_U\})} f$$

$$Z = \sum_{r_1 \in \mathrm{ran}(R_1)} \dots \sum_{r_N \in \mathrm{ran}(R_N)} \prod_{f \in \mathrm{gr}(G[\boldsymbol{d}] \setminus \{g_U\})} f$$

  - Utility parfactors irrelevant for probabilistic behaviour

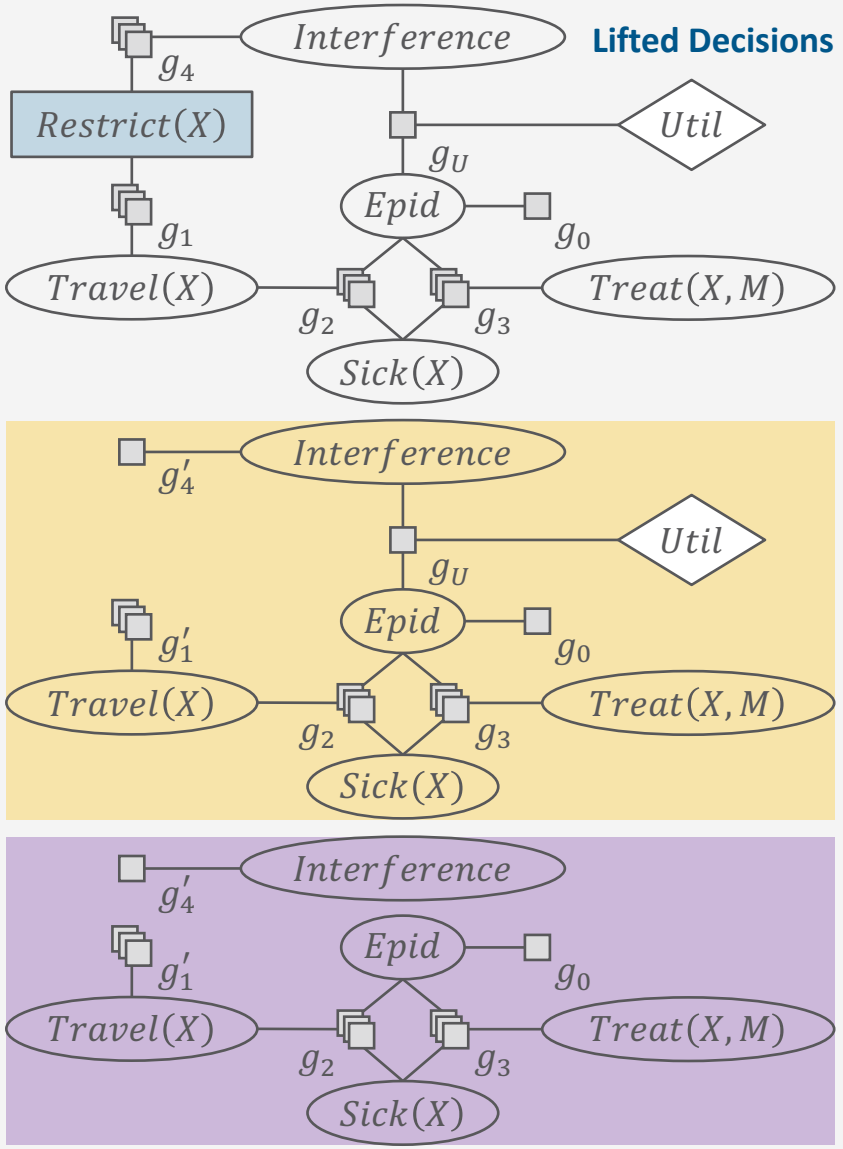
Semantics *multiplicative* with an inner product and outer sum: Multiply parfactors, then sum out PRVs.
→ Sum-product algorithms

Decision Model: Example

- Decision model
$$G = \{g_0, g_1, g_2, g_3, g_4, g_U\}$$
  - T constraints

- G with $\boldsymbol{d}_1 = \{ban\}$ set
$$G[\boldsymbol{d}_1] = \{g_0, g_1', g_2, g_3, g_4', g_U\}$$
  - $g_1' = \phi_1'(Travel(X))$
  - $g_4' = \phi_4'(Interference)$

- Model relevant for probabilistic query answering:
$$G[\boldsymbol{d}_1] \setminus \{g_U\} = \{g_0, g_1', g_2, g_3, g_4'\}$$



Expected Utility Queries

- Given a decision model $G = \{g_i\}_{i=1}^{n} \cup \{g_U\}$
  - One can ask queries for (conditional) marginal distributions or events as before given an action assignment $\boldsymbol{d}$ based on the semantics, $P_G[\boldsymbol{d}]$
  - New query type: query for an expected utility (EU)
    - What is the expected utility of making decisions $\boldsymbol{d}$ in G?

$$eu(\boldsymbol{e}, \boldsymbol{d}) = \sum_{\boldsymbol{r} \in \mathrm{ran}(\mathrm{gr}(\mathrm{rv}(g_U) \setminus \boldsymbol{E} \setminus \boldsymbol{D}))} P(\boldsymbol{r} | \boldsymbol{e}, \boldsymbol{d}) \cdot \phi_U(\boldsymbol{r}, \boldsymbol{e}, \boldsymbol{d})$$

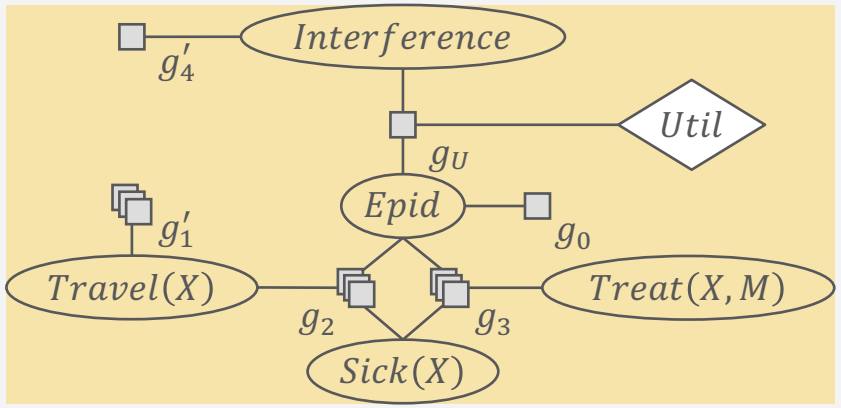
- $P(\boldsymbol{r}|\boldsymbol{e}, \boldsymbol{d})$ means that the PRVs not occurring in this expression need to be eliminated accordingly
  - I.e., $\boldsymbol{V} = \mathrm{rv}(G) \setminus \boldsymbol{D} \setminus \boldsymbol{E} \setminus \mathrm{rv}(g_U)$

EU Query: Example

- Expected utility of $\boldsymbol{d}_1 = \{ban\}$ in $G = \{g_0, g_1, g_2, g_3, g_4, g_U\}$

$$eu(\boldsymbol{d}_1) = \sum_{i \in \mathrm{ran}(Interference)} \sum_{e \in \mathrm{ran}(Epid)} P(e, i | \boldsymbol{d}_1) \cdot \phi_U(e, i)$$

  - With $\boldsymbol{e} = \emptyset$
- Compute $P(Epid, Interference | \boldsymbol{d}_1)$ in G
  - By computing $P(Epid, Interference)$ in $G[\boldsymbol{d}_1]$
    - E.g., using LVE with model $G[\boldsymbol{d}_1] \setminus \{g_U\} = \{g_0, g_1', g_2, g_3, g_4'\}$
    - $G[\boldsymbol{d}_1]$ depicted on the right



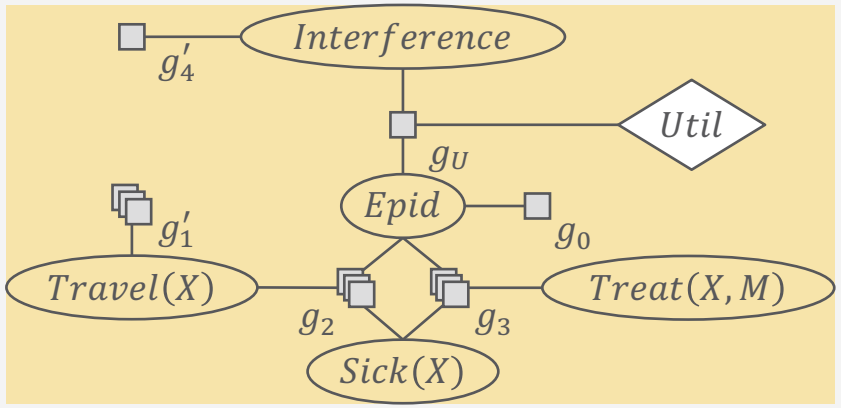
EU Query: Example

- Compute $P(Epid, Interference)$ in $G[\boldsymbol{d}_1] = \{g_0, g_1', g_2, g_3, g_4', g_U\}$
  - Using LVE, eliminate all other PRVs in $G[\boldsymbol{d}_1]$:
    1. Eliminate $Treat(X, M)$
    2. Eliminate $Travel(X)$
    3. Eliminate $Sick(X)$
    4. Multiply all factors and normalise result
       - Result: $P(Epid, Interference)$ in $G[\boldsymbol{d}_1]$: $\phi(Epid, Interference)$
       - Corresponds to $P(Epid, Interference|\boldsymbol{d}_1)$ in G

Parfactors g_1' and g_4' would look differently, had we set $\boldsymbol{d}_2 = \{free\}$.

I	ϕ_4'
$false$	0
$true$	1

$Tl(X)$	ϕ_1'
$false$	1
$true$	0

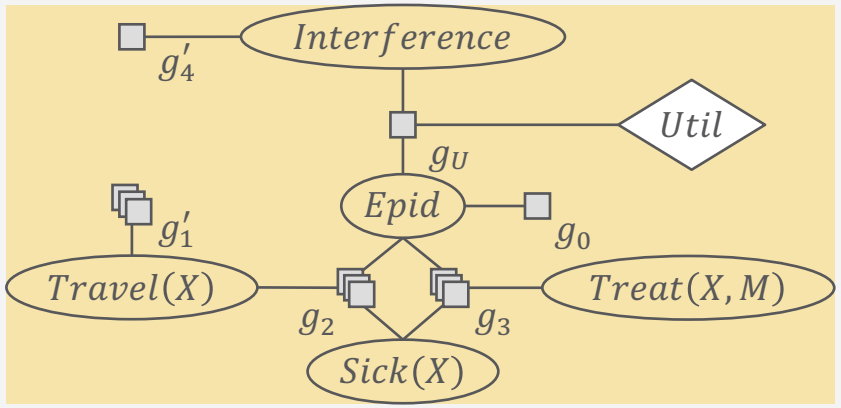


EU Query: Example

- Calculations with $|\mathrm{dom}(M)| = 2, |\mathrm{dom}(X)| = 3$:
  1. Sum out $Treat(X, M)$, exponentiate result for M

E	$S(X)$	$Tt(X,M)$	ϕ_3
$false$	$false$	$false$	9
$false$	$false$	$true$	1
$false$	$true$	$false$	5
$false$	$true$	$true$	6
$true$	$false$	$false$	3
$true$	$false$	$true$	4
$true$	$true$	$false$	4
$true$	$true$	$true$	5

E	$S(X)$	ϕ_3'
$false$	$false$	$(9+1)^2 = 100$
$false$	$true$	$(5+6)^2 = 121$
$true$	$false$	$(3+4)^2 = 49$
$true$	$true$	$(4+5)^2 = 81$



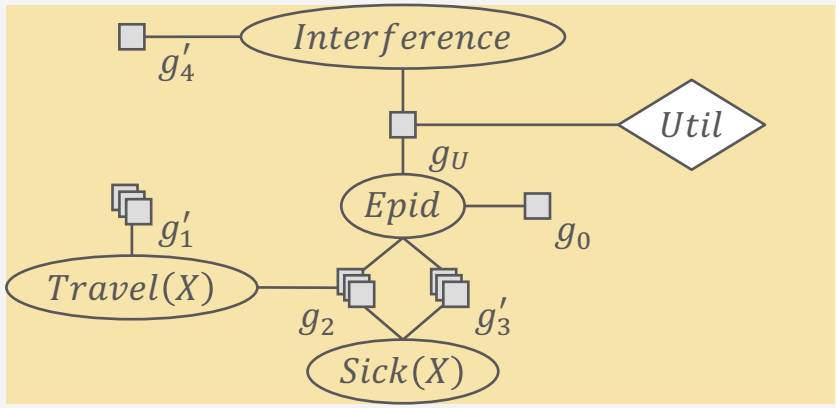
EU Query: Example

- Calculations with $|\mathrm{dom}(M)| = 2, |\mathrm{dom}(X)| = 3$:
  2. Multiply g_1', g_2, sum out $Travel(X)$

E	$S(X)$	$Tl(X)$	$\phi_2 \cdot \phi_1'$
$false$	$false$	$false$	$10 \cdot 1 = 10$
$false$	$false$	$true$	$9 \cdot 0 = 0$
$false$	$true$	$false$	$4 \cdot 1 = 4$
$false$	$true$	$true$	$2 \cdot 0 = 0$
$true$	$false$	$false$	$8 \cdot 1 = 8$
$true$	$false$	$true$	$3 \cdot 0 = 0$
$true$	$true$	$false$	$5 \cdot 1 = 5$
$true$	$true$	$true$	$1 \cdot 0 = 0$

E	$S(X)$	ϕ_{12}'
$false$	$false$	$10 + 0 = 10$
$false$	$true$	$4 + 0 = 4$
$true$	$false$	$8 + 0 = 8$
$true$	$true$	$5 + 0 = 5$

$Travel(X)$	ϕ_1'
$false$	1
$true$	0



EU Query: Example

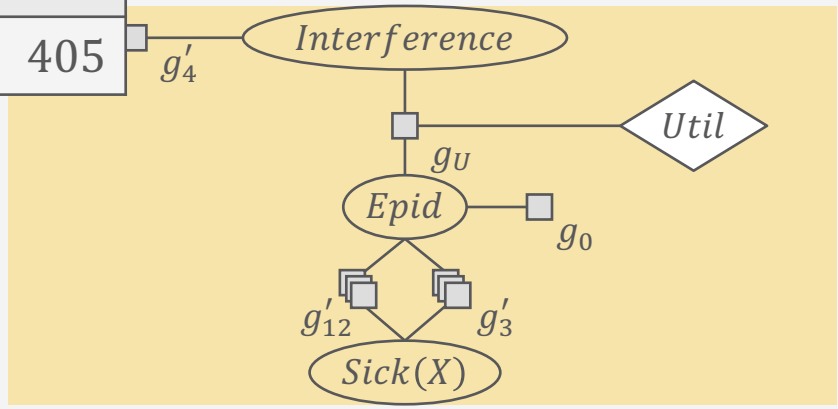
- Calculations with $|\mathrm{dom}(M)| = 2, |\mathrm{dom}(X)| = 3$:
  3. Multiply g'_{12}, g'_3, sum out $Sick(X)$, exponentiate for X

E	$S(X)$	ϕ'_{12}
$false$	$false$	10
$false$	$true$	4
$true$	$false$	8
$true$	$true$	5

E	$S(X)$	ϕ'_3
$false$	$false$	100
$false$	$true$	121
$true$	$false$	49
$true$	$true$	81

E	$S(X)$	$\phi'_{12} \cdot \phi'_3$
$false$	$false$	$10 \cdot 100 = 1000$
$false$	$true$	$4 \cdot 121 = 484$
$true$	$false$	$8 \cdot 49 = 392$
$true$	$true$	$5 \cdot 81 = 405$

E	ϕ'_{123}
$false$	$(1000 + 484)^3 = 3{,}268{,}147{,}904$
$true$	$(392 + 405)^3 = 506{,}261{,}573$



EU Query: Example

- Calculations with $|\mathrm{dom}(M)| = 2, |\mathrm{dom}(X)| = 3$:
  3. Multiply g'_{123}, g_0, g'_4, normalise

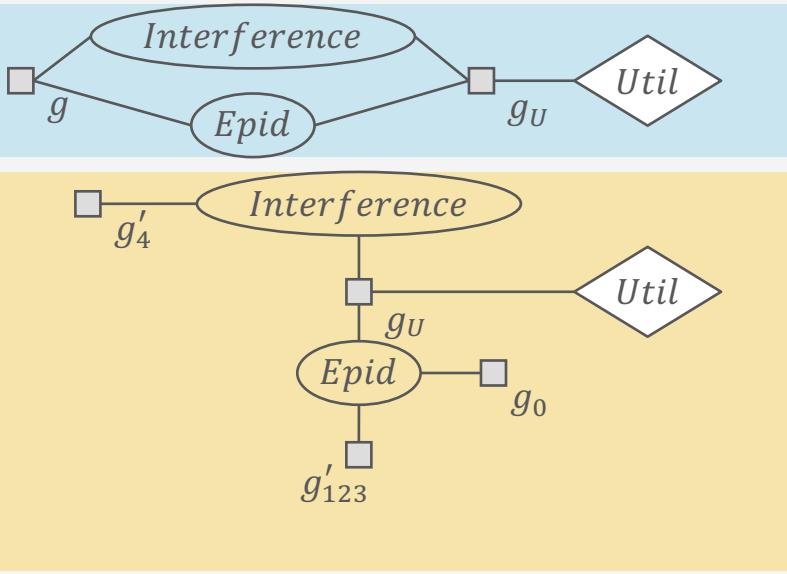
E	ϕ'_{123}
$false$	$(1000 + 484)^3 = 3{,}268{,}147{,}904$
$true$	$(392 + 405)^3 = 506{,}261{,}573$

E	ϕ_0
$false$	10
$true$	1

I	ϕ'_4
$false$	0
$true$	1

I	E	$\phi'_{123} \cdot \phi_0 \cdot \phi'_4$	ϕ
$false$	$false$	$3{,}268{,}147{,}904 \cdot 10 \cdot 0 = 0$	0.000
$false$	$true$	$506{,}261{,}573 \cdot 1 \cdot 0 = 0$	0.000
$true$	$false$	$3{,}268{,}147{,}904 \cdot 10 \cdot 1 = 30{,}268{,}147{,}904$	0.984
$true$	$true$	$506{,}261{,}573 \cdot 1 \cdot 1 = 506{,}261{,}573$	0.016

Result after normalising:
$g = \phi(Interference, Epid)$



EU Query: Example

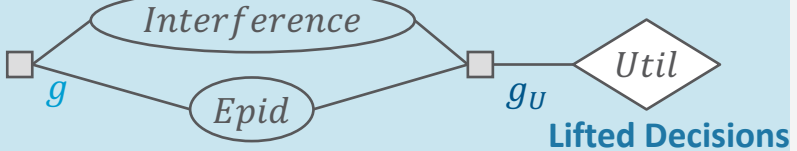
- Result $\phi(Epid, Interference)$ for $P(e, i|\boldsymbol{d}_1)$ in G
- Expected utility of $\boldsymbol{d}_1 = \{ban\}$ in $G = \{g_0, g_1, g_2, g_3, g_4, g_U\}$

$$eu(\boldsymbol{d}_1) = \sum_{i \in \mathrm{ran}(Interference)} \sum_{e \in \mathrm{ran}(Epid)} P(e, i|\boldsymbol{d}_1) \cdot \phi_U(e, i)$$

$$= \sum_{i \in \mathrm{ran}(Interference)} \sum_{e \in \mathrm{ran}(Epid)} \phi(e, i) \cdot \phi'_U(e, i)$$

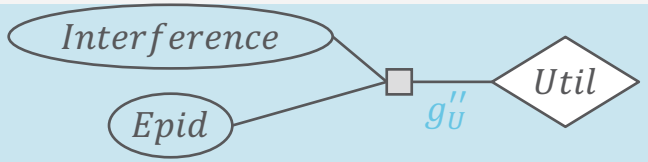
$$= \sum_{i \in \mathrm{ran}(Interference)} \sum_{e \in \mathrm{ran}(Epid)} \phi''_U(Epid = e)$$

$$= \phi'''_U(.)$$



Lifted Decisions

I	E	ϕ	$Util$
$false$	$false$	0.000	10
$false$	$true$	0.000	−10
$true$	$false$	0.984	−20
$true$	$true$	0.016	2



I	E	$Util$
$false$	$false$	0.000 · 10 = 0.000
$false$	$true$	0.000 · (−10) = 0.000
$true$	$false$	0.984 · (−20) = −19.680
$true$	$true$	0.016 · 2 = 0.032

.	$Util$
.	0 + 0 − 19.680 + 0.032 = −19.360

Answering EU-Queries (with LVE)

- Given a decision model $G = \{g_i\}_{i=1}^{n} \cup \{g_U\}$, evidence $\boldsymbol{e}$, and an action assignment $\boldsymbol{d}$ (*)
  - Absorb $\boldsymbol{e}$ and $\boldsymbol{d}$ in G
  - Calculate the posterior, $P(\boldsymbol{R}|\boldsymbol{e},\boldsymbol{d})$, of the *Markov blanket of the utility node*
    - I.e., $\boldsymbol{R} = \mathrm{rv}(g_U) \setminus \mathrm{rv}(\boldsymbol{d}) \setminus \mathrm{rv}(\boldsymbol{e})$ (remaining PRVs in g_U after previous step)
    - Using LVE: With $\boldsymbol{R}$ as the query terms, eliminate all non-query terms in G, i.e., call $\mathrm{LVE}(G \setminus \{g_U\}, \boldsymbol{R}, \emptyset)$
      - Evidence already absorbed, decisions made $\rightarrow \boldsymbol{e} = \emptyset$ in the call
  - Calculate the expected utility by summing over the range values of $\boldsymbol{R}$:

$$eu(\boldsymbol{e},\boldsymbol{d}) = \sum_{\boldsymbol{r} \in \mathrm{ran}(\boldsymbol{R})} P(\boldsymbol{r}|\boldsymbol{e},\boldsymbol{d}) \cdot \phi_U(\boldsymbol{r})$$

    - Using LVE: Eliminate remaining PRVs in G
      - Result: parfactor mapping empty argument to a single value (U)

(*) We need to talk about evidence and action assignments later.

MEU Problem

- Given a decision model G and evidence $\boldsymbol{e}$, maximum Expected Utility (MEU) problem:
  - Find the action assignment that yields the highest expected utility in G
  - Formally,

$$\mathrm{meu}(G|\boldsymbol{e}) = \left(\boldsymbol{d}^*, eu(\boldsymbol{e}, \boldsymbol{d}^*)\right)$$

$$\boldsymbol{d}^* = \arg\max_{\boldsymbol{d} \in \mathrm{ran}(\boldsymbol{D})} eu(\boldsymbol{e}, \boldsymbol{d})$$

Additive semantics with inner sum and outer max: Sum up utilities, then pick maximum
→ Max-sum algorithms

  - For an exact solution, $\mathrm{meu}(G|\boldsymbol{e})$ requires an algorithm to go through all $\boldsymbol{d} \in \mathrm{ran}(\boldsymbol{D})$
    - Size of $\mathrm{ran}(\boldsymbol{D})$ exponential in $|\mathrm{gr}(\boldsymbol{D})|$

Alternative specification

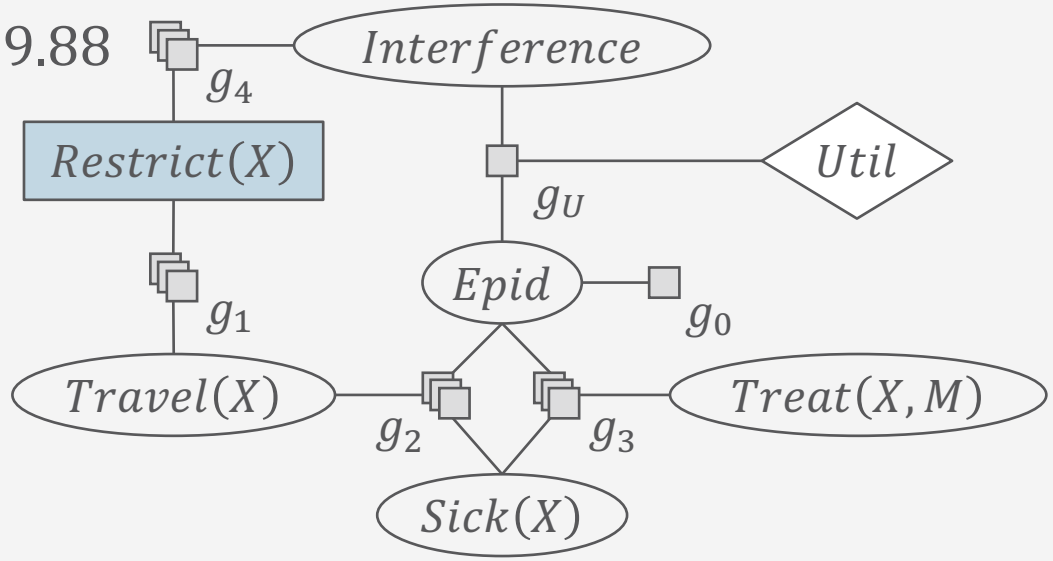
$$\mathrm{meu}(G|\boldsymbol{e}) = \left(\arg\max_{\boldsymbol{d} \in \mathrm{ran}(\boldsymbol{D})} eu(\boldsymbol{e}, \boldsymbol{d}), \max_{\boldsymbol{d} \in \mathrm{ran}(\boldsymbol{D})} eu(\boldsymbol{e}, \boldsymbol{d})\right)$$

MEU Problem: Example

- Problem instance with $G = \{g_0, g_1, g_2, g_3, g_U\}$, $\boldsymbol{e} = \emptyset$:

$$\text{meu}(G) = \left(\boldsymbol{d}^*, eu(\boldsymbol{d}^*)\right) \quad \boldsymbol{d}^* = \arg\max_{\boldsymbol{d} \in \{\boldsymbol{d}_1, \boldsymbol{d}_2\}} eu(\boldsymbol{d})$$

  - $\boldsymbol{d}_1 = \{ban\}, \boldsymbol{d}_2 = \{free\}$
  - Expected utility of $\boldsymbol{d}_1 = \{ban\}$: $eu(\boldsymbol{d}_1) = -19.36$
  - Expected utility of $\boldsymbol{d}_2 = \{free\}$: $eu(\boldsymbol{d}_2) = 9.88$
- Solution
  - $\boldsymbol{d}^* = \underset{\boldsymbol{d} \in \{\boldsymbol{d}_1, \boldsymbol{d}_2\}}{\text{argmax}}\, eu(\boldsymbol{d}) = \boldsymbol{d}_2$
  - $\text{meu}(G) = (\boldsymbol{d}_2, 9.88)$
  - Decision that leads to maximum EU: No travel restrictions



Lifted MEU

$$\text{meu}(G|\boldsymbol{e}) = \left(\boldsymbol{d}^*, eu(\boldsymbol{e}, \boldsymbol{d}^*)\right)$$
$$\boldsymbol{d}^* = \arg\max_{\boldsymbol{d} \in \text{ran}(\boldsymbol{D})} eu(\boldsymbol{e}, \boldsymbol{d})$$

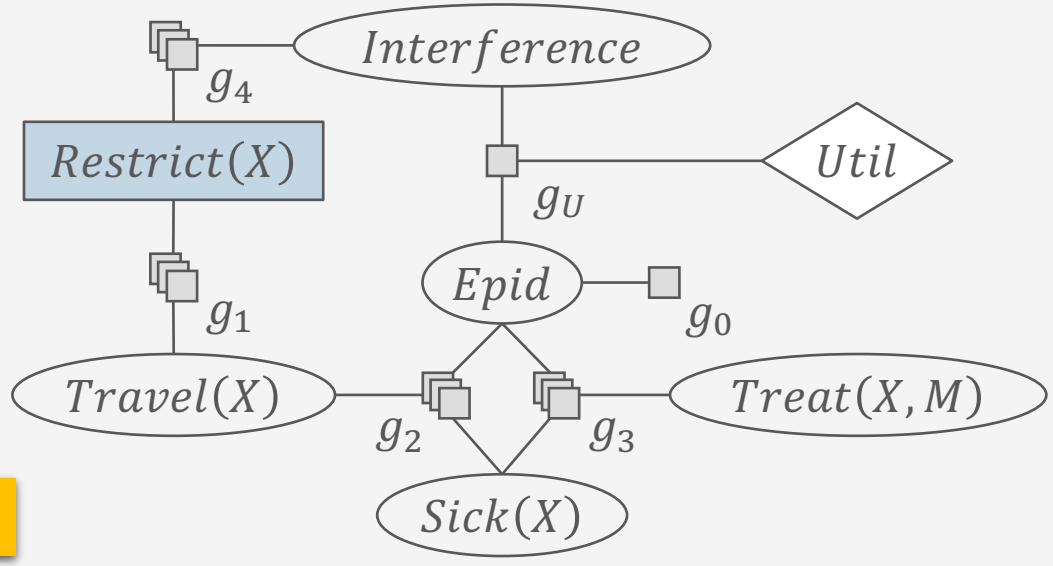
- In terms of semantics, $\boldsymbol{d} \in \text{ran}(\boldsymbol{D})$ means
  - Grounding $\boldsymbol{D}$ and going through all possible combinations of assignments to $gr(\boldsymbol{D})$
- But: grounding is bad!
  - Combinatorial explosion: number of action assignments to test exponential in size of $gr(\boldsymbol{D})$
  - Grounds any parfactor in G containing a logvar of $\boldsymbol{D}$
- Also: Grounding to full extent often unnecessary
  - Within groups of indistinguishable constants, the same decision will lead to its maximum influence in the MEU solution
    - Only need to test each assignment for complete group
- Thus: Test out all possible combinations of assignments w.r.t. the groups occurring in G
  - No longer exponential in the size of $gr(\boldsymbol{D})$!

$$\text{meu}(G|\boldsymbol{e}) = \left(\boldsymbol{d}^*, eu(\boldsymbol{e}, \boldsymbol{d}^*)\right)$$
$$\boldsymbol{d}^* = \arg\max_{\boldsymbol{d} \in \text{ran}(\boldsymbol{D})} eu(\boldsymbol{e}, \boldsymbol{d})$$

Lifted MEU: Groups

- In parameterised models without evidence (or evidence for complete domains), $\boldsymbol{d} \in \text{ran}(\boldsymbol{D})$ means
  - Going through all possible combinations of assignments to $\boldsymbol{D}$
    - One group per logical variable
- In models with evidence affecting parfactors containing decision PRVs, $\boldsymbol{d} \in \text{ran}(\boldsymbol{D})$ means
  - Going through all possible combinations of assignments for each group of constants after evidence handling
    - Specifically, after shattering
- Effect: size exponential in number of groups

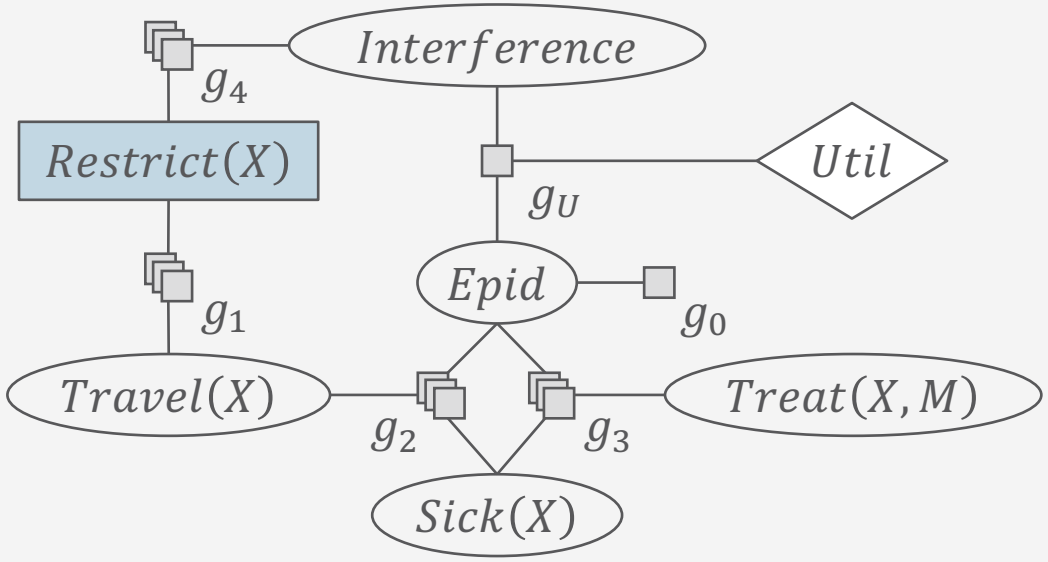
(*) Now is later.



$$\text{meu}(G|\boldsymbol{e}) = \left(\boldsymbol{d}^*, eu(\boldsymbol{e}, \boldsymbol{d}^*)\right)$$
$$\boldsymbol{d}^* = \arg\max_{\boldsymbol{d} \in \text{ran}(\boldsymbol{D})} eu(\boldsymbol{e}, \boldsymbol{d})$$

Lifted MEU: Groups – Example

- Decision model $G = \{g_0, g_1, g_2, g_3, g_4, g_U\}$
  - Decision PRV $Restrict(X)$ with range $\{ban, free\}$
  - Evidence $\boldsymbol{e} = \{Sick(X') = true\}, \text{dom}(X') = \{x_1, \dots, x_{10}\}$
  - Overlap in domain of $X, X' \rightarrow$ Shattering duplicates g_1, g_2, g_3, g_4
    - For $\text{dom}(X') = \{x_1, \dots, x_{10}\}$, $\text{dom}(X'') = \{x_{10}, \dots, x_n\}$
    - Alternative: Duplicate + restrict constraints
- Action assignments
  - $R \triangleq Restrict, b \triangleq ban, f \triangleq free$
  - $\boldsymbol{d}_1 = \{R(X'') = b, R(X') = b\}$
  - $\boldsymbol{d}_2 = \{R(X'') = b, R(X') = f\}$
  - $\boldsymbol{d}_3 = \{R(X'') = f, R(X') = b\}$
  - $\boldsymbol{d}_4 = \{R(X'') = f, R(X') = f\}$



Answering EU-Queries for MEU

- Given a decision model $G = \{g_i\}_{i=1}^{n} \cup \{g_U\}$, evidence $\boldsymbol{e}$, and an action assignment $\boldsymbol{d}$ for groups in G after shattering
  1. Calculate the posterior, $P(\boldsymbol{R}|\boldsymbol{e},\boldsymbol{d})$, of the *Markov blanket of the utility node*
     - I.e., $\boldsymbol{R} = rv(g_U) \setminus rv(\boldsymbol{a}) \setminus rv(\boldsymbol{E})$ (remaining PRVs in g_u's after previous step)
     - Using LVE: With $\boldsymbol{R}$ as the query terms and $\boldsymbol{e},\boldsymbol{d}$ as evidence, eliminate all non-query terms in G, i.e., call
       $$\mathrm{LVE}(G \setminus \{g_U\}, \boldsymbol{R}, \boldsymbol{e} \cup \boldsymbol{d})$$
  2. Calculate the expected utility by summing over the range values of $\boldsymbol{R}$:
     $$eu(\boldsymbol{e},\boldsymbol{d}) = \sum_{\boldsymbol{r} \in \mathrm{ran}(\boldsymbol{R})} P(\boldsymbol{r}|\boldsymbol{e},\boldsymbol{d}) \cdot \phi_U(\boldsymbol{r})$$
     - Using LVE: Eliminate remaining PRVs in $\{g\} \cup \{g_U\}, g = \mathrm{LVE}(G \setminus \{g_U\}, \boldsymbol{R}, \boldsymbol{e} \cup \boldsymbol{d})$, i.e., call
       $$\mathrm{LVE}(\{g\} \cup \{g_U\}, \boldsymbol{R}, \boldsymbol{e} \cup \boldsymbol{d})$$
       - $\boldsymbol{e},\boldsymbol{d}$ not yet handled in g_U; alternatively: absorb $\boldsymbol{e},\boldsymbol{d}$ at beginning in G
       - Result: parfactor mapping empty argument to a single value (U)

LVE for MEU Problems

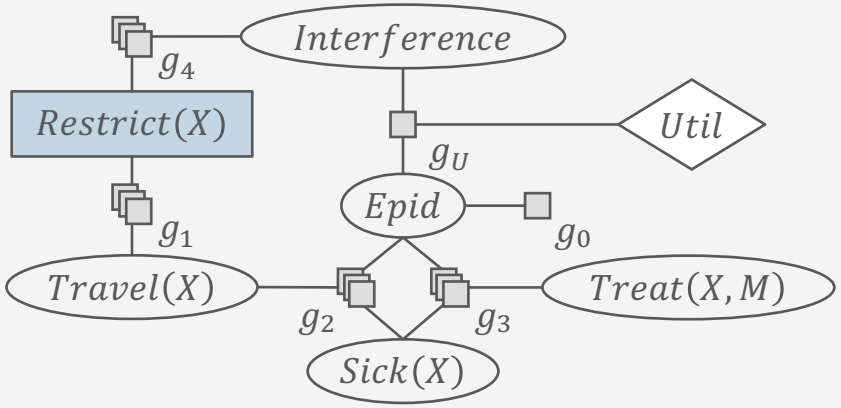
function MEU–LVE$(G = \{g_i\}_{i=1}^{n} \cup \{g_U\}, \boldsymbol{e})$
    Absorb $\boldsymbol{e}$ in G
    $\boldsymbol{d}^* \leftarrow \emptyset$
    $eu_{max} \leftarrow -\infty$
    for each action assignment $\boldsymbol{d}$ in G **do**
        $g \leftarrow \mathrm{LVE}(G \setminus \{g_U\}, rv(g_U), \boldsymbol{d})$ ▷ g normalised
        $eu \leftarrow \mathrm{LVE}(\{g_U, g\}, \emptyset, \boldsymbol{d})$
        if $eu > eu_{max}$ **then**
            $\boldsymbol{d}^* \leftarrow \boldsymbol{d}$
            $eu_{max} \leftarrow eu$
    return $\boldsymbol{d}^*$

LVE-MEU

- Modify to save all assignments that lie within ε-margin

Structure in Multi-attribute Settings

- So far: Set of attributes without structure
  - Single utility functions mapping to one utility
    - Example: $\phi_U(Interference, Epid)$
- Cases with structure:
  1. **Set of (distinguishable) attributes with structure**
     - Set of utility functions, mapping to interim utilities, combined into one overall utility
  2. **Set of indistinguishable attributes**
     - Utility parfactor mapping to an interim utility PRV, which is combined into one utility
  3. **Sets of distinguishable and indistinguishable attributes**
     - Set of utility parfactors and utility factors, combined into one utility
  - Considering structure requires a combination function Ξ

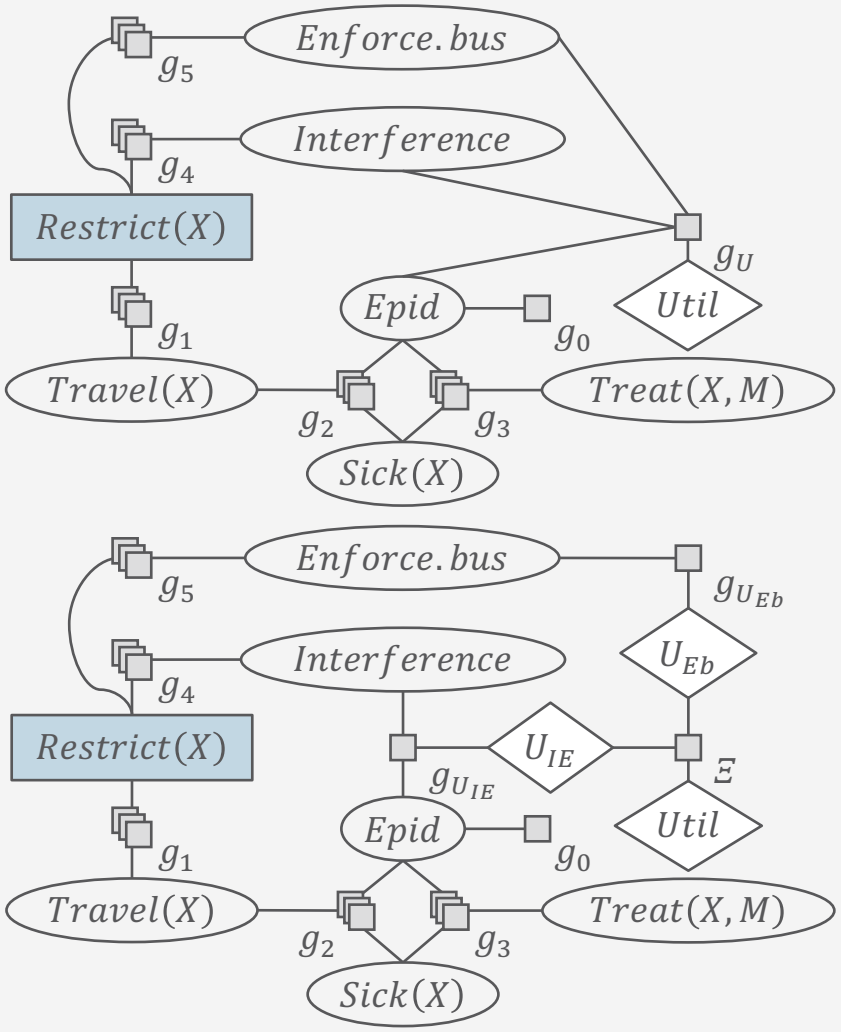


1. Set of Attributes with Structure

- Set of attributes that show MPI → Utility function ”factorises” into sets of functions over attributes, combined with a combination function Ξ, i.e.,
$$U(r_1, \dots, r_M) = \Xi[\phi_1(r_1), \dots, \phi_M(r_M)]$$
  - I.e., each $\phi_i(r_i)$ maps to its own interim utility, U_i, combined into an overall utility U through Ξ
  - More general: Each f_i has a set of random variables $\boldsymbol{r}_i$ as input with $\boldsymbol{r} = \{r_1, \dots, r_M\} = \bigcup_{i=1}^{m} \boldsymbol{r}_i$
- Extended syntax: Decision model
$$G = \{g_i\}_{i=1}^{n} \cup \{g_u\}_{u=1}^{m} \cup \{\Xi\}$$
  - Refer to submodel of potential parfactors by G_P and to submodel of utility factors by G_U
  - $g_i = \phi_i(\mathcal{A}_i)_{|C_i}$ parfactors with (decision) PRVs as arguments
  - $g_u = \phi_{U_u}(\mathcal{R}_u)$ utility *factors*, each mapping to a utility variable U_u
  - Ξ a combination function, combining all U_u into one U, i.e.,
$$\phi_U(r_1, \dots, r_M) = \Xi\left[\phi_{U_1}\left(\pi_{\mathcal{R}_1}(r_1, \dots, r_M)\right), \dots, \phi_{U_m}\left(\pi_{\mathcal{R}_m}(r_1, \dots, r_M)\right)\right]$$

1. Set of Attributes with Structure: Example

- Example:
  - $\phi_{U_{IE}}(Interference, Epid)$
    utility factor over $Interference, Epid$
  - $\phi_{U_{Eb}}(Enforce.bus)$
    utility factor over $Enforce.bus$
    - (Effort it takes to enforce travel restriction on busses)
  - Ξ a combination function, combining U_1, U_2 into $Util$
    - Could rewrite model using Ξ into a model containing only one utility factor g_U (shown above)
      - $\phi_U(Interference, Epid, Enforce.bus)$
        $= \Xi[\phi_{U_{IE}}(Interference, Epid), \phi_{U_{Eb}}(Enforce.bus)]$



1. Set of Attributes with Structure: EU Query & MEU Problem

- Given a decision model $G = G_P \cup G_U \cup \{\Xi\} = \{g_i\}_{i=1}^{n} \cup \{g_u\}_{u=1}^{m} \cup \{\Xi\}$
  - Query for an expected utility (EU): change in sum over $\mathrm{rv}(G_U)$ instead of $\mathrm{rv}(g_U)$

$$eu(\boldsymbol{e}, \boldsymbol{d}) = \sum_{\boldsymbol{v} \in \mathrm{ran}(\mathrm{rv}(G_U) \setminus \boldsymbol{E} \setminus \boldsymbol{D})} P(\boldsymbol{v}|\boldsymbol{e}, \boldsymbol{d}) \cdot \Xi\left[\phi_{U_1}\left(\pi_{\mathcal{R}_1}(\boldsymbol{v}, \boldsymbol{e}, \boldsymbol{d})\right), \ldots, \phi_{U_m}\left(\pi_{\mathcal{R}_m}(\boldsymbol{v}, \boldsymbol{e}, \boldsymbol{d})\right)\right]$$

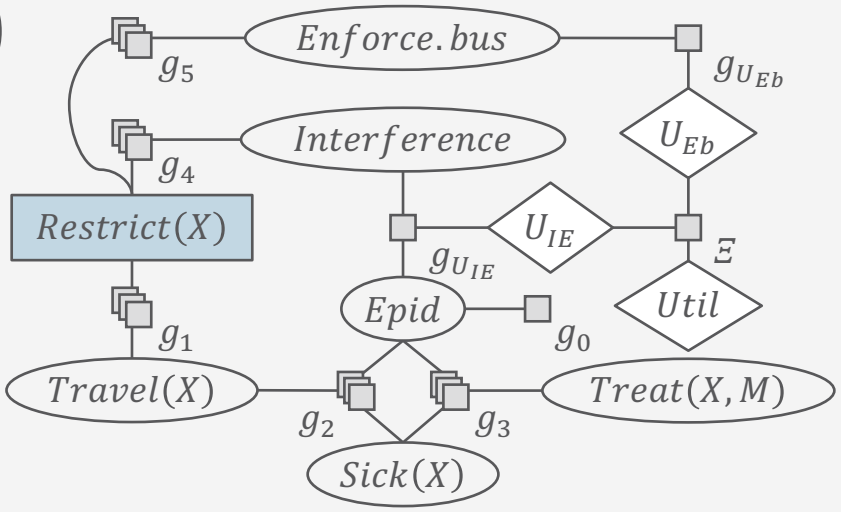
    - If Ξ addition, then

$$eu(\boldsymbol{e}, \boldsymbol{d}) = \sum_{\boldsymbol{v} \in \mathrm{ran}(\mathrm{gr}(\mathrm{rv}(G_U) \setminus \boldsymbol{E} \setminus \boldsymbol{D}))} P(\boldsymbol{v}|\boldsymbol{e}, \boldsymbol{d}) \cdot \sum_{g_u \in G_U} \phi_{U_u}\left(\pi_{\mathcal{R}_u}(\boldsymbol{v}, \boldsymbol{e}, \boldsymbol{d})\right)$$

      - Works like MULTIPLY, i.e., like a join, but with *summing* of utilities instead of multiplying of potentials

  - MEU problem: *no changes*

$$\mathrm{meu}(G|\boldsymbol{e}) = \left(\boldsymbol{d}^*, eu(\boldsymbol{e}, \boldsymbol{d}^*)\right)$$
$$\boldsymbol{d}^* = \underset{\boldsymbol{d} \in \mathrm{ran}(\boldsymbol{D})}{\mathrm{argmax}}\, eu(\boldsymbol{e}, \boldsymbol{d})$$



1. Set of Attributes with Structure: Additive Join

- Operator:

Operator 1 Additive join of utility factors

Operator ADD
Inputs:
(1) Utility factor $f_{u'} = \phi_{u'}(\mathcal{R}_{u'})$
(2) Utility factor $f_{u''} = \phi_{u''}(\mathcal{R}_{u''})$
Output: Utility factor $\phi_u(\mathcal{R}_u)$ such that
(1) $\mathcal{R}_u = \mathcal{R}_{u'} \bowtie \mathcal{R}_{u''}$ and
(2) for each valuation $\boldsymbol{r} \in ran(\mathcal{R}_u)$ with $\boldsymbol{r}_{u'} = \pi_{\mathcal{R}_{u'}}(\boldsymbol{r})$ and $\boldsymbol{r}_{u''} = \pi_{\mathcal{R}_{u''}}(\boldsymbol{r})$

$$\phi_u(\boldsymbol{r}) = \phi_{u'}(\boldsymbol{r}_{u'}) + \phi_{u''}(\boldsymbol{r}_{u''})$$

Postcondition: $G_U \equiv G_U \setminus \{f_{u'}, f_{u''}\} \cup \text{ADD}(f_{u'}, f_{u''})$

- Example

$$\begin{aligned}
&\phi_U(Interference, Epid, Enforce.bus) \\
&= \Xi\left[\phi_{U_{IE}}(Interference, Epid), \phi_{U_{Eb}}(Enforce.bus)\right] \\
&= \phi_{U_{IE}}(Interference, Epid) + \phi_{U_{Eb}}(Enforce.bus) \\
&= \text{add}\left(g_{U_{IE}}, g_{U_{Eb}}\right)
\end{aligned}$$

I	E	U_{IE}
$false$	$false$	10
$false$	$true$	-10
$true$	$false$	-20
$true$	$true$	2

Eb	U_{Eb}
$false$	0
$true$	-10

I	E	Eb	U_{IE}
$false$	$false$	$false$	$10 + 0 = 10$
$false$	$false$	$true$	$10 - 10 = 0$
$false$	$true$	$false$	$-10 + 0 = -10$
$false$	$true$	$true$	$-10 - 10 = -20$
$true$	$false$	$false$	$-20 + 0 = -20$
$true$	$false$	$true$	$-20 - 10 = -30$
$true$	$true$	$false$	$2 + 0 = 2$
$true$	$true$	$true$	$2 - 10 = 8$

U_{Eb}

$M)$

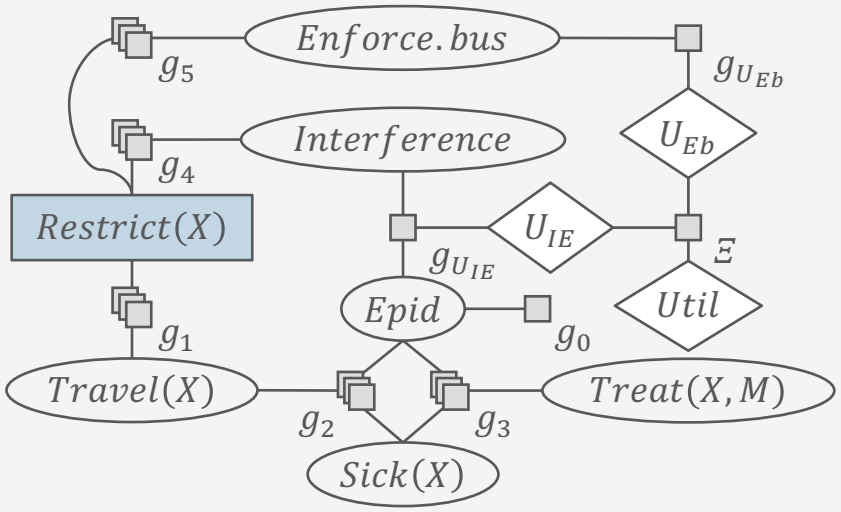
1. Set of Attributes with Structure: MEU-LVE

- Implement ADD operator
  - LVE with ADD operator referred to as LVE^{ADD}
- Changes in MEU-LVE
  - Input: decision model $G = G_P \cup G_U = \{g_i\}_{i=1}^{n} \cup \{g_u\}_{u=1}^{m}$
  - In for-loop:

$$g \leftarrow \text{LVE}(M \setminus G_U, \text{rv}(G_U), \boldsymbol{d}) \quad \triangleright\ g \text{ normalised}$$
$$eu \leftarrow \text{LVE}^{\text{ADD}}(G_U \cup \{g\}, \emptyset, \boldsymbol{d})$$

    - If Ξ not addition, need to implement (change LVE^{ADD} call)
    - Combines G_U into one g_U before multiplying with g and summing out the remaining variables

Splitting a single utility function into set of utility factors has upside of needing to learn / specify fewer entries BUT:
Complexity still exponential in M as combined into g



1. Set of Attributes with Structure: Simplification

- Assume (conditional) independence between the different $\mathrm{rv}(f_u)$ given $\boldsymbol{e}, \boldsymbol{d}$, i.e., $P(\boldsymbol{v}|\boldsymbol{e},\boldsymbol{d}) = \prod_{u'=1}^{m} P(\boldsymbol{r}_{u'}|\boldsymbol{e},\boldsymbol{d})$

$$eu(\boldsymbol{e},\boldsymbol{d}) = \sum_{\boldsymbol{v}\in\mathrm{ran}(\boldsymbol{V})} P(\boldsymbol{v}|\boldsymbol{e},\boldsymbol{d}) \cdot \Xi\left[\phi_{U_1}(\boldsymbol{r}_1), \ldots, \phi_{U_m}(\boldsymbol{r}_m)\right]$$

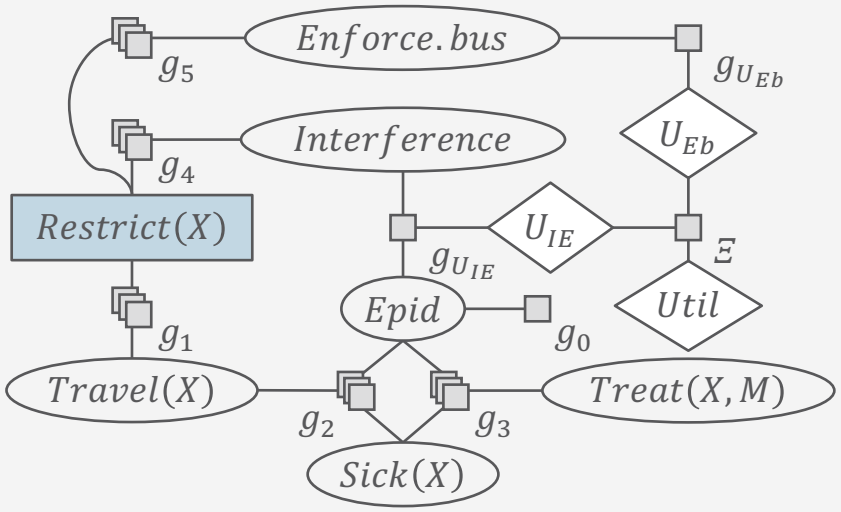
$$= \sum_{\boldsymbol{v}\in\mathrm{ran}(\boldsymbol{V})} P(\boldsymbol{v}|\boldsymbol{e},\boldsymbol{d}) \cdot \sum_{u=1}^{m} \phi_u(\boldsymbol{r}_u)$$

$$= \sum_{u=1}^{m} \sum_{\boldsymbol{r}_u\in\mathrm{ran}(\mathrm{rv}(g_u))} P(\boldsymbol{r}_u|\boldsymbol{e},\boldsymbol{d}) \cdot \phi_u(\boldsymbol{r}_u)$$

Query on $\mathrm{rv}(f_u)$ for each utility factor
→ Use multi-query algorithm like LJT

Only yields correct result under stochastic independence

- Idea similar to Boyen-Koller algorithm
- Preferential and stochastic independence do not follow from each other!



Derivation

$$eu(\boldsymbol{e},\boldsymbol{d}) = \sum_{\boldsymbol{v}\in\mathrm{ran}(\boldsymbol{V})} P(\boldsymbol{v}|\boldsymbol{e},\boldsymbol{d}) \cdot \sum_{u=1}^{m} \phi_u(\boldsymbol{r}_u) = \sum_{\boldsymbol{v}\in\mathrm{ran}(\boldsymbol{V})} \sum_{u=1}^{m} P(\boldsymbol{v}|\boldsymbol{e},\boldsymbol{d}) \cdot \phi_u(\boldsymbol{r}_u) = \sum_{u=1}^{m} \sum_{\boldsymbol{v}\in\mathrm{ran}(\boldsymbol{V})} P(\boldsymbol{v}|\boldsymbol{e},\boldsymbol{d}) \cdot \phi_u(\boldsymbol{r}_u)$$

$$= \sum_{u=1}^{m} \sum_{\boldsymbol{v}\in\mathrm{ran}(\boldsymbol{V})} \prod_{u'=1}^{m} P(\boldsymbol{r}_{u'}|\boldsymbol{e},\boldsymbol{d}) \cdot \phi_u(\boldsymbol{r}_u)$$

$$= \sum_{u=1}^{m} \sum_{\boldsymbol{r}_1\in\mathrm{ran}(\boldsymbol{R}_1)} \dots \sum_{\boldsymbol{r}_m\in\mathrm{ran}(\boldsymbol{R}_m)} P(\boldsymbol{r}_1|\boldsymbol{e},\boldsymbol{d}) \cdot \dots \cdot P(\boldsymbol{r}_m|\boldsymbol{e},\boldsymbol{d}) \cdot \phi_u(\boldsymbol{r}_u)$$

$$= \sum_{u=1}^{m} \sum_{\boldsymbol{r}_1\in\mathrm{ran}(\boldsymbol{R}_1)} P(\boldsymbol{r}_1|\boldsymbol{e},\boldsymbol{d}) \cdot \dots \cdot \sum_{\boldsymbol{r}_m\in\mathrm{ran}(\boldsymbol{R}_m)} P(\boldsymbol{r}_m|\boldsymbol{e},\boldsymbol{d}) \cdot \phi_u(\boldsymbol{r}_u)$$

$$= \sum_{u=1}^{m} \sum_{\boldsymbol{r}_u\in\mathrm{ran}(\boldsymbol{R}_u)} P(\boldsymbol{r}_u|\boldsymbol{e},\boldsymbol{d}) \cdot \phi_u(\boldsymbol{r}_u) \cdot \underbrace{\sum_{u'=1,u'\neq u}^{m} P(\boldsymbol{r}_{u'}|\boldsymbol{e},\boldsymbol{d})}_{=1}$$

$$= \sum_{u=1}^{m} \sum_{\boldsymbol{r}_u\in\mathrm{ran}(\boldsymbol{R}_u)} P(\boldsymbol{r}_u|\boldsymbol{e},\boldsymbol{d}) \cdot \phi_u(\boldsymbol{r}_u)$$

(probability distributions → sums to 1)

1. Set of Attributes with Structure: Simplification – Example

- Example: $\boldsymbol{d}_1 = \{ban\}$
  - $P(E, I, Eb|\boldsymbol{d}) = P(E|\boldsymbol{d}) \cdot P(I|\boldsymbol{d}) \cdot P(Eb|\boldsymbol{d})$

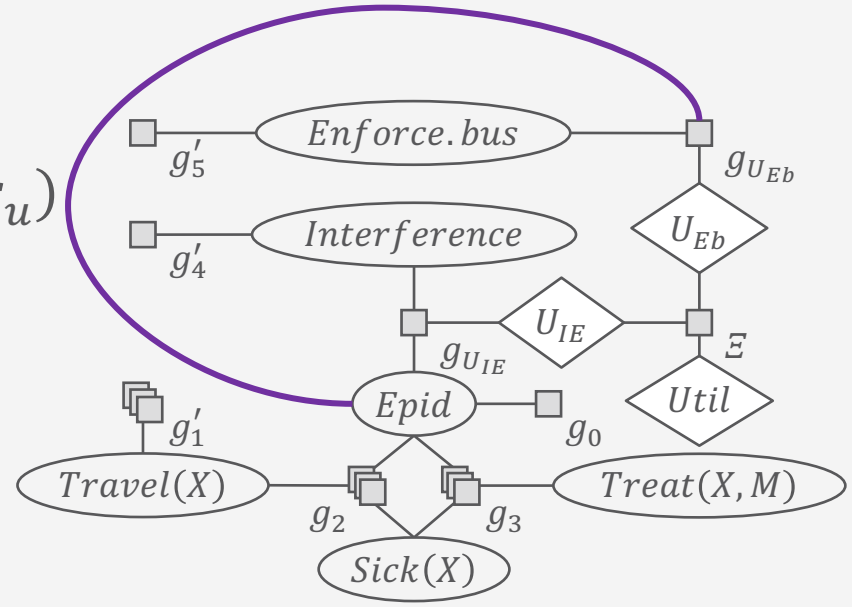
$$eu(\boldsymbol{d}_1)$$

$$= \sum_{eb \in \mathrm{ran}(Eb)} \sum_{i \in \mathrm{ran}(I)} \sum_{e \in \mathrm{ran}(E)} P(eb, i, e|\boldsymbol{d}) \cdot \sum_{u=1}^{2} \phi_u(\boldsymbol{r}_u)$$

$$= \sum_{eb \in \mathrm{ran}(Eb)} \sum_{i \in \mathrm{ran}(I)} \sum_{e \in \mathrm{ran}(E)} P(eb|\boldsymbol{d}) \cdot P(i|\boldsymbol{d}) \cdot P(e|\boldsymbol{d}) \cdot \sum_{u=1}^{2} \phi_u(\boldsymbol{r}_u)$$

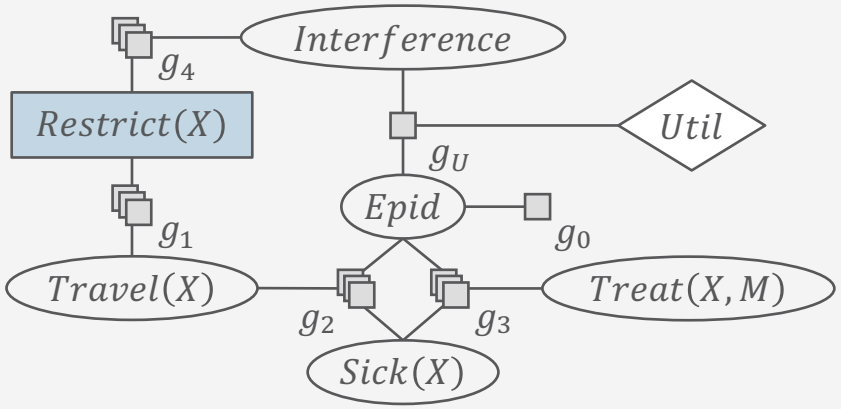
$$= \sum_{eb \in \mathrm{ran}(Eb)} P(eb|\boldsymbol{d}) \cdot \phi_{Eb}(eb) + \sum_{i \in \mathrm{ran}(I)} P(i|\boldsymbol{d}) \cdot \sum_{e \in \mathrm{ran}(E)} P(e|\boldsymbol{d}) \cdot \phi_{IE}(i, e)$$

If adding $Epid$ as an input to $g_{U_{Eb}}$,
$P(E, I, Eb|\boldsymbol{d}) \neq P(E, Eb|\boldsymbol{d}) \cdot P(E, I|\boldsymbol{d})$



Structure in Multi-attribute Settings

- So far: Set of attributes without structure
  - Single utility functions mapping to one utility
    - Example: $\phi_U(Interference, Epid)$
- Cases with structure:
  1. **Set of (distinguishable) attributes with structure**
     - Set of utility functions, mapping to interim utilities, combined into one overall utility
  2. **Set of indistinguishable attributes**
     - Utility parfactor mapping to an interim utility PRV, which is combined into one utility
  3. **Sets of distinguishable and indistinguishable attributes**
     - Set of utility parfactors and utility factors, combined into one utility
  - Considering structure requires a combination function Ξ

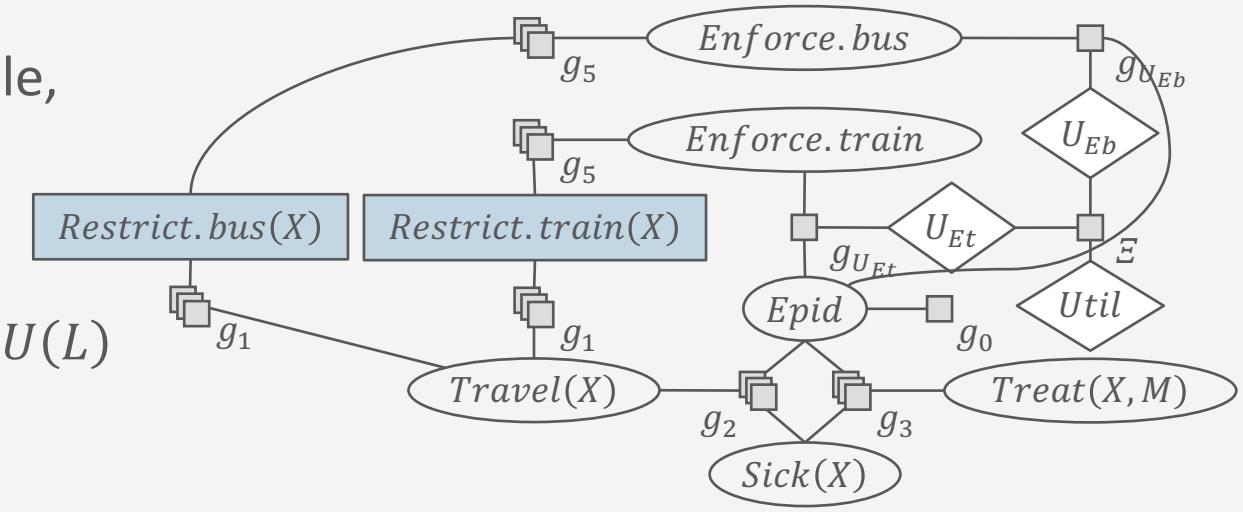


2. Set of Indistinguishable Attributes

- Indistinguishable attributes $R_1, \dots, R_M$ that show MPI → Utility function ”factorises” into a set of *indistinguishable* functions f_i over indistinguishable attributes
  - Utility function:

$$U(r_1, \dots, r_M) = \Xi[\phi_1(r_1), \dots, \phi_M(r_M)] = \Xi[\phi_u(r_1), \dots, \phi_u(r_M)]$$

    - All ϕ_i are ϕ_u, mapping to an interim utility variable U_i
    - If Ξ addition, then $U(r_1, \dots, r_M) = M \cdot \phi_u(r_u)$
  - *Precondition*: For the f_i to be indistinguishable, the R_i need to be indistinguishable
    - Encode indistinguishable attributes as PRV $R(L)$, $|\mathrm{dom}(L)| = M$
    - Then, encode interim utilities U_i as utility PRV $U(L)$
    - Logical variables of utility PRV always follow logical variables in PRVs of utility function

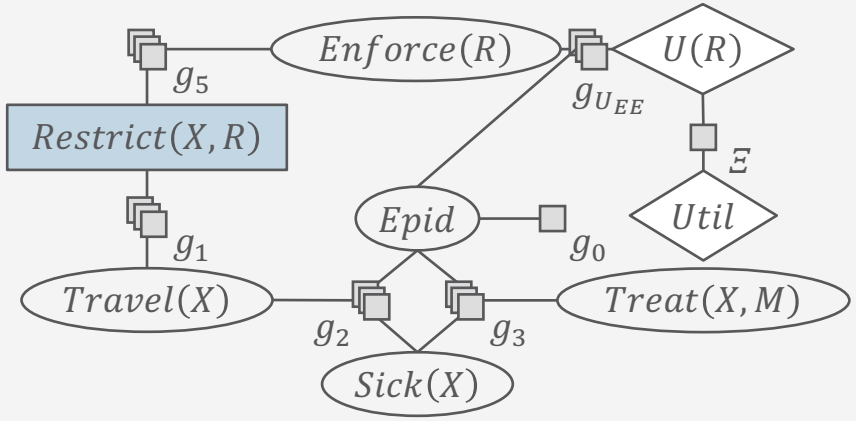


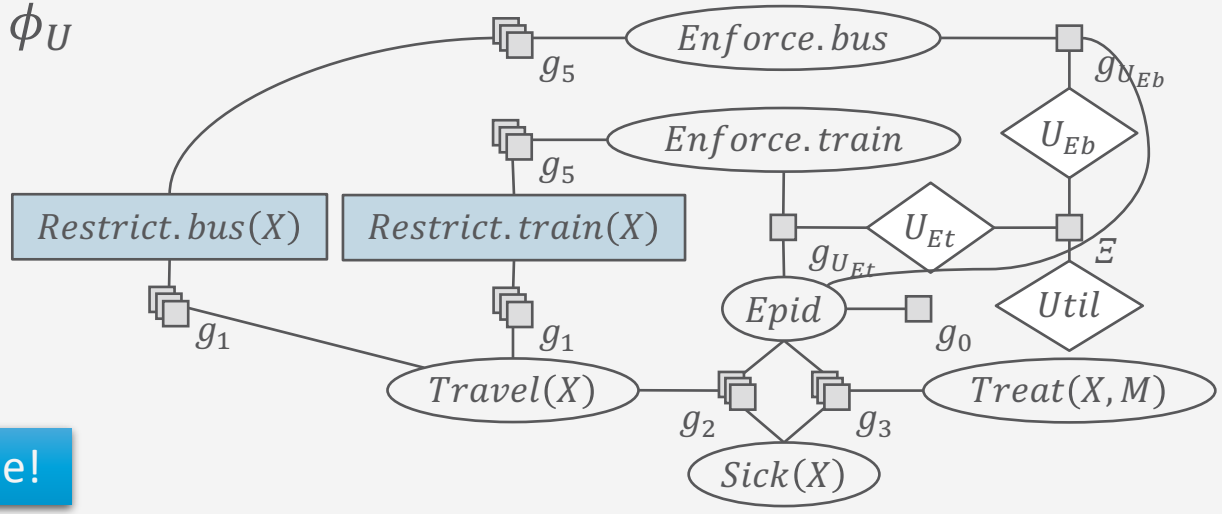
2. Set of Indistinguishable Attributes

- Extended syntax: Decision model

$$G = \{g_i\}_{i=1}^{n} \cup \{g_U\} \cup \{\Xi\}$$

  - $g_i = \phi_i(\mathcal{A}_i)_{|C_i}$ parfactors with (decision) PRVs as arguments
  - $g_U = \phi_{U(\mathcal{L})}(\mathcal{A})$ a utility *parfactor* and $U(\mathcal{L})$ a utility PRV
    - $\mathcal{L} = \mathrm{lv}(\mathcal{A})$ holds
    - $\mathrm{gr}(g_U) = \{f_1, \dots, f_m\}$, all f_i with utility function ϕ_U
  - Ξ a combination function, combining $U(\mathcal{L})$ into one U in lifted way (for liftability)
    - Addition yields a multiplication
      - Compare multiplication leading to an exponentiation in multiplicative semantics

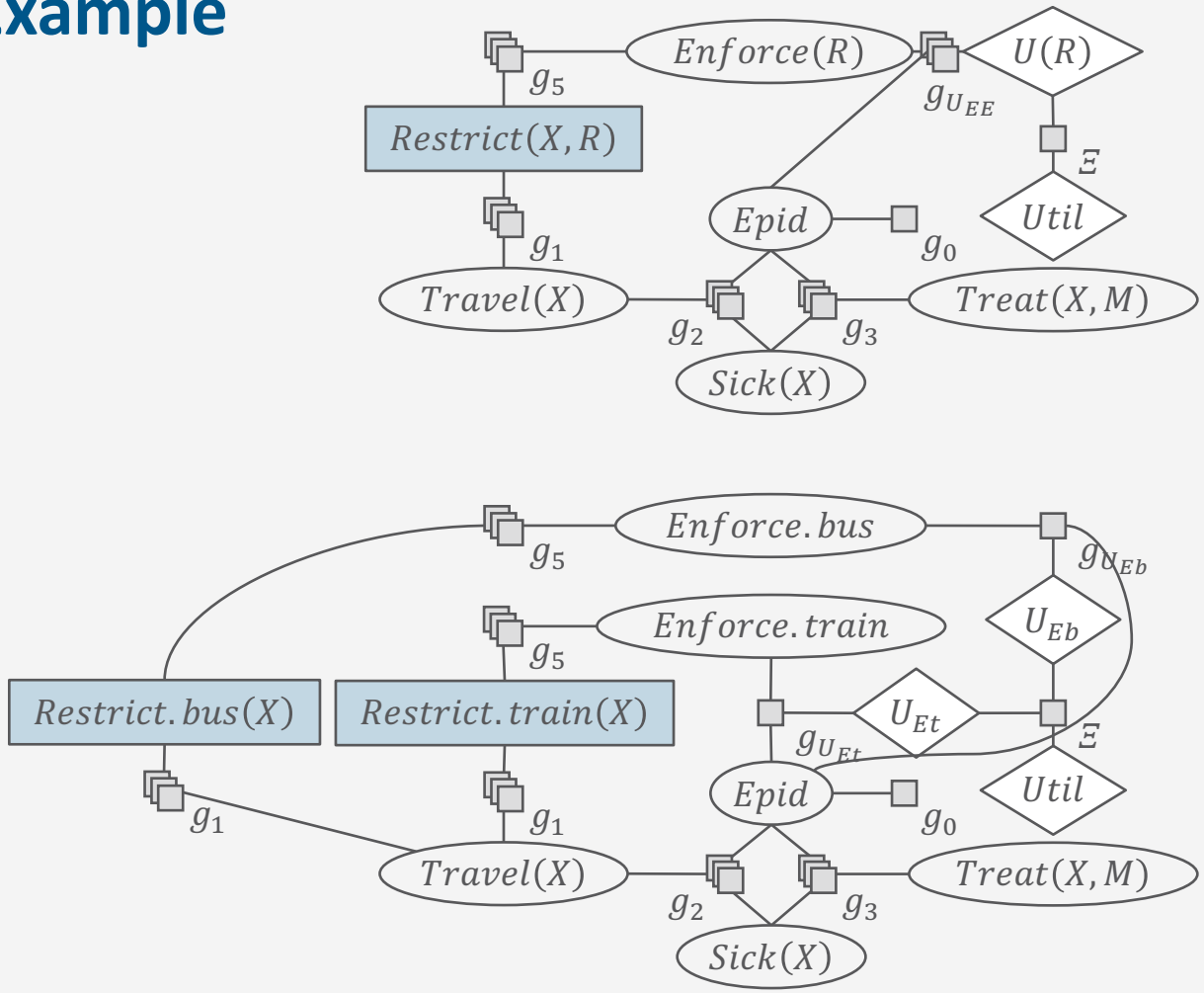




As of now, logical variables in utility model possible!

2. Set of Indistinguishable Attributes: Example

- Example:
  - Assume that effort for enforcing travel restrictions on busses and trains is identical
  - Ground:
    - Utility factor $\phi_{U_{Eb}}(Epid, Enforce.bus)$
    - Utility factor $\phi_{U_{Et}}(Epid, Enforce.train)$
  - Lifted:
    - Utility parfactor $\phi_{U(R)}\big(Epid, Enforce(R)\big)$
      - T constraint with $\mathrm{dom}(R) = \{train, bus\}$
  - Combination function: addition
    - Lifted: multiplication with $|\mathrm{dom}(R)|$



2. Set of Indistinguishable Attributes: EU Query & MEU Problem

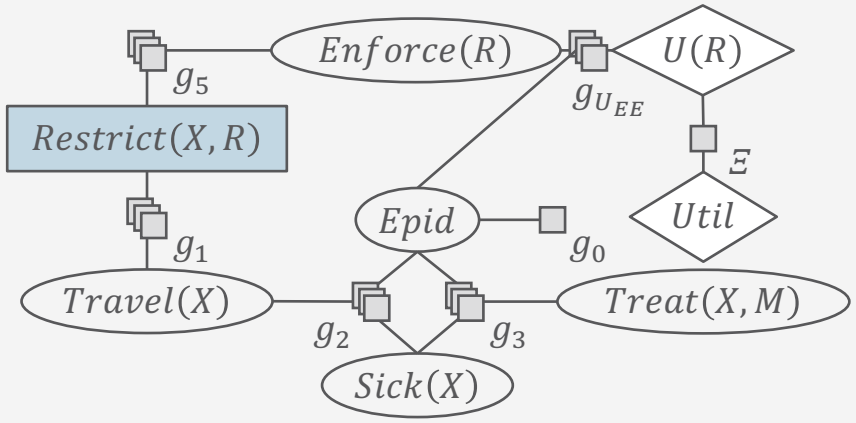
- Given a decision model $G = G_P \cup G_U \cup \{\Xi\} = \{g_i\}_{i=1}^{n} \cup \{g_U\} \cup \{\Xi\}$
  - Query for an expected utility (EU): sum over $\mathrm{gr}(\mathrm{rv}(g_U))$

$$eu(\boldsymbol{e}, \boldsymbol{d}) = \sum_{\boldsymbol{v} \in \mathrm{ran}(\mathrm{gr}(\mathrm{rv}(G_U) \setminus \boldsymbol{E} \setminus \boldsymbol{D}))} P(\boldsymbol{v} | \boldsymbol{e}, \boldsymbol{d}) \cdot \Xi \left[\phi_{U_1} \left(\pi_{\mathcal{R}_1}(\boldsymbol{v}, \boldsymbol{e}, \boldsymbol{d}) \right), \dots, \phi_{U_m} \left(\pi_{\mathcal{R}_m}(\boldsymbol{v}, \boldsymbol{e}, \boldsymbol{d}) \right) \right]$$

  - MEU problem: *no changes*

$$\mathrm{meu}(G | \boldsymbol{e}) = \left(\boldsymbol{d}^*, eu(\boldsymbol{e}, \boldsymbol{d}^*) \right)$$
$$\boldsymbol{d}^* = \underset{\boldsymbol{d} \in \mathrm{ran}(\boldsymbol{D})}{\mathrm{argmax}}\, eu(\boldsymbol{e}, \boldsymbol{d})$$

- But: Given semantics, EU query calculation not lifted!
  → Can we avoid grounding?

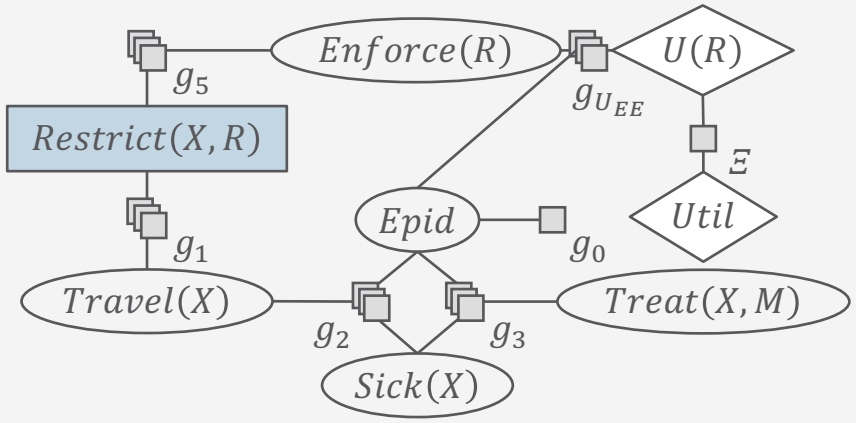


2. Set of Indistinguishable Attributes: Liftability

- Given a decision model $G = G_P \cup G_U \cup \{\Xi\} = \{g_i\}_{i=1}^{n} \cup \{g_U\} \cup \{\Xi\}$
  - Query for an expected utility (EU): sum over $\mathrm{gr}(\mathrm{rv}(g_U))$

$$eu(\boldsymbol{e}, \boldsymbol{d}) = \sum_{\boldsymbol{v} \in \mathrm{ran}(\mathrm{gr}(\mathrm{rv}(G_U) \setminus \boldsymbol{E} \setminus \boldsymbol{D}))} P(\boldsymbol{v}|\boldsymbol{e}, \boldsymbol{d}) \cdot \Xi \left[\phi_{U_1} \left(\pi_{\mathcal{R}_1}(\boldsymbol{v}, \boldsymbol{e}, \boldsymbol{d}) \right), \ldots, \phi_{U_m} \left(\pi_{\mathcal{R}_m}(\boldsymbol{v}, \boldsymbol{e}, \boldsymbol{d}) \right) \right]$$

- Changes in calculations for $eu(\boldsymbol{e}, \boldsymbol{d})$ with $\mathrm{rv}(G_U)$ now containing logical variables
  → $P(\boldsymbol{v}|\boldsymbol{e}, \boldsymbol{d})$ a parameterised query with $\boldsymbol{V} = \mathrm{rv}(G_U)$
  → If query liftable, then $\boldsymbol{V}$ as CRVs in answer → liftable
  - But: logical variables in g_U not counted
  → If Ξ addition: additive count-conversion for utility parfactors
  → Sum then over range of CRVs (include $Mul(h)$!)
  → Lifted calculations: Sum polynomial in domain sizes



2. Set of Indistinguishable Attributes: Additive Count-Conversion

- Operator:

Operator 2 Count-conversion for utility parfactors

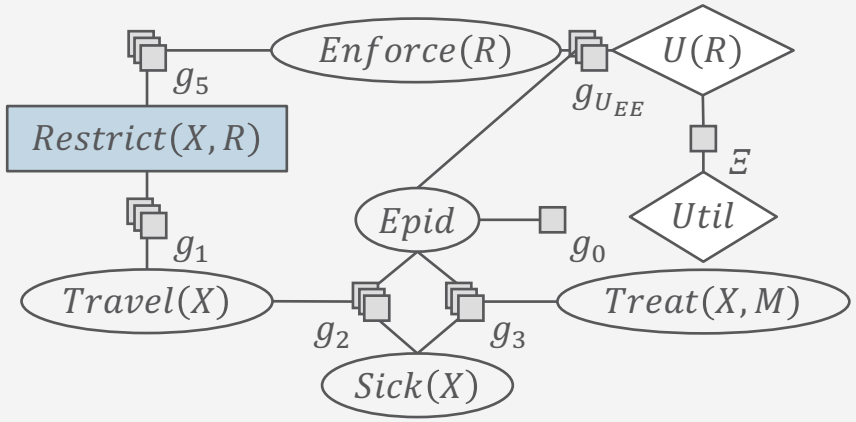
Operator COUNT-CONVERT
Inputs:
(1) Utility parfactor $g_u = \phi_{U(\boldsymbol{X})}(\mathcal{A})_{|C}$
(2) logical variable $X \in lv(\mathcal{A})$ and $X \in \boldsymbol{X}$, to count in g_u
Preconditions:
(1) There is exactly one atom $A_i \in \mathcal{A}$ with $X \in lv(A_i)$.
(2) X is count-normalised w.r.t. $\boldsymbol{Z} = lv(\mathcal{A}) \setminus \{X\}$ in C.
(3) For all counted logical variables $X^{\#}$ in g: $\pi_{X,X^{\#}}(C) = \pi_X(\pi_{\boldsymbol{X}}(C)) \times \pi_{X^{\#}}(\pi_{\boldsymbol{X}}(C))$.
Output: utility parfactor $\phi'_{U'}(\mathcal{A}')_{|C}$ such that
(1) $U' = \#_X[U(\boldsymbol{X})]$,
(2) $\mathcal{A}' = (A_1, \ldots, A_{i-1}) \circ A'_i \circ (A_{i+1}, \ldots, A_n)$, $A'_i = \#_X[A_i]$, and
(3) for each valuation $\boldsymbol{a}'$ to $\mathcal{A}'$ with $a'_i = h$,

$$\phi'_{U(\boldsymbol{X})}(\ldots, a_{i-1}, h, a_{i+1}, \ldots) = \sum_{a \in ran(A_i)} h(a_i)\phi_{U(\boldsymbol{X})}(\ldots, a_{i-1}, a_i, a_{i+1}, \ldots)$$

where h is a histogram $\{(a_i, n_i)\}_{i=1}^{m}$ with $m = |ran(A_i)|$, $a_i \in ran(A_i)$, $n_i \in \mathbb{N}$, and $\sum_{a_i \in \mathcal{R}(A_i)} h(a_i) = ncount_{X|\boldsymbol{Z}}(C)$, and $h(a_i) = n_i$.
Postcondition: $G_U \equiv G_U \setminus \{g_u\} \cup$ COUNT-CONVERT(g_u, X)

Compare multiplicate count-conversion:

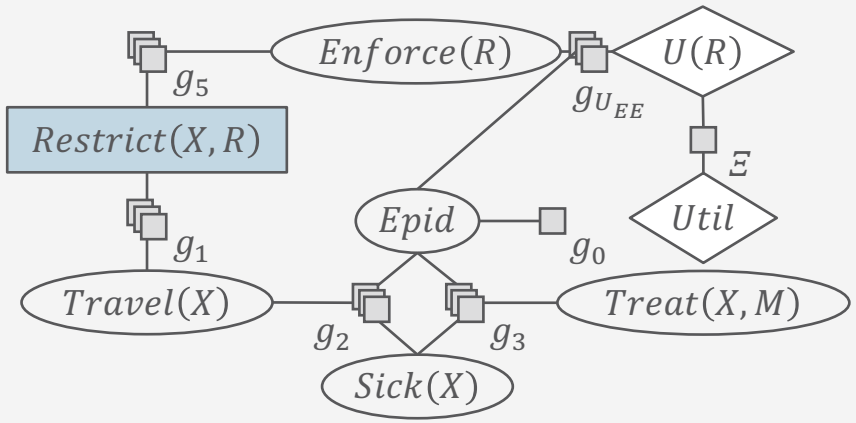
$$\phi'(\ldots, a_{i-1}, h, a_{i+1}, \ldots) = \prod_{a_i \in ran(A_i)} \phi(\ldots, a_{i-1}, a_i, a_{i+1}, \ldots)^{h(a_i)}$$



Lifted Decisions

E	$E(R)$	ϕ
$false$	$false$	10
$false$	$true$	4
$true$	$false$	8
$true$	$true$	5

E	$E(R)$	$\phi_{U(R)}$
$false$	$false$	5
$false$	$true$	0
$true$	$false$	-5
$true$	$true$	-10



E	$\#_R[E(R)]$	$\phi^{\#}$	ϕ^n
$false$	[0,2]	$4^0 \cdot 10^2 = 100$	0.263
$false$	[1,1]	$4^1 \cdot 10^1 = 40$	0.105
$false$	[2,0]	$4^2 \cdot 10^0 = 16$	0.042
$true$	[0,2]	$5^0 \cdot 8^2 = 64$	0.168
$true$	[1,1]	$5^1 \cdot 8^1 = 40$	0.105
$true$	[2,0]	$5^2 \cdot 8^0 = 40$	0.105

E	$\#_R[E(R)]$	$\phi'_{\#_R[U(R)]}$
$false$	[0,2]	$0 \cdot 0 + 2 \cdot 5 = 10$
$false$	[1,1]	$1 \cdot 0 + 1 \cdot 5 = 5$
$false$	[2,0]	$2 \cdot 0 + 0 \cdot 5 = 0$
$true$	[0,2]	$0 \cdot -10 + 2 \cdot -5 = -10$
$true$	[1,1]	$1 \cdot -10 + 1 \cdot -5 = -15$
$true$	[2,0]	$2 \cdot -10 + 0 \cdot -5 = -20$

$\phi^{\#} \cdot \phi'_{\#_R[U(R)]}$
$0.263 \cdot 10 = 2.630$
$0.105 \cdot 5 = 0.525$
$0.042 \cdot 0 = 0.000$
$0.168 \cdot -10 = -1.680$
$0.105 \cdot -15 = -1.575$
$0.105 \cdot -20 = -2.100$

$$eu = 1 \cdot 2.630 + 2 \cdot 0.525 + 1 \cdot 0 + 1 \cdot -1.68 + 2 \cdot -1.575 + 1 \cdot -2.1 = -3.25$$

Sum of $\phi^{\#}$ potentials $= 1 \cdot 100 + 2 \cdot 40 + 1 \cdot 16 + 1 \cdot 64 + 2 \cdot 40 + 1 \cdot 40 = 380$

2. Set of Indistinguishable Attributes: Simplification

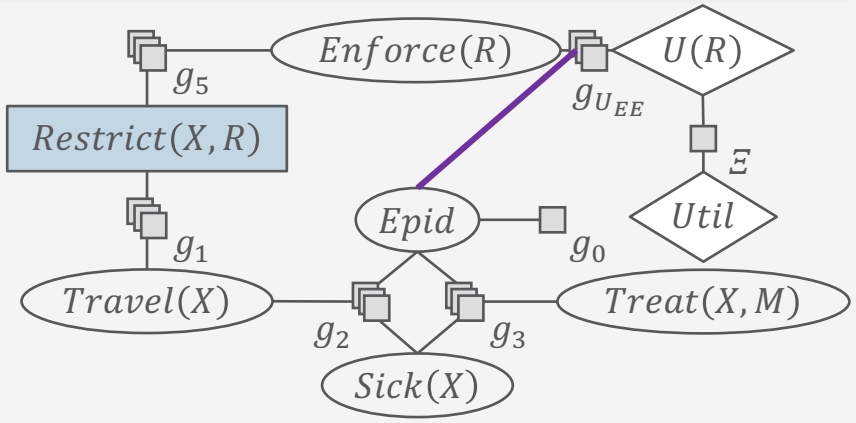
- Assume all groundings are independent
  - $\forall \mathcal{R}(\boldsymbol{x}), \mathcal{R}(\boldsymbol{y}) \in \mathrm{gr}\big(\mathrm{rv}(g_U)\big) : (\mathcal{R}(\boldsymbol{x}) \perp \mathcal{R}(\boldsymbol{y}) | \boldsymbol{e}, \boldsymbol{d})$
- Then,

$$eu(\boldsymbol{e}, \boldsymbol{d}) = \sum_{u=1}^{m} \sum_{\boldsymbol{a}_u \in \mathrm{ran}(\mathrm{rv}(g_U))} P(\boldsymbol{a}_u | \boldsymbol{e}, \boldsymbol{d}) \cdot \phi_u(\boldsymbol{a}_u)$$
$$= m \cdot \sum_{\boldsymbol{a}_u \in \mathrm{ran}(\mathrm{rv}(g_U))} P(\boldsymbol{a}_u | \boldsymbol{e}, \boldsymbol{d}) \cdot \phi_u(\boldsymbol{a}_u)$$

  - $P(\boldsymbol{a}_u | \boldsymbol{e}, \boldsymbol{d})$ a representative query, i.e., a query over $\boldsymbol{A}_u = \mathrm{rv}(g_U)$ with a representative grounding $\boldsymbol{x}$ of its logical variables $\boldsymbol{X} = \mathrm{lv}(\boldsymbol{A}_u)$
  - $m = |\mathrm{gr}(g_U)|$

Lifted calculation:
- Sum *independent* of domain sizes m
- Multiplication with domain size in $O(\log m)$

Here, groundings are not independent because of $Epid$; without $Epid$, the groundings would be independent (of each other and anything else in the model)



We will see an example later.

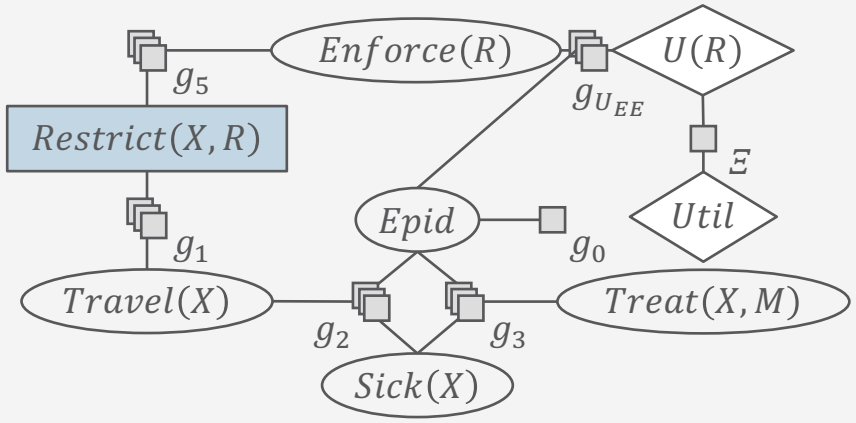
2. Set of Indistinguishable Attributes: MEU-LVE

- Implement ADD-COUNT-CONVERT operator
  - LVE with ADD operator and ADD-COUNT-CONVERT operator referred to as LVE$^{\text{addCC}}$
- Changes in MEU-LVE
  - Input: decision model $G = G_P \cup G_U = \{g_i\}_{i=1}^{n} \cup \{g_U\}$
  - In for-loop:

$g \leftarrow \text{LVE}(G \setminus G_U, \text{rv}(G_U), \boldsymbol{d})$ ▸ g normalised
$eu \leftarrow \text{LVE}^{\text{addCC}}(G_U \cup \{g\}, \emptyset, \boldsymbol{d})$

    - If Ξ not addition, need to implement (change $\text{LVE}^{\text{addCC}}$ call)
    - Count-converts the PRVs in g_U before multiplying with g and summing out the remaining variables
      - If PRVs in g_U not count-convertible → Ground logical variable and join partially grounded utility parfactors using ADD operator

If parameterised query *liftable*, then:
Complexity polynomial in M



2. Set of Indistinguishable Attributes: Logical Variables in Utility PRVs

- Definition says $\mathcal{L} = \text{lv}(\mathcal{A})$ holds for a utility parfactor $\phi_{U(\mathcal{L})}(\mathcal{A})$ a utility *parfactor* and $U(\mathcal{L})$ a utility PRV
- What about $\mathcal{L} \subset \text{lv}(\mathcal{A})$?

  ✗ Given grounding semantics, *not valid* as combination not defined
  - Example: $\phi_{Util}(Restrict(X), Epid)$
    - Groundings: $\phi_{Util}(Restrict(alice), Epid), \phi_{Util}(Restrict(eve), Epid), \phi_{Util}(Restrict(bob), Epid)$
- What about $\mathcal{L} \supset \text{lv}(\mathcal{A})$?

  ✓ Given grounding semantics, *valid* as only more utility factors occur
  - Example: $\phi_{U(R,C)}(Enforce(R), Epid), |\text{dom}(C)| = 3$
    - Groundings: $\phi_{U(b,c_1)}(Enforce(b), Epid), \phi_{U(b,c_2)}(Enforce(b), Epid), \phi_{U(b,c_3)}(Enforce(b), Epid),$
      $\phi_{U(t,c_1)}(Enforce(t), Epid), \phi_{U(t,c_2)}(Enforce(t), Epid), \phi_{U(t,c_3)}(Enforce(t), Epid),$

2. Set of Indistinguishable Attributes: Eliminating Logical Variables

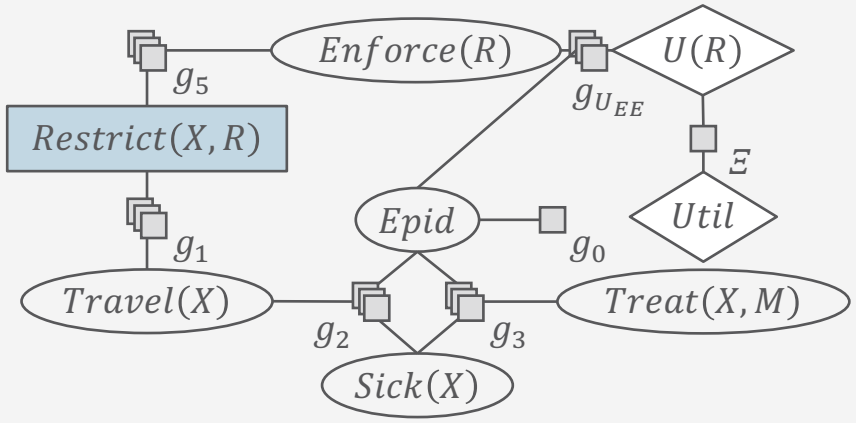
- Grounding a utility parfactor with additional logical variables in its utility PRV leads to copies of utility factors over the same inputs that can be combined based on Ξ
  - Eliminate beforehand as a first step to simplify a model
- Operator for eliminating additional logical variables in a utility PRV of a utility parfactor

Operator 4 Logical variable elimination in utility PRVs

Operator ELIM-LOG-VARS
Inputs:
(1) Utility parfactor $g_u = \phi_{U(\boldsymbol{X})}(\mathcal{A})_{|C}$
(2) Logical variables $\boldsymbol{Y} \subseteq \boldsymbol{X}$
Preconditions:
(1) $\boldsymbol{Y}$ do not occur in $\mathcal{A}$, i.e., $\boldsymbol{Y} \cap lv(\mathcal{A}) = \emptyset$.
(2) $\boldsymbol{Y}$ are count-normalised w.r.t. $lv(\mathcal{A})$ in C.
Output: utility parfactor $\phi'_{U(\boldsymbol{Z})}(\mathcal{A})_{|C'}$ such that
(1) $\boldsymbol{Z} = \boldsymbol{X} \setminus \boldsymbol{Y}$,
(2) $C' = C \setminus \boldsymbol{Y}$ (remove $\boldsymbol{Y}$ and its constants from C), and
(3) for each valuation $\boldsymbol{a}$ to $\mathcal{A}$,

$$\phi'_{U(\boldsymbol{Z})}(\boldsymbol{a}) = ncount_{\boldsymbol{Y}|\boldsymbol{Z}}(C) \cdot \phi_{U(\boldsymbol{X})}(\boldsymbol{a})$$

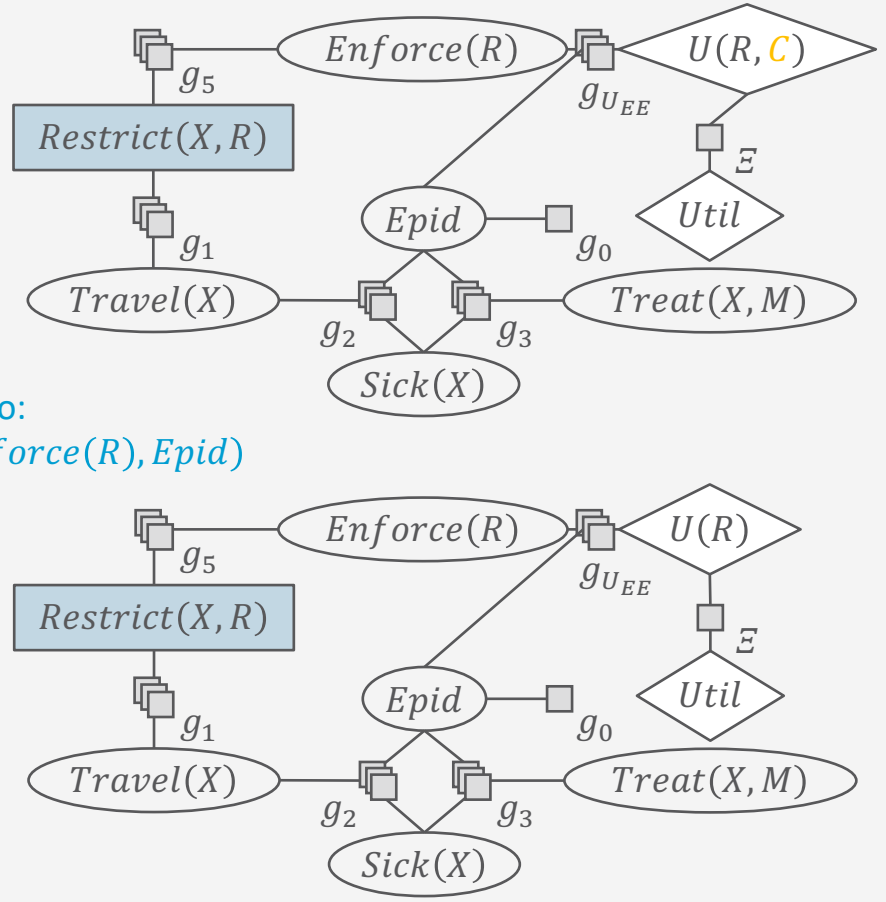
Postcondition: $G_U \equiv G_U \setminus \{g_u\} \cup \text{ELIM-LOG-VARS}(g_u, \boldsymbol{Y})$



2. Set of Indistinguishable Attributes: Eliminating Logical Variables

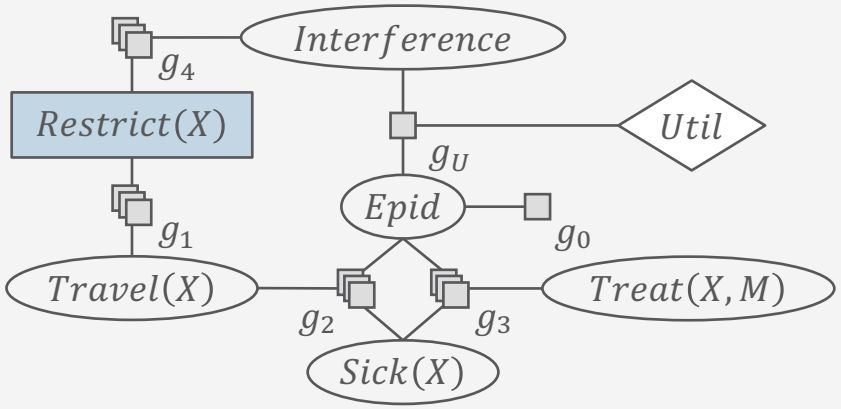
- Example: $\phi_{U(R,C)}(Enforce(R), Epid)$, $|\mathrm{dom}(C)| = 3$
  - New utility parfactor: $\phi_{U(R)}(Enforce(R), Epid)$
    - For all $en \in \mathrm{ran}(Enforce(R)), ep \in \mathrm{ran}(Epid)$:
      $$\phi_{U(R)}(en, ep) = 3 \cdot \phi_{U(R,C)}(en, ep)$$
  - Ground comparison:
    - For all $en \in \mathrm{ran}(Enforce(b)), ep \in \mathrm{ran}(Epid)$:
      $\phi_{U(b,c_1)}(en, ep) + \phi_{U(b,c_2)}(en, ep) + \phi_{U(b,c_3)}(en, ep) = 3 \cdot \phi_{U(R,C)}(en, ep)$
    - For all $en \in \mathrm{ran}(Enforce(t)), ep \in \mathrm{ran}(Epid)$:
      $\phi_{U(t,c_1)}(en, ep) + \phi_{U(t,c_2)}(en, ep) + \phi_{U(t,c_3)}(en, ep) = 3 \cdot \phi_{U(R,C)}(en, ep)$

Combine into: $\phi_{U(R)}(Enforce(R), Epid)$



Structure in Multi-attribute Settings

- So far: Set of attributes without structure
  - Single utility functions mapping to one utility
    - Example: $\phi_U(Interference, Epid)$
- Cases with structure:
  1. **Set of (distinguishable) attributes with structure**
     - Set of utility functions, mapping to interim utilities, combined into one overall utility
  2. **Set of indistinguishable attributes**
     - Utility parfactor mapping to an interim utility PRV, which is combined into one utility
  3. **Sets of distinguishable and indistinguishable attributes**
     - Set of utility parfactors and utility factors, combined into one utility
  - Considering structure requires a combination function Ξ



3. Sets of Distinguishable & Indistinguishable Attributes

- Full expressiveness in terms of syntax: Allows for a set of utility parfactors as utility model
- Full decision model:
  - Syntax

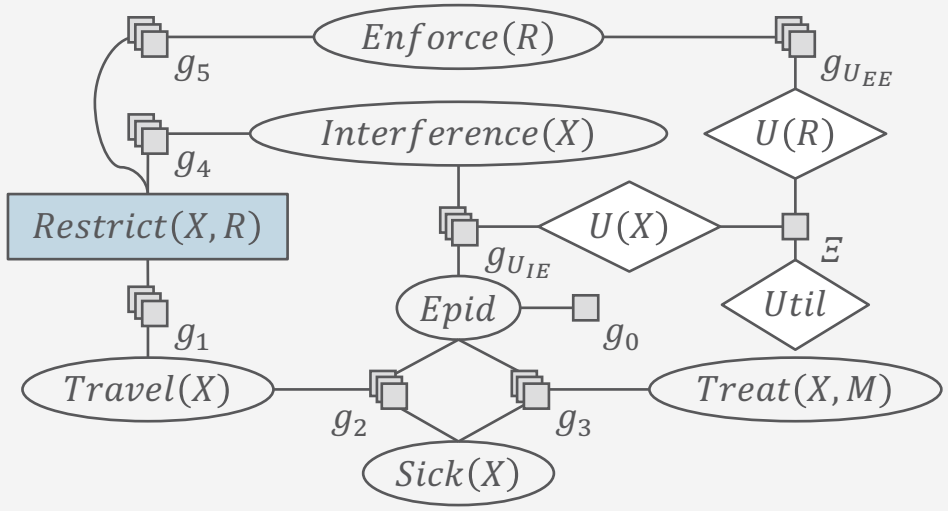
$$G = \{g_i\}_{i=1}^{n} \cup \{g_u\}_{u=1}^{m} \cup \{\Xi\}$$

    - $g_i = \phi_i(\mathcal{A}_i)_{|C_i}$ parfactor with (decision) PRVs as arguments
    - $g_u = \phi_{U_u(\mathcal{L}_u)}(\mathcal{A}_u)_{|C_u}$ utility parfactor, mapping to a utility PRV $U_u(\mathcal{L}_u)$ with $\mathcal{L}_u = \mathrm{lv}(\mathcal{A}_u)$
    - Ξ a combination function, combining all $U_u(\mathcal{L}_u)$ into one U
  - Semantics: grounding semantics
    - Given an action assignment $\boldsymbol{d}$, full joint $P_{G_P}[\boldsymbol{d}]$ over grounding, multiplying, and normalising
    - EU queries sum over $\mathrm{gr}\big(\mathrm{rv}(G_U)\big)$ → Liftable parameterised query or simplification for liftability
    - MEU problem: With $\boldsymbol{D}$ the decision PRVs in G

$$\mathrm{meu}(G|\boldsymbol{e}) = \big(\boldsymbol{d}^*, eu(\boldsymbol{e}, \boldsymbol{d}^*)\big) \qquad \boldsymbol{d}^* = \underset{\boldsymbol{d} \in \mathrm{ran}(\boldsymbol{D})}{\mathrm{argmax}}\, eu(\boldsymbol{e}, \boldsymbol{d})$$

3. Sets of Distinguishable & Indistinguishable Attributes: Example

- Decision model $G = G_P \cup G_U \cup \{\Xi\}$
  - $G_P = \{g_i\}_{i=0}^{5}$
    - $g_0 = \phi_0(Epid)$
    - $g_1 = \phi_1\big(Restrict(X,R), Travel(X)\big)$
    - $g_2 = \phi_2\big(Epid, Sick(X), Travel(X)\big)$
    - $g_3 = \phi_3\big(Epid, Sick(X), Treat(X,M)\big)$
    - $g_4 = \phi_4\big(Restrict(X,R), Enforce(R)\big)$
    - $g_5 = \phi_5\big(Restrict(X,R), Interf.(X)\big)$
  - $G_U = \{g_u\}_{u=0}^{5}$
    - $g_{U_{EE}} = \phi_{U(R)}\big(Enforce(R)\big)$
    - $g_{U_{IE}} = \phi_{U(X)}(Interference(X), Epid)$
  - Ξ addition with additive operators for LVE
- In EU query
  - Independences given $Restrict(X,R)$
    - Between utility parfactor PRVs ✓
    - Between groundings of $Enforce(R)$ ✓



MEU Problems: Alternative Solution Approach

- Solving an MEU problem in decision model G with $\Xi(\boldsymbol{V})$ as short form for utility model:

$$\mathrm{meu}(G|\boldsymbol{e}) = \left(\boldsymbol{d}^*, eu(\boldsymbol{e}, \boldsymbol{d}^*)\right), \boldsymbol{d}^* = \underset{\boldsymbol{d}\in\mathrm{ran}(\boldsymbol{D})}{\mathrm{argmax}}\, eu(\boldsymbol{e}, \boldsymbol{d}) = \underset{\boldsymbol{d}\in\mathrm{ran}(\boldsymbol{D})}{\mathrm{argmax}} \sum_{\boldsymbol{v}\in\mathrm{ran}\left(\mathrm{gr}(\mathrm{rv}(G_U)\setminus\boldsymbol{E}\setminus\boldsymbol{D})\right)} P(\boldsymbol{v}|\boldsymbol{e}, \boldsymbol{d}) \cdot \Xi(\boldsymbol{v})$$

- So far: for each $\boldsymbol{d}$, set $\boldsymbol{d}$, eliminate all PRVs not in G_U, eliminate remaining PRVs
  - Advantage: Reduced model by setting $\boldsymbol{d}$ (possible independences)
  - Disadvantage: possibly large $P(\boldsymbol{v}|\boldsymbol{e}, \boldsymbol{d})$ has to be computed
- Alternative: Compute a maximum-a-posteriori (MAP) assignment for the decision PRVs
  - Eliminate all non-decision PRVs in G_P by summing out, eliminate the decision PRVs by *maxing* out (replace sum operation by max-out operation)
    - Max-out: for each remaining world, pick the assignment with maximum value and store
  - Advantage: Does not require computing $P(\boldsymbol{v}|\boldsymbol{e}, \boldsymbol{d})$, easier to exploit factorisation
  - Disadvantage: Only a ranking (no true expected utility), no further independences through $\boldsymbol{d}$

Some References

- MEU in parfactor-based decision models
  - Warning: not as detailed as in these slides

Version using an early version of LVE, mashing early parfactor graphs and MLNs:
Udi Apsel and Ronan I. Brafman. Extended Lifted Inference with Joint Formulas. In: *UAI-11 Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, 2011.

MEU-LVE: Marcel Gehrke, Tanya Braun, Ralf Möller, Alexander Waschkau, Christoph Strumann, and Jost Steinhäuser. Towards Lifted Maximum Expected Utility. In: *Proceedings of the First Joint Workshop on Artificial Intelligence in Health in Conjunction with the 27th IJCAI, the 23rd ECAI, the 17th AAMAS, and the 35th ICML*, 2018.
Marcel Gehrke, Tanya Braun, Ralf Möller, Alexander Waschkau, Christoph Strumann, and Jost Steinhäuser. Lifted Maximum Expected Utility. In: *Artificial Intelligence in Health*, 2019.

- Markov logic decision networks (MLDNs)
  - MLN + parameterised decisions + utility weights
    - Probability + utility weights per first-order formula
  - Use weighted model counting to solve MEU problem

MLDNs: Aniruddh Nath and Pedro Domingos. A Language for Relational Decision Theory. In: *Proceedings of the International Workshop on Statistical Relational Learning*, 2009.
MLDNs + WMC: Udi Apsel and Ronan I. Brafman. Lifted MEU by Weighted Model Counting. In: *AAAI-12 Proceedings of the 26th AAAI Conference on Artificial Intelligence*, 2012.

- Decision-theoretic Probabilistic Prolog (DTProbLog)
  - Utilities of DTProbLog programs combined into EU over theory defined by programs

DTProbLog: Guy Van den Broeck, Ingo Thon, Martijn van Otterlo, and Luc De Raedt. DTProbLog: A Decision-Theoretic Probabilistic Prolog. In: *AAAI-10 Proceedings of the 24th AAAI Conference on Artificial Intelligence*, 2010.

Interim Summary

- Decision models
  - Probabilistic graphical model extended with decision and utility variables
- Parfactor-based version
  - Decision PRVs, utility PRVs, utility parfactors, combination function
  - Collective decisions for groups of indistinguishable constants
- EU queries, MEU problem
  - Find set of actions (decisions) that lead to maximum expect utility
  - MEU-LVE using calls to LVE and LVE operators to answer EU queries
    - Combination function addition → additive join + count-conversion

Outline: 7. Lifted Decision Making

A. Utility theory
- Preferences, maximum expected utility (MEU) principle
- Utility function, multi-attribute utility theory

B. Static decision making
- Modelling, semantics, inference tasks
- Inference algorithm: LVE for MEU as an example

C. Sequential decision making
- Modelling, semantics, sequential MEU problem
- Inference algorithm: LDJT for MEU as an example
- Acting

Decision Making over Time

So far:

- Calculate the expected utility of a decision and its effect on the (current) state

With time

- Decisions need to be made at each time step
- Each time step yields a utility
- Decisions/actions have an effect not only on the current state/utility but also on the future and therefore, future decisions

→ Need to consider a temporal sequence of decisions and project its effect into the future

→ Requires calculating the expected utility over a sequence

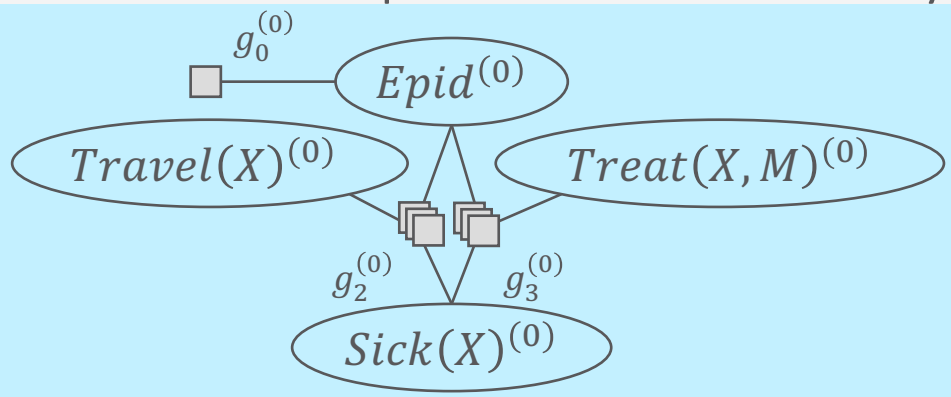
Sequential Parfactor-based Model

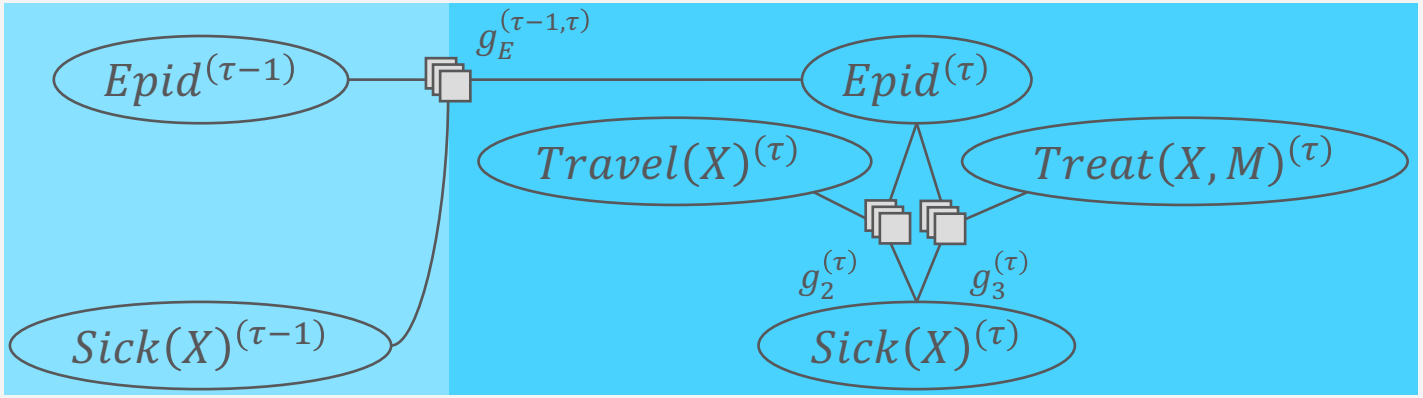
- Sequential parfactor-based model $(G^0, G^{\rightarrow})$ with
  - $G^0 = \left\{g_i^{(0)}\right\}_{i=1}^{n_0}, g_i^{(0)} = \phi_i^{(0)}\left(A_1^{(0)}, \ldots, A_{k_i}^{(0)}\right)$
  - $G^{\rightarrow}$ a 2-slice model, i.e., $G^{\rightarrow} = \left\{g_i^{(\tau)}\right\}_{i=1}^{n} \cup \left\{g_j^{(\tau-1,\tau)}\right\}_{j=1}^{m}$
  - Example: $G^0 = \left\{g_2^{(0)}, g_3^{(0)}\right\} \cup \left\{g_0^{(0)}\right\}, G^{\rightarrow} = \left\{g_2^{(\tau)}, g_3^{(\tau)}\right\} \cup \left\{g_E^{(\tau-1,\tau)}\right\}$
    - No assumptions about observability

Semantics: Unroll model M for T steps, ground, build full joint.

$$M^{(0:T)} = M^0 \cup \bigcup_{t=1}^{T} M^{\rightarrow|\tau=t}$$

$$P_M^T = \frac{1}{Z} \prod_{\zeta \in gr\left(M^{(0:T)}\right)} \zeta$$





Query Answering Problem & Query Types

Inference Tasks

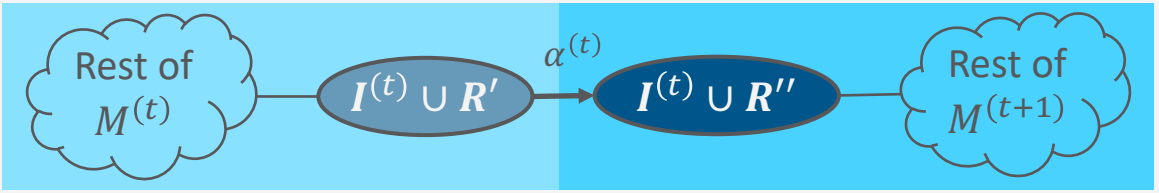
- Formally, query on a dynamic model M over template random variables $\boldsymbol{R}$:

$$P\left(\boldsymbol{S}^{(\pi)} \middle| \boldsymbol{e}^{(0:t)}\right)$$

  - where t the current step, $\boldsymbol{S} \subseteq \boldsymbol{R}$ query terms, $\boldsymbol{E} \subseteq \boldsymbol{R}$ evidence terms, and usually $\boldsymbol{S} \cap \boldsymbol{E} = \emptyset$
- Filtering: $\pi = t$
- Prediction: $\pi > t$
- Hindsight: $\pi < t$

Inference Algorithm: LDJT

- Two FO jtrees $(J^0, J^{\rightarrow})$ with interfaces for separation between past and present
- LJT as a subroutine for query answering
- Forward/backward messages to move in time

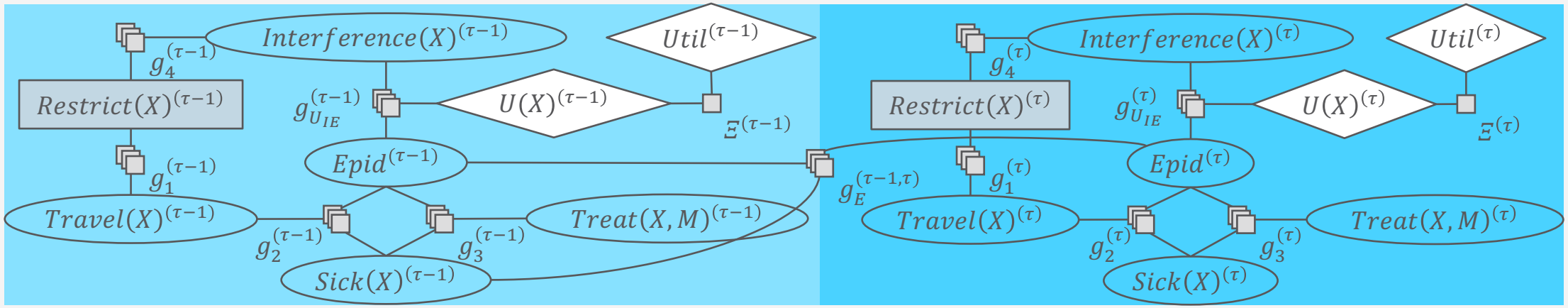


Decision Making over Time

- Basis: a sequential model $(G^0, G^{\rightarrow})$
  - Describe behaviour over time using interslice parfactors
  - Within a slice, describe intra-slice (episodic) behaviour

→ Extend intra-slice parts with decision + utility PRVs

  - Intra-slice behaviour described using a decision model
  - Inter-slice behaviour allows for predicting effect of decision on next step



A First Version of a Sequential Decision Model

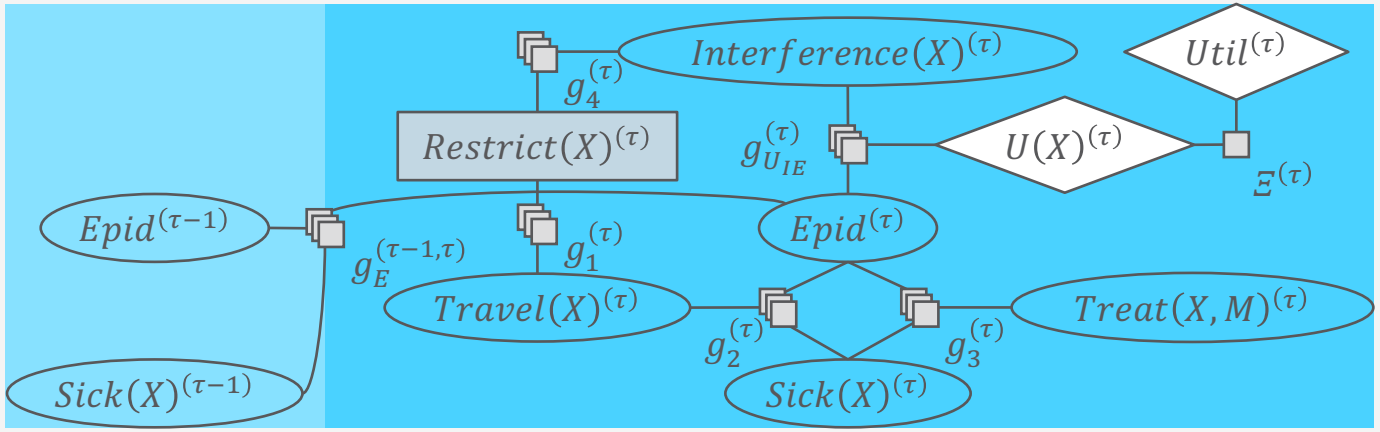
- $G = (G^0, G^{\rightarrow})$ is a sequential model where $G^0, G^{\rightarrow}$ are decision models, i.e.,
  - G^0 is a decision model describing the intra-time slice behaviour for $t = 0$

$$G^0 = \left\{g_i^{(0)}\right\}_{i=1}^{n_0} \cup \left\{g_u^{(0)}\right\}_{u=1}^{m_0} \cup \left\{\Xi^{(0)}\right\}$$

  - $G^{\rightarrow}$ is a decision model describing the intra- and inter-slice behaviour for $t > 0$

$$G^{\rightarrow} = \left\{g_i^{(\tau)}\right\}_{i=1}^{n} \cup \left\{g_u^{(\tau)}\right\}_{u=1}^{m} \cup \left\{g_j^{(\tau-1,\tau)}\right\}_{j=1}^{k} \cup \left\{\Xi^{(\tau)}\right\}$$

  - with
    - $g_i^{(t)}, g_j^{(\tau-1,\tau)}$ (potential) parfactors
    - $g_u^{(t)}$ utility parfactors
    - $\Xi^{(t)}$ a combination function
    - $t \in \{0, \tau\}$



Assumptions about Actions

- Assumptions in terms of time / sequential behaviour…
  - Discrete steps
  - Markov-1
  - Stationary process
- … lead to assumptions/ constraints on decisions
  - Actions can be carried out from one step to the next (no duration)
  - Effect/outcome of actions immediately captured in the next step
  - Actions do not affect the stationary process

Actions over Time

- Given a set of decision PRVs
  - $\boldsymbol{D}^{(\tau)} = \left\{D_1^{(\tau)}, \dots, D_n^{(\tau)}\right\}$
- Sequential sequence of decisions = sequential action assignment
  - Compound event for a sequence of steps from t_1 to t_2, written as $\boldsymbol{d}^{(t_1:t_2)}$, i.e.,
$$\boldsymbol{d}^{(t_1:t_2)} = \left(\boldsymbol{d}^{(t_1)}, \boldsymbol{d}^{(t_1+1)}, \dots, \boldsymbol{d}^{(t_2-1)}, \boldsymbol{d}^{(t_2)}\right)$$
    - $\boldsymbol{d}^{(t_i)} = \left\{D_1^{(t_i)} = d_1^{(t_i)}, \dots, D_n^{(t_i)} = d_n^{(t_i)}\right\}, t_i \in \{t_1, \dots, t_2\}$
- Decision making over time usually involves calculating the best sequence starting at current step t, making decisions for κ steps, given evidence up until t and previous decisions
  - I.e., finding the best $\boldsymbol{d}^{(t:t+\kappa)}$ given $\boldsymbol{e}^{(1:t)}$ and $\boldsymbol{d}^{(1:t-1)}$

So far, we used utilities to determine “the best”. How can we use them with time?

Utilities over Time

- One has to iterate over sequential action assignments and pick the one that yields the maximum expected utility
  - Each individual slice has a utility (or reward) → Multi-attribute utility theory
  - All individual utilities need to be combined into one utility of the complete sequence using a sequential combination function Υ

$$U\left(\boldsymbol{r}^{(0)}, \boldsymbol{r}^{(1)}, \boldsymbol{r}^{(2)}, \dots\right) = \Upsilon\left[U\left(\boldsymbol{r}^{(0)}\right), U\left(\boldsymbol{r}^{(1)}\right), U\left(\boldsymbol{r}^{(2)}\right), \dots\right]$$

  - Assumption: Preference of one sequence over the other does not depend on time
    → Preferences are stationary
    - Formally, if two state sequences $\left(\boldsymbol{r}^{(0)}, \boldsymbol{r}^{(1)}, \boldsymbol{r}^{(2)}, \dots\right)$ and $\left(\boldsymbol{s}^{(0)}, \boldsymbol{s}^{(1)}, \boldsymbol{s}^{(2)}, \dots\right)$ have the same starting state ($\boldsymbol{r}^{(0)} = \boldsymbol{s}^{(0)}$), then the two sequences $\left(\boldsymbol{r}^{(1)}, \boldsymbol{r}^{(2)}, \dots\right)$ and $\left(\boldsymbol{s}^{(1)}, \boldsymbol{s}^{(2)}, \dots\right)$ should be preference-ordered in the same way as $\left(\boldsymbol{r}^{(0)}, \boldsymbol{r}^{(1)}, \boldsymbol{r}^{(2)}, \dots\right)$ and $\left(\boldsymbol{s}^{(0)}, \boldsymbol{s}^{(1)}, \boldsymbol{s}^{(2)}, \dots\right)$
      → Preference independence between the different slices
      → Use additive combination function (Υ)

Sequential Combination Functions

- General sequential combination function Υ

$$U(\boldsymbol{r}^{(0)}, \boldsymbol{r}^{(1)}, \boldsymbol{r}^{(2)}, \dots) = \Upsilon[U(\boldsymbol{r}^{(0)}), U(\boldsymbol{r}^{(1)}), U(\boldsymbol{r}^{(2)}), \dots]$$

- Additive
  - Sum over individual utilities

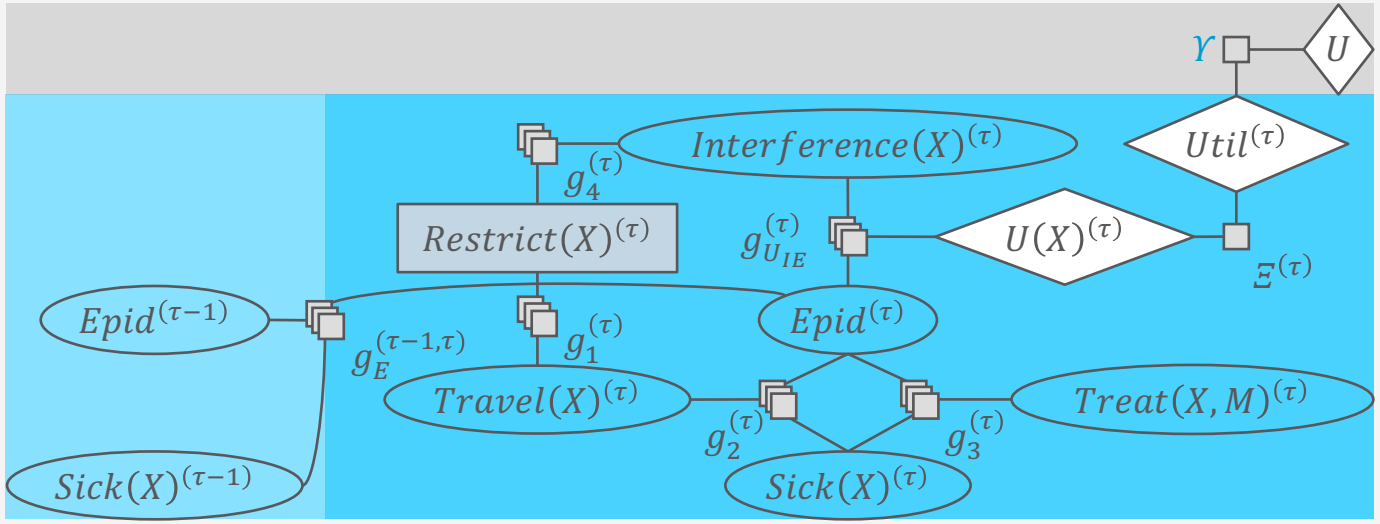
$$U(\boldsymbol{r}^{(0)}, \boldsymbol{r}^{(1)}, \boldsymbol{r}^{(2)}, \dots) = U(\boldsymbol{r}^{(0)}) + U(\boldsymbol{r}^{(1)}) + U(\boldsymbol{r}^{(2)}) + \cdots$$

- Discounted
  - Using a discount factor $\gamma \in [0,1]$
    - Reward now may be more important than one in t steps
    - If $\gamma = 1$: additive
  - Utility of a sequence = sum over discounted individual utilities

$$U(\boldsymbol{r}^{(0)}, \boldsymbol{r}^{(1)}, \boldsymbol{r}^{(2)}, \dots) = \gamma^0 \cdot U(\boldsymbol{r}^{(0)}) + \gamma^1 \cdot U(\boldsymbol{r}^{(1)}) + \gamma^2 \cdot U(\boldsymbol{r}^{(2)}) + \cdots = \sum_{t=0} \gamma^t U(\boldsymbol{r}^{(t)})$$

Sequential Decision Model

- Sequential decision model $G = (G^0, G^{\rightarrow}, \Upsilon)$ is given by
  - Decision model G^0 describing the intra-time slice behaviour for $t = 0$ (as before)
  - Decision model $G^{\rightarrow}$ describing the intra- and inter-slice behaviour for $t > 0$ (as before)
  - Sequential combination function Υ such as (discounted) addition
- Semantics given by unrolling G for T steps to form an episodic decision model as defined before
  - Ground to get a propositional model



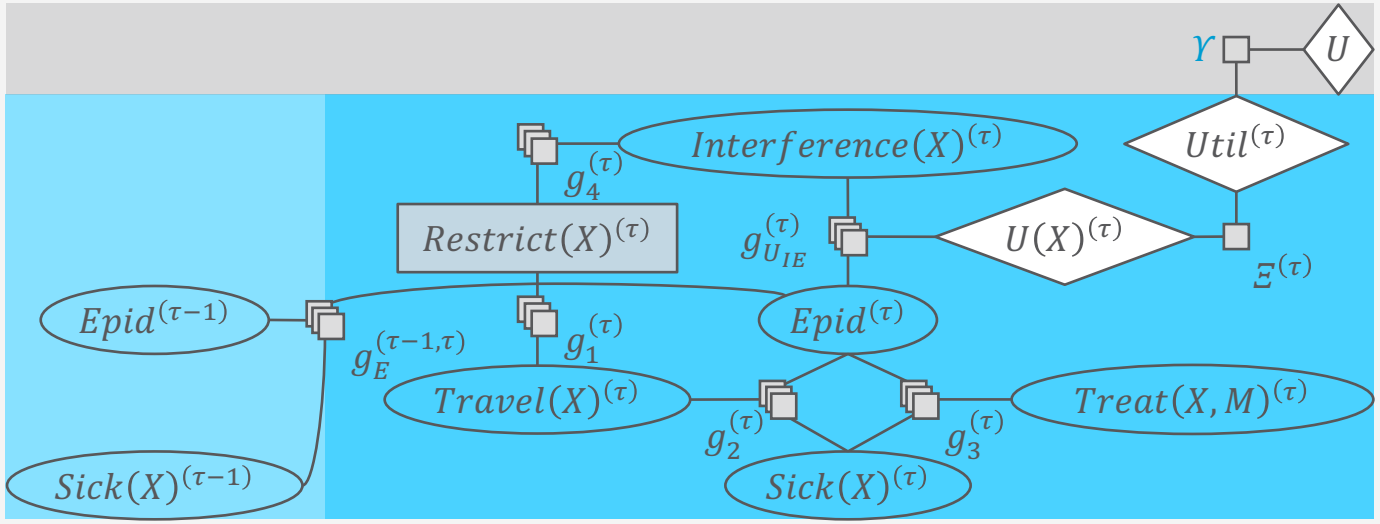
Sequential EU Query

- Sequential EU query at a current step t with a horizon κ in a sequential decision model G

$$eu\left(\boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}, \boldsymbol{d}^{(t:t+\kappa)}\right)$$

$$= \sum_{\boldsymbol{v}^{(t:t+\kappa)} \in \mathrm{ran}\left(rv\left(G_U^{(t:t+\kappa)}\right)\right)} P\left(\boldsymbol{v}^{(t:t+\kappa)} \middle| \boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}, \boldsymbol{d}^{(t:t+\kappa)}\right) \sum_{k=0}^{\kappa} \gamma^k \Xi^{(t+k)}\left(G_U^{(t+k)}\right)$$

  - Additive/discounted combination (Υ)
  - In terms of semantics, unroll G for $t + \kappa$ steps and answer an "episodic" EU query over "future" decisions $\boldsymbol{d}^{(t:t+\kappa)}$ given observations $\boldsymbol{e}^{(1:t)}$ and "past" decisions $\boldsymbol{d}^{(1:t-1)}$ as evidence
    - Observation: $G_U^{(1:t-1)}$ irrelevant at t

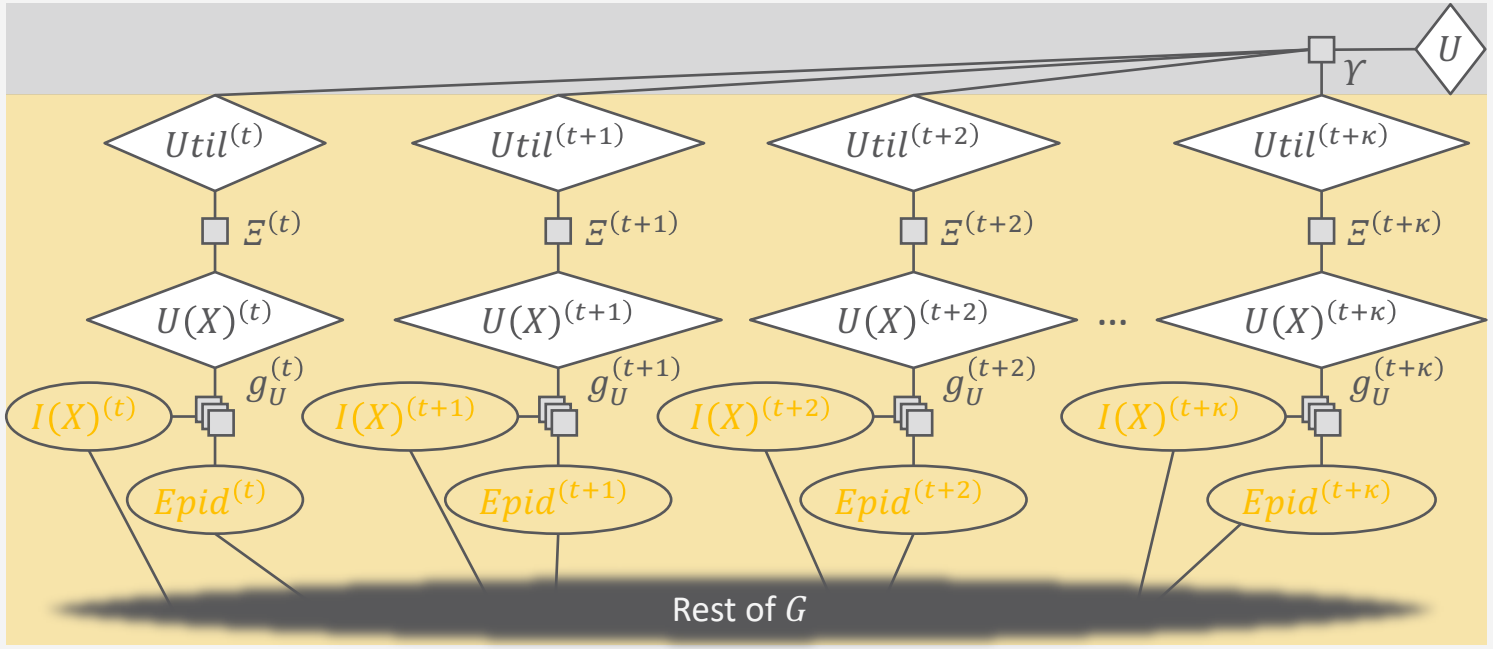


EU Query over Time: Exact

$$eu\left(\boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}, \boldsymbol{d}^{(t:t+\kappa)}\right) = \sum_{\boldsymbol{v}^{(t:t+\kappa)} \in \mathrm{ran}\left(rv\left(G_U^{(t:t+\kappa)}\right)\right)} P\left(\boldsymbol{v}^{(t:t+\kappa)} \middle| \boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}, \boldsymbol{d}^{(t:t+\kappa)}\right) \sum_{k=0}^{\kappa} \gamma^k \Xi^{(t+k)}\left(G_U^{(t+k)}\right)$$

- Problem with answering an EU query exactly: Query terms from all slices $t : t + \kappa$ involved
  - Unlikely that query terms are independent
  - Even assuming current model $G^{(t)}$ independent from the past with $\boldsymbol{e}^{(1:t-1)}, \boldsymbol{d}^{(1:t-1)}$ set, still need to unroll model for κ steps

Realistically **not** computable!



EU Query over Time: Approximate

- In episodic model, assume independence between PRVs of utility parfactors given $\boldsymbol{e}, \boldsymbol{d}$ to factorise query
  - Set of smaller queries instead of one large
  - Exact answer if independences hold
- Does that help with sequential models?
  – Yes and no.
  - Assuming independence allows for factorising query into queries for each slice
  - BUT: Query answer will be approximate as independence between slices not reasonable in a sequential model just given $\boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}$
    - Otherwise, a set of episodic models suffice

EU Query over Time: Approximate

- $eu\left(\boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}, \boldsymbol{d}^{(t:t+\kappa)}\right)$ when assuming the inter-slice independence:

$$eu\left(\boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}, \boldsymbol{d}^{(t:t+\kappa)}\right)$$

$$= \sum_{\boldsymbol{v}^{(t:t+\kappa)} \in \mathrm{ran}\left(rv\left(G_U^{(t:t+\kappa)}\right)\right)} P\left(\boldsymbol{v}^{(t:t+\kappa)} \middle| \boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}, \boldsymbol{d}^{(t:t+\kappa)}\right) \sum_{k=0}^{\kappa} \gamma^k \cdot \Xi^{(t+k)} \overbrace{\left[\phi_{U_1}^{(t+k)}\left(\boldsymbol{r}_1^{(t+k)}\right), \ldots, \phi_{U_m}^{(t+k)}\left(\boldsymbol{r}_m^{(t+k)}\right)\right]}^{\boldsymbol{r}_u^{(t+k)} = \pi_{\boldsymbol{R}_u}\left(\boldsymbol{v}^{(t:t+\kappa)}\right)}$$

$$= \sum_{\boldsymbol{v}^{(t:t+\kappa)} \in \mathrm{ran}\left(rv\left(G_U^{(t:t+\kappa)}\right)\right)} \prod_{k'=0}^{\kappa} P\left(\boldsymbol{v}^{(t+k')} \middle| \boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}, \boldsymbol{d}^{(t:t+k')}\right) \sum_{k=0}^{\kappa} \gamma^k \cdot \Xi^{(t+k)} \left[\phi_{U_1}^{(t+k)}\left(\boldsymbol{r}_1^{(t+k)}\right), \ldots, \phi_{U_m}^{(t+k)}\left(\boldsymbol{r}_m^{(t+k)}\right)\right]$$

$$= \sum_{k=0}^{\kappa} \gamma^k \sum_{\boldsymbol{v}^{(t+k)} \in \mathrm{ran}\left(rv\left(G_U^{(t+k)}\right)\right)} P\left(\boldsymbol{v}^{(t+k)} \middle| \boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}, \boldsymbol{d}^{(t:t+k)}\right) \cdot \Xi \left[\phi_{U_1}^{(t+k)}\left(\boldsymbol{r}_1^{(t+k)}\right), \ldots, \phi_{U_m}^{(t+k)}\left(\boldsymbol{r}_m^{(t+k)}\right)\right]$$

$$= \sum_{k=0}^{\kappa} \gamma^k \cdot eu\left(\boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t+k-1)}, \boldsymbol{d}^{(t+k)}\right)$$

One EU query per step

EU Query over Time: Approximate

- Solving a sequential EU query reduces to solving individual EU queries per step

$$eu\left(\boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}, \boldsymbol{d}^{(t:t+\kappa)}\right) = \sum_{k=0}^{\kappa} \gamma^k \cdot eu\left(\boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t+k-1)}, \boldsymbol{d}^{(t+k)}\right)$$

  - Solve individual EU queries based on intra-slice assumptions
    - E.g., assuming additive combination function or independence between intra-slice utility parfactors

$$\begin{aligned}
&eu\left(\boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}, \boldsymbol{d}^{(t:t+\kappa)}\right) \\
&= \sum_{k=0}^{\kappa} \gamma^k \sum_{\boldsymbol{v}^{(t+k)} \in \mathrm{ran}\left(rv\left(G_U^{(t+k)}\right)\right)} P\left(\boldsymbol{v}^{(t+k)} \middle| \boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}, \boldsymbol{d}^{(t:t+\kappa)}\right) \sum_{u=1}^{m} \phi_u^{(t+k)}\left(\boldsymbol{r}_u^{(t+k)}\right) \\
&= \sum_{k=0}^{\kappa} \gamma^k \sum_{u=1}^{m} \sum_{\boldsymbol{r}_u^{(t+k)} \in \mathrm{ran}\left(\mathrm{rv}\left(g_u^{(t+k)}\right)\right)} \phi_u^{(t+k)}\left(\boldsymbol{r}_u^{(t+k)}\right) P\left(\boldsymbol{r}_u^{(t+k)} \middle| \boldsymbol{e}^{(1:t)}, \boldsymbol{d}^{(1:t-1)}, \boldsymbol{d}^{(t:t+\kappa)}\right)
\end{aligned}$$

Sequential MEU Problem

- Given a sequential decision model G, evidence $\boldsymbol{e}^{(0:t)}$, previous decisions $\boldsymbol{d}^{(0:t-1)}$, and a horizon (integer) κ
- Sequential MEU problem
  - Find the sequential action assignment that yields the highest expected utility in G
  - Formally,

$$MEU\left(\boldsymbol{e}^{(0:t)}, \boldsymbol{d}^{(0:t-1)}, \kappa\right) = \left(\boldsymbol{d}^{*}, eu\left(\boldsymbol{e}^{(0:t)}, \boldsymbol{d}^{(0:t-1)}, \boldsymbol{d}^{*}\right)\right)$$

$$\boldsymbol{d}^{*} = \underset{\boldsymbol{d}^{(t:t+\kappa)} \in \mathrm{ran}\left(\boldsymbol{D}^{(t:t+k)}\right)}{\arg\max} eu\left(\boldsymbol{e}^{(0:t)}, \boldsymbol{d}^{(0:t-1)}, \boldsymbol{d}^{(t:t+\kappa)}\right)$$

  - Size of $\mathrm{ran}\left(\boldsymbol{D}^{(t:t+\kappa)}\right)$ exponential in number of groups (as with episodic decision making) and κ

Solving a Sequential MEU Problem: MEU-LDJT

- Given a sequential decision model G, evidence $\boldsymbol{e}^{(0:t)}$, previous decisions $\boldsymbol{d}^{(0:t-1)}$, a horizon (integer) κ, and incoming filtering, prediction, hindsight queries $\boldsymbol{Q}$
- Procedure:
  - Construct FO jtrees $J^0, J^{\rightarrow}$
  - For $t = 0, \ldots,$
    - Instantiate $J^{(t)}$, add $\alpha^{(t-1)}$, enter $\boldsymbol{e}^{(t)}$, pass messages, answer $\boldsymbol{Q}$
    - MEU loop: For each possible sequential action assignment $\boldsymbol{d}^{(t:t+\kappa)}$
      - $eu \leftarrow$ Calculate EU query, exactly or approximately
      - if $eu > eu^*$, then store $\boldsymbol{d}^{(t:t+\kappa)}$ in $\boldsymbol{d}^*$, setting $eu^* \leftarrow eu$
      - Output $\boldsymbol{d}^*$ or act out $\boldsymbol{d}^*$: Set $\boldsymbol{d}^{(t)}$ (and send $\boldsymbol{d}^{(t)}$ to an execution platform)
    - Calculate $\alpha^{(t)}$

Calculates MEU problem anew in each step; adapt such that $\boldsymbol{d}^*$ is acted out for κ steps, or until following $\boldsymbol{d}^*$ no longer appears safe

- Store $\boldsymbol{d}^*$ and, instead of going into the MEU loop, set the next $\boldsymbol{d}^{(t)}$ from $\boldsymbol{d}^*$

Solving a Sequential MEU Problem: MEU-LDJT

- Calculating a sequential EU query for $\boldsymbol{d}^{(t:t+\kappa)}$ in a sequential FO jtree
  - Exact:
    - Unroll FO jtree for κ steps, set $\boldsymbol{d}^{(t:t+\kappa)}$, compute probability query over $\mathrm{rv}\left(G_U^{(t:t+\kappa)}\right)$, calculate expected utility
  - Approximate:
    - Overall expected utility $U \leftarrow 0$
    - For $k = 0, \dots, \kappa$
      - Instantiate FO jtree $J^{(t+k)}$, add $\alpha^{(t+k-1)}$ (if not already done)
      - Set $\boldsymbol{d}^{(t+k)}$, pass (or update) messages, compute probability query over $\mathrm{rv}\left(G_U^{(t+k)}\right)$, calculate expected utility for $t+k$ (set of EU queries if assuming intra-slice independence), add result to U
      - Calculate $\alpha^{(t+k)}$

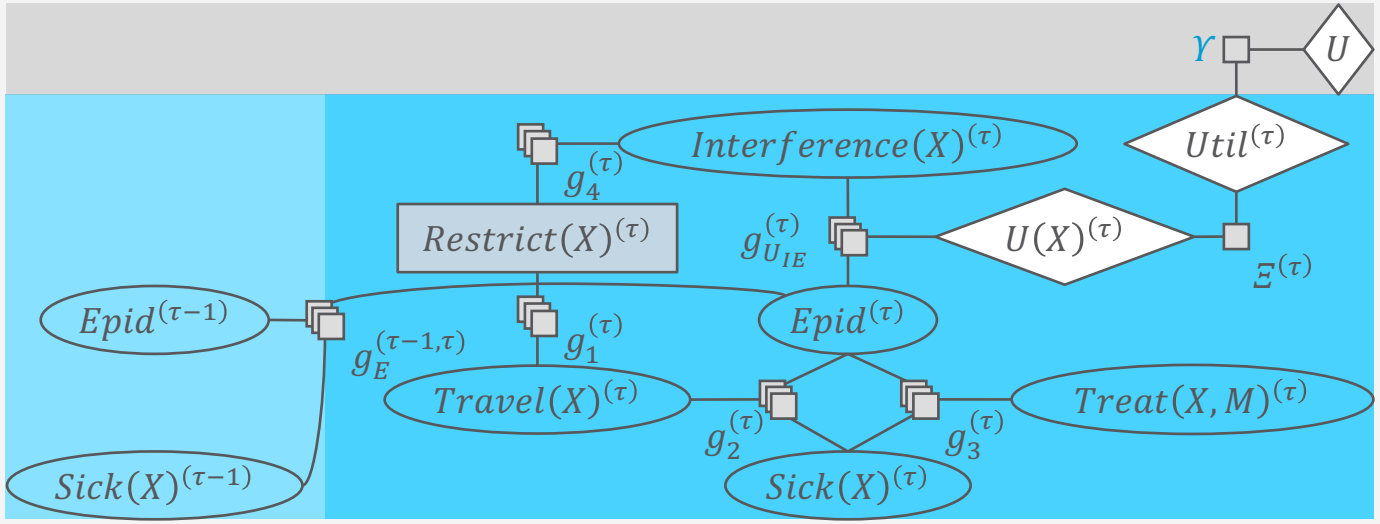
Sequential MEU Problem: Example

- Given no evidence, current time step $t = 2$, and $\kappa = 2$

$$\boldsymbol{d}^* = \arg\max_{i \in \{1,\dots,8\}} eu\left(\boldsymbol{d}_i^{(2:4)}\right)$$

  - Test eight sequential action assignments
    - For each $\boldsymbol{d}_i^{(2:4)}$
      - Set $\boldsymbol{d}_i^{(2:4)}$ in unrolled FO jtree $\boldsymbol{J}^{(2:4)}$ and answer EU query $eu\left(\boldsymbol{d}_i^{(2:4)}\right)$ or
      - Set $\boldsymbol{d}_i^{(2:4)}$ incrementally in $\boldsymbol{J}^{(t+k)}$ and add up EU queries $eu\left(\boldsymbol{d}_i^{(t+k)}\right)$
  - Output argmax $\boldsymbol{d}_i^{(2:4)}$
    - Or act out $\boldsymbol{d}_i^{(2)}$

	$d^{(2)}$	$d^{(3)}$	$d^{(4)}$
$\boldsymbol{d}_1^{(2:4)}$	ban	ban	ban
$\boldsymbol{d}_2^{(2:4)}$	ban	ban	$free$
$\boldsymbol{d}_3^{(2:4)}$	ban	$free$	ban
$\boldsymbol{d}_4^{(2:4)}$	ban	$free$	$free$
$\boldsymbol{d}_5^{(2:4)}$	$free$	ban	ban
$\boldsymbol{d}_6^{(2:4)}$	$free$	ban	$free$
$\boldsymbol{d}_7^{(2:4)}$	$free$	$free$	ban
$\boldsymbol{d}_8^{(2:4)}$	$free$	$free$	$free$



Acting

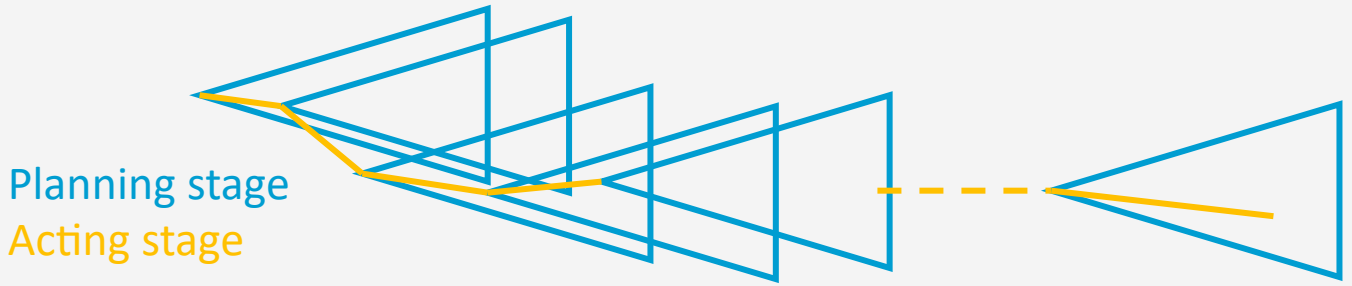
- After computing a sequential action assignment, assignment is acted out, including:
  - Actuators get commands to carry out the action behind a current assignment,
  - Internal state is updated: Decision PRVs are set, (messages are passed,) and then time moves on
- Agent can then continue to act according to the sequential assignment or recalculate as new evidence comes in
  - Recalculate once every $k \leq \kappa$ steps

Next three slides adapted from material provided by Dana Nau, http://www.laas.fr/planning

Using Decision Making in Acting

- Receding horizon:
  - Call meu, obtain a sequential action assignment π, perform 1st action, call meu again …
    - meu refers to an implementation solving an MEU problem
  - Like game-tree search (chess, checkers, etc.)
- Useful when unpredictable things are likely to happen
  - Re-plans immediately
- Potential problem:
  - May pause repeatedly while waiting for meu to return

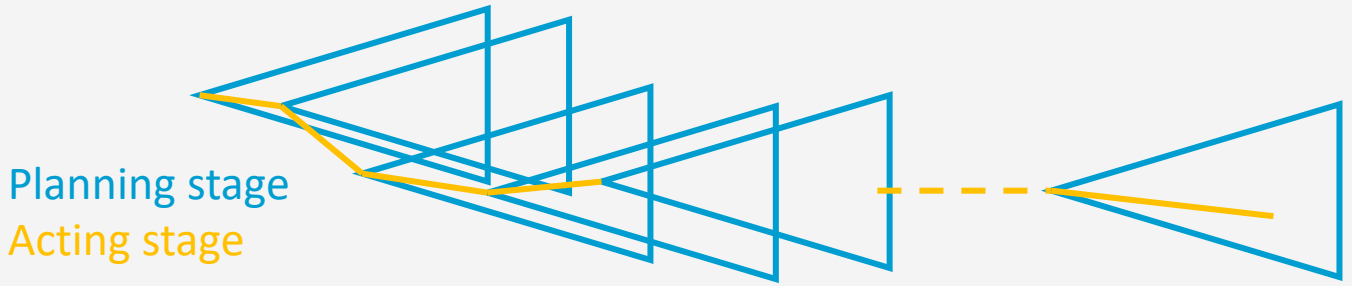
```
Run-MEU(G, κ)
   while e ← new evidence do
      absorb e in G
      π ← meu(G, κ)
      d ← pop-first(π)
      perform d
      update G with d
```



Using Decision Making in Acting

- Lazy look-ahead:
  - Call meu, execute the sequential action assignment as far as possible, do not call meu again unless necessary
  - Simulate tests whether the assignment will execute correctly
    - Lower-level refinement, physics-based simulation, prediction accuracy < some threshold
- Potential problems
  - May might miss opportunities to replace π with a better assignment
  - Without Simulate, may not detect problems until it is too late

```
Run-Lazy-MEU(G,κ)
   while e ← new evidence do
      absorb e in G
      π ← meu(G,κ)
      while π ≠ () and
            Simulate(G,π) ≠ failure do
         d ← pop-first(π)
         perform d
         update G with d
```



Using Decision Making in Acting

- May detect opportunities earlier than Run-Lazy-MEU
  - But may miss some that Run-MEU would find
- Without Simulate, may fail to detect problems until it is too late
  - Not as bad at this as Run-Lazy-MEU

```
Run-Concurrent-MEU(G,κ)
   π ← ⟨⟩
   // thread 1 + 2 run concurrently
thread 1:
   while e ← new evidence do
      absorb e in G
      π ← meu(G,κ)
thread 2:
   while e ← new evidence do
      absorb e in G
      if π ≠ () and
            Simulate(G,π) ≠ failure then
         d ← pop-first(π)
         perform d
         update G with d
```

Offline vs. Online Decision Making

- Online decision making: Depends on how far one looks into the future
  - Can extend the look-ahead further and further, propagating future effects back to present to see if the current best decision still holds
  - Extending the look-ahead to infinity, one will eventually reach a fix point: *offline decision making*
- Offline decision making
  - Solving a partially observable Markov decision process (POMDP) – Ch. 17 in Russell/Norvig's AIMA3
    - Basically find the steady state in terms of actions that leads to the maximum expected utility
      - Look-up table, fast during acting, huge overhead beforehand
      - Reacting to extreme situations/evidence no possible
    - In basic form, no longer factorised model but a transition function over complete state space
  - Part of the lecture: *Automated Planning and Acting* (next winter term)

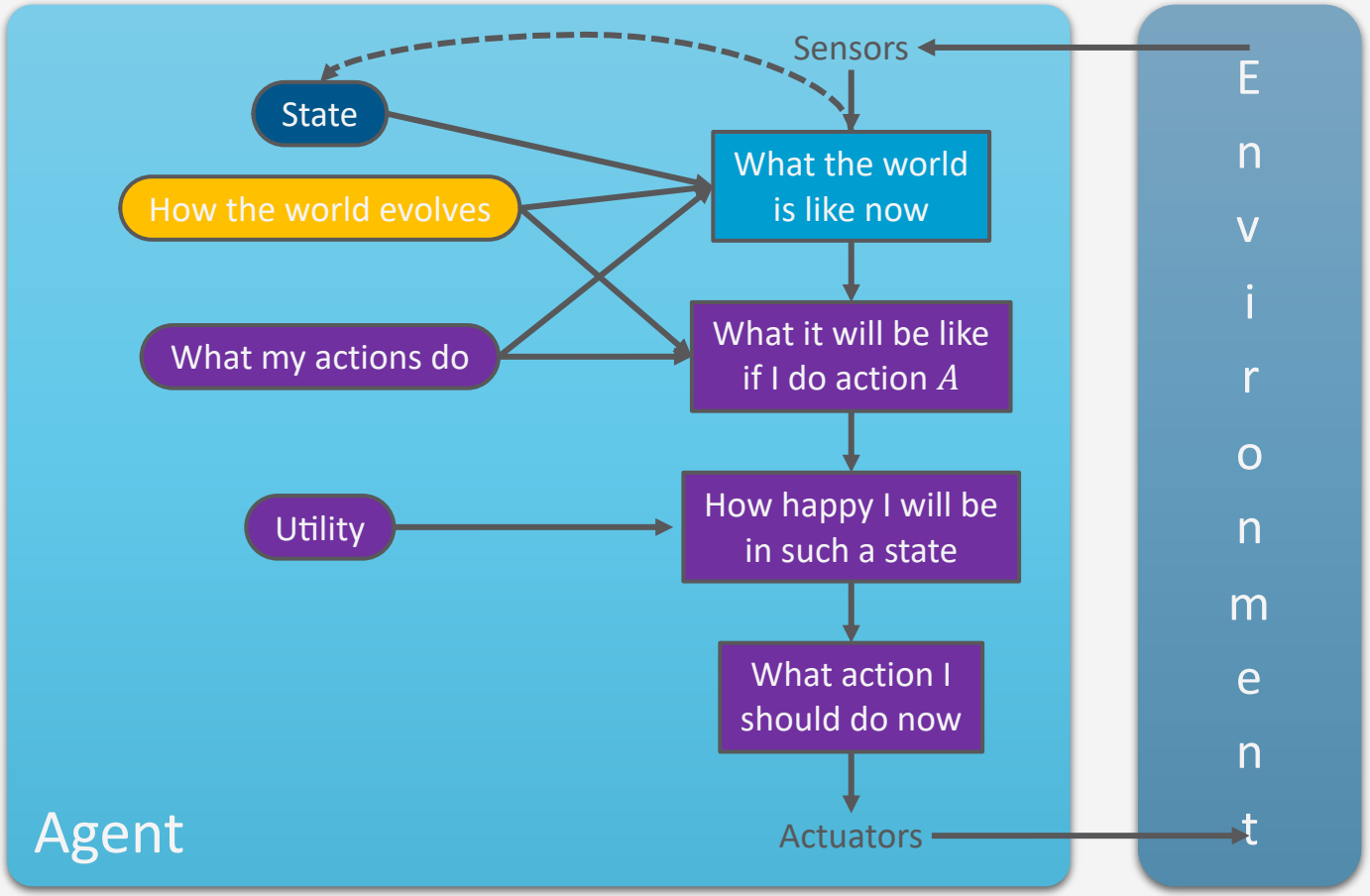
Interim Summary

- Sequential decision models as a combination of
  - Sequential models for temporal / sequential aspect
  - Decision models for decision making
- Sequential expected utilities
  - Additive or discounted utilities
  - Exact solution virtually impossible to compute
  - Approximation by summing over EU query result for each step
- Inference based on LDJT
  - Proceed in time to answer EU queries part of a sequential EU query
- Acting
  - Immediate look-ahead, lazy run until failed simulation or sequence acted out, both concurrently

Contents in this Lecture Related to *Utility-based Agents*

- Further topics
  3. (Episodic) PRMs
  4. Lifted inference (in episodic PRMs)
  5. Lifted learning (of episodic PRMs)
  6. Lifted sequential PRMs and inference
  7. Lifted decision making

Sequential Decision Models
- Uncertainty modelled by probabilities
- Relational aspect using logical variables
- Temporal aspect by time indexing
- Decisions and effects by actions & utilities in a sequential model



Outline: 7. Lifted Decision Making

A. Utility theory
- Preferences, maximum expected utility (MEU) principle
- Utility function, multi-attribute utility theory

B. Static decision making
- Modelling, semantics, inference tasks
- Inference algorithm: LVE for MEU as an example

C. Sequential decision making
- Modelling, semantics, sequential MEU problem
- Inference algorithm: LDJT for MEU as an example
- Acting

$\Longrightarrow$ Next: Continuous Space