Composite graphical models for reasoning about uncertainties, feasibilities, and utilities

Cédric Pralet[1,3], Gérard Verfaillie[2], and Thomas Schiex[3]

[1] LAAS-CNRS, Toulouse, France cpralet@laas.fr
[2] ONERA, Centre de Toulouse, France gerard.verfaillie@onera.fr
[3] INRA, Castanet Tolosan, France tschiex@toulouse.inra.fr

Abstract. In this paper, we define a generic algebraic framework that covers several AI formalisms used to represent uncertainties, feasibilities, or utilities. This includes hard, soft, mixed, quantified, or stochastic constraint satisfaction problems, Bayesian and Gibbsian networks, chain graphs, influence diagrams, probabilistic or possibilistic Markov decision processes. This is made possible by the fact that these formalisms can be described as "graphical models", using local relations to express a global knowledge. Composite graphical models include different types of relations to model uncertainties on the environment, feasibilities on decisions, and utilities expressing preferences or hard requirements. Usual queries can then be interpreted in this framework as a sequence of quantification/elimination of variables with dedicated operators. Using the proposed framework, it will be possible to better understand the links between existing formalisms and to develop generic efficient parameterized algorithms.

1 Existing frameworks

In the last decades, several representation frameworks have been developed in the AI community to reason about uncertainties, feasibilities, and utilities. *Constraint satisfaction problems* (CSP [1]) represent problems in which local constraints between discrete variables express facts or requirements. Valued/semiring CSP [2] extend constraint networks to represent uncertain facts or soft requirements (covering classical, fuzzy, additive, lexicographic, probabilistic CSP…). *Mixed CSP* [3] introduce a distinction between controllable and uncontrollable variables. Similarly, *quantified CSP* [4] and *stochastic CSP* [5] extend CSP with universally and randomly quantified variables. Similar works exist in the SAT (boolean satisfiability) community [6]. *Bayesian networks* [7] represent problems that involve local oriented dependencies between random variables. *Chain graphs* [8] extend Bayesian networks by adding non-oriented dependencies, as in *Gibbsian networks*. In another direction, *influence diagrams* [9] extend Bayesian networks by adding non-random decision variables and utility variables. *Markov decision processes* (MDP), possibly partially observable [10, 11], can represent sequential decision problems under uncertainty. Finally, *factored MDP* [12] extend MDP to variable-based representations. Related works use possibility theory [13, 14], or Spohn's theory of epistemic beliefs [15].

2 Motivations

One can observe significant similarities between these frameworks, which share common algebraic properties. We present some examples to illustrate this point.

Graphical models Graphical models are defined by a finite set of variables $V = \{x_1, \ldots, x_n\}$, each variable having a finite domain of values $Dom(x_i)$, and by a finite set of local functions $L = \{f_1, \ldots, f_m\}$ taking their values in a specific domain. The combination of these functions with a specific operator offers a space-tractable definition of a global (joint) function on all variables.

In the CSP (or SAT) framework [1], the set L contains relations (or constraints/clauses) mapping tuples of values to $\{true, false\}$. A usual query on a CSP is to find an assignment of the variables satisfying all constraints. It can be formulated as $\exists x_1 \exists x_2 \ldots \exists x_n \ (f_1 \wedge f_2 \wedge \ldots \wedge f_m)$ (or "does there exist a value for x_1, x_2, $\ldots$ x_n such that the conjunction of the constraints is satisfied"). Assuming $true \succ false$, checking consistency is equivalent to computing:

$$\max_{x_1} \max_{x_2} \ldots \max_{x_n} (f_1 \wedge f_2 \wedge \ldots \wedge f_m) \tag{1}$$

The values that optimize this query define a solution. In this query, local relations are *combined* with $\wedge$, and the *elimination* (or projection) operator max is applied to each variable. Replacing $\wedge$ by their respective combination operator $\oplus$ (resp. $\times$) in Query 1 yields the usual query of valued (resp. totally ordered semiring) CSP [2].

Similarly, in a *Bayesian network* [7], L is a set of local conditional probability distributions $L = \{P(x_i \,|\, pa(x_i))\,,\, x_i \in V\}$ where $pa(x_i)$ is the set of parent variables of x_i defined by a directed acyclic graph (DAG) G. Such a network represents a joint probability distribution $P(V)$ on all variables as a combination of local conditional probability distributions, exactly as the combination of local constraints in a CSP implicitly models a global constraint on all variables. Various queries can be considered on a Bayesian network, such as computing $P(x_1)$, the probability distribution on x_1, equal to:

$$\sum_{x_2} \cdots \sum_{x_n} \left(\prod_{x_i \in V} P(x_i \,|\, pa(x_i)) \right) \tag{2}$$

In this case, local functions are combined with $\times$, and elimination is done with $+$, instead of $\wedge$ and max respectively for checking consistency of a CSP.

These graphical models are said to be *simple* because they only involve *one type of variable* (decision variables for CSP, random variables for Bayesian networks), *one type of function* (constraints for CSP, conditional probability distributions for Bayesian networks), *one type of combination operator* ($\wedge$ for CSP, $\times$ for Bayesian networks), and *one type of elimination operator* (max for checking consistency of a CSP, $+$ for computing probabilities in a Bayesian network). However, many related formalisms have introduced *several types of variables, functions, combination and elimination operators*. Such frameworks can informally be denoted as *composite* graphical models.

Composite graphical models *Stochastic CSP* [5] involves two kinds of variables: decision variables d_i, and stochastic variables s_j describing the environment. A global probability distribution on stochastic variables is expressed as a combination of local probability distributions that can be simply $P(s_j)$ if the stochastic variables are assumed to be mutually independent. A set of local constraints $C = \{c_1, \ldots, c_m\}$ is also defined. An example of query with two decisions steps is:

$$\max_{d_1} \max_{d_2} \sum_{s_1} \max_{d_3} \max_{d_4} \sum_{s_2} (P(s_1) \times P(s_2)) \times (c_1 \times \ldots \times c_m) \tag{3}$$

Uncertainties given by the local probability distributions are combined with $\times$, utilities expressed by the constraints are combined with $\times$, uncertainties and utilities are combined together with $\times$, environment (stochastic) variables are eliminated with $+$, and decision variables are eliminated with max.

Another example of a composite graphical model is given by *influence diagrams* [9] or *finite horizon possibilistic and probabilistic MDP* [10, 11, 14]. A MDP describes the evolution of the environment by time-steps. An environment variable s_t describing the state of the environment is associated with each time-step t, as well as a decision variable d_t representing the decision made at time t. In the following, the initial state s_1 is assumed to be known.

In a *probabilistic MDP*, uncertainties on the evolution of the environment are described with local probability distributions $P(s_{t+1} \mid s_t, d_t)$ of being in state s_{t+1} at time $t+1$, given s_t and d_t. Utilities on the environment and on the decisions are specified via local additive rewards $R(s_t, d_t)$ associated with each time-step. If there are T time-steps, the associated query is:

$$\max_{d_1} \sum_{s_2} \max_{d_2} \ldots \sum_{s_T} \max_{d_T} \left(\prod_{t \in [1, T-1]} P(s_{t+1} \mid s_t, d_t) \right) \times \left(\sum_{t \in [1, T]} R(s_t, d_t) \right) \tag{4}$$

to obtain an optimal policy under uncertainty. Uncertainties given by the probabilities are combined with $\times$, utilities given by the rewards are combined with $+$, uncertainties and utilities are combined with $\times$, environment variables (the s_t) are eliminated with $+$, and decision ones (the d_t) are eliminated with max.

In a *possibilistic MDP*, uncertainties on the evolution of the environment are described with possibility distributions $\pi(s_{t+1} \mid s_t, d_t)$ of being in state s_{t+1} at time $t+1$, given s_t and d_t. Utilities on the environment and on the decisions are specified via local preferences $\mu(s_t, d_t)$ associated with each time-step. If there are T time-steps, the query associated with the pessimistic version of possibilistic MDP is:

$$\max_{d_1} \min_{s_2} \max_{d_2} \ldots \min_{s_T} \max_{d_T} \max \left(1 - \min_{t \in [1, T-1]} \pi(s_{t+1} \mid s_t, d_t) \ , \ \min_{t \in [1, T]} \mu(s_t, d_t) \right) \tag{5}$$

Uncertainties given by the possibilities are combined with min, utilities given by the preferences are combined with min, an uncertainty un and a utility ut are combined with $\max(1 - un, ut)$, environment variables (the s_t) are eliminated with min, and decision ones (the d_t) are eliminated with max. It appears that only the combination and elimination operators differ between Equations 4 and 5.

In MDP, the environment may restrain the possible decisions, as in STRIPS planning, where preconditions on actions are imposed. If $f(s_t, d_t)$ is a local boolean relation expressing whether a decision d_t is feasible or not in state s_t, Equation 4 becomes $\max_{d_1 \,/\, f(s_1,d_1)} \sum_{s_2} \max_{d_2 \,/\, f(s_2,d_2)} \ldots \sum_{s_T} \max_{d_T \,/\, f(s_T,d_T)} \alpha$, where $\alpha = \left(\prod_{t\in[1,T-1]} P(s_{t+1} \mid s_t, d_t)\right) \times \left(\sum_{t\in[1,T]} R(s_t, d_t)\right)$. To avoid indexed max operations, it is also possible to write this equation as:

$$\max_{d_1} \sum_{s_2} \max_{d_2} \ldots \sum_{s_T} \max_{d_T} \left(\bigwedge_{t\in[1,T]} f(s_t, d_t) \right) \star \alpha \qquad (6)$$

with $\star$ an operation used to combine feasibilities and utilities, defined by $\mathit{true} \star ut = ut$, $\mathit{false} \star ut = \Diamond$, $\Diamond$ being "ignored" by the elimination operators, i.e. $\max(ut, \Diamond) = \min(ut, \Diamond) = ut + \Diamond = ut$. Again, a sequence of variable eliminations is performed on a combination of local relations, local feasibility constraints being combined with $\wedge$, and combined with uncertainties and utilities with $\star$.

Global objective Equations 1 to 6 show that queries associated with existing frameworks all correspond to a sequence of application of elimination operators on a combination of local functions. The only differences between these frameworks lies in the types of the local functions involved and in the way information is combined and eliminated. The existing generic approach of valuation algebras [16, 17] is suitable for simple graphical models, but, being restricted to *simple* graphical models with one type of information, it cannot be *directly* used to solve queries on composite graphical models.

In order to deal in the most general way with graphical models where one may distinguish between environment variables (whose value is uncertain) and decision variables (whose value is chosen), between local relations expressing uncertainties, feasibilities and utilities, using several type of combination operators (for combining uncertainties, feasibilities, and utilities) and of elimination operators (typically min, max, or + may alternate in general queries), we designed a new algebraic framework, called *Uncertainty-Feasibility-Utility networks* (UFUs). Using such networks, simple or complex queries can be formulated uniformly using elimination operators (acting as quantifiers). This framework enables the integration of several algorithmic approaches independently explored in each framework: exact methods relying on tree search or variable/bucket elimination, approximate algorithms using sampling or local search, and local inference mechanisms such as local consistency enforcing (now available on general classes of functions [19]) and global constraints. It will also bring to light and capitalize on the many common underlying algorithmic ideas, therefore avoiding reinventing similar methods.

The Uncertainty-Feasibility-Utility networks framework is based on three main components that make the backbone of this paper:

- a *generic algebraic structure*, presented in Section 3, introduces operators specifying how to combine uncertainties, feasibilities, utilities, how to combine uncertainties with utilities, feasibilities with utilities, and how to eliminate on uncertainties, feasibilities, and utilities;

– a *problem* (Section 4) is then defined as a set of local functions between variables. Variables are either environment variables (with an uncertainty measure on the value they take), or decision variables (when their value is decisionally chosen by an agent). Three types of local functions are defined: uncertainty relations (modeling uncertainties on the environment variables), feasibility relations (modeling feasibility constraints on decision variables), and utility relations (modeling preferences or hard requirements, on any variable).
– *queries* on a problem (Section 5) are finally formulated as sequences of variable eliminations, e.g. to compute an optimal policy under uncertainty.

Example To give flesh to our definitions, we consider the following example: *Peter invites John and Mary (a divorced couple) to a business dinner in order to convince them to invest in his company. Peter knows that if John is present at the end of the dinner, he will invest* 10 *k\$. The same holds for Mary with* 50 *k\$. Peter knows that John and Mary will not come together (one of them has to baby-sit their child), that at least one of them will come, and that the case "John comes and Mary does not" occurs with a probability of* 0.6*. As for the menu, Peter can order fish or meat for the main course, and white or red for the wine. However, the restaurant refuses to serve fish and red wine together. John does not like white wine and Mary does not like meat. If the menu does not suit them, they will leave the dinner. If John comes, Peter does not want him to leave the dinner because he is his best friend.*

3 Definition of a generic algebraic structure

We start by describing the algebraic structure on which uncertainties, feasibilities, and utilities are defined. This structure involves ordered sets, combination and elimination operators, as well as specific elements.

Uncertainties can be expressed in several forms, as shown in the examples of probabilistic and possibilistic MDP (see Section 2). A first well-known form uses *probabilities*, as in probabilistic MDP. In this case, probabilities are combined with $\times$ (under independency hypothesis), and elimination on probabilities (usually called marginalisation) is done with $+$. But uncertainties can also be expressed as *possibility degrees* in $[0, 1]$, as in possibilistic MDP. In this case, possibilities are combined with an operator that can be min, and are eliminated with max. An interesting subcase appears when possibility degrees are either 0 or 1, i.e. when the uncertainties describe which world is completely possible or impossible. Uncertainties can then be combined with $\wedge$, and eliminated with $\vee$. A last example is given by Spohn's theory of epistemic beliefs [15]: in this case, uncertainties are *surprise degrees* in $\mathbb{N} \cup \{+\infty\}$, 0 being associated with non-surprising situations, and $+\infty$ being associated with completely surprising (impossible) situations. Surprise degrees are then combined with $+$ and eliminated with min.

To generalize those various frameworks reasoning about uncertainties, we define an *uncertainty structure* as a tuple $S_{un} = (E_{un}, \preceq_{un}, \oplus_{un}, \otimes_{un})$ such that:

- E_{un} is a set of elements called uncertainty degrees, totally ordered by $\preceq_{un}$, and with a minimal element 0_{un} (0_{un} will be associated with impossibility).
- $\oplus_{un}$ is a binary, associative, commutative, monotonic, closed operator on E_{un} (elimination operator), with 0_{un} as neutral element.
- $\otimes_{un}$ is a binary, associative, commutative, monotonic, closed operator on E_{un} (combination operator), with a neutral element $1_{un} \in E_{un}$ and 0_{un} as annihilator. Moreover, $\otimes_{un}$ distributes over $\oplus_{un}$.

It is a totally ordered commutative semiring. The previously described uncertainty modeling frameworks can easily be shown to be instances of uncertainty structures. Note that Dempster-Shafer belief functions [20] are not subsumed.

Feasibilities are used to express the fact that a decision is feasible or not. Therefore feasibilities are expressed on $\{true, false\}$. This set is equipped with the total order $\preceq_f$ verifying $false \prec_f true$. As a conjunction of decisions is feasible iff each decision of the conjunction is feasible, feasibilities are combined with $\wedge$. As a disjunction of decisions is feasible iff at least one decision of the disjunction is feasible, feasibilities are eliminated with $\vee$. Thus, feasibilities are expressed on a structure called a *feasibility structure*, which is always $S_f = (\{true, false\}, \preceq_f, \vee, \wedge)$. Note that S_f is also an uncertainty structure.

Utilities A usual approach to take into account utilities and uncertainty is the famous probabilistic expected utility theory [21], used e.g. in stochastic CSP and probabilistic MDP. In this theory, utilities are combined with $+$, and the expected utility formula $\sum_i p_i \times U_i$ combines uncertainties and utilities with $\times$ and eliminates utilities with $+$.

But other options exist. If uncertainties are possibility degrees as in possibilistic MDP, the possibilistic pessimistic expected utility theory [13] can be used: utilities are combined with min (as a risk minimization), an uncertainty un and a utility ut are combined with $\max(1 - un, ut)$, and utilities are eliminated with min. Last, if uncertainties are surprise degrees, the qualitative utility theory for Spohnian beliefs [15] can be used: when only "positive" utilities exist, it corresponds to combining utilities with $+$, combining uncertainties and utilities with $+$, and eliminating utilities with min.

To generalize these examples, we define first a *utility structure*, on which utilities are expressed, and then a *combination operator between uncertainties and utilities*. A utility structure is a tuple $S_{ut} = (E_{ut}, \preceq_{ut}, \oplus_{ut}, \otimes_{ut})$ such that:

- E_{ut} is a set of elements called utility degrees, totally ordered by $\preceq_{ut}$, and which contains a minimal element $\perp_{ut}$, associated with unacceptability.
- $\oplus_{ut}$ is a binary, associative, commutative, monotonic, closed operator on E_{ut} (elimination operator), with a neutral element $0_{ut} \in E_{ut}$ denoting indifference, between positive and negative utilities.
- $\otimes_{ut}$ is a binary, associative, commutative, monotonic, closed operator on E_{ut} (combination operator), with a neutral element $1_{ut} \in E_{ut}$ and $\perp_{ut}$ as annihilator.

A utility structure is different from an uncertainty structure: 0_{un} is the minimum element of E_{un} whereas 0_{ut} is not necessarily the minimum element of E_{ut}; both positive and negative utilities may exist. Then, a combination operator between uncertainties and utilities is an operator $\otimes_{un/ut} : E_{un} \times E_{ut} \to E_{ut}$ such that:

- $\otimes_{un/ut}$ is monotonic: for a given uncertainty, the higher the utility, the better, and the more a positive utility is believed (and the less a negative one), the better: $(ut_1 \preceq_{ut} ut_2) \to (un \otimes_{un/ut} ut_1 \preceq_{ut} un \otimes_{un/ut} ut_2)$
  $((un_1 \preceq_{un} un_2) \wedge (0_{ut} \preceq_{ut} ut)) \to (un_1 \otimes_{un/ut} ut \preceq_{ut} un_2 \otimes_{un/ut} ut)$
  $((un_1 \preceq_{un} un_2) \wedge (ut \preceq_{ut} 0_{ut})) \to (un_2 \otimes_{un/ut} ut \preceq_{ut} un_1 \otimes_{un/ut} ut)$
- utilities are "weighted" by uncertainties. These linearity axioms give a usual "lottery" semantics [21] to combined uncertainties and utilities. These axioms are also important for algorithmic reasons. Despite the strength of these axioms, they cover all the previous frameworks (see Table 1):
  $un_1 \otimes_{un/ut} (un_2 \otimes_{un/ut} ut) = (un_1 \otimes_{un} un_2) \otimes_{un/ut} ut$
  $un \otimes_{un/ut} (ut_1 \oplus_{ut} ut_2) = (un \otimes_{un/ut} ut_1) \oplus_{ut} (un \otimes_{un/ut} ut_2)$
  $(un_1 \oplus_{un} un_2) \otimes_{un/ut} ut = (un_1 \otimes_{un/ut} ut) \oplus_{ut} (un_2 \otimes_{un/ut} ut)$
  $1_{un} \otimes_{un/ut} ut = ut$, and $0_{un} \otimes_{un/ut} ut = 0_{ut}$

Finally, to combine feasibilities and utilities, we use, as in Equation 6, a *combination operator between feasibilities and utilities* $\star : \{true, false\} \times (E_{ut} \cup \{\Diamond\}) \to (E_{ut} \cup \{\Diamond\})$ such that for any $ut \in E_{ut} \cup \{\Diamond\}$, $true \star ut = ut$ and $false \star ut = \Diamond$, where $\Diamond$ is a special element ignored by elimination operators: $\max(ut, \Diamond) = \min(ut, \Diamond) = ut \oplus_{ut} \Diamond = ut$ (imposing $\max(ut, \Diamond) = \min(ut, \Diamond) = ut$ is fine since $\preceq_{ut}$ is a total order on E_{ut}, and not on $E_{ut} \cup \{\Diamond\}$). $\star$ enables us to write $\max_{x, f(x)=true} d(x)$ as $\max_x f(x) \star d(x)$, which gives feasibilities the same status as uncertainties and utilities. We also extend $\otimes_{un/ut}$ by $un \otimes_{un/ut} \Diamond = \Diamond$, which implies $un \otimes_{un/ut} (f \star ut) = f \star (un \otimes_{un/ut} ut) = f \star un \otimes_{un/ut} ut$.

An uncertainty-utility structure is defined as a triple $(S_{un}, S_{ut}, \otimes_{un/ut})$ such that S_{un} is an uncertainty structure, S_{ut} is a utility structure and $\otimes_{un/ut}$ is a combination operator between uncertainties and utilities. The structure for feasibility constraints is always $S_f = (\{true, false\}, \preceq_f, \vee, \wedge)$. Similarly, $\star$ does not change. Classical uncertainty-utility structures satisfying all the previous axioms are shown in Table 1. Our dinner problem uses probabilistic expected utility (row 1 in Table 1).

	E_{un}	$\preceq_{un}$	$\oplus_{un}$	$\otimes_{un}$	$0_{un}, 1_{un}$	E_{ut}	$\preceq_{ut}$	$\oplus_{ut}$	$\otimes_{ut}$	$\bot_{ut}, 0_{ut}, 1_{ut}$	$\otimes_{un/ut}$
1	$\mathbb{R}^+$	$\leq$	$+$	$\times$	$0, 1$	$\mathbb{R} \cup \{-\infty\}$	$\leq$	$+$	$+$	$-\infty, 0, 0$	$\times$
2	$\mathbb{R}^+$	$\leq$	$+$	$\times$	$0, 1$	$\mathbb{R}^+$	$\leq$	$+$	$\times$	$0, 0, 1$	$\times$
3	$[0, 1]$	$\leq$	max	min	$0, 1$	$[0, 1]$	$\leq$	max	min	$0, 0, 1$	min
4	$[0, 1]$	$\leq$	max	min	$0, 1$	$[0, 1]$	$\leq$	min	min	$0, 1, 1$	$\max(1-un, ut)$
5	$\mathbb{N} \cup \{\infty\}$	$\geq$	min	$+$	$\infty, 0$	$\mathbb{N} \cup \{\infty\}$	$\geq$	min	$+$	$\infty, \infty, 0$	$+$

Table 1. Uncertainty-utility structure in - 1. probabilistic expected utility - 2. probabilistic expected satisfaction - 3. optimistic possibilistic expected utility - 4. pessimistic possibilistic expected utility - 5. non-bipolar qualitative utility for Spohnian beliefs.

4 Definition of problems

We can now combine this algebraic structure with the notion of graphical models using local functions as in Equations 1 to 6. Formally, a local function on E is a function $L_i : Dom(S(L_i)) \to E$, where $S(L_i)$ is a set of variables called the *scope* of L_i. A local function on E_{un} (the set of uncertainty degrees) is called an *uncertainty relation*, a local function on $\{true, false\}$ (the set of feasibility degrees) is called a *feasibility relation*, and a local function on E_{ut} (the set of utility degrees) is called a *utility relation*. A feasibility relation can obviously be modeled as a constraint; an uncertainty relation can be modeled as a constraint if it takes its values in $\{0_{un}, 1_{un}\}$ (forbidden tuples are mapped on 0_{un}, allowed ones on 1_{un}); similarly, a utility relation can be modeled as a constraint if it takes its values in $\{\perp_{ut}, 1_{ut}\}$. Eliminating a variable x with an operator op from a local function L_i on a set of variables S consists of computing, for each assignment A of $S - \{x\}$, $(op_x\, L_i)(A) = op_{a \in Dom(x)}\, L_i(A.(x = a))$, i.e. a function on $S(L_i) - \{x\}$.

Given a finite set of variables V, partitioned into V_E, the set of environment variables, and V_D, the set of decision ones, we want to express:

- a global uncertainty degree $\mathrm{U{\scriptsize N}}_V$ on the environment variables as a combination of uncertainty relations. $\mathrm{U{\scriptsize N}}_V$ satisfies *normalization conditions*, according to which the disjunction of all the situations gives the same uncertainty degree 1_{un}. The normalization will appear locally through local normalization conditions imposed on uncertainty relations;
- a global feasibility degree F_V on the decision variables as a combination of feasibility relations. F_V satisfies *normalization conditions*, according to which one decision is feasible in any situation; this is semantically justified since e.g. doing nothing is always possible (even if unacceptable). The normalizations will appear locally through local normalization conditions imposed on feasibility relations;
- a global utility degree $\mathrm{U{\scriptsize T}}_V$ on all variables as a combination of utility relations. $\mathrm{U{\scriptsize T}}_V$ expresses preferences or hard requirements. No normalization is imposed here, because local utilities can always be combined without generating any impossibility; their combination can only generate *unacceptability*.

Definition 1 *A problem Pb on an uncertainty-utility structure $(S_{un}, S_{ut}, \otimes_{un/ut})$ is a tuple (V, G, UN, F, UT) where:*

- *$V = \{x_1, x_2, \ldots\}$ is a finite set of* variables*. V is partitioned into V_E (the set of environment variables) and V_D (the set of decision variables). Each variable x_i has a finite* domain *$Dom(x_i)$. For $S \subset V$, $Dom(S)$ denotes the Cartesian product of the domains of variables in S.*
- *G is a DAG whose vertices are called* components*. The components form a partition of V such that each component intersects only one of V_E or V_D. We note $\mathcal{C}_E$ (resp. $\mathcal{C}_D$) the set of components included in V_E (resp. V_D) and $pa(c)$ the variables belonging to a parent of c in G.*
- *$UN = \{UN_1, UN_2, \ldots\}$ is a finite set of* uncertainty relations*; each UN_i is associated with a component $c \in \mathcal{C}_E$ noted $c(UN_i)$, s.t. $S(UN_i) \subset (c \cup pa(c))$; for any $c \in \mathcal{C}_E$, the local normalization $\bigoplus_{un\, c} \left(\otimes_{un\, c(UN_i)=c} UN_i\right) = 1_{un}$ must hold.*

– $F = \{F_1, F_2, \ldots\}$ *is a finite set of* feasibility relations*; each* F_i *is associated with a component* $c \in \mathcal{C}_D$ *noted* $c(F_i)$*, s.t.* $S(F_i) \subset (c \cup pa(c))$*; for any* $c \in \mathcal{C}_D$*, the local normalization* $\bigvee_c \left(\wedge_{c(F_i)=c} F_i\right) = true$ *must hold.*
– $UT = \{UT_1, UT_2, \ldots\}$ *is a finite set of* utility relations*.*

The DAG structure makes uncertainty and feasibility relations implicitly oriented (between variables in $pa(c)$ and variables in c) or not (inside a component). Note that there cannot be any undirected relation between an environment variable and a decision variable, as they cannot belong to a same component: either a decision influences the environment, or the environment restrains the possible decisions. Intuitively, the DAG models independences.

The dinner problem can be modeled using six variables: bp_J and bp_M (value t or f), representing John's and Mary's presence at the beginning, ep_J and ep_M (value t or f), representing their presence at the end, mc (value $fish$ or $meat$), for the main course choice, and w (value $white$ or red), for the wine choice. $V_D = \{mc, w\}$ and $V_E = \{bp_J, bp_M, ep_J, ep_M\}$.

The DAG of components encoding independences can be built using e.g. causal reasoning. For example, it is possible to infer that bp_J and bp_M are linked by a correlation relation and are not causally influenced by other variables, that ep_J is causally determined by bp_J and w, but not by ep_M (the corresponding links are oriented to ep_J), that mc and w are linked by a non-oriented feasibility relation expressing the impossibility to order fish with red wine... The result is shown in Figure 1a. Implicitly, the DAG means that the joint probability distribution $P(bp_J, bp_M, ep_J, ep_M \mid mc, w)$ can be expressed as $P(bp_J, bp_M) \otimes_{un} P(ep_J \mid bp_J, bp_M, mc, w) \otimes_{un} P(ep_M \mid bp_J, bp_M, mc, w)$, and that the global feasibility degree can be expressed as $F(mc, w)$.

The uncertainty relations express $P(bp_J, bp_M)$, $P(ep_J \mid bp_J, bp_M, mc, w)$, and $P(ep_M \mid bp_J, bp_M, mc, w)$. A first uncertainty relation UN_1 expresses $P(bp_J, bp_M)$, the probability of presence of John and Mary at the beginning: $UN_1(bp_J = t, bp_M = f) = 0.6$, $UN_1(bp_J = f, bp_M = t) = 0.4$, and $UN_1(bp_J = t, bp_M = t) = UN_1(bp_J = f, bp_M = f) = 0$. Then, $P(ep_J \mid bp_J, bp_M, mc, w)$ can be specified as $UN_2 \otimes_{un} UN_3$. UN_2 expresses that if John is absent at the beginning, he will be absent at the end: $UN_2(A){=}0$ if $A{=}(ep_J{=}t, bp_J{=}f)$, 1 otherwise. Equivalently UN_2 is the constraint $(bp_J{=}\ f) \to (ep_J{=}\ f)$. UN_3 is the constraint $(bp_J{=}\ t) \to ((ep_J{=}\ t) \leftrightarrow \neg(w{=}\ white))$. Similarly, $P(ep_M \mid bp_J, bp_M, mc, w) = UN_4 \otimes_{un} UN_5$ with UN_4 and UN_5 defined as constraints. Note that unlike Bayesian networks, one can specify the probability of a variable given its parents by several relations.

Concerning feasibilities, $F(mc, w)$ is specified as a unique feasibility relation F_1, expressing that Peter cannot order fish with red wine: $F_1 : \neg((mc = fish) \wedge (w = red))$. The association of relations with components is shown in Figure 1a.

As for utilities, a binary utility relation expresses that Peter does not want John to leave the dinner: $UT_1 : (bp_J = t) \to (ep_J = t)$. Two unary utility relations UT_2 and UT_3 on ep_J and ep_M resp. express the gains expected from the presences at the end: for instance, $UT_2(ep_J = t) = 10$ and $UT_2(ep_J = f) = 0$.

All the local relations are drawn as a composite graphical model in Figure 1b.

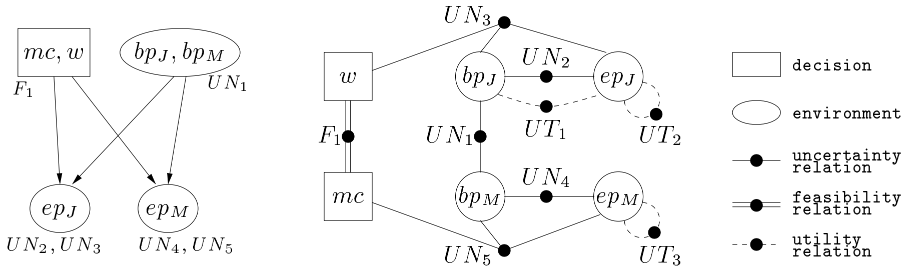

Fig. 1. (a) DAG of components (b) Uncertainty-Feasibility-Utility network.

Back to existing frameworks Let us consider the formalisms described in Section 2 again. Representing a CSP (hard or soft) in our framework can be easily done by defining a problem as $Pb = (V, G, \emptyset, \emptyset, UT)$: all variables in V are decision variables, G is reduced to a unique decision component containing all variables, and the constraints can be considered as utility relations.

A Bayesian network can be modeled as $Pb = (V, G, UN, \emptyset, \emptyset)$: all variables in V are environment variables (they are random variables), G is the DAG of the Bayesian network, and $UN = \{P(x_i \,|\, pa(x_i)), x_i \in V\}$. There are not any feasibility or utility relations. Chain graphs differ from Bayesian networks in that instead of expressing conditional probabilities $P(x_i \,|\, pa(x_i))$ on variables, they express conditional probability distributions $P(c_i \,|\, pa(c_i))$ on components c_i of a DAG, each $P(c_i \,|\, pa(c_i))$ being expressed in a factored form. A chain graph can be modeled as $Pb = (V, G, UN, \emptyset, \emptyset)$, with G the DAG of components of the chain graph, and UN the set of factors of each $P(c_i \,|\, pa(c_i))$. Without the notion of components, such a framework would not have been subsumed.

One can model a stochastic CSP as $Pb = (V, G, UN, \emptyset, UT)$, where V_E is the set of stochastic variables, V_D is the set of decision ones, G is a DAG which depends on the relations between the stochastic variables, UN is the set of probability distributions on the stochastic variables, and UT is the set of constraints.

A finite horizon probabilistic MDP can be modeled as $Pb = (V, G, UN, F, UT)$. If there are T time-steps, then $V_E = \{s_t, t \in [1, T]\}$ and $V_D = \{d_t, t \in [1, T]\}$; G is a DAG of components such that each component contains only one variable, such that the parents of an environment component $\{s_{t+1}\}$ are $\{s_t\}$ and $\{d_t\}$, and such that the unique parent of a decision component $\{d_t\}$ is $\{s_t\}$; $UN = \{P(s_{t+1}|s_t, d_t), t \in [1, T-1]\}$, $F = \{f(s_t, d_t), t \in [1, T]\}$, and $UT = \{R(s_t, d_t), t \in [1, T]\}$. The modeling of a possibilistic MDP is similar.

It is easy to prove that SAT, stochastic SAT, quantified boolean formulas, quantified CSP, factored MDP, influence diagrams or STRIPS planning are also subsumed by this formalism. Yet, frameworks such as *partially ordered* semiring CSP [2] are not encompassed, because of the assumption of a total order on E_{ut}. Note that our example could not have been modeled with influence diagrams, which do not include the notion of feasibility.

Conditional independence This definition offers additional semantic properties if the notion of conditional uncertainty distributions is introduced. An uncertainty distribution γ_S is an uncertainty relation over S satisfying the normalization condition $\oplus_{un\,S}\,\gamma_S = 1_{un}$. It can be extended to any subset S' of S by applying $\oplus_{un}$ to eliminate variables in S–S'. As conditional probabilities are defined using division, conditional uncertainties are defined via a conditioning operator over E_{un} denoted $\oslash_{un}$. When such an operator defined on $\{(un_1, un_2) \in E_{un}\times E_{un} \mid un_1 \preceq_{un} un_2, 0_{un} \prec_{un} un_2\}$ and verifying $(un_1 \oslash_{un} un) \oplus_{un} (un_2 \oslash_{un} un) = (un_1 \oplus_{un} un_2) \oslash_{un} un$ (linearity), $0_{un} \oslash_{un} un = 0_{un}$, $un \oslash_{un} un = 1_{un}$, $(un_1 \oslash_{un} un_2) \otimes_{un} un_2 = un_1$ and $(un_1 \oslash_{un} un) \otimes_{un} ((un_2 \otimes_{un} un) \oslash_{un} un) = (un_1 \otimes_{un} un_2) \oslash_{un} un$ (simplification) exists, the uncertainty structure is said to be *conditionable*. All structures of Table 1 are conditionable.

In a conditionable uncertainty structure, let γ_S be an uncertainty distribution on S, and S', S'' be disjoint subsets of S. $\gamma_{S' \cup S''} \oslash_{un} \gamma_{S''}$, noted $\gamma_{S' \mid S''}$, is a conditional uncertainty distribution on S' given S''. It is an uncertainty distribution on S' for any assignment of S'', and verifies $\gamma_{S' \cup S''} = \gamma_{S' \mid S''} \otimes \gamma_{S''}$. If S_1, S_2 and S_3 are disjoint subsets of S, then, S_1 and S_2 are conditionally independent w.r.t. γ given S_3 iff $\gamma_{S_1 \cup S_2 \mid S_3} = \gamma_{S_1 \mid S_3} \otimes_{un} \gamma_{S_2 \mid S_3}$: the problem can be split into one part depending on $S_1 \cup S_3$, and another one depending on $S_2 \cup S_3$.

If a DAG of components is used to model conditional independences of the global uncertainty degree $\mathrm{U}\textsc{n}_V$, then $\mathrm{U}\textsc{n}_V$ can be expressed as a combination of normalized factors (one factor per component), and each factor may be further factorized. The same kind of factorization can be obtained for F_V. Conversely, it is possible to prove that the global uncertainty degree defined from a problem $Pb = (V, G, UN, F, UT)$ as the combination of the uncertainty relations in UN is an uncertainty distribution conditional independences of which are encoded by G. The same holds for feasibilities. See [22] for more details.

5 Definition of queries

Given a problem, the goal is now to answer queries on it. As shown in Section 2 (see Equations 1 to 6), a query corresponds to a sequence of variable eliminations applied on a combination of local functions. As the combination of local functions is defined by a problem, the only element not yet defined is the sequence of eliminations; it corresponds to our definition of queries. A query will enable us to ask questions such as: "*how to maximize the investment if the restaurant chooses the main course first and Peter is pessimistic about this choice, and then Peter chooses the wine before knowing who is present at the beginning and at the end*". In this case, the sequence of eliminations would be $(\min, \{mc\}).(\max, \{w\})$ $.(\oplus_{ut}, \{bp_J, bp_M, ep_J, ep_M\}))$. Formally:

Definition 2 *A query Q is a pair (Pb, Sov) where $Pb = (V, G, UN, F, UT)$ is a* problem *and Sov is a sequence of* operator-variables *pairs, such that the operators are* min*,* max *or $\oplus_{ut}$, and such that each variable appears at most once in Sov. Variables that appear in Sov are* quantified variables*, the others are called* free variables.

A query is correct *iff, if quantified, environment variables are quantified by $\oplus_{ut}$ and decision ones by* min *or* max *(adequation between the nature of a vari-*

able and its operator[4]*),and iff for any variables x and y of different nature (one decision variable, one environment variable), when the component of x is an ascendant in the DAG G of the component of y, then x appears at the left of y in Sov, or x is a free variable (respect of causality).*

Thus, a query is correct if it satisfies constraints on the elimination order and on the type of eliminations. Note that adjacent sets of variables S_1 and S_2 eliminated with the same operator *op* can be gathered with $(op, S_1).(op, S_2) = (op, S_1 \cup S_2)$, because elimination operators are associative and commutative.

Property 1 *There exists at least one correct query without free variables on a problem (because of the DAG structure).*

Query meaning The answer $Ans(Q)$ to a correct query $Q = (Pb, Sov)$ can be defined inductively as a function of A, an assignment of the free variables:

$$Ans((Pb, (op, \{x_i\}) \ . \ Sov'))(A') = \underset{a \in Dom(x_i)}{op} Ans((Pb, Sov'))(A'.(x_i = a)) \quad (7)$$

$$Ans((Pb, \emptyset))(A') = \left(\left(\bigwedge_{F_i \in F} F_i \right) \star \left(\underset{UN_i \in UN}{\otimes_{un}} UN_i \right) \otimes_{un/ut} \left(\underset{UT_i \in UT}{\otimes_{ut}} UT_i \right) \right)(A') \quad (8)$$

Equation 8 expresses that, if all the problem variables are assigned, the answer to the query is the combination of the uncertainty degree, the feasibility degree, and the utility degree of the corresponding complete assignment. Equation 7 expresses that, if the variables are not all assigned and x_i is the first quantified variable with *op* as an operator, the answer to the query is obtained by applying the elimination operator *op* to all the values of x_i. When min/max operators are used on a decision variable, this means optimal decisions are sought. All or part of the values that optimize the query can be recorded if needed during the evaluation of the answer to a query. Equivalently, $Ans(Q)$ can be written: $Ans(Q) = Sov \left((\wedge_{F_i \in F} F_i) \star (\otimes_{un UN_i \in UN} UN_i) \otimes_{un/ut} (\otimes_{ut UT_i \in UT} UT_i) \right)$.

Queries enable us to consider various situations in terms of *observability*. If an environment quantified variable x appears at the left of a decision quantified variable y (e.g. $Sov = \ldots (\oplus_{ut}, \{x\}) \ldots (\max, \{y\}) \ldots$), this means that the value of x is known (observed) when a value for y is chosen. Conversely, if $Sov = \ldots (\max, \{y\}) \ldots (\oplus_{ut}, \{x\}) \ldots$, x is not observed before choosing y.

In another direction, queries enable us to consider various situations in terms of *optimistic or pessimistic attitude*, as each decision variable can be quantified with min or max. Assume that a decision is made by another agent via a decision variable y. It is possible to perform either $(\max, \{y\})$ if one is optimistic about the behavior of the other agent, or $(\min, \{y\})$ if one is pessimistic.

Semantical foundations based on the lottery theory [21] do exist for the definition of the query meaning: using conditional distributions, it is possible to give a second definition of $Ans(Q)$, where each step involving a environment variable is considered as a lottery, and each step involving a decision variable is considered as an optimization step among the feasible decisions.

---
[4] An environment variable can be quantified by min or max, if $\oplus_{ut}$=min or max.

Theorem 1 *The two definitions of $Ans(Q)$ are equivalent if conditional uncertainties can be defined (see [22] for a proof).*

With the second (semantic) definition, computationally expensive quantities are involved: the first (operational) definition is algorithmically more suitable.

Queries on the dinner problem What is the maximum investment Peter can expect (and which associated decision(s) should he make) if he chooses the menu without knowing who will come ? The associated query is:

$$(Pb, (\max, \{mc, w\}).(\oplus_{ut}, \{bp_J, bp_M, ep_J, ep_M\}))$$

The answer is 6 (k\$) with $\{(mc = meat), (w = red)\}$. But if Peter knows who comes, the query becomes:

$$(Pb, (\oplus_{ut}, \{bp_J, bp_M\}).(max, \{mc, w\}).(\oplus_{ut}, \{ep_J, ep_M\}))$$

The answer is 26 (k\$) with a 20 (k\$) gain from the observability of who is present. The decision is $\{(mc = meat), (w = red)\}$ if John is present and Mary is not, $\{(mc = fish), (w = white)\}$ otherwise. Consider now the query introduced at the beginning of Section 5:

$$(Pb, (\min, \{mc\}).(\max, \{w\}).(\oplus_{ut}, \{bp_J, bp_M, ep_J, ep_M\}))$$

The answer is $\bot_{ut} = -\infty$: in the worst main course case, even if Peter chooses the wine, the situation can be unacceptable. Finally, the query

$$(Pb, (\oplus_{ut}, \{bp_J, bp_M, ep_J, ep_M\}).(\max, \{mc, w\}))$$

is not correct: it runs counter to causality, as the menu has to be chosen before knowing who is present at the end. These examples show how a query enables us to consider various situations in terms of observability, and in terms of optimistic or pessimistic attitude.

Back to existing frameworks Let us consider again the examples of Section 2. Looking for a solution to a CSP (Equation 1) or to a totally ordered soft CSP corresponds to the query $Q = (Pb, (\max, V))$, with Pb the expression of the CSP in our framework and V the set of variables of the CSP. Computing the probability distribution on a variable x_1 for a Bayesian network (Equation 2) modeled as Pb corresponds to $Q = (Pb, (+, \{x_2, \ldots, x_n\})$. These examples are *mono-operator queries*, involving only one type of elimination operator.

Let us now consider *multi-operator queries*. The search for an optimal policy under uncertainty for a stochastic CSP (Equation 3) modeled as Pb, corresponds to the query $Q = (Pb, (\max, \{d_1, d_2\}).(+, \{s_1\}).(\max, \{d_3, d_4\}).(+, \{s_2\}))$. For a finite horizon MDP with T time-steps (Equations 4 to 6), the query is $Q = (Pb, (\max, \{d_1\}).(\oplus_{ut}, \{s_2\}).(\max, \{d_2\}) \ldots (\oplus_{ut}, \{s_T\}).(\max, \{d_T\}))$, where $\oplus_{ut}$ equals $+$ with probabilistic MDP, and min with possibilistic pessimistic MDP. With a quantified CSP, elimination operators min and max alternate. With influence diagrams, the unique query (production of a decision that maximizes expected utility) alternates max on decisions and $\oplus_{ut} = +$ on random variables.

Answering queries For mono-operator queries on simple graphical models, generic algorithms as provided by valuation algebras [16, 17] or bucket elimination [18] can be considered. For multi-operator queries, we have:

Theorem 2 *Computing the answer to a query is a PSPACE-hard problem.*

Proof. Quantified Boolean Formulas (QBF) is a PSPACE-complete problem which can be easily reduced to our framework: all variables are decisions, clauses are utility relations combined with $\wedge$*, and the query alternates* min *(for universally quantified variables) and* max *(for existentially quantified variables).* □

The first inductive definition of the meaning of a query Q actually defines a naive exponential time algorithm to compute $Ans(Q)$ using a tree-exploration procedure with a fixed variable ordering (the one of Sov) that collects elementary uncertainties, feasibilities, and utilities. According to the nature of the operator used, each level in the tree consists in applying a min, max or $\oplus_{ut}$ operator on the values collected. This blunt approach, as existing approaches for quantified constraints or formulas [4], uses Sov to fix the variable order. A better approach would be to transform a multi-operator query into an optimized set of mono-operator queries bringing to light parallelisms and extra variable order freedom.

6 Conclusion

In the last decades, AI has witnessed the design and study of numerous frameworks for reasoning about uncertainties, feasibilities and utilities. We have tried to crystallize their inherent mathematical structure to build a unified formalism covering hard, valued, quantified, mixed, and stochastic CSP, Bayesian networks, probabilistic or possibilistic MDP, influence diagrams... Compared to related works [16, 18, 17], our proposal is the only one able to deal with generic composite graphical models, in which there may be several types of variables (decision or environment), several types of local relations (uncertainties, feasibilities, utilities), and several types of combination and elimination operators.

From an algorithmic point of view, approximate algorithms using sampling and local search could be considered. Anyway, the implicit tree-search algorithm embodied in the first operational inductive definition of the value of a query offers a first naive approach. The simultaneous use of local consistencies [19, 23], global constraints, branch and bound (and possibly game algorithms), variable/bucket elimination algorithms [18], as well as caching strategies that exploit the problem-structure as in [24, 25], are obvious candidates to improve this basic schema, provided that they are extended to take into account the composite nature of the Uncertainty-Feasibility-Utility networks framework. In another direction, so as to more precisely justify the algebraic structure used in the framework, works in decision theory such as the one in [26] could also be considered.

Acknowledgement. The work described in this paper was partially conducted within the EU Integrated Project COGNIRON ("The Cognitive Companion") and funded by the European Commission Division FP6-IST Future and Emerging Technologies under Contract FP6-002020.

References

1. Mackworth, A.: Consistency in Networks of Relations. Artificial Intelligence **8** (1977) 99–118
2. Bistarelli, S., Montanari, U., Rossi, F., Schiex, T., Verfaillie, G., Fargier, H.: Semiring-Based CSPs and Valued CSPs: Frameworks, Properties and Comparison Constraints **4** (1999) 199–240
3. Fargier, H., Lang, J., Schiex, T.: Mixed Constraint Satisfaction: a Framework for Decision Problems under Incomplete Knowledge. In Proc. of AAAI-96
4. Bordeaux, L., Monfroy, E.: Beyond NP: Arc-consistency for Quantified Constraints. In Proc. of CP-02
5. Walsh, T.: Stochastic Constraint Programming. In Proc.of ECAI-02
6. Littman, M., Majercik, S., Pitassi, T.: Stochastic Boolean Satisfiability. Journal of Automated Reasoning **27** (2001) 251–296
7. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann (1988)
8. Frydenberg, M.: The Chain Graph Markov Property. Scandinavian Journal of Statistics **17** (1990) 333–353
9. Howard, R., Matheson, J.: Influence Diagrams. In Readings on the Principles and Applications of Decision Analysis. 1984
10. Puterman, M.: Markov Decision Processes, Discrete Stochastic Dynamic Programming. John Wiley & Sons (1994)
11. Monahan, G.: A Survey of Partially Observable Markov Decision Processes: Theory, Models, and Algorithms. Management Science **28** (1982) 1–16
12. Boutilier, C., Dearden, R., Goldszmidt, M.: Stochastic Dynamic Programming with Factored Representations. Artificial Intelligence **121** (2000) 49–107
13. Dubois, D., Prade, H.: Possibility theory as a basis for qualitative decision theory. In Proc. of IJCAI-95
14. Sabbadin, R.: A Possibilistic Model for Qualitative Sequential Decision Problems under Uncertainty in Partially Observable Environments. In Proc. of UAI-99
15. Giang, P., Shenoy, P.: A qualitative linear utility theory for Spohn's theory of epistemic beliefs. In: Proc. of UAI-00
16. P. Shenoy. Valuation-based Systems for Discrete Optimization. In Proc. of UAI-90.
17. Kolhas, J.: Information Algebras: Generic Structures for Inference. Springer (2003)
18. Dechter, R.: Bucket Elimination: a Unifying Framework for Reasoning. Artificial Intelligence **113** (1999) 41–85
19. Cooper, M and Schiex, T.: Arc consistency for soft constraints. Artificial Intelligence **154** (2004) 199–227
20. G. Shafer. A mathematical theory of evidence. Princeton University Press (1976)
21. von Neumann, J., Morgenstern, O.: Theory of Games and Economic Behaviour. Princeton University Press (1944)
22. C. Pralet, G. Verfaillie and T. Schiex. Belief-feasibility-desire networks. http://www.laas.fr/∼cpralet/techreport.ps Technical report LAAS-CNRS 05129, 2005.
23. J. Larrosa and T. Schiex. In the quest of the best form of local consistency for weighted CSP. In Proc. of IJCAI-03.
24. A. Darwiche. Recursive conditioning. Artificial Intelligence **126** (2001) 5–41
25. P. Jégou and C. Terrioux. Hybrid Backtracking bounded by Tree-decomposition of Constraint Networks. Artificial Intelligence **146** (2003) 43–75
26. F.C. Chu and J.Y. Halpern. Great Expectations. Part I: On the Customizability of Generalized Expected Utility. In Proc. of IJCAI-03.