Division of Informatics, University of Edinburgh

Centre for Intelligent Systems and their Applications

Duality in Knowledge Sharing

by

Marco Schorlemmer

Informatics Research Report EDI-INF-RR-0134

Division of Informatics **June 2002**
http://www.informatics.ed.ac.uk/

Duality in Knowledge Sharing

Marco Schorlemmer

Informatics Research Report EDI-INF-RR-0134

DIVISION *of* INFORMATICS
Centre for Intelligent Systems and their Applications

June 2002

Abstract :

I propose a formalisation of knowledge sharing scenarios that aims at capturing the crucial role played by an existing duality between ontological theories one wants to merge and particular situations that need to be linked. I use diagrams in the Chu category and colimits over these diagrams to account for the reliability and optimality of knowledge sharing systems. Furthermore, I show how we may obtain a deeper understanding of a system that shares knowledge between a probabilistic logic program and Bayesian belief networks by re-analysing the scenario in terms of the present approach.

Keywords : Knowledge sharing, ontologies, channel theory, Chu spaces, probabilistic logic programming, Bayesian belief networks

Duality in Knowledge Sharing

W. Marco Schorlemmer

Centre for Intelligent Systems and their Applications
Division of Informatics
The University of Edinburgh

Abstract

I propose a formalisation of knowledge sharing scenarios that aims at capturing the crucial role played by an existing duality between ontological theories one wants to merge and particular situations that need to be linked. I use diagrams in the Chu category and colimits over these diagrams to account for the reliability and optimality of knowledge sharing systems. Furthermore, I show how we may obtain a deeper understanding of a system that shares knowledge between a probabilistic logic program and Bayesian belief networks by re-analysing the scenario in terms of the present approach.

Keywords: Knowledge sharing, ontologies, channel theory, Chu spaces, probabilistic logic programming, Bayesian belief networks.

1 Introduction

The task of building knowledge-intensive systems has changed with the arrival and generalised use of the Internet. It has shifted from the task of producing stand-alone knowledge-based systems, designed by a reduced team of closely cooperating knowledge engineers, to the task of integrating highly distributed systems that reason with knowledge compiled by separate knowledge engineers within different sources of knowledge. Nowadays knowledge components, such as ontologies, problem solvers, or knowledge bases, are publicly available on the World-Wide Web, and consequently, researchers have addressed over the last years the problems that arise with the attempt of successfully sharing and reusing knowledge in such distributed environments.

Reliable sharing of formally represented knowledge is, in general, a very difficult task. Different knowledge components may have been specified in different logical languages using completely distinct terminology, although having been designed to be used in the same application domain; and most likely they may have been implemented following different computational paradigms. Ontologies have been advocated as a solution for knowledge sharing, and a lot of effort has been put in devising methodologies for the design, development, and deployment of ontologies [16, 11, 12, 24, 5]. Very briefly, an ontology is an explicit specification of a conceptualisation, and usually consists of a hierarchy of concepts, relations between these concepts, and a set of axioms that constrain the possible interpretations of concepts and relations.

Although ontologies have been proposed as a silver bullet for knowledge sharing, it has been shown that, for some knowledge-sharing scenarios, the integration of ontologies by aligning, merging, or unifying concepts and relations as specified by their respective theories turns out to be insufficient [7]. A closer analysis of these scenarios reveals that successful and reliable knowledge sharing between two systems goes closely together with an agreed understanding of an existing duality between the merge of local ontologies into a global one, and the identification of potential situations in which the sharing of knowledge is going to take place.

Two issues arise from the above analysis. First of all, that the approach to knowledge sharing presented here is heavily influenced by Barwise and Seligman's perspective of information flow and by the logic of distributed systems they propose [3]. They advocate in the second of their 'Principles of Information Flow':

> Information flow crucially involves both, types and their particulars. [3]

Types constitute the 'vocabulary' used to describe a local component, and the particulars, also called tokens, are the instances of these components, or the situations in which they are going to be used.

The second issue is that duality is a recurrent theme in logic and mathematics, and this has been thoroughly studied within category theory by means of Chu spaces [1, 13, 2, 19]. Actually, Chu spaces also form the foundations of a mathematical theory of formal concept formation [9], as well as of the above mentioned theory of information flow [3].

In his 'Categorical Manifesto', Goguen gave seven guidelines, which he called *dogmas*[1], for using category theory in computing science [10]. Fields

[1]Goguen called them dogmas as a reminder that they should not be taken too dogmat-

like formal software engineering have gained a deep insight into specific tools and techniques thanks to use of category-theoretical concepts [21, 4]. It is therefore sensible to assume that category theory can also help us to provide a deeper insight into the complex tasks that concern knowledge engineers. In particular, Goguen's *sixth dogma* is absolutely relevant to the challenge of knowledge sharing:

> Given a species of structure, say widgets, then the result of interconnecting a system of widgets to form a super-widget corresponds to taking the *colimit* of the diagram of widgets in which morphisms show how they are interconnected. [10]

In this paper I shall introduce a formalisation of knowledge-sharing scenarios where the category of 'widgets' is going to be the category of Chu spaces and Chu transforms. As already mentioned, this is influenced by Barwise and Seligman's perspective of information flow. I am not introducing new scenarios here; my claim is that the formalisation proposed in this paper favours the rigorous modelling and analysis of reliable and optimal knowledge-sharing scenarios. I discuss the role played by duality in such scenarios in Section 2, and introduce the formal framework in Section 3. In that section, as well as in Section 4, I illustrate some of the advantages of the suggested formalisation by re-analysing a knowledge-sharing scenario first studied by Corrêa da Silva et al. in [6]. I conclude in Section 5 discussing related work and future perspectives.

2 The Role of Duality

2.1 Aligning Ontologies

In order to explain what I mean under *duality* and in order to illustrate its role in knowledge sharing, let us start with an example taken from [22], which shows the issues one has to take into account when attempting to align the English concepts river and stream with the French concepts fleuve and rivière. According to Sowa,

> In English, size is the feature that distinguishes river from stream; in French, a fleuve is a river that flows into the sea, and a rivière is either a river or a stream that runs into another river. [22]

This explains how the concepts need to be merged. Notice that the above quote requires an agreed understanding on how to distinguish between big

ically.

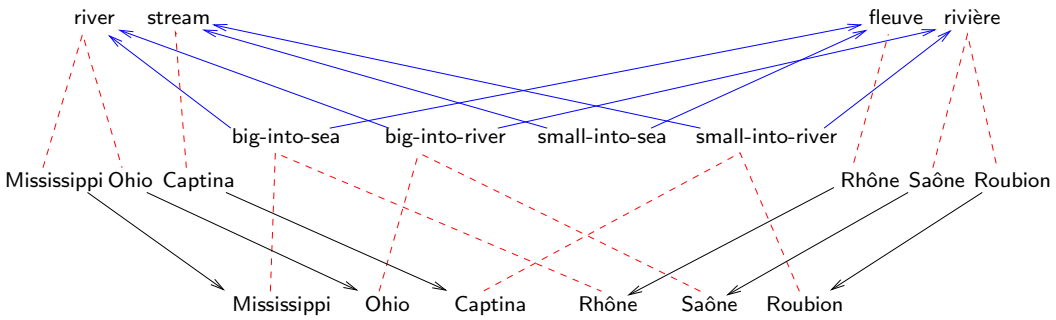

Figure 1: Agreed understanding

and small rivers, and between rivers that run into a sea or into other rivers, yielding four types of instances of 'water-flowing entities': big-into-sea, big-into-river, small-into-sea, and small-into-river.

Figure 1 shows how both, English and French speakers, base their concepts upon this agreed understanding, although English and French speakers don't distinguish between some types of instances. For example, English speakers call both, big-into-sea and big-into-river, a river, while French speakers don't distinguish between big-into-river and small-into-river, and call both types a rivière.

But aligning the ontologies requires the classification of particular instances of river, stream, fleuve, and rivière according with the agreed understanding, since it is this agreed way of classification which will determine how the concepts river, stream, fleuve, and rivière are going to be related to each other. The ultimate goal is to determine the connections that link particular instances of type river or stream with particular instances of type fleuve or rivière, in a way that they respect the agreed understanding. This is done by connecting only those instances that conform to the same type according to the agreed understanding, as illustrated in Figure 2.

The resulting classification of connections $\langle$Mississippi, Rhône$\rangle$, $\langle$Ohio, Saône$\rangle$, and $\langle$Captina, Roubion$\rangle$ into the four concepts river, stream, fleuve, and rivière, determines a theory of how these concepts are related (e.g., that a fleuve is also a river, or that a stream is also a rivière, but not vice versa). Figure 2 shows what in channel theory is known as a an *information channel* [3]. It captures, by means of two pairs of contravariant functions, an existing duality between concepts and instances: Each pair consists of a map of concepts on the so called *type level* and map of instances on the so called *token level*, and pointing in the opposite direction. From a channel-theoretic perspective, Figure 2 actually illustrates us that sharing knowledge involves a flow

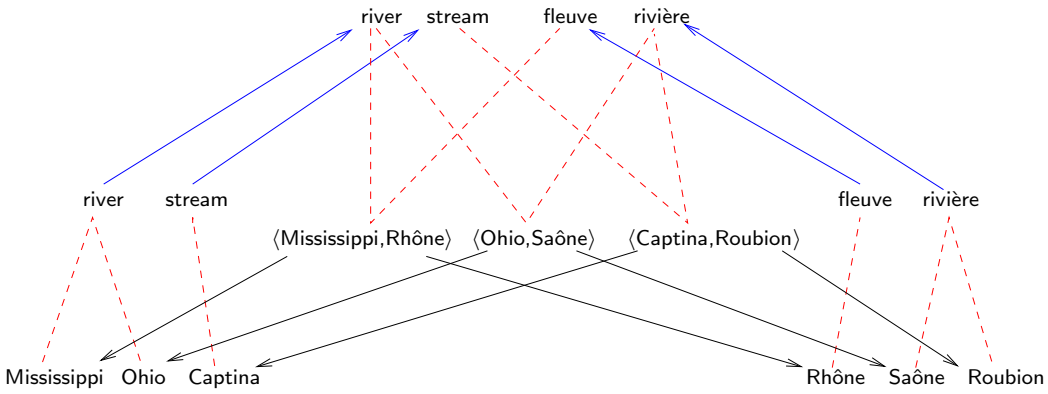

Figure 2: Aligning ontologies

of information that crucially depends on how the instances of different agents are connected together. The following table shows the *classification relation*, i.e., the connections as classified according to the concept types involved in the example:

	river	stream	fleuve	rivière
⟨Mississippi, Rhône⟩	1	0	1	0
⟨Ohio, Saône⟩	1	0	0	1
⟨Captina, Roubion⟩	0	1	0	1

The merged set of concepts {river, stream, fleuve, rivière} actually has an additional structure that we can deduce from the way the connections of instances are classified with respect to these concepts. Through techniques from formal concepts analysis [9], for instance, we can make such structure explicit in the form of a *concept lattice*, as shown in Figure 3. The concept hierarchy represented in this lattice depends on the choice of instances and its classification with respect to the agreed understanding. The fact that no instances were classified as of type small-into-sea was crucial in this example. Notice, also, that the resulting lattice has a node labelled with the concept river $\wedge$ rivière, which is a formal concept that did not exist in the original vocabularies. It corresponds to the instances of 'water-flowing entities' that, although big, flow into other rivers, like Ohio and Saône.

2.2 Sharing Inferential Knowledge

The principle of aligning ontologies outlined in Section 2.1, which takes the duality arising between sets of concepts to be merged and instances to be

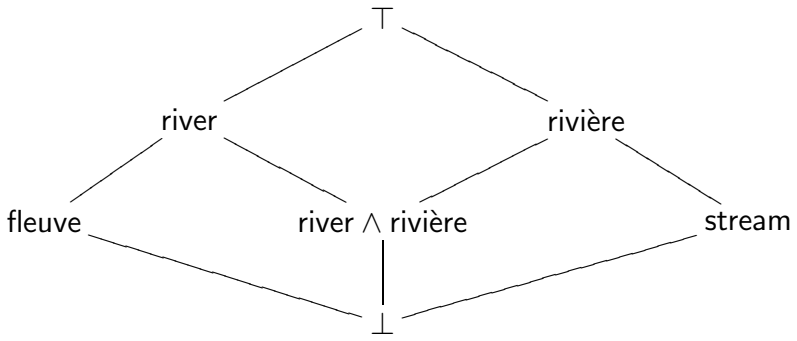

Figure 3: Concept lattice

linked into account, is a case of a general pattern that reveals to be crucial for the reliable sharing of knowledge. Corrêa da Silva et al. have studied several knowledge-sharing scenarios where a common ontological theory turned out to be insufficient for the reliable sharing of knowledge [7]. They showed several problems arising from this insufficiency and proposed alternative solutions to them.

For instance, when attempting to share knowledge between two knowledge bases with inference engines based on different substructural logics [20], like relevant logic (rl) and linear logic (ll), a common ontological theory for the declarative knowledge is insufficient: A system based on linear logic querying a system based on relevant logic needs to know if the foreign system has used the following inference rule, called *contraction*, when answering the query, because the former will not accept a proof involving this rule:

$$\frac{\Gamma[(X;Y);Y] \vdash A}{\Gamma[X;Y] \vdash A}$$

In the above rule, A stands for a formula, X and Y are structures (compositions of formulas with the operator ';'), and $\Gamma[-]$ denotes the context in which these structures occur.

The scenario shown in Figure 4 is similar to the example of Section 2.1. Here, instead of aligning concepts in order to know when a instance of river corresponds to an instance of fleuve or rivière, we are interested in aligning sequents of relevant logic and linear logic in order to know for which a proof P of a sequent $\Pi \vdash_{rl} B$ in relevant logic amounts to a proof R of the same sequent $\Pi \vdash_{ll} B$ in linear logic.

The fact that both knowledge-based systems agree with respect to the

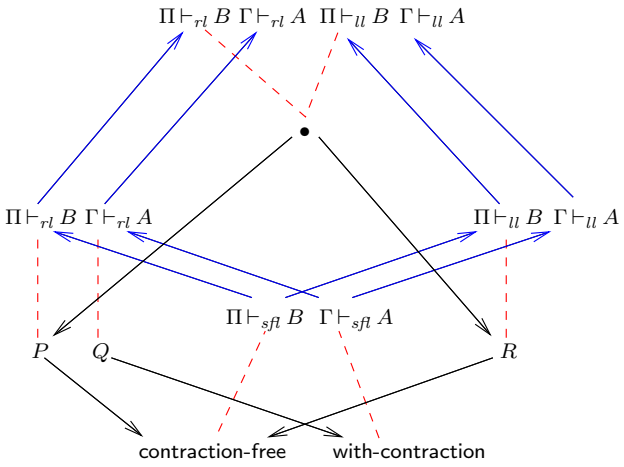

Figure 4: Sharing inferential knowledge

logical language in which they express their knowledge and formulate their queries, is given by the fact that the type level of the pair of contravariant functions shown in Figure 4 are identities. But, any attempt to successfully share knowledge between both knowledge-based systems has to identify those tokens of proofs that both systems can actually share, which in this particular example means *contraction-free proofs*.

Corrêa da Silva et al. [7] propose to use structurally-free logic (*sfl*) [8] as a mechanism for distinguishing between contraction-free proofs from proofs with contraction, and hence for inferring the necessary link at the token level. Although they did not express it in channel-theoretic terms, what they actually did amounts to use structurally-free logic to define a classification that captures the agreed understanding that ultimately determines an information channel between relevant logic and linear logic.

A proof of a sequent $\Gamma \vdash A$ is mapped to the sequence of combinators[2], which keeps track of the application of structural inference rules (e.g., left-associativity, commutativity, contraction). For instance, the proof Q of sequent

$$A \to (A \to B) \vdash A \to B$$

shown in Figure 5 is mapped to the (one-element) sequence $\langle \mathbf{W} \rangle$, because a structural rule has been used only once (see [7] for further details). The fact that the combinator **W** happens to be in this sequence indicates that the proof uses contraction, and, consequently, the token level of the contravariant

[2]A combinator is a λ-term with no free occurrences of variables

$$\dfrac{\dfrac{\dfrac{\dfrac{A \to (A \to B) \vdash A \to (A \to B) \qquad A \vdash A}{A \to (A \to B); A \vdash A \to B}(\to E) \qquad A \vdash A}{(A \to (A \to B); A); A \vdash B}(\to E)}{A \to (A \to B); A \vdash B}(\mathbf{W})}{A \to (A \to B) \vdash A \to B}(\to I)$$

Figure 5: Proof Q of $A \to (A \to B) \vdash A \to B$

pair of functions should map the proof Q to the token with-contraction (see Figure 4).

3 Patterns of Knowledge Sharing

The solution sketched above actually illustrates Corrêa da Silva et al.'s awareness of the implicit type/token duality that one needs to take into account for a reliable sharing of knowledge. I therefore make this duality explicit and propose a mathematical model that attempts to capture this recurrent pattern in different knowledge-sharing scenarios. For this reason I need to move to a more general model of information flow than the one covered by Barwise and Seligman [3]. Classification relations in channel theory are crisp relations, with two-valued characteristic functions: Either a token a is classified as of type α or not (see Section 2.1). But knowledge-sharing scenarios as the one analysed in [6], and re-analysed here in Section 3.2, require the characteristic function of a classification relation to be many-valued. I therefore take refuge in the more general concept of *Chu space* [19] (although I shall continue using some of the information-flow terminology), and build an abstract notion of *knowledge-sharing scenario* upon this concept.

Definition 1 A *Chu space* $\mathbf{A}$ over a set S is a triple (X, r, A) consisting of a set X of *tokens*, a set A of *types*, and a function $r : X \times A \to S$, constituting the *matrix*.

In the Chu space literature, tokens are usually called *points*, while types are called *states* —Chu spaces have been mainly used for defining models of concurrency [13]. Other terms that have been used are: situations, objects, instances, for points; and predicates, attributes, concepts, for states.

In this paper I choose to use Barwise and Seligman's information-flow terminology of tokens and types. Actually, their framework deals exclusively with dyadic Chu spaces, i.e., Chu spaces over two-element sets; they call them *classifications* [3].

Definition 2 Let $\mathbf{A} = (X, r, A)$ and $\mathbf{B} = (Y, s, B)$ be two Chu spaces. A *Chu transform* $f : \mathbf{A} \to \mathbf{B}$ is a pair $f = (f^\star, f_\star)$ of functions $f^\star : A \to B$ and $f_\star : Y \to X$ such that, for all $y \in Y$ and $a \in A$, $r(f_\star(y), a) = s(y, f^\star(a))$:

$$\begin{array}{ccc} a & \xmapsto{f^\star} & f^\star(a) \\ \Big| & & \Big| \\ r(f_\star(y),a) & = & s(y,f^\star(a)) \\ \Big| & & \Big| \\ f_\star(y) & \xleftarrow[f_\star]{} & y \end{array}$$

In the Chu space literature, Chu transforms are usually defined forwards on tokens and backwards on types. Again, I follow Barwise and Seligman's information-flow perspective and define them forwards on types and backwards on tokens.

A knowledge-sharing scenario, as we have seen in the previous examples, can be as simple as a pair of Chu transforms with common domain, but in general it consists of a complex system of agreed understandings. Hence it involves multiple Chu spaces, modelling the knowledge embedded in several agents, which are interconnected by different Chu transforms, modelling the way in which the knowledge is shared:

Definition 3 A *knowledge-sharing scenario* $\mathfrak{S} = (\mathcal{S}, \mathcal{T})$ consists of a finite family of Chu spaces $\mathcal{S} = \{\mathbf{A}_i\}_{i \in I}$, together with a finite family of Chu transforms $\mathcal{T} = \{f_j\}_{j \in J}$ with domain and codomain in $\mathcal{S}$.

A knowledge-sharing scenario is therefore a categorical diagram in the Chu category, and, in this respect, Barwise and Seligman's *distributed systems* [3] are knowledge-sharing scenarios with dyadic Chu spaces.

3.1 Sharing Probabilistic Knowledge

In order to show the potential advantages of having knowledge-sharing scenarios formally specified as a family of Chu spaces and Chu transforms, let us re-analyse the scenario studied by Corrêa da Silva et al. in [6] within the context of the above formalisation. The scenario consists of a system that extends a probabilistic logic program with queries to Bayesian belief networks. The system attempts to solve queries with the logic program, and

poses failed queries to a belief network. Let us, therefore, recall some theoretical preliminaries of probabilistic logic programming and Bayesian belief networks.

3.1.1 Probabilistic logic programming

A *probabilistic logic program* Π is a logic program that may be used to compute degrees of belief as probabilities of events associated to queries with respect to a *probability space* $(\Omega, \mathcal{A}, P)$, where

1. Ω is a set of *possible worlds*: a set of truth-value assignments to the formulas of the logical language associated to program Π;

2. $\mathcal{A}$ is an *algebra*: a subset of 2^Ω, such that $\Omega \in \mathcal{A}$, for all $X, Y \in \mathcal{A}$, $X \cup Y \in \mathcal{A}$, and for all $X, Y \in \mathcal{A}$, $Y \setminus X \in \mathcal{A}$;

3. $P : \mathcal{A} \rightarrow [0, 1]$ is a *probability measure*: a real-valued function, such that $P(\Omega) = 1$ and for all disjoint sets $X, Y \in \mathcal{A}$, $P(X \cup Y) = P(X) + P(Y)$.

Since the probability measure P is not defined over 2^Ω, but only over a subset $\mathcal{A}$, the operational semantics of probabilistic logic programming computes an interval in which the degree of belief of a query ϕ lies. Let E_ϕ denote the event associated to the formula ϕ, i.e., $E_\phi = \{w \in \Omega \mid \phi \text{ is true in } w\}$. The interval is determined by the inner and outer probability measures of E_ϕ, defined as

$$\begin{aligned} P_*(E_\phi) &= sup\{P(X) \mid X \subseteq E_\phi, X \in \mathcal{A}\} \\ P^*(E_\phi) &= inf\{P(X) \mid E_\phi \subseteq X, X \in \mathcal{A}\} \ . \end{aligned}$$

A helpful notion is that of *basis* of an algebra. It is a partition $\{X_i\}_i$ of Ω, such that $X_i \in \mathcal{A}$ and there exists no $X \in \mathcal{A}$ such that $X \subseteq X_i$. With a basis, the definitions of inner and outer probability measure are as follows:

$$\begin{aligned} P_*(E_\phi) &= P(\bigcup\{X \mid X \subseteq E_\phi, X \text{ is in the basis of } \mathcal{A}\} \\ P^*(E_\phi) &= P(\bigcup\{X \mid X \cap E_\phi \neq \emptyset, X \text{ is in the basis of } \mathcal{A}\} \ . \end{aligned}$$

In the reamining of the paper I will use ϕ, χ, ψ to denote both, formulas as well as events associated to formulas. Hence, I will write $P_*(\phi)$ for $P_*(E_\phi)$ (and $P^*(\phi)$ for $P^*(E_\phi)$). For further details on probabilistic logic programming, see [17].

Let Π be the probabilistic logic program of Figure 6. It is assumed that the truth-values of clauses are the same across all possible worlds, hence the

$$
\begin{array}{lcl}
p(X) \leftarrow q(X) \wedge r(X) & : & [1.0, 1.0] \\
q(X) \leftarrow \neg s(X) & : & [1.0, 1.0] \\
r(X) \leftarrow t(x) & : & [1.0, 1.0] \\
s(a) & : & [0.3, 0.8] \\
s(b) & : & [0.0, 0.0] \\
t(a) & : & [0.6, 1.0]
\end{array}
$$

Figure 6: A probabilistic logic program

intervals $[1.0, 1.0]$ that annotate clauses. An interval $[p_1, p_2]$ annotating a fact ϕ means that $P_*(\phi) = p_1$ and $P^*(\phi) = p_2$. According to program Π, the degree of belief of query $p(a)$ lies in interval $[0.0, 0.7]$, and is computed using the following rules:

For clauses of the form $\phi \leftarrow \chi$:

$$
\begin{array}{rcl}
P_*(\phi) & = & P_*(\chi) \\
P^*(\phi) & = & P^*(\chi)
\end{array}
$$

For clauses of the form $\phi \leftarrow \neg\chi$:

$$
\begin{array}{rcl}
P_*(\phi) & = & 1 - P^*(\chi) \\
P^*(\phi) & = & 1 - P_*(\chi)
\end{array}
$$

For clauses of the form $\phi \leftarrow \chi \wedge \psi$:

$$
\begin{array}{rcl}
P_*(\phi) & = & max(0, P_*(\chi) + P_*(\psi) - 1) \\
P^*(\phi) & = & min(P^*(\chi), P^*(\psi))
\end{array}
$$

3.1.2 Bayesian belief networks

Let $V = \{V_1, \ldots, V_n\}$, $n \geq 1$, be a set of variables. A *configuration* c_V is a conjunction of truth-value assignments to the variables in V. Let $\mathcal{C}_V$ denote the set of all configurations c_V. A *Bayesian belief network* $\mathcal{B} = (\mathcal{G}, \mathcal{P})$, consists of

1. an acyclic graph $\mathcal{G} = (V, A)$ with nodes $V = \{V_1, \ldots, V_n\}$, $n \geq 1$ and arrows A, and

2. a set $\mathcal{P} = \{P_{V_i} \mid V_i \in V\}$ of real-valued functions $P_{V_i} : \mathcal{C}_{V_i} \times \mathcal{C}_{prec(V_i)} \rightarrow [0, 1]$, where $prec(V_i)$ denotes the set of variables that immediately precede V_i in the graph.

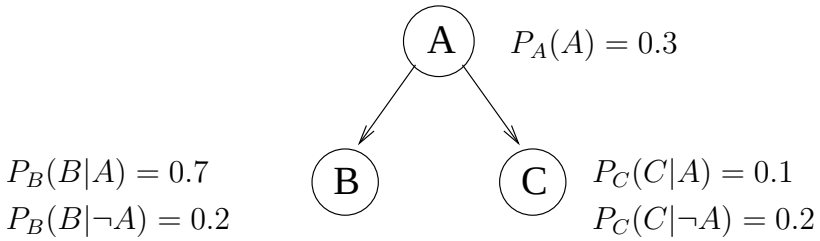

Figure 7: A Bayesian belief network

The set $\mathcal{P}$ of real-valued functions uniquely determines a joint probability function $P_V : V \to [0, 1]$, that can be extended to conditional probabilities using Bayes's rule, yielding a function $\mathsf{P} : \mathcal{C} \times \mathcal{C} \to [0, 1]$, where $\mathcal{C}$ is the set of all configurations, i.e., $\mathcal{C} = \{c_W \mid W \subseteq V\}$. For further details on Bayesian belief networks, see [25].

3.1.3 Linking programs with networks

Notice that with program Π of Figure 6 we cannot compute an interval for the degree of belief of $p(b)$, because there is no probability interval associated to the fact $t(b)$. For these failing queries, Corrêa da Silva et al. have studied how to use Bayesian belief networks to calculate the probability of facts missing in the program. Such scenario is sensible when the networks compile knowledge that the logic program is lacking. For instance, in the case above, we may have the network of Figure 7 available to compute an interval for $t(b)$, and choose to place the degree of belief of $t(b)$ between the probabilities $\mathsf{P}(A \wedge B|C)$ and $\mathsf{P}(A|C)$ of two nested events, which places its degree of belief in interval $[0.12535, 0.17647]$. For further details, see [6]. Now, let us model this engineering decision as an abstract knowledge sharing scenario, and see what we can draw from it.

3.2 A Scenario for Probabilistic Knowledge Sharing

The flow of knowledge is modelled by the following Chu transforms, capturing the duality existing between the merging of those syntaxes used to predicate about the shared situations and its correspondences, and the linking of those shared situations through which the knowledge flows reliably. I am going to model the flow of inner probabilities only, as for outer probabilities the scenario is similar.

3.2.1 Modelling the agents

First of all let us model the knowledge systems involved in our scenario as Chu spaces. We model the knowledge compiled by a probabilistic logic program Π with Chu space $\mathbf{P} = (\mathcal{S}, r, B_\Pi)$ as follows:

- the set of tokens $\mathcal{S}$ is a set of probability spaces,
- the set of types B_Π is the Herbrand base of program Π, and
- the matrix is defined by the inner probability of events associated to formulas with respect to probability spaces, i.e., $r((\Omega, \mathcal{A}, P), \phi) = P_*(\phi)$, for $(\Omega, \mathcal{A}, P) \in \mathcal{S}$ and $\phi \in B_\Pi$.

And we model the knowledge compiled by a Bayesian belief network $\mathcal{B} = (\mathcal{G}, \mathcal{P})$, $\mathcal{G} = (V, A)$, with Chu space $\mathbf{N} = (\mathcal{C}, s, \mathcal{C})$ as follows:

- the set of tokens $\mathcal{C}$ is the set of all configurations c_W with $W \subseteq V$,
- the set of types is also the set $\mathcal{C}$ of all configurations, and
- the matrix is defined by the conditional probability of configurations, i.e., $s(c_U, c_W) = \mathsf{P}(c_U | c_W)$, where P is the joint probability function determined by the functions in $\mathcal{P}$.

Next, let us model, with additional Chu spaces and Chu transforms, how the knowledge is to be shared.

3.2.2 Querying the logic program

The extended program, as we would like to have it, is to be defined over the same vocabulary; hence, its set of types will be the same as for Chu space $\mathbf{P}$, although it will differ on the tokens, i.e., the probability spaces, because we want to alter the degree of belief of some of the queries (by consulting a belief network). Let us formalise the extended program with Chu Space $\mathbf{X} = (\mathcal{S}^\diamond, t, B_\Pi)$. As we shall see later, probability spaces in $\mathcal{S}^\diamond$ extend the spaces in $\mathcal{S}$ with subsets of possible worlds according to some evidence in the belief network that the program consults.

We would like the probability spaces used to interpret the extended program not to change the degrees of belief of a chosen fragment $A \subseteq B_\Pi$ of the program's queries as they were before the extension. (In our example of Section 3.1.3, A was supposed to be the set of queries that succeed in Π, i.e., facts and its consequences.) A pair of Chu transforms $f : \mathbf{A} \to \mathbf{P}$ and $g : \mathbf{A} \to \mathbf{X}$ from a intermediate Chu space $\mathbf{A}$ is going to capture this requirement. But, as we shall see later in Section 4.1, keeping the same degrees

of belief will not be possible *per se*. Instead, they have to be weighed with respect to the extended probability spaces, and consequently, the matrix t is going to be defined by conditional inner probabilities.

Chu space $\mathbf{A} = (\mathcal{S}, r', A)$ formalises the chosen fragment of program Π. The matrix r', again, is defined by the inner probabilities. On the types, $f^\star$ and $g^\star$ are the inclusion of A into B_Π. On tokens, though, $f_\star$ is the identity of probability spaces —obviously, $f = (f^\star, f_\star)$ is a Chu transform. I shall postpone until Section 4.1 the precise definition of $g_\star$. It projects probability spaces used to interpret the extended program to spaces used to interpret the original program:

$$\begin{array}{ccccc} B_\Pi & \xleftarrow{f^\star} & A & \xrightarrow{g^\star} & B_\Pi \\ \Big| r & & \Big| r' & & \Big| t \\ \mathcal{S} & \xrightarrow[f_\star]{} & \mathcal{S} & \xleftarrow[g_\star]{} & \mathcal{S}^\diamond \end{array}$$

3.2.3 Consulting the belief network

We want to link queries of a chosen fragment $B \subseteq B_\Pi$, disjoint from fragment A, to events that will determine its inner probability. These events are conditional events represented by pairs of configurations in the belief network. When posing a query we start from an evidence, represented by configuration c_W. As before, a pair of Chu transforms $h : \mathbf{B} \to \mathbf{X}$ and $k : \mathbf{B} \to \mathbf{N}$ from an intermediate Chu space $\mathbf{B} = (\mathcal{S}', s', B)$ capture this link together with the requirement that the probability stays the same. Types of $\mathbf{B}$ are those formulas of the Herbrand base that the knowledge engineer decides to link to the network. (In our example of Section 3.1.3, they were taken from the set of queries that the program failed to solve.) The set of tokens consists of a set of probability spaces $\mathcal{S}'$, in principle distinct from $\mathcal{S}$ of $\mathbf{P}$. It will be determined by how the knowledge engineer decides evidences in the belief network should select subsets of possible worlds, as we shall see next; recall that we want to change the degree of belief of some queries precisely by consulting a belief network. The matrix s', again, is defined by the inner probabilities.

Let $\mathcal{B}$ be the Bayesian belief network, with variables V. Let Ω' be a set of possible worlds, and let $\{X_{c_V}\}_{c_V \in \mathcal{C}_V}$ be a partition of Ω', so that each set X_{c_V} in the partition is in bijection with a configuration c_V of all variables of $\mathcal{B}$. Let P be the joint probability of $\mathcal{B}$. On types, $k^\star$ maps a query $\psi \in B$ to a configuration c_U, $U \subseteq V$ (according to the decision of the knowledge engineer). On tokens, $k_\star$ maps a configuration $c_W \in \mathcal{C}$ of $\mathcal{B}$, with $W \subseteq V$, to a probability space $(\Omega', \mathcal{A}', P')$, such that

- $\mathcal{A}'$ is the algebra that has subsets X_{c_V}, with $c_V \in \mathcal{C}_V$, as its basis.
- $P'(X) \stackrel{def}{=} \sum_{X_{c_V} \subseteq X} P'(X_{c_V})$, where $P'(X_{c_V}) \stackrel{def}{=} \mathsf{P}(c_V|c_W)$.

Obviously, we cannot freely select Ω' and its partitions. The type-level decision that fixes $k^\star$ will constrain our choice of set of possible worlds and the way we partition it to determine $\mathcal{A}'$ and P'. Actually, this constraint arises from wanting k to be a Chu transform, and so respect the degrees of beliefs with respect to the conditional probabilities compiled in the network. This is asserted in the following lemma.

Lemma 4 *Let $\mathcal{X}$ be the set of all those X_{c_V} of the partition of Ω' for which ψ is true in all possible worlds $w \in X_{c_V}$; let $c_U = k^\star(\psi)$, and let c_W be a configuration in $\mathcal{C}$. The contravariant pair of functions $k = (k^\star, k_\star)$ is a Chu transform if and only if $P'(\bigcup \mathcal{X}) = \mathsf{P}(c_U|c_W)$.*

PROOF: Obvious, because of the definition of $P'_*(\psi)$ in terms of the basis of the algebra. □

As before with Chu transform g, I shall postpone the precise definition of Chu transform h until Section 4.1. On types, $h^\star$ is the inclusion from B into B_Π. On tokens, $h_\star$ projects probability spaces used to interpret the extended program to spaces that arise from the way $k_\star$ selects a set of possible worlds from a given evidence c_W:

$$\begin{array}{ccccc} B_\Pi & \xleftarrow{h^\star} & B & \xrightarrow{k^\star} & \mathcal{C} \\ \Big| t & & \Big| s' & & \Big| s \\ \mathcal{S}^\diamond & \xrightarrow[h_\star]{} & \mathcal{S}' & \xleftarrow[k_\star]{} & \mathcal{C} \end{array}$$

3.2.4 The scenario

The entire knowledge-sharing scenario consists of all Chu transforms we have been introducing so far linked together, and constitute a diagram in the Chu category (see Figure 8).

Being able to define the scenario is not a guarantee for the existence of a knowledge-sharing system that conforms to the requirements captured in the Chu transforms of the scenario. Although we have already fully defined the Chu transforms at the type level, we have not said anything about the particular probability spaces in $\mathcal{S}^\diamond$ in which the knowledge is going to be shared, yet; especially if such probability spaces actually exist. In other

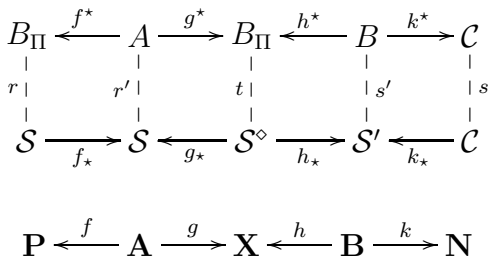

Figure 8: The entire knowledge-sharing scenario and its representation as diagram in the Chu category

words, the existence of tokens in a Chu space that covers the whole scenario is crucial for the system to share knowledge reliably. This is the subject of the next section.

4 Reliable and Optimal Knowledge Sharing

One of the advantages of a precise formalisation of what we consider to be a knowledge-sharing scenario is that such formalisation yields us with a precise way to check if a system reliably covers a scenario, and also to identify when a system is optimal with respect to the knowledge it shares:

Definition 5 Let $\mathfrak{S} = (\mathcal{S}, \mathcal{T})$ be a knowledge scenario with $\mathcal{S} = \{\mathbf{A}_i\}_{i \in I}$. A *knowledge-sharing system* for $\mathfrak{S}$ is a Chu space $\mathbf{S}$, together with Chu transforms $g_i : \mathbf{A}_i \to \mathbf{S}$ such that, for all $f : \mathbf{A}_j \to \mathbf{A}_k$ in $\mathcal{T}$, $g_k \circ f = g_j$. System $\mathbf{S}$ is *reliable* if its set of tokens is non-empty; it is *optimal* if it is reliable and, for any other reliable system $\mathbf{S}'$ with Chu transforms $g'_i : \mathbf{A}_i \to \mathbf{S}'$, there exists a unique Chu transform $h : \mathbf{S} \to \mathbf{S}'$ such that $h \circ g_i = g'_i$.

A knowledge-sharing system for a scenario $\mathfrak{S}$ is a cocone with base diagram $\mathfrak{S}$. Because of the duality of Chu spaces and Chu transforms, such a cocone is a cocone on the type level, but a cone on the token level. The system is reliable if the cone is not trivial, i.e., it is not the empty set. In that case it will have situations (tokens) in which the knowledge is actually shared. Notice that the dual approach to knowledge sharing based on Chu spaces is crucial, since no link on the token level means that no shared situation is possible.

The optimality of a reliable system is modelled as a colimit. This is sensible, because the Chu category has colimits [1]. The colimit condition

amounts to say, roughly, that the optimal system is minimal with respect to the theories it merges, but maximal with respect to the situations it covers, according to the scenario $\mathfrak{S}$. Consequently, if the colimit has an empty set of tokens, no reliable system for the scenario can exist.

It would be desirable, given a knowledge-sharing scenario, to find the optimal system for it. In this paper I am not going to discuss how such optimal system is computed. On might suspect that this will not always be possible to do in practise. Instead, I show that from the notion of reliable system we can justify sufficient conditions for a consistent flow of knowledge between the agents of a knowledge-sharing scenario.

4.1 A reliable system for probabilistic knowledge sharing

With respect to the knowledge-sharing scenario first proposed in [6] and re-analysed here in Section 3.2, Corrêa da Silva et al. mention that we need to extend the logic program with a completely fresh subset of possible worlds, and that we need independence assumptions of the algebras and probability measures involved. They do not, though, go into the details explaining how such subset of possible world is selected and how the independence assumptions arise.

The advantage of having the scenario formalised according to the abstract notion of knowledge-sharing scenario proposed in Definition 3 is that, with the notion of cocone of a scenario, as given in Definition 5, we are now in a position of justifying sufficient conditions for reliable sharing of knowledge. We have already seen in Section 3.2.3 how knowledge engineering decisions on the type level, namely for linking formulas with configurations of a belief network, constrain the token level choice of valid probability spaces.

Recall the scenario formalised in Section 3.2, and let us denote it with $\mathfrak{S}$. Recall, also, that I have denoted with B_Π the Herbrand base of program Π, and with A and B two disjoint fragments of it. Recall, also, that $\mathcal{S}$, $\mathcal{S}'$ and $\mathcal{S}^\diamond$ were sets of probability spaces, and that $\mathcal{C}$ was standing for the set of all configurations of the belief network.

Let $\mathbf{S}$ be the Chu space $(\mathcal{O}, u, B_\Pi \cup \mathcal{C})$, i.e., with atoms of the Herbrand base and configurations as types, and with a subset $\mathcal{O} \subseteq \mathcal{S} \times \mathcal{C}$ of pairs of probability spaces $(\Omega, \mathcal{A}, P)$ and configurations c_W as tokens. In $\mathcal{O}$ we only want to have those pairs such that $k_\star$ selects fresh subsets of possible worlds (disjoint from Ω). Let $(\Omega, \mathcal{A}, P) \in \mathcal{S}$ and $c_W \in \mathcal{C}$, and let $(\Omega', \mathcal{A}', P') = k_\star(c_W)$. Let us define $\mathcal{O}$ as follows. A pair $\langle (\Omega, \mathcal{A}, P), c_W \rangle \in \mathcal{O}$ if and only if $\Omega \cap \Omega' = \emptyset$.

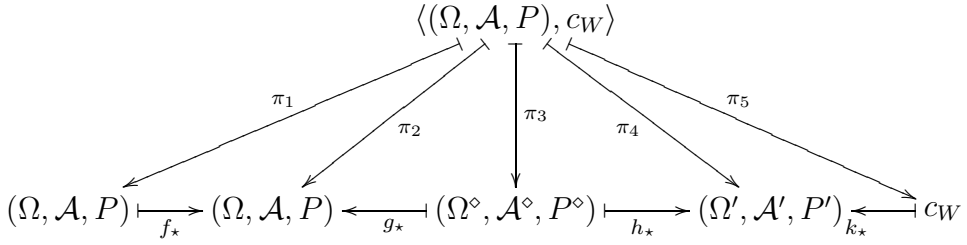

Figure 9: Token level for reliable knowledge sharing

To complete the definition we need to define projections π_1–π_5 (see Figure 9). The injections on the type level are all inclusions. Projections π_1 and π_5 are obvious, and $\pi_2 = \pi_1$, and $\pi_4 = k_\star \circ \pi_5$. The crucial projection is π_3 that projects pairs of probability spaces and configurations to shared probability spaces in $\mathcal{S}^\diamond$. Let us define π_3 as follows.

Given a pair $\langle(\Omega, \mathcal{A}, P), c_W\rangle$ and supposing that $k_\star(c_W) = (\Omega', \mathcal{A}', P')$,

$$\pi_3(\langle(\Omega, \mathcal{A}, P), c_W\rangle) \stackrel{def}{=} (\Omega^\diamond, \mathcal{A}^\diamond, P^\diamond)$$

where

1. $\Omega^\diamond \stackrel{def}{=} \Omega \cup \Omega'$,
2. $\mathcal{A}^\diamond$ is the algebra whose basis is the union of the basis of $\mathcal{A}$ and $\mathcal{A}'$, and
3. for all $X \in A^\diamond$,

$$P^\diamond(X) \stackrel{def}{=} P(X \cap \Omega)P^\diamond(\Omega) + P'(X \cap \Omega')(1 - P^\diamond(\Omega))$$

such that $P^\diamond(\Omega) \neq 0$.

We still need to define the token-level functions of Chu transforms g and h that we had postponed from Sections 3.2.2 and 3.2.3. But, first of all let us go back to Chu space $\mathbf{X} = (\mathcal{S}^\diamond, t, B_\Pi)$. We said in Section 3.2 that the spaces in $\mathcal{S}^\diamond$ were extending the original probability spaces with subsets of possible worlds according to some evidence in the belief network that the program consults. We also mentioned that the matrix t was to be defined by conditional inner probabilities. Actually, types in $\mathbf{X}$ are formulas B_Π *together with two distinguish subsets* $A, B \subseteq B_\Pi$ that form a parition of B_Π, and tokens are probability spaces $(\Omega^\diamond, \mathcal{A}^\diamond, P^\diamond)$ *together with two distinguish subsets* $\Omega, \Omega' \subseteq \Omega^\diamond$; the matrix t is then defined as follows:

$$\begin{array}{rcll} t((\Omega^\diamond, \mathcal{A}^\diamond, P^\diamond), \phi) & = & P^\diamond_*(\phi|\Omega) & \text{for } \phi \in A \\ t((\Omega^\diamond, \mathcal{A}^\diamond, P^\diamond), \psi) & = & P^\diamond_*(\psi|\Omega') & \text{for } \psi \in B \end{array}$$

The definition of $g_\star$ follows. Function $h_\star$ is defined analogously. Given a probability space $(\Omega^\diamond, \mathcal{A}^\diamond, P^\diamond)$ and a subset of possible worlds $\Omega \in \mathcal{A}^\diamond$ such that $P^\diamond(\Omega) \neq 0$,

$$g_\star(\Omega^\diamond, \mathcal{A}^\diamond, P^\diamond) \stackrel{def}{=} (g_\star(\Omega^\diamond), g_\star(\mathcal{A}^\diamond), g_\star(P^\diamond))$$

where

1. $g_\star(\Omega^\diamond) \stackrel{def}{=} \Omega$,
2. $g_\star(\mathcal{A}) \stackrel{def}{=} \{X \cap \Omega \mid X \in \mathcal{A}^\diamond\}$, and
3. for all $X \in \mathcal{A}$,
$$g_\star(P)(X) \stackrel{def}{=} \frac{P^\diamond(X)}{P^\diamond(\Omega)} \ .$$

Obviously, there may be many ways to choose subset Ω, so $g_\star$ is not unique, in general. Nevertheless, the Chu space $\mathbf{S}$ and its projections π_1–π_5 defined above only are going to form a reliable system for those scenarios for which $g_\star(\Omega^\diamond)$ is the original set of possible worlds used to interpret the program in Chu space $\mathbf{P}$, and $h_\star(\Omega^\diamond)$ is the set of possible worlds selected by the consultation of the belief network.

The following lemma asserts that $(g_\star(\Omega^\diamond), g_\star(\mathcal{A}^\diamond), g_\star(P^\diamond))$ is indeed a probability space, and therefore $g_\star$ is well defined.

Lemma 6 *Let $(\Omega^\diamond, \mathcal{A}^\diamond, P^\diamond)$ be a probability space; its image $g_\star(\Omega^\diamond, \mathcal{A}^\diamond, P^\diamond) = (g_\star(\Omega^\diamond), g_\star(\mathcal{A}^\diamond), g_\star(P^\diamond))$ is a probability space, too.*

PROOF: $g_\star(\mathcal{A}^\diamond)$ is an algebra: $\Omega \in g_\star(\mathcal{A}^\diamond)$, because $\Omega \in \mathcal{A}^\diamond$. Let $X, Y \in g_\star(\mathcal{A}^\diamond)$, therefore there exist $X^\diamond, Y^\diamond \in \mathcal{A}^\diamond$ such that $X \cap \Omega = X^\diamond$ and $Y \cap \Omega = Y^\diamond$. But, since $\mathcal{A}^\diamond$ is an algebra, $X^\diamond \cup Y^\diamond \in \mathcal{A}^\diamond$, and hence $(X \cup Y) \cap \Omega \in \mathcal{A}^\diamond$. Consequently $X \cup Y \in g_\star(\mathcal{A}^\diamond)$. Analogously $X \setminus Y \in g_\star(\mathcal{A}^\diamond)$.

$g_\star(P^\diamond)$ is well defined: $P^\diamond(\Omega) \neq 0$ and $g_\star(\mathcal{A}^\diamond) \subseteq \mathcal{A}^\diamond$ and hence in the domain of $g_\star(P^\diamond)$: Suppose $X \in g_\star(\mathcal{A}^\diamond)$, but $X \notin \mathcal{A}^\diamond$. Then, by definition of $g_\star(\mathcal{A}^\diamond)$, $X \cap \Omega \notin g_\star(\mathcal{A}^\diamond)$ But, since $X \in g_\star(\mathcal{A}^\diamond)$, $X \subseteq \Omega$, and consequently $X \cap \Omega = X$. Therefore $X \notin g_\star(\mathcal{A}^\diamond)$, which is a contradiction.

$g_\star(P^\diamond)$ is a probability measure: $g_\star(P^\diamond)(\Omega) = \dfrac{P^\diamond(\Omega)}{P^\diamond(\Omega)} = 1$, and for disjoint sets $X, Y \in \mathcal{A}$,

$$\begin{aligned} g_\star(P^\diamond)(X \cup Y) &= \frac{P^\diamond(X \cup Y)}{P^\diamond(\Omega)} = \frac{P^\diamond(X) + P^\diamond(Y)}{P^\diamond(\Omega)} = \frac{P^\diamond(X)}{P^\diamond(\Omega)} + \frac{P^\diamond(Y)}{P^\diamond(\Omega)} \\ &= g_\star(P^\diamond)(X) + g_\star(P^\diamond)(Y) \ . \end{aligned}$$

□

For g and h to be Chu transforms it was necessary that the matrix t of Chu space $\mathbf{X}$ was defined by conditional probabilities. It is now easy to check that g is a Chu transform. All we need to see is that, if $\phi \in A$, then $P^{\diamond}_{*}(\phi|\Omega) = P_{*}(\phi)$. Suppose $\phi \in A$, $X \subseteq \phi$ and $X \in \mathcal{A}^{\diamond}$. By definition of $g_{\star}$,

$$P(X \cap \Omega) = \frac{P^{\diamond}(X \cap \Omega)}{P^{\diamond}(\Omega)} = P^{\diamond}(X|\Omega)$$

Therefore, since $\phi \in A$ and $X \cap \Omega \in \mathcal{A}$,

$$sup\{P^{\diamond}(X|\Omega) \mid X \subseteq \phi, X \in \mathcal{A}^{\diamond}\} = sup\{P(X) \mid X \subseteq \phi, X \in \mathcal{A}\}$$

and consequently, $P^{\diamond}_{*}(\phi|\Omega) = P_{*}(\phi)$. We check that h is a Chu transform analogously.

In Section 3.2.2 I mentioned that keeping the same degrees of belief for the extended program as for the original program was not possible. If we wanted the probability spaces used to interpret the extended program not to change the degrees of belief of a chosen fragment $A \subseteq B_{\Pi}$ we would need to require that

$$P^{\diamond}_{*}(\phi) = P_{*}(\phi)$$

for $\phi \in A$. By the definition of $g_{\star}$ we would have that for all $\phi \in A$ and all $X \in \mathcal{A}^{\diamond}$ such that $X \subseteq \phi$,

$$P^{\diamond}(X) = \frac{P^{\diamond}(X \cap \Omega)}{P^{\diamond}(\Omega)}$$

and hence X would need to be independent from Ω. The same would apply to Ω', for $\psi \in B$. But this does not make any sense, since we originally wanted to query belief network because, for $\psi \in B$, ψ was not true in any possible world in Ω. So we cannot assume independence of ψ and its subsets with respect to Ω and Ω'.

With the scenario $\mathfrak{S}$ and the system $\mathbf{S}$ completely defined (see Figure 10), we are now in position of stating the following theorem:

Theorem 7 *System* $\mathbf{S}$ *is a reliable knowledge-sharing system for scenario* $\mathfrak{S}$ *if, for all tokens* $\langle(\Omega, \mathcal{A}, P), c_W\rangle$ *in* $\mathbf{S}$, $g_{\star}(\Omega \cup \Omega') = \Omega$ *and* $h_{\star}(\Omega \cup \Omega') = \Omega'$, *where* Ω' *is the set of possible worlds of* $k_{\star}(c_W)$.

PROOF: I prove that the $g \circ \pi_3 = \pi$ and $h \circ \pi_3 = k \circ \pi_5$, so that $\mathbf{S}$ is indeed the vertex of a cocone (of a cone on the token level). To proof the first equality, I show that $g_{\star}(\Omega^{\diamond}, \mathcal{A}^{\diamond}, P^{\diamond}) = (\Omega, \mathcal{A}, P)$.

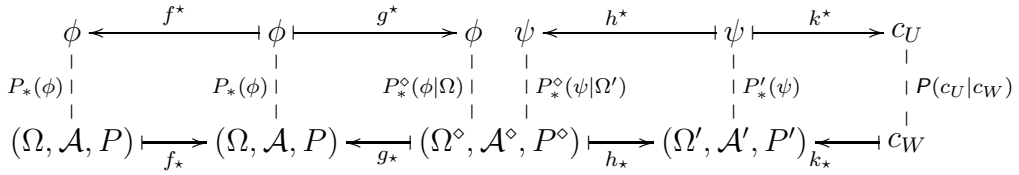

Figure 10: Type and token level of the knowledge-sharing scenario

We are given with $g_\star(\Omega^\diamond) = \Omega$. Suppose $Y \in \mathcal{A}$. This implies $Y \in \mathcal{A}^\diamond$ by definition. But, since $Y \subseteq \Omega$, we have that $Y \cap \Omega = Y$. Consequently, $Y \in \{X \cap \Omega \mid X \in \mathcal{A}^\diamond\} = g_\star(\mathcal{A}^\diamond)$. Now suppose $Y \in g_\star(\mathcal{A}^\diamond)$. Hence, $Y = X \cap \Omega$ with $X \in \mathcal{A}^\diamond$. There exists $X_1, \ldots, X_n$ in the basis of $\mathcal{A}$ and $X_{n+1}, \ldots, X_m$ in the basis of $\mathcal{A}'$, such that

$$X = X_1 \cup \cdots \cup X_n \cup X_{n+1} \cup \cdots \cup X_m$$

But, since $Y = X \cap \Omega = X_1 \cup \cdots \cup X_n$, we have that $Y \in \mathcal{A}$. Consequently, $g_\star(\mathcal{A}^\diamond) = \mathcal{A}$. Let $X \in \mathcal{A}$. By definition of $g_\star$ we have that

$$g_\star(P^\diamond)(X) = \frac{P(X \cap \Omega)P^\diamond(\Omega) + P'(X \cap \Omega')(1 - P^\diamond(\Omega))}{P(\Omega \cap \Omega)P^\diamond(\Omega) + P'(\Omega \cap \Omega')(1 - P^\diamond(\Omega))}$$

Furthermore, since Ω and Ω' are disjoint,

$$g_\star(P^\diamond)(X) = \frac{P(X \cap \Omega)P^\diamond(\Omega)}{P(\Omega)P^\diamond(\Omega)}$$

and because $X \subseteq \Omega$ and $P(\Omega) = 1$, we get

$$g_\star(P^\diamond)(X) = P(X)$$

Consequently, $g_\star(P^\diamond) = P$. The second equality is proved in a similar fashion. □

5 Conclusion

I have proposed categorical diagrams in the Chu category as a formalisation of knowledge-sharing scenarios. This approach makes the duality between merged terminology and shared situations explicit. In terms of information

flow, this crucial duality accounts for the insufficiency of merging ontological theories alone. Some of these insufficiencies have been studied by Corrêa et al. in [7], and the work presented here formalises some of their informal intuitions. A similar approach for modelling the sharing of knowledge that is also based on Barwise and Seligman's channel theory is Kent's Information Flow Framework [14]. It attempts to provide a formal foundation for the development of upper level ontologies [15].

Here, I have proposed cocones and colimits over diagrams in the Chu category as formal definitions of the reliability and optimality of knowledge-sharing systems, and I have shown the potential uses of this approach by re-analysing in detail a scenario for the sharing of probabilistic knowledge first proposed by Corrêa et al. in [6]. This analysis provides a deeper understanding and more precise justifications of sufficient conditions for reliable flow of information between a probabilistic logic program and Bayesian belief networks.

The following questions arise from our work. First, there is the question about the scope of our formalism and its validity for the analysis of other knowledge-sharing scenarios not considered in this paper. We suspect that it will be applicable to a wide variety of scenarios, because of its abstract nature by being based on general theories of channel theory, Chu space theory, and category theory. Next, there is the question of how optimal systems for knowledge sharing can be computed based on the formalism presented here. We have not addressed this question, yet, but, of course, it would be desirable to do so in future.

Finally, there is the question of the deployment of the theoretical work to practical applications. We are currently working in two directions, here. On one hand we are exploring the use of channel theory and Chu space theory for devising algorithms for ontology alignment and merging that extend and complement previous work based on formal concept analysis [23]. And on the other hand we are implementing a framework that attempts to provide some level of (semi-)automatic support during the engineering process of distributed knowledge-intensive computational systems based on the theoretical ideas presented in this paper [18]. Such framework should force knowledge engineers to be aware of the necessity of an agreed understanding on both, the type and the token level of knowledge sharing scenarios.

Acknowledgements

I am grateful to Flávio S. Corrêa da Silva for helpful comments concerning the knowledge-sharing scenarios discussed in this paper.

This work is supported under the Advanced Knowledge Technologies

(AKT) Interdisciplinary Research Collaboration (IRC), which is sponsored by the UK Engineering and Physical Sciences Research Council under grant number GR/N15764/01. The AKT IRC comprises the Universities of Aberdeen, Edinburgh, Sheffield, Southampton and the Open University.

References

[1] M. Barr. **-Autonomous Categories*, volume 752 of *Lecture Notes in Mathematics*. Springer, 1979.

[2] M. Barr. The Chu construction. *Theory and Applications of Categories*, 2(2):17–35, 1996.

[3] J. Barwise and J. Seligman. *Information Flow: The Logic of Distributed Systems*. Cambridge University Press, 1997.

[4] M. Cerioli et al., editors. *Algebraic System Specification and Developement: A Survey and Annotated Bibliography, 2nd Edition, 1997*. Number 3 in BISS Monographs. Shaker, 1998.

[5] B. Chandrasekaran, J. R. Josephson, and V. R. Benjamins. What are ontologies, and why do we need them? *IEEE Intelligent Systems*, 14(1):20–26, 1999.

[6] F. S. Corrêa da Sliva et al. Knowledge sharing between a probabilistic logic and Bayesian belief networks. In *Proceeding of the International Conference on Processing and Management of Uncertainty*, 2000.

[7] F. S. Corrêa da Sliva et al. On the insufficiency of ontologies: problems in knowledge sharing and alternative solutions. *Knowledge-Based Systems*, 15(3):147–167, 2002.

[8] J. M. Dunn and R. K. Meyer. Combinators and structurally free logic. *Logic Journal of the IGPL*, 5(4):505–538, 1997.

[9] B. Ganter and R. Wille. *Formal Concept Analysis*. Springer, 1999.

[10] J. Goguen. A categorical manifesto. *Mathematical Structures in Computer Science*, 1(1):49–67, 1991.

[11] T. Gruber. Towards principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43:907–928, 1995.

[12] N. Guarino. Formal ontology, conceptual analysis and knowledge representation. *International Journal of Human-Computer Studies*, 43:625–640, 1995.

[13] V. Gupta. *Chu Spaces: A Model of Concurrency*. PhD thesis, Stanford University, 1994.

[14] R. E. Kent. The information flow foundation for conceptual knowledge organization. In *Sixth International Conference of the International Society for Knowledge Organization*, 2000.

[15] R. E. Kent. A KIF formalization of the IFF category theory ontology. In *IJCAI 2001 Workshop of the IEEE Standard Upper Ontology*, 2001.

[16] R. Neches et al. Enabling technology for knowledge sharing. *AI Magazine*, 12(3):16–36, 1991.

[17] R. Ng and V. S. Subrahmanian. Probabilistic logic programming. *Information and Computation*, 1992.

[18] S. Potter, D. Robertson, and W. M. Schorlemmer. Formal knowledge management in distributed environments. Technical report, CISA, Informatics, University of Edinburgh, 2002. Submitted.

[19] V. R. Pratt. The Stone gamut: A coordination of mathematics. In *Logic in Computer Science*, pages 444–454. IEEE Computer Society Press, 1995.

[20] G. Restall. *An Introduction to Substructural Logics*. Routledge, 2000.

[21] D. Sannella and A. Tarlecki. Essential concepts of algebraic specification and program development. *Formal Aspects of Computing*, 9:229–269, 1997.

[22] J. Sowa. *Knowledge Representation and Reasoning*. Brooks/Cole, 2000.

[23] G. Stumme and A. Maedche. Ontology merging for federated ontologies on the semantic web. In *International Workshop on Foundations of Models for Information Integration*, Lecture Notes in Artificial Intelligence. Springer, 1995.

[24] M. Uschold and M. Gruninger. Ontologies: Principles, methods, and applications. *Knowledge Engineering Review*, 11(2):93–155, 1996.

[25] L. C. van der Gaag. Bayesian belief networks: Odds and ends. *The Computer*, 39:97–113, 1996.