The *Information Flow* Foundation for Fusing Inferences

Bo Hu
IAM Group, ECS
University of Southampton, UK
bh@ecs.soton.ac.uk

Abstract

A practical and efficient approach to hybrid reasoning is to fuse different specialised systems that are designed particularly for different types of problems. We provide formal treatments to the fusion of inferences using *information flow* as the theoretical framework. The fusion of DL- and Constraint-based systems is used as an example to demonstrate the applicability.

Motivation and background

Taxonomies have been the fundamental structures in various disciplines of natural sciences. Classifying taxonomies has been adopted as one of the critical capabilities that are sought after in the *Semantic Web* initiative to be of assistance in organising the geometrically increasing knowledge on the Internet (Berners-Lee, Hendler, & Lassila 2001). Recently, substantial research has been devoted to the development of more capable inferential systems. As a result, Description Logic (DL) based taxonomic reasoning systems have improved significantly. It is our contention that the homogeneous approach adopted by such systems is limited in three ways: (i) the inevitable limitation of resources has emphasized a "trade-off", i.e. the expressive power is restricted to ensure computational tractability, completeness and decidability; (ii) the specialist nature of their reasoning means that they are only successful at carrying out particular tasks; and (iii) due to the diversity of the real-world applications, any new extensions to an inferential system are likely to incur new limitations to be discovered.

Imagine a scenario that one has to build a decision support system with the underlying ontology containing not only concepts such as *parent*, recursively defined as "human beings who have human beings as children", but also numeric restrictions, e.g. a *first class student* has to score "B" in all the courses that he/she enrolled in. With the current progress of the DL-based reasoning, representing and classifying such heterogeneous knowledge are not difficult. Numeric constraints can be introduced as the *concrete datatype* in $\mathcal{SHOQ}(\mathrm{D})$ and reasoned with an extended tableau-based algorithm (Horrocks & Sattler 2001). So far so good, the problem seems solved as system developers have already thought about the situations that a knowledge engineer might come across and been well prepared for it. However, reality is not always as perfect as we assume it would be. It might be the case that a more complete specification is after for the *first class student* which leads to a further restriction as

Example 1 *A first class student is a student who passes all the courses that he/she enrolled in and scores "A" in two thirds of them.*

Questions arise as no single existing DL-based system is capable of tackling such numeric constraints when constructing the taxonomies. Even if there is one, we will soon explore its limitation and face the same problem, especially in the context of *Semantic Web* where hybrid knowledge abounds.

This, to some extent, underpins the conclusion that because developing a general and omnipotent reasoning system is practically difficult, *ad hoc* adaptations of existing systems to meet the inferential requirements are inevitable (Doyle & Patil 1991). However extending the underlying algorithm of a reasoning system might be very difficult and impractical for knowledge engineers to achieve at the application stage, as such a process might require in-depth knowledge of the reasoning systems and extensive programming skills. Hence, it is reasonable to believe that if a knowledge model is too expressive to be analysed solely within the framework of one type of reasoning paradigms, alternative ones could be used jointly. It is equally reasonable to envision a mechanism that shifts the integration efforts from knowledge engineers to software agents (systems). *Inference fusion* (Hu, Arana, & Compatangelo 2003) was proposed to address such issues.

In this paper, we try to establish the theoretical ground of *inference fusion* on which the representation and reasoning powers of DL-based systems and constraint-based systems (referred to as constraint solvers, CS for short, hereafter) can be dynamically joined based on the target problems. We do so with the hope that such a formal treatment can be generalised to other sorts of inferences and can open the door to supervised automated fusion of inferences.

Inference fusion

Inference fusion was proposed as the framework to integrate DL- and constraint-based inferences. It consists of a network of communicating systems each with homogeneous reasoning capabilities from within and contributing to a heterogeneous reasoning problem from without. As illustrated in Figure 1, an interfacing engine (**IE**) evaluates reasoning requests submitted to a collection of reasoning systems which are bundled together as an *engine pool*. Note that in Figure 1 links among systems are ignored to emphasize the dynamic nature of such connections. Upon receiving the request, **IE** decides whether or not its local capabilities are sufficient to deal with the knowledge model associated with the request. In the former case, **IE** could decompose the knowledge model into several homogeneous parts, scheduling and issuing specific reasoning requests to the specialised systems, e.g. ***Inf***$_1$ to ***Inf***$_5$. The results of the inferences performed by these other systems would be merged by **IE**, thus providing an overall answer to the original request. This process is referred to as *inference fusion* (Hu, Arana, & Compatangelo 2003). It provides a means to bring respectively homogeneous inferential systems together to form a virtual reasoner with heterogeneous deductive power.

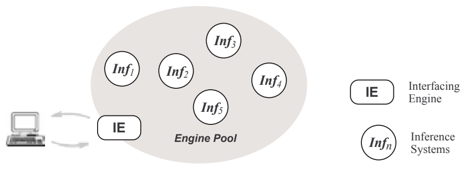

Figure 1: A view of the engine pool

Inference fusion can be generalised to be applied in cases where a joint capability is necessary while finding a single system to perform the reasoning task is impractical due to the availability and/or complexity.

Preliminary

In this section we will give a brief overview of languages and technologies that are relevant in solving Example 1.

Description logics

DLs are a family of knowledge representation and reasoning formalisms that have recently attracted substantial research interest, specially, after the DL-based web ontology modelling languages (e.g. OWL DL, http://www.w3.org/2004/OWL/) were considered to be of crucial importance for the *Semantic Web* initiative (Berners-Lee, Hendler, & Lassila 2001). DLs are based on the notions of concepts (i.e. unary predicates) and properties (i.e. binary predicates). Using different constructors, composite concepts can be built up from atomic ones.

Let *CN* denote a concept name, C and D be arbitrary concepts, R be a role name, *n* be a non-negative integer and $\top$, $\bot$ denote the top and the bottom. A *concept definition* in DLs is either *CN* $\dot{\sqsubseteq}$ C (partial definition) or *CN* $\doteq$ C (full definition). An interpretation $\mathcal{I}$ is a tuple $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where the nonempty set $\Delta^{\mathcal{I}}$ is the domain of $\mathcal{I}$ and the $\cdot^{\mathcal{I}}$ function maps each concept to a subset of $\Delta^{\mathcal{I}}$ while each role to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. The syntax and semantics of some DL-based constructors are illustrated in Table 1.

Symbol	**Syntax**	**Semantics** (*Interpretation*)
	$\top$ / $\bot$	$\Delta^{\mathcal{I}}$ / $\emptyset$
	A	$\mathsf{A}^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ (A as atomic concept)
	R	$\mathsf{R}^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
$(\mathcal{U})$	$\mathsf{C} \sqcup \mathsf{D}$ ($\mathsf{C} \sqcap \mathsf{D}$)	$\mathsf{C}^{\mathcal{I}} \cup \mathsf{D}^{\mathcal{I}}$ ($\mathsf{C}^{\mathcal{I}} \cap \mathsf{D}^{\mathcal{I}}$)
	$\forall \mathsf{R.C}$	$\{ c \mid \forall d : \langle c, d \rangle \in \mathsf{R}^{\mathcal{I}} \rightarrow d \in \mathsf{C}^{\mathcal{I}} \}$
$\mathcal{C}$	$\neg \mathsf{C}$	$\Delta^{\mathcal{I}} \setminus \mathsf{C}^{\mathcal{I}}$
$\mathcal{E}$	$\exists \mathsf{R.C}$	$\{ c \mid \exists d : \langle c, d \rangle \in \mathsf{R}^{\mathcal{I}} \wedge d \in \mathsf{C}^{\mathcal{I}} \}$
$\mathcal{N}$	$(\phi_n \mathsf{R})$	$\{c \mid \sharp\{d : \langle c, d\rangle \in \mathsf{R}^{\mathcal{I}}\}\ \phi\ n\}$ $(\phi \in \{\geq, \leq\})$

Table 1: some DL-based constructors

With the unambiguous semantics defined for each and every concept constructor, DLs lend themselves to powerful reasoning algorithms that automatically classify the domain knowledge in hierarchical structures, memorised to provide a cache for further reasoning.

Constraint satisfaction problems

A Constraint Satisfaction Problem (CSP) consists of a sequence of variables, a set of respective domains, and a set of constraints.

Definition 1 (Constraint Satisfaction Problems) *A Constraint Satisfaction Problem (CSP) is a triple ($\mathfrak{V}$, ω, $\mathfrak{C}$) where $\mathfrak{V}$ is a sequence of constrained variables, $v_1, \ldots, v_n$, ω is a mapping that associates $v_i \in \mathfrak{V}$ with a respective domain $\mathfrak{D}_i$. Subsets of the Cartesian product of $\mathfrak{D}_1, \ldots, \mathfrak{D}_n$ are referred to as* ***constraints*** *(denoted as $\mathfrak{C}$), i.e. $\mathfrak{D}_i \times \ldots \times \mathfrak{D}_j$ $(1 \leq i \leq j \leq n)$ indicating the compatible values of $v_i, \ldots, v_j$.*$\Box$

The *domain of constrained variable* v_x (denoted as $\mathfrak{D}_x$) is a set of values that can be assigned to v_x; the assignment of a variable is normally represented as a *label* that is a variable-value pair assigning the value to the variable; and a *k-compound label* simultaneously assigns k variables each a value from the respective domains. Essentially, a constraint on a set of k variables is a set of k compound labels. For instance, $\langle x\, 0\rangle$, $\langle y\, 0\rangle$, $\langle z\, 1\rangle$ giving the compatible values of x, y, and z. A solution to a CSP is an assignment indicating the values that different variables can take simultaneously without violating any constraints in $\mathfrak{C}$.

Solving a CSP can be considered as a repetitive process of applying a sequence of algorithms. It starts with an initial, unstable *constraint store*, which contains a sequence of variables with the respective domains and constraints defined on a subset of the variables. A series of operators responsible for propagating constraints are applied to the *constraint store* iteratively. So-called *constraint propagation*, also referred to as *consistency* algorithms, is a set of techniques seeking to reduce a CSP into a problem that can be proved to be insoluble or is easier to be solved. The constraint propagation continues until the *constraint store* reaches a stable status. In some cases, either a solution is found after propagating the constraints and removing the inconsistent values from $\mathfrak{D}_i$ or a backtrack-free search is obtained. Propagation techniques can also be used along with search algorithms. Nowadays, general CSP solving, e.g. *consistency* techniques, and specialised techniques have been integrated into a constraint programming (CP) framework—the interested reader can refer to (Tsang 1993) for a thorough survey of CSP techniques.

Fusing heterogeneous inferences

The fusion of heterogeneous inferences is, by no means, an unexplored area, e.g. (Brachman, Fikes, & Levesque 1983), ILOG Configurator [1], and the heterogeneous configuration system (Christoph Ranze *et al.* 2002), etc. In this paper, We formalise the notion of *inference fusion* (Section) using *information flow* and the Channel Theory, which provide the theoretical foundation for constructing virtual Hybrid Reasoning Systems (HRSs). Such a formalisation helps to ensure that fusing of deductions from different inferential systems—in our example, a DL-based taxonomic reasoner with numeric constraints—are carried out in a semantically consistent and adaptive manner. "Adaptive" is in the sense that reasoning systems participating in an HRS can be substituted by those with different expressive and deductive power so that a different combination of systems is obtained to meet the requirement of new reasoning requests. "Semantically consistent" is in the sense that for all the systems contributing to the fusion, no change to the underlying reasoning algorithms is necessary, i.e. the mapping is performed at high abstract level. Essentially, this approach could be generalised to automatically merge physically and logically distributed inferential systems on demand.

Information flow

Information Flow is the mathematical model proposed by Barwise and Seligman as a formalisation capturing the information flowing in any sorts of distributed systems ranging from physically distributed control systems to abstract ones (Barwise & Seligman 1997). It is based on the idea that information exchange within a distributed system results from regularities that enable the information possessed by one component of a distributed system to be understood by another component. Such a notion is formalised in *Channel Theory*.

In Channel Theory, a *channel* connecting components that participating in the information exchange by means of mappings among *types* and *tokens*. The interacting components might have their own vocabulary, i.e. they might have different *types* and *tokens* and thus different *classification* between types and tokens; they might have different *constraints* regulating the behaviour of the information that they are conveying. Capturing all the above notions, is the so-called *local logic* defined as follows:

Definition 2 (Local logic) *A* local logic *is a four-tuple* $\mathfrak{L} = (I, T, \models, \vdash)$*, where* I *is a set of* tokens*;* T *is a set of* types*;* $\models$ *is a binary relation,* $\models : I \times F$*, called* classification*; and* $\vdash$ *is a binary relation* $\vdash : \Gamma \times \Delta$, $\Gamma \subseteq T$ *and* $\Delta \subseteq T$.□

The two binary relations give birth to two important parts of a local logic, namely the *classification* and the *theory*. The *classification* of a local logic is a three-tuple $\mathbb{C} = (I, T, \models)$ determining the type of an instance in I, i.e. $i \models t$ ($i \in I$, $t \in T$) means i is of type t. The *theory* of a local logic is a tuple $\mathbb{T} = (T, \vdash)$ which must satisfy the following conditions to be *regular* (Barwise & Seligman 1997):

Identity $\alpha \vdash \alpha$ for all $\alpha \in T$;

Weakening If $\Gamma \vdash \Delta$, then $\Gamma, \Gamma' \vdash \Delta, \Delta'$, for all $\Gamma, \Gamma', \Delta, \Delta' \subseteq T$; and

Global Cut If $\Gamma, \Sigma_0 \vdash \Delta, \Sigma_1$ for each partition $\langle \Sigma_0, \Sigma_1 \rangle$ of any $\Sigma \subseteq T$, then $\Gamma \vdash \Delta$, for all $\Gamma, \Delta \subseteq T$.

Note that commas on the left hand side of $\vdash$ are interpreted conjunctively while on the right hand side disjunctively.

To facilitate the information exchange among separate components of a distributed system, *infomorphism* (Barwise & Seligman 1997) is defined between the local logics of each component involved in the exchange.

Definition 3 (Infomorphism) *An* infomorphism *between the classifications of local logics* $\mathfrak{L} = (I, T, \models, \vdash)$ *and* $\mathfrak{L}' = (I', T', \models', \vdash')$ *is a contravariant pair of functions* $f = \langle f^{\rightarrow}, f_{\leftarrow} \rangle$*, where,* $f^{\rightarrow} : T \rightarrow T'$ *and* $f^{\leftarrow} : I \leftarrow I'$*, so that, for all* $i' \in I'$ *and* $t \in T$*,* $f_{\leftarrow}(i') \models t$ *iff* $i' \models f^{\rightarrow}(t)$

An infomorphism *can be expanded to the whole local logic if* $f^{\rightarrow}$ *is also a theory interpretation,* $f^{\rightarrow} : \mathbb{T} \rightarrow \mathbb{T}'$. □

To some extent, the fusion of heterogeneous inferential systems is equivalent to exchanging the information of inferences among components of a distributed virtual HRS. Hence, the *infomorphism* provides the theory foundation and reasonable motivation to propose the morphism of inferences.

Linking inferential systems

Obviously, fusing inferences is more than establishing a common language within a community of inferential systems, which would be the task of ontology mapping or schema matching. Though *inference fusion* requires the existence of such a language, the desiderata of fusion is to make transparent the inferential process. In other words, in a distributed environment, one system ***Inf*** could access the knowledge possessed by another ***Inf'*** through either translating the knowledge itself into an admissible format and processing the translated knowledge with its own capacity, or let ***Inf'*** do its job and only retrieve the results. It is the latter that we are after. Such an idea is formalised as flow of inferences that is discussed in the following sections.

Before doing that, we have to think philosophically first. "What is inference?" is a question on a par with "what is the truth?" It is the ability to draw the lines between observed data and unobserved parameters and/or underlying laws. Analogously, an inferential system is a computer program that can produce true sentences based on one or more true sentences. Two different potential uses of the term "inferential system" in the literature need to be clarified to avoid any further confusion. We refer to the computer program that implements reasoning algorithms but is not populated with any knowledge models the "*abstract inferential system*" while the program with certain knowledge models that it can work with the "*concrete inferential system*".

Definition 4 (Inferential system) *An* inferential system *is a three-tuple* ***Inf*** $= (M, C, \sigma)$*, where*

1. *M is the underlying knowledge model that* ***Inf*** *works on;*
2. *C is a set of numeric or symbolic constants as classification labels; and*

[1] http://www.ilog.com/products/configurator/

3. *$\sigma : M \times C$ is the algorithm that classifies a subset of M to categories in C.*

We refer to an *inferential system* defined as above a *concrete inferential system*, ***Inf***$_\mathrm{c}$, while the tuple (C, σ) an *abstract inferential system*, denoted as ***Inf***$_\mathrm{a}$. Examples of ***Inf***$_\mathrm{c}$ are knowledge base systems of which the knowledge base management system (KBMS) is the *abstract inferential system* together with other housekeeping functionalities. The contents of the knowledge base are the underlying knowledge model that a KBMS works on. For simplicity, ***Inf*** is used instead of ***Inf***$_\mathrm{c}$ if no confusion is likely.

In practice, it might be impossible to represent the underlying knowledge model of every type of inferential systems using a uniform approach. In our framework, we are only interested in a particular type of knowledge models, i.e. those can be characterised as a *local logic* and leave out other types at this stage. We will refer to such knowledge models the *admissible knowledge models*.

Definition 5 (Admissible Knowledge Model) *An admissible knowledge model, **M$_{ad}$**, of an inferential system **Inf** is a four-tuple **M$_{ad}$** $= (X, F, \models, \vdash)$ where*

1. *F is a set of well-formed formulae (*types *in* Def 2*);*
2. *X is a set of instances that can be classified against F (*tokens *in* Def 2*);*
3. *$\models$: $X \times F$ classifying an instances with respect to F so that $x \models g$ ($g \subseteq F$ and $x \subseteq X$) iff x satisfies g;*
4. *$\vdash$: $F \times F$, a reflexive, transitive and anti-symmetric relation (a partial order) on the subsets of F, so that $g \vdash g'$ $(g, g' \subseteq F)$ iff wherever g holds with respect to certain interpretations, g' holds in the same interpretations.*□

For instance, in a *concrete* DL-based system, X is a finite set of individuals and C a finite set of concepts. $\models$ can be materialised as the instantiation relationship between individuals and the concepts that the individuals instantiate, while $\vdash$ the subsumption relationship among concepts. Similarly, in CSP, X is the Cartesian product of $\mathfrak{D}_i$ (i.e. the set of all the possible assignments), F is the set of constraint expressions describing what values can be assigned to constrained variables simultaneously, $\models$ assigns each constraint expression $g \subseteq F$ an element $x \in X$ so that the constrained variables in g can take simultaneously without falsifying any constraints in F, and $g \vdash g'$ holds when every assignment satisfying g also satisfies g'

Analogue to the *infomorphism* regulating the information flowing from one component of a distributed system to another, we propose the concept of *infermorphism* that facilitates the transformation of inference results while preserves their veridicality to the original systems.

Definition 6 (Infermorphism) *Given two inferential systems based on admissible knowledge models **Inf** $= (M_{ad}, C, \sigma)$ and **Inf** $= (M'_{ad}, C', \sigma')$, an inference morphism (*infermorphism *for short) is a contravariant pair $f^* = \langle f^{\rightarrow}, f_{\leftarrow} \rangle$ such that*

1. *$f^* : M_{ad} \rightleftharpoons M'_{ad}$ is a morphism of their underlying knowledge models M_{ad} and M'_{ad}; and*
2. *$f^{\rightarrow} : C \rightarrow C'$ is an interpretation of the classification labels C in C'.*□

Representing as local logic

In order to justify the fusion of inferences from different systems, one has to demonstrate that the knowledge model of a *concrete inferential system* is indeed a local logic (denoted as $\mathfrak{L}_\mathrm{M}$), the common abstraction facilitating mutual understanding among systems. This amounts to show that 1) the theory of $\mathfrak{L}_\mathrm{M}$ is *regular* and 2) the *normal token* of $\mathfrak{L}_\mathrm{M}$ satisfy all the constraints specified in the theory of $\mathfrak{L}_\mathrm{M}$.

These two conditions follow naturally from the definition of *admissible knowledge model* (**Def.** 5) which determines a local logic $\mathfrak{L}_{\mathrm{M}_{ad}}$ whose theory is the closure of $\vdash$ under **Identity**, **Weakening** and **Global Cut**. When considering a particular case of the inference fusion, we only need to demonstrate that the knowledge model of every system subject to the fusion is admissible and the different sets of classification labels are mutually interpretable.

Inference fusion

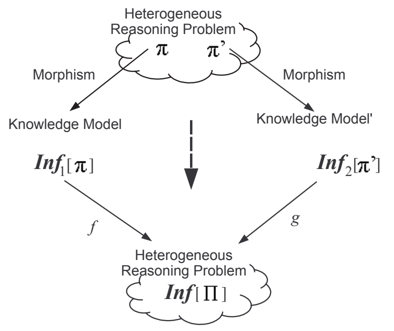

Figure 2: Relating inferential systems with a binary channel

To enable a neat fusion of inferences, a virtual HRS is constructed as the point of reference for all the respectively homogeneous componential systems involved in the heterogeneous reasoning. As depicted in Figure 2, by expressing an inferential system using *local logic*, the *inference fusion* can be formally defined as below.

Definition 7 (Inference Fusion) *Given **Inf**$_i$ the inferential systems ($i \in I$, an index set), **Inf** the channel connecting **Inf**$_i$ and **Inf**$_j$ and f_i the infermorphism functions from **Inf**$_i$ to **Inf**, the fusion of inference with respect to a heterogeneous problem is*

$$\textbf{\textit{Inf}}[\Pi] = \sum^{i \in I} f_i(\textbf{\textit{Inf}}_i[\pi_i])$$

where Π is the heterogeneous problem that is divided into respectively homogeneous sub-problems, π_i.□

$\mathfrak{L}_\mathrm{M}[$***Inf***$[\Pi]]$ is used to capture the local logic for the virtual inferential system populated with Π whose tokens are n-tuples $\langle t_1 \ldots t_n \rangle$ such that t_i is normal token of $\mathfrak{L}_\mathrm{M}[$***Inf***$_i[\pi_i]]$ and whose types are the union of the types of $\mathfrak{L}_\mathrm{M}[$***Inf***$_i[\pi_i]]$. For instance, a *first class student* John would have to satisfy all the DL-based restrictions defined on concept *student* that makes him a normal token in the "DL-sense". Meanwhile, he also has to meet the numeric constraints that, at the same time, makes him a normal token in the "CSP-sense".

In many practical cases, what really concerns us is the projection of one inferential system upon to another while the latter plays a dominant role in fusing their inferences. Be such a case, when we try to use DL-based inference to find out whether *first class student* John is at the same time a *student*—instantiating the DL concept *student*.

Definition 8 (Image of Inference) *The image of* $\boldsymbol{Inf}_i[\pi_i]$ *in* $\boldsymbol{Inf}_j[\pi_j]$ *induced by channel* $\boldsymbol{Inf}$ *is* $f_j^{-1}(\, f_i(\boldsymbol{Inf}_i[\pi_i]))$ *that is*

1. $\mathfrak{L}_M[\boldsymbol{Inf}_j[\pi_j]] = f_j^{-1}(\, f_i(\mathfrak{L}_M[\boldsymbol{Inf}_i[\pi_i]])\,)$*; and*
2. $\boldsymbol{Inf}_j[C_i] = f_j^{-1}(\, f_i(\boldsymbol{Inf}_i[C_i])\,)$.

and vise versa where $i, j \in I$.□

The declarative reading of Def. 8 is projecting $\boldsymbol{Inf}_i$ onto $\boldsymbol{Inf}_j$ is equivalent to recapturing the *local logic* of $\boldsymbol{Inf}_i$ in $\boldsymbol{Inf}_j$ and interpreting the classification labels of $\boldsymbol{Inf}_i$ in $\boldsymbol{Inf}_j$.

Based on Def. 7 and Def. 8, a heterogeneous reasoning problem Π such as the one described in Example 1 can be processed as illustrated in Table 2.

1. Π is fragmented into homogeneous components $\pi_1, \ldots, \pi_n$, based on the known capabilities of participating reasoners.
2. Each of the homogeneous components, π_i ($i \leq n$), is processed by a specialised inferential system, $\boldsymbol{Inf}_i$, and the results are made transparent to others.
3. *Infermorphism* is generated between $\boldsymbol{Inf}_j$ and $\boldsymbol{Inf}_k$ ($j \neq k, j \leq n, k \leq n$) (e.g. $\boldsymbol{Inf}_1$ and $\boldsymbol{Inf}_2$ illustrated in Figure 2).
4. For any two systems connected by an *infermorphism*, the knowledge model of the source system is recaptured in the target via *infermorphism channel* while the classification labels of the source are interpreted in the target.
5. The sum of all inferential systems presents the results of the *inference fusion* process.

Table 2: Fusing Steps

Linkages

Infermorphism must be rooted in the mapping between types and tokens of the knowledge models of different systems. We proposed *Linkages* to materialise such mappings and use f instead of $f^{\rightarrow}$ when no confusion is likely.

Definition 9 (*Linkage*) *In an* inference fusion *problem* $\boldsymbol{Inf}[\Pi] = \sum^i f_i(\boldsymbol{Inf}_i[\pi_i])$*, let* α *be a type in the theory determined by* $\mathfrak{L}_M[\boldsymbol{Inf}[\Pi]]$ *and* α_i *be the type in the theory determined by* $\mathfrak{L}_M[\boldsymbol{Inf}_i[\pi_i]]$*. If* $\alpha_j = f_j^{-1}(f_k(\alpha_k))$*, the relation* $f_j(\alpha_j) = \alpha = f_k(\alpha_k)$ *induced by* α *is called the* linkage *between* $\boldsymbol{Inf}_j$ *and* $\boldsymbol{Inf}_k$ ($i, j, k \in I$)*.*□

Intuitively, *linkages* map the intrinsic data structures of different systems. They, responsible for the inter-system communications, are the carriers of the inbound and outbound flows of information for an inferential system. *Linkage* must satisfy the following requirements:

- They should be such that no modification must be made to the underlying algorithms of either inferential system, i.e. *linkages* link the data structures semantically.
- They should have enough expressive power to reflect the relevant inferential results from one system to the other without causing knowledge loss or increasing the computational complexity in either system, i.e. *linkages* link the native data structures only.

So, what is the *first class student*?

Let's go back to Example 1. As proposed in (Hu, Arana, & Compatangelo 2003), the idiosyncratic definition of *first class student* can be represented using a DL-like language, $\mathcal{DL}(\mathrm{D})/\mathrm{S}$. Reasoning with an ontology containing such concepts amounts to defining the *infermorphism* between DL- and constraint-based systems. The heterogeneous problem of such a particular case is $\Pi = \{\pi_{\mathrm{DL}}, \pi_{\mathrm{CSP}}\}$.

In the *first class student* case, we are seeking for taxonomies taking into account the non-DL inferences that are induced through $\boldsymbol{Inf}$ as $f_{\mathrm{DL}}^{-1}(\, f_{\mathrm{CS}}(\boldsymbol{Inf}_{\mathrm{CS}}[\pi_{\mathrm{CSP}}]))$. And the taxonomies with respect to the whole problem (both DL-based definitions and CSP restrictions) are obtained as:

$$\boldsymbol{Inf}_{\mathrm{DL}}[\Pi] = f_{\mathrm{DL}}^{-1}(\, f_{\mathrm{CS}}(\boldsymbol{Inf}_{\mathrm{CS}}[\pi_{\mathrm{CSP}}])) + \boldsymbol{Inf}_{\mathrm{DL}}[\pi_{\mathrm{DL}}]$$

where $\boldsymbol{Inf}_{\mathrm{DL}}$ stands for the DL-based systems, $\boldsymbol{Inf}_{\mathrm{CS}}$ the constraint solvers. From the foundation work laid out in previous sections, we can see that the core of the above equation is the *infermorphism* from $\boldsymbol{Inf}_{\mathrm{CS}}$ to $\boldsymbol{Inf}_{\mathrm{DL}}$ which is the morphism between *local logics*, i.e. from $\mathfrak{L}_{\mathrm{CS}} = \mathfrak{L}_{\mathrm{M}}[\boldsymbol{Inf}_{\mathrm{CS}}[\pi_{\mathrm{CSP}}]]$ to $\mathfrak{L}_{\mathrm{DL}} = \mathfrak{L}_{\mathrm{M}}[\boldsymbol{Inf}_{\mathrm{DL}}[\pi_{\mathrm{DL}}]]$ and the interpretation of classification labels of $\boldsymbol{Inf}_{\mathrm{CS}}$ in $\boldsymbol{Inf}_{\mathrm{DL}}$.

Linking constraints and DL concepts

So far morphism is defined with respect to the entire knowledge models. This, however, might result in *unsoundness* as *surjectiveness* between tokens from $\mathfrak{L}_{\mathrm{DL}}$ and $\mathfrak{L}_{\mathrm{CS}}$ is not guaranteed even though both $\mathfrak{L}_{\mathrm{DL}}$ and $\mathfrak{L}_{\mathrm{CS}}$ are *sound* (Barwise & Seligman 1997). Consequently, inferences passed along the path from $\boldsymbol{Inf}_{\mathrm{CS}}$ to $\boldsymbol{Inf}_{\mathrm{DL}}$ might not be reliable. This is solved by considering only a subset of the tokens of which the mapping can be performed in a meaningful way and the *surjectiveness* is achieved by constructing images for tokens that do not have correspondences.

A direct consequence of applying constraint propagation is the reduction of the original CSP, variable domains pruned and inconsistent values removed. If the original CSP is satisfiable (i.e. no variable is associated with an empty domain), with the help of constraint solvers, it is possible to obtain and leverage two kinds of sequent (ordering) to facilitate a sound *infermorphism*, which are based on two observations. Firstly, DL-based systems can specify subsumption relationships among concepts via the "told" mechanism. For instance, with RACER (Haarslev & Möller 2003), one can specify concept A to be subsumed by concept B as: (implies A B). Other DL-based systems either have similar operators or provide mechanisms to fulfil such functionalities. Secondly, because constraints can be considered as sets of *k-compound labels* giving the legal values that constrained variables can take simultaneously (Tsang 1993), an inclusion ordering among different sets of *k-compound labels* can be established and manipulated. Hence, by associating DL-based (existing or purposely introduced) atomic concepts with sets of *k*-compound labels from constraint solvers, mappings between sequents can be established between DL- and constraint-based systems.

Step-by-step fusion of *Inf*$_{\mathrm{DL}}$ and *Inf*$_{\mathrm{CS}}$

Step one: fragmenting heterogeneous problem We fragment any $\mathcal{DL}(\mathrm{D})/\mathrm{S}$-concepts into DL-oriented components π_{DL} and CSP-oriented ones π_{CSP}. Since an *infermorphism* f : ***Inf***$_{\mathrm{CS}} \leftrightarrows$ ***Inf***$_{\mathrm{DL}}$ is sought after, new atomic concepts are introduced into π_{DL} to ensure a surjective mapping between $\mathfrak{L}_{\mathrm{DL}}$ and $\mathfrak{L}_{\mathrm{CS}}$.

Step two: determining the local logics Within the context of fusing DL- and constraint-based inferences, Def. 4 can be made more specific as:
DL-based system is a three-tuple where M is the set of all axioms and assertions, i.e. TBox$\cup$ABox, σ is materialised by the underlying tableau-based algorithm classifying a set of axioms and/or assertions to a category in $C = \{\top, \bot, \mathtt{unknown}\}$.
Constraint solver is a three-tuple where M is the union of the set of constrained variables ($v_{i\in I}$) with respective domain ($\mathfrak{D}_{i\in I}$) and the set of all constraints defined over $v_{i\in I}$, σ is the set of constraint propagation and searching algorithms that determines whether a subset of M is satisfiable or not, and $C = \{\mathtt{yes}, \mathtt{no}\}$.

It is evident that both DL systems and constraint solvers are based on *admissible knowledge models* and determine the corresponding local logics. For instance, after consistence check, the classification of the knowledge model of a DL-based system is $(\mathcal{A}, \mathcal{T}, \models_{\mathrm{ins}})$ where $\models_{\mathrm{ins}}$ is the instantiation relationship between individuals from ABox and concepts from TBox. Clearly, such a classification determines a local logic whose theory is equivalent to $(\mathcal{T}, \sqsubseteq)$ where $\sqsubseteq$ stands for the subsumption relationship among concepts.

Similarly, after fully propagating all the constraints, the knowledge model of a constraint solver can be regarded as a classification $(\mathfrak{D}^k, \mathfrak{C}, \lambda)$ where $\mathfrak{D}^k$ is the Cartesian product of $\mathfrak{D}_i$ specifying the set of possible combinations of values of all constrained variables, $\mathfrak{C}$ is the set of all constraints and λ assigns $\delta \in \mathfrak{D}^k$ to variables in $c \in \mathfrak{C}$. Ideally, after propagating all constraints, a full path consistency (Mohr & Henderson 1986) can be achieved and thus every $\delta \in \mathfrak{D}^k$ satisfies $\mathfrak{C}$. It is evident that classification $(\mathfrak{D}^k, \mathfrak{C}, \lambda)$ determines a local logic whose theory is $(\mathfrak{C}, \preceq)$ where $\preceq$ is an operator of which if the set of constraints on the left hand side holds those on the right hand side holds as well.

Step three: establishing *linkages* *Linkages* are established between atomic concepts introduced to emulate numeric constraints on one hand and sets of *k-compound labels* on the other. Meanwhile, interpreting $\{\mathtt{yes}, \mathtt{no}\}$ in $\{\top, \bot, \mathtt{unknown}\}$ is straightforward.

Step four and Step five: solving the problem π_{DL} and π_{CSP} are reasoned about using DL-based systems and constraint solvers respectively. Inferences are moved via the *infermorphism channel* that ensures a mutual understanding between different systems. Results of ***Inf***$_{\mathrm{DL}}[\Pi]$ are presented as the solution to the heterogeneous problem.

Conclusions

We presented a formalisation of inferential systems, and used such a formal representation to capture the fusion of heterogeneous inferences. *Concrete inferential systems* consist of underlying knowledge models, classification labels and reasoning algorithms of which the sum of the latter two is referred to as *abstract inferential systems*. Fusion of inference is achieved through morphism between the local logics of inferential systems (logically equivalent to the underlying knowledge models of inferential systems) and the interpretation of those classification labels of the source systems in the target ones.

The advantages of founding *inference fusion* upon the *information flow* framework can be seen from the potential of an automated treatments for integrating distributed reasoning systems. Part of such automation has been explored with respect to a particular instance of *inference fusion*—the integration of DL-based taxonomic reasoning and constraint satisfaction problems—by way of an example. The integration of other inferential systems is under consideration.

Acknowledgements

This work is supported by the UK Engineering and Physical Sciences Research Council (EPSRC) under the AKT IRC grant GR/N15764.

References

Barwise, J., and Seligman, J. 1997. *Informaiton Flow: The Logic of Distributed Systems*. Cambridge University Press.

Berners-Lee, T.; Hendler, J.; and Lassila, O. 2001. The Semantic Web. *Scientific American* 28–37.

Brachman, R.; Fikes, R.; and Levesque, H. 1983. Krypton: A functional approach to knowledge representation. *IEEE Computer* 16(10):67–73.

Christoph Ranze, K.; Scholz, T.; Wagner, T.; Günter, A.; Herzog, O.; Hollmann, O.; Schlieder, C.; and Arlt, V. 2002. A structure based configuration tool: Drive solution designer-DSD. In *AAAI/IAAI-2002*, 845–852.

Doyle, J., and Patil, R. S. 1991. Two theses of knowledge representation: language restrictions, taxonomic classification, and the utility of representation services. *Artificial Intelligence* 48:261–297.

Haarslev, V., and Möller, R. 2003. *RACER User's Guide and Reference Manual Version 1.7.7.* University of Hamburg, Computer Science Department.

Horrocks, I., and Sattler, U. 2001. Ontology Reasoning in the $\mathcal{SHOQ}(\mathrm{D})$ Description Logic. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI'01)*, 199–204. Morgan Kaufmann.

Hu, B.; Arana, I.; and Compatangelo, E. 2003. Facilitating dl-based hybrid reasoning with *Inference Fusion*. *Knowledge-Based Systems* 16(5-6):253–260.

Mohr, R., and Henderson, T. C. 1986. Arc and path consistency revisited. *Artificial Intelligence* 28:225–233.

Tsang, E. P. K. 1993. *Foundations of Constraint Satisfaction*. Academic Press.