An Expressive Efficient Representation: Bridging a Gap between NLP* and KR**

Jana Z. Sukkarieh

University of Oxford, Oxford, England
jana.sukkarieh@clg.ox.ac.uk

Abstract. We do not know how humans reason, whether they reason using natural language (NL) or not and we are not interested in proving or disproving such a proposition. Nonetheless, it seems that a very expressive transparent medium humans communicate with, state their problems in and justify how they solve these problems is NL. Hence, we wished to use NL as a Knowledge Representation(KR) in NL knowledge-based (KB) sytems. However, NL is full of ambiguities. In addition, there are syntactic and semantic processing complexities associated with NL. Hence, we consider a quasi-NL KR with a tractable inference relation. We believe that such a representation bridges the gap between an expressive semantic representation (SR) sought by the Natural Language Processing (NLP) community and an efficient KR sought by the KR community. In addition to being a KR, we use the quasi-NL language as a SR for a subset of English that it defines. Also, it is capable of a general-purpose domain-independent inference component which is, according to semanticists, all what it takes to test a semantic theory in any NLP system. This paper gives only a flavour for this quasi-NL KR and its capabilities (for a detailed study see [14]).

1 Introduction

The main objection against a semantic representation or a Knowledge representation is that they need experts to understand[1]. Non-experts communicate via a natural language (usually) and more or less they understand each other while performing a lot of reasoning. Nevertheless, for a long time, the KR community dismissed the idea that NL can be a KR. That's because NL can be very ambiguous and there are syntactic and semantic processing complexities associated with NL. Recently, researchers started looking at this issue again. Possibly, it has to do with the NL Processing community making some progress in terms of processing and handling ambiguity and the KR community realising that a lot of knowledge is already 'coded' in NL and one should reconsider the way they handle expressivity and ambiguity to make some advances on this front. We have chosen one of these KR, namely, [8], as a starting point to build a KR system using as input a simplified NL that this KR defines. One of the interesting things about this system is that we are exploring a novel meaning representation for English that no one has incorporated in an NLP system before and building a system that touches upon most of the areas of NLP (even if it is restricted). We extended the basic notion and extended its proof theory for it to allow deductive inferences that semanticists agree that it is the best test for any NLP semantic capacity (see FraCas project[2]). Before we started our work, no one built an automatic system that sanctions the FraCas deductive inferences. By

* Natural Language Processing.
** Knowledge Representation.
[1] Though expert systems tend now to use more friendly representations, the latter still need experts.
[2] http://www.cogsci.ed.ac.uk/~fracas/

extending the KR we extended the subset of English that this KR defines. In the following section we give a description of the NL-like KR, McLogic (related articles [16], [15]). In section 3 and 4 we give an idea of the inference task and the controlled English that McLogic defines, respectively (see details in [14], [13], respectively). We conclude by summarising and emphasising the practical implications of our work.

2 NL-Like KR: McLogic

McLogic is an extension of a Knowledge Representation defined by McAllester et al [8], [9]. Other NL-like Knowledge representations are described in [5], [10] and [7], [6], [11] but McAllester et. al have a tractable inference relation. For McLogic NL-like means it is easy to read, natural-looking 'word' order [3]. The basic form of McLogic, we call it McLogic$_0$, and some extensions built on top of it are presented next.

2.1 The Original Framework: McLogic$_0$

McLogic$_0$ has two logical notions called class expressions and formulae. In the following, we define the syntax of McLogic$_0$ and along with it we consider examples and their corresponding denotations. For the sake of clarity, we write V(e) to mean the denotation of e.

Building Blocks Table 1 summarises the syntax of McLogic$_0$. The building unit for sentences or formulae in McLogic$_0$ is a class expression denoting a set. First, **the constant symbols**, *Pooh*, *Piglet*, *Chris − Robin* are class expressions that denote singleton sets consisting of the entities Pooh, Piglet and Chris-Robin, respectively. Second, the monadic predicate symbols like *bear*, *pig*, *hurt*, *cry* and *laugh* are all **monadic class expressions** that denote sets of entities that are bears, pigs or are hurt, that cry or laugh respectively. Third, expressions of the form **(R (some s))** and **(R (every s))** where **R** is a binary relation and **s** is a class expression such as (*climb*(*some tree*)) and (*sting*(*every bear*)), where

$$\mathrm{V}((climb\ (some\ tree))) = \{y \mid \exists\, x.x \in tree \wedge climb(y,x)\}$$

and

$$\mathrm{V}((sting\ (every\ bear))) = \{y \mid \forall\, x.x \in bear \longrightarrow sting(y,x)\}.$$

Fourth, a class expression can be **a variable symbol** that denotes a singleton set, that is, it varies over entities in the domain. Finally, a class expression can be **a lambda expression** of the form $\lambda\, x.\phi(x)$, where $\phi(x)$ is a formula and x is a variable symbol and V($\lambda\, x.\phi(x)$)= $\{d \mid \phi(d)\ is\ true\}$. An example of a lambda expression will be given in the next section.

Well-Formed Sentences A sentence in McLogic$_0$ is either:

- an atomic formula of the form (*every s w*) or (*some s w*) where *s* and *w* are class expressions, for example, (*every athlete energetic*),
- or a negation of these forms, for example, *not*(*every athlete energetic*),
- or a boolean combination of formulae such as (*every athlete healthy*) *and* (*some researcher healthy*).

[3] with the exception of Lambda expressions which are not very English as you will see below!

Table 1. The syntax of McLogic$_0$. R is a binary relation symbol, s and w are class expressions, x is a variable and $\phi(x)$ is a formula.

Class Expressions	**Example**
a constant symbol	c, *Pooh*
a monadic predicate symbol	student
$(R(some\ s))$	$(climb(some\ tree))$
$(R(every\ s))$	$(climb(every\ tree))$
a variable symbol	x
$\lambda\, x.\phi(x)$	$\lambda\, x.(x\ (likes\ (Pooh))$
A Well-Formed formula	**Example**
$(every\ s\ w)$	$(every\ athlete\ energetic)$
$(some\ s\ w)$	$(some\ athlete\ energetic)$
Negation of a formula	$not(some\ athlete\ energetic)$
Boolean combinations of formulae	$(every\ athlete\ energetic)\ and$ $(every\ athlete\ healthy)$

The meaning for each sentence is a condition on its truth value:

$(every\ s\ w)$ is true if and only if (iff) $V(s) \subset V(w)$.

$(some\ s\ w)$ is true iff $V(s) \cap V(w) \neq \varnothing$.

The negation and the Boolean combination of formulae have the usual meaning of logical formulae. Note that since variables and constants denote singleton sets, the formulae $(some\ Pooh\ cry)$ and $(every\ Pooh\ cry)$ are equivalent semantically. For a more natural representation, we can abbreviate this representation to $(Pooh\ cry)$. Now, that we have defined formulae, we can give an example of a lambda expression:

$$\zeta = \lambda\, x.(some/every\ x\ (read(some\ book))),$$

that can be written as

$$\zeta = \lambda\, x.(x\ (read(some\ book))),$$

where

$$\mathrm{V}(\zeta) = \{d \mid (d\ (read(some\ book))) \text{ is true}\}.$$

The above is an informal exposition of the meaning of the constituents.
Before we go on to describe the extensions, it is important to note that McLogic$_0$ is "related to a large family of knowledge representation languages known as concept languages or frame description languages (FDLs)... However, there does not appear to be any simple relationship between the expressive power ... [of $McLogic_0$] and previously studied FDLs "[9]. Consider, for example, translating the formula $(every\ W\ (R(some\ C)))$ into a formula involving the expression of the form $\forall\, R.C$[4] you will find that it seems there is no way to do that.

[4] Recall that an object x is a member of the class expression $\forall\, R.C$ if, for every y such that the relation R holds between x and y, the individual y is in the set denoted by C.

2.2 A richer McLogic

As we did earlier with McLogic$_0$, we provide the syntax of the extension together with some examples of formulae and their associated meanings. The extensions that are needed in this paper are summarised in table 2. The main syntactic and semantic innovation than McLogic$_0$ is the **cardinal class expression** that allows symbolising "quantifiers" other than 'every' and 'some' as it will be made clear later.

Table 2. The syntax of the extensions required. R is a binary relation symbol (R^{-1} is the inverse of a binary relation), R_3 is a 3-ary relation and R_3^{-1} is an inverse of a 3-ary relation, s and t are class expressions, Q_1 and Q_2 can be symbols like *every*, *some*, *at_most_ten*, *most*, and so on, *Mod* is a function that takes a class expression, s, and returns a subset of s.

Class Expressions Example

$s + t$	$happy + man$
$s\$t$	$client\$representative$
$s\#Mod$	$drive\#fast$
$\neg(s)$	$\neg(man)$
$(R_3\ (Q_1\ s)\ (Q_2\ t))$	$(give(some\ student)\ (some\ book))$
$(R^{-1}(Q_1\ s))$	$(borrow^{-1}(some\ student))$
$(R_3^{-1}(Q_1\ s)\ (Q_2\ t))$	$(give^{-1}(some\ book)\ (some\ librarian))$
$Q_2 * s$	$more\_than\_one * man$

Formulae	**Example**
$(N * s\ t)$	$(more\_than\_one * man\ snore)$

2.3 Syntax of McLogic − McLogic$_0$, with Example Denotations

The two logical notions, namely, class expressions and formulae, are the same as McLogic$_0$. In addition to these two, we introduce the notion of a function symbol.

Definition 1. *If f is an n-place function symbol, and $t_1, \ldots, t_n$ are class expressions then $f(t_1, \ldots, t_n)$ is a class expression. There are special function symbols in this language, namely, unary symbols, $\neg$, 2-ary operators, $+$, \$, and $*$.*

We introduce first the 3 operators, $+$, and \$ and $\neg$. The operator, $*$, will be introduced after we define a cardinal class expression.

Definition 2. *Given two class expressions s and t:*

- *$s + t$ is a class expression.*
- *$s\$t$ is a class expression.*
- *$\neg(s)$ is a class expression.*

$s + t$ is defined to be the intersection of the two sets denoted by s and t. $s\$t$ is defined to be the union of the two sets denoted by s and t. Moreover, $\neg(s)$ is defined to be the complement of the set denoted by s.

Example 1. Since *man* and (*eat*(*some apple*)) are class expressions then *man*+(*eat*(*some apple*)), *man*\$(*eat*(*some apple*)), and ¬(*man*) are class expressions. They denote the sets $V(man) \cap V((eat(some\ apple)))$, $V(man) \cup V((eat(some\ apple)))$, and $(V(man))^c$ (complement of the set denoting *man*), respectively.

It is important to mention special unary functions that take a class expression and returns a subset of that class expression. Hence, if we let *mod_func* be one of these functions then $mod\_func(s_1) = s_2$, where s_1 is a class expression and $\mathrm{V}(s_2)$ is a subset of $\mathrm{V}(s_1)$. We will write $s_1 \# mod\_func$ for $mod\_func(s_1)$. For example , let f be an example of such a function, then $f(drive) \subseteq V(drive)$ and $f(man) \subseteq V(man)$ and so on.
Now, we introduce a cardinal class expression.

We group symbols like *most*, *less_than_two*, *more_than_one*, *one*, *two*, *three*, *four*, $\cdots$ under the name cardinal class expressions.

Definition 3. *A cardinal class expression, N, represents a positive integer N and denotes the set* $V(N) = \{X \mid X \text{ is a set of } N \text{ objects}\}$.

Note that a cardinal class expression does not have a meaning independently of entities (in some specified domain). This is motivated by the way one introduces a (abstract) number for a child, it is always associated with objects.
Having defined a cardinal class expression, the operator, $*$ can be defined:

Definition 4. *Given a cardinal class expression N and a class expression that is not a cardinal class expression s, then* $N * s$ *is defined to be* $V(N) \cap V(P(V(s)))$ *where* $P(V(s))$ *is the power set of the denotation of s. In other words,* $N * t$ *is interpreted as* $\{X \mid X \subseteq t \wedge \mid X \mid = N\}$.

The operation, $*$, has the same meaning as $+$, that is, intersection between sets. However, the introduction of a new operator is to emphasize the fact that $*$ defines an intersection between sets of sets of entities and not sets of entities.

Example 2. Given the cardinal class expression *ten* and the non-cardinal class expression *book*, we can form $ten * book$ and this denotes the set of all sets that consist of ten books.

Introducing the operator, $*$, and the class expression $N * s$ allows us to introduce the class expression $(R\ (N * s))$ where R is a binary relation or the inverse of a binary relation. For the sake of presentation, we define an inverse of a binary relation:

Definition 5. *Given a binary relation R, the inverse,* R^{-1}*, of R is defined as such:* $d\ R\ d'$ *iff* $d'\ R^{-1}\ d$.

Examples can be $(borrow(ten{*}book))$, $(buy(more\_than\_one{*}mug))$ or $(buy^{-1}(john))$. To stick to NL, we can use *boughtby* for buy^{-1} and so on. Being class expressions, the last 3 examples will denote sets: a set of entities that borrow(ed) ten books, a set of entities that buy (bought) more than one mug and a set of entities that were bought by john. Hence, the definition:

Definition 6. *A class expression of the form* $(R(N * s))$ *denotes the set* $\{x \mid \exists\, y \in N * s \wedge \forall\, i.(i \in y \longleftrightarrow xRi)\}$.

Moreover, the introduction of $*$ and $N * s$ allows the introduction of a formula of the form $(N * s\ t)$.

Example 3. Given the class expressions *two*, *man*, *old* and (*borrow* (*more_than_one*$*$ *book*)), some of the formulae we can form are: $(two * man\ old)$, $(two * old\ man)$, $(two * man\ (borrow(more\_than\_one * book)))$

We want the above formulae to be true if and only if 'two men are old', 'two old (entities) are men' and 'two men borrow(ed) more than one book', respectively. The truth conditions for a formula of the form $(N * s\ t)$ are given as follows:

Definition 7. *$(N * s\ t)$ is true if and only if some element in $N * s$ is a subset of t. In other words, there exists a set $X \subseteq s$ that has N elements such that $X \subseteq t$ or for short N s's are t's.*

The last extension in this paper, namely, a class expression with a 3-ary relation, will be introduced in what follows:

Definition 8. *A class expression with a 3-ary relation is of the form:*

- $(R(some\ s)\ (every\ t))$,
- $(R(some\ s)\ (some\ t))$,
- $(R(every\ s)\ (every\ t))$, *or*
- $(R(every\ s)\ (every\ t))$, ...

In a more general way :

$(R(Q_1\ s)\ (Q_2\ t))$,

$(R(Q_3 * s)\ (Q_4 * t))$,

$(R(Q_1\ s)\ (Q_4 * t))$,

$(R(Q_3 * s)\ (Q_2\ t))$,

where, Q_i for $i = 1, 2$ is either some or every, Q_i for $i = 3, 4$ are cardinal class expressions, R is a 3-ary relation or an inverse of a 3-ary relation, s and t are non-cardinal class expressions.

Examples of such class expressions can be :

$(give(mary)\ (some\ book))$,

$(hand(some\ student)\ (some\ parcel))$,

$(send(more\_than\_one * flower)\ (two * teacher))$.

It is natural that we make these class expressions denote entities that 'give mary some book', that 'hand(ed) some student some parcel' and that 'send more than one flower to two teachers', respectively. In particular, we define the following:

$\mathrm{V}((R(some\ s)\ (every\ t))) = \{y \mid \exists\, x \in s. \forall\, z. z \in t \longrightarrow \langle y, z, x\rangle \in R\}$.
$\mathrm{V}((R(some\ s)\ (some\ t))) = \{y \mid \exists\, x \in s \wedge \exists\, z \in t \wedge \langle y, z, x\rangle \in R\}$.
$\mathrm{V}((R(every\ s)\ (every\ t))) = \{y \mid \forall\, x. x \in s \longrightarrow \forall\, z. z \in t \longrightarrow \langle y, z, x\rangle \in R\}$.
$\mathrm{V}((R(every\ s)\ (some\ t))) = \{y \mid \forall\, x. x \in s \longrightarrow \exists\, z. z \in t \wedge \langle y, z, x\rangle \in R\}$.
To allow for the change of quantifiers, we consider the form

$(R(Q_1\ s)\ (Q_2\ t))$

whose denotation is

$\{y \mid \exists\, X. X \subset s \wedge \exists\, Z. Z \subset t$ such that $\forall\, x. \forall\, y. x \in X \wedge z \in Z \longrightarrow \langle y, z, x\rangle \in R\}$,

where the cardinality of X and Z depend on Q_1 and Q_2 respectively. Basically, we are just saying that $(R(Q_1\ s)\ (Q_2\ t))$ is the set of elements that relate, through R, with elements in s and elements in t and that the number of elements of s and t in concern depend on the quantifiers Q_1 and Q_2, respectively. For completeness, we define the inverse of a 3-ary relation and give an example:

Definition 9. *Given a 3-ary relation R, the inverse, R^{-1}, of R is defined as:*

$\langle Arg_1, Arg_2, Arg_3\rangle \in R$ *iff* $\langle Arg_2, Arg_3, Arg_1\rangle \in R^{-1}$

For example, V(($give^{-1}$(*some student*) (*john*))) is:

$\{y \mid \exists\, x \in student.\langle y, john, x\rangle \in give^{-1}\} =$

$\{y \mid \exists\, x \in student.\langle x, y, john\rangle \in give\}.$

We have described above the syntax of the extension of McLogic_0, and gave example denotations. As we said earlier, McLogic_0 together with the extensions will be called McLogic in this paper. To see a formal semantics for McLogic you can consult [14]. Having described the logic we go on to describe the inferences McLogic sanctions.

3 The Inference Task McLogic Sanctions

We are concerned with NL inferences but not with implicatures nor suppositions. Moreover, we do not deal with defeasible reasoning, abductive or inductive reasoning and so on. In our work [14], we focus on deductive (valid) inferences that depend on the properties of NL constructs. Entailments from an utterance U, or several utterances U_i that seem "natural", in other words, that people do entail when they hear U. For example, in the following, D_1 are deduced from Scenario S_1:

– S_1:
(1) a. some cat sat on some mat.
b. The cat has whiskers.
⇓
D_1: some cat exists,
some mat exists,
some cat sat on some mat,
some cat has whiskers (cat_1 has whiskers),
some whiskers exist.
⇓
whiskers sat on some mat

We define a **structurally-based inference** to be one that depends on the specific semantic properties of the syntactic categories of sentences in NL. For example,

– S_2: most cats are feline animals.
⇓
D_2: most cats are feline,
most cats are animals.

– S_3 Smith and Jones signed the contract.
⇓
D_3: Smith signed the contract.

D_2 depend on the monotonicity properties of generalised quantifiers and D_3 on those of conjoined Noun Phrases - among other classes that the FraCas deal with.

3.1 Inference Set

The proof theory is specified with a set of deductive inference rules with their contrapositives. The original inference set, which corresponds to McLogic_0 consists of 32 inference rules. We added rules that correspond to the extensions and that are induced by the structurally-based inferences under consideration. In the following, we list some of the rules [5]. We assume these rules are clear and the soundness of

[5] The numbers are not in sequence as to give an idea of the rules added.

each rule can be easily shown against the formal semantics of the logic (consult [14] for a formal semantics).

(7) $(every\ C\ C)$
(19) $(every\ C\ C\$W)$
(20) $(every\ W\ C\$W)$
(21) $(every\ C+W\ C)$
(22) $(every\ C+W\ W)$
(38) $(every\ C\#W\ C)$

(10) $\frac{(some\ C\ exists)}{(some\ C\ C)}$
(11) $\frac{(some\ C\ W)}{(some\ C\ exists)}$
(12) $\frac{(some\ C\ W)}{(some\ W\ C)}$
(13) $\frac{(every\ C\ W),(every\ W\ Z)}{(every\ C\ Z)}$
(14) $\frac{(every\ C\ Z),(some\ C\ W)}{(some\ Z\ W)}$
(15) $\frac{(some\ C\ W),(at\_most\_one\ C)}{(every\ C\ W)}$
(16) $\frac{(every\ C\ W),(at\_most\_one\ W)}{(at\_most\_one\ C)}$
(17) $\frac{(not(at\_most\_one\ C))}{(some\ C\ exists)}$
(18) $\frac{(some\ C\ exists)}{(not(every\ C\ W))}$
(23) $\frac{(every\ C\ Z),(every\ W\ Z)}{(every\ C\$W\ Z)}$
(24) $\frac{(every\ Z\ C),(every\ Z\ W)}{(every\ Z\ C+W)}$
(25) $\frac{(some\ C\ Z)}{(some\ C+Z\ exists)}$
(26) $\frac{(not(some\ S\ exists))}{(every\ T\ (R(every\ S)))}$
(27) $\frac{(every\ C\ W\$Z),(not(some\ C\ W))}{(every\ C\ Z)}$
(28) $\frac{(every\ C\ W\$Z),(not(some\ C\ Z))}{(every\ C\ W)}$
(29) $\frac{(every\ C\ W)}{(every\ (R(some\ C))\ (R(some\ W)))}$
(30) $\frac{(every\ C\ W)}{(every\ (R(every\ W)),(R(every\ C)))}$
(31) $\frac{(some\ C\ W)}{(every\ (R(every\ C))\ (R(some\ W)))}$
(32) $\frac{(some\ (R(some\ C))\ exists)}{(some\ C\ exists)}$
(35) $\frac{(more\_than\_one*C\ W)}{(more\_than\_one*C+W\ exists)}$
(36) $\frac{(more\_than\_one*C+W\ exists)}{(more\_than\_one*C\ C)}$
(37) $\frac{(N_1*C\ W),(every\ C\ Z)}{(N_1*Z\ W)}$
(39) $\frac{(N_1*C\ W)}{(N_1*W\ C)}$
(40) $\frac{(N_3*C\ W),(every\ C\ Z)}{(N_3*C\ W+Z)}$
(41) $\frac{(D\ W\ Z),(every\ Z\ C)}{(D\ W\ C)}$
(42) $\frac{(at\_most\_N*C\ W))}{(at\_most\_N*C\ W\#Z))}$
(43) $\frac{(at\_most\_N*C\ W))}{(at\_most\_N*C+Z\ W))}$
(44) $\frac{(at\_most\_N*C\ (R(some\ W)))}{(at\_most\_N*C\ (R(some\ Z+W)))}$
(45) $\frac{(at\_most\_N*C\ W))}{(at\_most\_N*C\ Z+W))}$
(46) $\frac{(more\_than\_one*C\ W)}{(some\ C\ W)}$
(47) $\frac{(N_4*C\ Z)}{(N_4*C\$W\ Z)}$

In the above, C, W and Z are class expressions. D is a representation for a determiner that is monotonically increasing on 2nd argument. N_1 belongs to $\{at\_least\_N, N, more\_than\_one\}$ and N_3 belongs to $\{most, more\_than\_one, some(sg), at\_least\_N\}$ and N_4 in
$\{more\_than\_one, N, at\_most\_N, at\_least\_N\}$ where N is a cardinal.
In the following, we show a few examples of structurally-based inferences licensed by the properties of monotonicity of GQs that are sanctioned by the above inference rules. It is enough, in these examples, to consider the representation without knowing how the translation is done nor which English constituents is translated to what. I leave showing how McLogic is used as a typed SR to a different occasion. The fact that McLogic is NL-like makes the representation and the proofs in this suite easy to follow.

3.2 Illustrative Proofs

Examples 1, 3, 4 and 5 are selected from The Fracas Test Suite [1]. The Fracas test suite is the best benchmark we could find to develop a general inference component. "The test suite is a basis for a useful and theory/system-independent semantic tool" [1]. From the illustrative proofs, it is seen what we mean by not using higher-order constructs and using what we call combinators, like "sine", "cosine" instead.

'some' is monotonically increasing on second argument as it is shown in examples 1 and 2.

Argument 1 ***IF*** *Some mammals are four-legged animals* ***THEN*** *Some mammals are four-legged.*

$$\dfrac{\dfrac{(more\_than\_one * mammal\ four-legged + animal)}{(more\_than\_one * four-legged + animal\ mammal)}\,r39 \qquad (every\ four-legged + animal\ four-legged)}{\dfrac{(more\_than\_one * four-legged\ mammal)}{(more\_than\_one * mammal\ four-legged)}\,r39}\,r37$$

Argument 2 ***IF*** *some man gives Mary an expensive car* ***THEN*** *some man gives Mary a car.*

To deal with 3-ary relations, we need to augment rules, like (29), (30) and (31). For this particular case, rule (29) is needed. Hence, we add the rules in table 3. Now,

Table 3. Augmenting rule 29 to cover 3-ary relations with 'some' and 'every' only. C, W, Q are class expressions (not cardinal ones nor time class expressions).

(3-ary_1) $\frac{(every\ C\ W)}{(every\ (R(some\ Q)\ (some\ C))\ (R(some\ Q)(some\ W)))}$

(3-ary_2) $\frac{(every\ C\ W)}{(every\ (R(every\ Q)\ (some\ C))\ (R(every\ Q)\ (some\ W)))}$

(3-ary_3) $\frac{(every\ C\ W)}{(every\ (R(some\ C)\ (some\ Q))\ (R(some\ W)\ (some\ Q)))}$

(3-ary_4) $\frac{(every\ C\ W)}{(every\ (R(some\ C)\ (every\ Q))\ (R(some\ W)\ (every\ Q)))}$

we can provide the proof of argument 2:

SINCE $(every\ expensive + car\ car)$ THEN
$(every\ (give(Mary)\ (some\ expensive + car))\ (give(Mary)\ (some\ car)))$ using rule 3-ary_1. MOREOVER, SINCE
$(some\ man\ (give(Mary)\ (some\ expensive + car)))$ THEN
$(some\ (give(Mary)\ (some\ expensive + car))\ man)$ using rule 12. THE TWO CONCLUSIONS IMPLY $(some\ (give(Mary)\ (some\ car))\ man)$ using rule 14. THE LAST CONCLUSION IMPLIES
$(some\ man\ (give(Mary)\ (some\ car)))$.

The determiner 'some' is monotonically increasing on first argument. Using rule 14 together with $(every\ irish + delegate\ delegate)$ justify the required conclusion in argument 3.

Argument 3 ***IF*** *Some Irish delegate snores,* ***THEN*** *Some delegate snores.*

'every' is monotonically decreasing on first argument as in the following:

Argument 4 ***IF*** *every resident of the North American continent travels freely within Europe* ***AND*** *every canadian resident is a resident of the North American continent* ***THEN*** *every canadian resident travels freely within Europe.*

The transitivity rule (rule 13) proves the conclusion of argument 4.

'At most N' is monotonically decreasing on second argument as it is shown in example 5. Rule 42 proves this property for 'at most ten':.

Argument 5 ***IF*** *At most ten commissioners drive,* ***THEN*** *At most ten commissioners drive slowly.*

The inference rules that McLogic is equipped with sanction all the examples given in the Fracas test suite licensed by the properties of GQs. Here, we have only included some of them. The following examples are given in [4] and we show that their validity is easily shown by our proof system.

Argument 6 ***IF*** *no teacher ran* ***THEN*** *no tall teacher ran yesterday. Alternatively, in McLogic,* ***IF*** $not(some\ teacher\ run)$ ***THEN*** $not(some\ tall+teacher\ run\#yesterday)$.

(*every tall+teacher teacher*) *using rule 22. This together with not*(*some teacher run*) *imply not*(*some run tall + teacher*). *Moreover,* (*every run#yesterday run*) *using rule 38. Using rule 14, the last two results justify not*(*some run#yesterday tall + teacher*). *The contrapositive of rule 12 gives the required result.*

Argument 7 ***IF*** *every teacher ran yesterday* ***THEN*** *every tall teacher ran. Alternatively, in McLogic,* ***IF*** (*every teacher run#yesterday*) ***THEN*** (*every tall + teacher run*).

Using the transitivity rule 13 for the premise together with (*every run#yesterday run*) *yields* (*every teacher run*). *Again the transitivity rule together with* (*every tall + teacher teacher*) *justifies the conclusion.*

Argument 8 ***IF*** *every student smiled* ***AND*** *no student who smiled walked* ***THEN*** *no student walked.*

Since (*every student student*) *(rule 7) and* (*every student smile*) *then* (*every student student+ smile*) *using rule 24. Using acontrapositive of rule 14, the last result together with not*(*some student + smile walk*) *justify not*(*some student walk*).

Table 4. Again rules similar to rule 29 that cover 3-ary relations but with 'some', 'every' and cardinal class expressions. C, W, Q are class expressions (not cardinal ones nor time class expressions) and N is a cardinal class expression.

(3-ary_5) $\frac{(every\ C\ W)}{(every\ (R(some\ Q)\ (N*C))\ (R(some\ Q)(N*W)))}$

(3-ary_6) $\frac{(every\ C\ W)}{(every\ (R(every\ Q)\ (N*C))\ (R(every\ Q)\ (N*W)))}$

(3-ary_7) $\frac{(every\ C\ W)}{(every\ (R(N*Q)\ (N*C))\ (R(N*Q)\ (N*W)))}$

(3-ary_8) $\frac{(every\ C\ W)}{(every\ (R(N*C)\ (some\ Q))\ (R(N*W)\ (some\ Q)))}$

(3-ary_9) $\frac{(every\ C\ W)}{(every\ (R(N*C)\ (every\ Q))\ (R(N*W)\ (every\ Q)))}$

(3-ary_10) $\frac{(every\ C\ W)}{(every\ (R(N*C)\ (N*Q))\ (R(N*W)\ (N*Q)))}$

Fyodorov et.al use similar rules. rule 7 is what they call reflexivity rule, rule 24 is what they call conjunction rule. We do not have monotonicity rules as such because monotonicity can be proved through other rules. The last example, they have is the following:

Argument 9 ***IF*** *exactly four tall boys walked* ***AND*** *at most four boys walked* ***THEN*** *exactly four boys walked*

Assuming that 'exactly C Nom VP' to be semantically equivalent to 'at least C Nom VP' and 'at most C Nom VP', where C is a cardinal, the argument is reduced to showing that 'at least four boys walked'. Given (*at_least_four* $*$ *tall + boy walk*) and (*every tall + boy boy*) then (*at_least_four* $*$ *boy walk*) using rule 37.

In all the examples above, combining 'determiners' other than 'some' and 'every' is minimal. For example, the argument

Argument 10 ***IF*** *some managers own at least two black cars,* ***THEN*** *some managers own at least two cars.*

is true but the proof won't go through unless we introduce another rule like (29) but with other 'determiners', namely: $\frac{(every\ P\ Q)}{(every\ (R(N*P))\ (R(N*Q)))}$. This, as well, makes it necessary to account for different 'determiners' in case of a 3-ary relation. The rules required are in table 4 above.

Similar augmentations should occur for rules 30 and 31, whether a binary relation accounting for 'determiners' other than 'some' and 'every' or accounting for a 3-ary relation.
It is important to say that these inferences are licensed independently of the different scope possibilities of quantifiers. For instance, a property of 'every' licenses the inference 'every man likes a woman' from the sentence, S, 'every man likes a beautiful woman', independently of any interpretation given to S. We described the logic and the inferences it allows. In the next section, for the purpose of completeness only, we describe, very briefly, how the logic and the inferences fit into the picture of defining a computer processable controlled English.

4 CLIP: The Controlled English McLogic defines

Definition 10. ***CLIP*** *is a sublanguage of English with the following properties:*

- *It is syntactically and semantically* ***processable by a computer***
- *Each sentence in CLIP has* ***a well-formed translation in McLogic****.*
- ***The ambiguities in a sentence are controlled in a way that the interpretation of that sentence allow inferences required in FraCas D16 [1]***
- *The* ***vocabulary*** *is controlled only as far as the syntactic category.*

The word CLIP implicitly 'clips' a part of the 'whole', that is, dialect or sublanguage not full English. Here is an example:

> **Calvin:** Susie understands some comic books. Many comic books deal with serious issues. All superheroes face tough social dilemmas. It is not true that a comic book is an escapist fantasy. Every comic book is a sophisticated social critique.
> **Hobbes:** Most comic books are incredibly stupid. Every character conveys a spoken or graphic ethical message to the reader before some evil spirit wins and rules.

McLogic_0 is the basic building block for CLIP. To start with, an English sentence belongs to CLIP if, and only if, it has a well-formed translation in McLogic_0. Further, we extended McLogic_0 to account for more English constituents motivated by the structurally-based inferences in the FraCas test suite. Inferences with their corresponding properties, premises and conclusion add to the expressivity of the dialect. To emphasize the above idea, we consider some kind of recursive view:

Base Case: McLogic_0
Recursive Step: $McLogic_n$ depends on $McLogic_{n-1}$

However, it is not an accumulative one-way hierarchy of languages since CLIP and the reasoning task motivates the extension.

5 Conclusion

We believe that the NLP community and the KR community seek common goals, namely, representing knowledge and reasoning about such knowledge. Nonetheless, the two communities have difficult-to-meet desiderata, namely, expressivity and taking information in context into account for a semantic representation and efficient reasoning for a KR. Finding one representation that is capable of both is still a challenge. We argue that an NL-like KR may bring that gap closer. Trying to achieve our aim of an NL-based formal KR led to an interesting inquiry into (not in order of importance):

- a novel meaning representation for English
- a KR in relation to the following:
  Given English utterances $U_1, ..., U_n$ and an English utterance C, a machine has to decide whether C follows from $U_1 \wedge ... \wedge U_n$.
- A way for an NL expert Knowledge-Based System (KBS) to provide a clear justification for the line of reasoning used to draw its conclusions.

An experimental system has been developed. The system has been tested (as a guideline for its development) on two substantial examples: the first is a well-known test case for theorem provers ('Schubert's Steamroller') [12] and the second is a well-known example from the Z programme specification literature ('Wing's library problem' [2]). Further, our aim was a general reasoning component that could handle a test suite which consists of a set of structurally-based deductions; that is, deductions licensed by specific properties of English constituents, made independently of the domain. As we mentioned earlier, as a guideline for structurally-based inferences, we have used the FraCas test suite. Hence, we extended, in a formal manner, McLogic_0 and accordingly the KR system that takes a restricted but still powerful sublanguage of English as input. As far as we know, no one has provided a deductive computational engine covering all the types of inference illustrated or listed in the FraCas test suite or incorporated McLogic_0 (nor even extending it) into an NLP system before. The practical value of the study can be seen, at least, in the following way:

1. For a knowledge engineer, having a NL-like (i.e. transparent) KR makes it easier to debug any KB reasoning system.
2. Since McLogic's inference set is domain-independent and aims to be rich enough to be a test for the semantic capacity of any NLP system then McLogic could be used in an advanced question-answer sytem in any domain.

References

1. Cooper, R. and Crouch, D. and van Eijck, J. and Fox, C. and van Genabith, J. and Jaspars, J. and Kamp, H. and Milward, D. and Pinkal, M. and Poesio, M. and Pulman, S.: The Fracas Consortium, Deliverable D16. With additional contributions from Briscoe, T. and Maier, H. and Konrad, K. (1996)
2. Diller A.: Z An Introduction to Formal Methods. Wing's Library Problem. John Wiley and Sons Ltd. (1994)
3. Dowty, D.R. and Wall, R.E. and Peters, S.: Introduction to Montague Semantics. D. Reidel Publishing Co., Boston. (1981)
4. Fyodorov, Y. and Winter, Y. and Francez, N.: A Natural Logic inference system. Inference in Computational Semantics. (2000)
5. Hwang C. H. and Schubert L.K.: Episodic Logic: A comprehensive, Natural Representation for Language Understanding. Minds and Machines. **3 number 4** (1993) 381–419

6. Iwanska, L. M. : Natural Language Processing and Knowledge Representation. Natural Language is a Powerful Knowledge Representation system. MIT press. (2000) 7–64
7. Iwanska L. M.: Natural Language Is A Representational Language. Knowledge Representation Systems based on Natural Language. AAAI Press. (1996) 44–70
8. McAllester, D. and Givan, R.: Natural Language Syntax and First-Order Inference. Artificial Intelligence. **56** (1992)1–20
9. McAllester, D. and Givan, R. and Shalaby, S. : Natural Language Based Inference Procedures Applied to Schubert's Steamroller. AAAI. (1991)
10. Schubert, L.K. and Hwang, C. H.: Natural Language Processing and Knowledge Representation. Episodic Logic Meets Little Red Riding Hood-A comprehensive Natural Representation for Language Understanding. AAAI Press; Cambridge, Mass.: MIT. (2000) 111–174
11. Shapiro S.C. : Natural Language Processing and Knowledge Representation. SNePs: A logic for Natural Language Understanding and Commonsense Reasoning. AAAI Press; Cambridge, Mass.: MIT. (2000) 175–195
12. Stickel M. E.: Schubert's Steamroller Problem: Formulations and Solutions. Journal of Automated Reasoning. **2** (1986) 89–101
13. Sukkarieh, J. Z.: Mind Your Language! Controlled Language for Inference Purposes. To appear in EAMT/CLAW03. Dublin, Ireland. (2003)
14. Sukkarieh, J. Z.: Natural Language for Knowledge Representation. University of Cambridge. http://www.clp.ox.ac.uk/people/staff/jana/jana.htm. (2001)
15. Sukkarieh, J. Z.: Quasi-NL Knowledge Representation for Structurally-Based Inferences. P. Blackburn and M. Kolhase (eds). Proceedings of the Third International Workshop on Inference in Computational Semantics, Siena, Italy. (2001)
16. Sukkarieh, J. Z. and Pulman, S. G.: Computer Processable English and McLogic. H. Bunt et al. (eds) Proceedings of the Third International Workshop on Computational Semantics. Tilburg, The Netherlands. (1999)