dynamic ontology which includes the dynamic relation among proper nouns, except for the names of places.

We refer to this dynamic ontology "Rhizome type ontology" as described by Deleuze & Guattari[6]. We want to develop this ontology and make self-generate meta-data function by consulting new words to Wikipedia's database automatically. Of course, it is impossible that the system could compile Wikipedia's data completely automatically. However, it is more probable that the system compiles data to some degree, and then will be able to help system engineers or administrators, if the data format of Wikipedia became easy to consult automatically.

7 Conclusion

The Media Browser for the Digital Asia Project has the "technology that limits accurate intelligence", as the system can classify texts by focusing on the characteristics of the news items and through a correct understanding of "a proper nouns and time". Our system also has "Architecture that guarantees the transparency" as the system is open to the public and is automatically updated through the synchronization of content from Wikipedia.We believe that the future ontology will be Rhyzome type ontology which supports a technology that limits accurate intelligence, and an architecture that guarantees the transparency.

[6] Gilles Deleuze and F´elix Guattari "Mille Plateaux", Editions de Minuit, 1980

InterOntology. Proceedings of the First Interdisciplinary Ontology Meeting, Tokyo, Japan, 26-27 February 2008, 127-138

16. Remarks on Logic for Process Descriptions in Ontological Reasoning: A Drug Interaction Ontology (DIO) Case Study

Mitsuhiro Okada[1], Barry Smith[2], and Yutaro Sugimoto[1]

[1] Logic Group, Department of Philosophy, Keio University
2-15-45 Mita, Minato-ku, Tokyo, 108-8345, Japan
{mitsu,sugimoto}@abelard.flet.keio.ac.jp
[2] Department of Philosophy and Center of Excellence in Bioinformatics, University at Buffalo
701 Ellicott Street, Buffalo, New York 14203, USA
phismith@buffalo.edu

Abstract. We present some ideas on logical process descriptions, using relations from the DIO (Drug Interaction Ontology) as examples and explaining how these relations can be naturally decomposed in terms of more basic structured logical process descriptions using terms from linear logic. In our view, the process descriptions are able to clarify the usual relational descriptions of DIO. In particular, we discuss the use of logical process descriptions in proving linear logical theorems. Among the types of reasoning supported by DIO one can distinguish both (1) basic reasoning about general structures in reality and (2) the domain-specific reasoning of experts. We here propose a clarification of this important distinction between (realist) reasoning on the basis of an ontology and rule-based inferences on the basis of an expert's view.

1 Introduction

The purpose of this report is to discuss some issues related to the logical way of describing and analyzing processes using linear logic or of some similar logical calculus. We are, in particular, interested in applying such a logical framework to the analysis of certain types of generating processes represented especially in biomedical ontologies. We shall discuss some logical descriptions of the underlying reasoning in DIO (Drug Interaction Ontology [Yoshikawa *et al.* 2003]) as an example. On the linear logical process description level one can distinguish, among the types of reasoning supported by DIO, both (1) certain basic types of reasoning about the reality which DIO represents, and (2) certain domain-specific types of reasoning on the part of experts. This distinction is however not explicitly visible on the original first-order predicate logical representation of DIO. We think that a clarification of the distinction is important, and our contribution to such clarification falls into three parts:

1. [Okada *et al.* 2006] presents a way to transform some primitive relations usually represented in traditional predicate-logical form (or by means of equivalent graph-theoretical artifacts) into more structured process descriptions on the level of linear logic. The drug

interaction relations targeted in this report are process-properties (or certain associated functions) of drugs, involving both drug components and cellular components in which these drugs are active. We demonstrate a way to spell out the primitive relations of DIO in lower-level process terms. We show by means of examples how on this level a generation process can be described more precisely for the analysis of reasoning involved in DIO.

2. As is known from recent advances of logic such as linear logic, substructural logics and some process calculi, the process description level of logic brings a distinction between *consumption-sensitive* and *repeatedly usable resources*. On this level of logical process descriptions, a resource that is repeatedly usable (potentially infinitely reusable) resources in the process of reasoning may be treated as a kind of universal, while the consumable resources are treated as instances in the same reasoning. (Compare the distinction between instances and types in [Smith 2004].) The process description level of logic also offers the facility for describing concurrent subprocesses [Girard 1987,Okada 1998]. Additional expressiveness is hereby provided both by the distinction between the consumption-sensitive instances and the repeatedly usable component names and by the ability to handle concurrent or parallel occurrences of subprocesses, and this added expressiveness offers an enhanced logical tool for automated reasoning. Thus on the one hand it allows and for new kinds of abstract simulations of processes such as drug interactions, which on the other it can potentially allow also for the use of automated theorem proving (i.e., symbolic computation) techniques. To realize this potential will involve addressing a number of logical challenges presented by the computational complexity of the approach. But there already exist various resource-sensitive concurrent logic programming style symbolic theorem provers based on linear logics, that cover the proposed framework of this paper (cf. [Miller 1996], [Hodas *et al.* 1994], and [Banbara *et al.* 1998]).
3. We shall also discuss what kinds of linear logic and similar reasoning should be used in addition to those referred to in the original DIO paper. Some higher levels of reasoning presented in the latter are analyzed and classified using the technical terms of proof theory. For example, some examples of expert reasoning referred to in the DIO paper in discussion of the relation of *inhibition* require non-linear logical modal inferences in addition to the standard process description level. This additional modal operator (related to an expert's reasoning about a quantitative property), allows the domain experts using DIO to treat what is a repeatedly reusable resource on the basic process description level as a partially consumption-sensitive resource in the specific context of reasoning about inhibition effects brought about thorough the concurrent use of different drugs. These kinds of inference rules are additional ones; they do not, in our opinion, belong in the basic inference level of the ontology proper.

The significance and merits of the ontology method have been recognized in a wide range of areas, including biomedical informatics, semantic web engineering and others. Ontologies provide a framework for describing and structuring data in a way which can be shared by many different users and for many different purposes, and also make for more efficient use and for easier accessibility of the data to external users. Current ontology technology is not well-suited to support reasoning about time and process, especially where concurrent processes are involved. The process description level of logic (shared by linear logic, some variants of substructural logic, and process calculi) can to some extent help to fill this gap. Such logics provide the expressive power to deal with parallel or concurrent subprocesses. This added expressiveness can be useful for logical analyses especially when the logical process descriptions are used for analysis of simulations of real-world process on the symbolic level.

2 Linear Logic for Process Descriptions

The process description level of logic rests on a distinction between two kinds of resources: those which are the consumption-sensitive and those which are consumption-non-sensitive. The distinction is hidden in traditional relational logics, either classical, intuitionistic or modal logics. It is well-known that such traditional logics (for example standard first-order predicate logic) can be obtained from linear logic by forgetting the information on the consumption sensibility [Girard 1987,Okada 2004]. As a result, Barwise and Perry once claimed that the traditional logical proof-theory is in fact applicable only to mathematics in the strict sense according to which mathematical facts are defined by the fact that they cannot be consumed [Barwise *et al.* 1983]; a true mathematical lemma still remains true even after it has been used for the proof of a theorem. A mathematical proof (for example, as represented by traditional first-order predicate logic) can be viewed as a proving process description; however, a traditional first-order predicate logical proof represents a very special kind of process, where the use of a resource (e.g., a lemma or an axiom) does not require us to take account of this use as an expense.

On the other hand, in our reasoning about reality, and in logical process descriptions of specific aspects of reality, we often face the situation where the use of some resource require it to be treated as expense. Resources can be consumed. Certainly in the normal circumstances of reasoning the relational queries performed against the logical framework of an ontology (specifically, a biological ontology using a description logic framework) need only a relational answer, the situation is different where consumption-sensitive information processes are needed in reasoning. This is so for example where one would like to use an ontology-based automated reasoning system to create a symbolic abstract simulation embodying a distinction between consumption sensitive and non-sensitive continuants and their effects. In our case study on DIO, we consider some drug components as consumption-sensitive finite resources while treating enzymes (for example those which facilitate the binding of a drug with some cell components) as repeatedly usable resources within the cell.

Linear logic proposed by Girard [Girard 1987] (c.f. [Okada 1998]) and its substructural logic variants and process calculi variants are known to provide a logical framework for such a new situation occurring in describing consumption-sensitive and concurrency sensitive processes.[3] For example, instead of the traditional logical connective $\wedge$ ("and"), linear

[3] There are some other approaches in which the traditional first-order logic is refined in order to capture actions and changes of state. The situation calculus proposed by J. McCarthy[McCarthy *et al.* 1969] is one such approach.

logic provides two different kinds of logical connectives $\otimes$ (the superpositive "and") and & (the selective "and") (c.f. [Okada 2006] for the distinction), where $A \otimes B$ means "A and B hold in parallel (at the same time)" while A & B means "Either A or B can be chosen to hold (as you like) but only one of them at once."

The traditional logical implication "$\rightarrow$" is replaced in linear logic by the linear implication "$\multimap$", where $A \multimap B$ means: "By the consumption of A, B is generated." With the explicit appearance of the resource consumption relation, the conjunction "A and B" naturally yields the co-existence of A and B. We use "comma" ("A, B") to denote the co-existence A and B. We can then introduce a stronger notion of co-existence to express the specific case of molecular binding "$A \otimes B$".[4] [5]

On the other hand, where the amount of a resource of A is (potentially infinite) this is expressed as: $!A$, with the help of the linear logic modal operator !. ($!A$ is such that one can consume it as many times as one wants without any loss.) By using this modal operator ! one can gain back the full logical expressiveness of the traditional logical truth (i.e., eternal or timeless truth) within the framework of linear logic. Hence, linear logic contains the traditional logic (with the help of modal operator) as a modal expression part (c.f. [Girard 1987,Okada 2004]), and linear logic itself can be considered as a refined (or fine-grained) form of the traditional logic, rather than as a logic *different* from the traditional logic.

The linear logical implication $A \multimap B$ means: "by consuming A, B is generated." In other words, when A holds and $A \multimap B$ holds, then B can be obtained at the expense of A. Traditional logical implication does not have to take account of the expense of consumption. In the context of traditional logic, one can conclude from the two premises,

If $0 \leq x$ then $100 \leq f(x)$
$0 \leq x$

to the conclusion:

$100 \leq f(x)$

and also still have the truth of $0 \leq x$. That is to say, the traditional implication $A \rightarrow B$ means that when A holds and $A \multimap B$ holds, then B holds, where A continues to hold even after B is obtained from A and $A \rightarrow B$. (For a precise list of the formal inference rules, see [Girard 1987,Okada 1998].)

The traditional implication may be expressed by the linear implication (the resource-consumptional implication) together with the bang operator !. Thus the traditional $A \rightarrow B$

[4] In the original notation of linear logic, the symbol $\otimes$ is used for the parallel operator; we use a slight different symbolism for it in this paper. See [Okada 1998], [Girard 1995].

[5] The logical inference rule for "," is:

$$\frac{C \vdash A \quad D \vdash B}{(C, D) \vdash (A, B)}$$

may be represented by the linear logical formula: $(!A) \rightarrow B$.[6] [7]

One problem of employing the traditional relational logic may be explained as follows: Assume that a relational expression $Generate(A, B)$ is used for the generation of B by incurring A as expense. When A is given, if $Generate(A, B)$ then B is generated and A is used up. We can then express "by spending a quarter q one gets a candy c", as $Generate(q, c)$ and "by spending a quarter q one gets an ice cream i" as $Generate(q, i)$. If we use predicates $Quarter(x)$, $Condy(x)$, $IceCream(x)$ instead of names for universals, we can specify these generation properties as follows (where we assume that if q is a quarter coin, then q generates a candy c and that if q is a quarter coin, then q generates an icecream i):

$$\forall q(Quarter(q) \rightarrow \exists c(Candy(c) \wedge Generate(q, c)))$$

$$\forall q(Quarter(q) \rightarrow \exists i(IceCream(i) \wedge Generate(q, i)))$$

[8] Now we assume that a single quarter coin q_0 is given, so that $Quarter(q_0)$ holds. Then, following the traditional logic, one could conclude among other things,

$$\exists c \exists i(Candy(c) \wedge IceCream(i))$$

(One has obtained both a candy instance and an ice-cream instance (from the assumption that the single quarter coin q_0 is given).)

Moreover:

$$Quarter(q_0)$$

(One still has the quarter coin q_0.)

This kind of naive use of traditional logic brings a number of inconveniences. In this example, the resource q is consumed when c or i is produced and a single resource q cannot provide both c and i at the same time. One could consider a more precise setting with a triple relation $Generate(q, v, c)$ when v is a vending machine and $Generate(q, v, c)$ represents "a quarter coin q with the vending machine v generates a candy c." With this relation, not

[6] This corresponds to the traditional intuitionistic implication in the sense of Heyting since essentially ! follows the structure of Gödel's S4 modal interpretation (see [Okada 2004]) and the Gödel-Kolmogorov style modification or a S5-based Gödel modal interpretation corresponds to the classical mathematical implication.

[7] The standard rules for the bang-modal operator ! are formulated as follows:

! -left

(dereliction-left) $\dfrac{A, \Gamma \vdash \Delta}{!A, \Gamma \vdash \Delta}$ (contraction-left) $\dfrac{!A, !A, \Gamma \vdash \Delta}{!A, \Gamma \vdash \Delta}$

[8] One could also abbreviate these two propositions as follows:

$$\forall q \in Quarter \; \exists c \in Candy \; (Generate(q, c))$$

$$\forall q \in Quarter \; \exists i \in IceCream \; (Generate(q, i))$$

only the consumption-sensitive instances of quarter coin and candy but also the repeatedly usable vending machine v, can be represented as being involved in the process-relation. (Alternatively, $Generate(q, v, c)$ may also be understood as a representation of a function of the vending machine v.) In linear logic, $Generate(q, v, c)$ tells that at a particular spatio-temporal location, for example, a quarter q may be changed into a candy c at the expense of the quarter coin q by employing a vending machine v. Using the primitive connectives of linear logic this reads: $!(q, !v) \multimap c$.

In this report, we are particularly interested in cases where the same process classes are described or captured via relations between different classes of continuants as occurs in chemical reaction processes studied in chemistry and in the biosciences. For example, a reaction process class P, when instantiated in a cell, involves instances of continuant classes a, b, c in such a way that P is a process of the reaction of a drug element a with enzyme b to produce effect c in the cell. This kind of drug effect is represented in the Drug Interaction Ontology (DIO) in [Yoshikawa *et al.* 2003], which includes the following primitive relations:

- a and b are combined by a reaction to form c
- a facilitates the production of c in an environment in which b co-exists (for example, when b is some enzyme in a certain cell).
- a inhibits the facilitating process P under conditions where b co-exists (for example, in a case of simultaneous use of another drug).

DIO captures these basic relations graphically by means of certain triadic (graph-theoretic) representations. For example, "a facilitates a reaction process producing c giving the co-existence of repeatedly or continuously available environmental resource class b" was represented graphically as a triadic relations abbreviated: $Facilitate(a, b, c)$. Similarly, $Bind(a, b, c)$, $Inhibit(a, b, c)$ were represented as a triadic (ternary) relations in DIO. The goal was to find symbolic prototypes of certain drug interactions. In earlier work carried out jointly between our group and the Konagaya-Yoshikawa group, we formalized the types of relational reasoning used in Yoshikawa-Konagaya papers on DIO using linear logic descriptions of processes and where a symbolic prototype for a series of reactions can be expressed as a linear-logical proof [Okada *et al.* 2006]. We also compared this type of symbolic logical inference method to find prototypes for the simulation of interactions with other simulation methods (e.g., [Yamaguchi *et al.* 2007]).

Linear logic (and certain related logical calculi such as the π-calculus [Milner 1999]) can also be used to represent reactions involving parallel sub-processes (the concurrent parts of sub-processes). Here again we can use the linear-logical process description framework as a means to create symbolic simulation prototypes. If simulate concurrent processes in terms of proofs, then a proof discovered by a bottom-up proof search can then be interpreted as a simulation prototype on the abstract symbolic level.

For example, the triadic basic relation $Facilitate(a, b, c)$ of DIO can be used in representations of consumption processes where an input is consumed to generate an output. We formulate the consumption relation by means of a linear-logical consumption relation as in [Okada *et al.* 2006], using the following abbreviations: a_1 : input, a_2 : repeatedly-available-environmental-resource and a_3 : output. Now the triadic relation of $Facilitate(a_1, a_2, a_3)$ is

linear-logically described as follows:

$$!((a_1, !a_2) \multimap a_3)$$

The following consequence relation then holds

$$!a_2, Facilitate(a_1, a_2, a_3) \vdash (a_1 \multimap a_3), !a_2$$

This means that if there is an environmental resource (for example an enzyme) $!a_2$, and if $Facilitate(a_1, a_2, a_3)$, then

- if a_1 is added, a_3 is generated, (in other words, addition of a_1 generates a_3), and
- $!a_2$ is still available

If

- $!a_2$ exists as an environmental resource; and
- $Facilitate(a_1, a_2, a_3)$ simultaneously holds

Then the state $a_1 \multimap a_3$ which means "if instance a_1 is provided, instance a_3 can be generated" can be deduced via linear-logical deduction. The above conclusion tells us that the state $(a_1 \multimap a_3)$ can be obtained while at the same time the enzyme $!a_2$ still remains and is ready to be used for another (or for concurrent) reactions in the same cell.

To describe biomolecular binding, the $\otimes$ (or "tensor") symbol is used as a connective as in [Okada *et al.* 2006]. The linear-logical expression $a_1 \otimes a_2$ stands for the binding of a_1 and a_2. The linear logical expression $(a_1, a_2) \multimap a_1 \otimes a_2$ means that if a_1 and a_2 exist, and it is known that they will actually bind together, then the consumption of a_1 and a_2 will generate the bound molecule $a_1 \otimes a_2$. Some examples of the specific reactions dealt with by the DIO can be expressed via the linear- logical process descriptions as follows:
Ketoconazole, an anti-fungal drug, is known to be slowly metabolized by $CYP3A4$[9] forming stable complexes. We express this as follows:

$$(Ketoconazole, CYP3A4) \multimap Ketoconazole \otimes CYP3A4$$

This means that if *Ketoconazole* and $CYP3A4$ co-exist, then *Ketoconazole* and $CYP3A4$ can bind together.

The above-mentioned level of primitive-relations used in DIO is represented straightforwardly via the linear logical process descriptions using Horn clause expressions, as explained for example in [Okada 1998]. Linear logic theorem provers (or linear logical programming language), such as Forum [Miller 1996], LLP [Banbara *et al.* 1998], Lolli [Hodas *et al.* 1994], allow any queries using three primitive relations, such as *Facilitate* and *Bind*, to be not only formally expressed but also be examined and analyzed. When the theorem prover succeeds in proving a query, then the linear logical proof itself can

[9] Cytochrome P-450 isoform 3A4, one of the so-called drug metabolizing enzymes existing mainly in the human liver

be identified with a process to reach the state described by the query on the basis of the simulation methodology which sees the linear logical proofs as simulations of processes (c.f. [Okada 1998]).

Concurrent parts of subprocesses on the same "proofs-as-process" paradigm. In our opinion, the resultant expressive power is useful for the use of the proposed logical process description framework to examine logically described queries against ontologically annotated data describing drug interaction processes and to search possible symbolic simulation prototypes logically with the help of the linear logical theorem provers.

3 Inferences Concerning Inhibition Relations in DIO

In DIO-style reasoning, the user sometimes wishes to obtain certain information concerning relations of inhibition within the cell. Suppose for example that the production of a is normally generated by a drug b. But, with the use of another drug c in the same environment, the production of a is inhibited, or in other words, the amount of the production of a is substantially decreased due to the use of c.

To deal with such cases, DIO's authors [Yoshikawa *et al.* 2003] introduced another triadic-relation $Inhibit(a, b, c)$, and informally described how to reason with $Inhibit(a, b, c)$. In former work, we characterized that reasoning on linear logic. We introduce a new modal operator, which is called the quantitative modality, in symbol ∇. [10]

In our former work [Okada *et al.* 2006], we showed that basic reasoning on the inhibition-relation of DIO requires inferences in terms of this additional modality ∇ is a *domain-specific* modality, which falls outside the set of modalities which can be defined in strict linear logical terms. ∇a expresses "the amount of a decreases." In the presence of this quantitative modality ∇, $!a$, for example, is no longer assumed to be a consumption-non-sensitive (potentially infinitely re-usable) resource but is rather treated in such a way that we are required to be taken account of the decreasing effects brought about by expenses incurred along the way; $\nabla!a$ stands for: "$!a$ has an decreasing effect." [11]

The relation $Inhibit(a, b)$ of DIO is formalized as $a \multimap \nabla!b$, using ∇. Using ∇-rules, we can infer the inhibiting relations of DIO. For convenience, we also use the following

[10] This represents our thinking about the inhibiting-effects in the DIO-style reasoning.

∇left

$$\frac{A, \Gamma \vdash \Sigma}{\nabla A, \Gamma \vdash \nabla \Sigma}$$

∇right

$$\frac{\Gamma \vdash \Sigma, !A}{\Gamma \vdash \Sigma, A, \nabla!A}$$

[11] **Modality Rule I:** $\nabla right$

$$\frac{\Gamma \vdash \Sigma, !a}{\Gamma \vdash \Sigma, a, \nabla!a}\ \nabla right$$

An environmental resource $!a$ can be considered as the sum of a and $\nabla!a$. This means that if a part of $!a$ is used in the environmental resource $!a$, then the environmental resource is consumed, and the amount of usable environmental resource !a will decrease. This inference rule will be used to express the basic "inhibition" relation in DIO; a use of $!a$ which results in a product which may inhibit $!a$.

Modality Rule II: $\nabla left$

abbreviations: a_1: drug-1, a_2: enzyme, a_3: product, and a_4: drug-2.

In DIO we reason about inhibition as follows: "if there is an environmental resource $!a_2$ and if $Bind(a, b)$ actually holds, then if a_4 is added, a bound molecule $a_4 \otimes a_2$ together with a decrease in a_2 are generated. (The addition of a_1 and a_4 generates the decreased a_2.)" This reasoning is expressed as $(a_4, a_2) \multimap (a_4 \otimes a_2), !a_2 \vdash Inhibit(a_4, !a_2)$ with the additional ∇ operator. This is deducible in linear logic with the addition of the above-mentioned inference rules on ∇. [12]

The query "May Ketoconazole decrease the amount of CYP3A4 ?" was expressed in the above form in [Okada *et al.* 2006]. Any linear logical proof for the corresponding assertion (see footnote[12] for one such proof) describes a process to realize the state of the query, according to the "proofs-as-processes" interpretation (c.f. [Okada 1998]). [13]

$$Bind(a_4, a_2), !a_2, Facilitate(a_1, a_2, a_3) \vdash Inhibit((a_1, a_4), \nabla a_3)$$

The DIO incorporates reasoning to the effect that if $bind(a, b)$ currently holds, and if there is an environmental resource $!a_2$, and if $Facilitate(a_1, a_2, a_3)$ currently holds, then if a_1 and a_4 are added, a decrease in a_3 is generated (the addition of a_1 and a_4 generates a decreased a_3). In symbols:

$$Bind(a_4, a_2), !a_2, Facilitate(a_1, a_2, a_3) \vdash Inhibit((a_1, a_4), \nabla a_3)$$

where $Bind(a_4, a_2)$ is the abbreviation of $(a_4, a_2) \multimap (a_4 \otimes a_2)$, $Facilitate(a_1, a_2, a_3)$ stands

$$\frac{a, \Gamma \vdash b_1, b_2, ..., b_n}{\nabla a, \Gamma \vdash \nabla b_1, \nabla b_2, ..., \nabla b_n}\ \nabla left$$

Assume it is known that a and Γ produce $b_1, b_2, ...,$ and b_n, then one knows that some decrease of a entails some decrease of each of the products $b_1, ..., b_n$ (with constant availability of Γ).

[12] **Proof**:

$$\dfrac{\dfrac{\dfrac{a_4 \vdash a_4 \quad \dfrac{!a_2 \vdash !a_2}{!a_2 \vdash \nabla!a_2, a_2}\ \nabla right}{a_4, !a_2 \vdash \nabla!a_2, (a_4, a_2)} \qquad a_4 \otimes a_2 \vdash a_4 \otimes a_2}{(a_4, a_2) \multimap (a_4 \otimes a_2), a_4, !a_2 \vdash a_4 \otimes a_2, \nabla!a_2}\ \multimap left}{(a_4, a_2) \multimap (a_4 \otimes a_2), a_4, !a_2 \vdash \nabla!a_2}\ weakening\text{-}right}{(a_4, a_2) \multimap (a_4 \otimes a_2), !a_2 \vdash a_4 \multimap \nabla!a_2}\ \multimap right$$

[13] Here, we can adopt the token symbols a_2, a_4 to the query by the following interpretation.

$a_2 : CYP3A4$
$a_4 : Ketoconazole$

We can see,

$!a_2$ really exists as an environmental resource; and
a_2 and a_4 are actually bound together, namely $Bind(a_4, a_2)$ holds.

Therefore, the two premises hold. Then it can be proved that "a_4 may decrease the amount of a_2".

for $(a_1, !a_2) \multimap a_3$, and $Inhibit((a_1, a_4), a_3)$ stands for $(a_1, a_4) \multimap \triangledown !a_3$.

The DIO query "May Ketoconazole inhibit the generation of APC ?" can also be linear-logically expressed and answered in this way (cf.[Okada *et al.* 2006]).

Note that "non-monotonic reasoning" is used in these inferences regarding the presence in the resource of a_4 (*Ketoconazole* in the above example). The proof of a_3 is replaced by a proof of $\triangledown a_3$ under the assumptions that $Facilitate(a_1, a_2, a_3)$ and giving a_1 with $!a$. This non-monotonicity of the reasoning is one of the essential features of inhibition-related reasoning in DIO. It is well known that the fundamental proof theoretic properties (such as cut-elimination or proof-normalization) are closely related to narrowing the proof-search space for theorem proving.

Although the introduction of the quantitative modality $\triangledown A$ preserves the consistency of the standard linear-logical proof system, it causes heavy additional costs on logical analysis and automated proof search. This is because (1) the additional rule destroys the cut-elimination property (or proof-normalization property), a fundamental proof-theoretic property for efficient proof search, and (2) it destroys the basic model-checking property in model theory because of the non-monotonicity.

On the other hand, the addition of this new modality does not affect reasoning on basic logical process description level, i.e., it does not hamper what we can achieve using the standard (essentially Horn-Clause) formalism of linear logic. Technically speaking, it is a conservative extension of the standard linear logic; if some conclusion without the additional modality is obtained by the use of this new modality, one can obtain the same conclusion with a proof without the use of modality. In our opinion, this linear logical reasoning offers the basic reasoning on processes for ontological considerations, while the non-linear-logical modality rules are only situation-depending auxiliary rules.

This basic reasoning part is in fact a rather simple fragment of linear logic. There exist many linear logical theorem provers and formal proof-tools to support logical analyses and proof-searches for this range of linear logical proofs, such as Forum [Miller 1996], LLP [Banbara *et al.* 1998], Lolli [Hodas *et al.* 1994]. Such logical tools could accommodate the framework proposed in this report.

4 Conclusion

We presented basic ideas on logical process descriptions, with using some basic relations of the DIO (Drug Interaction Ontology) as examples and explaining how the primitive relations could be naturally decomposed to more basic structured logical process descriptions in terms of linear logic. In our opinion, the resultant process descriptions help to clarify the usual relational descriptions of DIO. In particular, we discussed the use of logical process descriptions in linear-logical proofs. On the linear logical process description level one could distinguish, among the types of reasoning allowed by DIO, certain basic reasoning on general and domain-neutral structures and also certain domain-specific types of reasoning carried out by experts. This distinction is not drawn explicitly in the original (first-order predicate logical, or graphical) relational descriptions level of DIO. We think that such a clarification of the distinction is important to distinguish domain-neutral reasoning supported by ontology from the rule-based inferences carried out by domain experts.

5 Acknowledgements

Smith's contributions to this work were funded in part by the National Institutes of Health through the NIH Roadmap for Medical Research, Grant 1 U 54 HG004028.

We would like to express our sincere thanks to Dr. Akihiko Konagaya and Dr. Sumi Yoshikawa, of RIKEN, for discussions at the early stage of our research related to this report. In particular, their presented reasoning of DIO was linear-logically formulated in our former work jointly with them (Okada *et al.* 2006). This report is based on this former work although the standpoint and analysis of the logical formulation taken in this report are different.

References

[Baader *et al.* 2003] Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., and Peter, F. Patel-schneider eds *The Description Logic Handbook*. Cambridge University Press, 2003.

[Banbara *et al.* 1998] Banbara, M., and Tamura, N. Compiling resources in a linear logic programming language. In Proceedings of the Workshop on Parallelism and Implementation Technology for Logic Programming Languages, pp.32-45, June 1998.

[Barwise *et al.* 1983] Barwise, J., and Perry, J. *Situations and Attitudes*, MIT Press, 1983.

[Girard 1987] Girard, J.-Y. Linear Logic, In: *Theoretical Computer Science* 50:pp.1-102, 1987.

[Girard 1995] Girard, J.-Y. Linear Logic: its syntax and semantics. In J.Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*. London Mathematical Society Lecture Note Series, Cambridge University Press, 1995.

[Hodas *et al.* 1994] Hodas, S.J., and Miller, D: Logic Programming in a Fragment of Intuitionistic Linear Logic, *Information and Computation*, 110(2):327-365, 1994.

[McCarthy *et al.* 1969] McCarthy, J., and Hayes, P.J. Some Philosophical Problems from the Standpoint of Artificial Intelligence. *Machine Intelligence* 4:pp.463-502, 1969.

[Miller 1996] Miller, D. Forum: A multi-conclusion specification language. *Theoretical Computer Science*, pp.165-201, 232, 1996.

[Milner 1999] Robin Milner. *Communicating and Mobile systems: the Pi-calculus*. Cambridge University Press, 1999.

[Okada 1998] Okada, M. An Introduction to Linear Logic: Phase Semantics and Expressiveness. In: *Theories of Types and Proofs*, eds. M.Takahashi-M.Dezani-M.Okada, Memoirs of Mathematical Society of Japan, vol.2 (1998) pp.255-295, second edition 1999.

[Okada 2004] Okada, M. Linear Logic and Intuitionistic Logic, *La revue internationale de philosophie* No. 230, pp.449-481, special issue "*Intuitionism*", 2004.

[Okada 2006] Okada M. A Linear Logical View of Intuitionistic Logic, in *Towards New Logic and Semantics: Franco-Japanese Collaborative Lectures on Philosophy of Logic*, Keio University Press, Tokyo, Japan, pp.185-229, 2006.

[Okada *et al.* 2006] Okada, M., Sugimoto, Y., Yoshikawa, S. and, Konagaya, A.: Drug Interaction Ontology (DIO) and the Resource-Sensitive Logical Inferences. *Essays Dedicated to Joseph A. Goguen*, Lecture Notes in Computer Science, 4060, pp.616-642, Springer, 2006.

[Smith 2004] Smith, B. The logic of biological classification and the foundations of biomedical on-

tology. In: Westerstahl D, editor. 10th International Conference in Logic Methodology and Philosophy of Science; Oviedo Spain: Elsevier-North-Holland; 2004.

[Smith *et al.* 2005] Smith, B., Ceusters,W., Klagges,B., Köhler,J., Kumar,A., Lomax,J., Mungall,C., Neuhaus,F., Rector L,A., and Rosse, C. Relations in biomedical ontologies, *Genome Biology* 2005, 6:R46, 2005.

[Yamaguchi *et al.* 2007] Yamaguchi, Y., Maruyama, T., Azuma, R., Yasunaga, M., and Konagaya, A.: Mesoscopic-level Simulation of Dynamics and Interactions of Biological Molecules Using Monte Carlo Simulation. VLSI Signal Processing 48(3): pp.287-299, 2007.

[Yoshikawa *et al.* 2003] Yoshikawa, S., and Konagaya, A.: DIO: Drug Interaction Ontology - Application to Inferences in Possible Drug-drug Interactions 2003, In: Proceedings of The 2003 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences (METMBS '03), June 23 - 26, (2003). Las Vegas, Nevada, USA: pp.231-236, 2003.

17. Ontology in Web Applications and Natural Language Processing

Noritada Shimizu[1], Fumihiro Kato[1], Ryosuke Sekido[2], Tatsuya Hagino[2], Jun Okamoto[3], and Shun Ishizaki[2]

[1] Graduate School of Media and Governance, Keio University
[2] Faculty of Environment and Information Studies, Keio University
[3] Research Institute at SFC, Keio University
5322 Endoh, Fujisawa, Kanagawa, 252-8520, Japan
{chiko,fumihiro,t05562rs,hagino,juno,ishizaki}@sfc.keio.ac.jp

Abstract. Ontology plays very important role in semantic web where machines can process web data and help to solve various problems. However, creating ontologies is difficult for ordinally users. In this paper, we show some attempts to automatically extract ontology information from current web pages and web activities.
Natural language processing (NLP) research has been using ontology (concept dictionary) since 1980's. It is crucial for text understanding systems. As one of the concept dictionaries, an associative concept dictionary is built by using large scale association experiments, and is used for NLP systems such as metaphor understanding or word sense disambiguation.

1 Ontology in Web Applications

The current web has a lot of useful information, but most of the information is written in HTML and can only be understood by human. When we try to solve some problem using the web, the most popular way is to use search engines, but they cannot directly solve the problem. They just list several pages related to our question expressed as a few keywords. We need to look through each page to really solve the problem. It is a time consuming process and we often wander to have some automatized mechanism.

Semantic web [Berners-Lee et al 2001] approaches this problem by creating web space for data which machines can understand and process. Data are expressed by RDF (Resource Description Framework). In semantic web, human readable content is written in HTML and its machine understandable meaning is written in RDF. For a given problem, software agent gathers RDF data from the web, combine them, apply rules, does some inference and lists the results. Human only needs to make the final decision of selecting suitable one from the results.

As semantic web dreams, if there are a lot of RDF data, we can solve various problems automatically. However, vocabularies used for RDF data need to be defined and it is quite difficult to do so. In addition, in order to make vocabularies useful, they need to be defined as an ontology. Creating ontologies is not an easy task for ordinary users and is becoming one of the big hurdles needed to be overcome in order to make semantic web more popular.