Using Coalgebra and Coinduction to Define Ontology-based Multi-agent Systems

Maja Hadzic*
Digital Ecosystems and Business Intelligence Institute (DEBII)
Curtin University of Technology, Perth, WA, Australia
E-mail: m.hadzic@curtin.edu.au
*Corresponding author

Elizabeth Chang
Digital Ecosystems and Business Intelligence Institute (DEBII)
Curtin University of Technology, Perth, WA, Australia
E-mail: e.chang@curtin.edu.au

Abstract: In the past, algebra has been predominantly used to define computer systems. However, computer systems are becoming more dynamic nowadays and algebra is not suitable anymore to define such systems. For this reason, coalgebra and coinduction have been introduced into the computer and information society. Coalgebra and coinduction present a powerful mechanism for representing many different kinds of dynamic systems using a common formal framework. Some dynamic systems, such as streams, automata and formal power series, have been viewed coalgebraically. However, ontology-based multi-agent systems have not been viewed coalgebraically yet. That is the aim of this paper. We make use of coinductive reasoning to provide a framework and define a dynamic process within the ontology-based multi-agent system. The principle is illustrated on an ontology-based multi-agent system specially designed to retrieve human disease information intelligently from a variety of heterogeneous and distributed information resources.

Keywords: coalgebra, coinduction, dynamic system, ontology-based multi-agent system, intelligent information retrieval.

Reference to this paper should be made as follows: Hadzic, M. and Chang, E. (2008) ‘Using Coalgebra and Coinduction to Define Ontology-based Multi-agent Systems’, *Int. J. Metadata, Semantics and Ontologies,* Vol. x, Nos. x, pp. xx-xx.

Biographical notes: Maja Hadzic holds PhD in Health Information Systems and is currently a Research Fellow at the Digital Ecosystems and Business Intelligence Institute of the Curtin University of Technology. She has a multi-disciplinary background and her current research interests include Ontologies, Multi-agent Systems, Data Mining, Digital Ecosystems and Mental Health.

Elizabeth Chang is Professor at the Curtin University of Technology. She is the Director of the Digital Ecosystems and Business Intelligence Institute. Professor Chang has an extensive background, knowledge and skills in academia, commerce and industry. She has done significant work in the areas of Ontologies, XML Technologies, Data Mining of Complex Data, P2P Applications, Trust, Reputation and Recommendations Technologies, Software Engineering and Fuzzy Systems Applications in IT and Internet Systems.

1 INTRODUCTION

1.1 Modelling Complex Information Systems

Agent-based computing is becoming one of the leading technologies in the world. Computer systems are advancing towards a greater ability to emulate human intelligence. This is not achieved through their processing power but through their adaptability and interactivity (Wegner and Goldin, 1999).

The complexity of modelling an information system comes from the complex nature of its solution space. Coalgebra presents a powerful mechanism for representing many different kinds of dynamic systems using a common formal framework (Cirstea, 2004; Goldin *et al.*, 2000).

Coalgebra proposes a general paradigm for modelling dynamic systems (Rutten, 2000). A dynamic system is represented as a tuple <S, αs> where:

- S represents the state space of the system
- αs is called the system dynamics and is in the form of S → F (S) which maps each state in S to a set of dynamics determined by the functor F (S)

Such a system is also called an F-coalgebra since its dynamics are determined by the functor F (S).

Coalgebra cannot really be utilised until the formulation of a general notion of coalgebraic bisimulation which is needed to formulate a general principle of coinduction. It is coinduction which constitutes the heart of the coalgebraic matter (Rutten, 2000). Coinduction is a stronger form of definition and reasoning for non-enumerable sets of infinite structures. Streams, automata and formal power series have been viewed coalgebraically (Rutten, 1999 and 2003). However, multi-agent systems have not been viewed coalgebraically yet. That is aim of this paper.

Coinductive reasoning allows finite computing agents to perform tasks with non-enumerable interactive behaviours (Wegner and Goldin, 2005). In this way, it provides a semantic framework for multi-agent interaction.

1.2 Overview of the Ontology-based Multi-agent System

1.2.1 Human Disease Information

Years of research in the area of human diseases have resulted in the accumulation of millions of papers, articles and journals totalling several petabytes in electronic form. The information covers different types of human diseases and produces a huge range of results regarding different disease types, symptoms, treatments and disease-causing factors (genetic and environmental) as well as candidate genes that could be responsible for the onset of these diseases. Most of the published results are accessible over the internet through journal databases and medical databases and text corpuses. However, two major problems are adversely affecting the efficiency and advancement of human disease research, namely, issues related to the lack of an existing infrastructure to support effective access and information retrieval within petabytes of distributed unstructured information and lack of tools to analyse the available information and derive useful knowledge from it. There is not a lack of information but its sheer volume is a factor hindering medical research. We propose the use of ontology-based multi-agent systems to remedy that situation.

In our previous work (Hadzic and Chang, 2005a; Hadzic *et al.*, 2006), we showed how ontologies can be used by multi-agent systems in intelligent information retrieval processes. Using ontologies over knowledge bases is preferred because ontologies, as high expressive knowledge models, increase system expressiveness and intelligence (Fernández *et al.*, 1997). We showed how ontologies can be used to support most processes involved in information retrieval: posing queries by users, problem decomposition and task sharing among different agents, analysis and selection of information, information assembly and structured presentation of results to the users.

The combination of ontology and multi-agent technology results in an intelligent and dynamic information retrieval system. Multi-agent systems integrate dispersed information resources. Relevant information is retrieved, selected and assembled according to the domain ontology. This technology provides not only a system for intelligent and dynamic information retrieval but is also a powerful mechanism for dynamic building of new knowledge. The two methodologies together (ontology and multi-agent systems) are the latest technologies with potential future use by various applications and for various purposes.

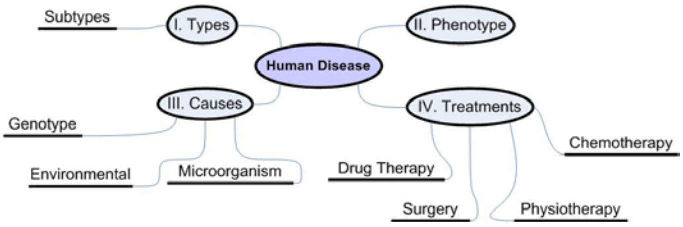

Figure 1 *Top-level hierarchy of the GHDO*

1.2.2 Generic and Specific Human Disease Ontologies

For the purpose of representing knowledge regarding human diseases, we designed an ontological model of Generic Human Disease Ontology (GHDO) (Hadzic and Chang, 2005a). The GHDO is composed of four sub-ontologies: disease types, causes, symptoms and treatments. The top level hierarchy is shown in Figure 1.

Each sub-ontology is unique and different from the other three. A disease is completely and uniquely characterised by describing each of its four sub-ontologies. This is the reason that each sub-ontology is called a ‘dimension’. We say that each disease is uniquely characterised by its four ‘dimensions’. It is impossible for two different diseases to have the same values for all of the four ‘dimensions’. In the case of different diseases, at least one ‘dimension’ needs to be different. Moreover, each ‘dimension’ is different from the others according to the knowledge it represents and the way this knowledge is being represented. The ‘Disease types’ sub-ontology is classifying ontology and is hierarchically supported. The ‘Symptoms’ sub-ontology is of a descriptive nature. The ‘Causes’ sub-ontology is scientifically supported. Mostly, it represents knowledge resulting from laboratory experiments. The ‘Treatments’ sub-ontology is a combination of classifying and scientific ontology. New drugs are the result of numerous lab experiments (scientific ontology) and all drugs used can be classified according to their nature (classifying ontology).

Structuring ontology according to the knowledge it represents is a fundamental aspect that supports the process of intelligent information retrieval. According to the user's query that part of the GHDO that matches this query is assembled together. This step results in a Specific Human Disease Ontology template (SHDO template). For the purpose of problem decomposition and task sharing among different agents, the SHDO template is firstly decomposed according to the four sub-ontologies. Each of the sub-ontologies is then further decomposed into the smaller sub-sub-ontologies. This process of decomposition goes on until the problems reach the size of atomic actions and can be performed by individual agents. After information retrieval, results from individual agents are analysed, compared and intelligently integrated together. Relevant information is assembled into the SHDO template. This step results in a Specific Human Disease Ontology (SHDO) that is presented to the user as the answer to his/her query. With focus on ontology, this process is shown in Figure 2.

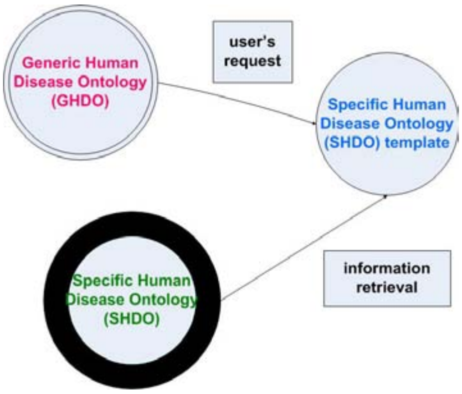
Generic Human Disease Ontology (GHDO)
user's request
Specific Human Disease Ontology (SHDO) template
Specific Human Disease Ontology (SHDO)
information retrieval

Figure 2 *GHDO, SHDO template and SHDO*

1.2.3 GHDO-based Holonic Multi-agent system

Multi-agent systems consist of autonomous agents with holonic properties that group together forming holons and holarchies. Such hierarchically structured multi-agent architecture provides a framework where each agent represents a node in the hierarchic tree. The GHDO-based Holonic Multi-agent Structure (GHMS) (Hadzic *et al.*, 2006) is a nested hierarchy of four holarchies. Each of the four GHDO 'dimensions' (Disease types, Symptoms, Causes and Treatments) is associated with one holarchy.

In this multi-agent system, we have four types of agents: Interface, 'Manager', Information and 'Smart' agents (Hadzic and Chang, 2005b). Interface agents assist users in composing queries and present the answers after information retrieval. According to the SHDO template, 'Manager' agents assign specific tasks to information agents in order to gather requested information in the most efficient way. Information agents retrieve requested information and hand over the retrieved information to 'Smart' agents. 'Smart' agents reason with this information, select relevant information, assemble it correctly and hand it over to interface agents for presentation to the user.

1.3 Content of This Paper

Within our system, implementation of the designed ontology within a multi-agent system results in a dynamic system. The Specific Human Disease Ontology template that is formed in the beginning of this process is manipulated many times until the process results in a Specific Human Disease Ontology (SHDO). Template decomposition, partition, addition of retrieved information to this template and template assembly, are some changes that occur throughout the process. We will address the dynamics of the ontology-based multi-agent system and complex problems later.

In Section 2, we give general definitions of three basic notions of universal coalgebra: Coalgebra, Homomorphism and Bisimulation. We discuss how these definitions can be applied in our information system to explain the principle of mapping attributes to ontology terms as well as the uniqueness of each disease-system in terms of its four sub-ontologies.

In Section 3, we discuss the basic facts of bisimulation and apply these to our information system. We prove that the inverse of each bisimulation is again bisimulation, and that composition and union of bisimulations is again a bisimulation. This is important in the information assembly step as smaller sub-sub-ontologies, defined as bisimulations are assembled.

In Section 4, we provide general definitions of Relators, Coproducts and Coequalisers to explain other characteristics of our information system. We explain how a Disease-relator is used to map relationships between terms in an SHDO template to relationships between the same terms and their attributes in SHDO. Coequalisers are used to solve situations of dynamic systems when the same action is performed twice in apparently identical circumstances appears to produce different effects. Coproducts are used to explain the principle of information assembly in the final steps of the information retrieval process.

Pullbacks and their compositions that are used to explain the principle of template decomposition for the purpose of task sharing between different information agents are defined in Section 5.

In Section 6, we give general definitions of the Final Systems and apply these to define the final stage in the information assembly process within our information system. This final step results in FinSHDO that is presented to the user in answer to his/her query.

We introduce the principle of Natural Transformation and Homomorphism Mapping in Section 7. We use the principle of Natural Transformation to define situations when a user is interested in retrieving the same information (same SHDO template) regarding two or more different diseases such as for the purpose of comparing retrieved information. We define Homomorphism Mapping to explain situations where a user is interested in the same information regarding two different diseases. Finally, we conclude this paper in Section 8.

2 COALGEBRA AND HOMOMORPHISM

Initial algebras have traditionally been used to model finite data types. Final Coalgebras, dual of algebras, were used in order to deal with infinite data types. Coalgebras turned out to be suitable as models for dynamical systems.

An important property of initial algebra is that they satisfy the principle of induction. Such a principle was missing for Coalgebras until the work that produced a theory of non-wellfounded sets. In this work, a proof principle, nowadays called Coinduction, was introduced (Aczel, 1998).

The three basic notions of universal algebra: algebra, homomorphism of algebras and congruence, turn out to correspond to: Coalgebra, Homomorphism of coalgebras and Bisimulation respectively (Aczel and Mendler, 1989).

2.1 Disease-Colagebra

Definition 1 *(of Coalgebra)* (Rutten, 2000):

Let F: Set → Set be a functor.

An F-coalgebra or F-system is a pair <S, αs> consisting of a set S and a function αs: S → F(S). The set S is called the carrier of the system which is also called the set of states. The function αs is called F-transition structure (or dynamics) of the system.

In our information system, each query for a specific disease acts as an F-system. By analogy, we call F-coalgebra a Disease-coalgebra. The analogy is shown in Figure 3.

S is analogous to a Specific Human Disease Ontology (SHDO) template (TO) that contains terms only. SHDO (SO) also contains attributes, values and other additional information. SHDO (SO) is an analogue for F(S).

The additional information contained in SHDO (SO) is retrieved from different databases through cooperatively working agents. Disease-transition structure (dynamics of the system, function αs) is analogous to F-transition structure and maps the SHDO template (TO) to SHDO (SO).

$$\begin{tikzcd}
S \arrow[d, "\alpha s"'] & \text{SHDO template } (\mathcal{TO})\text{: terms} \arrow[d, "\alpha s"] \\
F(S) & \text{SHDO } (\mathcal{SO})\text{: terms + attributes}
\end{tikzcd}$$

Figure 3 F-coalgebra used to represent the system dynamics

2.2 Disease-homomorphism and -isomorphism

Definition 2 *(of Homomorphism)* (Rutten, 2000):

Let <S, αs> and <T, αt> be two F-systems, where F is an arbitrary functor. A function f: S→ T is a Homomorphism of F-systems (or F-homomorphism) if:

F(f) ∘ αs = αt ∘ f.

$$\begin{tikzcd}
S \arrow[r, "f"] \arrow[d, "\alpha s"'] & T \arrow[d, "\alpha t"] \\
F(S) \arrow[r, "F(f)"'] & F(T)
\end{tikzcd}$$

Figure 4 *Homomorphism f of F-systems*

Intuitively, a homomorphism is a function that preserves and reflects F-transition structures. We sometimes write f : <S, αs> → <T, αt> to express that f is a Homomorphism. Homomorphism f of F-system is shown in Figure 4.

The Identity function on an F-system <S, αs> is always a Homomorphism and composition of two Homomorphisms is again a Homomorphism. Thus, the collection of all F-systems together with F-system Homomorphisms is a category, which we denote by Set_F.

Definition 3*(of isomorphism)* (Rutten, 2000):

An F-homomorphism f : S→ T with an inverse f^{-1}: T → S which is also a Homomorphism is called an Isomorphism between S and T. S ≅T means that there exists an Isomorphism between S and T.

We said that the F-system can be used to characterise a query for a specific disease in our information system. For instance, we have a query regarding bipolar disorders. The user's request is represented as a SHDO template. Bipolar disorder is characterised by F-coalgebra (we call it Bipolar-coalgebra) and function α maps attributes to this template. The template may cover information in different ontology 'dimensions' (disease types, symptoms, causes, treatments).

We assume that each disease is uniquely characterised by the four ontology 'dimensions'. In the case of different diseases at least one of the four 'dimensions' needs to be different. This means that if, for a specific disease, one 'dimension' is given, the other three 'dimensions' are specified as well.

Each 'dimension' is specific for a specific disease. This means that given a specified template for a particular disease and, if Disease-coalgebra (and αs) is used to map attributes to Symptoms sub-ontology, we can then use Disease-coalgebra (and αt) to map attributes to Treatments sub-ontology of SHDO template. The case of bipolar disorder is represented in Figure 5.

We can write: f : <Sym, αs> → <Tre, αt>. Collection of all Bipolar-systems together with Bipolar-system Homomorphisms is a category, which we denote by $Set_{Bipolar}$. This collection contains all possible templates requiring information for all possible combinations of the four 'dimensions' (or part of them) for bipolar disease. The number of these templates is infinite but they all have one

Disease-homomorphism (in our example, Bipolar-homomorphism).

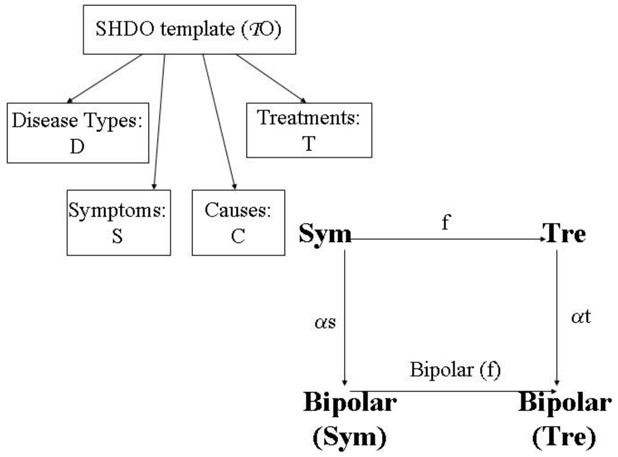

Figure 5 *Mappings between different ontology 'dimensions', Homomorphism f of Bipolar-system*

There exists an Isomorphism between different ontology 'dimensions' (sub-ontologies). Mappings are possible in both directions. For instance in the example above, we can map Treatments to Symptoms sub-ontology: f^{-1}: Tre → Sym.

The SHDO template is derived from and, therefore embedded in, the GHDO. Also, each of the four sub-ontologies is embedded in GHDO. Each of the sub-ontologies specified in the SHDO template is embedded in this template. Each sub-ontology can be further partitioned into smaller sub-sub-ontologies for the purpose of task sharing between different information agents. The resulting sub-sub-ontologies are embedded in the original sub-ontology.

2.3 Disease-monomorphism and –epimorphism

Definition 4 *(of monomorphism)* (Rutten, 2000):

An injective Homomorphism is called Monomorphism. Given systems S and T, we say that S can be embedded into T if there is a Monomorphism from S to T.

Definition 5 *(of epimorphism)* (Rutten, 2000):

A surjective Homomorphism is called Epimorphism. If there exists an Epimorphism from S to T, T is called a Homomorphic image of S. In that case, T is also called a quotient of S.

Theorem 1:

Every bijective Homomorphism is necessarily an Isomorphism.

Proof:

If $f : \langle S, \alpha s\rangle \to \langle T, \alpha t\rangle$ is an F-homomorphism and f^{-1}: $S \to T$ is an inverse of f then

$\alpha s \circ f^{-1} = F(f^{-1}) \circ \underline{F(f) \circ \alpha s} \circ f^{-1} \wedge F(f) \circ \alpha s = \alpha t \circ f$ (homomorphism condition)

$\Rightarrow \alpha s \circ f^{-1} = F(f^{-1}) \circ \underline{\alpha t \circ f} \circ f^{-1} = F(f^{-1}) \circ \alpha t$

This means that f^{-1} is Homomorphism.

Theorem 2:

Let S, T and C be systems and $f : S \to T$, $g : S \to C$ and $h : C \to T$ functions. S, T and C represent, for example, Symptoms, Treatments and Causes sub-ontologies and functor F represent functor Disease (see Figure 6).

- If $f = \circ h g$, g is surjective and f a nd g are Homomorphisms, then h is a Homomorphism.
- If $f = \circ h g$, h is injective and f and h are Homomorphisms, then g is a Homomorphism.

Proof:

Let $c \in C$ and let $s \in S$ be such that $g(s) = c$.

$F(h) \circ \alpha c(c) = F(h) \circ \underline{\alpha c \circ g(s)} \wedge \alpha c \circ g = F(g) \circ \alpha s$ (Homomorphism condition)

$\Rightarrow F(h) \circ \alpha c(c) = F(h) \circ \underline{F(g) \circ \alpha s} \wedge F(f) = F(h) \circ F(g)$

$\Rightarrow F(h) \circ \alpha c(c) = F(f) \circ \alpha s \wedge F(f) \circ \alpha s = \alpha t \circ f$ (Homomorphism condition)

$\Rightarrow F(h) \circ \alpha c(c) = \alpha t \circ f(s) \wedge f = h \circ g$

$\Rightarrow F(h) \circ \alpha c(c) = \alpha t \circ h \circ g(s) = \alpha t \circ h(c)$

$$\begin{tikzcd}
S \arrow[rr, "f"] \arrow[dr, "g"'] \arrow[ddd, "\alpha s"'] & & T \arrow[ddd, "\alpha t"] \\
& C \arrow[ur, "h"'] \arrow[d, "\alpha c"] & \\
& F(C) \arrow[dr, "F(h)"] & \\
F(S) \arrow[ur, "F(g)"] \arrow[rr, "F(f)"'] & & F(T)
\end{tikzcd}$$

Figure 6 *f : S → T, g : S → C and h : C → T functions*

3 BASIC FACTS ON BISIMULATION

3.1 Disease-bisimulation

We now come to the third fundamental notion of universal Coalgebra. A bisimulation between two systems is intuitively a transition structure respecting the relation between sets of states. For an arbitrary set functor F : Set → Set, it is defined as follows:

Definition 6 *(of bisimulation)* (Rutten, 2000):

Let $\langle S, \alpha s\rangle$ and $\langle T, \alpha t\rangle$ be F-systems. A subset $R \subseteq S \times T$ of the Cartesian product of S and T is called an F-bisimulation between S and T if there exists an F-transition structure $\alpha r : R \to F(R)$ such that the projections from R to S and T are F-homomorphisms. This is shown in Figure 7.

$$\begin{tikzcd}
S & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "\alpha r"] & T \arrow[d, "\alpha t"] \\
F(S) & F(R) \arrow[l, "F(\pi_1)"] \arrow[r, "F(\pi_2)"'] & F(T)
\arrow[from=1-1, to=2-1, "\alpha s"']
\end{tikzcd}$$

Figure 7 *<R, αr> is F-bisimulation between <S, αs> and <T, αt>*

We shall also say, making explicit reference to the transition structures, that <R, αr> is Bisimulation between <S, αs> and <T, αt>. If <S, αs> = <T, αt>, <R, αr> is called Bisimulation on <S, αs>. Two states s and t are called Bisimilar if there exists a Bisimulation R with <s, t> ∈ R.

Here, π_1 and π_2 are called Product Projections ($\pi_1 : S \times T \rightarrow S$ and $\pi_2 : S \times T \rightarrow T$) and $i : R \rightarrow S \times T$ is the Inclusion Map.

We say that two sub-ontologies O1 and O2 are Bisimilar as there exists a Bisimulation R with < O1, O2 > ∈ R. For the example above, we can say that Symptom and Treatment sub-ontologies are Bisimilar if there exists a Bisimulation R with < Sym, Tre > ∈ R (see Figure 8).

Generally, the principle of Disease-bisimulation is fulfilled within our information system for all different sub-ontologies (or parts of them). We have Disease-bisimulation between Symptoms and Treatments, Symptoms and Causes, Treatments and Causes, Disease Types and Symptoms, Disease Types and Treatments, and Disease Types and Causes sub-ontologies.

$$\begin{tikzcd}
Sym \arrow[d, "\alpha s"'] & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "\alpha r"] & Tre \arrow[d, "\alpha t"] \\
Disease\ (Sym) & Disease\ (R) \arrow[l, "Disease\ (\pi_1)"'] \arrow[r, "Disease\ (\pi_2)"'] & Disease\ (Tre)
\end{tikzcd}$$

Figure 8 *R, αr> is Disease-bisimulation between <Sym, αs> and <Tre, αt>*

3.2 Inverse of Disease-bisimulation

Theorem 3 (Rutten, 2000):

Let < R, αr > be a Bisimulation between systems S and T. The inverse R^{-1} of R is a Bisimulation between T and S.

Proof:

Let $i : R \rightarrow R^{-1}$ be the Isomorphism sending < s, t > ∈ R to < t, s > ∈ R^{-1}. Then < R^{-1}, $F (i) \circ \alpha r \circ i^{-1}$ > is a Bisimulation between T and S (see Figure 9).

Let < R, αr > be a Bisimulation between two Disease-systems, for example Symptoms and Treatments sub-ontologies. The inverse R^{-1} of R is then a Bisimulation between Treatments and Symptoms sub-ontologies. This is shown in Figure 10.

$$\begin{tikzcd}
 & & R^{-1} \arrow[d, "\alpha r^{-1}"] & \\
S \arrow[d, "\alpha s"'] & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[ur, "i"] \arrow[d, "\alpha r"] & \arrow[r] & T \arrow[d, "\alpha t"] \\
 & & F(R^{-1}) & \\
F(S) & F(R) \arrow[l, "F(\pi_1)"] \arrow[ur, "F(i)"] \arrow[rr, "F(\pi_2)"'] & & F(T)
\end{tikzcd}$$

Figure 9 *< R^{-1}, $F (i) \circ \alpha r \circ i^{-1}$ > is Bisimulation between T and S*

This principle of Bisimulation of the inverse R^{-1} of R is not only applicable to the Symptoms and Treatments sub-ontologies as shown in Figure 10, but also to all mappings between different sub-ontologies (and parts of those sub-ontologies) such as the Causes and Disease types sub-ontologies.

$$\begin{tikzcd}
 & & R^{-1} \arrow[d, "\alpha r^{-1}"] & \\
Sym \arrow[d, "\alpha s"'] & R \arrow[l, "\pi_1"'] \arrow[rr, "\pi_2"] \arrow[ur, "i"] \arrow[d, "\alpha r"] & & Tre \arrow[d, "\alpha t"] \\
 & & Disease\ (R^{-1}) & \\
Disease\ (Sym) & Disease\ (R) \arrow[l, "Disease\ (\pi_1)"] \arrow[ur, "Disease\ (i)"] \arrow[rr, "Disease\ (\pi_2)"] & & Disease\ (Tre)
\end{tikzcd}$$

Figure 10 *< R^{-1}, Disease $(i) \circ \alpha r \circ i^{-1}$ > is Bisimulation between Tre and Sym*

3.3 Composition of Disease-bisimulations

We want to show that the composition and union of Bisimulations is again a Bisimulation. The following lemma is helpful.

Lemma 1 (Rutten, 2000):

The image <f, g> (T) = {< f (t), g (t)> | t ∈ T} of two Homomorphisms f : T → S and g : T → U is a Bisimulation on S and U.

Proof:

Consider Figure 11.

Here, the function j is defined by j (t) = < f (t), g (t) >.

The function i is any right inverse for j (which exists by the axiom of choice because j is surjective): $j \circ i$.

π_1 and π_2 are projections.

Note that everything in this diagram commutes.

The set < f, g > (T) can be given a transition structure γ : < f, g > (T) → F (< f, g > (T)) by defining $\gamma = F (j) \circ \alpha t \circ i$.

Consider $F(\pi_1) \circ \gamma = F(\pi_1) \circ F(j) \circ \alpha t \circ i = F(\pi_1 \circ j) \circ \alpha t \circ i = F(f) \circ \alpha t \circ i$

$\wedge\ F(f) \circ \alpha t = \alpha s \circ f$ (Homomorphism condition)

$\Rightarrow F(\pi_1) \circ \gamma = \alpha s \circ f \circ I = \alpha s \circ \pi_1$

From this follows that $(< f, g > (T), \gamma)$ is a Bisimulation on S and U.

$$\begin{tikzcd}
 & {<f, g>(T)} \arrow[dl, "\pi_1"'] \arrow[dr, "\pi_2"] \arrow[d, shift left, "i"] & \\
S & T \arrow[l, "f"] \arrow[r, "g"'] \arrow[u, shift left, "j"] & U
\end{tikzcd}$$

Figure 11 *Image <f, g> (T) of two Homomorphisms f and g*

***Theorem 4** (composition theorem)* (Rutten, 2000):

The composition $R \circ Q$ of two Bisimulations $R \subseteq S \times T$ and $Q \subseteq T \times U$ is a Bisimulation between S and U.

Proof:

$R \circ Q$ is equal to the image $< r_1 \circ x_1, q_2 \circ x_2 > (X)$ of the pullback, as shown in Figure 12.

$$\begin{tikzcd}
 & & X \arrow[dl, "x_1"'] \arrow[dr, "x_2"] \arrow[dd] & & \\
 & R \arrow[dl, "r_1"'] \arrow[dr, "r_2"] & & Q \arrow[dl, "q_1"'] \arrow[dr, "q_2"] & \\
S & & T & & U
\end{tikzcd}$$

Figure 12 *The composition $R \circ Q$ of two Bisimulations R and Q*

Here x_i, r_i and q_i are projections.

Because F preserves pullbacks, the pullback X can be supplied with a transition structure, by (***) (see Section 5.2). By Lemma 1, $R \circ Q$ is a Bisimulation on S and U.

The principle of composition and union of Bisimulations is important in the process of combining and assembling different data coming from different agents, unifying them into one SHDO.

Consider a situation where a user has requested information regarding the causes of a disease. The Cause sub-ontology is then partitioned by 'Manager' agents into smaller sub-sub-ontologies. Many information agents are responsible for finding information regarding environmental and genetic causes of the disease in question. For example, one information agent collects information about how stress can influence the onset of that particular disease; a second agent collects information regarding gene mutations of human DNA that can cause that disease; and the third agent looks for information about accumulative or complementary effects of genetic and environmental factors on the development of that specific disease. The final SHDO is formed as a composition and union of its sub-ontologies and sub-sub-ontologies.

We prove that Bisimulation characteristics of Disease-systems of $Set_{Disease}$ do not change in the complex process of information retrieval.

$$\begin{tikzcd}
 & & Cau \arrow[dl, "c_1"'] \arrow[dr, "c_2"] \arrow[dd] & & \\
 & Env \arrow[dl, "e_1"'] \arrow[dr, "e_2"] & & Gen \arrow[dl, "g_1"'] \arrow[dr, "g_2"] & \\
Stress & & Comb & & Gene
\end{tikzcd}$$

Figure 13 *The composition Env ∘ Gen of two Bisimulations Env and Gen*

Let Cau, Env, Gen, Stress, Comb and Gene be Disease-systems representing Causes of disease, Environmental causes, Genetic causes, Stress (as environmental cause), Combination of different causes and Gene mutations sub-sub-ontologies respectively. The correct information composition can be defined as the composition Env ∘ Gen of two Bisimulations $Env \subseteq Stress \times Comb$ and $Gen \subseteq Comb \times Gene$, and is a Bisimulation between Stress and Gene. This is shown in Figure 13.

In our model, a disease-system may consist of different sub-systems. For example, Stress and Comb are sub-systems of system Env (see Figure 13). The sub-systems are generally defined as follows:

***Definition 11** (of Subsystems)* (Rutten, 2000):

Let $< S, \alpha s >$ be a system and let V be a subset of S with inclusion mapping $i : V \rightarrow S$. If there exists a transition structure αv on V such that $i : < V, \alpha v > \rightarrow < S, \alpha s >$ is a Homomorphism then V is called a Subsystem (or Sub-coalgebra) of S. Unions and intersections of sub-systems are again sub-systems.

3.4 Union of Disease-bisimulations

The mapping of one sub-ontology to another is a complex process so that many different bisimulations may be needed to achieve this conversion. For instance, some bisimulations operate on ontology bases, while others operate on ontology commitment layers. Because of the existence of various Bisimulations between different Disease-systems, we consider the following theorem.

***Theorem 5** (union theorem)* (Rutten, 2000):

The union $\cup_k R_k$ of a family $\{R_k\}_k$ of Bisimulations between systems Sym and Tre is again a Bisimulation.

Proof :

$\cup_k R_k$ is the image of : $\mathrm{Sym} \xleftarrow{k} (\sum_i R_i) \xrightarrow{l} \mathrm{Tre}$, where k and l are the Component-wise Projections.

Coproduct of a family of systems is again a system, by (**) (see Section 4.3).

It follows from Lemma 1 that the union is a Bisimulation.

Corollary 1:

The set of all Bisimulations between systems Sym and Tre is a complete lattice with

least upperbounds and greatest lowerbounds given by:

$\vee_k \mathbf{R}_k = (\cup_k R_k)$

$\wedge_k \mathbf{R}_k = \cup \{ R \mid R$ is a Bisimulation between Sym and Tre with $R \subseteq \wedge_k R_k \}$

In particular, the greatest Bisimulation between Sym and Tre exists and is denoted by : $\sim_{<\mathrm{Sym},\ \mathrm{Tre}>}$. It is the union of all Bisimulations:

$\sim_{<\mathrm{Sym},\ \mathrm{Tre}>} = \cup \{ R \mid R$ is a Bisimulation between Sym and Tre $\}$

We shall write $s \sim_{<\mathrm{Sym},\ \mathrm{Tre}>} t \Leftrightarrow < s, t > \in \sim_{<\mathrm{Sym},\ \mathrm{Tre}>} \Leftrightarrow$ there exists a Bisimulation R with $< s, t > \in R$.

The greatest Bisimulation on one single system Sym, denoted by $\sim_{\mathrm{Sym}}$, is a Bisimulation equivalence.

The Theorem 5 and Corollary 1 are also true for Bisimulations among other sub-ontologies (such as Causes and Disease types sub-ontologies), sub-sub-ontologies (for example, Environmental and Genetic causes) and parts of the ontologies.

The principle of least upperbounds and greatest lowerbounds can be used by the information system to adjust its min and max conversion, low and high selection and compromise between quality and quantity of data.

4 DISEASE-RELATORS, COPRODUCTS AND COEQUALISERS

4.1 Disease-relator

Definition 7 *(of relator)* (Cirstea, 2004):

F- relator is a mapping from relations to relations, taking relations on S × T to relations on F (S) × F (T).

A monotonic F-relator is required to satisfy some additional constraints including preservation of inclusions between relations and preservation of relational composition.

In our information system, this would mean that each query would have its Disease-relator corresponding to its SHDO template. This Disease-relator maps relations between terms in SHDO template and relationships between the same terms and their attributes in SHDO. Preserving the relationship order between ontology terms is especially important during template manipulation processes such as:

- Ontology decomposition for the purpose of task sharing between different information agents
- Adding attributes to ontology terms
- Assembling information that comes from different information agents into SHDO template

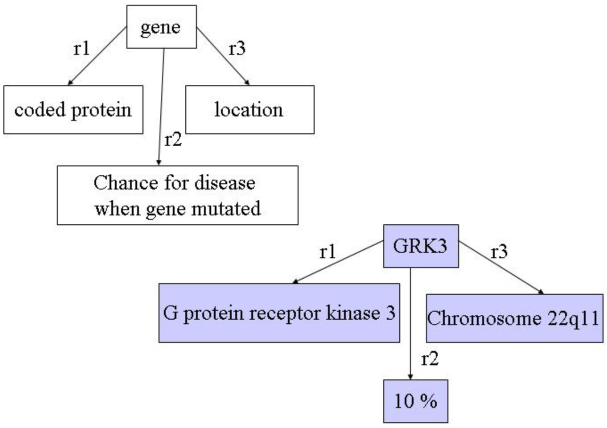
gene
r1
r3
coded protein
location
r2
Chance for disease when gene mutated
GRK3
r1
r3
G protein receptor kinase 3
Chromosome 22q11
r2
10 %

Figure 14 *Relationships between terms and its attributes within a part of Causes sub-ontology*

In Figure 14, we show an example where a user is interested in genetic causes of bipolar disorders, specifically gene mutations. A part of SHDO template is shown in the top left corner of the figure. The name of the gene, name of protein this gene codes for, location of this gene on DNA (which chromosome and where exactly on this chromosome) and effectiveness of the gene mutation on chance for disease development are all unknown. Between different terms of SHDO template, relations r1, r2 and r3 are specified.

The information agents provide the required information. The name of the gene is GRK3 (which means G protein receptor kinase 3). This gene is situated on chromosome 22 (position q11) and when this gene mutates, it is responsible for the development of bipolar disorder in 10% of the cases (Barrett *et al.*, 2003). This information needs to be assembled correctly.

The relations r1, r2 and r3 between terms within SHDO template need to be defined in the same way between attributes of these terms within SHDO. A disease-relator, specifically Bipolar-relator, maps from relations r1, r2, r3 in SHDO template to relations r1, r2 and r3 in SHDO. As the terms are mapped to their attributes, so are the relationships between these terms mapped to the relations between corresponding attributes.

4.2 Coproduct of Disease-systems

We consider Coproducts in Set $_F$.

***Definition 8** (of coproduct)* (Rutten, 2000):

The Coproduct (or Sum) of two F-systems <S, αs> and <T, αt> can be constructed as follows. Let $i_S : S \to (S + T)$ and $i_T : T \to (S + T)$ be the injections of S and T into their disjoint union. There is a unique function γ : (S + T)→ F (S + T) such that both i_S and i_T are Homomorphisms. This is presented in Figure 15.

$$\begin{tikzcd}
S \arrow[r, "i_S"] \arrow[d, "\alpha s"'] & S + T \arrow[d, "\gamma"] & T \arrow[l, "i_T"'] \arrow[d, "\alpha t"] \\
F(S) \arrow[r, "F(i_S)"'] & F(S + T) & F(T) \arrow[l, "F(i_T)"]
\end{tikzcd}$$

Figure 15 *Coproduct of two F-systems <S, αs> and <T, αt>*

During the process of information retrieval, the SHDO template is partitioned by 'Manager' agents into sub-ontologies and these sub-ontologies are further portioned into smaller sub-sub-ontologies for the purpose of task sharing among different information agents. Requested information is then retrieved and added onto these smaller partitions of the SHDO template. Resulting sub-ontologies and sub-sub-ontologies need to be correctly assembled into the SHDO. The SHDO is then presented back to the user.

The Coproduct (or Sum) of two Disease-systems, for instance <Sym, αs> and <Tre, αt>, is constructed by the analogous principle as described above. For Symptoms and Treatments sub-ontologies, we have: i_S : Sym→ (Sym + Tre) and i_T : Tre → (Sym + Tre) are injections of Sym and Tre into their disjoint union. There is a unique function γ : (Sym + Tre)→ Disease (Sym + Tre) such that both i_S and i_T are Homomorphisms. Identically, we can define coproduct of other Disease-systems, such as for Cause and Disease types sub-ontologies: <Cau, αc> and <DTy, αd>. We present this in Figure 16.

$$\begin{tikzcd}
Sym \arrow[r, "i_S"] \arrow[d, "\alpha s"'] & Sym + Tre \arrow[d, "\gamma"] & Tre \arrow[l, "i_T"'] \arrow[d, "\alpha t"] \\
Disease\ (Sym) \arrow[r, "Disease\ (i_S)"] & Disease\ (Sym + Tre) & Disease\ (Tre) \arrow[l, "Disease\ (i_T)"']
\end{tikzcd}$$

Figure 16 *Coproduct of two Disease-systems <Sym, αs> and <Tre, αt>*

The function γ acts on Sym as (Disease (i_S) ∘ αs) and on Tre as (Disease (i_T) ∘ αt). System < Sym + Tre, γ > has following universal property for any system < Uni, αu >, and Homomorphisms k : < Sym, αs > → < Uni, αu > and l : < Tre, αt >→ < Uni, αu >; there exists a unique Homomorphism h : < Sym + Tre, γ > → < Uni, αu > making the diagram shown in Figure 17 commute.

$$\begin{tikzcd}
 & Uni & \\
Sym \arrow[ur, "k"] \arrow[r, "i_S"'] & Sym + Tre \arrow[u, "h"] & Tre \arrow[ul, "l"'] \arrow[l, "i_T"]
\end{tikzcd}$$

Figure 17 *< Sym + Tre, γ > is the Coproduct of < Sym, αs > and < Tre, αt >*

4.3 Coequalisers of Disease-homomorphisms

***Definition 9** (of coequaliser)* (Rutten, 2000):

In Set $_F$, a Coequaliser of two homomorphisms can be constructed. Consider two Homomorphisms f : < S, αs > → < T, αt > and g : < S, αs > → < T, αt >. We have to find a system < U, αu > and a Homomorphism h : < T, αt > → < U, αu > such that:

1. h ∘ f = h ∘ g
2. for every Homomorphism h′ : < T, αt > → < U′, αu′ > such that ′ h∘ f = h′ ∘ g, there exists a uniq ue Homomorphism m : < U, αu >→ < U′, αu′ > with the property that m′ ∘ h = h′ .

$$\begin{tikzcd}
S \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] \arrow[d, "\alpha s"'] & T \arrow[r, "h"] \arrow[d, "\alpha t"] & U \arrow[d, "\alpha u"] \\
F(S) \arrow[r, shift left, "F(f)"] \arrow[r, shift right, "F(g)"'] & F(T) \arrow[r, "F(h)"'] & F(U)
\end{tikzcd}$$

Figure 18 *< U, αu > is an F-system and h is a Homomorphism*

Since f and g are functions f : S → T and g : S → T in Set, there exists a Coequaliser h: T→ U in Set. Consider F (h) ∘αt : T → F (U)

F (h)∘ $\underline{\alpha t \circ f}$ ∧ F (f)∘ αs = $\underline{\alpha t \circ f}$ (Homomorphism condition)

⇒ F (h) ∘$\underline{\alpha t \circ f}$ = F (h) ∘ F (f) ∘ αs = F (h ∘ f) ∘ αs = F (h ∘ g) ∘ αs = F (h) ∘ $\underline{F(g) \circ \alpha s}$

∧ αt ∘ g = $\underline{F(g) \circ \alpha s}$ (Homomorphism condition)

⇒ F (h) ∘αt ∘ f = F (h) ∘ αt ∘ g

Because F (h)∘ αt ∘ f = F (h) ∘ αt ∘ g and h : T → U is a Coequaliser, there exists a unique function αu : U → F (U) making the diagram shown in Figure 18 commute.

The results of Coproducts and Coequalisers can be summarised in the following:

Let F : Set → Set be any functor. In the category Set $_F$ of F-systems, all Coproducts and all Coequalisers exist and are constructed as in Set. **(**)**

In most domains of reasonable complexity, an information system cannot have complete control over its environment. Its control is rather partial. This means that the same action performed twice in apparently identical circumstances might appear to have different effects. For this reason, we consider the use of Coequalisers in our information system.

In dynamic systems, an apparently identical action can be performed in two different ways regardless of the fact that it is performed under the same conditions (same query and same environment) and for instance, by the same agent and with the same goal.

$$\begin{tikzcd}
\mathbf{Sym} \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] \arrow[d, "\alpha s"'] & \mathbf{Tre} \arrow[r, "h"] \arrow[d, "\alpha t"] & \mathbf{Uni} \arrow[d, "\alpha u"] \\
\mathbf{Disease\ (Sym)} \arrow[r, shift left, "Disease\ (f)"] \arrow[r, shift right, "Disease\ (g)"'] & \mathbf{Disease\ (Tre)} \arrow[r, "Disease\ (h)"] & \mathbf{Disease\ (Uni)}
\end{tikzcd}$$

Figure 19 *< Uni, αu > is an Disease-system and h is a Homomorphism*

We use two different functions f and g to describe two different ways of performing the same action. For instance, we want to map the Symptoms sub-ontology to the Treatments sub-ontology and this can be done in two different ways described by functions f and g. However, we need to process and assemble the resulting information further with the ultimate goal of presenting it to the user in a meaningful way.

We use the principle of Coequalisers as there is a need to achieve an 'equalised' state for the agent, enabling it to continue with the assembling process. In Figure 19, this state is represented as Uni.

5 PULLBACKS AND THEIR COMPOSITION

5.1 Pullbacks in Set$_{Disease}$

***Definition 10** (of Pullbacks)* (Rutten, 2000):

The functor F : Set → Set preserve Pullbacks and the Pullbacks exist in Set $_F$

Let f : < S, αs > → < T, αt > and g : < S, αs > → < T, αt > be Homomorphisms. The Pullback of f and g in Set is represented in Figure 20. Here is P = {< s, u > | f (s) = g (u)}.

$$\begin{tikzcd}
\mathbf{P} \arrow[r, "\pi_1"] \arrow[d, "\pi_2"'] & \mathbf{S} \arrow[d, "f"] \\
\mathbf{U} \arrow[r, "g"'] & \mathbf{T}
\end{tikzcd}$$

Figure 20 *Pullback of functions f and g in Set*

Because F preserves Pullbacks, there exists a Pullback of F (f) and F (g) in Set (see Figure 21).

$$\begin{tikzcd}
\mathbf{F\ (P)} \arrow[r, "F\ (\pi_1)"] \arrow[d, "F\ (\pi_2)"'] & \mathbf{F\ (S)} \arrow[d, "F\ (f)"] \\
\mathbf{F(U)} \arrow[r, "F\ (g)"'] & \mathbf{F\ (T)}
\end{tikzcd}$$

Figure 21 *Pullback of F (f) and F (g) in Set*

Once specified, the SHDO template needs to be further partitioned into sub-ontologies and sub-sub-ontologies for the purpose of task sharing among different agents. We use Pullbacks to define and reason about this process.

In our information system, the SHDO template is viewed as a result of Pullback of functions f and g in Set$_{Disease}$. This template is further partitioned into, for instance, Symptoms and Causes sub-ontology templates for the purpose of sharing information retrieval tasks among different agents. This is shown in Figure 22.

$$\begin{tikzcd}
\mathbf{SHDO\ template\ (\mathcal{T}O)} \arrow[r, "\pi_1"] \arrow[d, "\pi_2"'] & \mathbf{Sym} \arrow[d, "f"] \\
\mathbf{Cau} \arrow[r, "g"] & \mathbf{Tre}
\end{tikzcd}$$

Figure 22 *Pullback of functions f and g in Set $_{Disease}$*

The resulting sub-ontology templates are given to different information agents responsible for retrieval of information specified by that particular template. For instance, one agent can be responsible for information regarding causes and another agent can be responsible for information regarding symptoms of a particular disease. Because Disease preserves pullbacks, a pullback of Disease (f) and Disease (g) in $Set_{Disease}$ can be represented as in Figure 23.

$$\begin{tikzcd}
\text{Disease (TO)} \arrow[r, "\text{Disease }(\pi_1)"] \arrow[d, "\text{Disease }(\pi_2)"'] & \text{Disease (Sym)} \arrow[d, "\text{Disease (f)}"] \\
\text{Disease (Cau)} \arrow[r, "\text{Disease (g)}"'] & \text{Disease (Tre)}
\end{tikzcd}$$

Figure 23 *Pullback of F (f) and F (g) in Set* $_{Disease}$

5.2 Compositions of Pullbacks of Functions and Functors in Set$_{Disease}$

Consider $\alpha s \circ \pi_1 : P \rightarrow F(S)$ and $\alpha u \circ \pi_2 : P \rightarrow F(U)$ in Figure 24.

$$\begin{tikzcd}
P \arrow[r, "\pi_1"] \arrow[d, "\pi_2"'] & S \arrow[d, "f"] \arrow[ddrr, "\alpha s"] & & \\
U \arrow[r, "g"'] \arrow[ddrr, "\alpha u"'] & T \arrow[ddrr, "\alpha t"] & & \\
& & F(P) \arrow[r, "F(\pi_1)"] \arrow[d, "F(\pi_2)"'] & F(S) \arrow[d, "F(f)"] \\
& & F(U) \arrow[r, "F(g)"'] & F(T)
\end{tikzcd}$$

Figure 24 *Compositions of Pullbacks of functions and functors in Set*

Because $F(f) \circ \alpha s \circ \pi_1 = \alpha t \circ f \circ \pi_1 = \alpha t \circ g \circ \pi_2 = F(g) \circ \alpha u \circ \pi_2$, there exists (by the fact that F (P) is a Pullback) a unique function $\alpha p : P \rightarrow F(P)$ such that $F(\pi_1) \circ \alpha p = \alpha s \circ \pi_1$ and $F(\pi_2) \circ \alpha p = \alpha u \circ \pi_2$.

Thus, $< P, \alpha p >$ is an F-system and π_1 and π_2 are Homomorphisms (Rutten, 1999).

$< P, \alpha p >$ is a Pullback of f and g in Set $_F$. As a consequence, the Pullback $< P, \alpha p >$ is a Bisimulation on S and $U : P \subseteq S \times U$ and the projections π_1 and π_2 are Homomorphisms.

F (P) is by assumption, a weak Pullback. As a consequence, there exists a transition structure $\alpha p : P\rightarrow F(P)$ such that π_1 and π_2 are Homomorphisms.

If F : Set→ Set be a functor that preserves weak Pullbacks and $f : < S, \alpha s > \rightarrow < T, \alpha t >$ and $g : < S, \alpha s > \rightarrow < T, \alpha t >$ are Homomorphisms of F- systems, then the Pullback $< P, \pi_1, \pi_2 >$ of f and g in Set is a Bisimulation on S and T . **(***)**

$$\begin{tikzcd}
(TO) \arrow[r, "\pi_1"] \arrow[d, "\pi_2"'] \arrow[ddr, "\alpha t"] & Sym \arrow[d, "f"] \arrow[ddr, "\alpha s"] & \\
Cau \arrow[r, "g"] \arrow[ddr, "\alpha c"'] & Tre \arrow[ddr, "\alpha t"] & \\
& \text{Disease (TO)} \arrow[r, "\text{Disease }(\pi_1)"] \arrow[d, "\text{Disease }(\pi_2)"'] & \text{Disease (Sym)} \arrow[d, "\text{Disease (f)}"] \\
& \text{Disease (Cau)} \arrow[r, "\text{Disease (g)}"'] & \text{Disease (Tre)}
\end{tikzcd}$$

Figure 25 *Compositions of Pullbacks of functions and functors in Set* $_{Disease}$

Similarly, we develop a statement for our information system. In Figure 25, we consider $\alpha s \circ \pi_1 : TO \rightarrow$ Disease (Sym) and $\alpha c \circ \pi_2 : TO \rightarrow$ Disease (Cau). Because Disease (f) $\circ\ \alpha s \circ \pi_1 = \alpha t \circ f \circ \pi_1 = \alpha t \circ g \circ \pi_2 =$ Disease (g)$\circ\ \alpha c \circ \pi_2$, there exists (by the fact that Disease (TO) is a Pullback) a unique function αt : TO→ Disease (TO) such that Disease $(\pi_1) \circ \alpha t = \alpha s \circ \pi_1$ and Disease $(\pi_2) \circ \alpha t = \alpha c \circ \pi_2$. Thus, $< TO, \alpha t >$ is an Disease-system and π_1 and π_2 are Homomorphisms.

$< TO, \alpha t >$ is a pullback of f and g in Set $_{Disease}$. As a consequence, the pullback $< TO, \alpha t >$ is a Bisimulation on Sym and Cau : $TO \subseteq Sym \times Cau$ and the Projections π_1 and π_2 are Homomorphisms.

Disease (TO) is by assumption, a weak Pullback. As a consequence, there exists a transition structure αt : TO→ Disease (TO) such that π_1 and π_2 are Homomorphisms.

6 FINALITY

Definition 12 *(of Final system)* (Rutten, 2000):

An F-system < P, π > is Final if for any F-system < S, αs > there exists precisely one Homomorphism f_S : < S, αs > → < P, π >. This is shown in Figure 26.

$$\begin{tikzcd}
S \arrow[r, "\exists!\, f_S"] \arrow[d, "\forall!\, \alpha s"'] & P \arrow[d, "\pi"] \\
F(S) \arrow[r, "F(f_S)"'] & F(P)
\end{tikzcd}$$

Figure 26 *< P, π > Final F-system for < S, αs > F-system*

In other words, < P, π > is a final object in the category Set $_F$.

Theorem 6:

Let < P, π > be a Final F-system (Final Coalgebra). Then is π : P → F (P) an Isomorphism.

Proof:

Because ((F(P), F(π)) is an F-system, there exists by the finality of P a unique Homomorphism f : ((F(P), F(π)) → < P, π >.

$$\begin{tikzcd}
P \arrow[r, "\pi"] \arrow[d, "\pi"'] & F(P) \arrow[r, "f"] \arrow[d, "F(\pi)"] & P \arrow[d, "\pi"] \\
F(P) \arrow[r, "F(\pi)"'] & F(F(P)) \arrow[r, "F(f)"'] & F(P)
\end{tikzcd}$$

Figure 27 *π : P → F (P) an Isomorphism*

Again by finality, the composition of the Homomorphisms π and f (see Figure 27) is equal to 1_P, since 1_P is also a Homomorphism. It follows from the fact that f is a Homomorphism that the reverse composition equals the identity on F (P).

In our system, the overall query to be answered is composed as the SHDO template. This template is further decomposed into the four sub-ontologies. This kind of decomposition is hierarchical so that these sub-ontologies are further decomposed into smaller sub-sub-ontologies and so on.

The sub-problems (sub-ontologies) identified during the problem decomposition phase are individually solved by information agents. Those solutions are then integrated into an overall solution by 'Smart' agents. With problem decomposition, this stage may be hierarchical with partial solutions assembled at the different levels of abstractions. The retrieved information is assembled into the SHDO template. When the assembly reaches its final state, the resulting FinSHDO (shown in Figure 28) is presented to the user.

$$\begin{tikzcd}
\text{SHDO template (TO): terms} \arrow[r, "fin"] \arrow[d, "\alpha s"'] & \text{FinSHDO template (FinTO): terms} \arrow[d, "\alpha f"] \\
\text{SHDO (SO): terms + attributes} \arrow[r, "Disease (fin)"'] & \text{FinSHDO (FinSO): terms + attributes}
\end{tikzcd}$$

$$\begin{tikzcd}
TO \arrow[r, "fin"] \arrow[d, "\alpha s"'] & FinTO \arrow[d, "\alpha f"] \\
Disease (TO) \arrow[r, "Disease (fin)"'] & Disease (FinTO)
\end{tikzcd}$$

Figure 28 *< FinTO, αf > Final Disease-system for < TO, αs > Disease-system*

We say that Disease-system < FinTO, αf > is Final if for any Disease-system < TO, αs > there exists precisely one Homomorphism fin : < TO, αs > → < FinTO, αf >, as shown in Figure 28. In other words, < FinTO, αf > is a final object in the category Set $_{Disease}$. A Final Coalgebra FinTO → Disease (FinTO) is a fixed point of the functor Disease.

7 NATURAL TRANSFORMATION AND HOMOMORPHISM MAPPING

7.1 Natural Transformation from Disease 1-functor to Disease 2-functor

Definition 13 *(of Natural Transformation)* (Rutten, 2000):

Let F: Set → Set and G: Set → Set be two functors. A Natural Transformation ν from F to G, denoted by ν: F → G, is a family {νs : F (S) → G (S) | S ∈ Set } of functions.

For any function f: S → T, the diagram from Figure 29 commutes.

$$\begin{tikzcd}
F(S) \arrow[r, "F(f)"] \arrow[d, "vs"'] & F(T) \arrow[d, "vt"] \\
G(S) \arrow[r, "G(f)"'] & G(T)
\end{tikzcd}$$

Figure 29 *Natural Transformation ν from F to G*

We may have a situation where a user is interested in the same information (for example, information about causes and treatments) regarding two different diseases. For each disease, we specify a functor Disease.

$$\begin{tikzcd}
Bipolar\ (Cau) \arrow[r, "Bipolar\ (f)"] \arrow[d, "vc"'] & Bipolar\ (Tre) \arrow[d, "vt"] \\
Depression\ (Cau) \arrow[r, "Depression\ (f)"] & Depression\ (Tre)
\end{tikzcd}$$

Figure 30 *Natural Transformation ν from Bipolar to Depression Disease-functor*

In our example from Figure 30, we have a user who is interested in information regarding Causes and Treatments of Bipolar Disorder and Depression. This is quite a common situation as the two diseases are related: bipolar disorder is characterised by alternating depressive and manic episodes. A user may want to compare information regarding those diseases and identify possible links. We have a similar situation for diseases that have more different types, such as diabetes. The user may want to compare the same information for different disease types.

We said that for two Disease-functors, Bipolar : Cau→ Tre and Depression: Cau→ Tre, Natural Transformation ν from Bipolar to Depression, denoted by ν : Bipolar→ Depression, is a family of following functions: { νc : Bipolar (Cau) → Depression (Cau) | Cau ∊ Set $_{Disease}$} and { νs : Bipolar (Sym) → Depression (Sym) | Sym ∊ Set $_{Disease}$}.

7.2 Mapping of Disease 1-homomorphism to Disease 2-homomorphism

Any F-system < S, αs > can now be viewed as a G-system by composing αs with νs (Rutten, 2000).

Moreover, if f : < S, αs >→ < T, αt > is an F-homomorphism then it is also a G-homomorphism of the resulting G-systems. Similarly, any F-bisimulation between F-systems is also a G-bisimulation of the resulting systems, as shown in Figure 31. The above is summarised in the following theorem.

$$\begin{tikzcd}
S \arrow[d, "\alpha s"'] & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "\alpha r"] & T \arrow[d, "\alpha t"] \\
F(S) \arrow[d, "vs"'] & F(R) \arrow[l, "F(\pi_1)"] \arrow[r, "F(\pi_2)"'] \arrow[d, "vr"] & F(T) \arrow[d, "vt"] \\
G(S) & G(R) \arrow[l, "G(\pi_1)"] \arrow[r, "G(\pi_2)"'] & G(T)
\end{tikzcd}$$

Figure 31 *Mapping of F- homomorphism to G-homomorphism*

$$\begin{tikzcd}
S \arrow[r, "f"] \arrow[d, "\alpha s"'] & T \arrow[d, "\alpha t"] \\
F(S) \arrow[r, "F(f)"] \arrow[d, "vs"'] & F(T) \arrow[d, "vt"] \\
G(S) \arrow[r, "G(f)"'] & G(T)
\end{tikzcd}$$

Theorem 7:

A Natural Transformation ν : F→ G between set functors induces a functor, denoted by ν (-) : Set$_F$ → Set$_G$ which maps < S, αs > to < S, νs ∘ αs > and an F-homomorphism f : < S, αs > → < T, αt > to the G-homomorphism f : < S, νs∘ αs >→ < T, νt ∘ αt >. Moreover, this functor preserves Bisimulations: for any s and t, s ~$_F$ t ⇒ s ~$_G$ t.

Identically, in our example, we have that Bipolar-system < Cau, αc > can now be viewed as a Depression-system by composing αc with νc (see Figure 32).

$$\begin{tikzcd}
Cau \arrow[d, "\alpha c"'] & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "\alpha r"] & Tre \arrow[d, "\alpha t"] \\
Bipolar\ (Cau) \arrow[d, "vc"'] & Bipolar\ (R) \arrow[l, "Bipolar\ (\pi_1)"'] \arrow[r, "Bipolar\ (\pi_2)"] \arrow[d, "vr"] & Bipolar\ (Tre) \arrow[d, "vt"] \\
Depression\ (Cau) & Depression\ (R) \arrow[l, "Depression\ (\pi_1)"'] \arrow[r, "Depression\ (\pi_2)"] & Depression\ (Tre)
\end{tikzcd}$$

Figure 32 *Mapping of Bipolar-homomorphism to Depression-homomorphism*

A Natural Transformation ν : Bipolar→Depression between set functors induces a functor, denoted by ν∘ (-) : Set $_{Bipolar}$ → Set $_{Depression}$ which maps < Cau, αc > to < Cau, νc ∘ αc > and an Bipolar-homomorphism f : < Cau, αc > → < Tre, αt > to the Depression-homomorphism f : < cau, νc ∘ αc >→ < Tre, νt ∘ αt >. Moreover, this functor preserves Bisimulations: for any c and t, c ~$_{Bipolar}$ t ⇒ s ~$_{Depresson}$ t.

8 CONCLUSION

In this paper, we used general definitions and principles of Coinduction to explain dynamic processes within the proposed ontology-based multi-agent system. We defined each user's query as an F-system and called F-coalgebra *Disease-coalgebra* for our system. S is analogous to the Specific Human Disease Ontology (SHDO) template that contains only terms. The template may cover information in different ontology 'dimensions' (disease types, symptoms, causes, treatments). The SHDO also contains attributes, values and other additional information and is analogous to F(S). The Disease-transition structure (dynamics of the system, function αs) is analogous to the F-transition structure and maps SHDO template to SHDO.

We assumed that each disease is uniquely characterised by the four ontology 'dimensions' and each 'dimension' is specific for a specific disease. If one 'dimension' is given for a specific disease, the other three 'dimensions' can be calculated. This means that, given a specified template for a particular disease, and if Disease-coalgebra (and αs) is used to map attributes to the Symptoms sub-ontology, we can then also use Disease-coalgebra (and αt) to map attributes to the Treatments sub-ontology of the SHDO template. This is called *Homomorphism f of Disease-system* is written as f : <Sys, αs> → <Tre, αt>. The collection of all Disease-systems together is a category which we denote by Set $_{\text{Disease}}$. This collection contains all possible templates requiring information for all possible combinations of the four 'dimensions' (or part of them).

We say that two sub-ontologies O1 and O2 are Bisimilar as there exists a Bisimulation R with $< O1, O2 > \in R$. We define *Disease-bisimulation* between Sym and Tre if there exists a Disease-transition structure $\alpha r : R \rightarrow F(R)$ such that the projections from R to Sym and Tre are Disease-homomorphisms. Generally, the principle of Disease-bisimulation is fulfilled within our information system, for all different sub-ontologies (or parts of them). We have Disease-bisimulation between Symptoms and Treatments, Symptoms and Causes, Treatments and Causes, Disease Types and Symptoms, Disease Types and Treatments and Disease Types and Causes sub-ontologies.

For a Bisimulation < R, αr >between two Disease-systems, we defined *inverse* R^{-1} of R. This principle of Bisimulation of the inverse R^{-1} of R is applicable between any two Disease-systems.

We showed that *composition and union* of Bisimulations is again a Bisimulation, as one important feature in the process of combining and assembling different data coming from different agents that need to be unified into one SHDO. We proved that the Bisimulation characteristics of Disease-systems of Set $_{\text{Disease}}$ do not change during complex processes of information retrieval.

Each query has its *Disease-relator* corresponding to its SHDO template. This Disease-relator maps from relationships between terms in the SHDO template to relations between the same terms and their attributes in the SHDO. Preserving the relationship order within the template is important in various processes during information retrieval.

During the process of information retrieval, the SHDO template is partitioned by 'Manager' agents into sub-ontologies and these sub-ontologies are further portioned into smaller sub-sub-ontologies for the purpose of task sharing among different information agents. The requested information is then retrieved and added onto these smaller partitions of the SHDO template by information agents. The resulting sub-ontologies and sub-sub-ontologies need to be correctly assembled into SHDO by 'Smart' agents. For this purpose, we defined *Coproduct* (or Sum) of two Disease-systems, for instance <Sym, αs> and <Tre, αt>. Identically, we can define Coproducts of other Disease-systems, such as for Cause and Disease types sub-ontologies: <Cau, αc> and <DTy, αd>.

In dynamic systems, an apparently identical action can be performed in two different ways regardless of the fact that it is performed under the same conditions (same query and same environment) and for instance, by the same agent and with the same goal. We defined *Coequalisers* to deal with these situations. In this way, an agent can calculate an 'equalised' state and continue the information retrieval process.

The SHDO template is viewed as a result of *Pullback* of functions f and g in Set $_{\text{Disease}}$. Function f maps Symptoms to Treatments sub-ontology while function g maps Causes to Treatment sub-ontology. The resulting sub-ontology templates are given to different information agents responsible for retrieval of information specified by that particular template.

Relevant retrieved information is assembled into the SHDO template. This step results in the SHDO. We defined *Final state* of the assembly process as FinSHDO. This FinSHDO needs to be presented to the user. < FinSHDO, αf > is a final object in the category $\text{Set}_{\text{Disease}}$.

We may have a situation where a user is interested in the same information (for example, information about causes and treatments) regarding two different diseases. For this purpose, we defined *Natural Transformation* ν from one to another Disease-functor (for example, from Bipolar to Depression Disease-functor). We also defined mapping of one to another Disease-homomorphism (for example, Bipolar-homomorphism to Depression-homomorphism).

REFERENCES

Aczel, P. (1988) 'Non-well-founded sets', *CSLI* (Center for the Study of Languages and Information) *Lecture Notes 14.*

Aczel, P., Mendler, N. (1989) 'A final coalgebra theorem', *Proceedings category theory and computer science, Lecture Notes in Computer Sciences 389*, pp. 357-365.

Barrett, T.B., Hauger, R.L., Kennedy, J.L., Sadovnick, A.D., Remick, R.A., Keck, P.E., McElroy, S.L., Alexander, M., Shaw, S.H., Kelsoe, J.R. (2003) 'Evidence that a single nucleotide polymorphism in the promoter of the G protein receptor kinase 3 gene is associated with bipolar disorder', *Molecular Psychiatry,* Vol. 8, pp. 546-557.

Cirstea, C. (2004) 'On logics for coalgebraic simulation', *Theoretical Computer Science,* Vol. 106, pp. 63-90.

Fernández, M., Gómez-Pérez, A., Juristo, N. (1997) 'METHONTOLOGY: From ontological art towards ontological engineering', *Proceedings of AAAI'97 Spring Symposium Series, Workshop on Ontological Engineering*, pp. 33-40.

Goldin, D., Thalheim, B., Srinivasa, S. (2000) 'Information systems = databases + interaction: Towards principles of information system design', *Proceedings of the 19th International Conference on Conceptual Modeling*, pp. 140-153.

Hadzic, M., Chang, E. (2005a) 'Medical ontologies to support human disease research and control', *International Journal of Web and Grid Service*, Vol. 1/2, pp. 139-150.

Hadzic, M., Chang, E. (2005b) 'Ontology-based multi-agent systems support human disease study and control', *Proceedings of the International Conference on Self-Organization and Adaptation of Multi-agent and Grid Systems*, pp. 129-141.

Hadzic, M., Ulieru, M., Chang, E. (2006) 'Soft computing agents for e-health applied to the research and control of unknown diseases', *Information Sciences (special journal issue on 'Softcomputing meets Agents'*), Vol. 176, pp. 1190-1214.

Rutten, J.J.M.M. (1999) 'Automata, power series, and coinduction: Taking input derivatives seriously', *Proceedings of the 26th International Colloquium on Automata, Languages and Programming*, pp. 645-654.

Rutten, J.J.M.M. (2000), 'Universal coalgebra: A theory of systems', *Theoretical Computer Science,* Vol. 249, No.1, pp. 3-80.

Rutten, J.J.M.M. (2003), 'Behavioural differential equations: A coinductive calculus of streams, automata, and power series', *Theoretical Computer Science,* Vol. 308, No. 1-3, pp. 1-53.

Wegner, P., Goldin, D. (1999) 'Interaction as a framework for modeling', *Conceptual Modeling - Current Issues and Future Directions, Lecture Notes in Computer Science 1565,* pp. 243-257.

Wegner, P., Goldin, D. (2005), 'Interaction, computability, and Church's thesis'. Retrieved: 24 May, 2005, from: http://www.cs.brown.edu/people/pw/papers/bcj1.pdf.