1-1-2014

Reshare an Operational Ontology Framework for Research Modeling, Combining and Sharing

Mohammad Al Boni

Reshare: An operational ontology framework for research modeling, combining and sharing

By

Mohammad Al Boni

A Thesis
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in Computational Engineering
in the Bagley College of Engineering

Mississippi State, Mississippi

August 2014

Reshare: An operational ontology framework for

research modeling, combining and sharing

By

Mohammad Al Boni

Approved:

Roger L. King
(Major Professor and Graduate
Coordinator)

Derek T. Anderson
(Committee Member)

Song Zhang
(Committee Member)

Jason Keith
Interim Dean
Bagley College of Engineering

Name: Mohammad Al Boni

Date of Degree: August 15, 2014

Institution: Mississippi State University

Major Field: Computational Engineering

Major Professor: Roger L. King

Title of Study: Reshare: An operational ontology framework for research modeling, combining and sharing

Pages of Study: 69

Candidate for Degree of Master of Science

Scientists always face difficulties dealing with disjointed information. There is a need for a standardized and robust way to represent and exchange knowledge. Ontology has been widely used for this purpose. However, since research involves semantics and operations, we need to conceptualize both of them. In this thesis, we propose ReShare to provide a solution for this problem. Maximizing utilization while preserving the semantics is one of the main challenges when the heterogeneous knowledge is combined. Therefore, operational annotations were designed to allow generic object modeling, binding and representation. Furthermore, a test bed is developed and preliminary results are presented to show the usefulness and robustness of our approach. Moreover, two aggregation techniques for fusing ontology matchers are investigated as an initial work for building an algorithm which converts descriptive ontologies into operational ones.

Key words: ontology, generic modeling, dynamic binding, object oriented, generic visualization, ontology matching

DEDICATION

To my beloved country, Syria.

ACKNOWLEDGEMENTS

Iwould like to thank those who contributed to enrich this research

I thank Dr. Derek T. Anderson for the extended and the helpful discussions concerning ontology matching.

I would like to acknowledge and thank the Fulbright scholarship program for its financial support of the work performed herein.

Last but not least, I thank my committee for their comments on this thesis, and I thank Roger L. King for directing this research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Scientists have always been facing the challenge of science formalization.Research is being conducted in every knowledge domain. A new and novel methods and techniques are found, old ones become outdated because they are not fully compatible with the new proposed concepts. Also, scientists are working on similar problems and their work is eventually presented in literature, which is the only way to explore these new contributions. As a result, researchers do not know about the ongoing research that is conducted by others. Therefore, there is a demand for a standardized formalization to describe and share both old and current research. With the emergence of the internet, several standard formalization techniques were proposed. Most of them are based on extensible markup languages (XML), and one of these languages provides a good solution for this problem. Ontology is a knowledge representation language that allows computer-based agents to understand and handle a shared domain-based conceptualization. Ontologies are very easy to share and exchange. Ontologies have been used in many fields such as in semantic web, biology (e.g., the gene ontology [4]) and even in software engineering where it has been used with each phase of the software development life-cycle: analysis, design, implementation, integration, maintenance and retirement [39]. Moreover, Ontology allows computer-based agents

to understand and handle a shared domain-based conceptualization [17]. Descriptive ontologies have been used to describe the semantics (structures, meanings and relations) of a certain problem domain. However, such ontologies have created another challenge when dealing with heterogeneous knowledge domains where some ontologies may overlap or differ in the structure of the semantics. The variance between ontologies happens because of the subjectivity of using ontologies, i.e., it depends on the creators and how they are describing it. Consequently, a need for standardized ontologies emerges. Pease et al.[34] proposed an upper ontology from which others can inherit and extend their own concepts. Even though the Suggested Upper Merged Ontology (SUMO) helped dealing with heterogeneous knowledge from different sources, specific details still differ. Also, such upper ontologies increase the complexity of the structure.

Research experiments contain not only semantics, which can be effectively defined and described using regular ontology, but also they contain operations (datasets, source code, analysis, experiments, and/or results). Therefore, an operational ontology needs to be employed instead of the regular descriptive one. Furthermore, to insure robustness and scalability, we need to design our ontology in a generic way so that it can be easily extended without the need for any change in the core concepts. As a result, we considered the use of a software object oriented (OO) model to insure a high level of abstraction that is adaptable to multiple research domains. To tackle all these challenges, we proposed ReShare, an operational ontology framework for research modeling, combining and sharing that has the ability to design an operational model for any research domain. The framework is mainly divided into three phases: modeling a research, applying operations on that model, and

visualizing results. Moreover, in order to deal with the existing heterogeneous ontologies, several ontology matching techniques (supervised and unsupervised) are proposed. The main contribution of this work includes proposing ReShare , building two different mechanisms for ontology matching, creating two generic visualization tools, 3D dynamic scatter plot and parallel coordinates, and finally, developing a test bed to experiment the applicability of ReShare. Using this test bed, we modeled a research study case in the field of material sciences, conducted by AbuOmar et al. [2]. In that research, the authors analyzed the vapor-grown carbon nanofiber (VGCNF)/vinyl ester (VE) nanocomposites dataset in order to discover some hidden trends, patterns, and properties associated with this material system. A summary of the statistical experimental design and testing procedures to generate the VGFCNF/VE dataset is given later in this paper. A more detailed discussion is provided in [30, 32, 31].

CHAPTER 2

RELATED WORKS

A number of works have been put forth in the literature regarding the OO modeling using ontologies. Most of these studies focus on static objects modeling and descriptive representation using ontologies. One model was introduced by Siricharoen (2006) where an object has an identity, state and functions. It could be inherited from another object or associated with one or more objects. Values are stored in objects using attributes and properties. These attributes can be primitive types (string, integer ... etc), references to other objects or a set of values of these types [38]. This model was extended and detailed by Siricharoen (2009) to include a name, a set of attributes and operations. Each attribute has a name, type and visibility. On the other hand, operations may contain comments, conditions and/or initial values. Each object may have one or more relationships with other objects such as supertype relation [37]. Another model was described and proposed by Batanov and Vongdoiwang (2007): in which an object model consists of four kinds of entities was proposed: object/class, attributes/properties, methods/operations/functions and relations/associations. In addition, they developed an algorithm that converts a text-based description model of an object to an XML object model through several intermediate steps [5]. Other studies are concerned with the mapping between ontology and the OO model.

Evermann and Wand (2005) proposed a set of rules that set the foundation of mapping between objects and an ontology including attributes, associations, composition and aggregation [12]. These works present effective mechanisms for modeling and mapping objects. However, the resultant ontologies, from all of these studies, contain annotations that describe the structure and the associations of objects. None of them define fixed annotations that model any kind of knowledge. Therefore, different research domains will have different object models and different descriptive ontologies. To the best of our knowledge, a high level ontology that can model generic objects has not been reported in the literature.

Ontology was also used in many data source related applications, especially with online data sources and databases. One of the applications was Knowledge Bus in which an application-focused databases were generated from large ontologies such as Cyc and XSB [35]. Also, an API interface was generated to give the users the ability to manipulate the generated databases. Another study was presented by Gali, et al [14]. The authors developed an algorithm for converting descriptive ontology into a relational database. This algorithm created tables which correspond to ontology concepts. Then, relations had been established to specify concepts' associations. Furthermore, a similar work was established by Sebestyenova [36] where ontologies were converted not only to a relational database, but also to an OO one. The established databases were used in decision support systems. These models are very suitable for dealing with data sources. However, the outcome of these mechanisms is always descriptive. Because of this, handling and combining data sources from different research domains have become a very difficult task. Moreover, none of these works define a robust model that can be extended to work with new data sources

that may be introduced in the future. In this paper, we are addressing these problems by defining a standardized high level operational ontology. This ontology can be used to model and combine knowledge form any research domain. Also, this ontology can be extended to include new technologies such as new database engine, without the need for any change in the core concepts.

CHAPTER 3

RESHARE ONTOLOGY FRAMEWORK

In this chapter, the annotations of ReShare framework are presented. The annotations are mainly divided into seven correlated categories: Generic object, data source, data binding, operation, workflow, visualization and visualization channel. Figures 3.1, 3.3 and 3.4 show the annotations that defines the framework while figure 3.2 shows a detailed description of the object and data type properties.

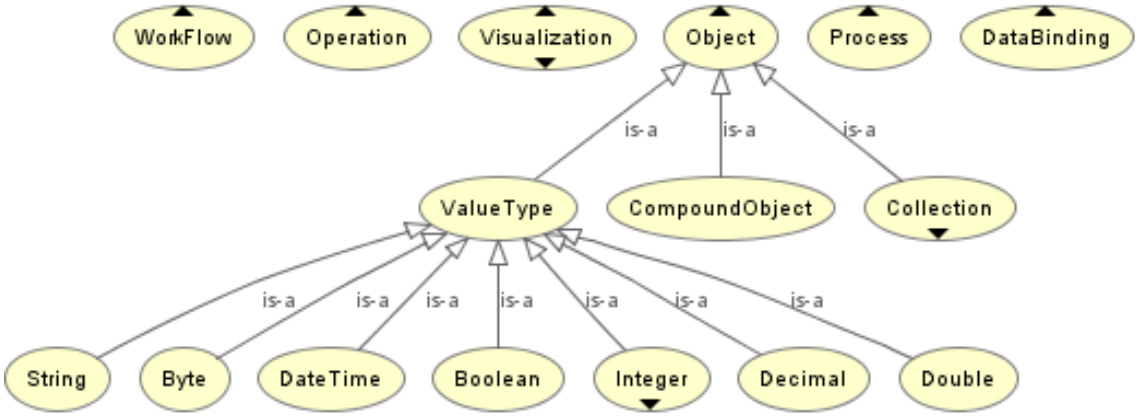
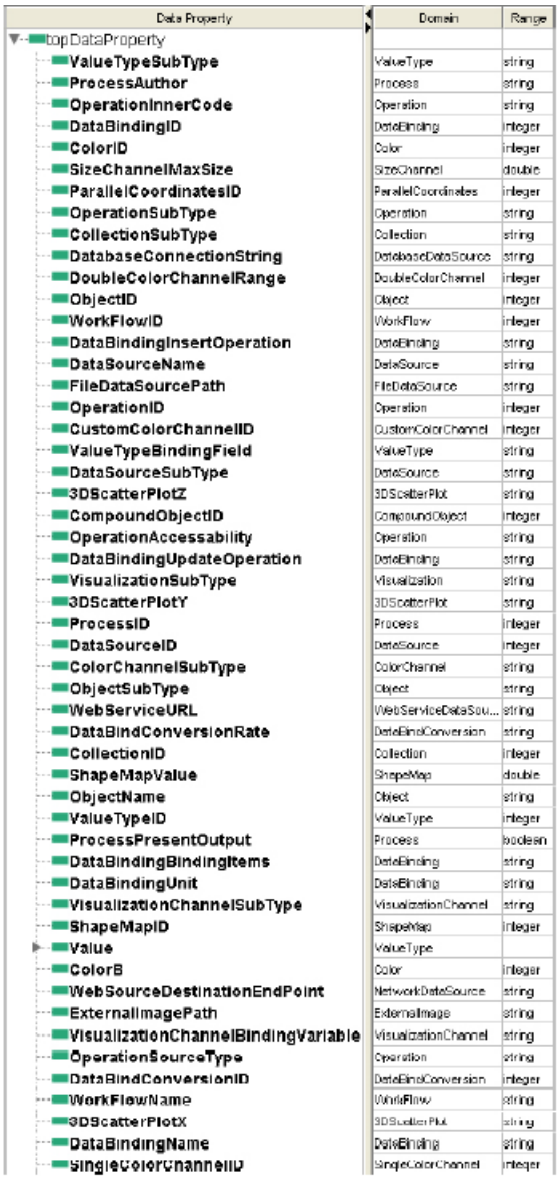
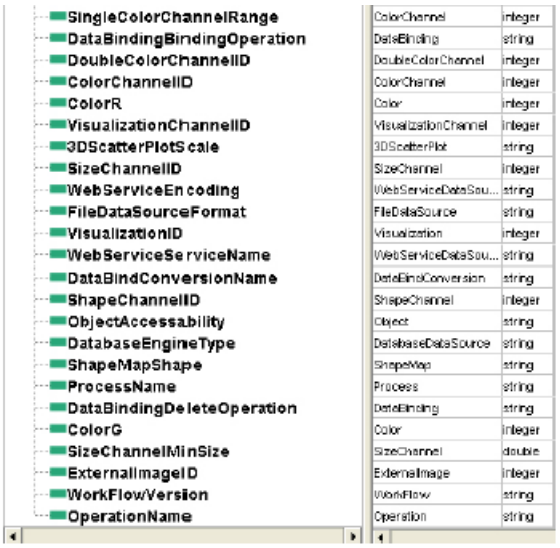


Figure 3.1

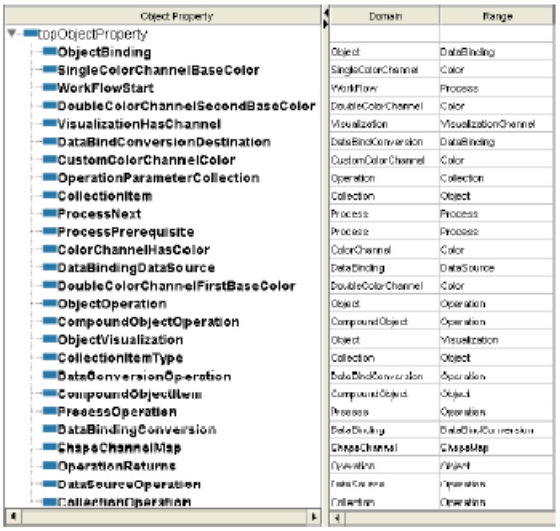
The main annotations in the ReShare ontology framework.



Data property matrix.



Data property matrix Cont.



Object property matrix.

Figure 3.2

Object and data property matrices.

3.1 Generic Object Model

In order to model a generic object for all objects used in any problem, we employed the software OO ideology in our framework. This ideology is generic, robust, dynamic, simply designed and easily understood. The annotation "object", shown in Figure 3.1, works as the parent for all other instances derived from it. Objects can be associated with one or more "DataBinding", "Operation" and/or "Visualization". Operations are used to define annotations for a function that can be performed on objects. DataBindings describe the mechanism of binding the object with one or more data sources. Visualizations specify the techniques through which objects are visualized. "DataBinding", "Operation" and "Visualization" annotations will be discussed in details later in this chapter. Objects can be of different types (value, collection or compound). First, value type is a variable that can hold a primitive value such as string, integer, boolean ... etc. Another object type is a collection that describes a set of elements and it can be from different data structures (array, linked-list, queue, stack, tree ... etc). In the collection, we can identify the "ItemType" which is an instance of the object annotation, e.g., we can model a collection of values, a collection of compound objects or a collection of collections. Finally, compound object represents a domain-based object that contains several items, each can be any of the types mentioned above. The main motivation behind using the software OO ideology, herein, is its robustness. Any change to the object model will not require a lot of adaptation. For example, if we have a collection of compound objects and we add a new field to the compound object, then the only required adaptation is adding new annotations for the new field. Afterwards,

the change will take effect in place, and the collection will contain the compound object's new design without any change to its annotations or associations.

In Tables 3.1, 3.2 and 3.3 a sample model of a student is presented to demonstrate the object model. This student has three value-type attributes: {StudentID, StudentFirstName, StudentLastName}, one compound object {StudentAddress} and one collection of courses {StudentCourses}. The courses that are taken by that student are compound objects. Also, the address of the student is a compound object that consists of six value-type attributes: {Country, State, City, Street, Building, Zipcode}. Likewise, Course is a compound object and it has four value-type attributes: {CourseName, CourseCode, CourseHours, Grade}. This high-level and easy-to-use structure provides the required robustness for the system where we can simply modify the objects during run-time. For example, we can add fields to the student such as (nationality, registration date ... etc). This change will not affect the system and will not require any change to the core system. On the contrary, we only need to add bindings for the new fields and it will be bound to the original fields.

3.2 Operation Model

The proposed operation model represents a generic function that takes a collection of parameters and returns an object (value, collection or compound) as an output. Operations can be of different categories such as read, insert, update, delete, operational, analytical ...etc. The operation source code can be written in any coding language (C++, C#, Java, Matlab ... etc). Operations can take a collection of objects as parameters. This collection of the parameters can be defined using the same object models discussed previously in

Table 3.1

Student object

Item	**Object attributes**		**Valuetype attributes**	
	SubType	**accessibility**	**SubType**	**Value**
StudentID	ValueType	public	integer	1
StudentFirstName	ValueType	public	String	'John'
StudentLastName	ValueType	public	integer	'Smith'
	Object attributes		**Collection attributes**	
	SubType	**accessibility**	**SubType**	**ItemType**
StudentCourses	Collection	public	Array	Course
	Object attributes		**Compound attributes**	
	SubType	**accessibility**		
StudentAddress	Compound	public		

Table 3.2

Student address object

Item	**Object attributes**		**Valuetype attributes**	
	SubType	**accessibility**	**SubType**	**Value**
Country	ValueType	public	string	'United States'
State	ValueType	public	string	'CA'
City	ValueType	public	string	'Los Angeles'
Street	ValueType	public	string	'1419 Westwood Blvd'
Building	ValueType	public	string	43
ZipCode	ValueType	public	integer	90024

Table 3.3

Student object

Item	**Object attributes**		**Valuetype attributes**	
	SubType	**accessibility**	**SubType**	**Value**
CourseName	ValueType	public	string	'Intro to Algorithms'
CourseCode	ValueType	public	string	'CSE4833'
CourseHours	ValueType	public	integer	4
Grade	ValueType	public	integer	93

section 3.1. Although parameters can be defined through the ontology, some parameters can also be given by the user at run-time.

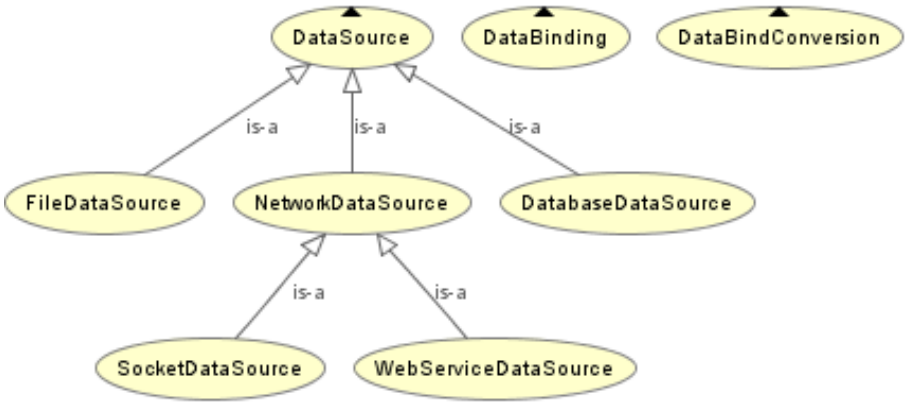


Figure 3.3

The data source model annotations.

3.3 Data Source Model

The proposed data source model, shown in Figure 3.3, is extendable, easy to use and gives the ability for on-the-fly dynamic data binding. All data sources will be handled through one root node "DataSource". Data sources can be of different types such as "File-DataSource", "NetworkDataSource", "DatabaseDataSource" or it can be any data source nodes added for future use. Moreover, using the operation scheme (discussed in section 3.2), we can define all the operations that are required to give an interface for that data source. Operations can be of any type: Read, Write, Update or Delete. Multiple operations of each type can also be defined. For example, we can define two operations to read a student's record from the database. The first operation filters students by their ID

while the other one filters them by their name. Each sub-node can define its own data properties. For an instance, “DatabaseDataSource” has two data properties {EngineType, ConnectionString}. Likewise, “FileDataSource” also has two data properties {FilePath, SourceFormat}. This model can easily be extended. Whenever a new data source is added, we only have to plug it in a suitable location in the data sources tree and define the operations that allowing the system to extract and manipulate data to/from it.

3.4 Data Binding Model

In this section, we will discuss the binding between the object model and the data sources. The data binding model consists of two main concepts “DataBinding” and “Data bindConversion”.

Data binding relates objects to a certain data source and defines the operations which need to be called from the data source in order to extract and manipulate the corresponding data. Also, each data binding can define a specific unit. Using the unit and the data bind conversion annotations, we can combine and compare data from different data sources. For example, we can compare the temperature of two materials, each one is stored in a different data source and each has its own different unit. Moreover, since we can associate data bindings with specific object models or items of the object models, we can partially bind objects from different data sources and combine the results to create complete objects.

Since data can be bound from different data sources, conversion mechanisms are required to determine the relation between data entries that refers to the same entity, but is represented by using different formats or units. Databind Conversion is used to specify

this relation. For example, a conversion node can define the rate between two temperature entities such as Celsius and Fahrenheit. Also, another node can define how to convert a student grade from 100-value-based format to letter-value-based format. In this case, we can define an operation that can carry out the conversion such as an operation that will take the 100-value-based grade and convert it to letter-value-based through specifying each latter range and the thresholds that define a start and an end for that range.

In Tables 3.4 and 3.5, a demonstrative example is presented to show how objects can be bound with different data sources on-the-fly. Student object (see section 3.1) will be bound with two different databases and a text file. The first database, "StudentDB", contains data for most of the fields in the Student, Course and Address objects. For the sake of demonstration, the Course's field, "Hours", is stored in a different database (SQLite[1]). The filtering has been done using the course ID as a mapping condition between the data extracted from the two different databases. Likewise, the grades of students are extracted from the text file where student ID and course code have been used as a filter for the data.

3.5 Workflow Model

A study may include a number of workflows, each represents a certain experiment. Each workflow may consist of many steps and these steps can be sequential or concurrent. In order to satisfy these requirements, two annotations were used ("WorkFlow" and "Process"). A workflow will start from certain processes, and a dependency diagram is determined by specifying each process's prerequisites and its next processes. Each process executes an operation designed according to the operation model and returns an object

[1]SQLite is a software library that implements transactional SQL database engine. http://www.sqlite.org/

Table 3.4

Three different data sources for student's object.

Item	**SubType**	**attributes**	
		attribute	**Value**
StudentDB	Database	EngineType	'SQLServer'
		ConnectionString	'Data Source=.\SQLEXPRESS;Attach DbFilename=...'
CourseDB	Database	EngineType	'SQLite'
		ConnectionString	'Data Source=CourseDB;...'
grades.txt	File	Path	'D:\...\grades.txt'
		SourceFormat	'txt'

Table 3.5

Mapping between student object and data sources.

Object	**Field**	**Data Source**		
		StudentDB	**CourseDB**	**Grades.txt**
Student	ID	✓		
	FirstName	✓		
	LastName	✓		
Address	Name	✓		
	State	✓		
	City	✓		
	Street	✓		
	Building	✓		
	ZipCode	✓		
Courses	Country	✓		
	Code	✓		
	Hours		✓	
	Grade			✓

(value, collection or compound). This object can be fed as an input for the next process. Also, processes can visualize their results using external visualization engines.

3.6 Visualization Model

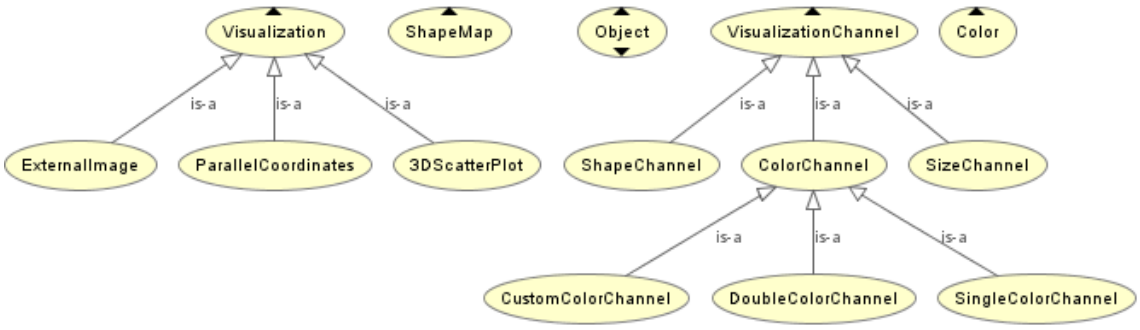


Figure 3.4

The annotations that describe the visualization model.

The Object model, presented in section 3.1, is designed according to the software OO methodology to allow a generic modeling. As a result, the visualization should take into consideration that generic nature of the object. The visualization model, shown in Figure 3.4, is built in a very interactive and dynamic way. Also, this model can be extendable to include any visualizations and visualization channels that will be added for future use. All visualizations will be handled through one root node “Visualization”. Visualizations can be of different types such as “ParallelCoordinates”, “3DScatterPlot” or “ExternalImage”. Each visualization can have one or more channels. Visualization channels include annotations for shape, size and color channels and visualization properties are also defined

in the ontology. Both parallel coordinates and 3D scatter plot visualization techniques are discussed in detail in chapter 4.2.

CHAPTER 4

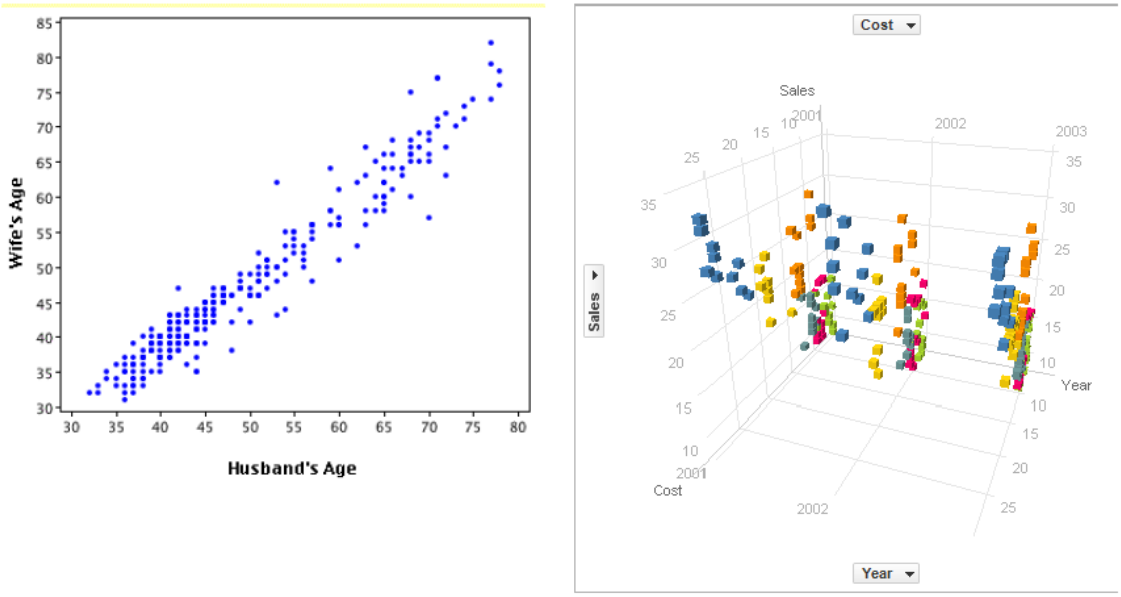
DYNAMIC VISUALIZATION

Multivariate visualization is a very challenging task. There is a high demand for effective multivariate visualizations since a huge amount of multivariate data exists. In this chapter, two different visualization techniques (3D scatter plot and parallel coordinates) are presented and compared. These visualizations are implemented in an interactive way to allow the user to gain full control on what and how to do the visualization. Also, several different visualization channels are used to support this task such as color maps, shape and size to enhance the visualization. Moreover, several trend detection mechanisms are also implemented to allow the user to understand the correlation between different data dimensions.

4.1 3D scatter plot

Scatter plots are effective visualization tools which are used to visualize discrete data in the Cartesian space. 2D scatter plots represent the relation between two variables (x,y) in a data set. Figure 4.1a shows an example of a 2D scatter plot of age values of a husband compared to his wife. It can be inferred from this plot that the age relation between married couples tends to be linear, i.e., both wife and husband are of the same age. On the other hand, 3D scatter plots are extended from the 2D ones where we can compare three different

variables together in one 3D Cartesian space (x,y,z). For example, in Figure 4.1b, sales data are visualized by using 3D scatter plot. This plot can be used to find the relation between the sales values, year of sales and cost.



(a) 2D scatter plot of Age.

(b) 3D scatter plot of sales data

Figure 4.1

Examples of scatter plot with 2 and 3 Dimensions.

The main disadvantage of scatter plots is their limitation to visualize a large number of variables. In this section, an interactive extension of the 3D scatter plot is presented to overcome the limitations of the static scatter plots. In order to increase the number of control variables, an extra scale variable with several visualization channels is used. Also, the user can dynamically bind each visualization channel with any of the variables which considerably increases the number of effective plots that can be extracted from the data.

Note, all of the following examples are built using the vapor-grown carbon nanofiber (VGCNF)/vinyl ester (VE) nanocomposites dataset (described in chapter 5.4).

4.1.1 Visualization Channels

4.1.1.1 Size Channel

First, size channel can be bound with a certain variable and, therefore, different values can be visualized with different point sizes. Figure 4.2 shows the effective use of the size channel to visualize different values of the VGCNF Weight Fraction variable at different points in the space. The user can dynamically specify the minimum and the maximum sizes of the values which are linearly normalized using equation 4.1. The normalization step returns a value between the range [0,1]. Then, this value is mapped to the new range using equation 4.2.

$$x_{norm} = \frac{x - Var_{min}}{Var_{max} - Var_{min}} \tag{4.1}$$

$$x_{scaled} = x_{norm} \cdot (NewRange_{max} - NewRange_{min}) + NewRange_{min} \tag{4.2}$$

4.1.1.2 Color Channel

Second, color channel is a very important visualization channel in visualizing both discrete and continuous variables. Three different color maps were developed and each map suits a certain variable type.

1. **Single Colored maps:** In these maps, a range of intensities between a color and the white color is developed. Using the HSB model, the user can specify a certain color, and a range of colors is built.
2. **Double Colored maps:** This one is the same as the previous one, but it uses doubled colors interface. Instead of using a gradual change from white to the selected color,

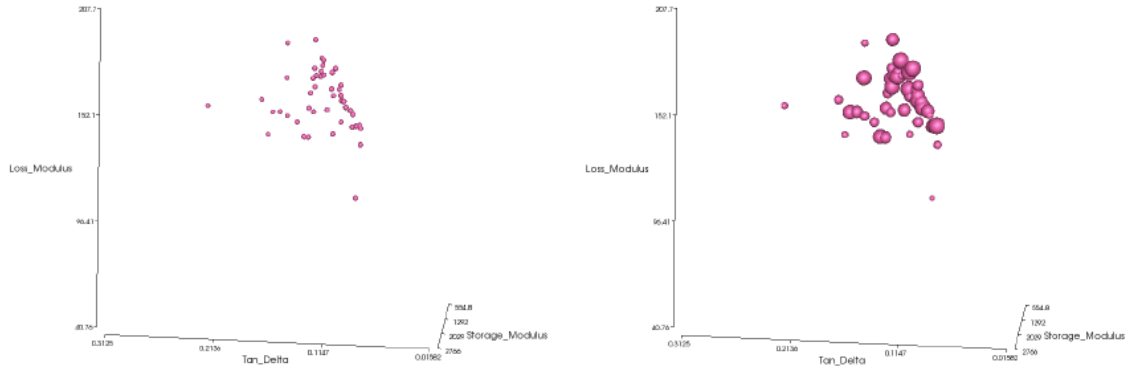


Figure 4.2

The effect of using the size visualization channel.

the user can choose two colors and the system generates the range of colors between them. This type of color maps is very effective to visualize variables with both negative and positive values such as Temperature. In such variables, the use of single color map is not effective and does not represent the range of values.

3. **Custom Color maps:** In this map, the user can select colors and build a map from these colors. The custom color map is very effective to visualize nominal variables. Figure 4.3 provides an illustration of the three color maps to visualize the VGCNF Weight Fraction variable.

4.1.1.3 Shape Channel

Third, Shape map is another effective visual channel which represents nominal Values. The user can dynamically bind values to certain shapes. Figure 4.4 shows an example of a shape map used to visualize different Mixing Methods.

Finally, Figure 4.5 represents a variation of visualization channels used to visualize the dataset.

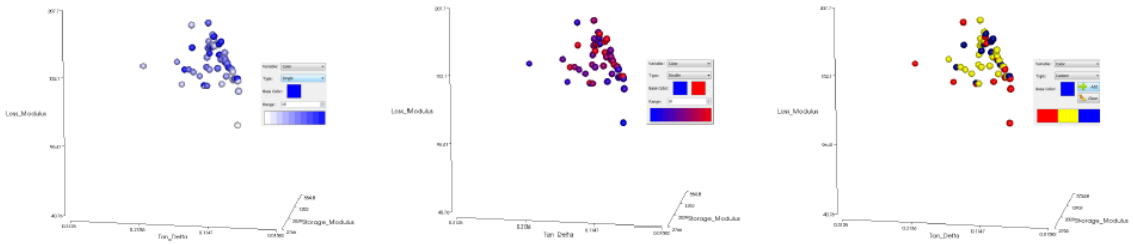


Figure 4.3

The effect of using different color maps.

Figure 4.4

The effect of using shape maps.

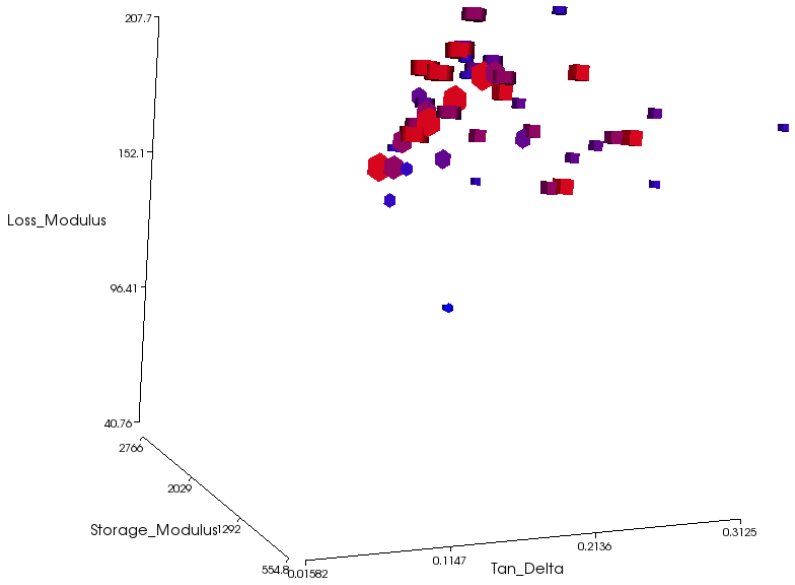


Figure 4.5

The effect of using different visualization channels simultaneously.

4.1.2 Data Trend Analysis

Data trend is the direction in which the data points tend to change by time. Finding the trend of the data is a very important and challenging task. Trend can be used to understand the hidden patterns in the data, find associations between variables and/or predict new data points based on the existing ones. In this model, the time scale can be replaced by any other variable which helps in studying how the data will behave when one of the variables is changing. At any given time, the user can show the data points at a previous scale values and can highlight the current ones as well (see Figure 4.6).

The main concept in finding the data trend is clustering. The user can define the desired number of clusters and then, at each scale value, the data is divided into a set of clusters of which the trend is calculated. Figure 4.7 shows the trend of the data set with respect

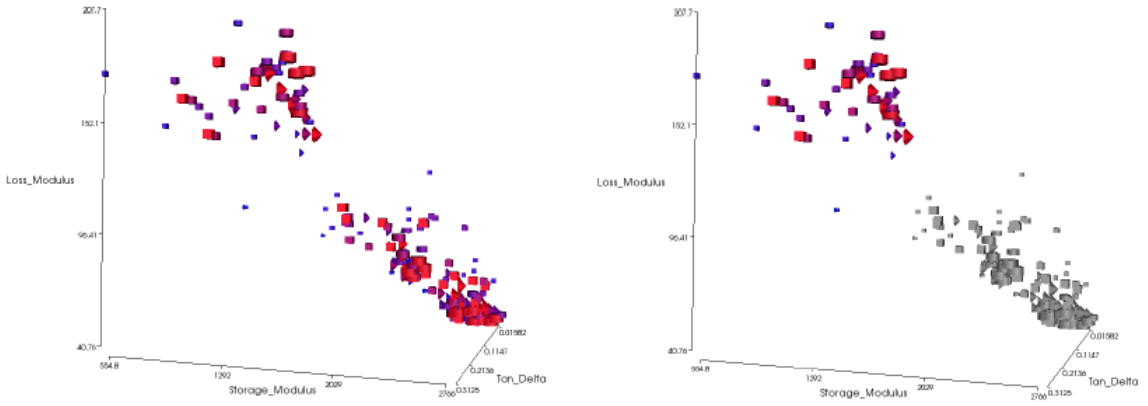


Figure 4.6

Visualize different data points from different scale values.

to the temperature scale. Also, the user can hide data points to only study the data trend.

Figure 4.8 shows a calculated trend on the same data set using two clusters.

Two different algorithms were developed to find the data trend:

- Cluster Based: This algorithm goes through the following steps:
  1. The user specifies the number of clusters (number of trends).
  2. If the number of clusters is 1, then there is no need for clustering:
     (a) At each scale value, the average coordinates (x,y,z) are calculated.
     (b) The data trend is a set of arrows that connects these averages.
  3. If the number of clusters is greater than 1:
     (a) At each scale value, the data points are divided into clusters using the Cmeans clustering algorithm provided by the vtk [1].
     (b) Then for each cluster, the average coordinates (x,y,z) are calculated.
     (c) The clusters' averages are connected using one of the linking methods described later in this section.
- Variable Based: To overcome the problems resulted from linking clusters in different scale values, a variable based approach is proposed. In this approach, the user selects a variable and the data points will be clustered based on that variable regardless to the number of clusters. Each value from the selected variable will represent a cluster. Figure 4.9 shows the use of variable based trend on the same data set and scale variable. It is clearly observed that this approach can accurately find the data trend.

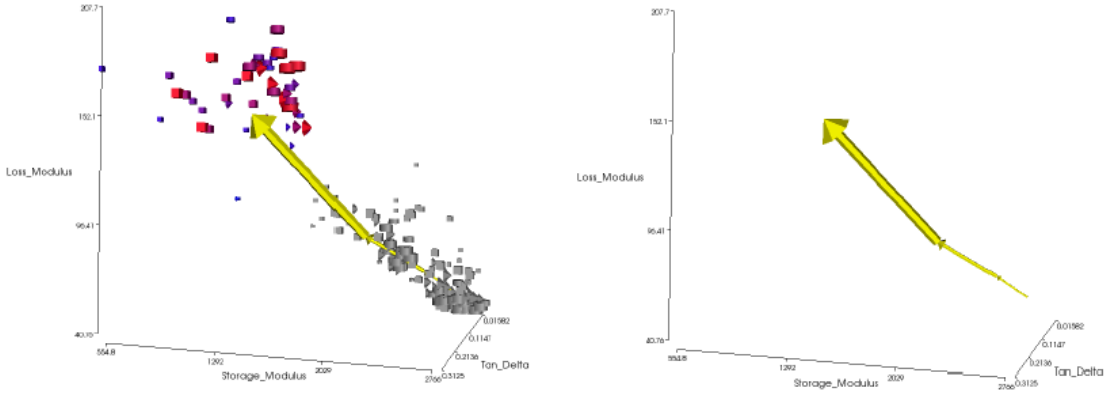


Figure 4.7

Data trend on data points with respect to temperature as a scale variable.

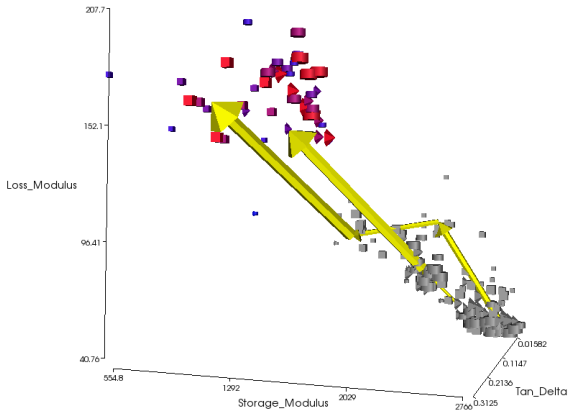


Figure 4.8

Data trend on data points with respect to temperature using two clusters.

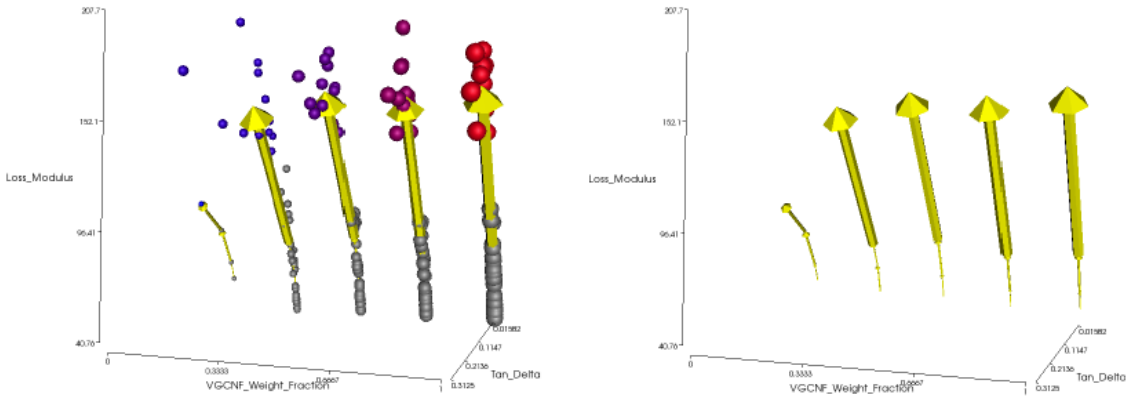


Figure 4.9

Data trend using variable based approach.

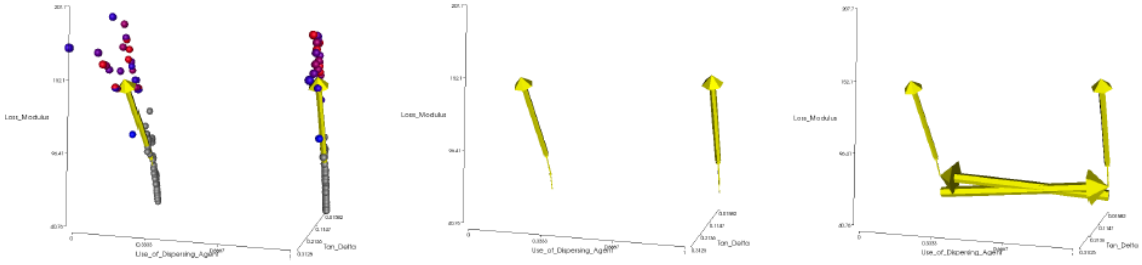


Figure 4.10

Different linking methods are used to connect clusters centers.

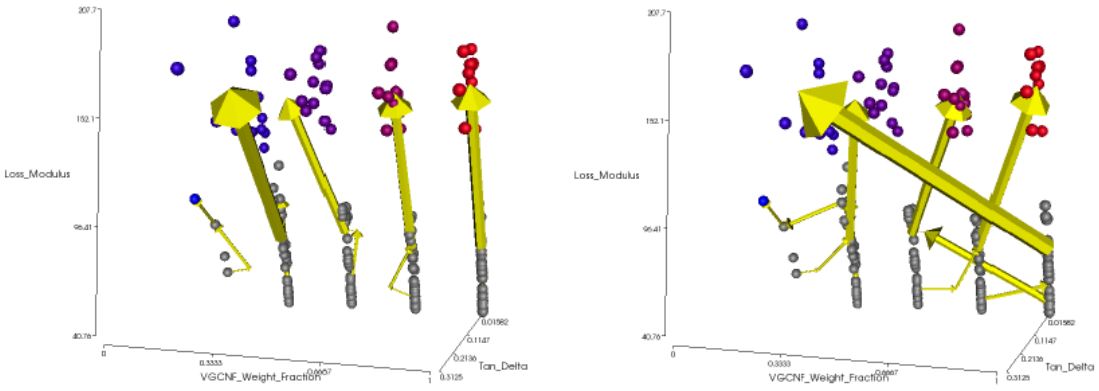


Figure 4.11

Random linking compared to closest linking with data trends using 5 clusters.

4.1.2.1 Linking Methods:

At each scale value a set of points (represent the center of each cluster) are calculated. However, there is no prior knowledge on which point from the first scale value will be connected to which point from the second and so forth. Two different linking methods were developed to connect these points:

1. Random Linking: Herein, the points are connected based on the order of the clusters (which is random). Therefore, this algorithm may not always show the desired results (see Figure 4.10).
2. To Closest Linking: In this method, the clusters' centers from one scale value are connected to the closest ones in the next scale value. The method is useful to overcome the random selection. However, linking to the closest point does not necessary means that both points belongs to the same trend, especially when using a larger number of clusters (see Figure 4.11).

4.2 Parallel Coordinates

Parallel coordinates visualization is a multivariate visualization in which a set of parallel axes represents a set of variables and each of the data points are visualized using a

line between these axes. Different variations of parallel coordinates can be found in [8]. In this section, parallel coordinates with dynamic color maps and data binding is presented. The same color visualization channel, presented in section 4.1.1.2, is applied to the parallel coordinates to enhance the readability of the data points. Figure 4.12 shows an example of the parallel coordinates applied on the dataset. The color map herein is also bound to a certain variable. Moreover, the order of the coordinates can be changed dynamically to examine the direct relation between two specific variables.

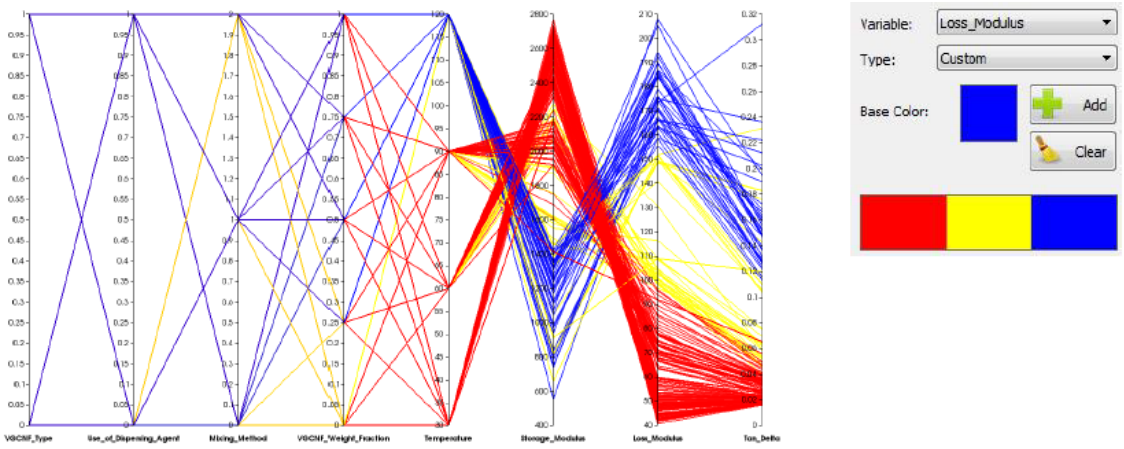


Figure 4.12

Parallel coordinates with a custom color maps.

CHAPTER 5

CASE STUDIES

In this chapter, a research experiment conducted by AbuOmar et al. [2] is modeled using the proposed ReShare annotations.This example will help to understand the correlation and associations between all categories of annotations discussed in section 3.6. In this chapter, the experiment is briefly described in terms of experimental design, the corresponding ReShare model is built and the outcome of this experiment is shown. Moreover, in order to study the robustness of ReShare, the data model is combined with a different model from different experiments in the same field [42, 24]. Then, results from the extended model are also shown in this chapter.

5.1 Experimental Design

The effect of five input design factors on the viscoelastic properties (storage, loss modulus and tan delta) of VGCNF/VE nanocomposites were investigated using a general mixed-level full factorial experimental design [29]. These carefully selected factors, based on the state-of-the-art formulation and processing procedures, included:

1. VGCNF type (designated as A),
2. use of a dispersing agent (B),
3. mixing method (C),
4. VGCNF weight fraction in parts per hundred parts of resin (phr) (D), and

5. the temperature (E) used in dynamic mechanical analysis (DMA) testing.

Experimental design factors and their associated levels are given in Table 5.1.

Table 5.1

The experimental design factors and their levels

Factor designation	**Factors**	**Level**				
		1	2	3	4	5
A	VGCNF type	Pristine	Oxidized	-	-	-
B	Use of dispersing agent	Yes	No	-	-	-
C	Mixing method	US[a]	HS[b]	HS/US	-	-
D	VGCNF weight fraction (phr[c])	0.00	0.25	0.50	0.75	1.00
E	Temperature (^{o}C)	30^{o}C	60^{o}C	90^{o}C	120^{o}C	-

[a] Ultrasonication
[b] High-shear mixing
[c] Parts per hundred parts of resin

A total of $2 \times 2 \times 3 \times 5 \times 4 = 240$ "treatment combinations" (different combinations of the factor levels in Table 5.1) were randomized to eliminate bias in preparing the specimens. Each treatment combination resulted in three specimens prepared from the same material batch [32, 31]. Each specimen was tested using a dynamic mechanical analyzer (single cantilever/flexure mode) to measure average storage modulus, loss modulus, and tan delta for each treatment combination. Storage and loss modulus are dynamic mechanical properties and they are indicative of the polymer nanocomposite's stiffness and energy dissipation capability, respectively [2]. The detailed description of the experimental design along with the corresponding statistical analysis were thoroughly explained in [31].

5.2 ReShare Model

Herein, the overall experiment model consists of two concurrent workflows: principal component analysis (PCA)[22] followed by fuzzy C-means (FCM) clustering[6], and a self-organizing map (SOM) study [23]. Both workflows use the same dataset, and they are applied on the same object model. Therefore, we will combine them into one workflow with two separate process sequences.

5.2.1 Object Model

The objects in this experiment belong to two categories:

1. Main Object: the model, used for this experiment, consists of 8 dimensions and both PCA and SOM take a set of records of that model. Therefore, the main object in our model (VGCNF) is a collection of compound objects (figure 5.1). VGCNF has eight items and it is bound to a data source through the data binding model described below. Detailed description of the VGCNF object is presented in Table 5.1.
2. Auxiliary Objects: few secondary objects were modeled to assist the execution of the model. First, VGCNF_Read_Parameters is a collection used as an input for the GetVGCNF operation (figure 5.1). It contains one ValueType item that stores the dataset physical location. PCA_Dim_List (figure 5.2) is another collection which is used to store the output of the PCA operation. PCA is an analytical study that converts N-dimensional data into M dimensions (where $M < N$). However, for easier visualization purposes, PCA was used to represent the data in 2 dimensional space. PCA_Dim_List has two items, each of them stores the value of one dimension. In Figure 5.3, we have three more auxiliary objects. PCA_CAVS_Parameters encapsulates the VGCNF collection and provides it as an input to the PCA_CAVS operation. Similarily, SOM_CAVS_Parameters encapsulates the collection and provides it as an input to the SOM_CAVS operation.Finally, FCM_CAVS_Parameters provides the PCA_Dim_List, resulted from the PCA, and the number of clusters to the FCM_CAVS operation.

5.2.2 Data Source and Data Binding Model

The dataset, used in this experiment, is stored as a text file. We defined this data source as an instance of the "FileDataSource" and associated it with a data binding and

Table 5.2

VGCNF object

Item	Object attributes		Valuetype attributes
	SubType	accessibility	SubType
VGCNF_Type	ValueType	public	integer
Use_of_Dispersing_Agent	ValueType	public	integer
Mixing_Method	ValueType	public	integer
VGCNF_Weight_Fraction	ValueType	public	double
Temperature	ValueType	public	double
Storage_Modulus	ValueType	public	double
Loss_Modulus	ValueType	public	double
Tan_Delta	ValueType	public	double

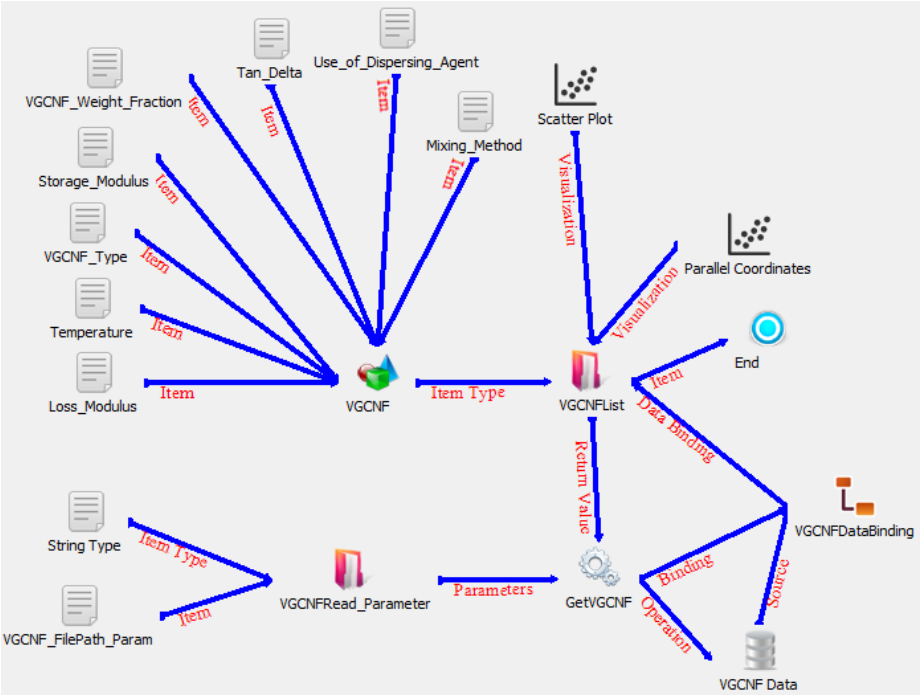


Figure 5.1

VGCNF object and its data source and data binding model.

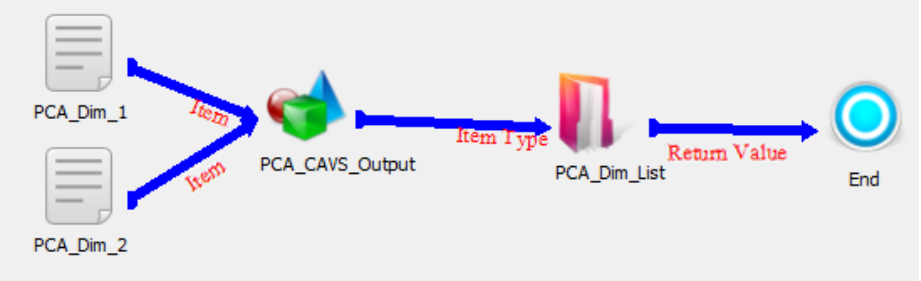


Figure 5.2

PCA parameters object model.

a read operation's annotations. The data binding ontology describes how the GetVGCNF operation will be executed on the data source in order to extract a collection of VGCNF's compound object (figure 5.1).

5.2.3 Workflow and Operation Model

We mainly have one workflow that starts with two separate process sequences: PCA followed by FCM and SOM. Also, we have a set of operations for reading the dataset and calling the experiment's functions. PCA, FCM and SOM were originally written in Matlab. Without any change to the original source code, we used a C# code (PCA_CAVS, FCM_CAVS and SOM_CAVS respectively) that calls the Matlab engine, passes the corresponding parameters, executes the functions and returns their outcome to our model. The final results are shown in Figs. 5.4 and 5.5. Note, the images, in Figs. 5.4 and 5.5, were generated by the Matlab R2012a engine, then exported and displayed in our test bed on-the-fly. These results matched with the original experiment results in [2]. In addition, Figures 5.1 through 5.5 are taken from our test bed.

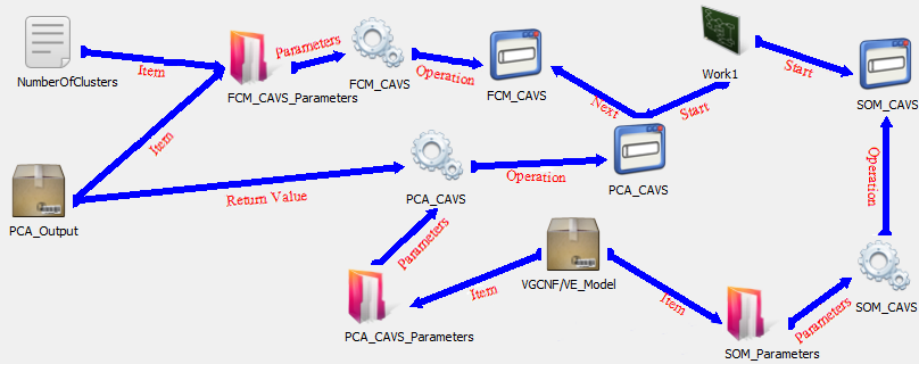


Figure 5.3

Experiment workflow and dependency between its processes.

5.2.4 Visualization Model

Two visualization nodes (Scatter Plot and Parallel Coordinates) are linked to the VGCNFList collection. Both visualizations extract the VGCNF's dimensions and dynamically loads it data through the data binding model.

5.3 Model Extension

It was suggested that the experimental design of the previous material sciences case study can be expanded by including more factors (inputs) and responses in the framework to study the comprehensive and unified mechanical characterization of VGCNF after introducing these variables. This includes curing environment, high shear mixing time (min), sonication time (min) as new inputs and true ultimate strength (MPa), true yield strength (MPa), engineering elastic modulus (GPa), engineering ultimate strength (MPa), flexural modulus (GPa) and flexural strength (MPa) as new responses [42, 24]. This expansion will only require adding nine value type nodes to the VGCNF object. Then, the same exact

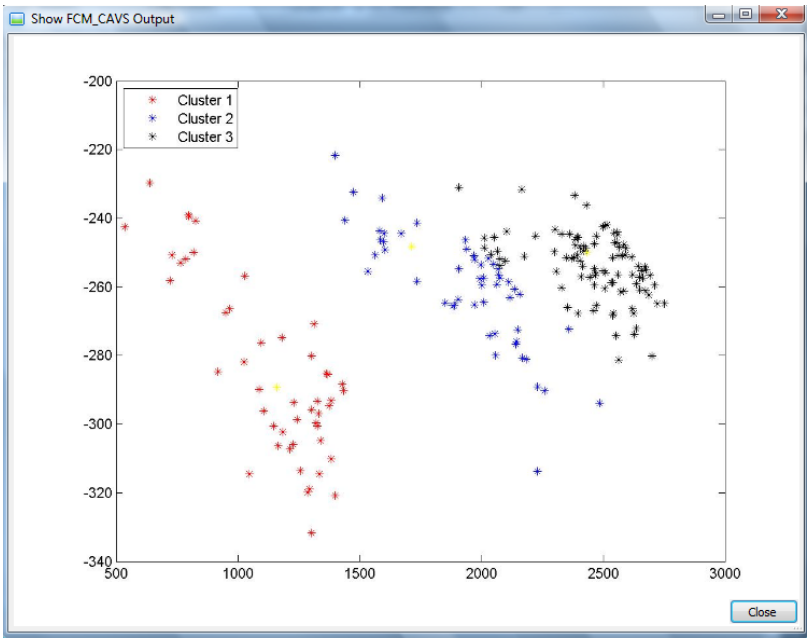


Figure 5.4

Visualized FCM results.

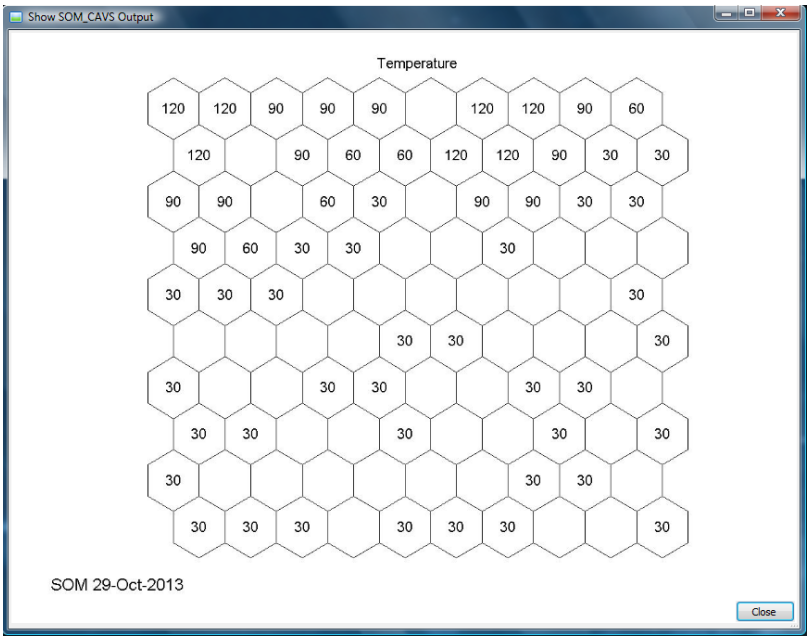


Figure 5.5

Visualized SOM results.

data binding model can be used to bind the object model with the data file that contains the newly added dimensions in the framework. The extended model is shown in figure 5.6.

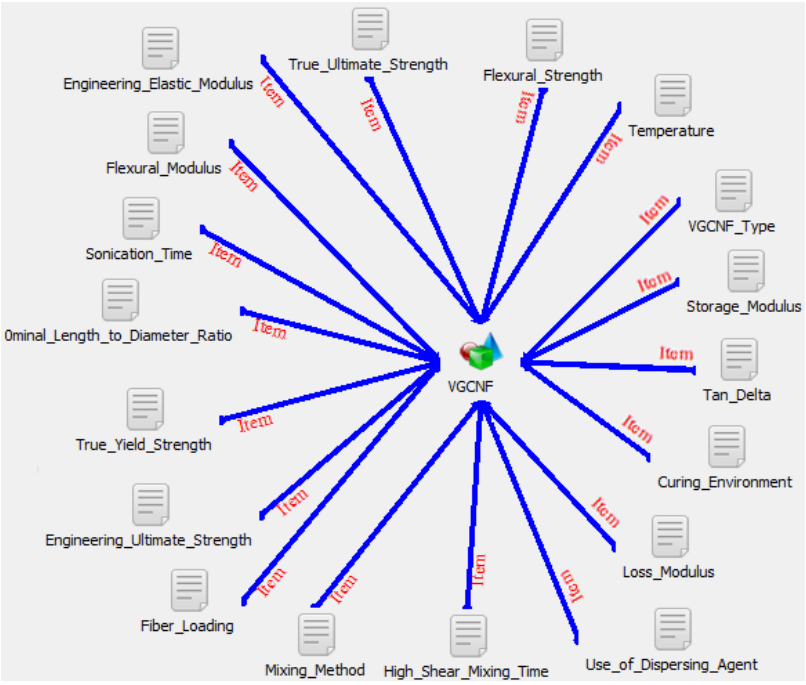


Figure 5.6

Extended VGCNF object including dimensions from new data sets.

5.4 Evaluation

The most important issue which needs to be addressed in the evaluation is the robustness of ReShare. How will the system react to any future adjustment in the objects' model? Any change in the object model (adding, editing and deleting dimensions) will not affect the system's functional behavior. In the previous related work [5, 12, 37, 38], any change in the object model would require building the ontology from scratch. For

instance, VGCNF was extended very easily by adding new dimensions and linking it to the proper data set. This extension did not require any modifications to any other part of the model (workflow, process, operations and visualizations). Also, in the previous studies, the authors generated a descriptive ontology. Therefore, it would be a difficult task to combine models from different research experiments. Using the proposed model in this paper, any change in model(s) that describe the experiment(s) will only require few modifications, and no change is required in the core ontology. Likewise, combining models from different research experiments has become easy to perform since we only need to combine dimensions and adjust the data binding model. Particularly, if the newly added VGCNF inputs and responses mentioned earlier in the expanded case study were utilized in a separate experimental model, then this model can be easily combined with the previous model (before adding the new dimensions) by linking the nodes from both objects to a single object model and then link that object with the proper data source. Also, what if a new technology is introduced? how will the system adapt to the new technologies? The related work in the area of mapping between data sources and ontologies [14, 35, 36] have proposed specific mechanisms for certain data sources such as a database. If a new data source is introduced in the future, these mechanisms will be no longer applicable. Therefore, researchers need to develop new ways to deal with these new types of potential data sources. Moreover, none of these studies dealt with heterogeneous data sources. Thus, if no robust framework is defined, it would be difficult to combine the research studies on the existing technologies with those of the new technologies. ReShare provides a solution for this problem through the operation model. For example, if a new type of data source is

introduced, only new annotations will be plugged into the data source ontology tree. Then, suitable operations should be defined to deal with the new data source, and objects can be bound to the new data source on-the-fly without changing the core of ReShare ontology structure.

CHAPTER 6

ONTOLOGY MATCHING

There is an enormous amount of descriptive ontologies that are available for public use such as Cyc [25] and Scone [13]. In order to make use of these descriptive ontologies, an effective conversion algorithm needs to be proposed. The input of this algorithm should be a single descriptive ontology. Therefore, an ontology matching technique is required to match several ontologies that describe the same concept. For example, there are many existing ontologies for the concept "Person" and these ontologies need to be matched, aligned and merged into a single descriptive ontology in order to convert it into ReShare's operational ontology model. Ontology matching is a process, applied on two ontologies, that tries to find, for each term within one ontology (the source), the best matched term in the second ontology (the destination). This problem can be approached using several mechanisms based on: string normalization, string similarity, data-type comparison, linguistic methods, inheritance analysis, data analysis, graph-mapping, statistical analysis, and taxonomy analysis [26]. A large number of ontology matchers have been put forth. However, different ontology matchers behave differently when applied to various knowledge domains. Also, because of the generic nature of ReShare, the proposed ontology matching should be robust enough to work effectively under all knowledge domains. In

this chapter, a fusion model based on fuzzy measure and integral is presented. This model aggregates results from several ontology matchers and proved to give the desired matching under different knowledge domain. Particularly, two methods for learning the fuzzy measure (supervised and unsupervised) are discussed in this chapter.

6.1 Introduction

The fuzzy integral (FI), introduced by Sugeno [40], is a powerful tool for data fusion and aggregation. The FI is defined with respect to a fuzzy measure (FM). The FM encodes the worth of different subsets of sources. Successful uses of the FI include, to name a few, multi-criteria decision making [15], image processing [41], or even in robotics [33].

It is well-known that different FMs lead the FI, specially the Choquet FI, to behave like various operators (max, min, mean, etc.) [41]. In some applications, we can define the FM manually. However, in other settings, it might only be possible to define the densities (the measures on just the singletons) and a measure building technique is used, e.g., a S-Decomposable measure such as the Sugeno λ-fuzzy measure. It is also very difficult, if possible at all, to specify a FM for relatively small N (number of inputs), as the lattice already has $2^N - 2$ free parameters! (e.g., for $N = 10$ inputs, $2^{10} - 2 = 1022$ free parameters). Also, input sources can be of different nature. For example, we may fuse values from sensors with algorithms outcomes. In such problems, it is hard to determine the importance of each input data source. Thus, many data-driven learning methods are used to learn the measure. One method is to use a quadratic program (QP) to learn the measure based on a given data set [15]. Although the QP is an effective approach to build

the measure using a data set, its complexity is relatively high and it does not scale well [27]. Other optimization techniques have been used to reduce the complexity and improve performance, e.g., genetic algorithms (GA) [3]. In this chapter, two different FM are used for the fusion. The first is learned through a GA while the other is a measure of agreement which based on ideas from crowd sourcing.

6.2 Related Work in Ontologies

A number of works have been put forth in the literature regarding combining several ontology matching techniques. In [46], Wang et al. used Dempster Shafer theory (DST) to combine results obtained from different ontology matchers. Each result is treated as a mass function and combined using the Dempster's Combination Rule. It is not clear that DST is the most suitable fit because in DST one typically only has access to a limited amount of evidence in different focal sets. However, the authors appear to have access to all information in the form of ontology term matching matrices. Other papers, such as [26, 45], have used a GA to learn the weight of each matching algorithm for an operator like a weighted sum. Herein, an improved set of constraint preserving GA (supervised) operators are put forth and more powerful non-linear aggregation operator, the FI, is investigated. Moreover, although these works improve the reliability of matching by aggregating multiple methods; these works used mechanisms (fitness functions) that depend on a reference ontology. Therefore, none of these methods are applicable when the reference ontology is not available. Without the reference ontology, all existing aggregation methods fail to capture the importance of each individual matcher and would likely result in a less desirable

performance. Thus, a unique way to measure agreement (unsupervised) can address this gap. The claim is, if we cannot determine the worth of each matcher with respect to a certain domain, then we look towards the agreement to help improve the robustness of an approach. The question is how to do this for ontology matching?

6.3 Fuzzy Measure and Integral Foundations

For a non-empty finite set of N input sources, $X = \{x_1, x_2, ..., x_N\}$, the FM is a (set-valued) function $g : 2^X \rightarrow [0, 1]$. The FM has three properties [16]:

1. $g(\emptyset) = 0$, $g(X) = 1$ (boundary conditions),

2. For $A, B \subseteq X$, such that $A \subseteq B$, $g(A) \leq g(B)$ (the monotonicity constraint).

3. If $A_n \in X$ and $A_1 \subseteq A_2 \subseteq A_3 \subseteq ...$, then $\lim_{n\rightarrow\infty} g(A_n) = g(\lim_{n\rightarrow\infty} A_n)$. However, this property is not applicable in the case of finite sets..

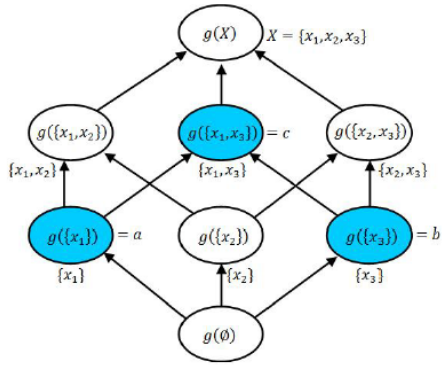
The FM can be discussed in terms of its corresponding lattice which is induced by the monotonicity constraints. The value in each node in the lattice is the measure for a certain subset of sources, i.e., the worth of a particular group. Figure 6.1 is an illustration of the FM for $N = 3$.

The Sugeno λ-measure is one of the well-known and widely used FMs. In the λ-measure, several sources can be combined using the rule

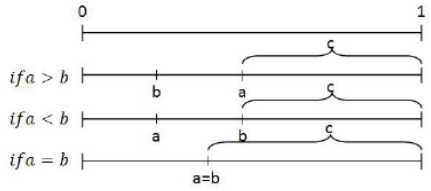
$$g(A \cup B) = g(A) + g(B) + \lambda g(A)g(B), \tag{6.1}$$

where λ is calculated using the formula

$$\lambda + 1 = \prod_{i=1}^{N}(1 + \lambda g^i), \qquad \lambda > -1. \tag{6.2}$$



(a) Lattice representation of the FM for $N = 3$.



(b) Interval view of a monotonicity constraint.

Figure 6.1

FM lattice for $N = 3$ and possible values for $g(\{x_1, x_3\})$.

Data is aggregated using the FI based on the FM. The FI can take many forms, two of which are of focus in our study. For a finite set $X = \{x_1, x_2, ..., x_N\}$, the discrete Sugeno and Choquet FIs are

$$\int_{Sugeno} h \circ g = \bigvee_{i=1}^{N} \left(h(x_{\pi_{(i)}}) \wedge G(x_{\pi_{(i)}}) \right), \tag{6.3}$$

$$\int_{Choquet} h \circ g = \sum_{i=1}^{N} h(x_{\pi_{(i)}}) \left[G(x_{\pi_{(i)}}) - G(x_{\pi_{(i-1)}}) \right], \tag{6.4}$$

where:

- h is the partial support function, $h : X \to [0, 1]$.

- $h(x_{\pi_{(i)}})$ is the evidence provided by source $\pi_{(i)}$.

- π is a re-permutation function such that $h(x_{\pi_{(1)}}) \geq h(x_{\pi_{(2)}}) \geq ... \geq h(x_{\pi_{(N)}})$.

- The value $G(x_{\pi_{(i)}}) = g(\{x_{\pi_{(1)}}, x_{\pi_{(2)}}, ..., x_{\pi_{(i)}}\})$ is the measure of a set of information sources.

- $G(x_{\pi_{(0)}}) = 0$.

Note, the $\int_{Sugeno} h \circ g$ is bounded between $\left[\bigwedge_{i=1}^{N} h(x_i), \bigvee_{i=1}^{N} h(x_i)\right]$ and their method is often explained as "the best pessimistic agreement".

6.4 Supervised Learning Using Genentic Algorithms

Many previous attempts exist to address constraint satisfaction in a GA. Typically, a penalty function is used to reduce the fitness of solutions that violate constraints [28]. However, even if infeasible solutions are close to an ideal minimum, it is not clear why or how to really balance the cost function to eventually help it converge to a quality valid solution. In the FM, we have an exponentially increasing number of constraints in N. It is a highly constrained problem. It is not likely that we will end up obtaining a valid FM for such a heavy constrained problem (or that one such infeasible solution could simply be made a quality feasible solution at the end). Therefore, penalty function strategy becomes very risky to use and will restrain the search space. Many researchers avoid dealing with the FM constraints by designing a GA that learn only the densities. Then, the rest of the lattice is populated using a measure deriving technique such as the Sugeno λ-fuzzy measure. However, a better solution would be to design a more intelligent set of GA operators for the constraints in a FM. In this case, we can efficiently search just the valid FM space and operate on valid FMs. Since all chromosomes in a population are always valid (started and remained valid by using our crossover and mutation operators), we did not have to modify the selection process. Only crossover and mutation need be defined to preserve the monotonicity constraints.

6.4.1 Constraint Preserving Crossover for the FM

In this subsection, a new GA crossover operation is described to preserves the monotonicity property of the FM. First, g($\emptyset$) and g(X) need not be considered as they are 0 and 1 by definition. It is the remaining $2^N - 2$ lattice elements that we are concerned with. In order to ensure a valid FM after crossover, we decompose the FM into a set of interval relations that present the allowable bounds for crossover. For example, consider a set $\{x_1, x_3\}$ of sources, $\{x_1\}$ and $\{x_3\}$, such that $g(\{x_1\}) = a$, $g(\{x_3\}) = b$ and $g(\{x_1, x_3\}) = c$. We know from the FM that $max(a, b) \leq c$. Figure 6.1 illustrates the valid interval ranges that a, b and c can possess in order to remain a valid (credible) FM.

Definition 1 (Intersected and non-inclusive intervals). Let $d = [d_1, d_2], e = [e_1, e_2]$ be two intervals such that $d \cap e \neq \emptyset$. Let d be the smaller of the two intervals, i.e., $d_2 - d_1 < e_2 - e_1$. If $d_1 \leq e_1$ or $d_2 \geq e_2$ then we call d and e intersected and non-inclusive intervals.

Note, this property is needed herein in order to identify candidate chromosomes to perform crossover on.

Definition 2 (Random density crossover). Let g_1 and g_2 be two FMs. Let $\leftrightarrow$ denote the random selection and swapping of one density from g_1, $b_1 = g_1(x_i)$, and g_2, $b_2 = g_2(x_j)$, where i,j are random numbers in $\{1, 2, .., N\}$.

Note, by itself, $\leftrightarrow$ does not guarantee a valid FM.

Definition 3 (Repair operator for $\leftrightarrow$). Let $\Leftrightarrow$ (explained in Prop 1) denotes an operation to repair a violation of the monotonicity constraint caused by $\leftrightarrow$.

Even though we discuss crossover of densities ($\leftrightarrow$), $\Leftrightarrow$ has the result which impacts multiple layers in the FM. As such, while we only cross two densities, we are in fact changing many values in a FM. It is important to note that our $\Leftrightarrow$ makes use of existing values in the FM to repair any violations in the other FM. This means no "new" information is injected. The values in two FMs are swapped rather (in the classical theme of crossover).

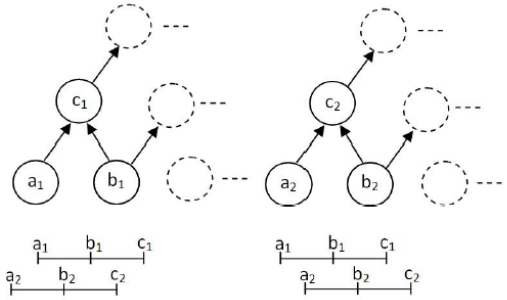


Figure 6.2

Two FMs with possible range conditions for swapping values across measures.

Proposition 1: Let g_1 and g_2 be two FMs with N input sources and let the measure on the densities $\{\{x_1\},\{x_2\},...,\{x_N\}\}$ be $\{g_1^1, g_1^2, ..., g_1^N\}$ and $\{g_2^1, g_2^2, ..., g_2^N\}$ respectively. Furthermore, let each measure be sorted individually such that $g_1^{\pi_1(1)} \leq g_1^{\pi_1(2)} \leq ... \leq g_1^{\pi_1(N)}$ and $g_2^{\pi_2(1)} \leq g_2^{\pi_2(2)} \leq ... \leq g_2^{\pi_2(N)}$ where π_1,π_2 are re-permutation functions. Furthermore, we make the assumption that each corresponding sorted sub-interval is intersected and non-inclusive between g_1 and g_2 (Def.1). Then, all admissible $\leftrightarrow$ (Def.2), followed by $\Leftrightarrow$ (Def.3), operations are guaranteed not to violate the FM monotonicity property.

Proof: This proposition is proved by considering all possible enumerable cases. First, we divide the problem into identical sub-problems, and we prove the proposition for the case of one interval pair. Next, this single case is extended to the case of multiple corresponding interval pairs.

First, g_1 and g_2 are individually sorted. Then, the intervals $[g_1^{\pi_1(1)}, g_1^{\pi_1(N)}]$ and $[g_2^{\pi_2(1)}, g_2^{\pi_2(N)}]$ are divide into sub-intervals, each with two sources i.e., $\{[g_1^{\pi_1(1)}, g_1^{\pi_1(2)}],[g_1^{\pi_1(2)}, g_1^{\pi_1(3)}], ..., [g_1^{\pi_1(N-1)}, g_1^{\pi_1(N)}]\}$ and $\{[g_2^{\pi_1(1)}, g_2^{\pi_1(2)}],[g_2^{\pi_1(2)}, g_2^{\pi_1(3)}], ..., [g_2^{\pi_1(N-1)}, g_2^{\pi_1(N)}]\}$. Next, we look at the case of a single corresponding interval pair between g_1 and g_2. Let $a_1 = g_1^{\pi_1(1)}$, $b_1 = g_1^{\pi_1(2)}$, $a_2 = g_2^{\pi_2(1)}$ and $b_2 = g_2^{\pi_2(2)}$. Since $[a_1, b_1], [a_2, b_2]$ are intersected and non-inclusive intervals (Def.1), then we have two cases (see Figure 6.2):

$$a_2 \leq a_1 \leq b_2 \leq b_1 \tag{6.5}$$

$$a_1 \leq a_2 \leq b_1 \leq b_2 \tag{6.6}$$

For case 6.5, there are four admissible crossover operations:

(1) $a_1 \leftrightarrow a_2$: no need to swap c_1 and c_2 and the new intervals are $[a_2, b_1], [b_1, c_1]$ and $[a_1, b_2], [b_2, c_2]$.

(2) $a_1 \leftrightarrow b_2$: Since $b_2 \leq b_1$, the new intervals are $[b_2, b_1], [b_1, c_1]$ and $[a_2, a_1], [a_1, c_2]$. No need to swap c_1 and c_2.

(3) $b_1 \leftrightarrow a_2$: Swap $c_1 \Leftrightarrow c_2$. The new intervals are $[a_2, a_1], [a_1, c_2]$ and $[b_2, b_1], [b_1, c_1]$.

(4) $b_1 \leftrightarrow b_2$: Swap $c_1 \Leftrightarrow c_2$. The new intervals are $[a_1, b_2], [b_2, c_2]$ and $[a_2, b_1], [b_1, c_1]$.

Case 6.6 is proved the same way i.e., perform $\leftrightarrow$ then check the resulted intervals and

apply $\Leftrightarrow$ when required. Because $\leftrightarrow$ is applied on one density in each FM, the proof of the first sub-interval case is easily generalized to the other cases.

Since Def.1 holds, we randomly select one density in g_1 to be swapped with a randomly swapped density in g_2. Then, $\Leftrightarrow$ is iteratively repeated up the lattice to fix all violations resulting from the $\leftrightarrow$ operator. At each layer in the lattice, which means for all measure values on sets of equal cardinality, intervals are compared and $\Leftrightarrow$ is applied when required.

6.4.2 Constraints Preserving Mutation

In mutation, a random value in the lattice will be changed. In order not to violate the monotonicity constraint, the new value of a randomly selected node should be greater than or equal to the maximum measure of the nodes coming into it and less than or equal to the minimum measure of those it goes into at the next layer. This is the only required modification to the traditional mutation by restricting the new value between a minimum and maximum thresholds. The new bounds can be defined as

$$g_m(C) \in \left[\bigvee_{A \subseteq C} g(A), \bigwedge_{C \subseteq B} g(B) \right] \tag{6.7}$$

6.4.3 Genetic Algorithm Implementation

The genetic algorithm used in the section go through the following steps:

- Encoding: Each chromosome is a vector of [0,1]-valued numbers that map lexographically to a FM e.g., $(g_1, g_2, ..., g_N, g_{1,2}, ..., g_{1,...,N-1})$. The length of a chromosome is $2^N - 2$ (where there are N similarity matching algorithms).
- Fitness Function: Semantic precision and recall are widely used to evaluate the performance of ontology matching algorithms [11].They are an information retrieval metrics [43] that have been used for ontology matching evaluation since 2002 [9].

Fmeasure is a compound metric that can reflect both precision and recall, and it is given by the formula:

$$Fmeasure = \frac{2 * precision * recall}{precision + recall}, \tag{6.8}$$

where precision is given by:

$$precision = \frac{|A \cap R|}{A}, \tag{6.9}$$

and recall is given by:

$$recall = \frac{|A \cap R|}{R}, \tag{6.10}$$

R is the reference ontology and A is the alignment resulted from the matching technique. First, the most fitted chromosome (FM) is used with the FI to calculate A. Then, A is used in computing precision and recall. Lastly, Fmeasure is obtained from the resulted precision and recall. The goal of our GA is to maximize the Fmeasure. We linearly scaled the plot (Fig 6.4) to aid visual display.

- Initialization: A population of size 100 is initialized using random numbers that preserve the fuzzy measure properties.
- Crossover: Crossover is performed in two phases. The first phase checks the intersection property between the parents' chromosomes so they will result in a valid offspring. The second phase performs Def.2 and 3.
- Selection: We adopted traditional roulette wheel selection.
- Stop condition: A maximum number of iterations is used as the stopping condition.

6.4.4 Experimental Results and Evaluation

Several tests were conducted on the I^{3}CON [21] data set. We compared the performance of our tool with existing tools using the precision, recall and Fmeausre (see figures 6.5a, 6.5b, 6.5c). Although our approach gives lower precision than FOAM, FALCON and SMOA individually, it gives a better Fmeasure. This occurs because we combine several results which reduce precision but increase recall considerably. Our tool provides a precision of 1 if and only if the precision of all combined sources is 1. Also, our tool

tends to give better results than GOALS. GOALS uses an OWA operator to combine results from different matchers. We conducted two experiments on the "AnimalsA.owl" and "AnimalsB.owl" using different OWAs. With an $OWA = [0.3, 0.4, 0.3]$ ($OWA = [0.7, 0.2, 0.1]$ respectively) we get an Fmeasure of 0.8571 (0.875 respectively) while our tool returns a value of 0.8936. Inconsiderate (underestimated or overestimated) selection of OWA values dose affect the performance of the system. Since different matchers give different results on different domains, it is difficult to choose the optimal selection of OWA that gives the best results. Our system solves this problem by using the GA to learn the weights. We can learn an OWA if its needed, or any other aggregation operators, but it does not need to be decided up front.

Also, figures 6.3a, 6.3b and 6.3c show the convergence of the GA using 10,20, and 30 iterations with crossover rate of 0.1, 0.2 and 0.3. Each graph is the average fitness value of the population at each iteration. We also report, at each x-location, the full range bounds (max and min). By comparing the results from different configurations, we found that a 10% crossover rate with 10 iterations was enough to achieve the desired results herein. With such configurations, the GA reached convergence with minimal computational effort. Also, we found that the range bounds decreases, which reflects convergence over time. Moreover, we found that all of the plots decrease at some point. This is caused by mutation where we are exploring a bigger space which may not have a high fitness value. To determine the performance of the constraint preserving crossover, we conducted two studies. Study 1 shows the average number of chromosomes that have a valid interval property (see figures 6.4a, 6.4b, 6.4c). This reflects the applicability of our crossover

model. We notice that the average number of suitable mates tend to decrease at each iteration. This happens because chromosomes will converge more at each iteration and will likely violate the interval property. Study 2 is conducted on the average number of attempts to find a suitable mate to crossover with (see figures 6.4d, 6.4e, 6.4f). We found that this number tends to increase on average at each iteration because fewer chromosomes will have a valid interval property. Thus, we have a lesser chance to find a valid candidate pair and the number of attempts increases.

6.5 Unsupervised Learning Using Crowd Sourcing

6.5.1 Measure of Agreement

In [44] we proposed a FM of agreement between N interval-valued inputs. That work was extended in [18] to overcome a natural bias with respect to the length of intervals. Herein, we consider the extension of this FM, g^{AG}, for ontology matchers. The result of an ontology matcher is a matrix of similarity values. Specifically, we put forth a similarity calculation based on the population standard deviation. It computes agreement between multiple ontology matchers in the context of g^{AG}.

A number of works have been put forth in the literature of distance and similarity indices. A recent survey is presented in [7]. However, these measures compute the similarity between two objects only. In our case, we have a set of N input sources. Each one represents a $(l * h)$ matrix resulted from a matching algorithm where l (h respectively) is the number of terms in the source (destination respectively) ontology. Herein, we need to calculate the similarity between all combinations of sources, i.e., between multiple matrices.

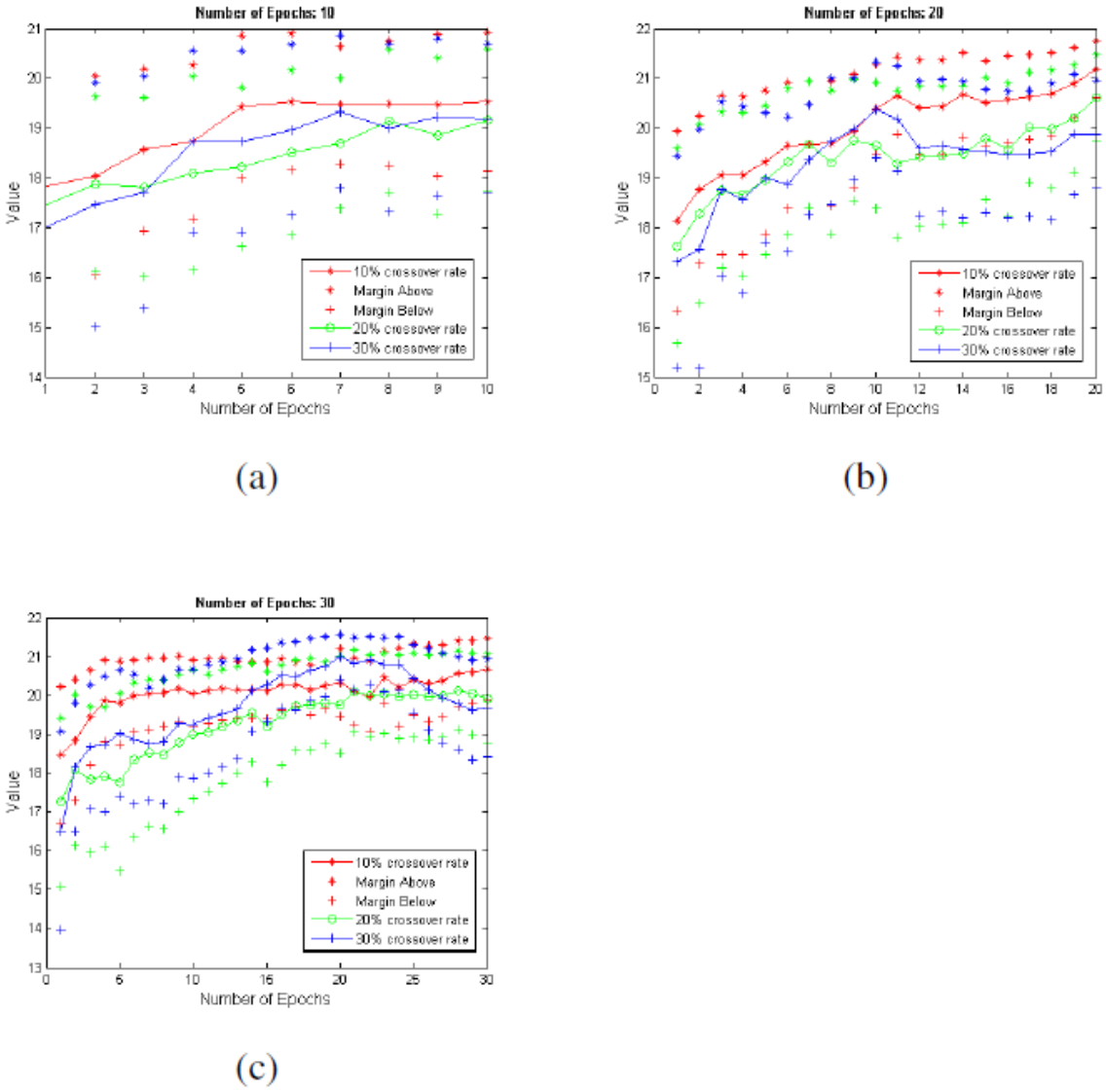


Figure 6.3

Analysis of GA behavior and performance for different parameters.

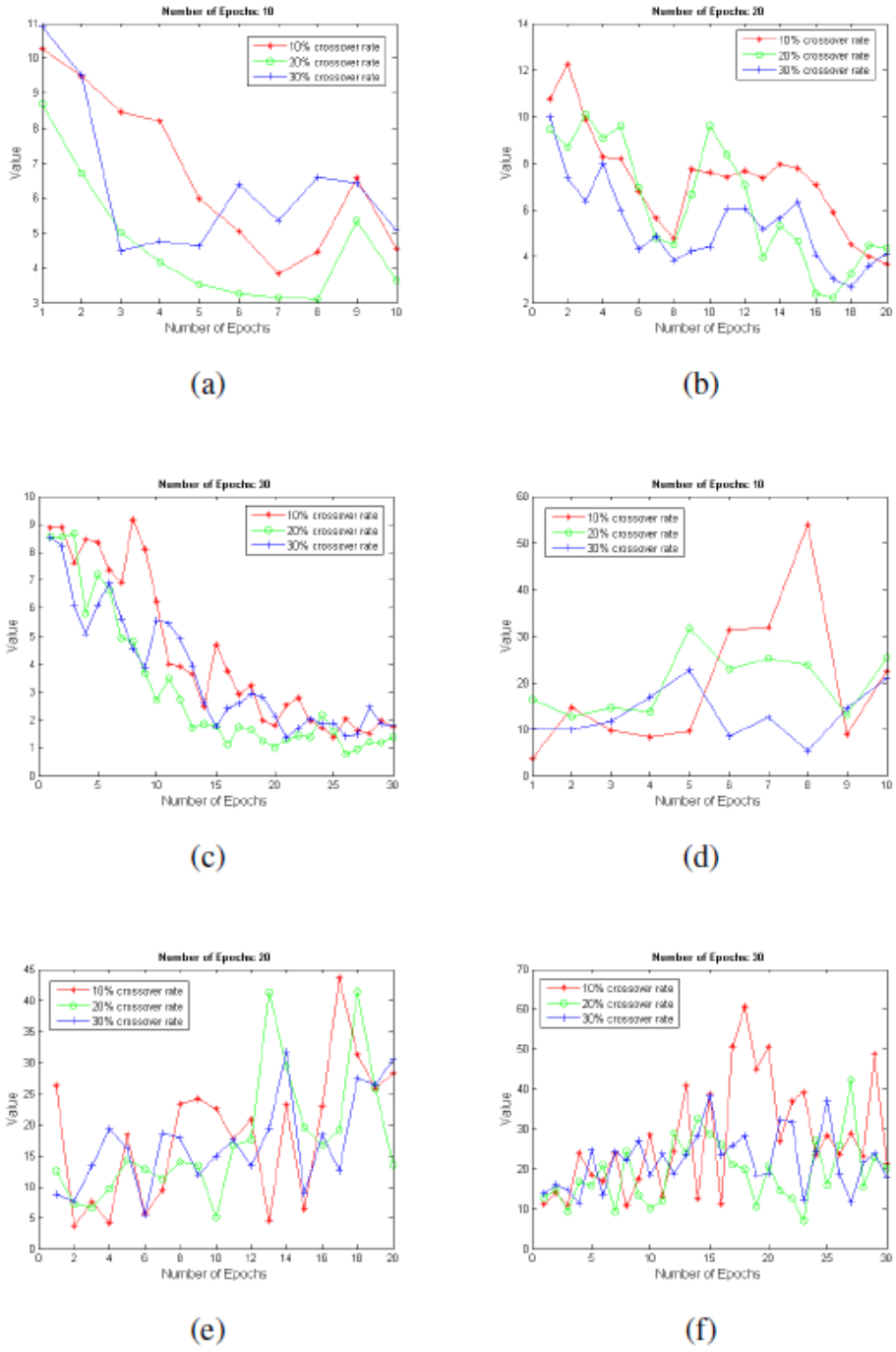
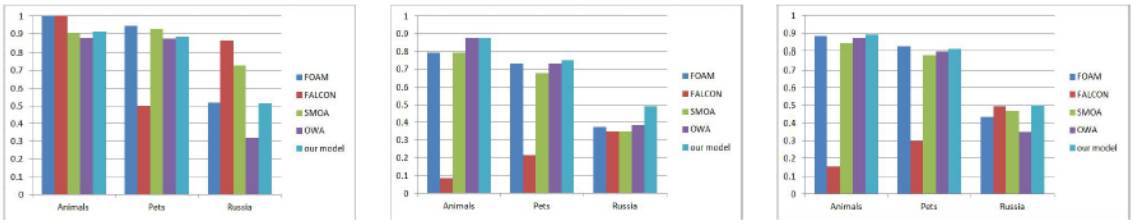


Figure 6.4

Average number of chromosomes with a valid interval property.



(a) (b) (c)

Figure 6.5

Ontology matching evaluation metrics: precision, recall, Fmeasure.

The population standard deviation is a statistical measure that shows the variation in the values of elements in the population, compared to the population mean value. For a population of M elements, $Y = \{y_1, y_2, ..., y_M\}$, the population standard deviation is

$$\sigma = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(y_i - \mu)^2}, \tag{6.11}$$

$$\mu = \frac{\sum_{i=1}^{M} y_i}{M}. \tag{6.12}$$

Herein, σ is used to measure distance between two or more matrices. Specifically, g^{AG} (a FM) is defined as

$$g^{AG}(A_0) = g^{AG}(A_1) = 0,$$

$$g^{AG}(A_i) = \bigvee_{k_1=1}^{i-1} \bigvee_{k_2=k_1+1}^{i} d_\sigma(C_{\pi_{(k_1)}}, C_{\pi_{(k_2)}})\ z_2 +$$

$$\bigvee_{k_1=1}^{i-2} \bigvee_{k_2=k_1+1}^{i-1} \bigvee_{k_3=k_2+1}^{i} d_\sigma(C_{\pi_{(k_1)}}, C_{\pi_{(k_2)}}, C_{\pi_{(k_3)}})\ z_3 +$$

$$... + d_\sigma(C_{\pi_{(1)}}, C_{\pi_{(2)}}, ..., C_{\pi_{(i)}}) z_i, \qquad i = [2 : N], \tag{6.13}$$

where $d_\sigma(C_{\pi_{(1)}}, C_{\pi_{(2)}}, ..., C_{\pi_{(i)}}) = \frac{\sum_{q=1}^{l}\sum_{j=1}^{h}\left(1-\sigma\left(C_{\pi_{(1)}}(q,j), C_{\pi_{(2)}}(q,j)..., C_{\pi_{(i)}}(q,j)\right)\right)}{l*h}$,

such that $A_0 = \emptyset, A_i = \{C_{\pi_{(1)}}, C_{\pi_{(2)}}, ..., C_{\pi_{(i)}}\}$ is the permuted set of matrices such that C_i is the matrix resulted from input source (matching algorithm) x_i, z_i is the weight of each term such that $z_2 \leq z_3 \leq ... \leq z_N$ (z_i can take any value, such as $\frac{i}{N}$, which puts more importance on larger sets of sources), and max is used as a t-conorm ($\bigvee$) operator. Note, since the values are each matrix are in [0,1], σ is also bound in [0,1]. Therefore, $1 - \sigma \in [0, 1]$.

In [18] we proved that g^{AG} is monotonic and non-decreasing. Also, in order to guarantee the boundary conditions ($g(\emptyset) = 0$ and $g(A_N) = 1$), g^{AG} is normalized by

$$\tilde{g}^{AG}(A_i) = \frac{g^{AG}(A_i)}{g^{AG}(A_N)}. \tag{6.14}$$

The following two descriptions help to describe the inner workings of $\tilde{g}^{AG}$:

1. The worth of an individual is defined to be zero. The proposed measure of agreement is based on groups of individuals and their consensus.
2. The agreement of a set of matchers includes the worth of all sub-combinations to better characterize and account for all sub-agreement in the smaller groups.

For example, Let x_1, x_2, and x_3 be ontology matchers and let C_1, C_2, and C_3 be the ontology alignment matrices returned by each of the matchers. Furthermore, let $d_\sigma(C_1, C_2) = 0.8$, $d_\sigma(C_1, C_3) = 0.3$, $d_\sigma(C_2, C_3) = 0.4$, and $d_\sigma(C_1, C_2, C_3) = 0.2$. Then, g^{AG} can be calculated using Equation (6.13):

$g^{AG}(\emptyset) = g^{AG}(x_1) = g^{AG}(x_2) = g^{AG}(x_3) = 0.$

$g^{AG}(\{x_1, x_2\}) = max(g^{AG}(x_1), g^{AG}(x_2)) * \frac{1}{3} + d_\sigma(C_1, C_2) * \frac{2}{3} = 0.53.$

Likewise, $g^{AG}(\{x_1, x_3\}) = 0.2, g^{AG}(\{x_2, x_3\}) = 0.267$, and $g^{AG}(\{x_1, x_2, x_3\}) =$

$max(g^{AG}(x_1), g^{AG}(x_2), g^{AG}(x_3)) * \frac{1}{3} + max(g^{AG}(\{x_1, x_2\}),\ g^{AG}(\{x_1,\ x_3\}),\ g^{AG}(\{x_2,$ $x_3\})) * \frac{2}{3} + d_\sigma(C_1, C_2, C_3) * \frac{3}{3} = 0.356 + 0.2 = 0.556.$

Finally, using the normalization in Equation (6.13), $\tilde{g}^{AG}$ is: $\tilde{g}^{AG}(\emptyset) = \tilde{g}^{AG}(x_1) = \tilde{g}^{AG}(x_2) = \tilde{g}^{AG}(x_3) = 0$, $\tilde{g}^{AG}(\{x_1, x_2\}) = 0.953$, $\tilde{g}^{AG}(\{x_1, x_3\}) = 0.359$, $\tilde{g}^{AG}(\{x_2, x_3\}) = 0.48$ and $\tilde{g}^{AG}(\{x_1, x_2, x_3\}) = 1$.

6.5.2 Experimental Results and Evaluation

In this section, several experiments were performed to show how the proposed technique behaves on different knowledge domains. We prefer the use of the I^{3}CON [21] dataset over OAEI [20] for two reasons. First, I^{3}CON provides a set of ontologies from diverse knowledge domains while OAEI focuses only on the bibliographic references domain. Having a wide range and real life related domains helps to support our claim that different ontology matchers have different performance characteristics in different domains. Also, all tests in OAEI (except for one) are systematically built, and researchers take that into consideration when designing their matchers. Therefore, matchers might not cover all cases found in real life problems.

Several tests were performed, and we compared the performance of our aggregation relative to the individual matchers. In Figure 6.6, we show the matching matrices returned by each individual algorithm and our aggregated result. The first (second respectively) dimension of the matrix corresponds to concepts in the source (destination respectively) ontology. In Figure 6.7, we compared the individual and fused alignments. Cells detected

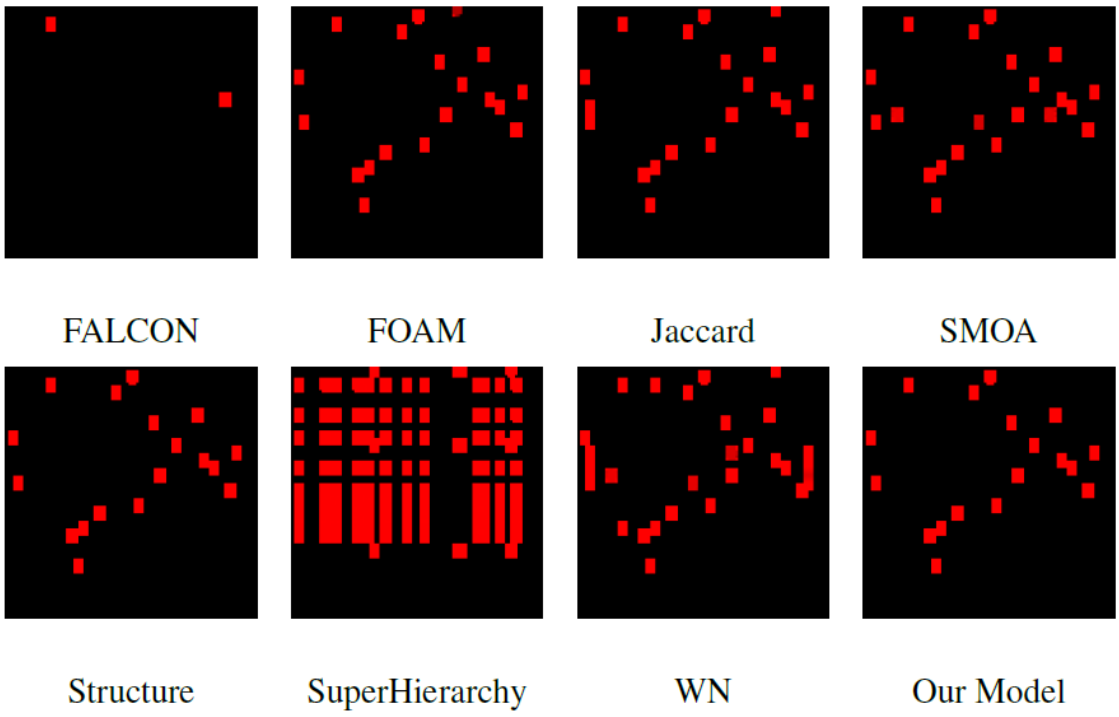


Figure 6.6

Visualization of different matchers for the Animal ontology.

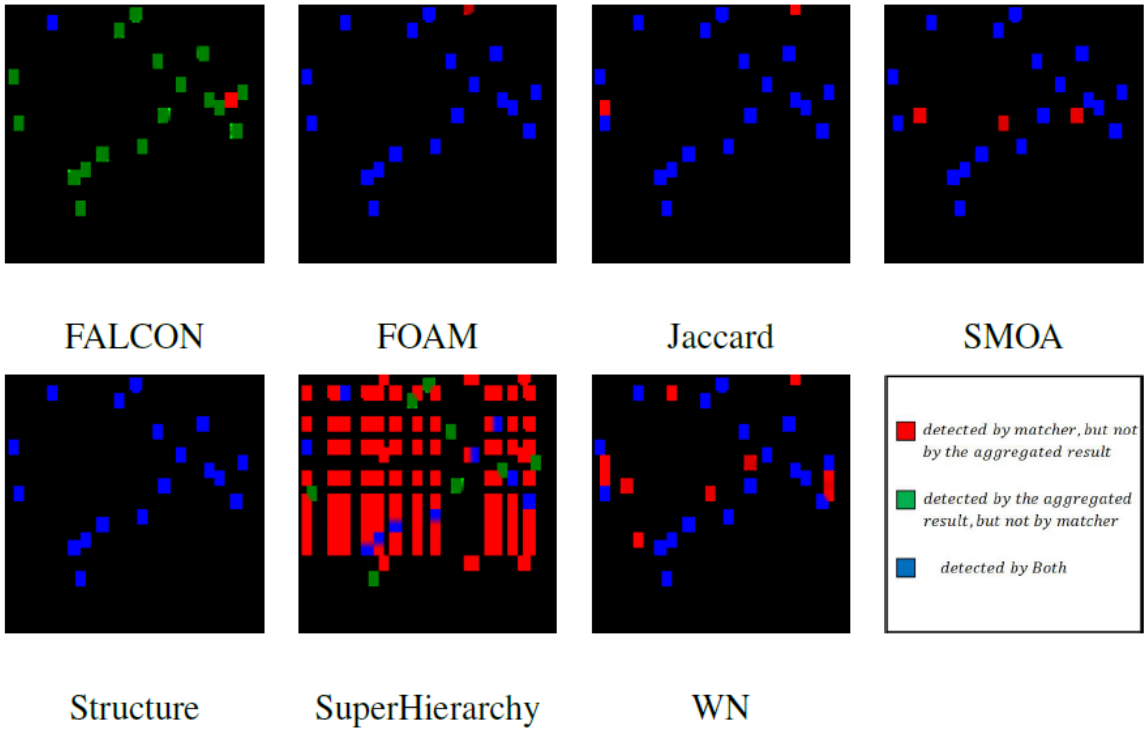
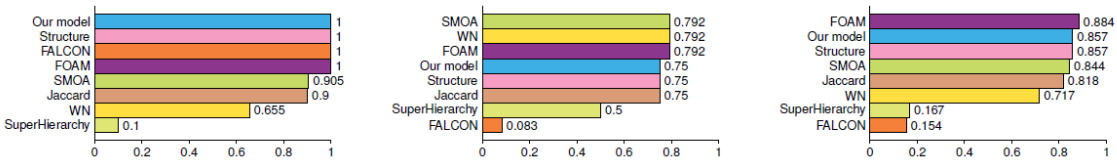


Figure 6.7

Visualization of the differences between the individual matchers.

by both algorithms are reported in blue. We are able to see how the fuzzy agreement measure captured the agreement between the different matchers.



(a) Precision (P) (b) Recall (Re) (c) FMeasure (F)

Figure 6.8

P, Re, and F for the Animal ontology.

In Figures 6.8 through 6.13, we compare the performance of the aggregated result with the individual matchers. The results show the inconsistent behavior of the existing matchers in different domains. For example, FALCON ([19]) has a very good precision in the "Animals" ontology (Figure 6.8), whereas in the "Pets" ontology (Figure 6.9) it has a poor performer (1 and 0.5 respectively). The inconsistent behavior is also manifested in the big difference in Recall between the "Animals" and the "Sport Events" ontologies (0.083 and 0.893 respectively), see Figures 6.8 and 6.10. Another good example is the Structure matcher, which got a Recall of 0.903 in the "Pets" ontology and scored just 0.007 for "Sport Event" (Figures 6.9 and 6.10). Even FOAM ([10]), which scored high in most ontologies (Figures 6.8, 6.9 and 6.11), scored low in the "Vehicles" and "Tourism" ontologies (Figures 6.12 and 6.13). No matcher is expected to universally outright solve this problem. On the other hand, the performance of our tool was relatively stable throughout all domains

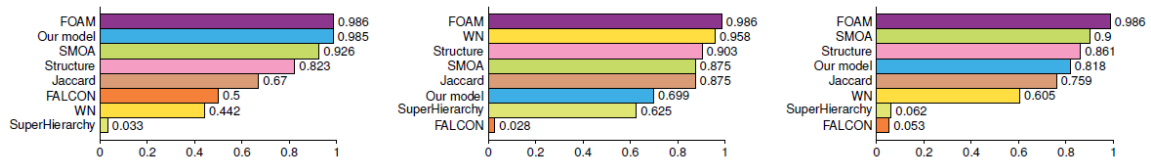


Figure 6.9

P, Re, and F for the Pets ontology.

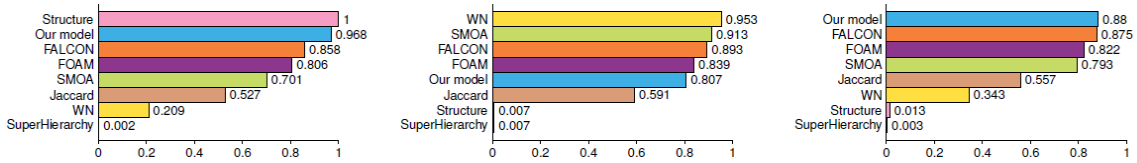


Figure 6.10

P, Re, and F for the SportEvent ontology.

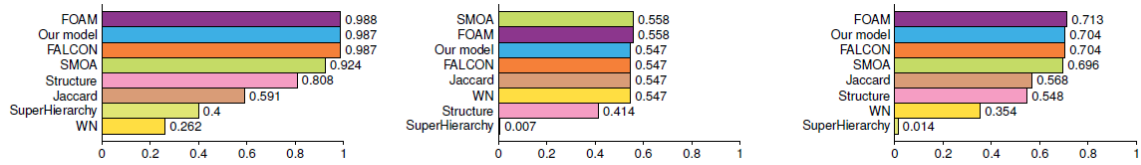


Figure 6.11

P, Re, and F for the Russia ontology.

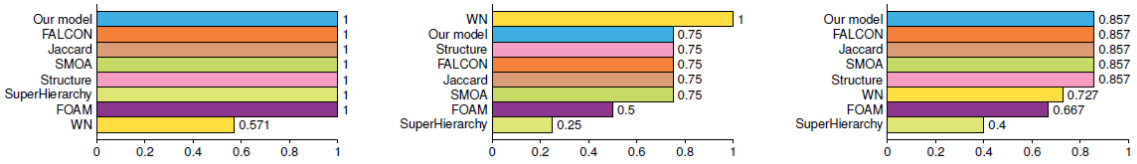


Figure 6.12

P, Re, and F for the Vehicles ontology.

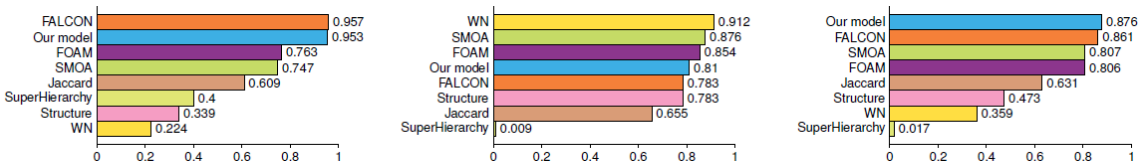


Figure 6.13

P, Re, and F for the Tourism ontology.

(Figures 6.8 through 6.13). We were able to achieve high Precision (top 2) in all domains —as expected— because when many matchers agree on an alignment cell, then it will most likely be valid. And for the same reason, our tool did not score very high at the Recall factor. The crowd sourcing model is cautious enough to wait for many votes from the matchers before deciding whether any cell should be considered in the output alignment or not. Given no prior knowledge on how good the individual matchers are and without the need for any reference ontology, our model did provide remarkably robust results when aggregating several matchers (top 3 with respect to Fmeasure).

CHAPTER 7

DISCUSSION AND CONCLUSIONS

In this thesis, we proposed ReShare: an operational ontology framework that provides scientists with the ability to build, combine and share their research experiments. In this framework, we employed the software OO methodologies to create a highly robust and adaptive ontology. This ontology can be used to model experiments in any research domain. We also developed a test bed, and we have successfully used it to model a research experiment in the field of materials science. This task was performed only by using few high level annotations and the results were also displayed in our test bed. Moreover, ReShare models can be exchanged between researchers by sharing the ontology which will facilitate the process of understanding and extending models that were previously designed. Furthermore, the main advantage of using ontology instead of the traditional XML is that it provides the ability to perform semantic search. This feature will increase the practicality of our work, and researchers can effectively look for models built using ReShare.

Also, an interactive 3D scatter plot and parallel coordinates visualizations were implemented. Both visualizations can be complementary to one another. 3D scatter plot is more useful to understand the data since it uses a Cartesian space. Nevertheless, it can visualize up to a certain number of control variables. On the other hand, parallel coordinates can

visualize any number of dimensions. However, it is less obvious to understand the data patterns between all the variables at the same time. Moreover, different data trend discovery algorithms were presented and a descriptive ontology model for these visualization tools was also proposed.

Finally, different aggregation methods (supervised and unsupervised) for fusing ontology matchers were proposed. In the first method, we proposed a new constraint preserving GA for learning FMs. Specifically, we proposed a new crossover and mutation operation and different ontology matching algorithms were fused using the FI learned by the GA. We showed that our framework can give satisfactory results on different study domains. While in the second method, we presented a novel way to aggregate ontology matchers without relying on the use of a reference ontology. Specifically, we employed the FI with respect to a FM of agreement to fuse different ontology matching algorithms. That is, when nothing is known about the quality of the individuals, we can look to identify agreement among the inputs and use that measure on the same data to guide its aggregation. In addition, we proposed the use of population standard deviation to measure the distance between two or more matrices. We showed that our model can give satisfactory and robust results in different study domains.

CHAPTER 8

FUTURE WORK

This research can be broadened in many different directions. First of all and most importantly, since we have a huge amount of descriptive ontologies, we need to design an algorithm that utilizes these existing ontologies to define operational models.This algorithm will also need to handle different ontologies that describe the same area of research. Therefore, the ontology matching work presented herein is very essential because the resultant ontology will be passed to the algorithm to convert it to ReShare model. Furthermore, we would like to develop an algorithm that converts the operation model, designed by ReShare, to a descriptive one. This will be very useful to generate ontologies in fields where such descriptive ontologies are not available.

In regard to ontology matching, we would like to extend the crowd-sourced ontology matcher to match between more than two ontologies. This multiple matching is required because there are innumerable existing descriptive ontologies in all knowledge domains, and these ontologies need to be all combined to create a complete unified model. For example, multiple descriptive ontologies exist for the concept “Person”, just to mention

some of them [1] [2] [3] [4] [5] [6] [7]. Thus, in order to build a single and complete "Person" model, all the previous ontologies and some others need to be matched and merged into a single ontology. Also, few important questions need to be addressed with regard to the matching technique itself, e.g., how to increase the Recall? Should we add more matchers to increase the number of voters in the crowd? If yes, this will considerably increase the number of free parameters in the FM ($2^N - 2$ for N sources) and thus increase complexity as well. How should we deal with the case of having a large number of matchers? Can we use the k-additive measure? This means that we will be listening to votes of certain number of matchers (k-matchers). The use of k-additive measure might be considered risky for a small k because the measure of agreement might lose its significance. To solve the issue of having large crowd, we would like to explore the use of the shapely index to determine the worth of each of the individual matching algorithm and, therefore, omit some of them and reduce the complexity. However, would this result in a biased crowd? And how would we deal with such a crowd? These questions and some others need to be answered to better understands and improve the performance of heterogeneous crowds.

Finally, the visualization work needs to be extended. More visualization tools can be added to the ontology model such as graph, bar and pie chart. Also, we would like to enhance the trend analysis algorithm by using more accurate clustering algorithm such as fuzzy c-means.

[1] http://daml.umbc.edu/ontologies/ittalks/person
[2] http://orlando.drc.com/daml/ontology/Person/current/
[3] http://ubot.lockheedmartin.com/ubot/2001/08/baby-shoe/shoeproj-ont.daml
[4] http://www.cs.umd.edu/projects/plus/DAML/onts/personal1.0.daml
[5] http://www.daml.ri.cmu.edu/ont/homework/atlas-cmu.daml
[6] http://www.ksl.stanford.edu/projects/DAML/ksl-daml-instances.daml
[7] http://www.davincinetbook.com:8080/daml/rdf/personal-info.daml

REFERENCES

[1] *Visualization Toolkit: An Object-Oriented Approach to 3D Graphics, 4th Edition*, 4th edition, Kitware, Dec. 2006.

[2] O. AbuOmar, S. Nouranian, R. King, J. Bouvard, H. Toghiani, T. Lacy, and C. P. Jr., "Data mining and knowledge discovery in materials science and engineering: A polymer nanocomposites case study," *Advanced Engineering Informatics*, 2013.

[3] D. T. Anderson, J. M. Keller, and T. C. Havens, "Learning fuzzy-valued fuzzy measures for the fuzzy-valued Sugeno fuzzy integral," *Computational Intelligence for Knowledge-Based Systems Design*, Springer, 2010, pp. 502–511.

[4] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, et al., "Gene Ontology: tool for the unification of biology," *Nature genetics*, vol. 25, no. 1, 2000, pp. 25–29.

[5] D. N. Batanov and W. Vongdoiwang, "Using ontologies to create object model for object-oriented software engineering," *Ontologies*, Springer, 2007, pp. 461–487.

[6] J. C. Bezdek, R. Ehrlich, and W. Full, "FCM: The fuzzy c-means clustering algorithm," *Computers & Geosciences*, vol. 10, no. 2, 1984, pp. 191–203.

[7] S.-H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *City*, vol. 1, no. 2, 2007, p. 1.

[8] J. Claessen and J. van Wijk, "Flexible Linked Axes for Multivariate Data Visualization," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 17, no. 12, 2011, pp. 2310–2316.

[9] H.-H. Do, S. Melnik, and E. Rahm, "Comparison of schema matching evaluations," *Proc. Workshop on Web, Web-Services, and Database Systems*, Erfurt (DE), 2002, vol. 2593 of *Lecture notes in computer science*, pp. 221–237.

[10] M. Ehrig and Y. Sure, "FOAM-framework for ontology alignment and mapping-results of the ontology alignment evaluation initiative," *Workshop on integrating ontologies*, 2005, vol. 156, pp. 72–76.

[11] J. Euzenat, "Semantic precision and recall for ontology alignment evaluation," *Proc. 20th International Joint Conference on Artificial Intelligence (IJCAI)*, 2007, pp. 348–353.

[12] J. Evermann and Y. Wand, "Ontology based object-oriented domain modelling: fundamental concepts," *Requirements Engineering*, vol. 10, no. 2, 2005, pp. 146–160.

[13] S. E. Fahlman, "The Scone Knowledge Base (home page),", `http://www.cs.cmu.edu/~sef/scone/`, 2006, [accessed 24-January-2014].

[14] A. Gali, C. X. Chen, K. T. Claypool, and R. Uceda-Sosa, "From ontology to relational databases," *Conceptual Modeling for Advanced Application Domains*, Springer, 2004, pp. 278–289.

[15] M. Grabisch, "The application of fuzzy integrals in multicriteria decision making," *European journal of operational research*, vol. 89, no. 3, 1996, pp. 445–456.

[16] M. Grabisch, M. Sugeno, and T. Murofushi, *Fuzzy measures and integrals: theory and applications*, Springer-Verlag New York, Inc., 2000.

[17] T. R. Gruber et al., "A translation approach to portable ontology specifications," *Knowledge acquisition*, vol. 5, no. 2, 1993, pp. 199–220.

[18] T. C. Havens, D. T. Anderson, C. Wagner, H. Deilamsalehy, and D. Wonnacott, "Fuzzy integrals of crowd-sourced intervals using a measure of generalized accord," *Fuzzy Systems (FUZZ), 2013 IEEE International Conference on*, 2013, pp. 1–8.

[19] W. Hu and Y. Qu, "Falcon-AO: A practical ontology matching system," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 6, no. 3, 2008, pp. 237–239.

[20] O. A. E. Initiative, "Ontology Alignment Evaluation Dataset,", `http://oaei.ontologymatching.org/tests/`, 2006, [accessed 11-October-2013].

[21] I. Interpretation and I. Conference, "Ontology Alignment Evaluation Dataset,", `http://www.atl.lmco.com/projects/ontology/i3con.html`, 2004, [accessed 11-October-2013].

[22] I. T. Jolliffe, *Principal Component Analysis*, second edition, Springer, Oct. 2002.

[23] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, 1990, pp. 1464–1480.

[24] J. Lee, S. Nouranian, G. W. Torres, T. E. Lacy, H. Toghiani, C. U. Pittman, and J. L. DuBien, "Characterization, prediction, and optimization of flexural properties of vapor-grown carbon nanofiber/vinyl ester nanocomposites by response surface modeling," *Journal of Applied Polymer Science*, vol. 130, no. 3, 2013, pp. 2087–2099.

[25] D. B. Lenat, "CYC: A large-scale investment in knowledge infrastructure," *Communications of the ACM*, vol. 38, no. 11, 1995, pp. 33–38.

[26] J. Martinez-Gil, E. Alba, and J. F. Aldana-Montes, "Optimizing ontology alignments by using genetic algorithms," *Proceedings of the workshop on nature based reasoning for the semantic Web. Karlsruhe, Germany*, 2008.

[27] A. Mendez-Vazquez and P. Gader, "Sparsity promotion models for the Choquet integral," *Foundations of Computational Intelligence, 2007. FOCI 2007. IEEE Symposium on*. IEEE, 2007, pp. 454–459.

[28] Z. Michalewicz, "Genetic algorithms, numerical optimization, and constraints," *Proceedings of the Sixth International Conference on Genetic Algorithms*. Morgan Kaufmann, San Mateo, CA, 1995, vol. 195, pp. 151–158.

[29] D. C. Montgomery, *Design and analysis of experiments*, 7th ed. edition, John Wiley & Sons, Hoboken, NJ, 2009.

[30] S. Nouranian, *Vapor-grown carbon nanofiber/vinyl ester nanocomposites: Designed experimental study of mechanical properties and molecular dynamics simulations*, doctoral dissertation, Mississippi State University, 2011.

[31] S. Nouranian, T. E. Lacy, H. Toghiani, C. U. Pittman, and J. L DuBien, "Response surface predictions of the viscoelastic properties of vapor-grown carbon nanofiber/vinyl ester nanocomposites," *Journal of Applied Polymer Science*, 2013.

[32] S. Nouranian, H. Toghiani, T. E. Lacy, C. U. Pittman, and J. Dubien, "Dynamic mechanical analysis and optimization of vapor-grown carbon nanofiber/vinyl ester nanocomposites using design of experiments," *Journal of Composite Materials*, vol. 45, no. 16, 2011, pp. 1647–1657.

[33] G.-Y. Pan, W.-Y. Wang, C.-P. Tsai, W.-Y. Wang, and C.-R. Tsai, "Fuzzy measure based mobile robot controller for autonomous movement control," *System Science and Engineering (ICSSE), 2011 International Conference on*, 2011, pp. 649–653.

[34] A. Pease, I. Niles, and J. Li, "The suggested upper merged ontology: A large ontology for the semantic web and its applications," *Working notes of the AAAI-2002 workshop on ontologies and the semantic web*, 2002, vol. 28.

[35] B. Peterson, W. A. Andersen, and J. Engel, "Knowledge bus: Generating application-focused databases from large ontologies," *Proceedings of the 5th Workshop KRDB-98, Seattle, WA, USA*. Citeseer, 1998.

[36] J. Sebestyénová, "Domain ontology based object-oriented and relational databases," *Proceedings of the 5th WSEAS International Conference on Applied Informatics and Communications*. World Scientific and Engineering Academy and Society (WSEAS), 2005, pp. 324–329.

[37] D. W. V. Siricharoen, "Ontology modeling and object modeling in software engineering," *International Journal of Software Engineering and Its Applications*, 2009.

[38] W. V. Siricharoen, "Ontologies and object models in object oriented software engineering," *IAENG International Journal of Computer Science*, vol. 33, no. 1, 2007, pp. 19–24.

[39] S. Staab and R. Studer, *Handbook on ontologies*, Springer, 2009.

[40] M. Sugeno, *Theory of fuzzy integrals and its applications*, doctoral dissertation, Tokyo Institute of Technology, 1974.

[41] H. Tahani and J. M. Keller, "Information fusion in computer vision using the fuzzy integral," *Systems, Man and Cybernetics, IEEE Trans on*, vol. 20, no. 3, 1990, pp. 733–741.

[42] G. W. Torres, S. Nouranian, T. E. Lacy, H. Toghiani, C. U. Pittman, and J. L. DuBien, "Statistical characterization of the impact strengths of vapor-grown carbon nanofiber/vinyl ester nanocomposites using a central composite design," *Journal of Applied Polymer Science*, vol. 128, no. 2, 2013, pp. 1070–1080.

[43] C. J. K. van Rijsbergen, *Information retrieval*, Butterworths, London (UK), 1975, http://www.dcs.gla.ac.uk/Keith/Preface.html.

[44] C. Wagner and D. Anderson, "Extracting meta-measures from data for fuzzy aggregation of crowd sourced information," *Fuzzy Systems (FUZZ-IEEE), 2012 IEEE International Conference on*, 2012, pp. 1–8.

[45] J. Wang, Z. Ding, and C. Jiang, "GAOM: genetic algorithm based ontology matching," *Services Computing, 2006*. IEEE, 2006, pp. 617–620.

[46] Y. Wang, W. Liu, and D. Bell, "Combining uncertain outputs from multiple ontology matchers," *Scalable Uncertainty Mgmt.*, Springer, 2007, pp. 201–214.