Soft Comput (2014) 18:267–284
DOI 10.1007/s00500-013-1056-0

METHODOLOGIES AND APPLICATION

IT2FS-based ontology with soft-computing mechanism for malware behavior analysis

Hsien-De Huang · Chang-Shing Lee · Mei-Hui Wang · Hung-Yu Kao

Published online: 20 June 2013

Abstract Antimalware application is one of the most important research issues in the area of cyber security threat. Nowadays, because hackers continuously develop novel techniques to intrude into computer systems for various reasons, many security researchers should analyze and track new malicious program to protect sensitive and valuable information in the organization. In this paper, we propose a novel soft-computing mechanism based on the ontology model for malware behavioral analysis: *Malware Analysis Network in Taiwan* (*MAN in Taiwan, MiT*). The core techniques of *MiT* contain two parts listed as follows: (1) collect the logs of network connection, registry, and memory from the operation system on the physical-virtual hybrid analysis environment to get and extract more unknown malicious behavior information. The important information is then extracted to construct the ontology model by using the Web Ontology Language and Fuzzy Markup Language. Additionally, *MiT* is also able to automatically provide and share samples and reports via the cloud storage mechanism; (2) apply the techniques of *Interval Type-2 Fuzzy Set* to construct the malware analysis domain knowledge, namely the *Interval Type-2 Fuzzy Malware Ontology* (*IT2FMO*), for malware behavior analysis. Simulation results show that the proposed approach can effectively execute the malware behavior analysis, and the constructed system has also released under GNU General Public License version 3. In the future, the system is expected to largely collect and analyze malware samples for providing industries or universities to do related applications via the established *IT2FMO*.

Keywords Malware behavioral analysis · Type-2 fuzzy set · Ontology · Fuzzy markup language · Soft computing

1 Introduction

In the past few years, how to reduce the damage caused by hackers or malware is an important issue for governments, universities, commercial organizations, and so on (Huang et al. 2011, 2012a, b). Many security researchers have proposed some new defenses to protect user personal, valuable, and confidential data. Unfortunately, security researchers always fall behind the hackers to find the vulnerabilities of the computer systems, which causes the computer systems to be damaged and confidential data to be stolen. Hence, the battle between hackers and security researchers never ends (Dai et al. 2011). Security researchers or industries have been using two popular approaches to malware analysis for a few years. One is based on the heuristic detection technology and another is based on the signature detection technology. However, security researchers require an automatic and effective analyzing tool or model for a rapid defense against unknown malicious attacks, so the behavior-based malware

Communicated by G. Acampora.

H.-D. Huang · H.-Y. Kao
Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan City, Taiwan
e-mail: TonTon@TWMAN.ORG

H.-Y. Kao
e-mail: hykao@mail.ncku.edu.tw

C.-S. Lee (✉) · M.-H. Wang
Department of Computer Science and Information Engineering, National University of Tainan, Tainan City, Taiwan
e-mail: leecs@mail.nutn.edu.tw

M.-H. Wang
e-mail: mh.alice.wang@gmail.com

detection approach becomes more and more popular due to its great potential for identifying previous unknown malicious software. This is because the accuracy of this approach relies on the ability to correctly recognize the patterns and models of the malware, especially in identifying previous unknown instances of malicious software (Dai et al. 2012).

Type-1 Fuzzy Set (T1FS) and Type-1 Fuzzy Logic System (T1FLS) have applied successfully in many areas including modeling, control, and data mining (Lee et al. 2005; Acampora and Loia 2005). Type-2 Fuzzy Set (T2FS) is characterized by Membership Functions (MFs), i.e., the membership value of a T2FS is a fuzzy set in [0, 1], not a crisp number. T2FS can express more fuzzy semantics of humans' thoughts, and recently it has attracted the researchers' attentions (Hagras 2004, 2007). It has been widely developed and successfully used in many practical real-world applications and many areas, including signal processing, human silhouette extraction, diet application, and pattern recognition design (Huang et al. 2012; Hagras and Wagner 2012; Wu 2012; Lee et al. 2010; Acampora and Loia 2007; Sahab and Hagras 2011; Yao et al. 2012). Interval Type-2 Fuzzy Set (IT2FS) is a special cases of T2FS (Castillo et al. 2011), which is currently the most widely used because of the reduction of computational cost (Mendel et al. 2006, 2007).

Ontology is a metadata schema that contains the vocabulary of concepts and their relationship. Each concept is with an explicitly definition and machine readable semantics (Carlsson et al. 2012). Also, it is a knowledge representation and structural frameworks for modeling information by means of an explicit specification or a sharing conceptualization in the field of artificial intelligence, which aims to formally express knowledge in a model and contain concepts with relationships between elements (Sanchez et al. 2006). Ontology has become a useful tool in understanding and structuring concepts of the information systems with different fields when the systems become much larger and more complex. It also has been used for various practical purposes (Valiente et al. 2012) and there are many developed systems to represent knowledge and communicate with intelligent agents based on ontological approaches, such as software development, information service management process, adaptive e-Learning, news summarization, CMMI assessment, and personal diabetic diet recommendations (Lee et al. 2005; Valiente et al. 2012; Lee and Wang 2009; Wang et al. 2009; Lau et al. 2009).

However, it has been widely pointed out that the traditional ontology is not suitable to deal with uncertain, vague, and imprecise knowledge to characterize the real-world scenarios (Bobillo and Straccia 2010). The fuzzy ontology is emerging as a useful methodology for knowledge representation in several semantic-oriented applications, and it can reflect the real-world uncertainty between the relationship and the conceptual information (De Maio et al. 2012). As a consequence, this paper tries to integrate then above-mentioned different kinds of the soft computing approaches to solve the uncertain problem with the cyber security. Typically, joint exploitation of fuzzy ontologies to be one supported framework for designing the fuzzy inference systems is one of the key research topics in the soft computing research areas (Ho et al. 2009; Orriols-Puig et al. 2011). There are many researchers explored the use of fuzzy ontologies, for example, Lee et al. (2005) proposed a fuzzy ontology for designing an intelligent decision making system for summarization system. Quan et al. (2006) presented the automatic fuzzy ontology generation for semantic help desk support and the automatic fuzzy ontology generation for semantic web.

The remainder of this paper is briefly described as follows: Section 1 introduces the purpose of this paper. Section 2 presents the background knowledge about Interval Type-2 Fuzzy Logic System (IT2FLS) and malware behavioral analysis. Section 3 describes Interval Type-2 Fuzzy Ontology for malware behavioral analysis. Then, Sect. 4 describes the FML-based malware similarity computing for *Malware Analysis Network in Taiwan* (*MiT*). Section 5 illustrates the framework of the proposed system and simulation results. Finally, the conclusion is made in Sect. 6.

2 Related work

2.1 Type-2 fuzzy logic system overview

Figure 1a shows a type-2 fuzzy set (T2FS). The interval T2FS (IT2FS) is a special case of T2FS. All secondary grades of an IT2FS are equal to one (Lau et al. 2009; Mendel 2001). Interval Type-2 Fuzzy Logic System (IT2FLS) uses the IT2FS to represent the inputs and/or outputs of the FLS (Mendel 2001) and is helpful to simplify the computation compared to the general T2FLS (Yao et al. 2012). Figure 1b shows the general structure of T2FLS and its operation is briefly described as follows (Acampora and Loia 2005; Hagras 2004, 2007).

The crisp inputs from the input sensors are first fuzzified into the type-2 fuzzy sets. Singleton fuzzification is usually used in IT2FLS applications due to its simplicity and suitability for the embedded processors and real-time applications. The input type-2 fuzzy sets then activate the inference engine to produce output type-2 fuzzy sets based on the fuzzy rule base. The T2FLS rule base remains the same as the T1FLS's but its membership functions (MFs) are represented by T2FS instead of T1FS. The inference engine combines the fired rules and gives a mapping from input T2FS to output T2FS. The output T2FS from the inference engine are then processed by the type-reducer which combines the output T2FS, performs a centroid

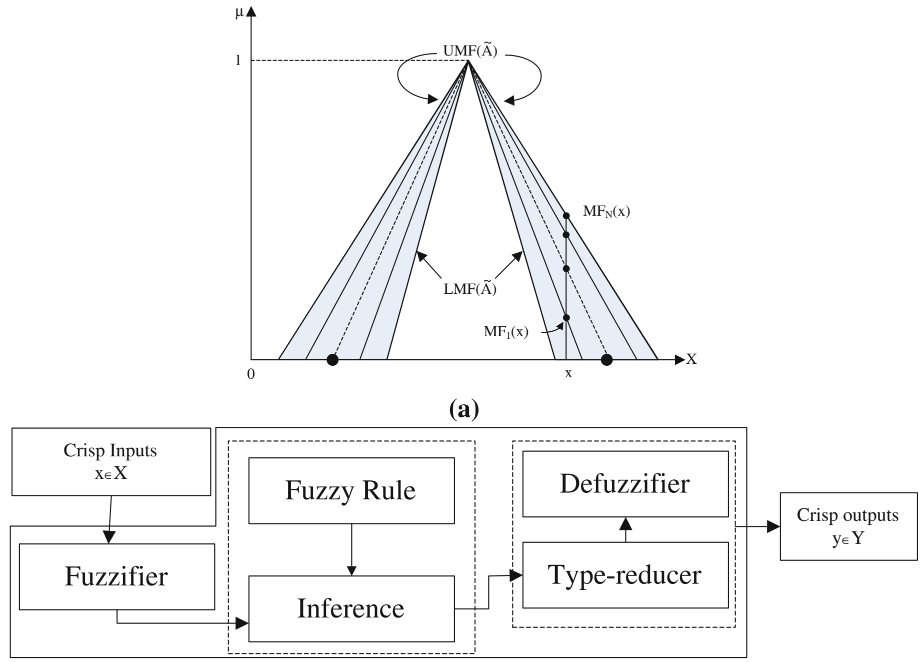

Fig. 1 **a** A type-2 fuzzy set and **b** general structure of the type-2 FLS

calculation, and leads to T1FS, called the Type-Reduced sets (Hagras 2007). There are different types of Type-Reduction methods, including *Centre of Sets*, *centroid*, *simple height*, and *modified height* Type-Reductions (Acampora and Loia 2005; Hagras 2004). In this paper, we use the *Centre of Sets* Type-Reduction to be the type-reduction method as it has a reasonable computational complexity. After the Type-Reduction process, the reduced output T2FSs are defuzzified to obtain crisp outputs that are sent to the actuators.

2.2 Malware behavioral analysis overview

The Internet and personal computers have rapidly advanced (Huang et al. 2011) in recent years so hackers and their malicious software packages like Botnet, Virus, Backdoor, and Trojan, attempt to steal user' data or illegally control computer systems. Such an illegal behavior has been recognized as one of the major security threats to the environment on the Internet such that a large amount of research is being made to try to find effective countermeasures to defend against the hackers' behavior and malware (Inoue et al. 2008). Security researchers are always proposing some new defenses to protect users' personal, valuable, and confidential information. However, they always fall behind the hackers. In other words, the battle between hackers and security researchers never has an ending (Dai et al. 2011).

In order to rapidly defend against unknown malicious attack, many security researchers and traditional malware detection systems use the signature matching techniques to develop an automatic effective analysis tool for detecting malware. However, this approach can be easily circumvented the attack of the malware because the polymorphic characters or metamorphic features of malware will mutate their signatures when the malicious software is spread from one host to another one (Dai et al. 2012). Nevertheless, it is a popular approach for malware analysis (Wagener et al. 2008). On the other hand, behavior-based malware detection approach has a greater potential for identifying previous unknown malware (Dai et al. 2012). Indeed, many researches provide malware analysis for monitoring malware's actions while it is running under a controlled environment like virtual machine (VM). This approach is a so-called virtual machine monitor (VMM), which can identify the malware behavior and what the malware has modified in the file system and/or the registry to quickly recover from the malware infection state. Therefore, a VMM approach is suitable for malware analysis, and most malware analyses are carried out under virtual machines (Wagener et al. 2008; Huang et al. 2010).

However, the transparency of the majority of VMs that are designed to detect the malware is not well enough until now. Malware developers have noticed such a situation that they have developed several techniques such as Anti-VM techniques to detect whether the malware is running under a virtualized environment or not. With the Anti-VM techniques, this causes the hackers to easily find the solutions to detect if the developed malware is running under VM-based environment and then avoid the detection from VMM. In most cases, malware can easily escape from the detection of the VMM to block the behavior of the propagation so that the detected malicious behavior from VM-based malware analysis sometimes may be different from the results of the physical environment.

3 Interval Type-2 fuzzy ontology for malware behavior analysis

3.1 Type-2 fuzzy ontology model

The type-2 fuzzy ontology model is introduced in this section. In order to make both machine and human to understand the designed ontology, Web Ontology Language (OWL) and Fuzzy Markup Language (FML) are both used in this paper to express the built ontology. In addition, we use Protégé to generate OWL for constructing the knowledge base of the ontology, then apply the FML to describe the fuzzy concept of Type-2 fuzzy ontology and perform the fuzzy inference for the malware behavior analysis. Figure 2 shows the built four-layer type-2 fuzzy ontology model by two views, including machine understandability and human semantic understandability. Table 1 shows the mapping and the brief descriptions between these two views. The built ontology model is composed of classes, object properties, data properties, and individuals for the machine understandability (Lee et al. 2005; Lee and Wang 2009), while the ontology model has four layers, including a domain layer, a category layer, a concept layer, and an instance layer for the human semantic understandability. The proposed ontology model can be mapped to the domain ontology for human semantic understandability and to the OWL for machine understandability (Huang et al. 2012; Wang et al. 2009). It enables developers to share common concepts and terms, and allows them to be described in a simple language. The descriptions of the built ontology model are shown as follows:

- There are three relations in the built ontology model, including a generalization, an aggregation, and an association. Their brief descriptions are shown as follows: (1) Generalization is "a-kind-of" or "is-a" relation. It is a way of structuring the description of a single object and relates classes; (2) Aggregation is often called "a-part-of" relation. It is a strong form of association and relates instances. An aggregated object is made up of components. Two distinct objects are involved and one of them is a part of the others; (3) Association is a physical or conceptual connection between object instances and a means to establish relationships among objects and classes.
- The domain name of the built ontology model is interval type-2 fuzzy ontology model.
- The category layer defines several categories labeled as *Category 1*, *Category 2*,…, and *Category n*", which are equally mapped to the classes of Protégé. There exists a generalization relation between the domain name in the domain layer and categories in the category layer.
- The concept layer defines several concepts labeled as "*Fuzzy Variable* FV_1, *Fuzzy Variable* FV_2,…, and *Fuzzy Variable* FV_n." Each concept in the concept layer is related to an instance sets in the instance layer for an application domain via a generalization relation. On the other hand, there exists an aggregation relation between concepts in the concept categories of the category layer. From the machine understandability view, concepts are mapped to the object properties or data properties of Protégé, which are used to express the relations of individuals. Ontology includes a vocabulary of terms, and specifications of their meanings. For example, a vocabulary of terms for FV_1 is *Fuzzy Number* FN_{11}, *Fuzzy Number* FN_{12},…, and *Fuzzy Number* FN_{1n}. The specification of *Fuzzy Number* FN_{11} is $\{[(a_{11}, b_{11}, c_{11}, d_{11}), (e_{11}, f_{11}, g_{11}, h_{11})]\}$, where $(a_{11}, b_{11}, c_{11}, d_{11})$ and $(e_{11}, f_{11}, g_{11}, h_{11})$ represent the parameters of the *begin support*, *begin core*, *end core*, and *end support* of the lower membership function (*LMF*) and the upper membership function (*UMF*), respectively.
- Instance layer contains the instances of the concepts in the concept layer and this layer has a mapping to the individuals of Protégé. There exists an association among instances in the instance layer. Besides, T1FS layer and the T2FS layer are defined in this layer in order to allow Protégé to represent T2FS. Object and data properties in Protégé are used to represent the relations between classes and individuals so there is a generalization between the instance layer and then category layer from the human semantic understandability view.

3.2 IT2FS for malware behavior ontology

Nowadays, many malware analysis toolkits are able to capture the information of the malicious behavior for the computer systems. However, there are very few malware behavioral analysis toolkits which can help security researchers to directly detect the malware after analyzing the captured malicious behavior. Most malware behavioral analysis toolkits still need domain experts to interpret the important semantics for the detected information of the malicious behavior and then judge it is a malware or not.

Therefore, this paper tries to exploit an ontological view of the malware behavior to define a more general and efficient detection methodology. Ontology provides a means to clarify the concepts and semantics of the malware

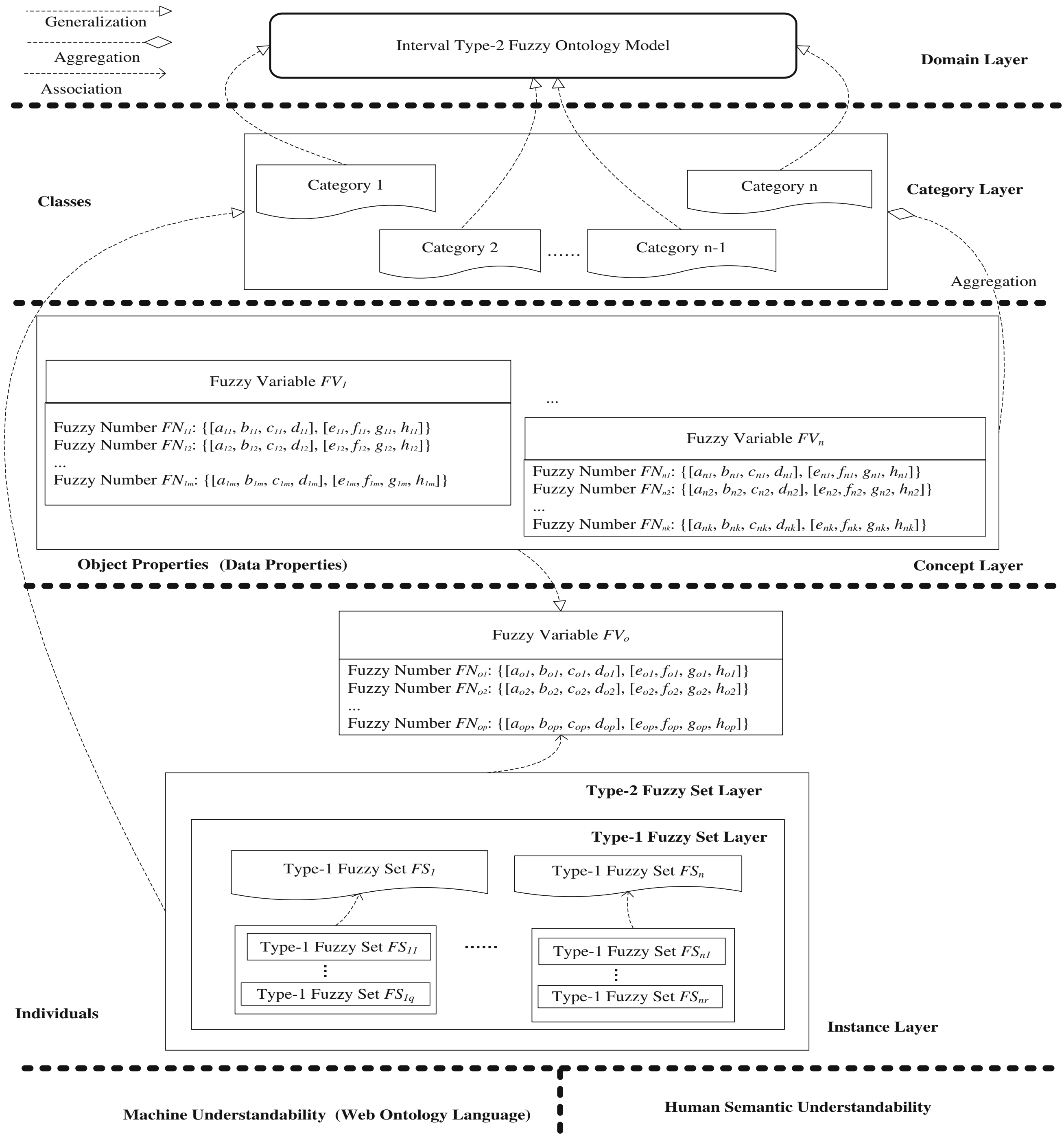

Fig. 2 Structure of the interval type-2 fuzzy ontology model

to avoid from some conceptual confusion. For example, the behavior of the Trojan and Botnets on the Internet may appear the normal one, but it still can capture the malicious connection information. Additionally, ontology can share common concepts or relationships to allow the problems of the malware analysis to be described in a formal semantic platform among intelligent agents or malware behavioral analysis toolkits. Indeed, ontology also includes a vocabulary of terms and the specifications of the terms' meanings to express the relations among concepts and definitions. Figure 3 shows the interval type-2 fuzzy malware behavioral ontology model based on Fig. 2 and its brief descriptions are listed as follows:

Table 1 Mapping between machine understandability and human semantic understandability

Machine understandability (OWL)		Human semantic understandability (FML)
Classes		Category layer
Classes of Protégé and categories of the category layer are both used to be the type or category, which can be defined as an extension or an intension		
Object properties (data properties)		Concept layer
Object properties and data properties with one value are used to link an individual to a class for machine understandability. Object properties and data properties for machine understandability and concepts for human semantic understandability both represent the relations between the input and the output of the ontology		
Type-1 Fuzzy set Type-2 Fuzzy set	Individual	Instance layer
There exist object properties or data properties between classes and individuals, which are mapped to categories and concepts, respectively, for the human semantic understandability. Individuals and instances are the basic components of an ontology		

- Domain layer denotes the name of the ontology, and herein, the domain name is interval type-2 fuzzy malware behavior ontology model.
- Category layer is composed of a variety of types of malware like *Botnet*, *Trojans*, *Backdoors*, *Viruses*, and *Rootkits*.
- Concept layer has some concepts, such as *File Hash* (*FH*), *IP Connection* (*IPC*), and *System Activity* (*SA*). Precisely, *File Hash* is a malware information which is computed by the *ssdeep* toolkit (http://ssdeep.sourceforge.net/), and denotes a hash value bounded in an interval [0, 100] to express the similar level to the known malicious sample. *IP Connection* (*IPC*), ranging between 0 and 100, denotes the counted number of TCP/IP connections from *InetSim* (http://www.inetsim.org/) to express the similar level to a known malicious sample calculated by the regular expression.
- *System Activity* (*SA*) denotes the generated behavioral similarity between the analyzed malicious sample and the known malicious sample which is calculated by the *regular expression* and ranges from 0 to 100. For example, if there is one malware which shows up a hundred kinds of the malicious behavior, one analyzed malicious sample is detected 65 kinds of the identical behavior computed by the regular expression, then *SA* is 65 %. In this paper, we use *Advanced Intrusion Detection Environment* (*AIDE*) (http://aide.sourceforge.net/) and *Network File System* (*NFS*) service (http://sourceforge.net/projects/winnfsd/) for Microsoft Windows to execute the *regular expression*.
- Instance layer contains *Similarity* (*SI*), the type-1 fuzzy set layer, and the type-2 fuzzy set layer. *Similarity* (*SI*) calculates the similarity between an unknown malware and a known malware according to the values of the *FH*, *IPC*, and *SA*, which come from PDF documents, DLL files, Windows Executables, and Office Documents existing in Microsoft Windows XP2, Microsoft Windows XP,…, and so on.

However, even though OWL enables a suitable representation of malware knowledge, it is not able to apply the advanced inference mechanism to derive the additional imprecise and vague knowledge in the scenario of the detection of the malwares. For this reason, we exploit the IT2FS and FML to bridge the gap among other methodologies in this paper. Table 2 lists the brief descriptions for the methodology to integrate IT2FS with ontology for the malware behavior analysis. How to define the parameters of the IT2FS for malware behavior analysis and apply OWL to FML-based soft computing will be presented in the next section. Furthermore, Sect. 4 will show the FML-based malware similarity computing mechanism for more details.

3.3 Web Ontology Language (OWL) for IT2FS-based ontology

Based on Fig. 3 and Table 2, there are some object properties, including (1) *Fuzzy Hash*: *FH_High*, *FH_Median*, and *FH_Low*; (2) *IP Connection*: *IPC_High*, *IPC_Medium*, and *IPC_Low*; (3) *System Activity*: *SA_High*, *SA_Median*, and *SA_Low*; and (4) *Similarity*: *SI_High*, *SI_Median*, and *SI_Low*, to match with the ontology described by Protégé. Figures 4a–c show the screenshots of the protégé to display the object properties, data properties, and ontograf, respectively. Table 3 shows the Partial OWL code for malware behavioral ontology.

4 FML-based malware similarity computing for *MiT*

4.1 Overview of malware analysis network in Taiwan (*MiT*)

Automated malware similarity analysis is definitely not a new technology. There are many published papers about the malware similarity analysis by using a variety of

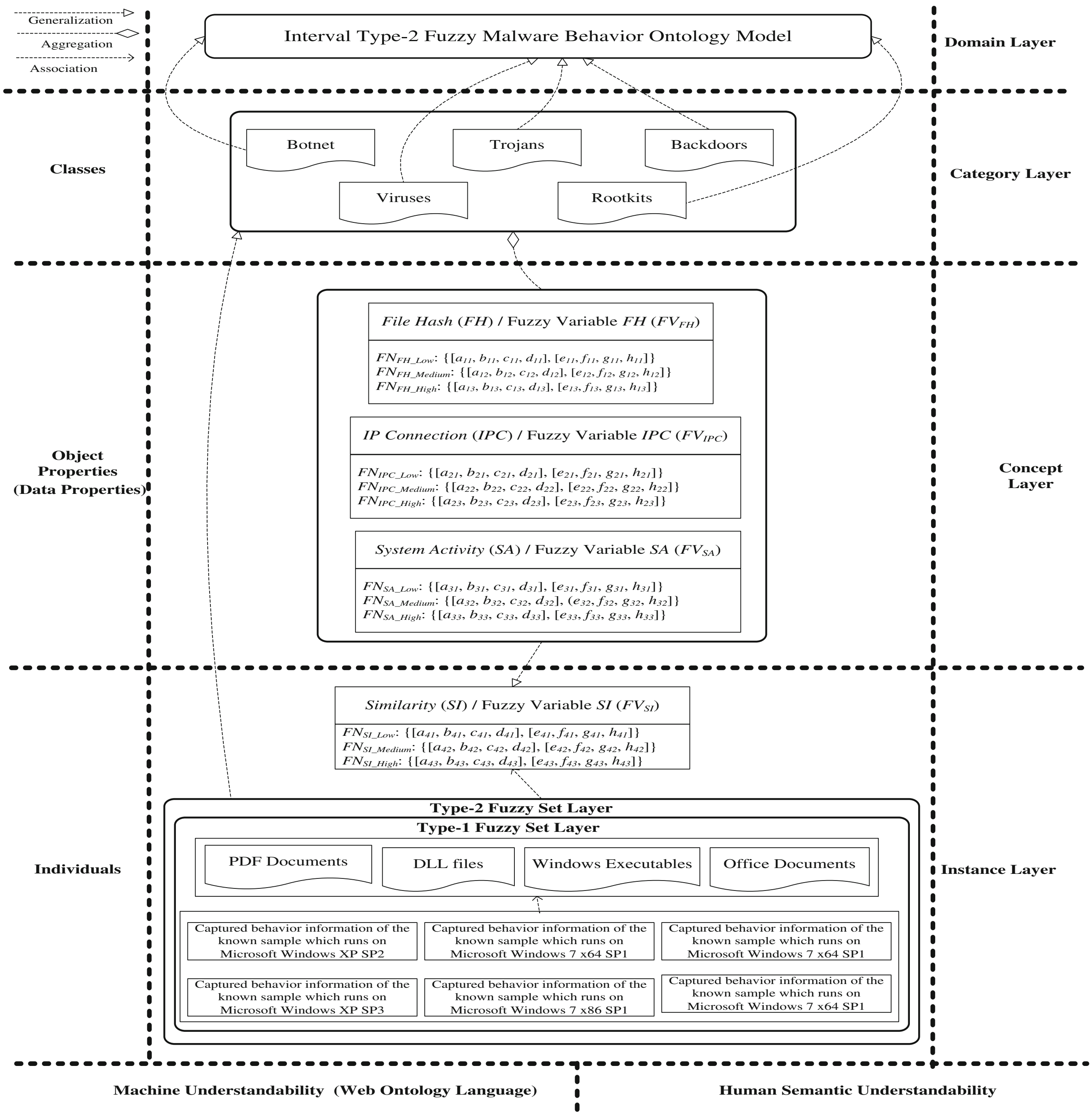

Fig. 3 Structure of the interval type-2 fuzzy ontology model for malware behavioral analysis

techniques. Some of them seem highly effective; however, there are very few papers freely describing their detailed implementations. In this paper, based on our previous physical environment analysis toolkit: *TWMAN* (Huang et al. 2010, 2011, 2012a, b; Inoue et al. 2008), we re-develop and then propose a new generation toolkit to analyze malware behavior (*Malware Analysis Network in Taiwan, MiT;* also known as *MAN in Taiwan*) to resolve some weaknesses of *TWMAN*. We use four items to describe the improvements in *MiT*:

- Mash up a VM as an analyzed platform and pre-check fuzzy hash value by *ssdeep* to improve the weakness. This because when the malware is analyzed under the physical environment, it will take a long time to restore the client's state to start the next analysis;

Table 2 Overview for IT2FS methodology for malware behavior analysis

Input:
Analyzed reports on the unknown malicious samples' behavior
Output:
Interval type-2 fuzzy malware analysis ontology with the variables *Fuzzy Hash*, *IP Connection*, *System Activity*, and *Similarity* in the malware behavioral knowledge base
Method:
Step 1: Interpret the various behavioral logs after comparing with the known malicious samples
Step 1.1: Compute the similarity of the *Fuzzy Hash*
Step 1.2: Compute the similarity of the *IP Connection*
Step 1.3: Compute the similarity of *System Activity*
Step 2: Calculate the *Similarity* based on the *Fuzzy Hash*, *IP Connection*, and *System Activity* to be the input of the IT2FLS
Step 3: Execute the interval type-2 fuzzy inference mechanism
Step 4: Establish the interval type-2 fuzzy malware behavioral knowledge base
Step 5: End

- Establish the developed system in the computer classroom and re-design it to be the distributed structure to decrease the hardware cost;
- Use a *Network File System* (NFS) to make the important directories in the client to directly share with the server to save the time that the system's image stores back to the server. Then, the stored image is matched with the clean one via the *Advanced Intrusion Detection Environment* (*AIDE*) toolkit to extract real-time malicious behavioral information with the *regular expression*;
- Implement the proposed IT2FLS to identify the malware behavior. Therefore, *MiT* is a virtual-physical hybrid environment and has been developed to automate malware behavior analysis, then to detect the unknown malicious software based on known malware, and finally to synchronize the analysis reports and malware samples for all users (Huang et al. 2012a, b; Lee et al. 2010) to resolve the above-mentioned troubles. Figure 5 shows the system structure and workflow of the *MiT* and Fig. 6 shows its component structure. Its operations are listed in Table 4.

4.2 Malware behavior knowledge base for MiT

Fuzzy Markup Language (FML) is a fuzzy-based markup language that can handle fuzzy concepts, fuzzy rule base, and the fuzzy inference engine at the same time. It is a novel computer language based on XML technologies for designing and implementing the Fuzzy Logic Controller (FLC) easily. Because FML is based on XML, it allows the designers to model the fuzzy system in a human-readable and hardware independent way. Hence, it is possible to implement the same fuzzy system on different hardware by avoiding additional design and development phases (Lee et al. 2010). To define a fuzzy concept having terms represented by the a type-2 fuzzy set, a tag named ***<Type2FuzzyVariable>*** is nested in ***<KnowledgeBase>*** tag (Lee et al. 2010). In addition, the tag named ***<Type2FuzzyTerm>*** is nested in ***<Type2FuzzyVariable>***. Every ***<Type2FuzzyTerm>*** tag uses two nested tags, ***<UMF>*** and ***<LMF>***, to define the upper MF (UMF) and lower MF (LMF), represented by a type-2 fuzzy set, respectively. In this paper, there are three input type-2 fuzzy variables and one output type-2 fuzzy variable defined in *MiT*. We define three linguistic terms, including *Low*, *Median* and *High* for the input fuzzy variables *File Hash* (*FH*), *IP Connection* (*IPC*), and *System Activity* (*SA*), respectively. Additionally, the output type-2 fuzzy variable *Similarity* (*SI*) also contains three linguistic terms, including *Low*, *Median* and *High* utilized in this paper. Table 5 shows the knowledge base with parameters of type-2 fuzzy sets for *MiT*. Figure 7 shows the type-2 fuzzy sets for the type-2 fuzzy variables *File Hash*, *IP Connection*, *System Activity*, and *Similarity*.

4.3 FML-based malware similarity computing

A rule base is regarded as the type-2 FML rule base if at least one of the considered fuzzy variables is a type-2 fuzzy concept (Lee et al. 2010; Acampora et al. 2012). Fuzzy inference mechanism defines the mapping from a given input T2FS to an output T2FS using the techniques of the Fuzzy Logic. Generally speaking, the fuzzy rules of a fuzzy system are the linguistics of IF–THEN statements involving fuzzy sets, fuzzy logic, and fuzzy inference to model the domain knowledge and represent the control strategy. Fuzzy rules play a key role in describing the expert control, modeling the knowledge, and linking the input variables of the fuzzy controllers to one or more output variables. For each rule, the inference engine looks up the membership values of the input fuzzy variables in the antecedent part of the

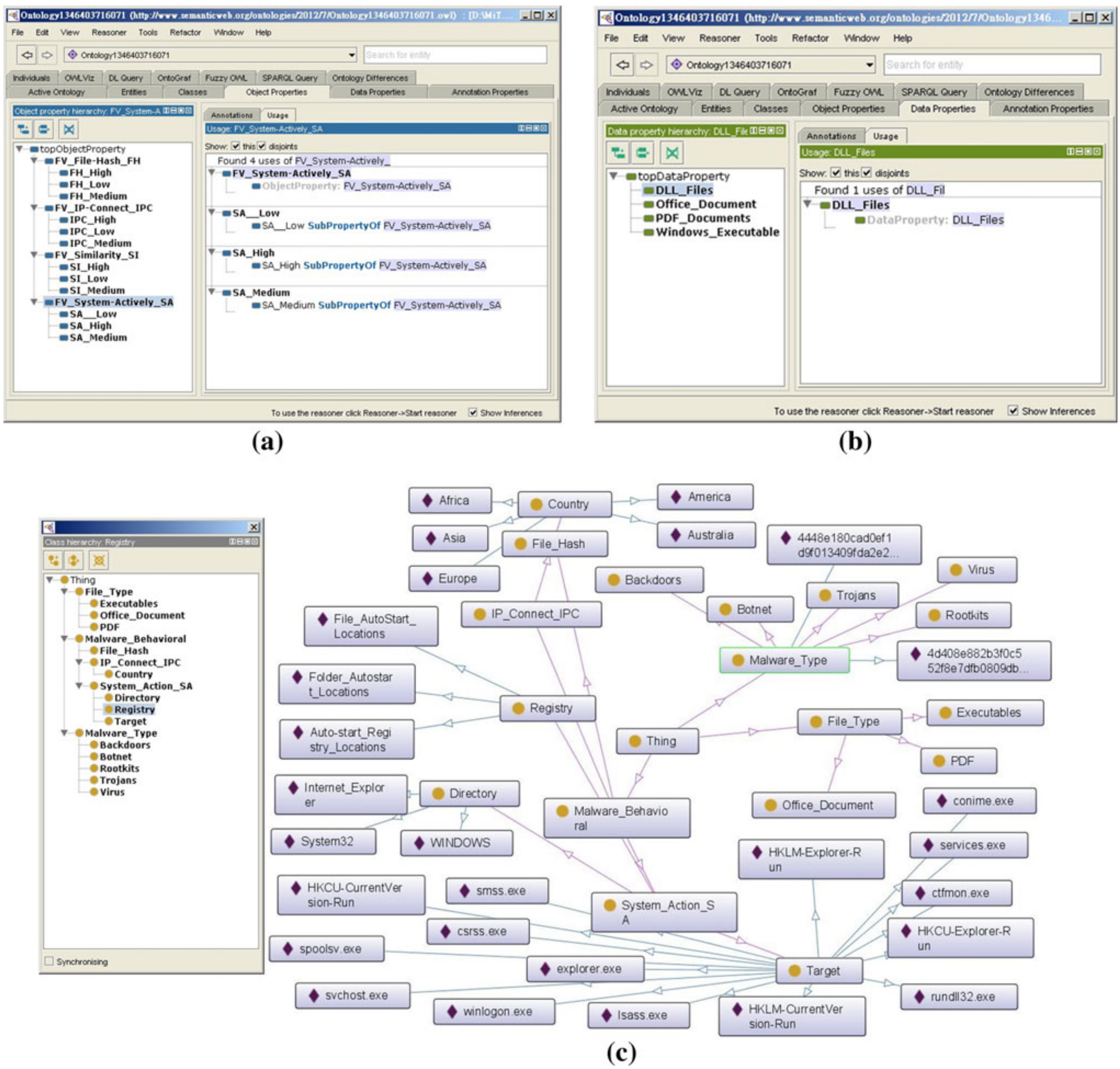

Fig. 4 The screenshot of ontology toolkit: **a** object properties, **b** data properties, and **c** Ontograf

rule. The "activation" of the premise of the rule inducts the conclusion of the rule, i.e., the outcome for the output fuzzy variable in the consequent part. Figure 8 shows the FML-based malware similarity computing structure for *MiT*. Table 6 shows the rule base of the FML-based *MiT*. Table 7 shows the partial FML code for the malware similarity computing.

5 Simulation results

Advanced Persistent Threat (APT), one of the novel attacking models by emails on the Internet, is a very serious security problem for the computer system until now. Therefore, our main work is to reduce a complex task of analyzing a huge amount of malware for e-mails to establish a knowledge model for future analysis work. Based on *MiT*, we partnered with Acer eDC company in Taiwan to produce a scanner for the e-mail attachments, then analyze if there exits the malware, and finally generate the reports. In this paper, we first download the 1,360 known malicious samples from malwaretipss (http://malwaretips.com/) and construct the established physical-virtual hybrid environment for testing the proposed approach. Second, the collected 1,360 known malicious samples are used as the compared baseline. Third,

Table 3 Partial OWL code for malware behavioral ontology

```
<?xml version="1.0"?>
<!DOCTYPE Ontology [
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY xml "http://www.w3.org/XML/1998/namespace" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >]>
  <Ontology xmlns="http://www.w3.org/2002/07/owl#"
       xml:base="http://www.semanticweb.org/ontologies/2012/7/Ontology1346403716071.owl"
       xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
       xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
       xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
       xmlns:xml="http://www.w3.org/XML/1998/namespace"
       ontologyIRI="http://www.semanticweb.org/ontologies/2012/7/Ontology1346403716071.owl">
      <Prefix name="xsd" IRI="http://www.w3.org/2001/XMLSchema#"/>
      <Prefix name="owl" IRI="http://www.w3.org/2002/07/owl#"/>
      <Prefix name="" IRI="http://www.w3.org/2002/07/owl#"/>
      <Prefix name="rdf" IRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#"/>
    <Prefix name="rdfs" IRI="http://www.w3.org/2000/01/rdf-schema#"/>
...
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#FH_High"/>
    <ObjectProperty IRI="#FV_File-Hash_FH"/>
    </SubObjectPropertyOf>
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#FH_Low"/>
    <ObjectProperty IRI="#FV_File-Hash_FH"/>
    </SubObjectPropertyOf>
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#FH_Medium"/>
     <ObjectProperty IRI="#FV_File-Hash_FH"/>
<SubObjectPropertyOf>
        <ObjectProperty IRI="#IPC_High"/>
    <ObjectProperty IRI="#FV_IP-Connect_IPC"/>
    </SubObjectPropertyOf>
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#IPC_Low"/>
    <ObjectProperty IRI="#FV_IP-Connect_IPC"/>
    </SubObjectPropertyOf>
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#IPC_Medium"/>
    <ObjectProperty IRI="#FV_IP-Connect_IPC"/>
    </SubObjectPropertyOf>
            <SubObjectPropertyOf>
        <ObjectProperty IRI="#SA_High"/>
                <ObjectProperty
IRI="#FV_System-Actively_SA"/>
            </SubObjectPropertyOf>
            <SubObjectPropertyOf>
                <ObjectProperty IRI="#SA_Medium"/>
<ObjectProperty IRI="#FV_System-Actively_SA"/>
            </SubObjectPropertyOf>
            <SubObjectPropertyOf>
                <ObjectProperty IRI="#SA__Low"/>
<ObjectProperty IRI="#FV_System-Actively_SA"/>
            </SubObjectPropertyOf>
            <SubObjectPropertyOf>
                <ObjectProperty IRI="#SI_High"/>
           <ObjectProperty IRI="#FV_Similarity_SI"/>
            </SubObjectPropertyOf>
            <SubObjectPropertyOf>
                <ObjectProperty IRI="#SI_Low"/>
            <ObjectProperty IRI="#FV_Similarity_SI"/>
            </SubObjectPropertyOf>
            <SubObjectPropertyOf>
                <ObjectProperty IRI="#SI_Medium"/>
            <ObjectProperty IRI="#FV_Similarity_SI"/>
            </SubObjectPropertyOf>
...
    <AnnotationAssertion>
...
    </AnnotationAssertion>
  </Ontology>
```

50 known malicious samples provided by Acer eDC company and additional 20 known non-malicious samples generated by OASE Lab. at National University of Tainan (NUTN) are used as the experimental samples for the proposed IT2FLS. Figures 9a, b show the screenshots of the 1,360 known malicious samples and 50 known malicious samples, respectively.

The proposed IT2FLS is implemented by Python language. According to the collected 1,350 known malicious samples, the proposed IT2FLS generates the similarity of the malwares for 70 experimental samples. Figure 10a shows the screenshot running the proposed IT2FLS. For example, when *file hash* is 70 %, *IP connection* is 70 %, and *system activity* is 70 %, the similar level to the malware is 72 %, which indicates the possibility that the experimental sample is regarded as a malware is *high*. On the contrary, when *file hash* is 17 %, *IP connection* is 34 %, and *system activity* is 15 %, the

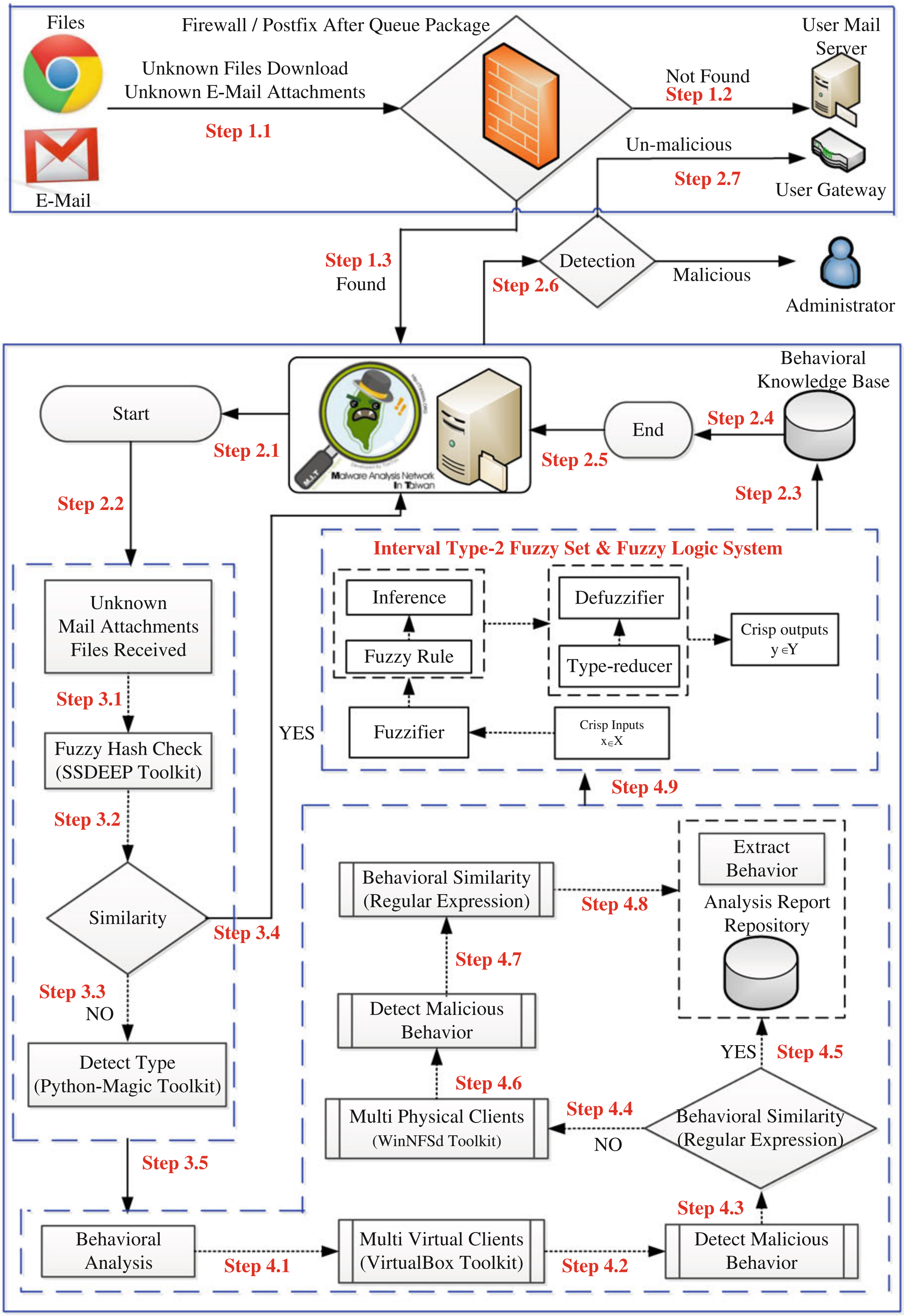

Fig. 5 System structure and workflow of the *MiT*

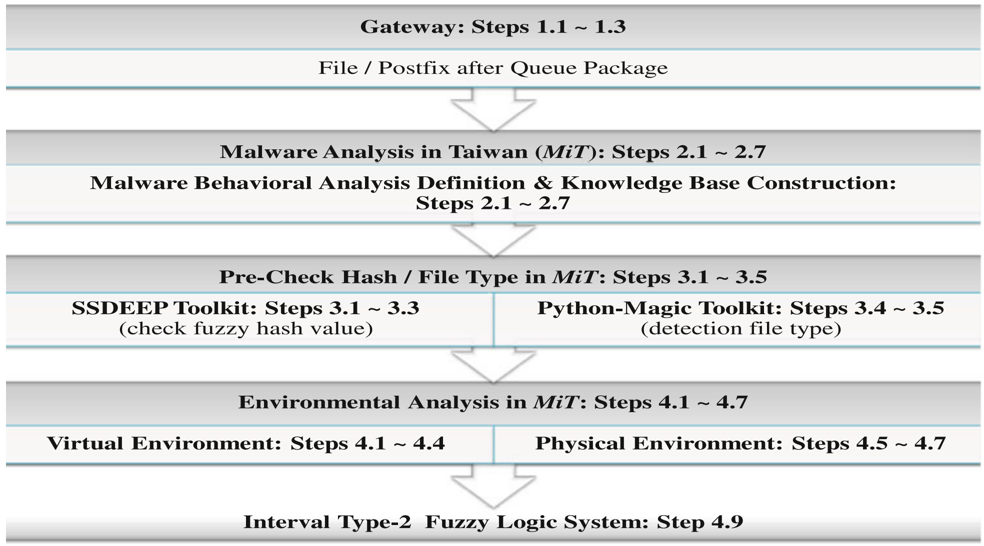

Fig. 6 Component structure of *MiT*

Table 4 Operations of the system structure of the *MiT*

Step 1: Gateway: File/Postfix after Queue Package

Step 1.1: Download the unknown files or attachments in the email

Step 1.2: Firewall/Postfix after queue package doesn't find the possible signatures of the malwares, so pass files or attachments to users

Step 1.3: Firewall/Postfix after queue package finds the possible signatures of the malwares, so pass files or attachments to *MiT*

Step 2: Malware Analysis in Taiwan (*MiT*)

Step 2.1: Enter *MiT*

Step 2.2: Start to execute *MiT*

Step 2.3: Acquire the output of the IT2FS and send the output to the behavioral knowledge base to make a match

Step 2.4: Stop the execution of *MiT*

Step 2.5: Send the matched result back to the *MiT*

Step 2.6: Retrieve the matched result

Step 2.7: If the matched result is un-malicious, the pass the unknown files or attachments to the users. If not, pass them to the administrator and make an alarm

Step 3: Pre-Check hash/file type in *MiT*

Step 3.1: Acquire the unknown files or attachments

Step 3.2: Compute the fuzzy hash values by using ssdeep

Step 3.3: Make a similarity comparison with the known malicious samples used as a baseline

Step 3.4: If the similarity is high, then send the unknown files or attachments back to the *MiT* to reduce the analyzed requests

Step 3.5: If the similarity is not high, then judge the format of the unknown files or attachments via the Python-Magic toolkit to decide to open them by Office or PDF

Step 4: Environmental analysis in *MiT*

Step 4.1: Stat to analyze the behavioral analysis

Step 4.2: Send the unknown files or attachments to the multi-virtual client to do an analysis

Step 4.3: Use the regular expression to compute the unknown files or attachments' behavioral information collected on the VM and make a match with the known malicious samples used as a baseline

Step 4.4: If the matched result is high, then directly send the unknown files or attachments to the analysis report repository

Step 4.5: If the matched result is not high, then send the unknown files or attachments to the multi-physical client

Step 4.6: Proceed with the malicious analysis under the physical environment

Step 4.7: Use the regular expression to compute the unknown files or attachments' behavioral information collected on the physical environment and make a match with the known malicious samples used as a baseline

Step 4.8: Send the unknown files or attachments to the analysis report repository no matter how the matched result is high or not

Step 4.9: Execute the proposed IT2FLS to make an inference

Step 5: End

Table 5 Parameters of type-2 fuzzy sets for *MiT*

Fuzzy variable	Type-2 fuzzy term	T2FS {[*a*, *b*, *c*, *d*], [*e*, *f*, *g*, *h*]}
File Hash (FH)	Low	{[0, 0, 5, 25], [0, 0, 10, 30]}
	Medium	{[15, 35, 45, 65], [10, 30, 50, 70]}
	High	{[55, 75, 90, 90], [50, 70, 90, 90]}
IP Connection (IPC)	Low	{[0, 0, 5, 25], [0, 0, 10, 30]}
	Medium	{[15, 35, 45, 65], [10, 30, 50, 70]}
	High	{[55, 75, 90, 90], [50, 70, 90, 90]}
System Activity (SA)	Low	{[0, 0, 5, 25], [0, 0, 10, 30]}
	Medium	{[15, 35, 45, 65], [10, 30, 50, 70]}
	High	{[55, 75, 90, 90], [50, 70, 90, 90]}
Similarity (SI)	Low	{[6, 18, 18, 30], [0, 18, 18, 36]}
	Medium	{[36, 45, 45, 54], [30, 45, 45, 60]}
	High	{[60, 72, 72, 84], [54, 72, 72, 90]}

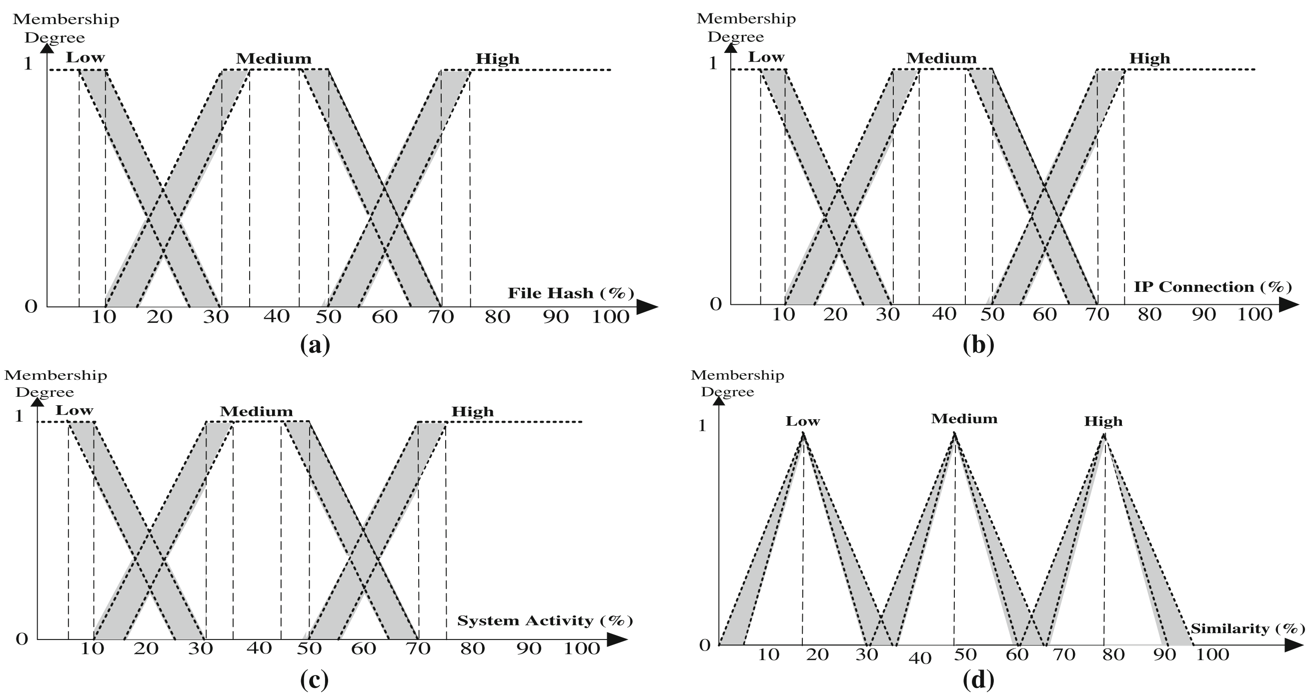

Fig. 7 Type-2 fuzzy sets for the type-2 fuzzy variables: **a** *File Hash*, **b** *IP Connection*, **c** *System Activity*, and **d** *Similarity*

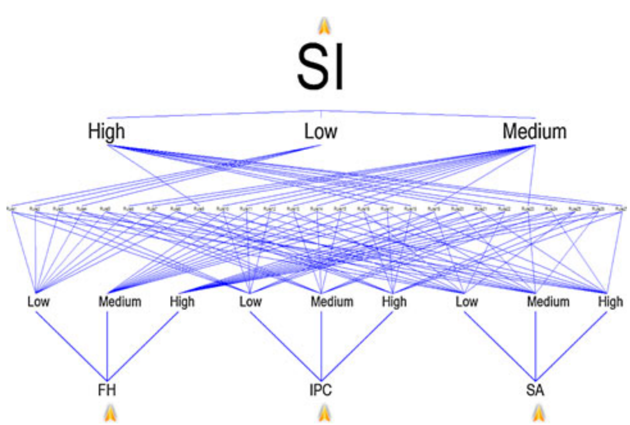

Fig. 8 FML-based malware similarity computing structure for *MiT*

similar level to the malware is 29 %, which indicates the possibility that the experimental sample is regarded as a malware is *low*.

Additionally, in order to further validate the reliability and accuracy of *MiT* with the proposed IT2FLS, we also use the VirusTotal (VT) website (https://www.virustotal.com) to analyze the 70 experimental samples. Figure 10b shows the screenshot running one sample on the VT, which indicates that this experimental sample is analyzed to be a malware by 29 out of 42 Antivirus vendors. Perhaps, the reason that 13 Antivirus vendors cannot recognize this experimental sample as a malware may be caused by the existence of the system's vulnerability or VT does not collect its signature.

Table 6 Fuzzy rules of the FML-based *MiT*

Rule no	Fuzzy variables/fuzzy linguistic terms			
	Input			Output
	File Hash (FH)	*IP Connection* (IPC)	System Activity (SA)	Similarity (SI)
1	Low	Low	Low	Low
2	Low	Low	Medium	Low
3	Low	Low	High	Low
4	Low	Medium	Low	Low
5	Low	Medium	Medium	Medium
6	Low	Medium	High	Medium
7	Low	High	Low	Medium
8	Low	High	Medium	Medium
9	Low	High	High	High
10	Medium	Low	Low	Medium
11	Medium	Low	Medium	Medium
12	Medium	Low	High	Medium
13	Medium	Medium	Low	Medium
14	Medium	Medium	Medium	Medium
15	Medium	Medium	High	Medium
16	Medium	High	Low	Medium
17	Medium	High	Medium	High
18	Medium	High	High	High
19	High	Low	Low	Medium
20	High	Low	Medium	High
21	High	Low	High	High
22	High	Medium	Low	High
23	High	Medium	Medium	Medium
24	High	Medium	High	High
25	High	High	Low	High
26	High	High	Medium	Hugh
27	High	High	High	High

$$Accuracy = \frac{(TN + TP)}{(TP + TN + FP + FN)} \times 100\,\% \quad (1)$$

$$Precision = \frac{TP}{(TP + FP)} \times 100\,\% \quad (2)$$

$$Recall = \frac{TP}{(TP + FN)} \times 100\,\% \quad (3)$$

The performance of the proposed approach is evaluated according to the criteria such as accuracy, precision, and recall. The accuracy, precision, and recall functions are calculated by Eqs. (1), (2) and (3), respectively. The criteria about defining parameters of true positive (*TP*), false positive (*FP*), false negative (*FN*), and true negative (*TN*) are listed in Table 8. *TP* and *TN* denote correct classifications. *FP* denotes the outcome is not correctly predicted as *Yes* but in fact, it is *No*. *FN* denotes the outcome is not correctly predicted as *No*, but in fact, it is *Yes*. Figure 11 shows the curves of accuracy, precision and recall when we use the VT website to simulate the 70 experimental samples. All values of precision are 100 % for each threshold in Fig. 11. The reason is because no any Antivirus vendors on the VT website analyze 20 known non-malicious experimental samples to be a malware so *FP* is always zero no matter what the threshold value is. Besides, Fig. 11 also shows that accuracy and recall has a tendency to decrease when the threshold value is increased. The curves of accuracy, precision and recall for using the IT2FLS to analyze the 70 experimental samples based on the 1,360 known malicious samples are shown in Fig. 12. Most values of accuracy in Fig. 11 are higher than the ones in Fig. 12 when the threshold is higher than 0.5.

The drawbacks when using VT website to make the analysis for the malwares are as follows: (1) If a brand new malware is uploaded to the VT website, the probability that any Antivirus vendor judges it is a malware is relatively very low because these Antivirus vendors have no its signature; (2) For *APT* attack, users cannot know if the attachment contains the malware or not until they manually upload it to the VT website to make the analysis. After that, VT website still cannot give users an answer because VT only tells the users how many Antivirus vendors consider it to be a malware. However, *MiT* with the proposed IT2FLS has some strengths to improve the VT website's weaknesses. Its strengths are as follows: (1) For *APT* attack, *MiT* is able to automatically proceed a malicious analysis. Compared to VT website, *MiT* is much more convenient than the VT website for the users; (2) Current malware-analyzing toolkits on the market only can do the analysis but cannot give users an answer after analyzing the suspicious unknown file or attachment. On the contrary, *MiT* can give users a possibility that the analyzed file or attachment contains a malware; (3) *MiT* can do the malicious analysis no matter whether the malware is with Anti-VM techniques because *MiT* is capable of operating in a virtual-physical hybrid environment; (4) *MiT* can simultaneously proceed the malicious analysis on various operation systems to reduce the probability of making an incorrect judgment only when the malware is actuated under a specific environment.

6 Conclusions and future work

In this paper, we present a novel interval type-2 fuzzy ontology methodology for a malware analysis system to analyze the malware behavior. Analyzing the malware behavior is full of uncertainty, the problem of detaching the similarity behavior from the known malicious behavior to be the baseline becomes even more complicated. To address this problem, the proposed approach with Anti-VM

Table 7 Partial FML code (a) knowledge base and (b) rule base for malware similarity computing

(a)

```
<?xml version="1.0"?>
<FuzzyController ip="localhost" name="">
  <KnowledgeBase>
    <Type2FuzzyVariable domainleft="0" domainright="100" name="FH" scale="" type="input">
      <Type2FuzzyTerm name="Low" hedge="Normal">
        <Type2TrapezoidShape>
          <UMF Param1="0" Param2="0" Param3="10" Param4="30" />
          <LMF Param1="0" Param2="0" Param3="5" Param4="25" />
        </Type2TrapezoidShape>
      </Type2FuzzyTerm>
      <Type2FuzzyTerm name="Medium" hedge="Normal">
        <Type2TrapezoidShape>
          <UMF Param1="10" Param2="30" Param3="50" Param4="70" />
          <LMF Param1="15" Param2="35" Param3="45" Param4="65" />
        </Type2TrapezoidShape>
      </Type2FuzzyTerm>
      <Type2FuzzyTerm name="High" hedge="Normal">
        <Type2TrapezoidShape>
          <UMF Param1="50" Param2="70" Param3="90" Param4="90" />
          <LMF Param1="55" Param2="75" Param3="90" Param4="90" />
        </Type2TrapezoidShape>
      </Type2FuzzyTerm>
    </Type2FuzzyVariable>
…
</KnowledgeBase>
```

(b)

```
<RuleBase activationMethod="MIN" andMethod="MIN" orMethod="MAX" name="RuleBase1"
type="mamdani">
  <Rule name="Rule1" connector="and" weight="1"
operator="MIN">
      <Antecedent>
        <Clause>
          <Variable>FH</Variable>
          <Term>Low</Term>
        </Clause>
        <Clause>
          <Variable>IPC</Variable>
          <Term>Low</Term>
        </Clause>
        <Clause>
          <Variable>SA</Variable>
          <Term>Low</Term>
        </Clause>
      </Antecedent>
      <Consequent>
        <Clause>
          <Variable>SI</Variable>
          <Term>Low</Term>
        </Clause>
      </Consequent>
    </Rule>

<Rule name="Rule27" connector="and" weight="1"
operator="MIN">
      <Antecedent>
        <Clause>
          <Variable>FH</Variable>
          <Term>High</Term>
        </Clause>
        <Clause>
          <Variable>IPC</Variable>
          <Term>High</Term>
        </Clause>
        <Clause>
          <Variable>SA</Variable>
          <Term>High</Term>
        </Clause>
      </Antecedent>
      <Consequent>
        <Clause>
          <Variable>SI</Variable>
          <Term>High</Term>
        </Clause>
      </Consequent>
    </Rule>
…
  </RuleBase>
</FuzzyController>
```

techniques can analyze some kinds of malwares. Compared to the results running on VT website, the simulation results also show the similar results for the malicious detection. In other words, by utilizing the IT2FLS, the proposed system obtains the good result for unknown and uncertain malware’s behavioral extraction and analysis. The

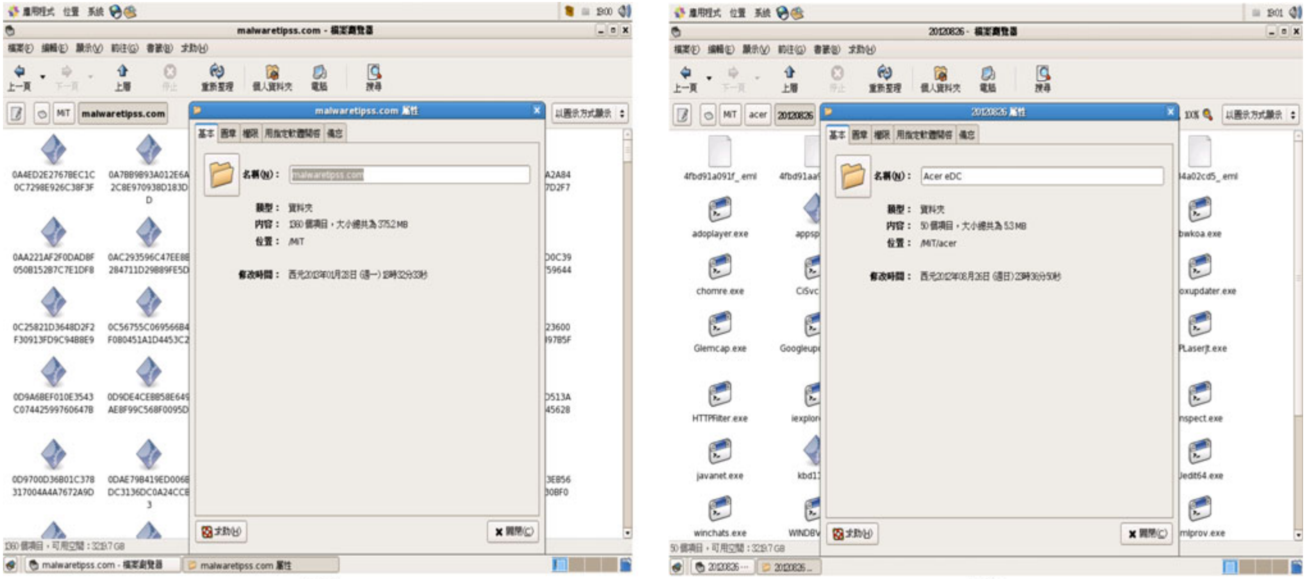

Fig. 9 **a** 1,360 known malicious samples and **b** 50 known malicious samples

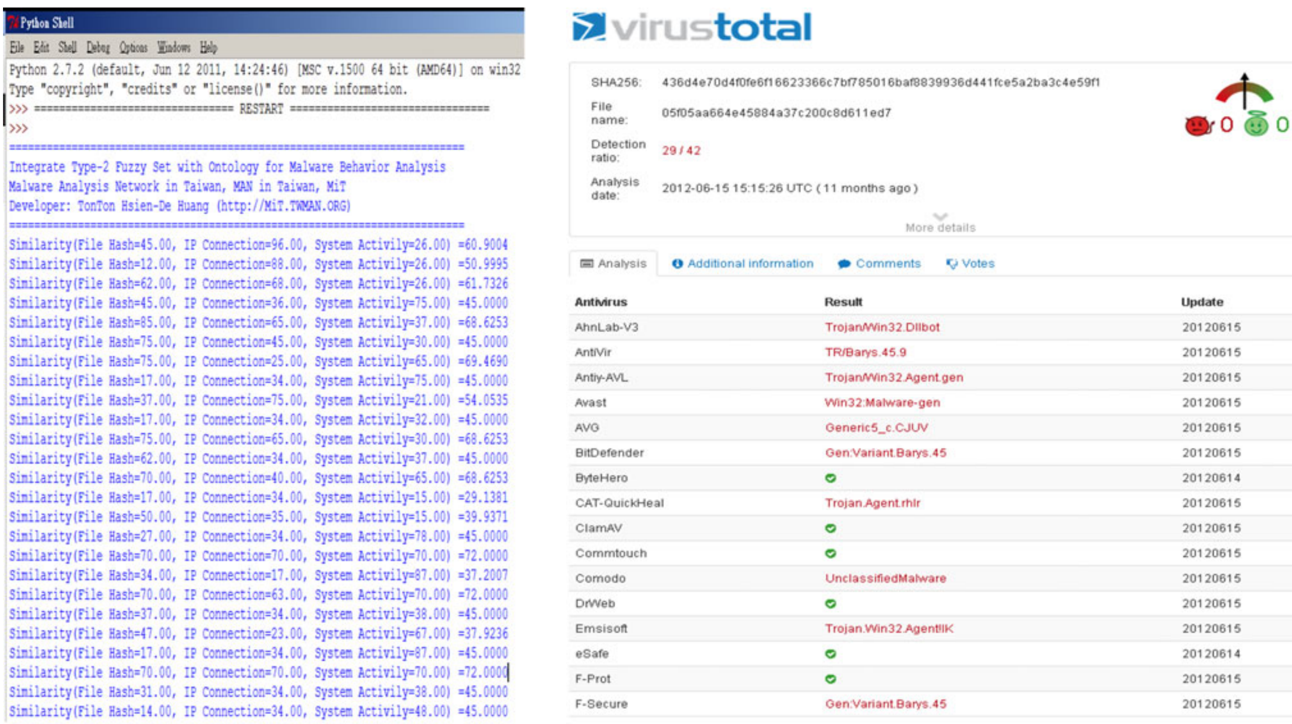

Fig. 10 **a** Screenshots running the proposed IT2FLS and **b** VT

Table 8 Criteria to define *TP*, *FP*, *FN*, and *TN*

Actual results	Prediction results	
	Yes	No
Yes	*TP* (true positive)	*FN* (false negative)
No	*FP* (false positive)	*TN* (true negative)

experimental results also show that the proposed IT2FLS can perform effectively. However, in this paper, *MiT* is still with some drawbacks, for example, (1) it seems impossible to collect all behavior of all possible malwares running in all kinds of operation systems, and (2) the inferred similarity is not enough high when the unknown file or

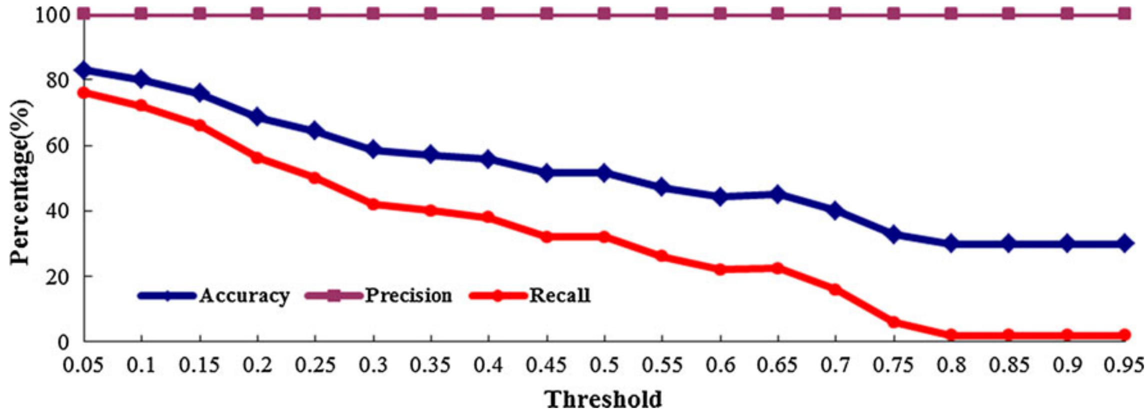

Fig. 11 Accuracy, precision, and recall curves when using VT website to make an analysis

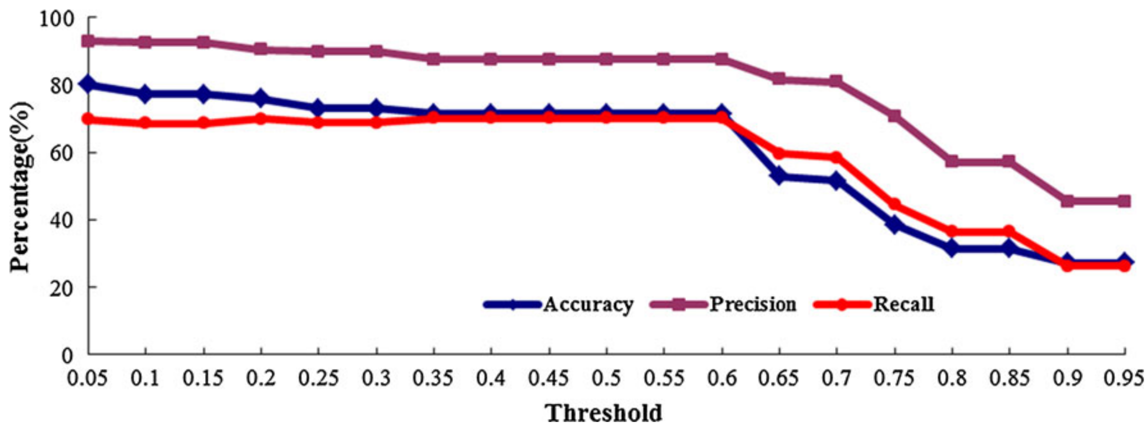

Fig. 12 Accuracy, precision and recall curves when using *MiT* to make an analysis

attachment is a malware, which is about 70–80 % ,and this causes accuracy, precision, and recall to decrease when the threshold value increases. In the future, we will do the following things to improve the current performance:

- Continue to cooperate with Acer eDC company to analyze more malwares to generate the analyzed reports for extracting the malware behaviour. Additionally, we also will generate more certainties to model the malware behavior to improve the accuracy of the analyzed results;
- Further expand to solve more complex problems and provide advanced services such as a cloud service for end users;
- Continue analyzing the behavior of the known malicious samples and define more reasonable range for the T2FS of the fuzzy variable to improve the proposed approach's performance;
- Intend to extend the proposed algorithm to be with the machine learning mechanism which will enable the system to be more robust in the analysis of the malicious behavior and enable the Footprint of Uncertainty (FOU) of the T2FS to be adaptive to the given behavioral conditions.

Acknowledgments The authors would like to thank National Science Council in Taiwan for its financial support under the grant NSC 101-2221-E-024-025. The authors also would like to thank Dept. Information and Learning Technology, National University of Tainan in Taiwan, National Cheng Kung University in Taiwan, and Acer eDC company in Taiwan for their kindly support with the Open Source research project *MiT*.

References

Acampora G, Loia V (2005) Fuzzy control interoperability and scalability for adaptive domotic framework. IEEE Trans Indus Inf 1(2):97–111

Acampora G, Loia V (2007) A proposal of an open ubiquitous fuzzy computing system for ambient intelligence. Comput Intell Agent-based Syst 72:1–27

Acampora G, Lee CS, Vitiello A, Wang MH (2012) Evaluating cardiac health through semantic soft computing techniques. Soft Comput 16(7):1165–1181

Bobillo F, Straccia U (2010) Representing fuzzy ontologies in OWL 2. In: 2010 IEEE World Congress on Computational Intelligence IEEE WCCI 2010, Barcelona, Spain, Jul 18–23, 2010

Carlsson C, Brunelli M, Mezei J (2012) Decision making with a fuzzy ontology. Soft Comput 16(7):1143–1152

Castillo O, Melin P, Alanis A, Montiel O, Sepulveda R (2011) Optimization of interval type-2 fuzzy logic controllers using evolutionary algorithms. Soft Comput 15(6):1145–1160

Dai SY, Fyodor Y, Kuo SY, Wu MW, Huang Y (2011) Malware profiler based on innovative behavior-awareness technique. In: 2011 IEEE 17th pacific rim international symposium on dependable computing (PRDC2011), Pasadena, California, USA, Dec 12–14, 2011

Dai SY, Fyodor Y, Wu MW, Huang Y, Kuo SY (2012) Holography: a behavior-based profiler for malware analysis. J Softw Practice Experience 42:1107–1136

De Maio C, Fenza G, Furno D, Loia V, Senatore S (2012) OWL-FC: an upper ontology for semantic modeling of fuzzy control. Soft Comput 16(7):1153–1164

Hagras H (2004) A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots. IEEE Trans Fuzzy Syst 12(4):524–539

Hagras H (2007) Type-2 FLCs: a new generation of fuzzy controllers. IEEE Comput Intell Mag 2(1):30–43

Hagras H, Wagner C (2012) Towards the widespread use of type-2 fuzzy logic systems in read world applications. IEEE Comput Intell Mag 7(3):14–24

Ho SH, Yang CL, Chen CY, Hsu CY, Chang YK (2009) An intelligent-mamdani inference scheme for healthcare applications based on fuzzy markup language. In: 2009 10th international symposium on pervasive systems, algorithms, and networks (ISPAN2009), Kaohsiung, Taiwan, Dec 14–16, 2009

Huang HD, Chuang TY, Tsai YL, CS Lee (2010) Ontology-based intelligent system for malware behavioral analysis. In: 2010 IEEE world congress on computational intelligence (IEEE WCCI 2010), Barcelona, Spain, Jul 18–23, 2010

Huang HD, Lee CS, Kao HY, Tsai YL, Chang JG (2011) Malware behavioral analysis system: TWMAN. In: 2011 IEEE symposium on computational intelligence for intelligent agent (IEEE SSCI 2011), Paris, France, Apr 11–15, 2011

Huang HD, Acampora G, Loia V, Lee CS, Kao HY (2011) Applying FML and fuzzy ontologies to malware behavioral analysis. In: 2011 IEEE international conference on fuzzy systems (FUZZ-IEEE 2011), Taipei, Taiwan, Jun 27–30, 2011

Huang HD, Lee CS, Hagras H, Kao HY (2012a) TWMAN+: A Type-2 fuzzy ontology model for malware behavior analysis. In: 2012 IEEE international conference on systems, man, and cybernetics (IEEE SMC 2012). COEX, Seoul, Korea, Oct 14–17, 2012

Huang HD, Acampora G, Loia V, Lee CS, Hagras H, Wang MH, Kao HY, Chang JG (2012b) Fuzzy markup language for malware behavioral analysis. In: Acampora G, Lee CS, Wang MH, Loia V (eds) On the power of Fuzzy Markup Language. Springer, Germany, pp 113–131

Inoue D, Yoshioka K, Eto M, Hoshizawa Y, Nakao K (2008) Malware behavior analysis in isolated miniature network for revealing malware's network activity. In: IEEE International Conference on Communications (ICC 2008), Beijing, China, May 19–23, 2008

Lau RYK, Dawei S, Yuefeng L, Cheung TCH, Jin-Xing H (2009) Toward a fuzzy domain ontology extraction method for adaptive e-learning. IEEE Trans Knowl Data Eng 21(6):800–813

Lee CS, Wang MH (2009) Ontology-based computational intelligent multi-agent and its application to CMMI assessment. Appl Intell 30(3):203–219

Lee CS, Jian ZW, Huang LK (2005) A fuzzy ontology and its application to news summarization. IEEE Trans Syst Man Cybern B Cybern 35(5):859–880

Lee CS, Wang MH, Hagras H (2010a) A Type-2 fuzzy ontology and its application to personal diabetic-diet recommendation. IEEE Trans Fuzzy Syst 18(2):374–395

Lee CS, Wang MH, Acampora G, Hsu CY, Hagras H (2010b) Diet assessment based on type-2 fuzzy ontology and fuzzy markup language. Int J Intell Syst 25(12):1187–1216

Mendel JM (2001) Uncertain rule-based fuzzy logic systems: introduction and new directions. Prentice Hall, Upper Saddle River

Mendel JM (2007) Type-2 fuzzy sets and systems:an overview. IEEE Computational Intelligence Maganine 2:20–29

Mendel JM, John RI, Liu F (2006) Interval type-2 fuzzy logic systems made simple. IEEE Trans Fuzzy Syst 14(6):808–821

Orriols-Puig A, Casillas J (2011) Fuzzy knowledge representation study for incremental learning in data streams and classification problems. Soft Comput 15(12):2389–2414

Quan TT, Siu CH, Fong ACM, Tru HC (2006) Automatic fuzzy ontology generation for semantic web. IEEE Trans Knowl Data Eng 18(6):842–856

Sahab N, Hagras H (2011) Adaptive non-singleton Type-2 fuzzy logic systems: a way forward for handling numerical uncertainties in real world applications. Int J Comput Commun Control 6(3):503–529

Sanchez FG, Bejar RM, Contreras L, Breis JTF, Nieves DC (2006) An ontology-based intelligent system for recruitment. Expert Syst Appl 31(2):248–263

Sun MK, Lin MJ, Chang M, Laih CS, Lin HT (2011) Malware virtualization-resistant behavior detection. In: 2011 IEEE 17th international conference on parallel and distributed systems (ICPADS 2011), Tainan, Taiwan, Dec 7–9

Valiente MC, Garcia-Barriocanal E, Sicilia MA (2012) Applying ontology-based models for supporting integrated software development and it service management processes. IEEE Trans Syst Man Cybern Part C Appl Rev 42(1):61–74

Wagener G, State R, Dulaunoy A (2008) Malware behaviour analysis. J Comput Virol 4(4):279–287

Wang MH, Lee CS, Hsieh KL, Hsu CY, Chang CC (2009) Intelligent ontological multi-agent for healthy diet planning. In: 2009 IEEE international conference on fuzzy system (FUZZ-IEEE 2009), Jeju Island, Korea, Aug 20–24

Wu D (2012) On the fundamental differences between Type-1 and interval Type-2 fuzzy logic controllers. IEEE Trans Fuzzy Syst 20(5):832–848

Yao B, Hagras H, Ghazzawi DA, Alhaddad MJ (2012) An interval Type-2 fuzzy logic system for human silhouette extraction in dynamic environments. In: 2012 International conference on autonomous and intelligent systems (AIS2012), Aviero, Portugal, Jun 25–27, 2012