

國立交通大學

資訊科學研究所

博士論文

設計與實作一個新的物件導向
規則式知識庫平台



Design and Implementation of a New Object-oriented
Rule Base Platform

研究生：林耀聰

指導教授：曾憲雄 博士

中華民國九十三年六月

設計與實作一個新的物件導向 規則式知識庫平台

學生：林耀聰

指導教授：曾憲雄 博士

國立交通大學電機資訊學院

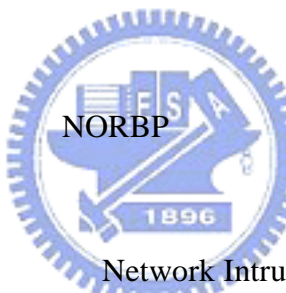
資訊科學系

摘要

近年來，用來模擬專家決策與思考的專家系統，被廣泛的應用在許多電腦資訊系統中，而專家系統所提供的知識處理能力，也成為下一個世代的資訊系統中相當重要的一項特徵。而在專家系統的範疇中，規則式知識庫 (Rule Base) 的技術，則是應用相當廣泛的一種手法，在這類規則式知識庫系統中，專家的知識被表示成容易理解與管理的規則，而規則式知識庫提供了這類資料儲存的功能，以及依據事實推論規則的能力；專家系統的開發者藉著此類工具，便可以在系統中提供知識處理的功能。

本篇論文中，我們提出了一個新的物件導向規則式知識庫平台 (New Object-oriented Rule Base Platform, NORBP)，並在其中提供一個更為彈性，有效率，容易維護，以及更有意義的知識表示法；同時也提供對應此種知識表示法所需的各類機制。在 NORBP 中，定義了這個平台中專屬的知識表示法 (Knowledge Representation) 與其對應的推論法、知識擷取法 (Knowledge Acquisition) 知識探勘法 (Knowledge Discovery) 以及知識融合法 (Knowledge Fusion)，藉由這些機制，提供更為完整的知識庫平台。

針對知識的表示與推論，本論文提出了一個新的物件導向式規則模式 (New Object-oriented Rule Model , NORM)，藉由物件導向的觀念，定義知識模組之間的關係。而在知識擷取方面，我們提出了基於語意距離的概念學習與知識擷取機制 (Concept Learning from Cases based on Semantic Distance for Knowledge Acquisition)；對於知識的探勘，則以資料探勘的技術為基礎，設計了針對使用者行為紀錄找出隱藏知識的知識探勘法 (Knowledge Discovery)。對於利用知識探勘所找出來的知識，最後我們可以利用知識融合的技術 (Knowledge Fusion)，將之與現有的知識加以融合，加強知識的結構並減少知識重複的狀況，藉此提昇知識庫中的知識品質。經過這幾個階段不斷的精練，可以讓利用此知識平台所建立的專家系統更為精確有效，提昇專家系統的效率與品質。



在本論文最後，我們實作了 NORBP 中各項的機制，並且進行相關的實驗，同時為了驗證整個系統的實用性，我們以電腦輔助教學 (Computer Assisted Learning , CAL) 與網路入侵偵測 (Network Intrusion Detection , NID) 為領域，設計兩套對應的專家系統作為本論文的實例說明。其中針對網路入侵偵測部分所實作的網路入侵偵測專家系統(NID-ES)，包含了整套 NORBP 架構的各項機制，藉此實作整套專家系統發展維護的生命週期 (Lifecycle)，並驗證本論文所提出架構在實際應用上的完整性。

關鍵詞：規則式知識庫，知識管理，知識表示，知識擷取，知識探勘，知識融合

Design and Implementation of a New Object-oriented Rule Base Platform

Student: Yao-Tsung Lin

Advisor: Dr. Shian-Shyong Tseng

Department of Computer and Information Science

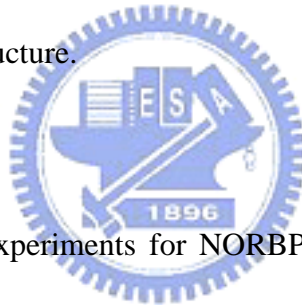
National Chiao Tung University

Abstract

In recent years, Expert System, the system to model expert's decision making process and help to build up knowledge systems, becomes more and more important in Computer Science domain for next generation computer systems. For constructing an Expert System, Rule Base is a widely used approach, where knowledge and expertise are represented as rules, a well-known logical knowledge representation. An inference engine is also part of a rule base, which can be used to process the rules of knowledge and inference as a human expert. It is easy to construct knowledge system using rule base since the representation of knowledge, the storage of knowledge, and the processing of knowledge are well designed in rule base system.

In this thesis, a New Object-oriented Rule Base Platform (NORBP) is proposed, which is designed to provide more flexible, efficient, maintainable, and meaningful knowledge representation, and also correspondingly knowledge systems mechanisms based on the lifecycle of an expert system we defined. In NORBP, several corresponding mechanisms are designed to provide a complete knowledge platform, including knowledge representation, knowledge acquisition, knowledge discovery,

and knowledge fusion. In NORBP, the New Object-oriented Rule Model (NORM) is designed to represent knowledge according to Object-oriented concept, and knowledge relations are defined to construct the knowledge model. In order to provide KA methodology in NORBP, Concept Learning from Cases based on Semantic Distance for Knowledge Acquisition is proposed based on NORM concepts. Moreover, to acquire the knowledge of users daily behaviors, Knowledge Discovery mechanism is used for extracting knowledge from huge amount of user activities. Newly discovered knowledge in Knowledge Discovery mechanism may be redundant or conflict to existing knowledge, and Knowledge Fusion mechanism in NORBP is proposed to fuse different knowledge sources for the same knowledge domain, resolve the conflict and redundant of knowledge, and reconstruct the knowledge model in more meaningful structure.



Some implementations and experiments for NORBP are also done in this work. A Computer Assisted Learning Expert System (CAL-ES) and a Network Intrusion Detection Expert System (NID-ES) are proposed as case studies for NORBP. In the NID-ES, the mechanisms for complete NORBP lifecycle are designed, in which four systems, including Two-Layer Network Intrusion Detection System, Intrusion Detection Knowledge Acquisition System, Intrusion Detection Knowledge Mining System, and Intrusion Detection Knowledge Bases Fusion System, are implemented according to each phase in NORBP.

Keywords: Rule Base, Knowledge Management, Knowledge Representation, Knowledge Acquisition, Knowledge Discovery, Knowledge Fusion

誌謝

畢業不是結束，而是新的研究生涯的起點。在這篇博士論文完成的同時，我更能深刻體認到自己在學術領域的渺小，並期許以此警惕自己隨時保持謙虛內斂的心情，不要因為這一點點的學術成就而自滿，更不要因此懈怠，忘記自己在研究這條路上的初衷。而畢業之後，雖然離開了交大這個朝夕相處的學術殿堂，但這碩博士七年來殷殷學風的薰陶，所帶來的影響將不僅止於此時此刻，更是深刻雋永的人生歷練。

回首來時路，當初自己在碩士畢業、邁入博士研究前所給自己的期望「不僅在學術領域上有所突破，更從而培養些許的領袖氣質」，現在看來，這博士研究五年生涯，所獲得的收穫更不只於此；如果說自己達成這了一點點的成就，首先就必須歸功於我的指導教授 曾憲雄 博士，由於他的指導，不單單讓我自己學術上奠定下一點點的根基，更重要的是，在待人處世方面，他更提供了一個良好的典範，而在事務的處理上，更讓我在不斷的磨練與指導修正中，學習如何成功的經營工作與人生；這些一切的收穫，遠遠超過完成博士論文，取得博士學位所帶來的意義；溢於文字外的心情僅能在此致上最深的感激。

此論文的完成，也非常感謝從論文研究計畫、校內口試到校外口試一路給予我許多論文修改建議的 孫春在 教授與 袁賢銘 教授；在校內口試中，從不同領域觀點給予我指導的 施仁忠 教授；以及在校外口試中就目前相關研究現況，給予我寶貴意見的清華大學 蘇豐文 教授、中央大學 李允中 教授、中央研究院 陳孟彰 博士與暨南國際大學 黃國禎 教授，由於他們的協助，讓此論文最後的成果能夠更加完整及契合主題。

當然更不能忘記的，是知識工程實驗室的夥伴們，在這五年中的交誼、討論、切磋、指教，都是我能夠順利走到這裡的助力；而後續的研究工作，更有賴於這些夥伴們不間斷的合作與努力。

當初毅然決定攻讀博士學位時，一直支持我的家人們，則是我持續向前的原動力。包括一路陪我走來，在我沮喪時支持我，在我忙碌時耐心等候的女友 雅婷，隨時給我所有需要的支持與關心的姊姊 苑宜，爸爸、媽媽，以及女友的父母、家人們。在感激之外，我也要把這一點點成就的榮耀，與他們一起分享。

Contents

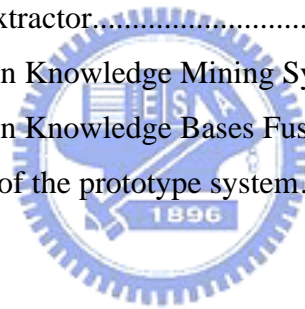
Abstract (In Chinese)	i
Abstract	iii
Acknowledgement	v
Contents	vi
List of Figures	viii
List of Tables	x
List of Algorithms	xi
Chapter 1 Introduction	1
Chapter 2 Related Works	6
2.1 Object-Oriented concept	6
2.2 Knowledge base maintenance.....	7
2.3 Knowledge Engineering.....	8
2.4 Ontology	10
2.5 Rule Base System	11
Chapter 3 A New Object-oriented Rule Base Platform	13
3.1 Lifecycle of an Expert System.....	13
3.2 A New Object-oriented Rule Base Platform	14
Chapter 4 New Object-oriented Rule Model (NORM)	17
4.1 Aerial View	18
4.2 New Object-oriented Rule Model (NORM)	22
4.3 Relation-based Inference mechanism	32
4.4 Modeling a Knowledge Base	35
Chapter 5 Knowledge Acquisition	42
5.1 Case Clustering Step	46

5.2 Concept Relationship Constructing Step	53
5.3 Knowledge Extracting Step	55
Chapter 6 Knowledge Discovery	57
6.1 Preprocessing Phase.....	58
6.2 Two-Layer Pattern Discovering Phase (2LPD)	64
6.3 Pattern Explanation Phase.....	69
Chapter 7 Knowledge Fusion.....	70
7.1 Relationship Graph and Partitioning Criteria	71
7.2 Knowledge Fusion Framework.....	78
Chapter 8 Implementation and Experiments.....	89
8.1 Implementation of NORM.....	89
8.2 Implementation of KA mechanism.....	91
8.3 Implementation of KF mechanism.....	95
8.4 Case Study: Computer Assisted Learning (CAL) Expert System	99
Chapter 9 A Network Intrusion Detection Expert System (NID-ES).....	108
9.1 The Architecture of Network Intrusion Detection Expert System (NID-ES)	108
9.2 Knowledge Representation and Detection Engine Design in NID-ES.....	111
9.3 Knowledge Acquisition in NID-ES	117
9.4 Knowledge Discovery in NID-ES	122
9.5 Knowledge Fusion in NID-ES	125
9.6 Discuss of NID-ES	128
Chapter 10 Conclusion	130
Reference	134

List of Figures

Figure 3.1: Lifecycle of knowledge management.....	13
Figure 3.2: Architecture of NORBP.....	15
Figure 4.1: The learning activity.....	19
Figure 4.2: Binding the new and existent knowledge in learning activity.....	20
Figure 4.3: The behavior of pondering over known information	21
Figure 4.4: New Object-oriented Rule Model (NORM).....	23
Figure 4.5: The knowledge class in a Rule-Base.....	24
Figure 4.6: A Reference relation example.....	29
Figure 4.7: The Reference relation and the Extension-of relation.....	30
Figure 4.8: The forward relation-based inference scheme.....	32
Figure 4.9: The Trigger action and Acquire action	34
Figure 4.10: A transformer example	34
Figure 4.11: The cooperation of KCs with different types of knowledge	38
Figure 5.1: The phases of Concept Learning	45
Figure 6.1: The Concept Diagram of Our Method.	58
Figure 6.2: Data Flow of Preprocessing Phase.	63
Figure 6.3: The Concept of 2LPD Phase	64
Figure 7.1: A relationship graph G_1	74
Figure 7.2: Part of the shared vocabulary ontology.....	77
Figure 7.3: The relationship graph G_2	77
Figure 7.4: The knowledge fusion framework.....	79
Figure 7.5: The un-partitioned relationship graph	83
Figure 7.6: The un-partitioned, pseudo-rules-added relationship graph.....	83
Figure 7.7: The relationship graph G_3 , before removing the pseudo rules	85
Figure 7.8: The relationship graph G_3	85
Figure 7.9: Ontology of RuleClass c_1	88
Figure 8.1: DRAMA Console	89
Figure 8.2: DRAMA Knowledge Extractor	90
Figure 8.3: DRAMA Rule Editor.....	90
Figure 8.4: The shared vocabulary ontology built by the domain expert.....	96
Figure 8.5: The execution time of the algorithms.....	98

Figure 8.6: The memory usage of the algorithms	98
Figure 8.7: Components for Learning Content Selection System	100
Figure 8.8: The architecture of prototype system	104
Figure 8.9: Login page of the NORM based Learning Management System	106
Figure 8.10: Selecting Learning content	107
Figure 9.1: The concept of NID-ES	109
Figure 9.2: The architecture of NID-ES	110
Figure 9.3: The architecture of Two-Layer Intrusion Detection System	112
Figure 9.4: MDE Server.....	115
Figure 9.5: Received IDML event.	115
Figure 9.5: The architecture of prototype system	119
Figure 9.5: Some screen shots of prototype system.....	120
Figure 9.5: The concept hirearchy of DoS.....	120
Figure 9.6: DRAMA editor	121
Figure 9.7: Using DRAMA extractor.....	122
Figure 9.8: Intrusion Detection Knowledge Mining System	123
Figure 9.9: Intrusion Detection Knowledge Bases Fusion System.....	126
Figure 9.10: The screenshots of the prototype system.....	128



List of Tables

Table 6.1: The Format of Standard Log Information.....	60
Table 7.1. The calculated semantic distances	77
Table 7.2: The classes and relationships of the generated ontology	87
Table 8.1. The experimental datasets of randomly selected categories	94
Table 8.2. The experimental result for the dataset in Table 7	94
Table 8.3. The datasets with different numbers of categories contained	94
Table 8.4. The experimental result for the dataset in Table 9	95
Table 8.5: Original categories and number of rules	95
Table 8.6: The partitions and rules $(\{k, l\}=\{1.0, 0\})$	96
Table 8.7: The partitions and rules $(\{k, l\}=\{1.0, 1.0\})$	97
Table 8.8: The partitions and rules $(\{k, l\}=\{2.0, 2.0\})$	97
Table 9.1: KDDCUP selected features.....	124



List of Algorithms

Algorithm 5.1. Knowledge Feature Clustering Algorithm	52
Algorithm 5.2. Knowledge Map Design Process.....	54
Algorithm 6.1: ReNumberSort algorithm, $\text{ReNumSort}(E^i)$	61
Algorithm 6.2: Preprocessing Algorithm, $\text{Preprocess}(E)$	63
Algorithm 6.3: Behavior Clustering Algorithm:	66
Algorithm 6.4: Sequential Pattern Mining Algorithm:	67
Algorithm 6.5: Algorithm of Two-Layer Pattern Discovering	68
Algorithm 7.1: Relationship Graph Construction Algorithm	80
Algorithm 7.2: Relationship Graph Partitioning Algorithm	84



Chapter 1 Introduction

In recent years, Expert System, the system to model expert's decision making process and help to build up knowledge systems, becomes more and more important in Computer Science domain for next generation computer systems. For constructing an Expert System, Rule Base is a widely used approach, where knowledge and expertise are represented as rules, a well-known logical knowledge representation. An inference engine is also part of a rule base, which can be used to process the rules of knowledge and inference rules as a human expert. It is easy to construct knowledge system using rule base since the representation of knowledge, the storage of knowledge, and the processing of knowledge are well designed in rule base system.



In this thesis, the lifecycle for a rule base knowledge system construction is first introduced, and the mechanisms in different phases of the lifecycle are also defined. There are four phases in the lifecycle, including knowledge representation/processing, knowledge acquisition, knowledge discovery, and knowledge fusion. To construct an expert system, the representation of knowledge must be first designed and decided, and also corresponding knowledge processing mechanism (inference engine). In order to prepare the knowledge required for constructing an expert system, a knowledge acquisition mechanism is useful to extract knowledge from expert. However, not only expertise should be considered in the expert system, but also the knowledge embedded in user's daily behavior is also the key for building a successful expert system; hence a knowledge discovery mechanism based on some data mining approach can be used. After that, for all the knowledge collected from knowledge acquisition, knowledge discovery, or got from other knowledge system, a knowledge

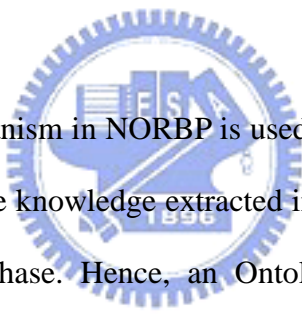
fusion mechanism fuse all the knowledge from different sources for expert system further usage.

According to the lifecycle defined, a New Object-oriented Rule Base Platform (NORBP) is proposed, which is designed to provide more flexible, efficient, maintainable, and meaningful knowledge representation, and also the knowledge systems mechanisms required for building a knowledge system. In NORBP, several corresponding mechanisms are designed to construct a complete platform for knowledge system. For knowledge representation and processing in NORBP, a New Object-oriented Rule Model (NORM) is designed to represent knowledge according to Object-oriented concept, and knowledge relations, including reference, extension-of, trigger, and acquire, are defined to construct the knowledge model. In order to process the knowledge represented in NORM knowledge model, an inference engine is also designed; thus NORM knowledge model has the abilities for knowledge base builder to represent, store, and process the expert knowledge.

In order to acquire knowledge from expert in NORBP, Concept Learning from Cases based on Semantic Distance for Knowledge Acquisition is proposed. In this mechanism, concept hierarchy information is used to extract ontology of the domain, and grouping domain cases into groups according to their semantic relatedness. These groups corresponding to NORM knowledge classes, and the interrelations of these groups will be constructed as the domain ontology. According to the ontology constructed, multi-layer knowledge acquisition mechanism will be used to extract the knowledge from expert by repertory grid approach.

As we have mentioned, knowledge can be not only extracted from experts by

knowledge acquisition, but also retrieved from user daily behaviors. In NORBP, a Knowledge Discovery mechanism based on Data Mining approach is proposed, which can be used to extract knowledge from huge amount of massive data. In this mechanism, there are three phases including Preprocessing phase, Two-Layer Pattern Discovering Phase, and Pattern Explanation Phase. In these three phases, the user behavior features will be first prepared as a feature vector, and then grouping user behaviors using clustering algorithm. After that the user behavior pattern will be mined using sequential pattern mining algorithm, where the pattern is related to the grouped user behaviors, and the mined patterns will be explained according to the signatures of different user behavior group. The explanation will be translated into rules using the patterns and signatures found.



The Knowledge Fusion mechanism in NORBP is used to fuse the knowledge from all different sources, including the knowledge extracted in Knowledge Acquisition phase and Knowledge Discovery phase. Hence, an Ontology-Based Knowledge Fusion Mechanism Using Graph Partitioning is proposed for this purpose. In this phase, rule-formatted knowledge from different sources will be fused by calculating several criteria, including Structural Succinctness Criterion, Intra-Cluster Semantic Clustering Criterion, and Inter-cluster Semantic Clustering Criterion, in which those two semantic criterion are calculated with some concept hierarchy and ontology information. With these criteria, the knowledge are fused not only considering the structural dependency of rules, but also considering the semantic relation of rules to keep the modularity of knowledge for better maintenance and performance.

Some prototypes of these mechanisms are designed and implemented, and also corresponding experiments are done to show the usability of these proposed

algorithms and mechanisms. Moreover, some NORBP mechanisms are used to design a Computer Assisted Learning Expert System, which can be used to solve the issue of adaptive learning in CAL domain. In this CAL-ES, knowledge about how to selection appropriate learning materials are organized as NORM knowledge model, and the inference of these knowledge are also handled by a NORM rule base system – DRAMA, which is a production system implemented according to NORM knowledge model.

A Network Intrusion Detection Expert System (NID-ES), which is designed based on NORBP concepts and utilities, is also proposed in this work to show the practical usage of NORBP. In NID-ES, the corresponding systems for complete lifecycle defined in NORBP are designed and implemented. A Two-Layer Network Intrusion Detection System is first designed to detect the possible intrusion behaviors on the network, in which the rules for intrusion detection are represented in NORM knowledge model, and processed the knowledge using DRAMA rule base system. We also design an Intrusion Detection Knowledge Acquisition System use the knowledge acquisition mechanism in NORBP, with WordNet and DDoS concept hierarchy to calculate the similarities of domain terminologies. According to the network features proposed in KDDCUP 1999, the feature vector for Knowledge Discovery in NORBP is defined, and hence the data mining algorithms designed in Intrusion Detection Knowledge Mining System can be applied for discovering user and intruder behavior patterns, and translated the patterns into rules. Finally, the DDoS concept hierarchy used in Knowledge Acquisition mechanism is also used in Intrusion Detection Knowledge Bases Fusion System to calculate the semantic criteria between rules and hence build the rule classes between the knowledge to be fused. With these four systems, an NID-ES can be constructed according to the lifecycle defined in NORBP.

The rest of thesis is organized as follows. Chapter 2 surveys the background knowledge of this work. Chapter 3 describes the whole architecture and introduces four parts of knowledge management in NORBP. From Chapter 4 to Chapter 7, the details of Knowledge Representation, Knowledge Acquisition, Knowledge Discovery, and Knowledge Fusion, are described respectively. The implementation and experiment of NORBP utilities are described in Chapter 8. In Chapter 9, NID-ES designed based on NORBP is introduced. Chapter 10 gives conclusions of this work.



Chapter 2 Related Works

2.1 Object-Oriented concept


The object-oriented technology provides a way to analyze problem effectively. Although this technology is independent of programming language, various languages that adapt this idea have been designed, e.g., C++, Smalltalk and so on. With those language tools, users can more easily focus on the problem itself without paying too much attention to the language syntax. In addition, some properties of the object-oriented technology, e.g., encapsulation, inheritance, dynamic binding, may improve the maintainability, reusability, and adaptability of software.

Most knowledge systems exploit the object-oriented technology. Based on the object-oriented concepts, knowledge can be divided into some classes. Only the required classes are loaded for inference. Thus, the requirement of system resources can be reduced and the performance can be improved.

The knowledge representation schemes with properties of object-oriented technology are effective on the maintainability of KBS. The property of encapsulation means that only the interface can be used to access the functions or data within a class. Similarly, there is an interface to access the rules or data that are encapsulated in a class of knowledge. Because the details of the knowledge are hidden, this feature can benefit managing a large knowledge base. Based on inheritance, knowledge base system provides the reusability. Moreover, the ability of dynamic binding allows knowledge representation more flexible.

2.2 Knowledge base maintenance

For most knowledge systems, maintaining knowledge is a very important task to keep the systems working properly. For example, when new knowledge comes into a knowledge system, how to combine it with existing knowledge, how to resolve conflicts and redundancies, and how to maintain modularity, etc, are the problems to be considered as the system grows. There are some researches [LT03][TSA02] focus on solving these related issues; hence the knowledge base can be maintained from time to time. When a knowledge system grows, the following issues should be considered:

- 
1. Modularity: Group knowledge into proper units (classes) according to the corresponding knowledge concept; highly modularized knowledge can be managed properly.
 2. Conflication: Avoid the conflication inside the knowledge, the conflication of knowledge may cause the process result of a knowledge base to be uncertain.
 3. Redundancy: Reduce redundant knowledge contained in the knowledge base; redundant knowledge can lower the performance of the knowledge base.
 4. Incomplete: Ensure the knowledge to be complete, which means for any given facts and problem, there is always some results can be obtained.

5. Complexity: Simplify the inter-relation between knowledge; complicated knowledge relationship makes the inference and explanation of knowledge to be harder.

To the best of our knowledge, there are some efficient algorithms have been proposed to deal with confliction, redundancy, and incomplete issues. However, for the modularity and complexity issues, it still lacks a systematic approach. It seems analyzing the knowledge and partitioning knowledge into less complex and more modular structure will be very helpful in the knowledge system maintenance.

2.3 Knowledge Engineering



Knowledge Engineering is the process of structuring, preparing, formalizing, and optimizing information and knowledge. Many topics related to process knowledge is so-called Knowledge Engineering. In this work, several specific types of knowledge engineering process are involved, including Knowledge Representation, Knowledge Acquisition, Data Mining and Knowledge Fusion.

Knowledge Representation is the way to representing and structuring knowledge into computer compliant data structure, and also provides corresponding mechanism to process the data structure of knowledge. There are several general types of knowledge representation, including Rules, Cases, and other special models (Decision Tree, Neuron Nets, etc). Many researches proposed different approaches to deal with all these different kinds of knowledge representation, and good knowledge representation

considering the performance, maintenance of the knowledge is always a major research area of the domain.

Knowledge Acquisition (KA) is a process to extract knowledge from experts or other knowledge sources and transfer the expertise into well-structured form to be used in knowledge based systems. There are quite many different kinds of KA approaches proposed in many researches [RH03][HW03][HY02][NF02][TL99][WW99], including interviewing with experts, Repertory Grids, machine learning, etc. As we know, Knowledge Engineer (KE), who is responsible for executing the process of KA, plays a major role in KA process to elicit the knowledge from experts and transfer the knowledge into structured format; and the preparation done by KE may obviously influence the KA result.

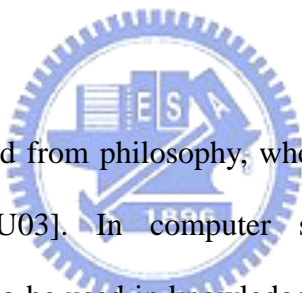


Data Mining is also a research area of knowledge engineering. Mining knowledge from huge amount of data is much more important in recent years since computer systems are widely used in many different areas and hence generate lots of transactions and log information. Quite many data mining researches focus on retrieving deep knowledge contained inside massive raw data, and hence using data mining in knowledge engineering area is becoming be a more and more important.

Expert systems are more and more popular in recent years, and the knowledge for the same domain may be implemented in different expert systems. For example, many Network Intrusion Systems are implemented based on knowledge base technologies, and many of them may have the knowledge for detecting the same intrusion behavior. Researches of Knowledge Fusion are proposed to help knowledge engineer combine the knowledge from different sources. Two main categories of approaches are applied

to the knowledge fusion problem: the hierarchical approaches and the non-hierarchical approaches. Hierarchical approaches include EPAM [FS84], COBWEB [FIS87], CLUSTER/2 [MSD81], CLUSTER/S [SM86], RESEARCHER [LEB86], CLASSIT [GLF90], LABYRINTH [TL89], AutoClass [TL91], SUBDUE [JHC00], and so on. Non-hierarchical approaches include the common subgraph approach [MG95] and the concept lattice approach [GMA95]. The common subgraph approach based on Sowa's conceptual graph and knowledge space [SOW84][SOW00] is efficient and accurate. The concept lattice approach provides an efficient way for knowledge fusion based on the formal concept analysis.

2.4 Ontology



The term ontology is borrowed from philosophy, where an Ontology is a systematic account of Existence [GRU03]. In computer science area, ontology is a conceptualized data structure to be used in knowledge systems or artificial intelligent systems. Based on the same ontology, different systems can communicate with each other, or the knowledge inside computer systems may be structured and presented more accurately.

In recent years, due to the increasing requirement for inducing domain knowledge into computer systems [HY02][NS01][KM03], many researches [AS03][MS01][FF99][ERI03][VAR01][SBA04][CTL03] were proposed to discover, represent, and use of ontology. Especially in knowledge based systems, ontology becomes a key to build a successful knowledge base; with ontology, more meaningful and accurate knowledge content for the users can be presented and used. Thus, building up the

ontology for knowledge system before developing the knowledge content helps lots in the knowledge acquisition process.

2.5 Rule Base System

Rule is a natural knowledge representation, in the form of the “IF ... Then...” structure and Rule Base System (RBS) is popular for real applications among expert systems. RBS consists of two components, inference engine and assertions. The assertions can be divided into a set of facts and a set of rules that can be fired by patterns in facts. The inference engine, an interpreter of an RBS, uses an iterative match-select-act cycling model. In act phase of the cycle, a fired rule may modify or generate some facts.



CLIPS [CLI98], one of the most successful expert system shell, which allows a knowledge base to be partitioned into modules, provides a feature called *defmodule*, and provides a more explicit method for controlling the execution of a system. Each module is able to inference sequentially and independently by inference engine. Different domain knowledge can be placed in different modules created by *defmodule* functions. Logically, related rules and facts can be collected into one module, which provides better maintenance and performance.

RBS has many advantages [REI91]. The first is naturalness of expression since experts rely on rules rather than on textbook knowledge. The second is modularity that permits RBS easy to construct, to debug, and to maintain. Restricted syntax and ability of explanation are also the advantages of RBS. Although RBS is powerful

enough in many applications, it has several disadvantages in maintenance and construction, e.g., the weak ability of incremental construction of knowledge [LO96].

Accordingly, many researches aim to integrate object-oriented and rule-based programming paradigms to take advantage of OO technology. There are two paradigms on the integration of objects and rules: incorporating rules into objects and embedding objects into rules. Knowledge objects are an integration of the object-oriented paradigm with logic rules [WU00]. Furthermore, many rule-base tools, which cooperate with OO technology, have been developed, e.g., COOL (CLIPS Object-Oriented Language) [CLI98].



Chapter 3 A New Object-oriented Rule Base Platform

3.1 Lifecycle of an Expert System

In recent years, Expert System, the system to model expert's decision making process and help to build up knowledge systems, becomes more and more important in Computer Science domain for next generation computers systems [GR89][NEG85][ROE88][SF02][TT02]. Rule Base System is a widely used approach to construct Expert System, in Rule Base System, the expertise is represented as rules, a well-known logical knowledge representation, and the Rule Base System provides the ability to process the knowledge and provide decisions or advices as human expert according to the facts of the environment.

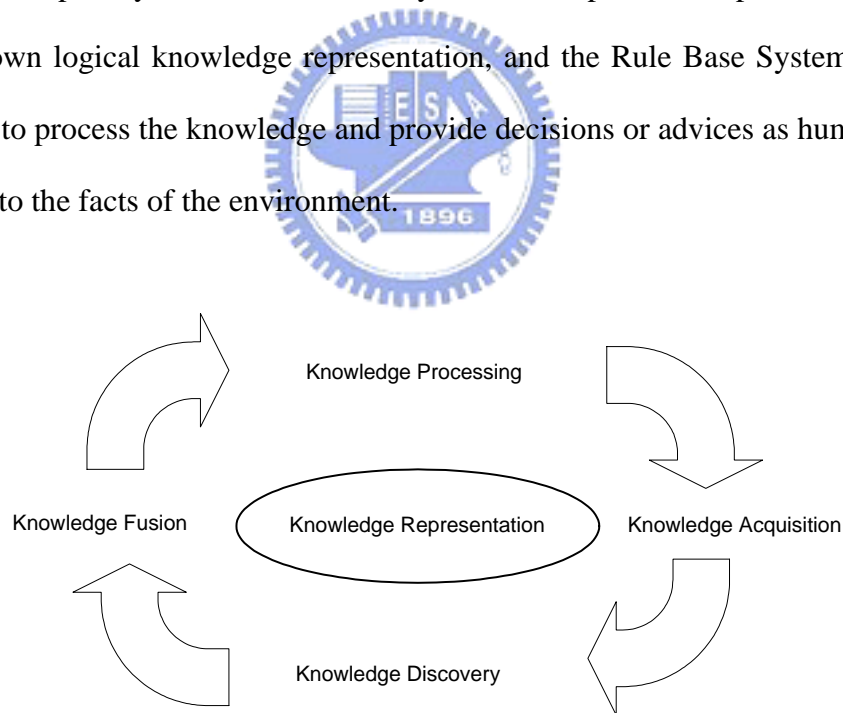
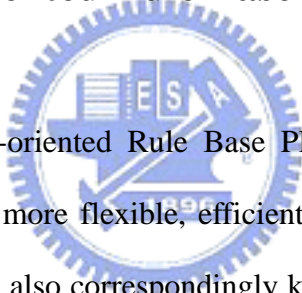


Figure 3.1: Lifecycle of knowledge management

Figure 3.1 shows the lifecycle of Knowledge Management for a Rule Base System. There are four phases for Knowledge Management, including Knowledge Process,

Knowledge Acquisition, Knowledge Discovery, and Knowledge Fusion. In this lifecycle, the way to process and representation knowledge is first selected, and the Knowledge Acquisition (KA) mechanism can be used to retrieve domain knowledge from experts. Also, for a running expert system, Knowledge Discovery mechanism can be used to extract knowledge may be embedded in user's behaviors. For different knowledge sources, e.g., the knowledge retrieved by KA process, the knowledge extracted in Knowledge Extraction process, Knowledge Fusion mechanism can be used to merge the knowledge and make them consistency for Knowledge Processing mechanism to use.

3.2 A New Object-oriented Rule Base Platform



In this thesis, a New Object-oriented Rule Base Platform (NORBP) is proposed, which is designed to provide more flexible, efficient, maintainable, and meaningful knowledge representation, and also correspondingly knowledge systems mechanisms. In NORBP, several corresponding mechanisms are designed to construct a complete knowledge platform, including knowledge representation, knowledge acquisition, knowledge extraction, and knowledge fusion, and the architecture of the knowledge platform is shown in Figure 3.2:

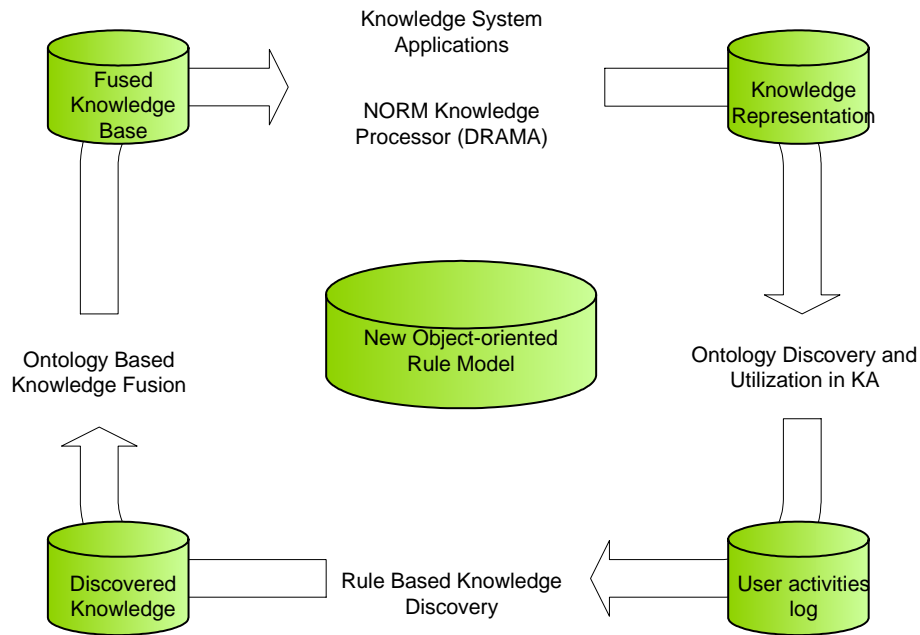


Figure 3.2: Architecture of NORBP

In NORBP, the New Object-oriented Rule Model (NORM) is designed to represent knowledge according to Object-oriented concept, and knowledge relations are defined to construct the knowledge model. By using NORM, knowledge can be organized and structured in more meaningful way according to the natural of human knowledge [DEE65][KIN70][GAG85][GLA87]. However, since NORM is a new designed knowledge model comparing to traditional knowledge model, and the knowledge structure in NORM can not be acquired in traditional knowledge acquisition approach. In NORBP, Concept Learning from Cases based on Semantic Distance for Knowledge Acquisition is proposed to help construct knowledge concepts, extract knowledge relations between concepts, acquire knowledge content in each knowledge concept, and build up the knowledge model of the selected knowledge domain.

Except extract knowledge from expert by KA methodology, for the knowledge hidden in our daily behaviors, machine learning and data mining approaches are usually used

for extracting knowledge from huge amount of massed data. The Knowledge Extraction mechanism proposed in NORBP is designed based on both machine learning and data mining approaches, and provides an efficient and useful methodology to extract patterns from the system records about user behavior, for example, exacting knowledge from the web access log or the user consuming transactions, etc. The patterns discovered are the knowledge about user behaviors and interests.

Even for the same knowledge domain, the knowledge are always increasing due to new discoveries and ideas, and Knowledge Discovery in NORBP provides an efficient way to retrieve new knowledge without re-acquiring knowledge from experts, which reduce the time for learning new knowledge and speedup the life cycle of knowledge. However, sometimes new knowledge extracted may be redundant or conflict to existing knowledge, and also the knowledge model should be re-constructed if the knowledge content is frequently updated; Knowledge Fusion mechanism in NORBP is proposed to fuse different knowledge sources for the same knowledge domain, resolve the conflict and redundant of knowledge, and reconstruct the knowledge model in more meaningful structure.

NORBP provides a definition of life cycle for knowledge management, and the mechanisms designed in NORBP are good tools for each process of knowledge management. In the following chapters, those NORBP tools designed will be detailedly described.

Chapter 4 New Object-oriented Rule Model (NORM)

Recently, knowledge management has become increasingly popular [CL02] [EA02]. Knowledge or expertise of experts in numerous domain should be extracted, managed and reused to improve the performance and reduce human resources needed for difficult tasks. In most cases, knowledge needs to be constructed incrementally no matter what type the knowledge is, and hence maintainability for knowledge base system (KBS) is very important since KBS needs to be updated frequently. According to the above considerations, the following features are important for knowledge maintenance and management. A simple and clear knowledge model with these features is proposed in this chapter.



Modularity

Modular knowledge elements can be used sequentially and independently by inference engine. Modular knowledge representation benefits the maintenance of a KBS because of its localizing the effects of specifying flows of information between modules.

Abstraction

Abstraction is an approach that helps us deal with complexity by emphasizing relevant characteristics and suppressing other details. In most knowledge-based applications, the details of knowledge are not cared about.

Reusability

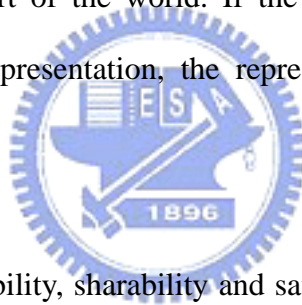
Knowledge reusability provides the facility of using original knowledge to build new knowledge. The property of inheritance is useful for knowledge reusing, yet a mechanism to reduce the knowledge conflict is needed.

Sharability

Sharable knowledge can be used to build up applications on various platforms. In another aspect, different knowledge-based system can also cooperate through the knowledge sharing.

Uncertainty reasoning

Uncertainty is an integral part of the world. If the ability of inexact reasoning is integrated into knowledge representation, the representation will be more natural [SAL93].



In order to increase the reusability, sharability and satisfy modularity and abstraction for knowledge base, a new model, New Object-oriented Rule Model, is proposed for managing rules under object-oriented paradigm.

4.1 Aerial View

Various kinds of knowledge are defined in psychology [GAG85][GAG84]; however expert system mainly deals with the procedural and declarative knowledge excluding motor skill, attitude, etc. Knowledge is constructed by lots of concept blocks, for example, the concept about identifying a bird, a fish and so on. By building ontology to connect different concept, a complete conceptual knowledge model to solve a

problem can be built. According to how people learn knowledge and ponder, three major kinds of relationships are defined between knowledge concepts. Thus, we define a clear knowledge framework and build a corresponding knowledge base system.

4.1.1 Human Learning

Learning is the most significant knowledge activity in our lives. A topic is required before people start the learning activity, for example, “To learn how to identify a bird” is the topic before we learn what a bird is. Knowledge about the topic will be built after successfully studying about the topic as shown in Figure 4.1.

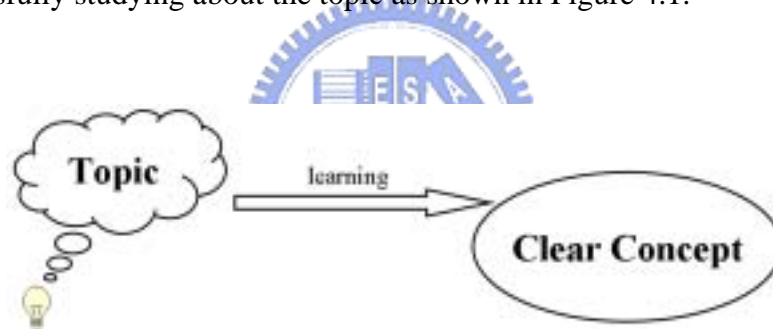


Figure 4.1: The learning activity

4.1.2 Knowledge Class

In this work, *Knowledge Class (KC)* is used to describe each concept. Learning is to study piece of knowledge, e.g., a domain concept, and to convert the knowledge into a KC. All the new knowledge is built upon the original knowledge according to educational psychology. In other words, learning is an activity to construct the

relationship between different KC, as shown in Figure 4.2 [DEE65] [KLA71].

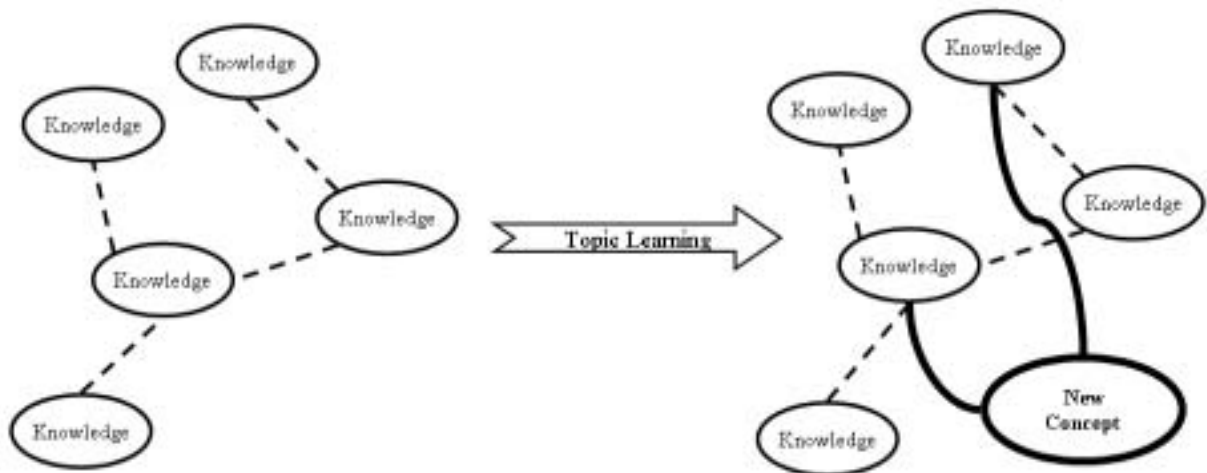


Figure 4.2: Binding the new and existent knowledge in learning activity.

Association



As we build domain knowledge inside our mind, association with existing knowledge is used to reduce the difficulty of learning. This kind of knowledge model is widely used in human knowledge processing. This relationship between domain concepts is seen as *reference*, i.e., to refer some existing knowledge.

Modification and Extension

Modification of knowledge is also a similar activity. Efficient learning is absorbing the existing knowledge and experience from other people, but these knowledge contents may be modified or corrected according to user's experience or some new definitions of knowledge. On the other hand, *extension* is similar to *modification* except that the knowledge can be not only overridden but also extended under

extension relation.

4.1.3 Inferring

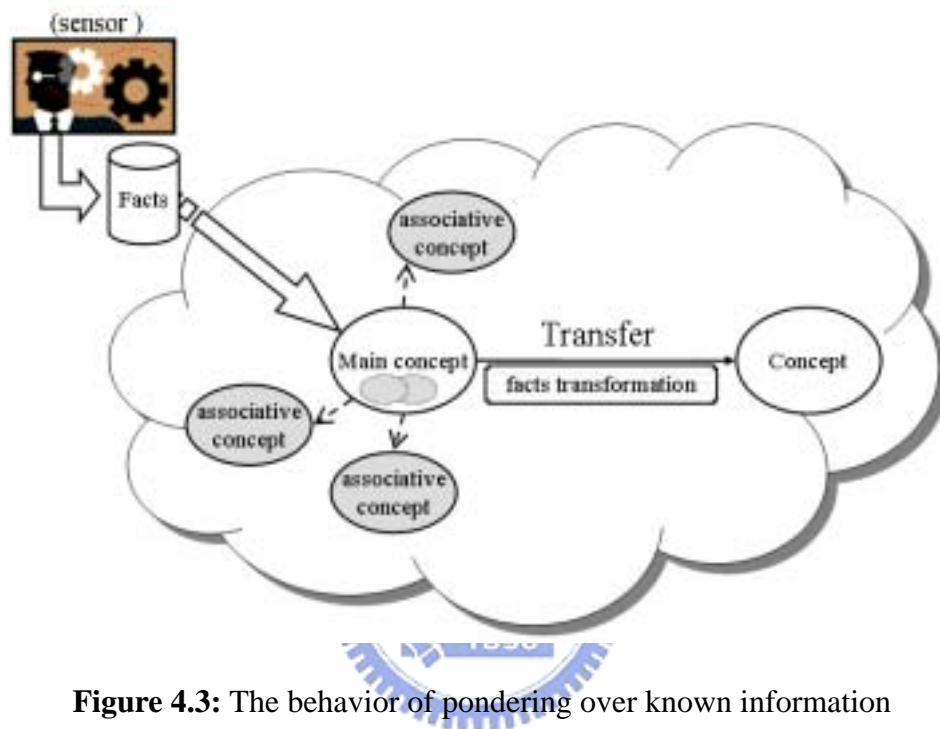


Figure 4.3: The behavior of pondering over known information

As shown in Figure 4.3, when human gets facts through sensor, the facts will be inferred with a specific concept in a domain and other three concepts can be associated according to their relationships. However, people may not consider all relevant knowledge at the same time, since too much effort may be required to solve the problem. Some inference skills are widely used in human thoughts to improve the performance of knowledge inference.

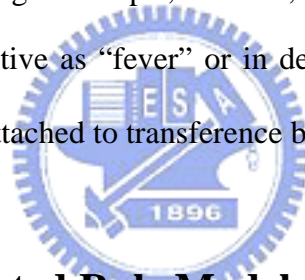
Transference

Sometimes, a problem can be transformed to another problem according to some

conditional judgment. For example, we may consider how to save water if we detect that climate will be drought. The transference is the activity of *triggering* thinking for another concept. On the other hand, a problem can be partitioned into some sub-problems when certain conditions are matched. For example, when a student is bad at mathematics, and then the knowledge of planning an extra mathematics course will be included; otherwise, the knowledge will not be included. This relation between two concepts is treated as *acquirement*.

Fact transform

In addition, the fact might have different name or meaning among concepts. For example, in different knowledge concepts, the fact, “the temperature of the body”, could be represented in adjective as “fever” or in degrees centigrade as “39 °C”. So fact transformations may be attached to transference between two concepts.



4.2 New Object-oriented Rule Model (NORM)

A knowledge model, New Object-oriented Rule Model (NORM), is proposed according to the above ideas in this section. There are various subjects of domain knowledge in mind, but a knowledge system is often concerned with only one domain. However, a subject may contain various concepts.

Because rule is the natural and common representation of knowledge, rule is chosen to represent knowledge of each concept. As shown in Figure 4.4, a rule base is defined as a container that deals with domain knowledge and contains various knowledge classes; hence, related facts collected from real world can be used for inference within

a knowledge class of corresponding concept.

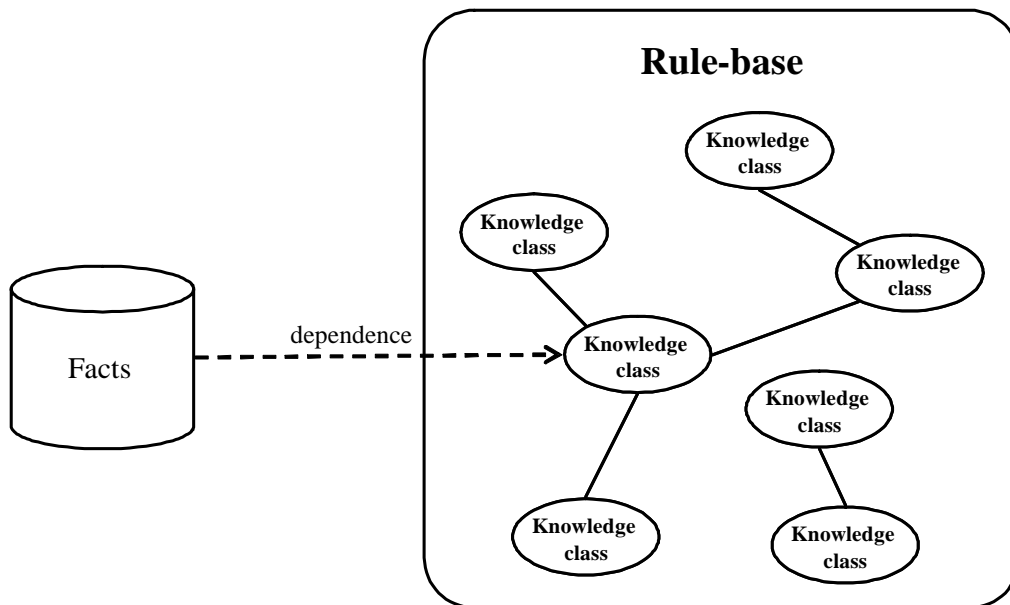


Figure 4.4: New Object-oriented Rule Model (NORM)

4.2.1 Facts and Fact-Collection

The facts represent all kinds of appearance in real world and are used when inferring. During inference process, the rules use facts to obtain reasonable conclusion. A fact consists of name, value, and possibility. A general expression for fact is as follows:

$$F: n = v (p)$$

Where

n: the name of fact, which is used to identify a fact

v: value

p: possibility

The value of a fact could be any type including string, integer, float, date, Boolean value. If the value or type of a fact is unknown, it can be set as NULL. In order to support uncertainty reasoning, the possibility represents degree of belief of a fact. The possibility value is confined to the interval [0, 1]. An activation of 1 is interpreted as “highly positive”, and zero as “uncertain”.

Fact Collection (FC)

Fact collection (FC) is a set of facts and contains the meaningful facts for inferring. An FC performs as working memory and every inference process should own an independent FC. In other words, the FC is a temporary run-time component and will not be stored in a knowledge base system.

4.2.2 Knowledge Class

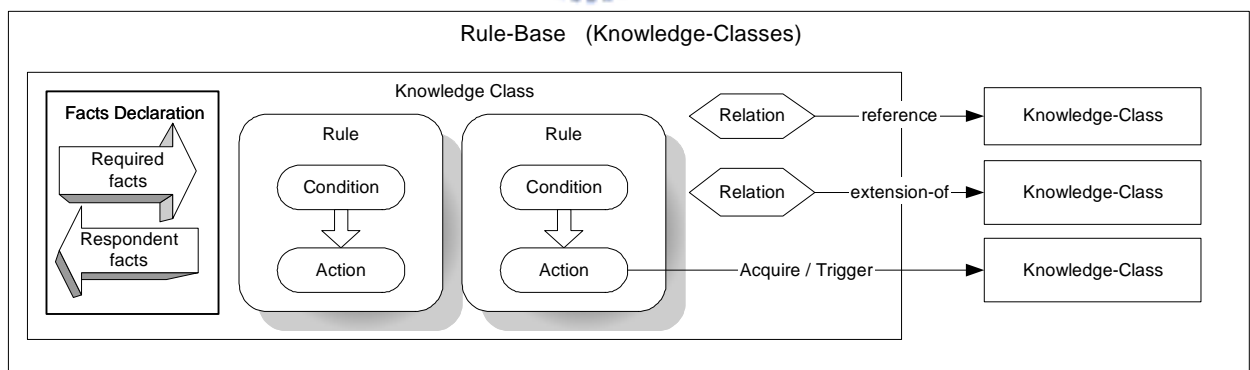


Figure 4.5: The knowledge class in a Rule-Base

A Knowledge Class (KC) represents a kind of concept. It consists of rules, relation with other KCs and fact declarations as shown in Figure 4.5. After aggregating

adequate facts in an FC, the facts could be inferred with a specific KC. During inferring, facts in an FC might be modified or generated. Finally, the conclusion could be drawn from the generated facts.

The fact declarations define which information is meaningful for a KC. There are two types of facts, the respondent facts and the required facts, included in the facts declared. The required facts are prerequisites for inferring under a concept, and on the other hand, the respondent facts are the interests of the conclusion. In other words, required fact is seen as input and the respondent fact as output.

A fact declaration consists of the name of fact and default value. If an FC does not contain some required facts before inferring, these facts should be initiated with the default value. On the other hand, if some respondent facts are not generated after inferring, these facts will be obtained with the default value as well. Thus, the fact declarations could be used to represent declarative knowledge.

4.2.3 Rule

A rule is the basic knowledge element in a rule-based system. The general formulation of a rule is shown as follows:

R: IF c THEN a (CF= μ), t, w

Where

c: condition part of a rule

a: action part of a rule

μ : certainty factor of a rule

t: threshold

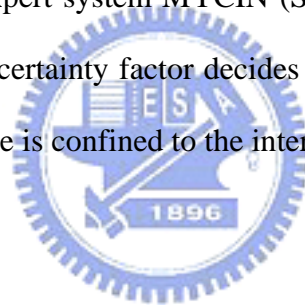
w: weight

Weight

The weight property allows the user to assign the priority to a rule. The rule with the highest priority will be fired first. The weight value should be an integer. If unspecified, the weight value for a rule defaults to zero.

Certainty-factor (CF)

In order to support uncertainty reasoning, the certainty factor model, which was first used in the medical expert system MYCIN (Shortliffe & Buchanan, 1975), is adopted. In CF model, the certainty factor decides the degree of belief of a rule in matching phase and its value is confined to the interval [-1, 1].



Condition

A condition is a Boolean expression, which are the criteria for a piece of knowledge. Various operators can be used in the expression such as arithmetic operator, Boolean operator, etc. In rule matching phase, the result of the Boolean expression is evaluated, i.e., estimating the degree of confidence of a rule. The value is affected by several factors including logical operation and possibility of used facts. Finally, the degree of confidence of a rule has to be multiplied by CF of the rule [GR89]. However, a rule is fired only when the degree exceeds a user-defined threshold t . For example,

F_1 : color = "red" (0.9)

R_1 : IF color = "red" THEN a, (CF = 0.8), 0.2, 0

Then the result of evaluating reliability is $0.9 * 0.8 = 0.72$ and R_1 will be fired since 0.72 is larger than the threshold t , 0.2.

Action

An action represents the effect when the criterion of a rule is matched. The action of a rule should be one of following four types:

Assignment

This action is to assign value to fact or to generate a new fact. Before assigning value to a fact, the possibility of the new value is considered first, which is the result of the minimal possibilities of facts in condition expression multiplying the CF of the matched rule. The assignment is executed only if the new possibility given to assigned fact is equal to or higher than current possibility of the fact, and the possibility of assigned fact will be modified as new possibility, too. For example, if the reliability of a rule is 0.8, and its action is to assign some value to a fact whose possibility is 0.9, the action will not perform. On the other hand, if the objective is a fact whose possibility is 0.7, the Assignment action will be completed successfully.

Trigger

The conditional transferences are divided into two kinds of actions: *Trigger* and *Acquire*. In Trigger relationship, it triggers another KC with current facts as knowledge transfer. In other words, the remnant knowledge in original KC

should not be considered. During inferring, present inference process of the FC aborts, and a new inference process will start with the triggered KC.

Acquire

The second action of transference is Acquire that represents the acquirement relation. After Acquire process, the original inference process will continue and only facts predefined in the acquired KC will be carried back. At the same time, the possibility of these returned facts is multiplied by CF of the fired rule.

4.2.4 Relation

The relationships between KCs are divided into two kinds - dynamic and static. The relationships mentioned including Trigger relation and Acquire relation are dynamic because they are activated conditionally in the action part of a rule.

Two new relations, including *Reference* and *Extension-of*, will be defined as static relations. These two relations are designed according to the natural of building knowledge of human. Since a KC may refer several KCs, the topology of all KCs is a directed graph.

Reference

Reference is used to represent the associations between different concepts. Through the Reference relation, the knowledge contained in referred KC is regarded as the base knowledge and it will be taken into consideration together with the knowledge defined in the KC. On the other words, Reference can be thought as an unconditional acquire relation between KCs.

For example, as shown in Figure 4.6, suppose we learn “wild goose” via the some

features of “goose” and the property, flyable, of “swallow”. Before considering whether the present facts indicate “wild goose”, the inference process first considers whether these facts conduct the property of “flyable” under concept of “swallow” and other properties under “goose”. Thus, the initial facts could be automatically generated. Therefore, Reference relations should be declared between KC of “wild goose” to KCs of “goose” and “swallow”.

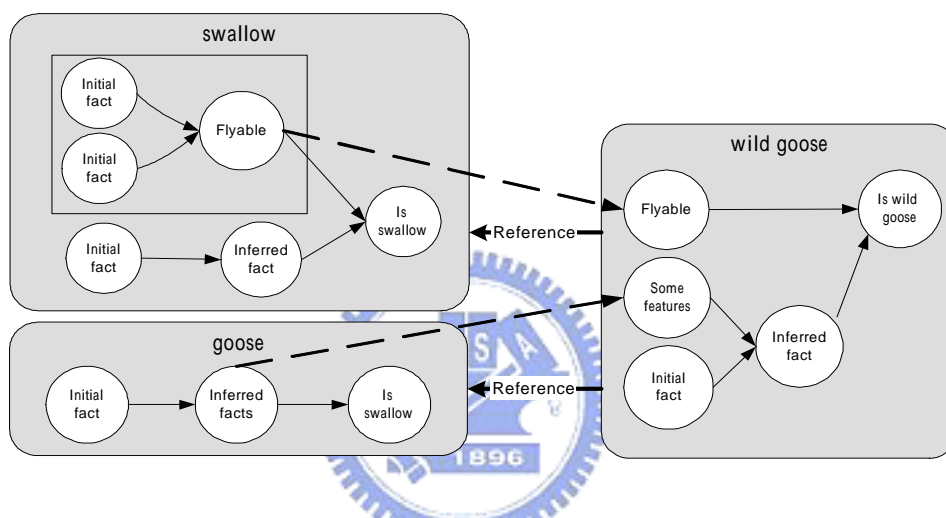


Figure 4.6: A Reference relation example

Extension-of

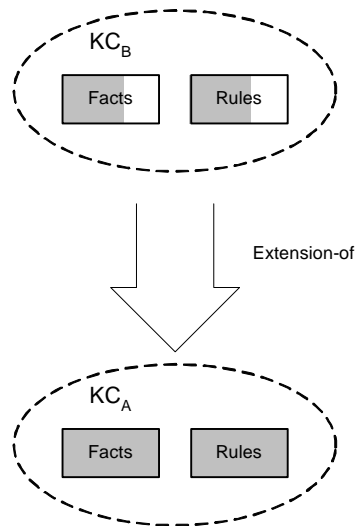


Figure 4.7: The Reference relation and the Extension-of relation

As shown in Figure 4.7, Extension-of is different from the Reference relation, a new KC may include all the knowledge contents of an existing KC through Extension-of relation. The activities of Extension-of relation include extension and modification. Therefore, it must support the overriding mechanism, including the overriding of Fact and Rule. For example, in Figure 4.7, if a knowledge class KC_B is extension of KC_A , and then KC_B will own respondent facts and required facts that KC_A owns. However, if there is a duplicate definition of fact in KC_B , the type and value of the fact will be based on the definition in KC_B . Overriding of rules in NORM is different from that of facts, which is defined as logical overriding. In logical overriding, if the rules in KC_A have the same action with KC_B , e.g., to assign value to the same fact, the action of KC_B will be taken instead of that of KC_A .

Finally, the relationships between KCs are not necessarily accurate and there may be some uncertainty of the fact declarations and the rule assertions in the relations. Relations can be asserted a certainty factor to reduce the degree of belief of default

facts and rules in the referred KCs. The detail of this process will be discussed in the next section.

4.2.5 Transformer

The transformer is used to transform the facts between two KCs, because the fact might be expressed in different measures. For example, the “temperature” may be measured in Fahrenheit or Celsius for different knowledge concepts. Therefore, the transformers may be attached to the relations between KCs.

4.2.6 Rule-base

In this model, a Rule-Base (RB) records various knowledge concepts in a specific domain and each Knowledge Class (KC) in the RB represents different concept of the domain knowledge.

In addition, RB is a unit of knowledge exchange and the meta-data of KCs supply relevant information for knowledge reuse, e.g., author, purpose, and so on.

Inferring

In cognitive structure of human [CQ69][KIN70][TUL83][TT73], there is a complex mechanism to map perceived facts to the concept of long time memory, and use the knowledge of the concept for solving problems. However, the ideal mechanism can not be easily implemented. In NORM knowledge model, a KC that contains the control knowledge, which is the knowledge about considering which kind of knowledge should be used to solve problem, must be specified before using the knowledge in NORM. For example, in the knowledge system about medical diagnosis, a KC contains the control knowledge of determining which type of

diagnosis KC, e.g. KC for Internal Medicine or Surgery, should be used.

4.2.7 Inference Process

The inference process with the model is described as follows. The first step is to select a Rule-Base. Because a knowledge system cannot contain all types of domain knowledge, specifying a knowledge domain, e.g., internal medicine diagnosis, travel planning and so on, is necessary before inferring. The second step is to collect the facts and specify a KC containing the corresponding control knowledge for the problem to be solved. According to the specified KC, the inference engine will perform the reasoning process. Finally, interesting information can be obtained from final fact value. Furthermore, the order of fired rules and causal relationship between those rules can be retrieved for explanation mechanism.

4.3 Relation-based Inference mechanism

In order to deal with the various relationships under NORM, the relation-based inference mechanism is proposed. A forward relation-based inference mechanism shown in Figure 4.8 includes following five modules.

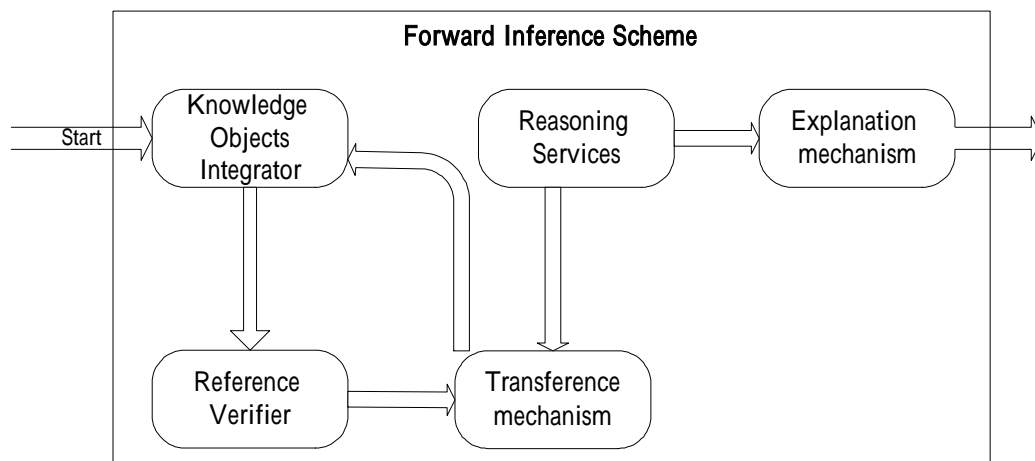
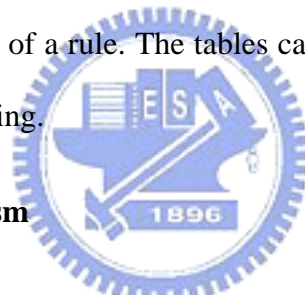


Figure 4.8: The forward relation-based inference scheme

4.3.1 Knowledge Class Integrator

This module integrates the rules and fact declarations through the Extension-of relations between KCs. Before inferring, it rewrites the action part of integrated rules and adjusts the certainty-factor value of these rules according to the Extension-of relation declaration. Similarly, it also combines the fact declarations of knowledge classes.

This module also creates the relation tables about the interaction between rules and facts, including what facts are used in condition part of a rule and what facts or KCs are affected by the action part of a rule. The tables can help to increase the efficiency of the rule matching in reasoning.



4.3.2 Transference Mechanism

This mechanism mainly performs the Trigger or Acquire during reasoning process. An FC is *KC-dependent* to a KC if if the FC is inferred with the KC. This module performs transference with changing the KC-dependence of an FC. In other words, it causes the FC to be KC-dependent to another KC, and restarts the inference process. As shown in Figure 4.9, for Trigger action, the original inference process will be terminated. Unlike Trigger, the action of Acquire copies the current FC to begin a new inference process with the target KC. After the new process, facts are returned to original FC according to the fact declarations in the target KC, and the original inference process will continue.

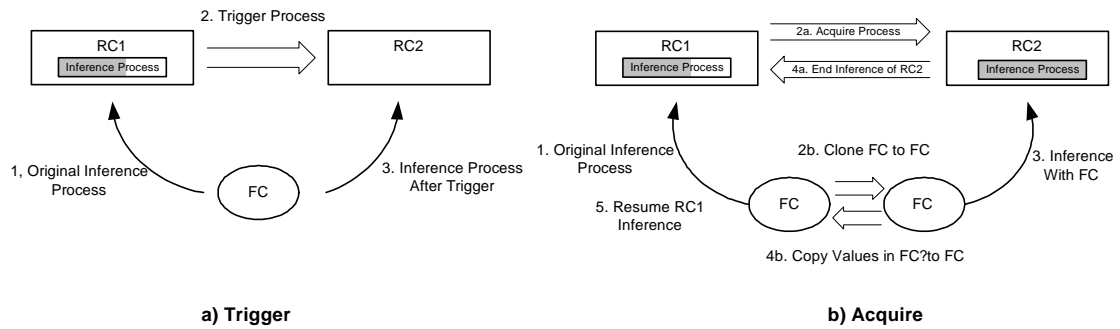


Figure 4.9: The Trigger action and Acquire action

Transformer

The transformer consisting of *TO* and *FROM* performs in transference mechanism. Before the transference, the specific facts were assigned new value according to the *TO* part of a transformer declaration. If there is a fact that owns the same name as the assertion of *Source-Fact*, the fact will be replaced or removed before the new inference process. Besides, for the *Acquire* action, the values of facts will be responded, and the facts will be changed according to operations defined in the part of *FROM*. For example, in Figure 4.10, while the action executes, the fact *F* of an *FC* is first converted into the fact *C* and then removed. After the action, the fact *C* will be transformed to *F* as well.

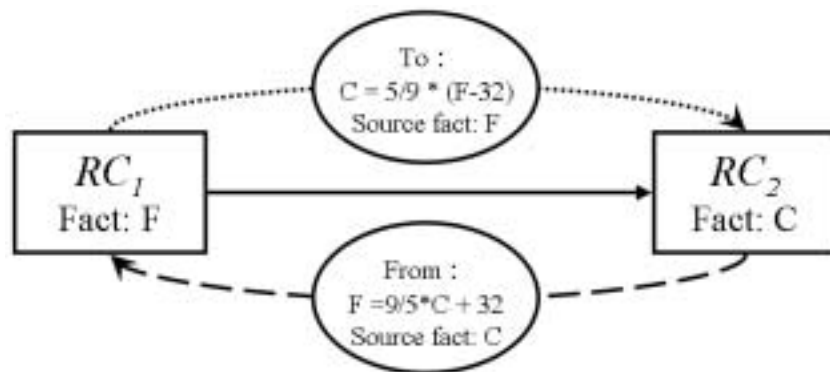


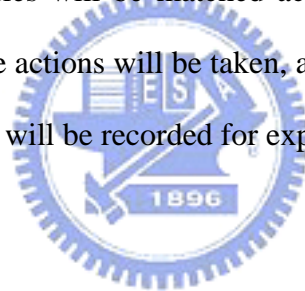
Figure 4.10: A transformer example

4.3.3 Reference Verifier

This module deals with the Reference relations between KCs. When an inference process initiated, Reference Verifier verifies the prerequisite of referenced KC and includes all the rules and facts of the KC through Reference relation. Included rules and facts will be used as a part of the knowledge to be processed during the inference process.

4.3.4 Reasoning Service

This module is used to do the actual inference process within the rules and facts from previous mechanisms. The rules will be matched according to the given facts, rule actions except the transference actions will be taken, and new fact value is assigned or generated. All the above steps will be recorded for explaining the inference process by Explanation Mechanism.



4.3.5 Explanation Mechanism

This module arranges conclusion in systematic form and provides the ability of explaining the conclusion. The conclusion is represented in three parts: the list of facts that are modified or generated during inferring, the cause relation between fired rules and facts, and the order of all fired rules. Thus, the inferring result can be explained.

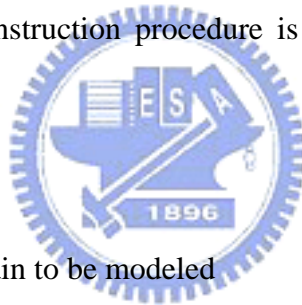
4.4 Modeling a Knowledge Base

Modeling a knowledge base [CL02][RW02] contains several processes, construction,

maintenance, reuse and refinement. In the life cycle of KBS, KB maintenance and refinement repeat recurrently. In this section, the methodologies of modeling a knowledge base under NORM will be described.

4.4.1 Construction

The first process of modeling a knowledge base is construction, i.e., transforming the domain knowledge of experts into knowledge representation format of NORM. In this section, a construction procedure is proposed to construct the knowledge systematically. In knowledge base construction, it is assumed that no prior knowledge of the similar domain exists, and Extension-of relation will not be used in construction process. The construction procedure is divided into the following six steps.



1. Select a knowledge domain to be modeled

Before designing a knowledge base, the domain of the KB must be first selected. If a large system is built, the domain of the system may be divided into several sub-domains.

2. Identify concepts in the domain and model the concepts

This phase is to analyze what concepts are contained in one domain, similar to the use-case analysis in OOA/OOD. A concept in knowledge base is used to solve a problem as use-case.

In cognitive psychologist, the knowledge can be divided into three categories: declarative, procedural and strategic [AND95][GLA87]. Declarative knowledge is

used to judge if the present facts correspond to things that the concept represents, and finally the result is obtained from the value of facts, e.g., deciding whether an entity is a bird according to the facts about its features.

Procedural knowledge contains the discrete steps or actions to be taken and the available alternatives to perform a given task. Thus a procedural concept is based on the visible facts to proceed planning for the concept, e.g., how to fix a bicycle. A plan may be generated from this kind of knowledge to solve a problem.

Strategic knowledge is used to decide course of action and regards the interrelationships and interdependencies among concepts. Strategic knowledge consists of reasoning strategy and control rules [KIN70]. In NORM, the control rules decide which KC will be used.

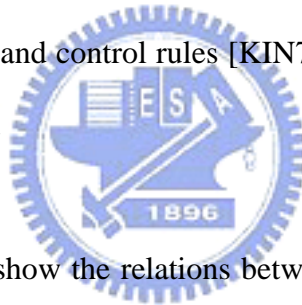


Figure 4.11 is an example to show the relations between the KCs containing one of the three types of knowledge. Procedural KC may acquire result inferring by other procedural or declarative KCs, or trigger a control KC. The Control KC can decide which KC will be used with existing facts. However, an inference process can start with any type of knowledge class.

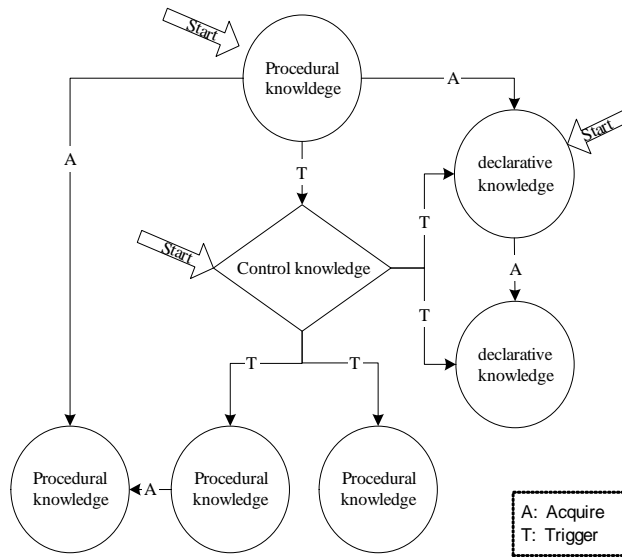


Figure 4.11: The cooperation of KCs with different types of knowledge

In this step, the type of concept to be modeled must be decided, and each concept must be mutual exclusive from each other.

3. Identify the relationships among concepts

Next, according to the exclusive relation of concepts, the type of their knowledge relation in NORM model must be found according to following basis. The concept with generalization is defined as Reference relation; the concept with causal relationship is defined as Trigger relation; at last, through further analysis, the sub problem or sub concept can be defined as Acquire relation.

4. Identify the features of each concept.

In this step, according to perception of experts, the features that affect each concept will be defined, and the facts in each concept will be used in designing corresponding KC.

Facts can be divided into two categories, respondent facts and required facts. Respondent facts possess the function of output, which means all of the relevant

features generated through inferring the basic information, can be categorized as respondent facts. On the other hand, all necessary basic information for inferring with KC is a type of required facts.

5. Design the transformer

When a KB is constructed from several KCs, the transformer may be needed between KCs to transform useful facts. A transformer should be designed if the format requirements of cognominal feature facts between two KCs are different. The transformer will be assigned to the relation between KCs except Extension-of relation.



6. Acquire knowledge of each concept

Because rules are chosen to represent knowledge of each concept in NORM, the knowledge should be transformed into rule form. According to the relations between KCs analyzed in previous steps, this step acquires the knowledge of experts about each KC. The acquisition process for one KC can rely on some developed KA methodology such as repertory grid. However, the rules dealing with Trigger and Acquire relation between KCs should be asserted.

In order to avoid redundant design of the rules, the knowledge of a KC can be acquired if the KC is the top of relationship hierarchy between KCs, i.e., it does not refer other KCs.

4.4.2 Maintenance and Reuse

There are some differences between maintenance and reuse of existing knowledge in NORM. Maintenance means the modifier is the originator of a knowledge base system, but reuse means that someone else uses existing KC and modifies it. Therefore, reusing an existing RB could be proceeded by building Extension-of relation.

Understanding an existing rule-base is the prerequisite to reuse or maintain it, which means user has to know the domain problem solved in the rule base, the concepts of KCs contained in the rule base, and the declarations of each fact in KCs. Thus, the process could be proceeded as follows.

1. Analyze the relationship of the new concept with original KCs.

In order to add a new concept to a rule-base, the relationships between new concept and original KCs must be known. In most cases, Reference is used to describe the relation between two KCs, which cooperates to solve a problem. Extension-of may be used if one KC is a modification of another KC and they have similar concept or solve the similar problem.

2. Identify the facts of the new KC.

According to the Extension-of or Reference relation, the key facts of new KC could be identified. In addition, the new concept may use features that are not declared in the referred KCs, and those feature facts should be declared in new KC.

3. Check conflict of fact definitions and design the transformers

The names of facts in two KCs should be unified. For example, if a KC use “fever” to

express a rise in the temperature of the body, the other KC shouldn't use "pyrexia" to express the same concept. However, if needed, the transformer can be designed according to type of fact value and the meaning of facts.

4. Acquire knowledge of the new concept

The step is similar to Step 6 in Section 4.1.

4.4.3 Refinement

Knowledge acquisition can be divided into two phases, initial phase and refinement phase, in which the initial knowledge base is refined to produce a high performance system [GWP88] [KIN01]. In this phase, the knowledge base should be corrected through a debug process and the relationships between KCs may be refined, e.g., the common concept of KCs can be extracted into an independent KC.



Chapter 5 Knowledge Acquisition

With the growth of the usage of information systems, more and more information and data are collected and summarized as cases, for example, the list of system vulnerabilities [CERT04], the bugs of operation system [MIC04], etc. Many of them collect the cases as a list or dictionary for user to browse, and may provide some search functions for users to retrieve the desired information. However, the knowledge behind these cases may be not well structured; it means when users try to access the information, they have to extract the information by comprehending the information.

Building a knowledge base provides more than a dictionary, it can provide the capability to process the knowledge for solving the issues raised by users; for example, an expert system, a type of knowledge system, may provide the service for users to request for solutions of a given problem, and inference the knowledge base behind to find appropriate answers.

Knowledge engineering is a process to build up knowledge system, and Knowledge Engineers (KE) play a very important role for building a successful knowledge system. One of the most important jobs for KE when building a knowledge system is to acquire knowledge from expert, which is usually so called a Knowledge Acquisition process. However, the KE is not necessary a domain expert since the major ability for a KE is to analyze and design the knowledge system systematically, and that means we can not expect a KE to have the ability to design the tools, e.g., the domain specific repertory grid, for acquiring knowledge from expert to build a successful

knowledge system.

In order to solve such problem, Concept Learning from Cases based on Semantic Distance Calculation for Knowledge Acquisition mechanism is proposed in this work. The basic idea is to provide a mechanism to extract concept information from previous work, including the list of information, the article or description about the domain information, as cases, in which the concept information is useful for building ontology to be used in knowledge acquisition process [LW02][CS99]. The information may be already filed or summarized in some information system as cases but can not be directly used as knowledge since most of them are just text information, but that doesn't mean there is no knowledge inside the information, our goal is to find useful knowledge from it as a base for KE to execute the knowledge acquisition process.



In this work, the mechanism proposed consists of Case Clustering, Concept Relationship Constructing, and Knowledge Extracting Steps. In Case Clustering step, we firstly collect all the cases of the domain for KA, e.g., the list of system vulnerabilities. And then *Knowledge Feature Clustering Algorithm (KFCA)* is proposed to group cases of text into knowledge concepts. In Concept Relationship Constructing step, the correlation between these concepts obtained in previous phase will be acquired from domain experts; hence, the desired domain ontology can be constructed. Moreover, in Knowledge Extracting step, experts will be asked to fill in the grid for each concept with their domain knowledge. The knowledge contained in these grids with their implicit meaning will be finally extracted by EMCUD[HT90] and TpKA [TT02] algorithms.

To evaluate the performance of our mechanism in building up the domain concept, an experiment based on categorized intrusions information has been done. In the experiment, the category information is used to evaluate the accuracy of the clustering result.

In this work, a mechanism for Concept Learning from Cases is proposed to construct the concepts of ontology; moreover, the process for acquiring concept relations and extracting the knowledge of concepts is also designed. In the entire flow, a Case Clustering Step is first applied to assist knowledge engineers to find concepts from cases of text information. Based on the discovered concepts, the knowledge engineer may design the knowledge acquisition mechanism to retrieve the required concept relations in Concept Relationship Constructing Step; and in Knowledge Extracting Step, the concept knowledge is also acquired from expert by repertory grid approach. Figure 1 shows two phases in our KA methodology:



Concept Learning Process

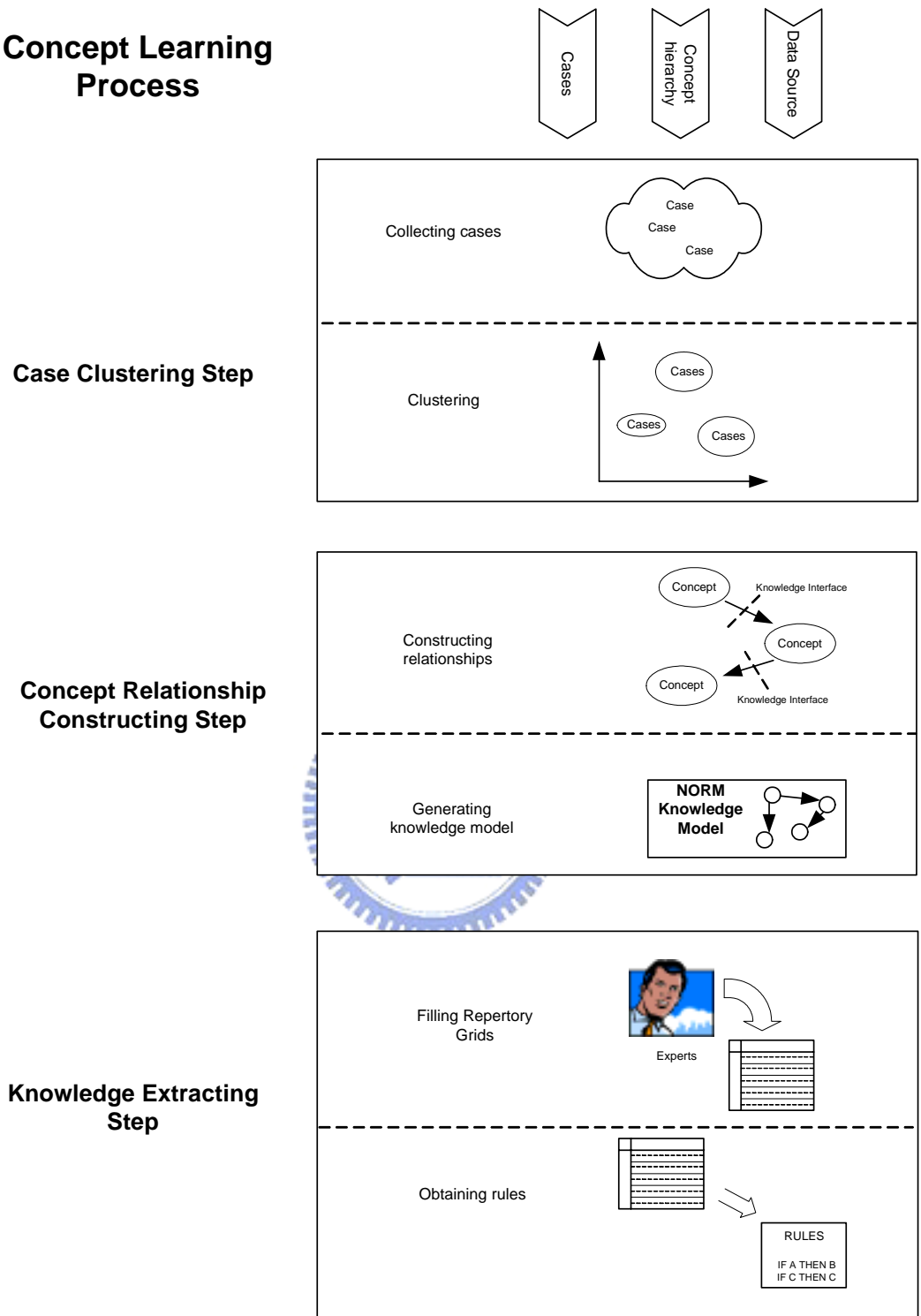
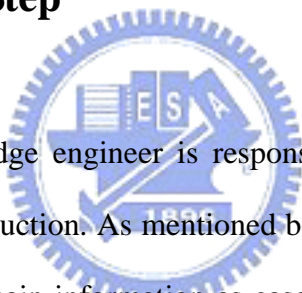


Figure 5.1: The phases of Concept Learning from Cased based on Semantic Distance Calculation

In Case Clustering Step, the domain cases collected will be clustered into clusters

according to the semantic similarities of cases. In Concept Relationship Constructing Step, the concept meaning of the clusters generated in previous step will be identified, and also the relationships between concepts will be acquired from expert by filling a relationship table; NORM(New Object-oriented Rule Model) [LT03] is used as the representation for concept relations and concepts. The knowledge about each concept obtained will be acquired using repertory grid approach in Knowledge Extracting Step, in which EMCUD [HT90] and TpKA [TT02] approaches are used to extract knowledge together with the implicit meaning of them.

5.1 Case Clustering Step

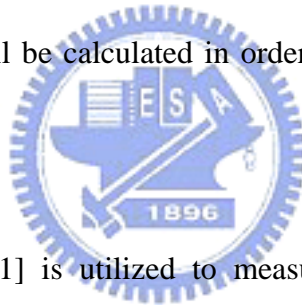


In this step, the knowledge engineer is responsible for collecting the domain information for concept construction. As mentioned before, many information system or web site provide some domain information as cases, for example, CERT provides and profiles the system vulnerabilities information, Book review and description, Bug Lists of software, etc. However, the list can be huge and hence hard to be used even search function provided. A knowledge system to help user retrieve the information (e.g., help to find appropriate books from user requirement) or take use of the implicit knowledge (e.g., diagnosis system vulnerability) will be helpful to improve the usage of such information.

A Knowledge Feature Clustering Algorithm (*KFCA*) is then proposed to cluster the knowledge, and also help construct the knowledge concepts in these cases. *KFCA* works along with the concept hierarchy information representing the relationships

between vocabularies, for example, e.g., WordNet [WOR03]. *KFCA* will use the concept hierarchy to calculate the similarity between cases. Once the similarity between cases is determined, each cluster consisting of similar cases can be obtained by KFCA.

In Knowledge Collection stage, the cases and corresponding descriptions are collected. In order to calculate the semantic similarity between these cases, the keywords in the descriptions for cases will be first extracted as features of the case using the concept hierarchy for calculating semantic relatedness, since only the keywords contained in the dictionary is useful for semantic calculation. After this process, the keywords used to represent the features for cases will be extracted. And then the semantic distances between different features will be calculated in order to get the similarities between cases.



The semantic distance [BH01] is utilized to measure the semantic heterogeneity between keywords. For two vocabularies v, v' , the semantic distance is given by $S_v(v, v') = \text{“the number of links from } v \text{ to } v' \text{ in the shared vocabulary concept hierarchy”} + 0.5 * \text{“the number of changes of directions”}$.

Definition 1: Semantic Distance of two vocabularies

The semantic distance function we use is based on the Hirst and St-Onge’s measure of semantic relatedness [HS98], and is defined as follows:

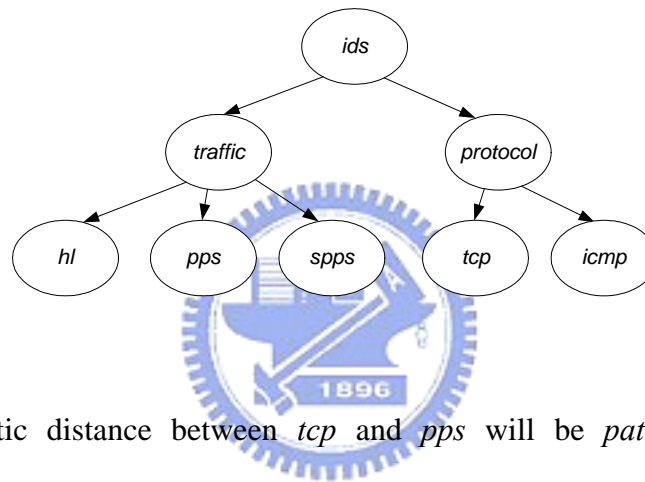
$$D_v(v_1, v_2) = path_length + c * direction_change, \quad \forall v_1, v_2 \in V,$$

where *path_length* is the length from v_1 to v_2 in the shared vocabulary ontology,

direction_change is the number of changes of direction in the path, and *c* is constant, which is set as 0.5 in this work. If the path does not exist, the function returns “infinity”. $D_v(v_1, v_2) = 0$ if and only if $v_1=v_2$.

Example 1: : Semantic Relatedness Calculation

Given a concept hierarchy for network intrusion related vocabularies as following:



then the semantic distance between *tcp* and *pps* will be $path_length + 0.5 * direction_change = 4 + 0.5*1 = 4.5$.

Heuristic 1: Similarity of vocabularies calculated from semantic distance

The similarity of vocabularies will be defined base on the semantic distance calculation, and two assumptions are given here: First, the similarity between two vocabularies decreases in half when the distance increase by 1. Second, the similarity value should be limited between 0 to 1. Hence, the similarity function is defined as following formula:

$$Sv(x, y) = 2^{-D_v(x,y)}$$

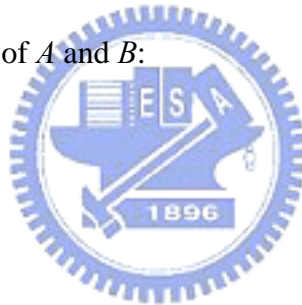
Also, since the distance between two vocabularies without connected path is defined as “infinity”, the similarity used here can prevent the *infinity* value dominate the result, which can occur when using distance instead of similarity.

And hence we would like to define the similarity function of cases as following definition.

Definition 2:Case Similarity

For two case A and B , assume the keywords extracted to represent $A=\{a_1, a_2, \dots, a_m\}$ and $B=\{b_1, b_2, \dots, b_n\}$, the Case Similarity KS is the average similarity between all pairs of keywords of A and B :

$$KS_{A,B} = \frac{\sum_{x=1}^m \sum_{y=1}^n S_v(x, y)}{m \times n}$$



Example 2: : Case Similarity Calculation

Assume there are three cases which describe three different kinds of network intrusions.

IIS Memory Leakage: A Flaw that may allow a malicious user to consume all available memory by sending lots of HTTP request to cause heavy load (hl).

ICMP flood: By sending lots of unreal ICMP packet, the victim host will get heavy load (hl) and busy in responding the ICMP request.

Ping of death: The machine crashed when doing ICMP echo pinging with corrupted fragmented packet.

With keyword extraction mechanism, we may get the keywords for these cases as $A =$ IIS Memory Leakage = $\{HTTP, hl\}$, $B =$ ICMP flood = $\{ICMP, hl\}$, and $C =$ Ping of death = $\{ICMP, corrupted\}$. And hence the similarities between these cases can be calculated according to Definition 3.2:

$$\begin{aligned}
 KS_{A,B} &= \frac{S_v(HTTP, ICMP) + S_v(HTTP, hl) + S_v(hl, ICMP) + S_v(hl, hl)}{m \times n} \\
 &= \frac{2^{-2.5} + 2^{-4.5} + 2^{-4.5} + 2^{-0}}{2 \times 2} \\
 &= 0.3163
 \end{aligned}$$

$$\begin{aligned}
 KS_{A,C} &= \frac{S_v(HTTP, ICMP) + S_v(HTTP, corrupted) + S_v(hl, ICMP) + S_v(hl, corrupted)}{m \times n} \\
 &= \frac{2^{-2.5} + 2^{-\infty} + 2^{-4.5} + 2^{-\infty}}{2 \times 2} \\
 &= 0.055
 \end{aligned}$$

Definition 3: The Similarity Matrix for cases

Since the case similarity calculation is repetitively resource consuming during clustering, the Similarity Matrix, which contains the similarity between all pairs of cases, is calculated first. Assume we have n cases to cluster, t_1 to t_n , the Similarity Matrix SM will be:

$$SM = \begin{bmatrix} KS_{t_1,t_1} & KS_{t_1,t_2} & \cdots & KS_{t_1,t_n} \\ & KS_{t_2,t_2} & \cdots & KS_{t_2,t_n} \\ & & \ddots & \vdots \\ & & & KS_{t_n,t_n} \end{bmatrix}$$

Since the distance between two keywords in the concept hierarchy is symmetric, SM is represented as an upper triangular matrix.

In this work, the data to be clustered is non-numeric so the cluster center can not be calculated as the numeric center. Hence, we propose a heuristic here to provide a fast and simple approach to calculate the center of a cluster.

Heuristic 2: For each case in a cluster, the sum of similarity values from it to all the other cases will be calculated, and the cluster center will be the case with maximum sum of similarity values.



Definition 4: The standard deviation of case cluster

The following formula is used to calculate the standard deviation of a cluster of cases.

$$SD_{C_i} = \sqrt{\frac{\sum_{t \in C_i} \left(\frac{1}{KS_{t,center-of-C_i}} \right)^2}{|C_i|}}$$

Based on the definitions and heuristics, the clustering algorithm we used here to group cases is given as follows:

Algorithm 5.1. Knowledge Feature Clustering Algorithm

Input: Cases to be clustered.

Output: Case clusters

Step 1. Calculate the Similarity Matrix of all cases.

Step 2. Select k as 2. Randomly find k cases as the initial cluster centers, cluster each case to the nearest cluster center.

Step 3. Find the new cluster center approximately by:

Step 3.1. For cluster C_i , $i = 1$ to k ,

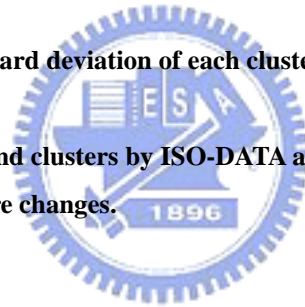
Step 3.2. Select sub-matrix, only rows and columns of the cases included in C_i from Similarity Matrix.

Step 3.3. Sum the value of each row, and select the case with the largest sum as the cluster center.

Step 3.4. Calculate the standard deviation of each cluster.

Step 4. Refine Cluster number and clusters by ISO-DATA approach.

Step 5. Go to Step 3 until no more changes.



Example 3: As in Example 2, the Similarity Matrix of those three cases is shown below:

$$\begin{array}{c} A \\ B \\ C \end{array} \begin{array}{ccc} A & B & C \\ \left[\begin{array}{ccc} 1 & 0.3163 & 0.055 \\ X & 1 & 0.261 \\ X & X & 1 \end{array} \right] \end{array}$$

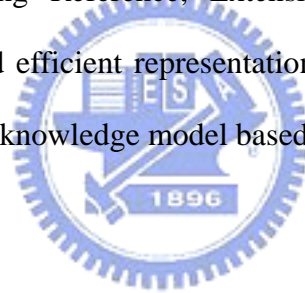
In this example, two clusters $C_1 \{ A, B \}$ and $C_2 \{ C \}$ can be easily obtained.

All cases selected by knowledge engineers to represent the domain will then be clustered into groups with similar cases, and each group corresponds to a specific

knowledge concept.

5.2 Concept Relationship Constructing Step

In order to construct the complete ontology of the knowledge domain, the knowledge clusters we extracted in previous stage must be connected with meaningful knowledge relations. In this stage, the relations between knowledge concepts will be extracted. To represent the knowledge model built in this stage, a New Object-oriented Rule Model (NORM) [LT03] is used. In NORM, knowledge relations can be constructed, including Reference, Extension, Trigger, Acquire. NORM provides more systematic and efficient representation for domain knowledge; hence experts are asked to construct knowledge model based on these relationships.



Definition 5: Relation Table

The Relation Table defines the relationship between knowledge concepts. Assume that we have n knowledge concepts, and the Relation Table for these knowledge concepts may look like:

	Concept 1	Concept 2		Concept n
Concept 1	X		<i>Reference</i>	
Concept 2	<i>Trigger</i>	X		<i>Extension</i>
			X	
Concept n		<i>Acquire</i>		X

For slot (Concept i , Concept j) in this table, the value will be the NORM relationship for Concept i to Concept j . Before acquiring the relationships between knowledge concepts, the cases in each knowledge concept must be reviewed and

redundancies in the concepts must be resolved. Since then, the knowledge engineers should ask the domain expert to design the relations between knowledge concepts. The procedure to be taken for design the relationships is described as follows:

Algorithm 5.2. Knowledge Map Design Process

Input: The case clusters come from previous stage.

Output: Relation Table between k knowledge concepts, and the meta knowledge of each relation.

Step 1. Resolve or eliminate the redundancies within knowledge concepts, and identify the meaning of each knowledge concept.

Step 1.1. Check each knowledge concept, and eliminate redundant cases.

Step 1.2. Explain the meaning of each cluster, and name the clusters with corresponding concept.

Step 2. Define the interface cases for each knowledge concept:

Step 2.1 Construct an empty relationship table.

Step 2.2 Fill in the knowledge relations according to NORM knowledge relations by experts.

Step 2.3 Ask experts to design the meta-rules to link and interact between knowledge concepts for each relation between knowledge concepts.

Step 3. Construct the ontology of knowledge concepts.

The procedure in this stage can help construct the knowledge concept relationships; in other words, the relations between knowledge concepts construct the ontology of the domain. As we have mentioned, NORM is used here to represent the ontology, hence we will use the Rule Class in NORM to represent concepts, and use those four kinds of relationships to build up the concept ontology.

5.3 Knowledge Extracting Step

So far, not only the ontology between knowledge concepts but also the cases of each knowledge concept are defined. In this stage, the knowledge engineers can design the grid for extracting knowledge from experts, and once the grid for extracting knowledge is designed, experts will be asked to fill in its appropriate values. The column header of the grid is the cases to be identified in a concept, and the row header of the grid is the union of keywords (features) of cases. For example, a grid for extracting knowledge about some different types of intrusions may be like Table 1.

Table 1. An example grid used for knowledge acquisition

	Ping of Death	ICMP flood	IIS memory leakage
ICMP	YES	YES	NO
TCP	NO	NO	YES
...		...	
Heavy Load	NO	YES	YES
Corrupted Packet	YES	NO	NO
Crashed	YES	NO	YES

From the filled grid, rule as the knowledge can be obtained. For example, one of the rules generated from above grids is shown as:

IF “ICMP” and “Corrupted Packet” and “Crashed”
then “Ping of Death”


For extracting the rules with embedded meaning, Embedded Meaning Capturing and Uncertainty Deciding (EMCUD) knowledge acquisition [HT90] based on Personal Construct Theory is used in this stage. Since ontology is discovered in previous phase,

the information about the relation and hierarchy between knowledge concepts is included in our knowledge extraction stage. Hence, Two Phase Knowledge Acquisition (TpKA) [TT02] mechanism is used to extract the knowledge with given concept relations and to find more meaningful and accurate knowledge content. With TpKA, the embedded meaning and certainty factor of knowledge will be reviewed according to the knowledge hierarchy built in previous phase.



Chapter 6 Knowledge Discovery

Rule base system is usually used in designing a knowledge based system, which is used to provide suggestions on decision making as a domain expert. However, since the knowledge in a rule base is usually acquire from one or few experts, that means there are many cases that the knowledge is generated according to their own experience, and some knowledge may be not included due to lack of experience. In order to make the rule base system to be more complete and smart, the knowledge of general users should also be discovered and used to refine the rule base system in knowledge systems.



In modern computer systems, user activities are usually recorded by system log information, which means there is some information regarding the user behaviors hidden in the log information. In our knowledge discovery mechanism, the log information of computer systems will be used to find the pattern of the user behavior, which can be the user knowledge for system operating, problem solving.

The input format of our method is the user activities records or logs sorted by the time. As shown in Figure 6.1, there are several phases in our method including Preprocessing Phase, Two-Layer Pattern Discovering Phase, and Pattern Explanation Phase. At first, the Preprocessing Phase could select activities logs stored in the data storage and aggregated these activities logs into a feature vector, which represents the behavior during a short period for further analysis. Furthermore, each user's behavior can be presented as a sequence of feature vectors. In Two-Layer Pattern Discovering Phase, there may be millions of distinct feature vectors, which will be first clustered

into several clusters. In this phase, two heuristics are proposed to detect outliers, which are quite different from normal behaviors, and these outlier clusters can be explained in Pattern Explanation Phase. Accordingly, some feature vectors which are similar in representing the same behavior may be grouped into one cluster. In other words, each feature vector can be mapped to a cluster label by a mapping function, and each user's behavior can be transformed into a sequence of cluster labels. Next, we are also concerned about patterns of single user's behaviors and common patterns of all users' behaviors to mine the patterns of users' behaviors. Since each pattern is represented as a sequence of clusters and each cluster has its own property set, the pattern discovered in previous phase can be represented as a sequence of property sets, can be determined to be normal or abnormal, and can be feedbacked into knowledge base in Pattern Explanation Phase.

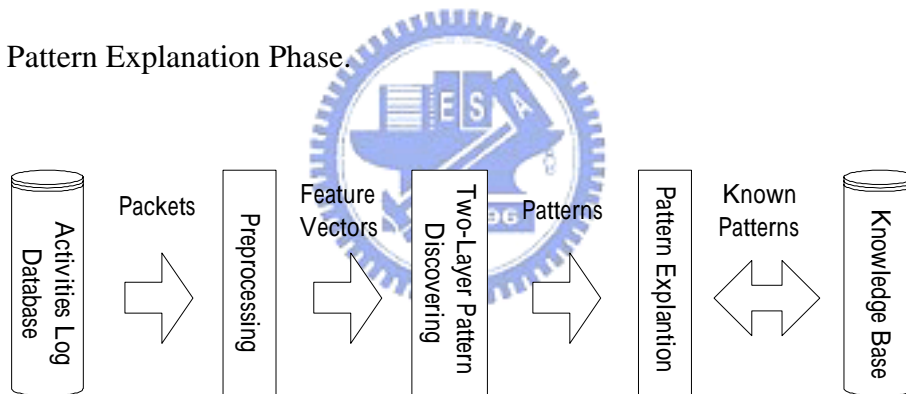


Figure 6.1: The Concept Diagram of Our Method.

6.1 Preprocessing Phase

Before presenting our method, the notations used in this paper will be defined in this section. For transforming original activities logs into a feature vector, which contains more useful information, the RENUMBER SORT ALGORITHM and the PREPROCESS ALGORITHM are also discussed in this section.

6.1.1 Definitions of Original Activity Log Database

Assume there are n users u_1, u_2, \dots, u_n . Each u_q can be represented by a unique ID, e.g., the user id of a web system or the customer ID of a shopping center, and let $U = \{u_1, u_2, \dots, u_n\}$.

$T = [t_0, t_0+wc]$ is the time interval concerned to collect activities logs where c is a constant, $t_0 = 0$, $t_i = t_{i-1} + c$, and $T^i = (t_{i-1}, t_i]$, $1 \leq i \leq w$.

$E^i = \langle e_1^i, e_2^i, \dots, e_{\alpha_i}^i \rangle$ is a sorted sequence of activities logs in time order during T^i and we assume $|E^i| = \alpha_i \leq \alpha$, for each i .

'•' is a concatenation operator, i.e., $E^1 \bullet E^2 = \langle e_1^1, e_2^1, \dots, e_{\alpha_1}^1, e_1^2, e_2^2, \dots, e_{\alpha_2}^2 \rangle$.

$E = E^1 \bullet E^2 \bullet \dots \bullet E^w$ is the whole activities logs we are concerned in T .

e -*id* is the event identifier which is defined by the triple fields $\langle \text{unique ID}, \text{action target}, \text{action} \rangle$, where $\text{unique ID} \in U$, and action target is the target of the user activities, e.g., the item to sale, and the action is the action taken by the user, e.g., POST, GET.

$ID(e_j^i)$ is an extracting function to extract the e -*id* of e_j^i .

6.1.2 Renumber Sort Algorithm

Since the information of single activities is not sufficient enough to represent the user behavior, several activities with same e -*id* selected from E^i are first aggregated during T^i and then transformed into a feature vector. However, when we do aggregation in this phase, if a User A do **GET** action to web page X , and the other action user A taken is **POST** web page Y , these two actions cannot be treated as the same event.

Notations:

$f_j^i = \mathbf{ReNumSort}(E^i)$ is the j th distinct e - id during T^i .

$S^i = \langle f_1^i, f_2^i, \dots, f_{\beta_i}^i \rangle$ is a sequence of feature vectors during T^i , where $\beta_i \leq \alpha_i$.

$F^i = \{ f_j^i \mid \text{for } 1 \leq j \leq \beta_i \}$, and $F = \bigcup_{i=1}^w F^i$.

S_q^i is a subsequence of S^i for $q \in U$.

$v_q = \langle S_q^1, S_q^2, \dots, S_q^w \rangle$ is a behavior vector of u_q .

$V = \{v_q \mid \text{for } q \in U\}$.

Table 4.1 presents the format of general activities log. The *Time* field indicates the occurred time of log. The *UID* field and *TARGET* field indicate unique ID for each user and the target item performing actions, respectively. The *ACTION* field indicates the action taken in the activity, for example, in network traffic, the ACTION may be the destination port, which implies the service has been requested by the user, e.g., FTP port is 21, Telnet port is 23, and HTTP port is 80. The information may contains in the activities log is different from applications to applications, for example, for consuming activities, maybe the sale amount, quantity will also be included, and the information can be also used in our algorithm.

Table 6.1: The Format of Standard Log Information.

Time	UID	TARGET	ACTION
------	-----	--------	--------	-----	-----	-----	-----	-----

In aggregating the activities into feature vector, we first sort the original activities database by RENUMBER SORT ALGORITHM to get the distinct e - id during T^i ,

saying f_j^i . For each activity during T^i , if there exists a previously defined feature vector entry is equal to the $e-id$ of the activity then replace it by aggregating the information for the same $e-id$. Otherwise, create and define a new feature vector entry. The RENUMBER SORT ALGORITHM is shown as follows.

Algorithm 6.1: ReNumberSort algorithm, ReNumSort(E^i)

Input: E^i

Output: F^i, S^i

Step1. $F^i = \phi, S^i = \langle \rangle, DistinctFlag = True, \beta_i = 0$.

Step2. For $j = 1$ to α_i ,

Step2.1. If $DistinctFlag = True$,

$\beta_i ++$,

$f_{\beta_i}^i = ID(e_j^i)$, Set $DistinctFlag = False$.

Step2.2. For $k = 1$ to β_i ,

If $ID(e_j^i) \neq f_k^i$, Set $DistinctFlag = True$.

Else

Replace f_k^i by merging e_j^i and f_k^i , Set $DistinctFlag =$

$False$,

EXIT.

Step3. For $j = 1$ to β_i ,

Put f_j^i into $F^i, S^i = S^i \bullet f_j^i$.

Step4. Return F^i, S^i

In Step2.2, the aggregation process to construct feature vector is specified by domain expert, which is designed based on the application and information we have in the activities log. For example, if the activities log we are mining is the consuming log of customers of a shop, we may aggregate the price user spent on the target item, the quantity of the items, and also the other information can be aggregated. The way to aggregate the information can be decided according to the knowledge of the domain expert who design the mining process.

6.1.3 Preprocessing Phase

As defined above, the feature vector is aggregated from the selected activities with same *e-id* during T^i , so the feature vector is also identified by the *e-id*. The feature vector f_j^i can be treated as a user behavior event, which represents the user's behavior during T^i . Therefore, the behavior of the user u_q during T can be represented by a sequence of feature vectors with time order.

In Table 6.1, the Time field indicates the starting time of the aggregated feature vector f_j^i . The *Duration* field indicates the interval between first and last activities with f_j^i during T^i . The *UID*, *TARGET*, *ACTION* fields are with the same definition in activity log information, and all the other fields are aggregated from **ReNumberSort** algorithm, which are the important information to represent the behavior user taken, for example, the count of the activities, the cost user spent, the quantity user taken, etc, any useful information which can also be calculated by aggregation algorithm are included.

As shown in Figure 6.2, the preprocessing phase has two major stages: the first stage is to select the packets from activities log database during time window T^i and second stage is to calculate the feature vectors F^i during T^i by aggregating the activities with f_j^i , for $1 \leq j \leq \beta$. Thus, we can have the sequence of feature vectors S^i and each user's behavior during T^i . Therefore, each user's behaviors during T can be represented as $v_q = \langle S_q^1, S_q^2, \dots, S_q^w \rangle$, for each $q \in U$.

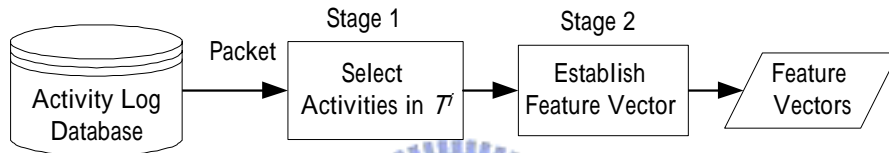


Figure 6.2: Data Flow of Preprocessing Phase.

Algorithm 6.2: Preprocessing Algorithm, Preprocess(E)

Input: E

Output: F, V

Step1. $F = \phi, V = \phi, v_q = \langle \rangle$.

Step2. For $i = 1$ to w ,

Select E^i from E ,

$(F^i, S^i) = \text{ReNumSort}(E^i)$,

$F = F \cup F^i, v_q = v_q \bullet S_q^i$.

Step3. For $q = 1$ to n ,

$V = V \cup \{v_q\}$.

6.2 Two-Layer Pattern Discovering Phase (2LPD)

In this section, the concept of Two-Layer Pattern Discovering Phase (2LPD) to discover unknown patterns will be first introduced. The related notations and algorithms of this phase will be next introduced.

6.2.1 Concept of 2LPD Phase

After the preprocessing phase, the original activities logs are already transformed into feature vectors F , and the user behaviors have already been represented by V . All of them will be treated as input in our 2LPD Phase including Behaviors Clustering Stage and Sequential Pattern Mining Stage to provide three detection strategies. Without loss of generality, we assume there are at most m clusters. The concept in Behaviors Clustering Stage is to group similar feature vectors into a cluster. And then user's behavior can be represented as a sequence of cluster labels in User's Sequence Transforming Stage. Therefore, the sequential patterns which are hidden in these users' patterns can be mined in Sequential Pattern Mining Stage. The concept of Two-Layer Pattern Discovering Phase is shown in Figure 6.3.

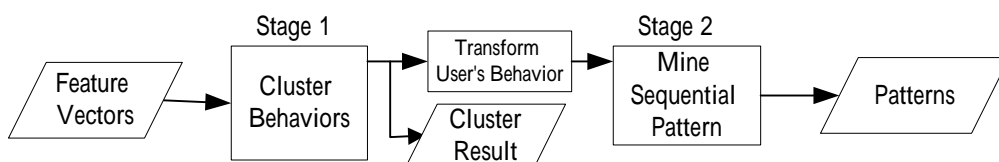


Figure 6.3: The Concept of 2LPD Phase

Notations:

$C = \{C_1, C_2, \dots, C_m\}$ is a set of clusters where C_i is a subset of F and $1 \leq i \leq m$.

OC , a subset of C , indicates the outlier cluster set.

$SEL_q(C_i)$ is a selecting function to select the feature vectors of u_q from C_i .

$M(f_j^i) = C_k$ if $f_j^i \in C_k$.

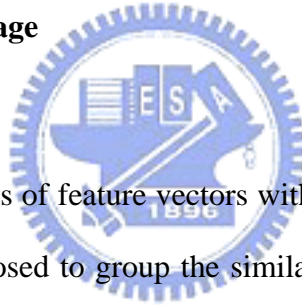
$M(S^i) = \langle M(f_1^i), M(f_2^i), \dots, M(f_{\beta_i}^i) \rangle$.

$M(S_q^i)$ is a subsequence of $M(S^i)$ for $q \in U$.

$M(v_q) = \langle M(S_q^1), M(S_q^2), \dots, M(S_q^w) \rangle$ is a sequence of cluster labels of u_q during T .

$M(V) = \{M(v_q) \mid \text{for } q \in U\}$.

6.2.2 Behavior Clustering Stage



As we know, there are millions of feature vectors with different values. The Behavior Clustering Stage is then proposed to group the similar feature vectors for the further mining. Since the number of clusters cannot be predicted in advance, a clustering algorithm with the capability of dynamic adjusting the number of clusters is used, e.g., ISODATA.

In general, the special patterns which happened not frequently and performed only by few users are usually treated as outlier and are interesting in data mining field; two heuristics of outlier clusters are proposed as follows:

Heuristic 3: A cluster is treated as an outlier cluster if the number of its members is smaller than a threshold θ_l .

Heuristic 4: A cluster is treated as an outlier cluster if the ratio of $|SEL_q(C_i)|$ and

$|C_k|$ is greater than a threshold θ_2 .

Since the system is starting with no priori knowledge about intrusion, thresholds θ_1 , θ_2 are set loose; e.g., $\theta_1 = \alpha w/m$, $\theta_2 = 0.5$. θ_1 will gradually decrease and θ_2 will gradually increase according to the patterns discovered in knowledge base.

After the execution of this phase, there may exist some outlier clusters containing information about outlier behaviors or outlier users. All of these discovered outlier clusters can be further analyzed in following phase.

Algorithm 6.3: Behavior Clustering Algorithm:

Input: F, k, θ_1, θ_2

Output: C, OC



Step1. Randomly choose k initial seeds as cluster centers.

Step2. Run ISODATA clustering algorithm to generate a number of cluster $C = \{C_1, C_2, \dots, C_m\}$.

Step3. For $i = 1$ to m ,

If $|C_i| \leq \theta_1$, put C_i into OC .

If $|SEL_q(C_i)|/|C_i| \geq \theta_2$, put C_i into OC .

Step4. Return (C, OC) .

6.2.3 User's Sequence Transforming Stage

As mentioned above, each user's behavior during T can be represented as a sequence

of features vectors $\mathbf{v}_q = \langle S_q^1, S_q^2, \dots, S_q^w \rangle$. Moreover, these feature vectors are grouped into several groups and each feature vector belongs to a unique cluster. Therefore, each user's behaviors can be transformed into a sequence of cluster labels $\mathbf{M}(\mathbf{v}_q) = \langle \mathbf{M}(S_q^1), \mathbf{M}(S_q^2), \dots, \mathbf{M}(S_q^w) \rangle$.

6.2.4 Sequential Pattern Mining Stage

Since all user patterns are concerned in this mining algorithm, not only the user patterns happened in general users, but also the subsequence of each individual user should be also mined and discovered. Therefore, all the patterns of embedded users' behaviors will be mined in Sequential Pattern Mining Stage. As each user has a sequence $\mathbf{M}(\mathbf{v}_q)$, a symbolic sequential mining algorithm, e.g., Agrawal and Strikant's mining algorithm [AS95] will be used to mine patterns from all users' sequence of behaviors.

Algorithm 6.4: Sequential Pattern Mining Algorithm:

Input: $\mathbf{M}(V), sup_1, conf_1, sup_2, conf_2, d$

Output: The subsequences of single user's behaviors and the subsequences of all users' behaviors

Step1. For $q = 1$ to n ,

According to $\mathbf{M}(\mathbf{v}_q)$, generate $\langle \mathbf{M}(S_q^1), \mathbf{M}(S_q^2), \dots, \mathbf{M}(S_q^d) \rangle, \langle \mathbf{M}(S_q^2),$

$\mathbf{M}(S_q^3), \dots, \mathbf{M}(S_q^{d+1}) \rangle, \dots, \langle \mathbf{M}(S_q^{w-d+1}), \mathbf{M}(S_q^{w-d+2}), \dots, \mathbf{M}(S_q^w) \rangle,$

Run Agrawal and Strikant's Mining Algorithm to obtain the subsequence of single user's behavior with $(M(v_q), sup_1, conf_1)$.

Step2. Run Agrawal and Strikant's Mining Algorithm to obtain the common subsequences of all users' behaviors with $(M(V), sup_2, conf_2)$.

Step3. Return the subsequences of single user's behaviors and the subsequences of all users' behaviors.

The window size d defined by domain expert for difference objects is suggested to be 150 to tradeoff between the accuracy of user patterns and efficiency of the algorithm. d is set to be large if we address long terms sequence of a user's behavior. Otherwise, we chose a small d .

Algorithm 6.5: Algorithm of Two-Layer Pattern Discovering: $2LPD(F, M(V), k, \theta_1, \theta_2, sup_1, conf_1, sup_2, conf_2, d)$

Input: $F, M(V), k, \theta_1, \theta_2, sup_1, conf_1, sup_2, conf_2, d$

Output: C, OC , and the subsequences of users' behaviors

Step1. $M(V) = \phi$.

Step2. $(C, OC) = \text{Cluster}(F, k, \theta_1, \theta_2)$.

Step3. For $q = 1$ to n ,

Transform v_q into $M(v_q)$,

Put $M(v_q)$ into $M(V)$.

Step4. Obtain all subsequences = $SEQUENTIAL(M(V), sup_1, conf_1, sup_2, conf_2, d)$.

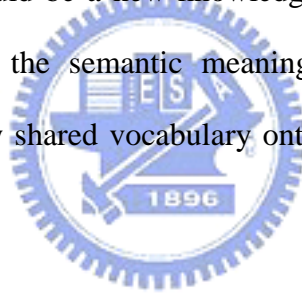
6.3 Pattern Explanation Phase

The goal of the Pattern Explanation Phase is to explain the meaning of the discovered pattern about user behavior, and transform the patterns into meaningful knowledge content for knowledge system. Since the heuristic used in behavior cluster is to cluster the similar behaviors, each cluster may have some properties, which can be extracted by analyzing the feature vector space related to each dimension. Using the property of standard derivation evaluation, the most significant attributes of the cluster can be obtained. For example, if the standard derivation value of an attribute of all feature vectors in a cluster is relatively small, the attribute is treated as a significant attribute and can be used as a property of the cluster. On the other hand, the value of this attribute with large standard derivation value will not be used since the distribution of attribute values may be sparse. Therefore, each cluster may be represented as a set of properties and domain expert can explain the meaning of the pattern. These discovered patterns can then be incrementally feedbacked to knowledge base. Hence, with this incremental learning and feedback mechanism the well-known patterns in knowledge base can be increased.

Chapter 7 Knowledge Fusion

In this paper, we try to provide structural and semantic knowledge fusion, represented by rules, using ontologies by hybrid approach. We can thus define our problem in more detail: Given a shared vocabulary ontology and a set of rule-based knowledge bases from different systems with different ontologies, the goal is to fuse all the knowledge bases to one with new ontology and to optimize the structural and the semantic meanings of the fused knowledge base.

Our goals are as follows: (1) Fuse multiple rule-based knowledge bases. The output of our proposed approaches should be a new knowledge base and a new ontology. (2) Optimize the structural and the semantic meanings of knowledge contained in knowledge base. (3) Use only shared vocabulary ontology for facilitating the fusion process.



There will be several criteria defined in the knowledge fusion mechanism proposed, including Structural Succinctness Criterion, Intra-Cluster Semantic Clustering Criterion, and Inter-cluster Semantic Clustering Criterion. For improving the rule structure to reduce the dependency between knowledge classes, the Structural Succinctness Criterion can be used. In order to group more related rules into one knowledge class, the Intra-Semantic Clustering Criterion calculates the semantic relations between rules in a knowledge classes for the fusion process to optimize. For better knowledge classes partitioning, the semantic distances between knowledge classes, which is so called Inter-cluster Semantic Clustering Criterion, will be calculated in the knowledge fusion process.

7.1 Relationship Graph and Partitioning Criteria

In this section, the intermediate knowledge representation relationship graph is introduced. The criteria to partition the relationship graph are also discussed in detail.

7.1.1 Definitions

We propose a representation, relationship graph, for expressing the structural and the semantic meanings of first-order logical rule bases. Before describing the representation, we firstly give some basic definitions, partly borrowed from the syntax of first-order logic [RN95][SOW00]. Assume that a first-order logical rule base contains n variables and m rules which are classified into t rule classes (partitions), where a rule class is a set of rules in the rule base which can be grouped by a given concept.

- . $TRUE$ = the logical constant representing “always true”.
- . $FALSE$ = the logical constant representing “always false”.
- . $EMPTY$ = the logical constant representing “empty”.
- . $V = \{v_1, v_2, \dots, v_n\}$ is the set of all first-order logical variables in the rule base, where v_i is the first-order logical variable of the rule base, where $1 \leq i \leq n$.
- . s = a first-order logical sentence, composed by variables and logical connectives.
- . $VAR(s) = \{v_i \mid s \text{ contains } v_i\}$.
- . $R = \{r_1, r_2, \dots, r_m\}$ is the set of all rules of the rule base, where r_i is a rule of 2-tuple (LHS_i, RHS_i) , $1 \leq i \leq m$. The LHS_i is the left-hand side (condition) sentence of r_i , and

RHS_i is the right-hand side (action) sentence of r_i .

. $C = \{c_1, c_2, \dots, c_t\}$ is the set of all partitions of the rule base. c_i , a rule class, is a subset of R , and $\bigcup_{1 \leq i \leq t} c_i = R$, $c_j \cap c_k = \varphi$, if $j \neq k$

. B is a first-order logical rule base of 3-tuple (V, R, C) .

Assume that there are u links in the relationship graph. Now we give the definitions about relationship graph.

. l_i is a link of 3-tuple $(V_i, CAUSE_RULE_i, EFFECT_RULE_i)$, $V_i \subseteq V$, $1 \leq i \leq u$,

$CAUSE_RULE_i = r_j$, where $V_i \subseteq VAR(LHS_j)$,

$EFFECT_RULE_i = r_k$, where $V_i \subseteq VAR(RHS_k)$.

. $L = \{l_1, l_2, \dots, l_u\}$ is the set of links of the relationship graph.

. $P = \{p_1, p_2, \dots, p_t\}$ is the set of all rule classes of the rule base, where p_i is a partition, $\bigcup_{1 \leq i \leq t} p_i = R$, $p_j \cap p_k = \varphi$, if $j \neq k$.

. $IN(p_i) = \{l_j \mid l_j \in L, CAUSE_RULE_j \notin p_i, EFFECT_RULE_j \in p_i\}$ is the set of incoming links of a partition p_i .

. $OUT(p_i) = \{l_j \mid l_j \in L, CAUSE_RULE_j \in p_i, EFFECT_RULE_j \notin p_i\}$ is the set of outgoing links of a partition p_i .

. $V(p_i) = \{l_j \mid l_j \in L, CAUSE_RULE_j \in p_i \text{ or } EFFECT_RULE_j \in p_i\}$ is the set of all links related to a partition p_i .

. I =incoming variables of a relationship graph, $I \subseteq V$, $I = \bigcup_i VAR(LHS_i) - \bigcup_i VAR(RHS_i)$,

where $1 \leq i \leq m$.

. O =outgoing variables of a relationship graph, $O \subseteq V$, $O = \bigcup_i VAR(RHS_i) - \bigcup_i VAR(LHS_i)$,

where $1 \leq i \leq m$.

. G = a relationship graph of 6-tuple (V, R, P, L, I, O) .

A rule base has one-to-one mapping to a relationship graph. A rule class of a rule base has one-to-one mapping to a partition of a relationship graph. A rule r_i is *connected* to a partition p_j if there exists an incoming link or an outgoing link between r_i and p_j , $1 \leq i \leq m, 1 \leq j \leq t$. We illustrate the definitions mentioned in Example 3. The rules in Example 3 are about the TCP SYN Flood Attack, in such attack the intruder sends huge TCR SYN packets to the victim computer, making the network of the victim unavailable.

Example 4

In this example, five variables relevant to the TCP SYN Flood Attacks are introduced: pps (packets per second), hl (heavy loading), $spps$ (SYN packets per second), sf (SYN Flood), and $alert$ (alert of the intrusion detection system). Therefore, the variables of the rule base are $V = \{pps, ht, util, hl, spps, sf, alert\}$. Assume that the rules of the rule base are $R = \{r_1, r_2, r_3\}$, as follows:

- r_1 : **If** GreaterThan(pps , 30000) **Then** ht
 r_2 : **If** $hl \wedge$ GreaterThan($spps$, 100) **Then** sf
 r_3 : **If** sf **Then** $alert$

Assume that the rule classes of the rule base are $C_1 = \{c_1, c_2\}$, $c_1 = \{r_1, r_2\}$, $c_2 = \{r_3\}$ and the rule base is $B_1 = (V, R, C_1)$. The relationship graph corresponding to B_1 is $G_1 = (V, R, L, P_1, I, O)$, which is illustrated in Figure 7.1. The links (which is represented by arrows) are $L = \{l_1, l_2\}$, $l_1 = (\{hl\}, r_1, r_2)$, $l_2 = (\{sf\}, r_2, r_3)$. $IN(p_1) = \varnothing$,

$OUT(p_1) = \{l_1\}$; $IN(p_2) = \{l_1\}$, $OUT(p_2) = \emptyset$. r_1 is connected to p_2 , r_2 is connected to p_1 , but r_3 is not connected to p_1 . The partitions of G_1 , corresponding to the C_1 , are $P_1 = \{p_1, p_2\}$, $p_1 = \{r_1\}$, $p_2 = \{r_2, r_3\}$. The incoming variables of G_1 is $I = \{pps, spps\}$, and the outgoing variable is $O = \{alert\}$. The virtual links connected to I and O are represented by dotted arrows.

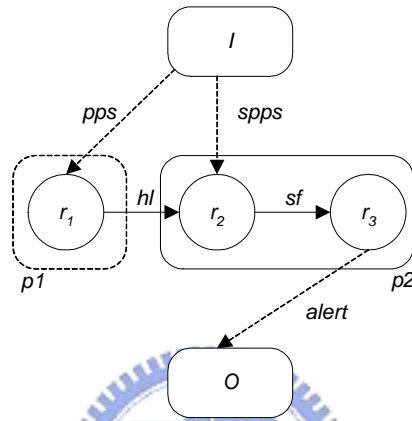


Figure 7.1: A relationship graph G_1

For each rule class in the rule base, there is exactly one mapping partition in the relationship graph. Note that the logical meaning of the rules is eliminated, because the relationship graph is used for an intermediate representation for partitioning (which will be introduced in the Section 4.2), not for logical inference.

7.1.2 Criteria of Relationship Graph Partitioning

As discussed before, the partitions of relationship graphs represent exactly the rule classes of rule bases. Therefore, “good relationship graph partitioning” means “good rule class classification”. We define two criteria for “good partitioning” according to our second goal, optimizing the structural and the semantic meanings, respectively.

The first criterion is to optimize the structural meanings of a relationship graph, or minimize the links cut by the partitioning if possible [KK98]. That is, for a relationship graph, the average inter-partition links (incoming links and outgoing links) should be minimized. Assume that a relationship graph G contains t partitions, $\{p_1, p_2, \dots, p_t\}$, each partition p_i contains $|IN(p_i)|$ incoming links and $|OUT(p_i)|$ outgoing links. Let $L_p(p_i)$ be the total number of the incoming and outgoing links of p_i ; that is, $L_p(p_i) = |IN(p_i)| + |OUT(p_i)|$. Let $L_G(G)$ be the average number of the incoming and outgoing links of G ; that is,

$$L_G(G) = \sum_{1 \leq i \leq t} L_p(p_i) / t \quad (1)$$

Therefore, the criterion about optimizing structural meanings is defined as follows:

Definition 6: Structural Succinctness Criterion

For a relationship graph G , minimize $L_G(G)$ if possible.

The second criterion is to optimize the semantic meanings of a relationship graph, or minimize the intra-partition semantic heterogeneity. The semantic distance [BH01] is utilized to measure the semantic heterogeneity. For two variables v, v' , the semantic distance is given by $S_v(v, v') =$ “the number of links from v to v' in the shared vocabulary ontology” + 0.5 * ”the number of changes of directions” (this definition will be explained in Section 5.2.1). For a partition p_i with u variables, the average semantic distance is given by

$$S_p(p_i) = \sum_{1 \leq j, k \leq u, j \neq k} S_v(v_j, v_k) / C(u, 2) \quad (2)$$

Let $S_G(G)$ be the average semantic distance of G ; that is,

$$S_G(G) = \sum_{1 \leq i \leq t} S_p(p_i) / t \quad (3)$$

Therefore, the criterion about optimizing semantic meanings is defined as follows:

Definition 7: Intra-Cluster Semantic Clustering Criterion

For a relationship graph G , minimize $S_G(G)$ if possible.

Except calculate the distance of links (facts) inside a cluster, we can also calculate the semantic distances between links of a cluster to that of all the other clusters, which is so called inter-cluster semantic clustering criterion. For better maintenance purpose, to separate the links (facts) which are quite different in the semantic meaning is helpful to distinguish the differences between clusters. Based on this idea, an Inter-cluster semantic clustering criterion is proposed to effect our clustering process to separate more irrelative links. For two clusters $c1$ and $c2$, inter-cluster semantic clustering criterion from cluster $c1$ to cluster $c2$ is given by $I(c1, c2) =$ “The average semantic distance from all links in $c1$ to all links in $c2$ ”, where the semantic distances from link to link is calculated as that is defined in Criterion 2. First, the Inter-cluster semantic distance is given by:

$$I_p(p_i) = \frac{1}{t-1} \sum_{1 \leq j \leq t, j \neq i} \frac{\sum_{1 \leq x \leq \|V(p_i)\|} \sum_{1 \leq y \leq \|V(p_j)\|} S_v(v_x, v_y)}{\|V(p_i)\| \times \|V(p_j)\|}, \text{ where } v_x \in V(p_i), \text{ and } v_y \in V(p_j).$$

and hence the $I(G)$ is defined as:

$$I_G(G) = \frac{1}{t} \sum_{1 \leq j \leq t} I(p_j)$$

Definition 8: Inter-cluster Semantic Clustering Criterion

For a relationship graph G , maximum $I_G(G)$ if possible.

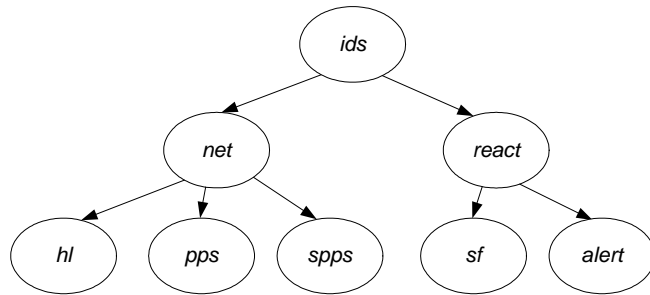


Figure 7.2: Part of the shared vocabulary ontology in the intrusion detection domain

Table 7.1. The calculated semantic distances

	<i>pps</i>	<i>hl</i>	<i>spps</i>	<i>sf</i>	<i>alert</i>
<i>pps</i>	0	2.5	2.5	4.5	4.5
<i>hl</i>	2.5	0	2.5	4.5	4.5
<i>spps</i>	2.5	2.5	0	4.5	4.5
<i>sf</i>	4.5	4.5	4.5	0	2.5
<i>alert</i>	4.5	4.5	4.5	2.5	0

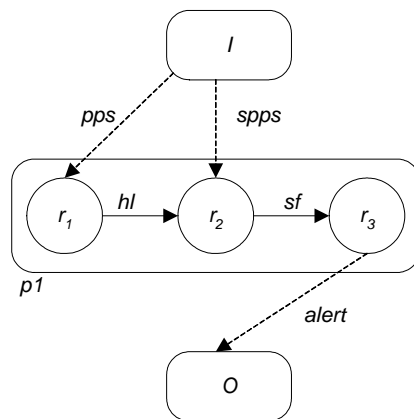


Figure 7.3: The relationship graph G_2

Example 5

Let relationship graph $G_2 = (V, R, L, P_2, I, O)$, $P_2 = \{p_3\}$, $p_3 = \{r_1, r_2, r_3\}$, as in Figure 7.2. The partitions of G_2 are different from those of G_1 in Example 3. Assume that the shared vocabulary ontology about the variables in V is shown in Figure 7.3. In the ontology mentioned above, the first three variables are in the category *net* (network signatures), the last two variables are in the category *react* (intrusion reaction), and all variables are above the category *ids* (intrusion detection system) category. For all of the variables in V , the semantic distances are given in Table 1. We evaluate the partitioning situations of G_1 and G_2 based upon the following three criteria:

- . *Criterion 1*: $L_G(G_1)$ is $(2+3) / 2 = 2.5$, and $L_G(G_2)$ is $(3) / 1 = 3$.
- . *Criterion 2*: $S_G(G_1)$ is $((2.5/1) + (23/6)) / 2 = 3.17$, and $S_G(G_2)$ is $(37/10) / 1 = 3.7$.
- . *Criterion 3*: $I_G(G_1)$ is $((25.5/8)+(25.5/8)) / 2 = 3.19$, and $I(G_2)$ is $0 / 1 = 0$.

In the example above, we conclude the partitioning of G_1 is better than that of G_2 according to *Criterion 1*, *Criterion 2*, and *Criterion 3*. These three criteria are used later in our partitioning algorithm for optimizing the structural and the semantic meanings.

7.2 Knowledge Fusion Framework

The process of our proposed approach consists of three phases: the preprocessing phase, the partitioning phase, and the ontology construction phase. The whole process is illustrated in Figure 7.4. Firstly, the preprocessing phase deals with syntactic problems such as format transformation and rule base cleaning, and construct the

relationship graph according to the cleaned, transformed flat rule base. Secondly, the relationship graph is partitioned according to *Criterion 1* and *Criterion 2* in the partitioning phase. Finally, the new ontology of the flat rule base is constructed using the partitioned relationship graph in the ontology construction phase. The three phases can be described in detail in the rest of this section.

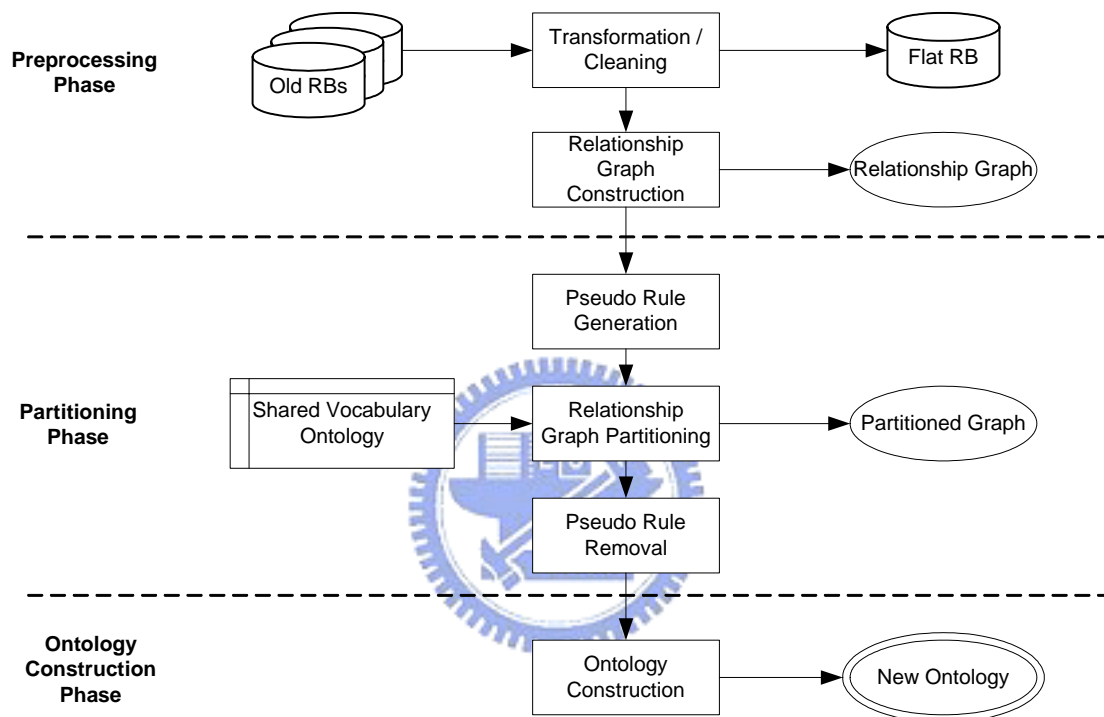


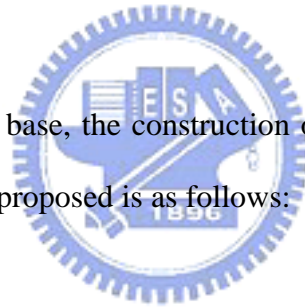
Figure 7.4: The knowledge fusion framework

7.2.1 Preprocessing Phase

The preprocessing phase consists of format transformation, rule base cleaning, and relationship graph construction. The format transformation of the source rule bases consists of two steps: one is to transform the rules to first-order logic, and the other is to remove the ontologies of the rule bases. In this paper, we assume that the syntactic heterogeneity is solved by ODBC, HTML, XML, and other related technologies

[VSV+01]. After preprocessing, the rules from all rule bases are then stored into a flat rule base. There is currently no ontology about the flat rule base. The ontology will be built in the relationship graph partitioning phase.

After all rules of the original rule bases are logically preprocessed, we should put all rules together and perform knowledge cleaning, such as validation and verification. The problem with rules includes redundancy, contradiction/conflict, circularity and incompleteness [GR89][RN95]. Directed Hypergraph Adjacency Matrix Representation [RSC97] is used to validate and verify the rules for completeness, correctness, and consistency. This cleaning step provides a basis for the relationship graph construction.



After cleaning the flat rule base, the construction of the relationship graph can be performed. The algorithm we proposed is as follows:

Algorithm 7.1: Relationship Graph Construction Algorithm

Input: A rule base $B = (V_B, R_B, C_B)$

Output: An un-partitioned relationship graph $G = (V_G, R_G, P, L, I, O)$

Step 1. Set $V_G = V_B, R_G = R_B$.

Step 2. For each two rules $r_1, r_2 \in R_G$, let S be the intersection of the variables of RHS_1 and the variables of LHS_2 , add the link (S, r_1, r_2) to L .

Step 3. Set I as the variables of all LHS sentences of all rules.

Step 4. Set O as the variables of all RHS sentences of all rules.

7.2.2 Partitioning Phase

Before introducing our proposed algorithm, we take a brief discussion about shared vocabulary ontology, semantic distance function, and pseudo rules in the following sections.

7.2.2.1 Shared Vocabulary Ontology and Semantic Distance Function

The shared vocabulary ontology can be constructed either by domain experts or by the general lexical reference system, such as WordNet [MBF+90]. If the knowledge sources to be fused are in the same or related domains, the customized shared vocabulary ontology for the domains is more proper than general one.

The semantic distance function we use is based on the Hirst and St-Onge's measure of semantic relatedness [HS98], and is defined as follows:

$$S_v(v_1, v_2) = path_length + c * d, \quad \forall v_1, v_2 \in V, \quad (4)$$

where $path_length$ is the length from v_1 to v_2 in the shared vocabulary ontology, d is the number of changes of direction in the path, and c is constant. If the path does not exist, the function returns “*infinity*”. $S_v(v_1, v_2) = 0$ if and only if $v_1 = v_2$.

7.2.2.2 Pseudo Rules

Before partitioning the relationship graph, we should firstly transform the incoming variables and outgoing variables of a relationship graph into two set of pseudo rules, Pseudo Incoming Rule Set and Pseudo Outgoing Rule Set, respectively. These pseudo

rules add connections among rules and help for dealing with shallow knowledge, of which the connected rules may be too few for generating partitions. Each of the incoming variables is transformed to a Pseudo Incoming Rule by the following format:

If *TRUE* Then <An_Incoming_Variable>

Similarly, each of the outgoing variables is transformed to a Pseudo Outgoing Rule by the following format:

If <An_Outgoing_Variable> Then *EMPTY*

The pseudo rules should be eliminated after partitioning the relationship graph. The removal of the pseudo rules is simply to discard all pseudo rules of all rule classes. If a rule class is empty after the removal, remove the rule class too.

Example 6

In this example, we start with the un-partitioned relationship graph from G_1 and G_2 , as illustrated in Figure 7.5. The rules are the same as those in Example 3. After the transformation, three pseudo rules are generated:

s_1 : **If *TRUE* Then *pps***

s_2 : **If *TRUE* Then *spps***

s_3 : **If *alert* Then *EMPTY***

The un-partitioned, pseudo-rules-added relationship graph for G_1 and G_2 is illustrated in Figure 7.6.

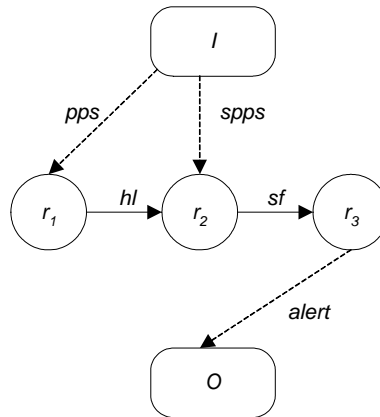


Figure 7.5: The un-partitioned relationship graph

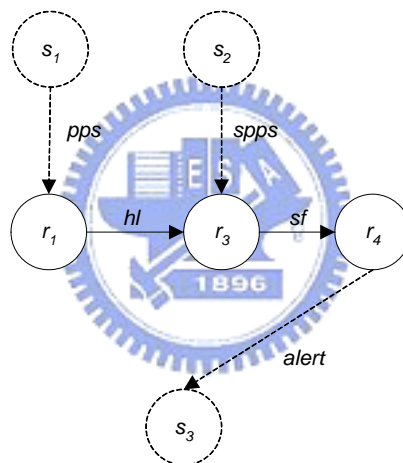


Figure 7.6: The un-partitioned, pseudo-rules-added relationship graph

7.2.2.3 The Partitioning Algorithm

After the un-partitioned relationship graph (including pseudo rules) is constructed, the partitioning process can be performed. Combining *Criterion 1*, *Criterion 2* and *Criterion 3*, the following function for a partition p_i is used for the partitioning process:

$$F_p(p_i) = L_p(p_i) + k * S_p(p_i) - l * I_p(p_i), \quad (5)$$

where k and l are defined before algorithm running, represents the weight of the importance of three criteria. The following algorithm is proposed based on the greedy growth concept.

Algorithm 7.2: Relationship Graph Partitioning Algorithm

Input: An un-partitioned relationship graph G , pseudo rules added.

Output: A partitioned relationship graph G' , pseudo rules not removed yet

Step 1. Randomly select a rule from rules of G , and add it to a new partition p .

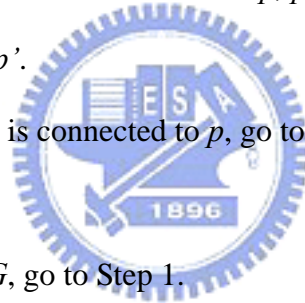
Step 2. Select rule r from G which is connected to p , $p' = p + \{r\}$, with minimal $F_p(p')$.

Step 3. If $F_p(p') < F_p(p)$, $p = p'$.

Step 4. If there is any rule that is connected to p , go to Step 2.

Step 5. Add p to G' .

Step 6. If there is any rule in G , go to Step 1.



Example 7

In this example, we continue with the un-partitioned, pseudo-rules-added relationship graph from G_1 and G_2 , as illustrated in Figure 7.7. Let $k=0.5$ and $l=0.5$ in the semantic distance function and $c=1$ in the algorithm. For the shared vocabulary ontology illustrated in Figure 7.3 (of which each path of any two nodes contains only one “change of directions”), the values of semantic distance function are the same as Table 1.

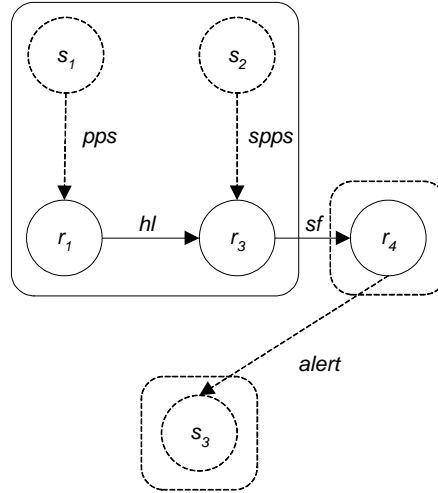


Figure 7.7: The relationship graph G_3 , before removing the pseudo rules

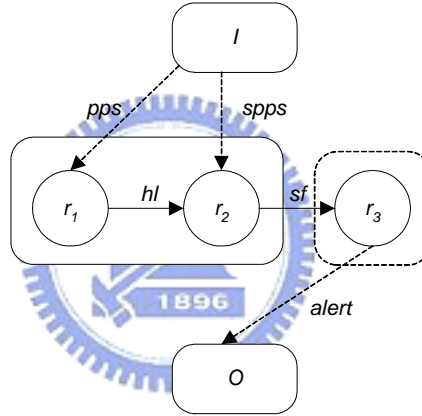


Figure 7.8: The relationship graph G_3

Firstly, we select r_2 randomly, and add it to a new partition p_4 ; $L_p(p_4) = 2 + 1 = 3$, $S_p(p_4) = (2.5 + 4.5 + 4.5) / 3 = 3.83$, $I_p(p_4) = 0$, $F_p(p_4) = L_p(p_4) + S_p(p_4) - I_p(p_4) = 6.83$. Consider three partitions $p_5 = \{r_2, r_1\}$, $p_6 = \{r_2, r_3\}$, and $p_7 = \{r_2, s_2\}$. $F_p(p_5) = 6.5$, $F_p(p_6) = 6.83$, and $F_p(p_7) = 4.83$. Therefore, p_7 is chosen to be the only one partition now. Consider partition $p_8 = \{r_2, s_2, r_1\}$ and $p_9 = \{r_2, s_2, r_3\}$; $F_p(p_8) = 4.17$, $F_p(p_9) = 4.5$; p_8 is chosen. Now consider $p_{10} = \{r_2, s_2, r_1, s_1\}$, $p_{11} = \{r_2, s_2, r_1, r_3\}$; $F_p(p_{10}) = 3.75$, $F_p(p_{11}) = 4.35$; p_{10} is chosen. Then consider $p_{12} = \{r_2, s_2, r_1, s_1, r_3\}$; $F_p(p_{12}) = 3.9 > F_p(p_{10})$; p_{10} is retained. Since all rules connected to p_{10} are checked, p_{10} is the

finally obtained partition.

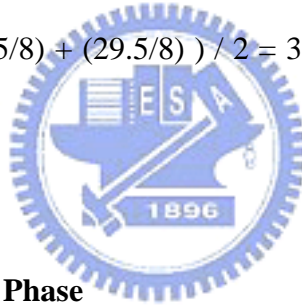
Now we pick s_3 as p_{11} . $F_p(p_{11}) = L_p(p_{11}) + S_p(p_{11}) - I_p(p_{11}) = 1 + 0 - 4 = -3$.

Consider partition $p_{12} = \{s_3, r_3\}$; $F_p(p_{12}) = 1 + 2.5 - 3.69 = -0.19$; p_{11} is confirmed.

Since there is only one rule r_3 left, it is a partition itself, p_{13} . Therefore, three partitions are generated, as illustrated in Figure 7.7. The final result of the algorithm, G_3 , is illustrated in Figure 7.8. $G_3 = (V, R, L, P_3, I, O)$. $P_3 = \{p_{10}, p_{13}\}$, $p_{10} = \{r_1, r_2\}$, $p_{13} = \{r_3\}$.

The evaluations of G_3 by the three criteria are as follows. The result G_3 is better than G_1 and G_2 , by both criteria.

- . *Criterion 1:* $L_G(G_3) = (2+3) / 2 = 2.5$, which is the same as G_1 but smaller than G_2 .
- . *Criterion 2:* $S_G(G_3) = ((21/6) + (2.5/1)) / 2 = 3$, which is smaller than G_1 and G_2 .
- . *Criterion 3:* $I_G(G_3) = ((29.5/8) + (29.5/8)) / 2 = 3.69$, which is larger than G_1 and G_2 .



7.2.3 Ontology Construction Phase

The final phase of our proposed framework is to construct ontology according to the partitioned relationship graph. Ontology includes many aspects of conceptualization [HPH01][RN95][SOW00]. Among them, two important aspects are discussed in our work: classes and relationships.

Three classes are generated by the relationship graph: Variable, Rule, and RuleClass, which map to the variables, rules, and partitions, respectively. The name of a RuleClass is given arbitrarily but uniquely. Table 2 shows the classes and relationships of the generated ontology.

Table 7.2: The classes and relationships of the generated ontology

Class	Relationships (Properties)		
	Property	Type	Description
Variable	Name	Unique Text	The name
Rule	Name	Unique Text	The name of a rule
	Rule Class	Rule Class Name	The rule class belonged
	Ante. Var.	Set of Variable	The LHS variables
	Cons. Var.	Set of Variable	The RHS variables
Rule-Class	Name	Unique Text	The name
	Rules	Set of Rule Name	The rules contained
	Key. Var	Set of Variable	The key variable
	In. Var.	Set of Variable	The incoming variables
	Out. Var.	Set of Variable	The outgoing variables

Three kinds of relationships, represented by properties, are generated by the relationship graph: the *Rule-RuleClass relationships*, the *Rule-Variable relationships*, and the *RuleClass-Variable relationships*. The Rule-RuleClass relationships map to the members of the partitions, and are represented by the properties *belongsTo* and *hasRule* of Rule and RuleClass, respectively. The Rule-Variable relationships *hasLHSVariables* and *hasRHSVariables* represented by the corresponding properties of rule are gained from the involved variables of LHS and RHS of the rules respectively. The RuleClass-Variable relationships *hasIncomingVariables* and *hasOutgoingVariables* represented by the corresponding properties of rule class are gained from the incoming variables and outgoing variables of the partitions respectively. In addition to the name, incoming and outgoing variables, a RuleClass contains another semantic relevant property, *hasKeyVariables*. The *hasKeyVariables* property is a set of the names of the lowest super-ordinates (most specific common subsumers) of all terms involved in the rule class in the shared vocabulary ontology. This property indicates that the key variables of a RuleClass, can briefly summarize the semantic meanings of a RuleClass.

Example 8

For the relationship graph G_3 in Example 6 (Figure 7.8), a RuleClasses c_1 , represented by DAML+OIL[HPH01], is shown in Figure 7.9.

<pre><RuleClass rdf:about="#c1"> <hasRule> <daml:oneOf rdf:parseType="daml:collection"> <Rule rdf:about="#r1"/> <Rule rdf:about="#r2"/> </daml:oneOf> </hasRule> <hasKeyVariable> <daml:oneOf rdf:parseType="daml:collection"> <Variable rdf:about="#ids"/> </daml:oneOf> </hasKeyVariable></pre>	<pre><hasIncomingVariable> <daml:oneOf rdf:parseType="daml:collection"> <Variable rdf:about="#pps"/> <Variable rdf:about="#sps"/> </daml:oneOf> </hasIncomingVariable> <hasOutgoingVariable> <daml:oneOf rdf:parseType="daml:collection"> <Variable rdf:about="#sf"/> </daml:oneOf> </hasOutgoingVariable> </RuleClass></pre>
---	---

Figure 7.9: Ontology of RuleClass c_1



Chapter 8 Implementation and Experiments

8.1 Implementation of NORM

DRAMA, a NORM based rule base platform, is a product of Coretech Inc [DRA03], Taiwan, which is developed in cooperation with Knowledge and Data Engineering Laboratory (KDB Lab.) of National Chiao Tung University, Taiwan. DRAMA is implemented using Java, and it includes DRAMA Server, DRAMA Console, DRAMA Knowledge Extractor, DRAMA Rule Editor.



Figure 8.1: DRAMA Console

DRAMA Server is implemented to manage rule bases, which is used to contain and process knowledge, and provide rule base services. NORM-modeled knowledge can be contained in DRAMA Server and inferred according to user given facts. DRAMA Console is a command mode interface for user to access DRAMA Server.



Figure 8.2: DRAMA Knowledge Extractor

DRAMA Knowledge Extractor is implemented by repertory grid mechanism [HT90] [TT02] a knowledge acquisition mechanism, to extract and retrieve knowledge from experts. The extracted knowledge will be transformed into NORM rules which will be used in DRAMA Server.



Figure 8.3: DRAMA Rule Editor

For the knowledge already defined in rule format, DRAMA Rule Editor with a GUI interface is provided for editing NORM knowledge class and rules. Differ from traditional rule base building tools, DRAMA Rule Editor is a user friendly GUI with

drag and drop operations.

Also, Application Programming Interface (API) to access DRAMA server is also provided for developing DRAMA integrated systems.

8.2 Implementation of KA mechanism

In order to evaluate the performance of our *KFCA* algorithm, a set of DoS intrusions and corresponding descriptions is used as the experimental data. Since the dictionary we used in the experiment is categorized, we can evaluate the accuracy of our experimental results by comparing. The cases used in the experiment is categorized as Table 4 and the concept hierarchy used to calculate semantic distance is shown as Figure 4:



Table 4: DoS Intrusion dataset

Category	Description	# of intrusions
Using system command	Attacking the system with system level operations, for example, gaining access to root account by some system commands.	11
Using packet content	Attacking the system with specific packet content, for example, WindowsNT PPTP flood denial add ^D (control-d) in the packet to crash remote PPTP host.	17
Using protocol vulnerability	Attacking the system by not following protocol definition, for example, ping of death attack using corrupted ICMP packets to attack the system.	21
Using system vulnerability	Attacking the system by taking advantage of system vulnerability, for example, Cisco DoS attack send data to specific port 7161 to crash the router.	15
Using service vulnerability	Attacking a service by taking advantage of any service specific vulnerability, for example, FTP ServU CWD overflow, which issue a CWD command followed by long argument.	46
Using hardware vulnerability	Attacking a specific hardware to crash it, for example, +++ATH0 modem hangup attack send ICMP request to hangup remote modem to disable the connection.	6
Consuming resources	Attacking system by trying consuming all the resources to	14

	provide service, for example, ICMP flood attack, which sends lots of ICMP packet to make the system busy responding the request and can not provide service anymore.	
Using application properties	Attacking system by taking advantage of an application specific vulnerability, for example, ICQ Denial of Service attack the remote client by sending specific request to an application CGI.	57

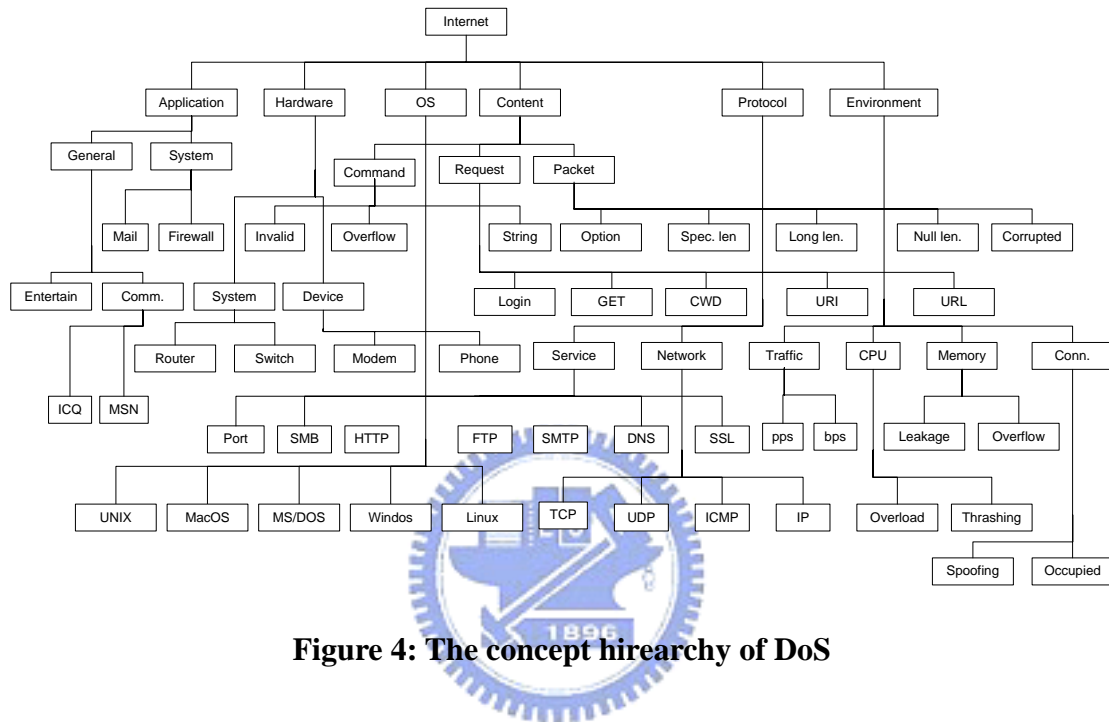


Figure 4: The concept hierarchy of DoS

In order to evaluate the performance of our knowledge clustering method, two indexes, purity and distribution, are defined to represent the accuracy of clustering result. One cluster can consist of different categories of cases, and the majority category of a cluster is defined as the category with most number of cases in this cluster comparing to other categories. Purity is the ratio of the majority category in a cluster, which is calculated for each cluster to represent how pure a cluster is. The distribution is calculated for each category, which is used to identify how cases of a category is distributed in different clusters; in other word, the distribution of a category is the number of clusters each of which contains at least one case of the

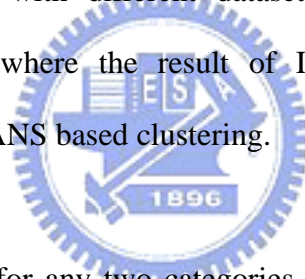
category. The formula to calculate these two indexes are shown below:

$$Purity = \frac{\|\{T_{max}\}\|}{\|\{T\}\|}, \text{ where } \{T_{max}\} \text{ is the set of cases which belongs to the majority}$$

category in a cluster.

$Distribution_{CAT} = \|\{C, \text{ where cluster } C \text{ contains terms of category } CAT\}\|$, where $\{C\}$ is the set of clusters each of which contains at least one of the case of category CAT .

Based on the above indexes, experiments have been done to show the accuracy of our clustering algorithm with different dataset and to see the influence of increasing variety of data, where the result of ISO-DATA based clustering is compared with that of K-MEANS based clustering.



In the first experiment, for any two categories selected, some of the cases are sampled as shown in Table 7. In addition to the ISO-DATA algorithm we used in the algorithm, K-MEANS algorithm is also implemented for comparing, thus we can not only show the index values for ISO-DATA based algorithm, but also K-MEANS based algorithm can be compared. After applying these data to our prototype system, the results are obtained as Table 6. It shows that the average purity values of all clusters for ISO-DATA based is 0.934 and higher than K-MEANS based algorithm. But for distribution value, ISO-DATA tends to divide the samples into smaller clusters, because ISO-DATA tries to improve the performance by splitting the clusters to reduce the standard deviation of clusters. For example, in Dataset 1, ISO-DATA clusters sample data into four clusters.

Table 8.1. The experimental datasets of randomly selected categories

	Category 1	Category 2
	Category	Category
Dataset 1	System command	Consume Resource
Dataset 2	Packet Content	System Vulnerability
Dataset 3	Service Vulnerability	Application Vulnerability
Dataset 4	Hardware Vulnerability	Protocol Vulnerability
Dataset 5	Protocol Vulnerability	Service Vulnerability

Table 8.2. The experimental result for the dataset in Table 7

Dataset	ISO-DATA Based		K-MEANS Based	
	Purity	Distribution	Purity	Distribution
1	1	2	0.72	2
2	0.93	3.5	0.71	2
3	0.85	4	0.77	2
4	1	2.5	1	2
5	0.89	3	0.79	2

In second experiment, ten percent of testing data are selected from different number of categories as shown in Table 7. After applying these data to our prototype system, the purity is shown in Table 8. The experimental result shows that as the number of categories of test data increases, the purity value for ISO-DATA based algorithm, unlike K-MEANS based algorithm, does not obviously decrease; and also as comparing to previous experiment, the purity value does not obviously decrease when the number of categories grows.

Table 8.3. The datasets with different numbers of categories contained

	# of categories	# of Intrusions
Dataset 1	2	38
Dataset 2	3	53

Dataset 3	4	67
Dataset 4	5	78
Dataset 5	8	187

Table 8.4. The experimental result for the dataset in Table 9

Dataset	ISO-DATA Based	K-MEANS Based
	Purity	Purity
1	1	0.738
2	0.941	0.752
3	0.925	0.713
4	0.913	0.671
5	0.852	0.563

All of these experiments show the worst purity value for ISO-DATA based algorithm is more than 0.8. It means more than 80% of cases are within the same category for each cluster obtained in our algorithm, and hence can be explained corresponding to a category. We also provide these clustering result to domain expert to review, and generally it was told that the result shows our algorithm clustering the cases into meaningful structure which can be useful for KA process about DoS knowledge.

8.3 Implementation of KF mechanism

In this section, we describe our experiments of the proposed knowledge fusion framework in the domain of network intrusion detection system, of which the ontologies vary dramatically and the real-time responses are required. The implementation is realized in Java (jdk1.3.1) on Intel Celeron 1G with 512MB RAM.

Table 8.5: Original categories and number of rules of two intrusion detection

systems

	Snort	Pakemon	Total
CGI	99	55	154
DOS	16	7	23
DNS	18	2	20
FTP	30	4	34
IIS	82	8	90
RPC	32	1	33
SMTP	19	14	33
Total	296	91	387

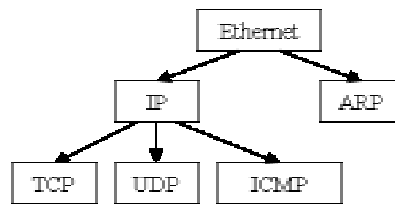


Figure 8.4: The shared vocabulary ontology built by the domain expert

We use the knowledge bases of two network intrusion detection system: Snort 1.8.1 [ROE99] and Pakemon 0.3.1 [TAK02]. In our experiment, we utilize only the intersection parts of rules of two intrusion detection systems. The categories and numbers of rules are shown in Table 3. Since Snort rules contain more information than Pakemon, for experimental purpose, the rules of snort will be transformed and simplified to the format similar to Pakemon rule format as following:

If $\langle protocol \rangle$ $\langle src_port \rangle$ $\langle dst_port \rangle$ $\langle content \rangle$ **Then** $\langle intru_type \rangle$ $\langle intru_name \rangle$

As illustrated in Figure 8.9, the shared vocabulary ontology about network intrusion detection system created by domain experts is quite simple, but is enough for the partitioning work. In the three experiments we made, the constant c is set to 0.5, and the constants $\{k, l\}$ are set to $\{1.0, 0\}$, $\{1.0, 1.0\}$, and $\{2.0, 2.0\}$ respectively. The results of these three experiments are shown in Tables 4, 5, 6, (the major category of each partition is shown in bolder form).

Table 8.6: The partitions and rules ($\{k, l\}=\{1.0, 0\}$)

Rules	Partitions	Rule composition in each partition
288	1	CGI*154, IIS*43, FTP*22, DNS*20, RPC*20, DOS*14, SMTP*15
99	1	IIS*47, SMTP*18, FTP*12, RPC*13, DOS*9

Table 8.7: The partitions and rules ($\{k, l\}=\{1.0, 1.0\}$)

Rules	Partitions	Rule composition in each partition
225	1	CGI*154, IIS*33, DNS*18, RPC*7, DOS*6, SMTP*4, FTP*2
67	1	IIS*65, CGI*1, DOS*1
30	1	SMTP*28, DNS*1, RPC*1
29	1	RPC*21, FTP*5, DOS*3
28	1	FTP*27, SMTP*1
9	1	DOS*7, IIS*2
5	1	RPC*4, DOS*1
3	1	DOS*3

Table 8.8: The partitions and rules ($\{k, l\}=\{2.0, 2.0\}$)

Rules	Partitions	Rule composition in each partition
67	1	CGI*66, IIS*1
31	1	IIS*30, SMTP*1
27	1	FTP*27
20	1	DNS*20
17	1	SMTP*17
8	1	RPC*8
6	1	IIS*6
5	1	RPC*5
4	1	DOS*4
3	3	DOS*3 / DOS*3 / IIS*3
2	7	(omitted)
1	179	(omitted)

When $k = 1.0$ and $l = 0$, the structural criterion dominates the results; therefore only 2 partitions are generated but the rules of the same category are not classified into the same partition. When $k = 1.0$ and $l = 1.0$, the semantic criteria, including intra-cluster and inter-cluster criterion, dominate the results, the classification of each partition is more clean (only few categories in a partition, and there is a majority category for a partition) but many 1-rule partitions are generated. But for the first cluster (the largest

cluster), inter-cluster criterion has no effect since there exists no other cluster in that time. When $k = 2.0$ and $l = 2.0$, the classification of each partition is even more clean. For lower semantic criterion weights, including k and l values, the partitions are fewer, but is more or less not quite clean. For higher semantic criterion weights, the partitions are more clean, but too much small partitions generated. The selection of k and l value can seriously effect the clustering result.

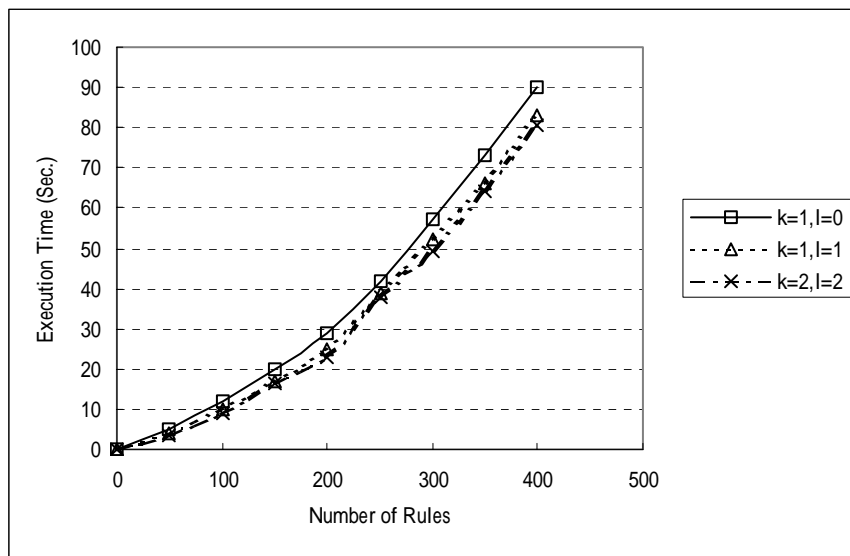


Figure 8.5: The execution time of the algorithms

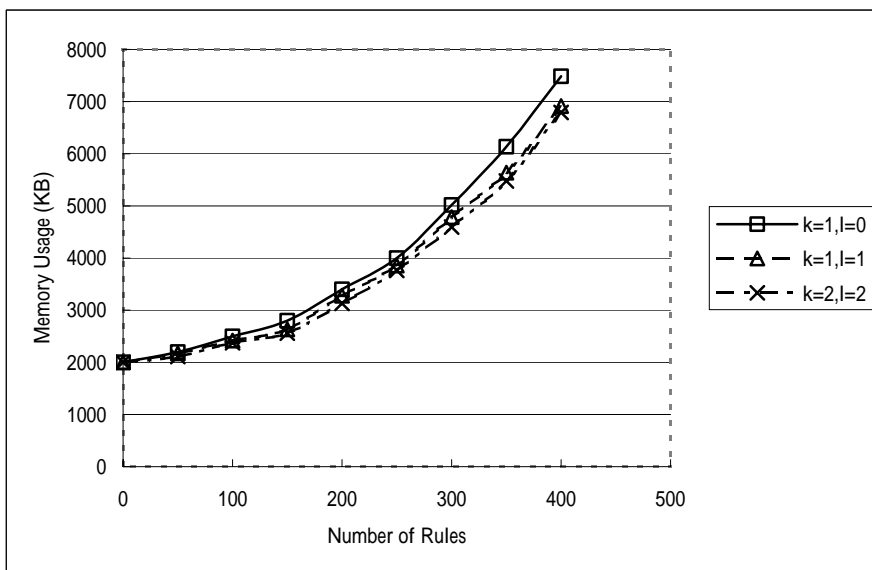


Figure 8.6: The memory usage of the algorithms

The execution time and memory usage are illustrated in Figure 8.10 and Figure 8.11, respectively. For lower k and l values (lower semantic criterion weights), each partition is bigger; therefore more time and more space are required to compute the three criterion values. And when semantic criteria get higher weights, each partition is smaller, and less time and less space are required.

8.4 Case Study: Computer Assisted Learning (CAL) Expert System

In CAL systems and researches [BS99][CHO96], Adaptive Learning is an important issue to be solved, and selecting appropriate learning content for different students is an important feature in Adaptive Learning. For different students in different learning situations, teachers want to provide different learning content to students to improve their learning performance. Therefore, processing teaching strategy which contains the knowledge about selecting learning content is important in CAL systems. However, traditional computer technologies like database query, which only select information according to some criteria of data instead of considering all of the factors influence learning achievement, is not suitable for expressing the knowledge of teachers to select learning content. Hence, Knowledge base system (KBS) is used in these systems for learning content selection purpose.

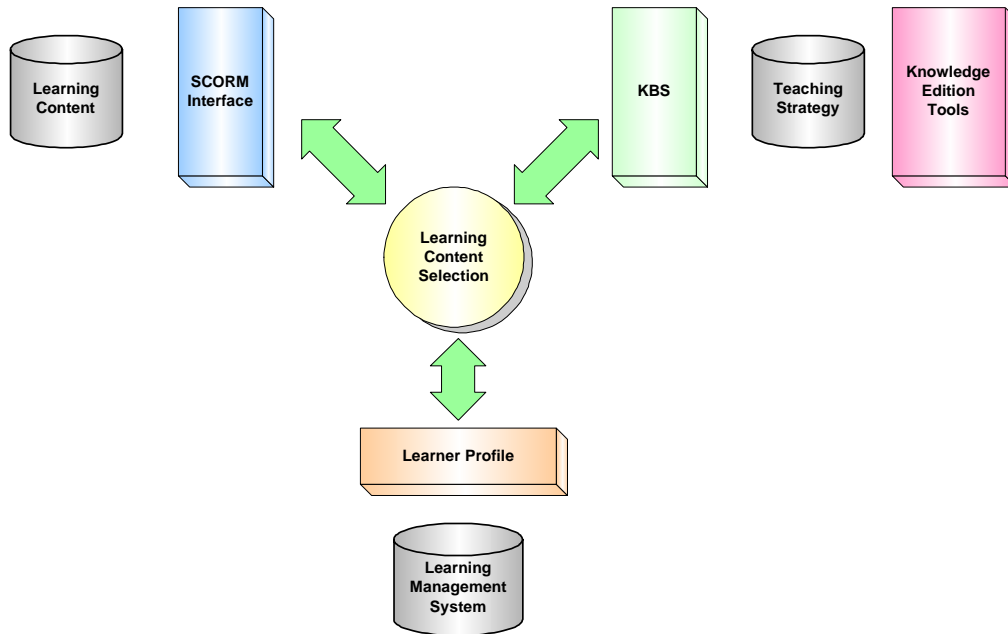


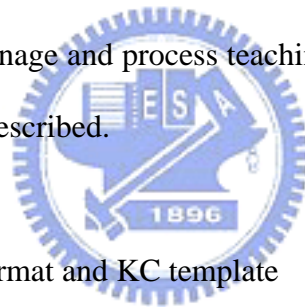
Figure 8.7: Components for Learning Content Selection System

As shown in figure 8.12, a Learning Content Selection System, which used to select appropriate learning content for students, consists of three components, including Learning Strategy, Student profile/records, and Learning Object. Each of these components should be managed by a specific system. In order to create, store, reuse and manage learning content, a Learning Content Management System (LCMS) is required. A Knowledge Base System is required for managing and processing the learning strategies, and a teaching platform is also required for monitoring and recording students' behaviors as learning profile.

Students are always different according to their learning achievements and learning behaviors even they study the same learning content. In order to improve their learning performance, teachers should prepare different learning content for different students, for example, teachers should provide easier learning content for the student with lower learning achievement. However, it is tedious and time consuming for a

teacher to prepare different learning content for all students, and a systematic and efficient mechanism to help selecting appropriate learning content for students is required. In our experiment, NORM knowledge model and DRAMA, is used in designing and implementing KBS for a CAL system to select appropriate learning content for learner, and solve the problem for teachers to prepare the learning content for different students.

In the following sections, the experiment including the usage of NORM rule base is introduced. First, the learning achievement of student and corresponding features is introduced, and then the meta data of the learning content, which contains the information and properties of learning content, is also designed. Finally, the platform to use NORM rule base to manage and process teaching strategy edited by teacher for selecting learning content is described.



- Design student profile format and KC template

According to previous studies of CAL [BS99] [CHO96], students' learning activities and corresponding learning achievements are important to find appropriate learning material for student to learn; for example, if a student is not good in mathematics according to the grades in exams, learning content about basic mathematics theorems should be included when we plan the topics for this student to learn; otherwise, these basic learning content should not be included.

In our prototype, following attributes are included in the learning profile of each student to represent his/her learning achievement of a learning topic:

Topic: The topic of the course to be recorded, for example, Mathematic, English, etc.

Grades: The grades got of the corresponding course or learning topic.

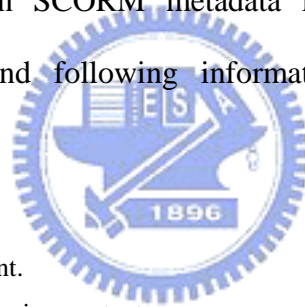
Progress: The progress of a course or a learning topic, maybe represent in percentage.

Learning Status: The learning status of a student in the corresponding course or learning topic, for example, study hard or normal.

Hence, a student's learning profile can be thought as a set of records to represent as the learning history of the student.

- Design SCORM compliant data format

According to the definition of SCORM Metadata, many information can be contained in the metadata for LCMS to understand and manage the learning content. For the LCMS system in this work, SCORM metadata is used for managing system imported learning content and finding appropriate learning content for student. However, not all the information contained in SCORM metadata is useful for learning content managing and retrieving, and following information is selected as managing information for our LCMS:



Title: The title of the learning content.

Keywords: The keywords of the learning content.

Version: The version of the learning content, useful to track the evaluation of the learning content.

Status: The status of the learning content, which maybe Draft, Final, etc.

Content Type: The content type of the data included in the learning content, which may be the data format of the learning content.

Requirements: The technical requirements to view the learning content, for example, Browser, Operating System, etc.

Interactive Type: The type of interaction between student and the learning content.

Interactive Level: The level of interaction between student and the learning content.

Learning Resource Type: The type of learning resource contained in the learning content.

End User: The type of the end user to use the learning content.

Fee: Indicate if fee is required to use the learning content.

Classification: The classification of the learning content.

As SCORM learning content needed to be managed, the above information contained

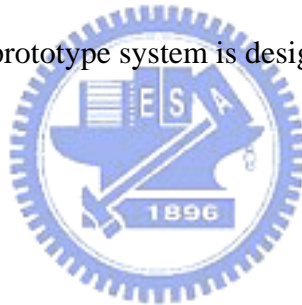
in the MANIFEST file of the SCORM learning object will be retrieved and stored into LCMS managing mechanism, and provide learning object searching, exchanging, and planning functionalities.

- Find a teaching domain and collect learning content

In our experiment, we select high school mathematic as the teaching domain, and learning content about high school mathematic are stored in the system and ready to provide to users of the system.

- Design the architecture

In order to provide learning content selection service based on teacher-defined strategy, the architecture of a prototype system is designed as shown in following figure:



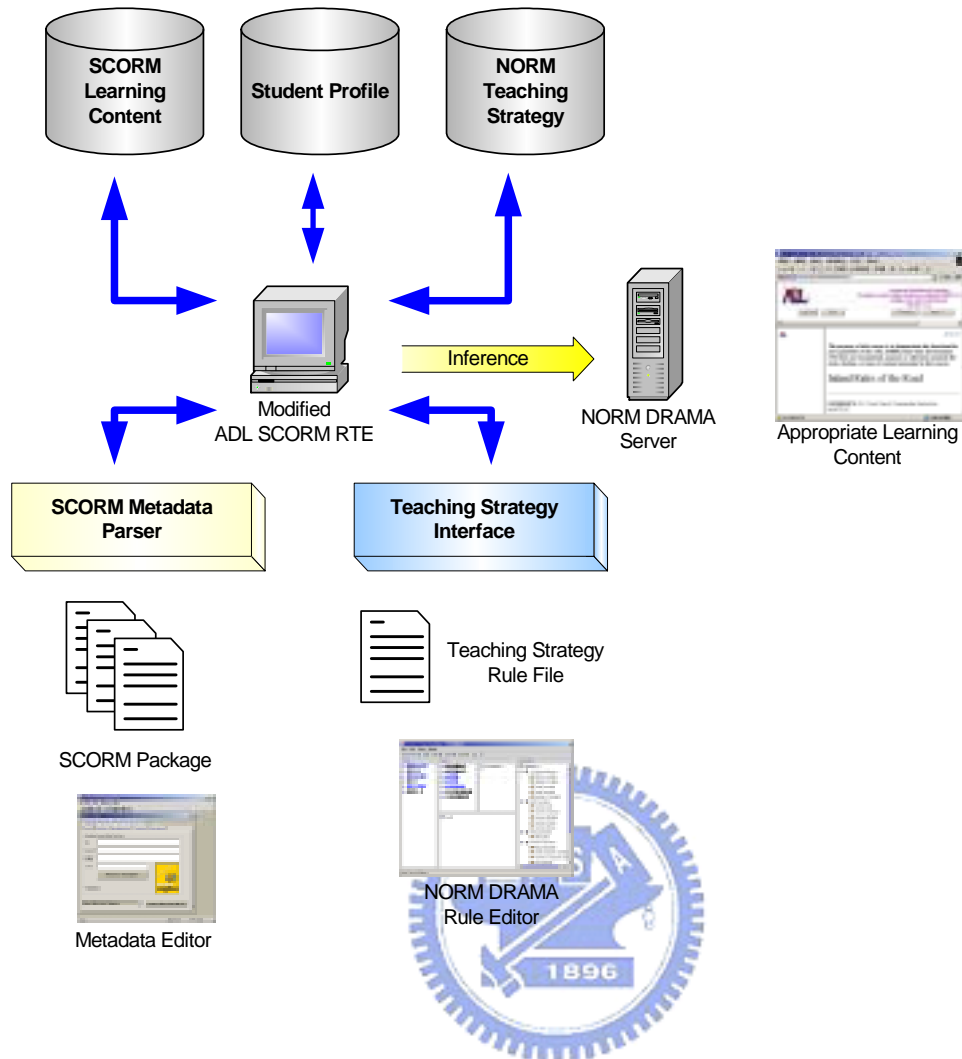


Figure 8.8: The architecture of prototype system

In this architecture, we use ADL SCORM Sample RTE (Sharable Content Object Reference Model (2003)) as the basic architecture to build an LCMS. ADL SCORM Sample RTE (Runtime Time Environment) is a basic LMS (Learning Management System) provided by ADL which satisfies SCORM RTE 2.0 standard. In SCORM Sample RTE, administrators can import SCORM packaged courses for learners of the system to study, and learners can register courses to start learning. However, currently there is no information for learners to understand what included in a course, how difficult the course is, etc. As the number of courses grows in this RTE, there will be a problem for learners to find appropriate course to study.

The metadata contained in SCORM packaging courses may include information about the course, which will be useful for learners to understand and select the course. Since the metadata of SCORM courses is formatted in XML, a SCORM Metadata Parser is implemented in the prototype to extract the meta information. On the other hand, the metadata for SCORM courses is generated using SCORM Meta-Data Generator Pro 1.2.0.

As we have mentioned, the selection of learning content considered not only the course information, but also the profile and learning history of learners must be considered. For this purpose, we also design a learner profile input interface for teachers to input students grades in each field, and will be used when learners trying to find appropriate learning content according to some learning content selection strategy defined by teachers.



In this prototype system, the learning content selection strategy (teaching strategy) is defined using NORM DRAMA rule editor as a rule file, and we designed a new function in the RTE for teachers to import new teaching strategy from rule file. When learners try to find appropriate learning content for him/her to study in some fields, they can use an imported teaching strategy and then select suitable learning content according to the meta information of courses and learner's learning profile. In order to process the knowledge included in teaching strategy, a NORM DRAMA Server is installed on the server, and when the RTE trying to process the teaching strategy, the prototype system will connect the NORM DRAMA Server and give corresponding facts for the server to infer. The result of the inference process will be used to select the learning content.

Currently, the server is hosted in “http://e-learning.nctu.edu.tw/norm” with high school mathematic learning material, and expect to be extended to all fields of high school education. Following are some snapshots of the NORM based Learning Management System.



Figure 8.9: Login page of the NORM based Learning Management System

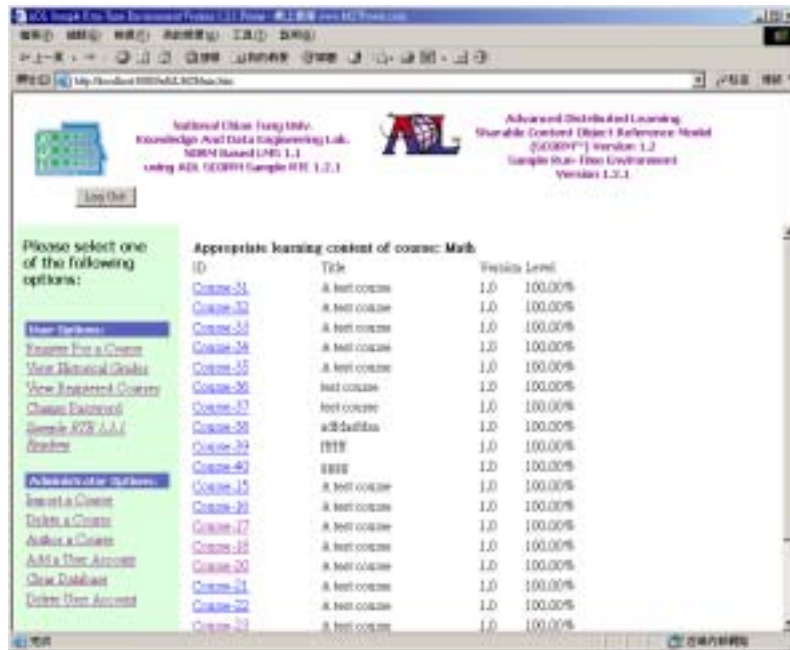


Figure 8.10: Selecting Learning content by inferring knowledge in DRAMA rule base.



Chapter 9 A Network Intrusion Detection Expert System (NID-ES) on New Object-oriented Rule Base Platform

9.1 The Architecture of Network Intrusion Detection Expert System (NID-ES)

In designing an intrusion detection system, three issues, including the representation of intrusion patterns, the tradeoff between complexity of detection process and system resources required, and the maintenance of expert knowledge, must be considered. To solve all these issues, a Network Intrusion Detection Expert System (NID-ES) based on the New Object-oriented Rule Base Platform is thus proposed. According to the definition of NORBP, there are four subsystems are designed, including Two Layer Network Intrusion Detection System, Intrusion Detection Knowledge Acquisition System, Intrusion Detection Knowledge Mining System, and Intrusion Detection Knowledge Bases Fusion System, and these four subsystems provide all the mechanisms required in the knowledge management lifecycle. The following figure shows the concept of NID-ES:

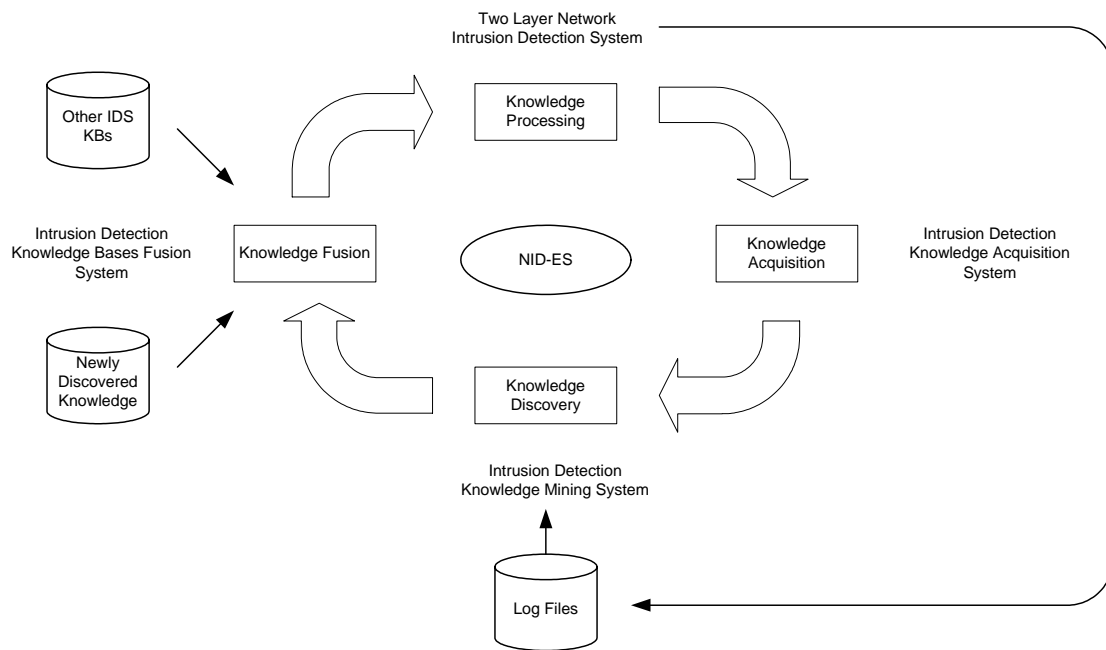


Figure 9.1: The concept of NID-ES

As shown in this figure, the four subsystems of NID-ES are designed according to the lifecycle of knowledge management and construct a complete knowledge platform. Where Two-Layer Network Intrusion Detection System is designed to process the knowledge for intrusion detection and monitor the network environment to find possible intrusion behaviors. NORM rule base and inference engine are used as the kernel of Two-Layer Network Intrusion Detection System. To Acquire the knowledge of intrusion detection from experts, an Intrusion Detection Knowledge Acquisition System, is designed to retrieve the knowledge we need for intrusion detection according to the Concept Learning from Cases based on Semantic Distance for Knowledge Acquisition process. To keep the system growing and being useful for new intrusion behavior, a knowledge discovery system, Intrusion Detection Knowledge Mining System, which follows the knowledge discovery process proposed in this work is used to extract new knowledge from user behaviors. For the newly

discovered knowledge and existing knowledge bases of other intrusion detection systems, the Intrusion Detection Knowledge Bases Fusion System will try to merge all knowledge and make it consistent. The fused knowledge base can be used for the intrusion detection engine to find new intrusions by improving the capability of the intrusion detection system; hence, the system can be improved and adapted to the new intrusions or challenges.

The following figure shows the detailed flow and relations between sub-systems of the NID-ES:

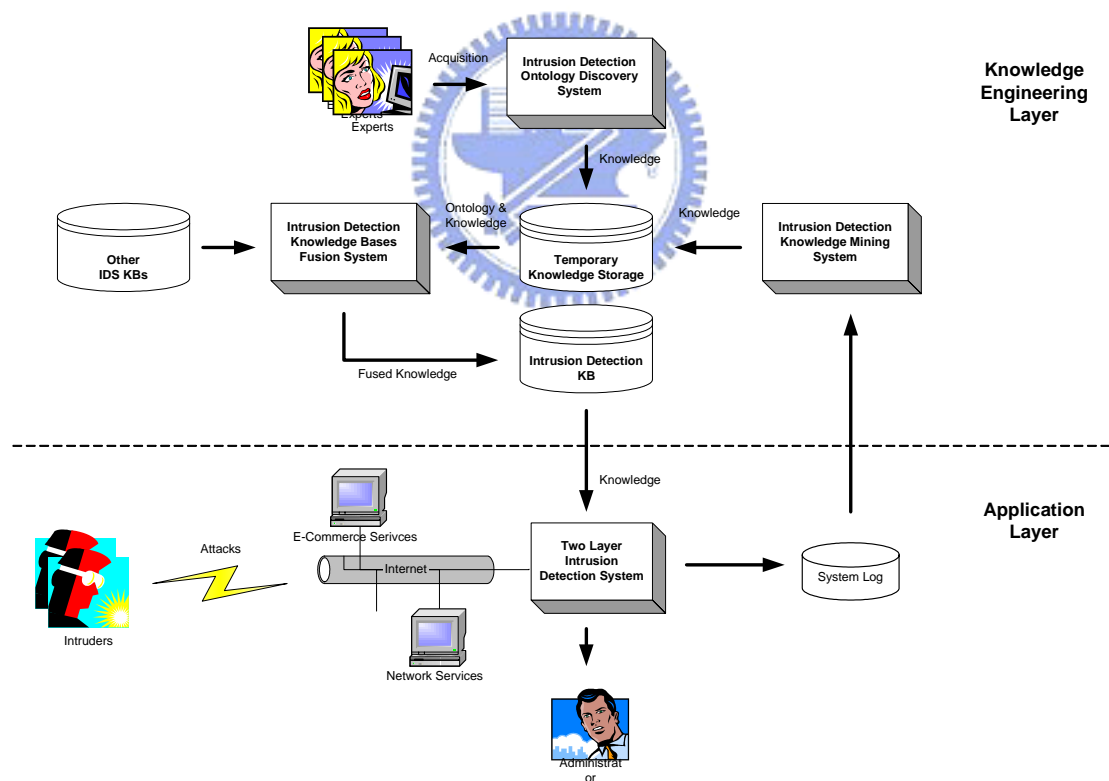


Figure 9.2: The architecture of NID-ES

In NID-ES, Two-Layer Intrusion Detection System is used to monitor the network behavior and find suspected intrusion behaviors. On the other hand, Intrusion

Detection Knowledge Acquisition System is used to extract the knowledge about network intrusion behavior from domain experts, all the knowledge acquired will be saved for other systems to use. At the same time, the Intrusion Detection Knowledge Mining System will use the system log of Two-Layer Intrusion Detection System to find embedded and interesting intrusion behavior patterns, and translate the patterns into rule formatted knowledge. After all, the Intrusion Detection Knowledge Bases Fusion System will use the knowledge from multiple sources, including the knowledge extracted from expert, the knowledge mined, and the knowledge of other intrusion detection knowledge bases, and merge them with meaningful structure for the intrusion detection engine to use. In the following sections, these sub-systems will be introduced detailedly.

9.2 Knowledge Representation and Detection Engine Design in NID-ES



In NID-ES, a Two-Layer Network Intrusion Detection System is proposed for processing the intrusion detection knowledge to monitor the network and detect intrusions, which combines the high efficiency for network activity monitoring of general IDS and the accuracy for knowledge expression of rule base system. In this system, network behaviors will be monitored and detected in different levels, including fundamental network connection layer and customized application layer. The following figure shows the architecture of Two-Layer Network Intrusion Detection System.

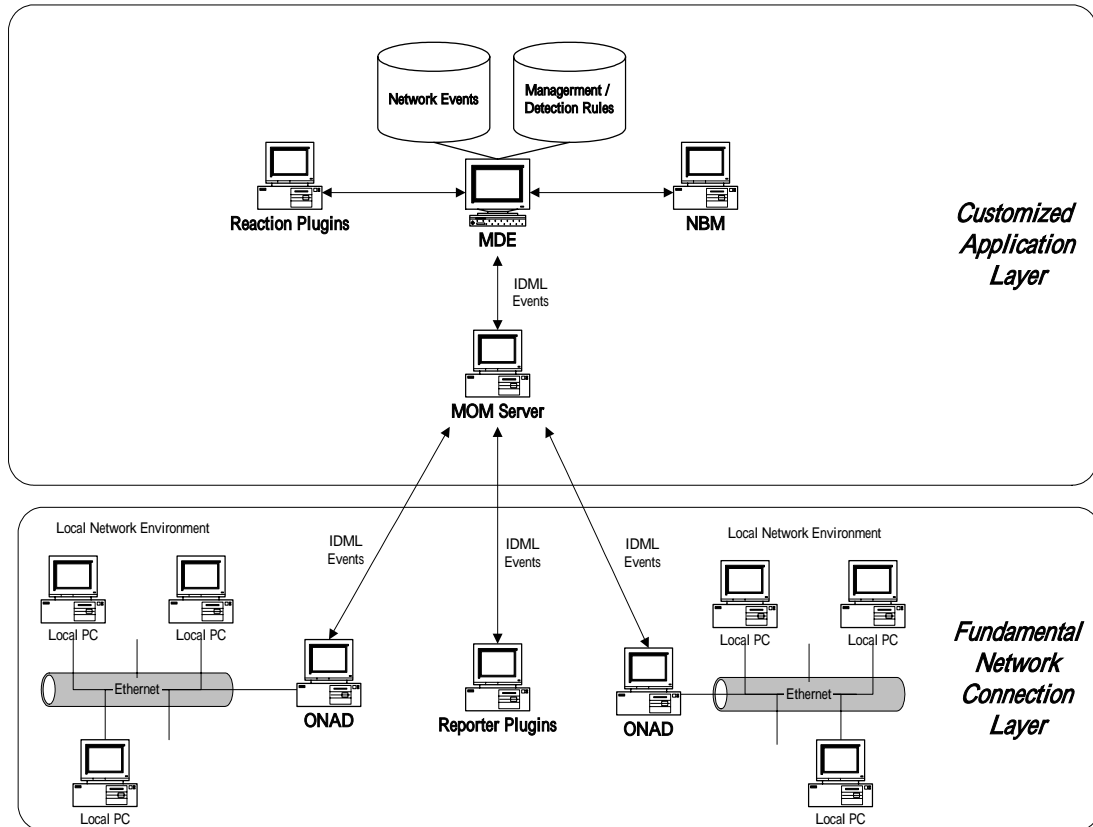
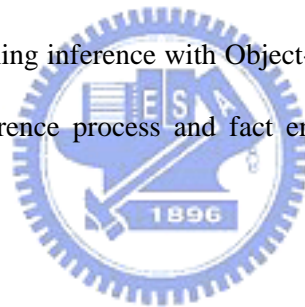


Figure 9.3: The architecture of Two-Layer Intrusion Detection System

For general and simple intrusion behaviors, Online Network Analyzer/Detector (ONAD) of the proposed model is used for signature based intrusion detection process, which is widely used for online intrusion detection. On the other hand, in order to detect more complicated intrusion behaviors, the alarm events and abstract network information generated from ONAD are further analyzed by Meta Detection Engine (MDE) using Rule Base technology, and logical expressions are used to express intrusions with complicated behavior pattern. Based on the concept, a Two-Layer Network Intrusion Detection System, including Fundamental Network Connection Layer and Customized Application Layer, is proposed and implemented. In the first Layer, ONAD is responsible for real-time detect intrusions in huge amount of network connection. And in the second Layer, MDE receives and analyzes events reported from ONAD and other applications to discover possible complicated intrusions using Rule Base inference technology.

In order to support intrusion detection and management monitoring, an Inference Engine is used in MDE. According to the intrusion detection rules and the management rules given by system manager or experts, the inference engine will use the facts from Fact Manager to trigger the rule inference and detect possible intrusions or management events. Since a rule based inference engine is used, more complicated rule chains are supported for experts to represent more complicated intrusion behaviors. Since logical expressions can be used to represent intrusion patterns or behavior patterns in MDE, more complicated intrusions and behaviors can be detected by MDE.

The NORM inference engine, DRAMA, is used as the Inference Engine of MDE, which can efficiently support forward chaining inference with Object-based inference mechanism. Since DRAMA supports forward inference process and fact encapsulation, MDE can efficiently manage the inference process.

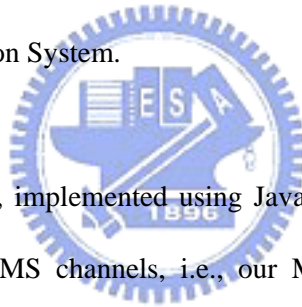


For intrusions detection system as MDE, performance is one of the important properties to be concerned. DRAMA is designed to have good performance in rule inference, which means MDE will be able to infer the rules and detect intrusions efficiently. Some modifications are made on DRAMA inference engine, including Facts retrieving and managing mechanism. On the other hand, instead of managing the facts using original fact management mechanism, Facts Manager in this system is used to index and retrieve necessary facts for the modified DRAMA inference engine.

According to the framework proposed, a prototype of Two-Layer Network Intrusion Detection System is implemented. In the prototype, the Meta Detection Engine is implemented as a centralized server for receiving events sent from client ONAD agents. The

ONAD is an enhanced version of IDML detection engine [17], to provide basic detection capability of detecting possible intrusions in a local area network based on the packet level information extracted from network data. When ONAD performs detection process on each local area network, the alerts, and extracted information of ONAD will be sent to MDE server.

However, when the ONAD sending information to MDE server, network handshaking and delaying may degrade the performance of this system. In order to enhance the performance of our prototype, MOM is used here for delivering information between MDE and ONAD. In this prototype, OpenJMS, which follows the JMS standard [30], is used as the MOM system. With OpenJMS, network traffic usage and system performance can be obviously enhanced for our Two-Layer Intrusion Detection System.



The MDE server of this system, implemented using Java Language, is designed to receive IDML events from UDP and JMS channels, i.e., our MDE server can not only receive information from JMS server, but also receive information from other applications which report IDML events using UDP datagram. OORB inference engine, which is also implemented in Java, is used in MDE for detect complicated intrusions. Also, several kinds of charts to show the network information are also implemented..

As we mentioned before, any application with the ability of sending IDML events to MDE server can be treated as the Reporter plugin for our system. In our implemented system, an SNMP information collector is designed and implemented to poll information from SNMP hardware or software and send to MDE server. According to the network address settings and OID information, SNMP collector will retrieve corresponding information from SNMP device and send the information in IDML event format to MDE server. With this SNMP collector,

SNMP information can be used as the source of events for MDE to detect, and enhance the detection ability of this prototype.

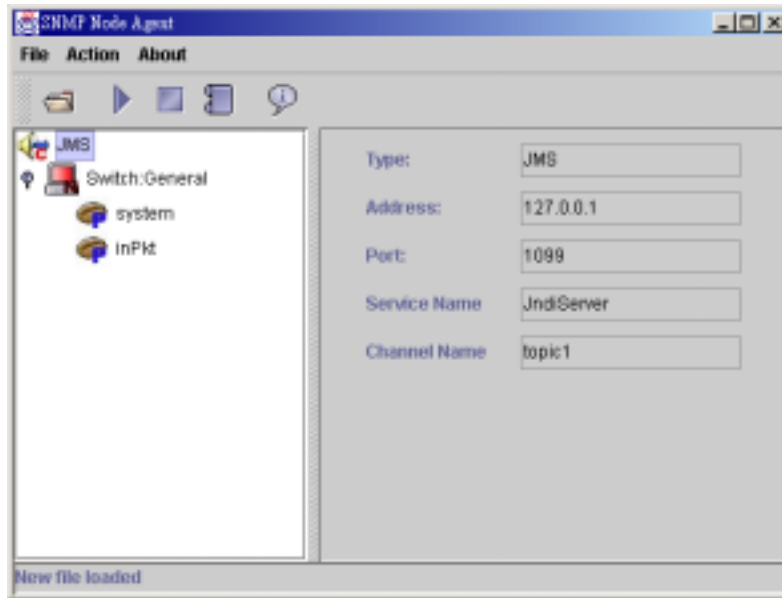


Figure 9.4: MDE Server. The left part shows the events received. The right part shows the configuration of the server.

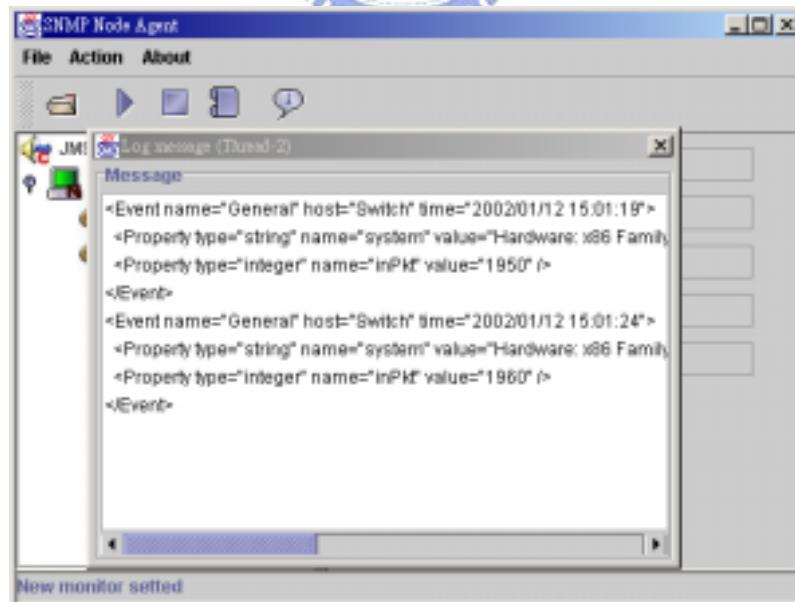


Figure 9.5: Received IDML event.

Some local network behaviors, which are presumed normal, may be intrusions after some signatures are detected. An example is IP spoofing attacks, which first denies the service of a client A and then spoofs A to connect server B. In this sections, detection model for TFN is described as follows.

TFN (Tribal Flood Network), a distributed denial of service attack, consists of an intrusion master (server) and zombie ants (clients). Unlike some specific intrusion detection tool for TFN, in our system, this intrusion can be modeled without modifying system kernel. In the experiment, the TFN attacking master is at 210.1.2.3, the TFN clients are at 140.113.87.101~105, and the victim is at 140.113.87.25.

The detection model consists of three steps. First, ONAD detects local signatures, which may be a TFN attack. Second, ONAD reports to MDE via MOM. Finally, MDE collects information from all local area network, confirms the intrusion and identifies the source IP of the TFN attacking master. The ONAD patterns, IDML events and MDE rules of each step are described as follows:

1. Patterns of ONAD to detect the local signatures about TFN (four rules only):

```
(Probe)
Pattern  IcmpTypeValue=8
        and ContentInclude="1234"
Alert    "DdosTfnProbe"

(BE)
Pattern  IcmpTypeValue=0
        and IcmpEchoId=456
        and IcmpEchoSeq=0
Alert    "DdosTfnClientCommandBE"

(LE)
Pattern  IcmpTypeValue=0
        and IcmpEchoId=51021
        and IcmpEchoSeq=0
Alert    "DdosTfnClientCommandLE"

(SR)
Pattern  IcmpTypeValue=0
        and IcmpEchoId=123
        and IcmpEchoSeq=0
        and ContentInclude="73 68 65 6C 6C 20 62 6F 75 6E 64 20 74
```


6F 20 70 6F 72 74"
Alert "DdosTfnServerResponse"

2. IDML event message sent to MDE via MOM (one example only):

```
<?xml version="1.0"?>  
<Event>  
  <Time>20020701125634</Time>  
  <Name>DdosTfnProbe</Name>  
  <Attribute>  
    <Name>SourceIp</Name>  
    <Value>140.113.87.101</Value>  
  </Attribute>  
  <Attribute>  
    <Name>DestinationIp</Name>  
    <Value>140.113.87.25</Value>  
  </Attribute>  
</Event>
```

3. Rules of MDE to detect and identify the TFN attacking master:

If $\exists x, y, z, \text{LargerThen}(\text{ProbeCount}(x, y), 1000)$
and $\text{BE}(x, z)$
and $\text{LE}(x, z)$
and $\text{SR}(z, x)$
Then $\text{Response}(\text{"TFN Attack, master IP:"} + z)$

Where x, y, z denote the sources of network behaviors. $\text{ProbeCount}(a, b)$ is the number of the "Probe" events with the source a and destination b . The relations of BE, LE, and SR are detected by the information sent from ONAD via MOM.

9.3 Knowledge Acquisition in NID-ES

Since WordNet has well-defined dictionary structure and open format structure, and WordNet provides concept hierarchy of vocabularies for general purpose terminologies, we use WordNet in our implementation for calculate the similarity between keywords. However, there are many different hierarchy relations defined in WordNet, and only following relations will be used:

Table 2. WordNet relations

Relation	Meaning
ANTONYM	Opposite-of relation (wet-dry), equal
CAUSE	Cause relation, related
ENTAILED_BY	Be-entailed (sleeping is entailed by snoring) relation
HYPERNYM	Is-A-kind-of relation, generalization
MEMBER_HOLONYM	A-member-of relation, generalization
PART_HOLONYM	A-part-of relation, generalization
PARTICIPLE_OF	Be-pertained relation, generalization
SIMILAR_TO	Similar-to relation, equal
SUBSTANCE_HOLONYM	A-part-of relation(in substance), generalization

When calculating the distance between keywords, we will use the shortest path of the keywords among all these relations. With the help of WordNet, we implement a general purpose *Knowledge Feature Clustering System*, in which the WordNet dictionary has been accessed via corresponding Application Programming Interface to extract the concept hierarchy from WordNet. Table 3 shows the detail developing environment settings: The architecture of the prototype system is shown in Figure 2.

Table 3. The related information of implementation

Attribute	Description	Value
Operating System	The OS to develop and execute the prototype system.	Platform Independent
Programming Language	The programming language used to develop the prototype system.	Java language with JDK 1.4.1
Word Relationship	The word relation used to calculate the similarity between words.	WordNet 1.7.1
Programming Library	The programming library used to access the word relationship and corresponding dictionary file.	JWNL (Jave WordNet Library)
KA Approach	The Knowledge Acquisition approach.	Two-Phase KA approach
KA Tool	The tool used as the interface to extract knowledge from expert.	Knowledge Extractor in DRAMA version 2.5 [DRA04]

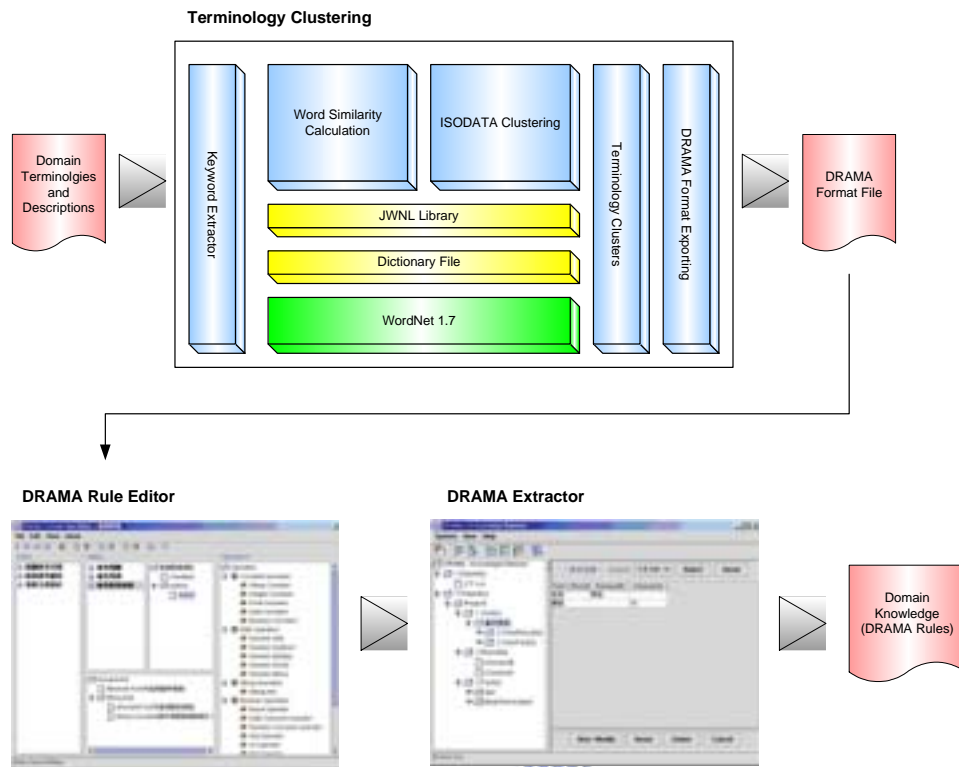


Figure 9.6: The architecture of prototype system

In this architecture, the domain cases and descriptions collected will first be loaded by Keyword Extractor module and corresponding keywords for these cases will be extracted. Since then, the similarity between these cases will be calculated by the Similarity Calculation module, while the JWNL library is used in this module to access the dictionary file of WordNet 1.7 to access the word relationships. ISO-DATA clustering algorithm is implemented here to cluster the cases into meaningful case clusters, which will be formatted as DRAMA format knowledge file. We can then use DRAMA rule editor to remove redundancy of cases, and finally use DRAMA extractor, which is a Repertory Grid application used to acquire knowledge from expert, to extract the knowledge of these knowledge concepts. Figure 3 is some screen shots of implemented knowledge clustering program:

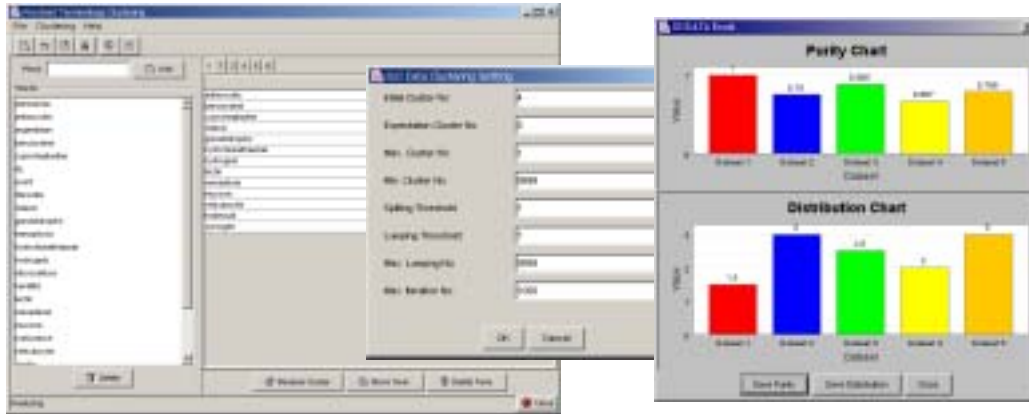


Figure 9.7: Some screen shots of prototype system

To improve the capability of this prototype system for specific domain, a concept hierarchy of Denial of Service (DoS), which is a type of network intrusions, is also included in the similarity calculation to get more accurate result for the domain of DoS. The concept hierarchy used is shown as following figure, which is summarized from existing DoS intrusions:

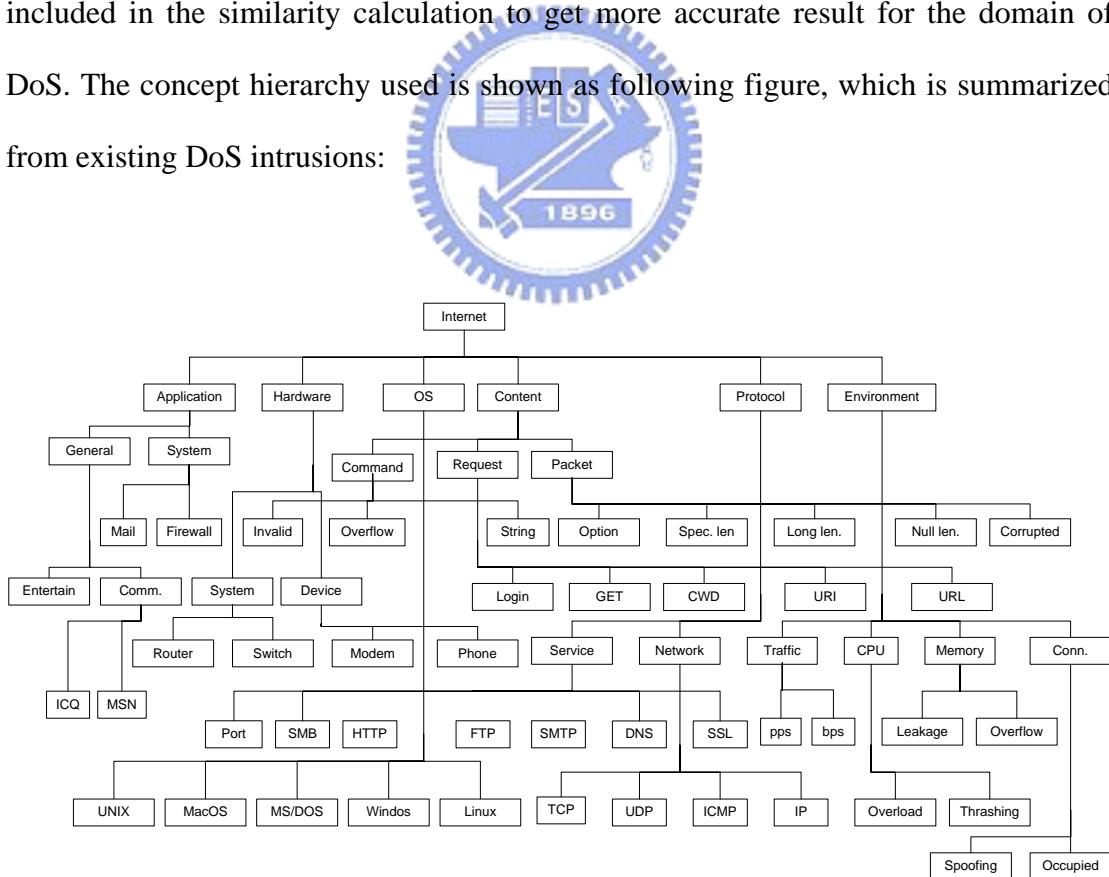


Figure 9.8: The concept hierarchy of DoS

After the terminologies are clustered into different rule class, DRAMA extractor, which is part of DRAMA rule base product [DRA03], is used here as the tool to acquire knowledge from expert. The clustering result of our algorithm will be translated into DRAMA format, and DRAMA rule editor is then used to review the content of each clustering and edit the knowledge cluster. After the knowledge cluster is adjusted, DRAMA knowledge extractor is further used to design the grids for extracting the relationships between concepts and retrieving the knowledge content of each concept. Figure 9.9 and Figure 9.10 show some screenshots of DRAMA utilities.

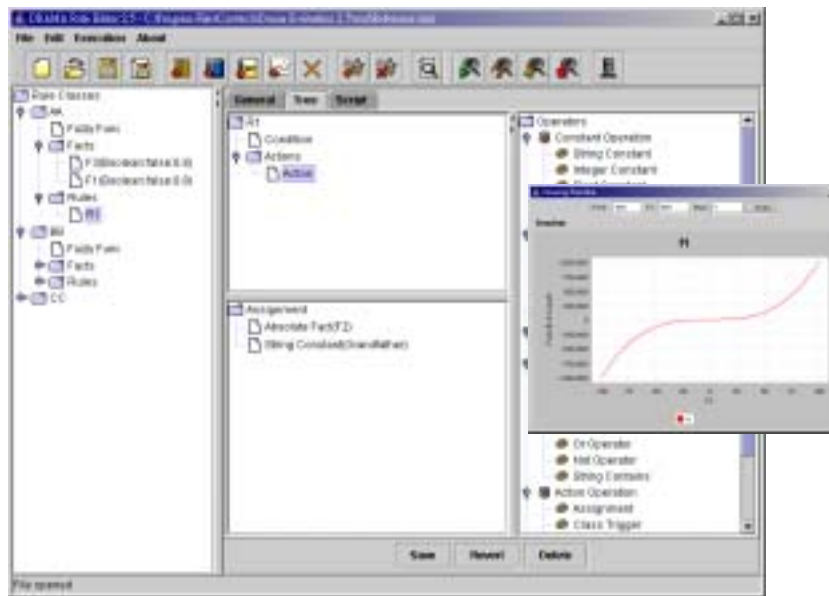


Figure 9.9: DRAMA editor

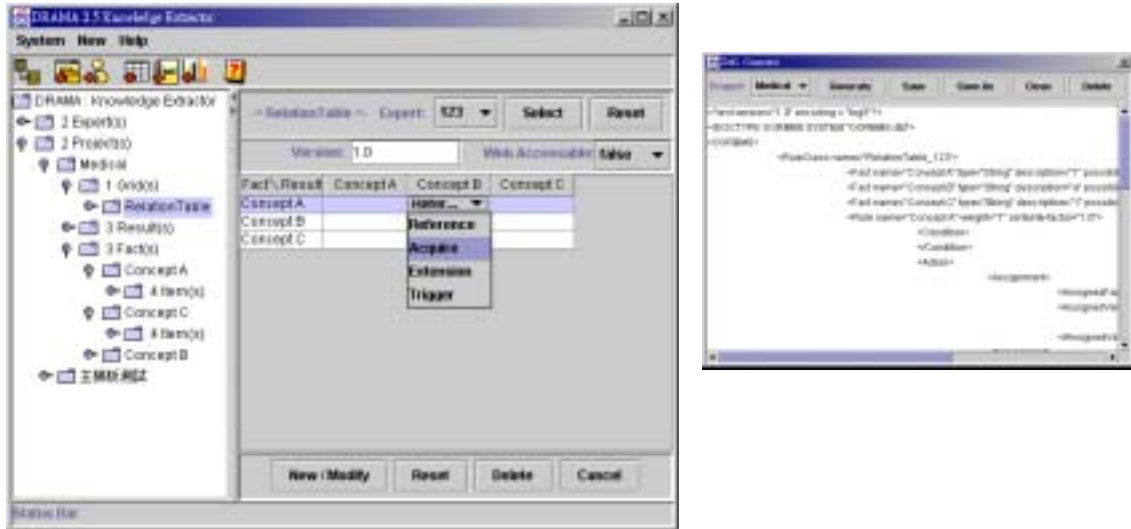
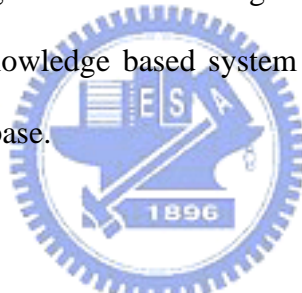


Figure 9.10: Using DRAMA extractor to extract knowledge concept relations

Since then, the knowledge contained in the grid can be extracted and translated into DRAMA rules, and a knowledge based system can be finally implemented by integrating the DRAMA rule base.



9.4 Knowledge Discovery in NID-ES

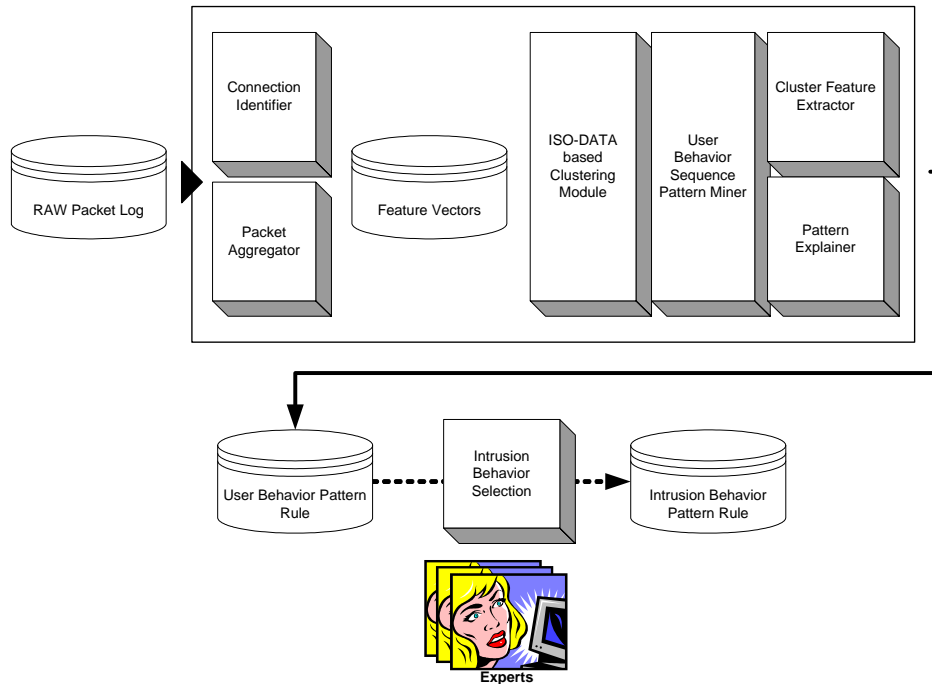


Figure 9.11: The architecture of Intrusion Detection Knowledge Mining System

For discovering embedded knowledge in user's (intruder's) behavior, the Intrusion Detection Knowledge Mining system is designed. Figure 9.11 shows the architecture of Intrusion Detection Knowledge Mining System. In this system, the raw packet information collected from IDS engine is used as the input. The system will first try to group the packets of the same connection according to network protocol. After the packets are grouped, the Packet Aggregator will summarize each connection into feature vector. The following shows the format of a network packet:

Time	SIP	DIP	DPort	SPort	Protocol	Flag	Length	...
------	-----	-----	-------	-------	----------	------	--------	-----

And for each connection, the Packet Aggregator will summarize the packets information into a feature vector as follows:

Time	Duration	SIP	DIP	DPort	SPort	Protocol	Flag	Traffic	Packet No.	...
------	----------	-----	-----	-------	-------	----------	------	---------	------------	-----

KDDCUP 1999 [KDD99], a data mining test for network intrusion detection, proposed a set of features for representing network connections. The features used in KDDCUP 1999 are referred and used to construct our feature vectors here. The following shows the features used in KDDCUP 1999 [KDD99]:

Table 9.1: KDDCUP selected features

Attribute	Data type
duration	Continuous
protocol_type	Symbolic
service	Symbolic
flag	Symbolic
src_bytes	Continuous
dst_bytes	Continuous
land	Symbolic
wrong_fragment	Continuous
urgent	Continuous
hot	continuous
num_failed_logins	continuous
logged_in	symbolic
num_compromised	continuous
root_shell	continuous
su_attempted	continuous
num_root	continuous
num_file_creations	continuous
num_shells	continuous
num_access_files	continuous
num_outbound_cmds	continuous
is_host_login	symbolic
is_guest_login	symbolic
count	continuous
srv_count	continuous
error_rate	continuous
srv_error_rate	continuous
error_rate	continuous
srv_error_rate	continuous
same_srv_rate	continuous
diff_srv_rate	continuous
srv_diff_host_rate	continuous
dst_host_count	continuous
dst_host_srv_count	continuous
dst_host_same_srv_rate	continuous
dst_host_diff_srv_rate	continuous
dst_host_same_src_port_rate	continuous
dst_host_srv_diff_host_rate	continuous
dst_host_error_rate	continuous
dst_host_srv_error_rate	continuous
dst_host_error_rate	continuous
dst_host_srv_error_rate	continuous

As the connections summarized as the feature vectors, the ISO-DATA Clustering Module will perform ISO data clustering to these vectors as proposed in our algorithm. Then User Behavior Sequence Pattern Miner will label the clusters generated in previous module, and translate users' behaviors into sequence of cluster labels. Sequential pattern mining algorithm will be applied to these user behavior sequences to find the patterns of user behavior. In order to explain the meaning of these user behavior sequences, the significant features of each cluster generated in ISO-DATA Clustering Module will be extracted using information theorem, which is done by Cluster Feature Extractor. After that each step of user behavior sequence can be explained by Pattern Explainer, which means each step corresponds to a set of features, e.g., ICMP # > 1000, and hence the meaning of each user behavior sequence can be translated into chaining rules.



However, not all the patterns discovered is an intrusion, many of them may be normal user behaviors or known intrusion behaviors. Except compare to original rules in the knowledge base, experts will be consulted to identify whether each pattern is useful or meaningless. The Intrusion Behavior Selection Process is the process for expert to select useful and meaningful intrusion behaviors. After all, in the following Knowledge Fusion process of the NID-ES will then fuse those meaningful patterns selected by experts into existing knowledge base.

9.5 Knowledge Fusion in NID-ES

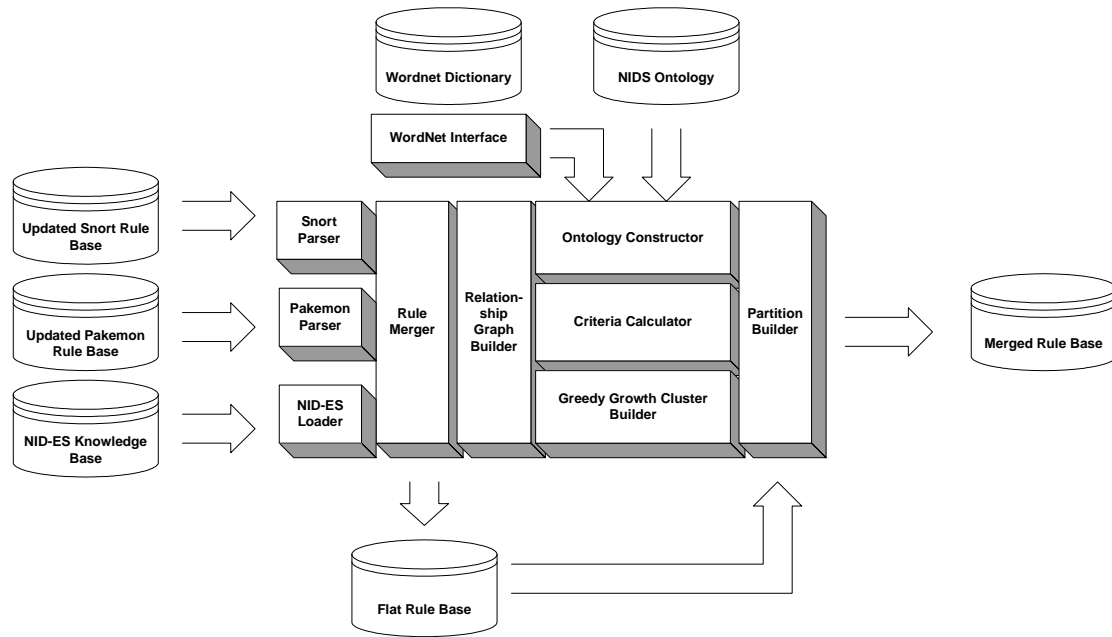


Figure 9.12: The architecture of Intrusion Detection Knowledge Bases Fusion System

In the Intrusion Detection Knowledge Bases Fusion System, several knowledge bases will be fused according to the knowledge fusion framework proposed. For each knowledge base, the corresponding parser / loader are designed to retrieve knowledge from knowledge base and translated into internal uniform format. After comparing several different knowledge base for intrusion detection, the following simplified format of intrusion detection rule is used:

If *<protocol>* *<src_ip>* *<src_port>* *<dst_port>* *<dst_ip>* *<content>*
Then *<intru_name>* *<intru_type>*

All these formatted rules will be gathered by Rule Merger, and stored into a temporary storage, which stored all rules in flat structure (no rule class / partition defined). After that the Relationship Graph Builder will be used to construct the relationship graphs between all rules, which will be used in forthcoming components

to do partitioning and clustering for these rules.

In the knowledge fusion process we proposed, a dictionary / concept hierarchy information will be helpful for calculating the semantic distances between rules; in Intrusion Detection Knowledge Bases Fusion System, not only a general purpose dictionary, WordNet, is used, but also a domain specific ontology [LT04] is also referred and used here. The domain ontology used is shown as figure 9.8.

Since WordNet is a general purpose dictionary and hence many specific domain terminology, e.g., DDoS, is not included in the dictionary, and hence make the semantic distance calculation to be less accurate. Including a domain specific dictionary / concept hierarchy as above is helpful for improve the usability and accuracy of this system. The dictionaries will be loaded and merged by Ontology Constructor, and then Criteria Calculator will use the ontology together with the relationship graph constructed in previous module to calculate the three criteria for determine the clustering result. Greedy Growth Cluster Builder is used to grow the rule clusters by the greedy growth algorithm defined. Finally, in the Partition Builder, the rules temporarily stored in the flat rule base will be partitioned into rule classes of NORM model, and hence the expert can use NORM utilities to review / edit / modify the knowledge base of fused knowledge base.

Followings are some screenshots of the prototype system for rule base partitioning. The system provide interface for loading multiple formats of rules, including Snort rules, Pakemon rules, and NORM DRAMA formatted rules. All these rules will be translated into the same format. After that the implemented system will cluster the rules according to WordNet (access through JWNL library) and user customized

dictionary. The cluster result can be exported as NORM DRAMA rule format, and also rule classes will be defined according to the clustering result; the outputted result can be loaded into DRAMA rule base and related utilities, users can use DRAMA utilities to modify / edit the result, and finally provide the result to Two-Layer Intrusion Detection System for monitoring the network.

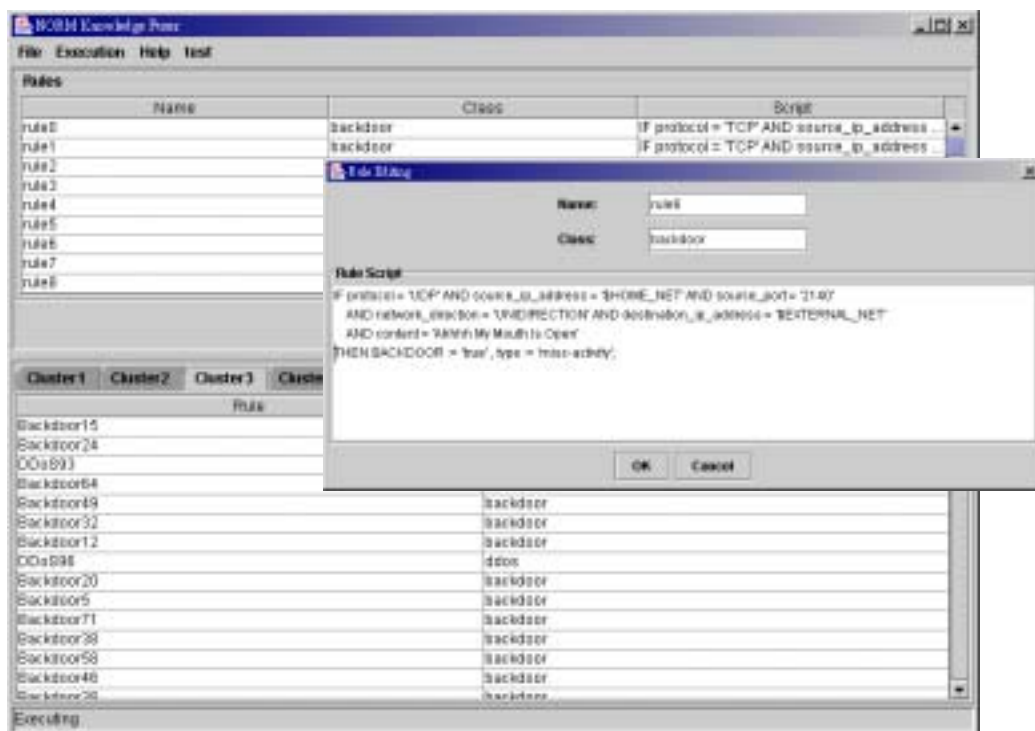


Figure 9.13: The screenshots of the prototype system

9.6 Discuss of NID-ES

In this chapter, a Network Intrusion Detection Expert System is proposed following NORBP architecture, and the systems, including Two-Layer Network Intrusion Detection System, Intrusion Detection Knowledge Acquisition System, Intrusion Detection Knowledge Mining System, and Intrusion Detection Knowledge Bases

Fusion System, for the four phases of KM lifecycle defined in NORBP are designed. Two-Layer Network Intrusion Detection is used to monitor network behaviors and detect intrusion behaviors according to the NORM knowledge base with the help of DRAMA rule base engine. Intrusion Detection Knowledge Acquisition System help to acquire the expert knowledge about intrusions and hence provide the knowledge base for entire expert system to be useful for detecting intrusion. Intrusion Detection Knowledge Mining System is designed to help extract the knowledge embedded in the users daily behaviors and intruders behaviors is also included, and hence we can obtain the knowledge about new behaviors or new intrusions without repeat the knowledge acquisition process and reduce the effort to make the system updated to new intrusions. After all, in order to maintain the knowledge structure, which is most meaningful in NORM knowledge model, Intrusion Detection Knowledge Bases Fusion System provide the mechanism to help manage the knowledge base by fuse all different knowledge sources and obtain the knowledge structure meaningful for system administrator to manage the system. With these mechanisms of NORBP, the expert system built can be evolutionary maintained and developed without modify the infrastructure of this expert system to be adaptive to growing knowledge and applications. And also, by using the implementations of NORM mechanisms, NID-ES can be realized with lower effort to implement entire expert system, which can be a very difficult task for building an expert system.

Chapter 10 Conclusion

In this thesis, a New Object-oriented Rule Base Platform (NORBP) was proposed, which was designed to provide more flexible, efficient, maintainable, and meaningful knowledge representation, and also correspondingly knowledge systems mechanisms. According to the lifecycle defined in this work, several mechanisms were designed to construct a complete knowledge platform, including the mechanisms for knowledge representation, knowledge acquisition, knowledge discovery, and knowledge fusion. In NORBP, the New Object-oriented Rule Model (NORM) was designed to represent knowledge according to Object-oriented concept, and knowledge relations were defined to construct the knowledge model. In order to acquire knowledge from experts in NORBP, Concept Learning from Cases based on Semantic Distance for Knowledge Acquisition was proposed base on NORM concepts. Moreover, for the knowledge embedded in users daily behaviors, Knowledge Discovery mechanism were used for extracting knowledge from huge amount of massive data. Newly discovered knowledge in Knowledge Discovery mechanism might be redundant or conflict to existing knowledge, and Knowledge Fusion mechanism in NORBP was proposed to fuse different knowledge sources for the same knowledge domain, resolve the conflict and redundant of knowledge, and reconstruct the knowledge model in more meaningful structure.

The mechanisms of NORBP are implemented and corresponding experiments were designed and done. The experiments showed that the algorithms and mechanisms designed in this work are useful for knowledge management. Moreover, two expert systems, including a Computer Assisted Learning Expert System (CAL-ES) and a

Network Intrusion Detection Expert System (NID-ES) were designed and proposed as case studies for implementing expert system using NORBP. In the CAL-ES proposed, knowledge about how to selection appropriate learning materials, which is usually so called an Adaptive Learning issue, was organized as NORM knowledge model, and the inference of these knowledge were also handled by a NORM rule base system – DRAMA, which is a production system implemented according to NORM knowledge model.

In NID-ES, the corresponding systems for complete lifecycle defined in NORBP were designed and implemented. A Two-Layer Network Intrusion Detection System was designed to detect the possible intrusion behaviors on the network, in which the rules for intrusion detection was represented in NORM knowledge model. We also designed an Intrusion Detection Knowledge Acquisition System based on the knowledge acquisition mechanism in NORBP, with WordNet and DDoS concept hierarchy to calculate the similarities of domain terminologies. According to the network features proposed in KDDCUP 1999, the feature vector for Knowledge Discovery in NORBP was defined, and hence the data mining algorithms designed in Intrusion Detection Knowledge Mining System could be applied for discovering user and intruder behavior patterns, and translated the patterns into rules. Finally, the DDoS concept hierarchy used in Knowledge Acquisition mechanism was also used in Intrusion Detection Knowledge Bases Fusion System to calculate the semantic criteria between rules and hence built the rule classes between the knowledge to be fused.

In the future, we will improve NORBP by improving each mechanism respectively. For NORM knowledge model, we would like to design a backward inference mechanism and corresponding algorithm, to deal with the knowledge relations

defined in NORM knowledge model, which is not a part of existing backward inference mechanism and make the mechanism designation to be more complicated. Also, knowledge validation and verification in NORM knowledge model is also part of our plan to improve then usability of NORM. Currently, the Knowledge Acquisition mechanism we proposed still required some expert effort to define the NORM knowledge relations between concepts, and that can be an issue during the knowledge acquisition process; a methodology help generating the knowledge relations from the relations between the features in different concepts will be useful to provide at least a semi-automatic mechanism to reduce experts effort. Currently in our Knowledge Discovery mechanism, ISODATA clustering algorithm is used to cluster the feature vector, but since we have to generate the features of each cluster, that means we have to consider to make the resulting cluster more significant to each other; a specific clustering algorithm can be designed for our Knowledge Discovery process; Other data mining and machine learning algorithm, e.g., association rules, decision tree, can be also used for the Knowledge Discovery process. Regarding the knowledge fusion mechanism, although we have proposed algorithms for knowledge fusion, the time and space complexity are still high. Now, we are trying to improve the performance of the algorithms by developing some analysis on the characteristics of the knowledge to derive the weight for structural and semantic criteria. On the other hand, construction the shared vocabulary dictionary is still a difficult task for the domain experts. The well-developed vocabulary dictionary like WordNet can be applied to help improve our algorithm in the future.

Also, for the NID-ES proposed, the fully implementation and experiment are planed to be done based on the utilities developed for each mechanism in the near future. For the implementation, the concept hierarchy for intrusion related concepts will be first

defined according to previous researches, which will help to analyze the signatures should be obtained from network traffic for all different kind of intrusions. Based on the concept hierarchy of intrusion related concepts and signatures designed for various types of intrusions, the ontology construction process in Knowledge Acquisition process can be more accurate, and also for Knowledge Discover process, the feature vector constructed will be more meaningful since it represents the user behavior features required for detecting an intrusion. And also for Knowledge Fusion phase, the concept hierarchy provide good information for calculating the semantic criterion. Moreover, experiments based on existing intrusion detection rule bases have been done to show the performance improvement after the rule bases are fused and partitioned, so far the experiment results show that the performance of the rule base partitioned have great performance improvement than original rule bases without rule partitioning.



Reference

- [ACM00] ACM, “KDD cup 1999 data,” <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>, 2000.
- [AND95] Anderson, J. R., “*Cognitive Psychology and its Implications*,” New York: W.H. Freeman and Company, 1995
- [AS+03] Alani, H., Sanghee, Kim, Millard, D.E., Weal, M.J., Hall, W., Lewis, P.H., and Shadbolt, N.R., “Automatic ontology-based knowledge extraction from Web documents,” *Intelligent Systems, IEEE*, Volume: 18 Issue: 1 , Jan.-Feb. 2003, pp. 14 –21
- [AS95] Agrawal, R. and Srikant, R, ”Mining sequential patterns,” in Proc. of 7th IEEE International Conference on Data Engineering, pp. 3-14, 1995.
- [BS99] Beishuizen, J. J. and Stoutjesdijk, E. T., ”Study strategies in a computer assisted study environment.,” *Int'l Journal of Learning and Instruction*, Vol. 9, pp. 281-301, 1999
- [BH01] Budanitsky, A., Hirst, G., “Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures”, Workshop on WordNet and Other Lexical Resources, Pittsburgh, June 2001
- [BHJ+93] Behrendt, W, Hutchinson, E, Jeffrey, and KG, Macnee, CA, MD Wilson, "Using an Intelligent Agent to Mediate Multibase Information Access", CKBS-SIG, Keele, September 1993
- [BOO91] Booch, G., “Object-oriented Design with Applications, 2nd ed.,” San Francisco, CA: Benjamin/Cummings, 1991
- [BRA93] Branch, S. T., CLIPS Reference Manual: Volume I Basic Programming Guide, NASA, JSC-25012, 1993.
- [BS84] Buchanan, B.G. and Shortliffe, E.H., ”Rule-based Expert Systems,” London: Adison-Wesley, 1984
- [BTW01] Boley, H., Tabet, S., and Wagner, G., “Design Rationale of RuleML: A Markup Language for Semantic Web Rules”, Proc. SWWS'01, Stanford, July/August 2001.
- [CER01] CERT, <http://www.cert.org/>, 2001.

- [CL02] Choi, Byounggu and Lee, Heeseok, "Knowledge management strategy and its link to knowledge creation process," *Expert Systems with Applications*, Vol. 23 (3), pp. 173-187, 2002
- [CH02] Chung-Hong Lee, and Hsin-Chang Yang, "Text mining of multilingual corpora via computing semantic relatedness," *Systems, Man and Cybernetics*, 2002 IEEE International Conference on , Volume: 5 , 6-9 Oct. 2002, pp. 5 pp. vol.5
- [CHO96] Chou , C., "A computer logging method for collecting use-reported inputs during formative evaluation of computer network-assisted distance learning," *Proc. of ED-Media'96*, 1996
- [CLI98] CLIPS, "CLIPS Reference Manual Volume I Basic Programming Guide," Software technology branch, Lyndon B. Johnson Space Center, Version 6.10, 1998
- [CQ69] Collins, A.M. and Quillian, M.R., "Retrieval time from semantic memory," *Journal of Verbal Learning and Verbal Behavior*, 1969
- [CTL03] Chen, Chang-Sheng, Tseng, Shian-Shyong and Liu, Chien-Liang, "A unifying framework for intelligent DNS management," *International Journal of Human-Computer Studies*, Vol 58, Issue 4, April 2003, pp. 415-445
- [DD00] Dickerson, J. E. and Dickerson, J. A., "Fuzzy network profiling for intrusion detection," in *Proc. of Fuzzy Information Processing Society*, 2000.
- [DEE65] Deese, J., "The Structure of Associations in Language and Thought," Baltimore: the Johns Hopkins Press, 1965
- [DRA03] DRAMA, Coretech Inc, <http://www.ctknow.com.tw/rule.htm>, 2003
- [DWT03] Davidovic, A., Warren, J., and Trichina, E., "Learning benefits of structural example-based adaptive tutoring systems," *Education, IEEE Transactions on* , Volume: 46 Issue: 2 , May 2003, pp. 241 -251
- [EA02] Elst, Ludger van and Abecker, Andreas, "Ontologies for information management: balancing formality, stability, and sharing scope," *Expert Systems with Applications*, Vol. 23 (4), pp. 357-366, 2002
- [ERI03] Eriksson, H., "Using JessTab to integrate Protege and Jess," *Intelligent Systems, IEEE*, Volume: 18 Issue: 2 , March-April 2003, pp. 43 -50
- [FAS98] Fensel, D., Angele, J., and Studer, R., "The knowledge acquisition and representation language, KARL," *Knowledge and Data Engineering, IEEE Transactions on* , Volume: 10 Issue: 4 , July-Aug. 1998, pp. 527 -550

- [FFF99] Frank, G., Farquhar, A., and Fikes, R., "Building a large knowledge base from a structured source (KA and ontology)," *Intelligent Systems, IEEE* , Volume: 14 Issue: 1 , Jan.-Feb. 1999, pp. 47 –54
- [FIS87] Fisher, D., "Knowledge Acquisition via Incremental Conceptual Clustering", *Machine Learning*, 2, 139-172, 1987.
- [FK85] Fikes, R. and Kehler, T., "The role of frame-based representation in reasoning," *Communications of the ACM*, Vol.28, NO.9, pp.904-920, 1985
- [FS84] Feigenbaum, E.A., and Simon, H., "EPAM-like Models of Recognition and Learning", *Cognitive Science*, 8. 1984.
- [GAG85] Gagné, R.M., "The Conditions of Learning and Theory of Instruction," N.Y.: Holt, Rinehart & Winston, 1985
- [GAG84] Gagné, R.M., "Learning outcomes and their effects," *American Psychologist*, Vol.39, pp.377-385, 1984
- [GJZ99] G. Webb, J. Wells, and Z. Zheng, "An Experimental Evaluation of Integrating Machine Learning with Knowledge Acquisition," *Machine Learning*, vol. 35, no. 1, 1999, pp. 5–23.
- [GLF90] Gennari, J.H., Langley, P., and Fisher, D., "Models of Incremental Concept Formation", J. Carbonell, Ed., *Machine Learning: Paradigms and Methods*, Amsterdam, The Netherlands: MIT Press, 11-62, 1990.
- [GMA95] Godin, R., Missaoui, R., Alaoui, H., "Incremental concept formation algorithms based on Galois (concept) lattices", *Computational Intelligence*, 11(2), 246-267, 1995
- [GR89] Giarratano, J. and Riley, G., "Expert Systems: Principle and Programming," Boston: PWS Publishing Company, pp.63-102, pp.509-513, 1989
- [GRU03] Gruber, Tom, "What is ontology," <http://www-ksl.stanford.edu/kst/what-is-an-ontology.html>, 2003
- [GRU93] Gruber, Tom, "A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199-220", 1993.
- [GT93] Gruber, Tom, "Toward principles for the design of ontologies used for knowledge sharing," Presented at the Padua workshop on Formal Ontology, March 1993.
- [GWP88] Ginsberg, A. and Weiss, S. M. & Politakis, P., "Automatic knowledge base

refinement for classification systems,” *Artificial Intelligence*, Vol.35, NO.2, pp.197-226, 1998

[GLA87] Glaser, R., “Thoughts on Expertise, In Schooler, C. & Schaie, K. W. (Eds.) *Cognitive Functioning and Social Structure over the Life Course*,” Norwood, New Jersey: Ablex Publishing Corporation, pp.81-94, 1987

[HEN01] Hendler, J., “Agents and the Semantic Web,” *Intelligent Systems, IEEE* , Volume: 16 Issue: 2 , March-April 2001, pp. 30 –37

[HPH01] Harmelen, F. van, Patel-Schneider, P. F. and Horrocks, I. (editors), “The DAML+OIL language”, <http://www.daml.org/2001/03/reference.html>, 2001

[HS98] Hirst, G. and St-Onge, D., *Lexical chains as representations of context for the detection and correction of malapropisms*, pp. 305–332, Fellbaum, 1998.

[HT90] Hwang, G. J. and Tseng, S. S., ”EMCUD: A knowledge acquisition method which captures embedded meanings under uncertainty,” *Int’l Journal of Man Machine Studies*, Vol. 33, pp. 431-451, 1990

[HW03] Hirasawa, S., and Wesley W. Chu, “Knowledge acquisition from documents with both fixed and free formats,” *Systems, Man and Cybernetics*, 2003. *IEEE International Conference on* , Volume: 5 , Oct. 5-8, 2003, pp. 4694 -4699

[HY+02] Hongmei Yan, Yingtao Jiang, Jun Zheng, and Bingmei Fu, “Internet-based knowledge acquisition and management method to build large-scale medical expert systems,” *EMBS/BMES Conference, 2002. Proceedings of the Second Joint* , Volume: 3 , 23-26 Oct. 2002, pp. 1885 -1886 vol.3

[ILG93] Ilgun K., “USTAT: A real-time intrusion detection system for UNIX,” in *Proc. of the IEEE Symposium on Research on Security and Privacy*, Oakland, CA, May 1993.

[ILG95] Ilgun, K., Kemmerer, R.A., and P. A. Porras, “State transition analysis: A rule-based intrusion detection system,” *IEEE Transactions on Software Engineering*, 21(3), March 1995.

[JD88] Jain, A. K. and Dubes, R. C., “*Algorithms for Clustering Data*,” Prentice-Hill, Englewood Cliffs, N.J., 1988.

[JF90] Jacob, R.J.K. and Froscher, J.N., “A software engineering methodology for rule-based systems,” *IEEE Transactions on Knowledge and Data Engineering*, Vol.2, NO.2, 1990

- [JGC00] Jonyer, I., Holder, L.B., and Cook, D.J., "Graph-Based Hierarchical Conceptual Clustering", International Journal on Artificial Intelligence Tools, 2000
- [KDD99] KDD Cup, "<http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>", KDD Coup, 1999.
- [KEM97] Kemmerer, R. A., "NSTAT: A model-based real-time network intrusion detection system," Technical Report TRCS-97-18, Department of Computer Science, University of California, Santa Barbara, November 1997.
- [KIN01] Kingston, J., "High Performance Knowledge Bases: four approaches to knowledge acquisition, representation and reasoning for workaround planning," Expert Systems with Applications, Vol. 23 (4), pp. 181-190, 2001
- [KIN70] Kintsch, W., "Models for Free Recall and Recognition, In DA Norman (Ed.), Models of Human Memory," NY: Academic Press, 1970
- [KK98] Karypis, G. and Kumar, V.. "Multilevel Algorithms for Multi-constraint Graph Partitioning", Proceedings of Supercomputing '98, 1998
- [KLA71] Klausmeier, H. J., "Cognitive operations in concept learning," Educational Psychologies, 1971
- [KM03] knowledge and its use in a functional way server," Expert Systems with Applications, Vol. 24, Issue 2, February 2003, pp 153-166.
- [KO02] Kurosu, M., and Ookawa, Y., "Effects of negative information on acquiring procedural knowledge," Computers in Education, 2002. Proceedings. International Conference on , 3-6 Dec. 2002, pp. 1371 -1372 vol.2
- [KS98] Lee, W., and Stolfo, S. J., "Data mining approaches for intrusion detection," in Proc. of the 1998 USENIX Security Symposium, 1998.
- [LEB86] Lebowitz, M., "Concept Learning in a Rich Input Domain: Generalization Based Memory", R. Michalski, J. Carbonell, and T. Mitchell, Eds., Machine Learning: An A.I. Approach, Morgan Kaufmann, 193-214, 1986.
- [LO96] Lee, S. and O'Keefe, R.M., "The effect of knowledge representation schemes on maintainability of knowledge-based systems," IEEE Transactions on Knowledge and Data Engineering, Vol.8, NO.1, pp.173-178, 1996
- [LSM99] Lee, W., Stolfo, S. J., and Mok, K. W., "A data mining framework for building intrusion detection models," in Proc. of 1999 IEEE Symposium on Security and Privacy, May, 1999.

- [LT04] Lin, S.C., Tseng, S.S., “Constructing Detection Knowledge for DDoS Intrusion Tolerance,” submitted to Expert Systems With Applications.
- [LTT03] Lin, Yao Tsung, Tseng, S. S., and Tsai, Chi-Feng, "Design and implementation of new object-oriented rule base management system," Expert Systems with Applications, Volume 25, Issue 3, October 2003, pp.369-385
- [LTL01] Lin, Y. T., Tseng, S. S., and Lin, S. C., “An intrusion detection model based upon intrusion detection markup language (IDML),” Journal of Information Science and Engineering Vol. 17, No.6, 2001, pp. 899-919, 2001
- [LV+03] Lopez de Vergara, J.E., Villagra, V.A., Asensio, J.I., and Berrocal, J., “Ontologies: giving semantics to network management models,” Network, IEEE, Volume: 17 Issue: 3, May-June 2003, pp. 15
- [MAR01] Marty, R., “Snort – the open source network IDS,” <http://www.snort.org/>, 2001.
- [MBF+90] Miller, G.A., Beckwith, R., Fellbaum C., Gross, D., and Miller, K., “Introduction to WordNet: An On-line Lexical Database”, Journal of Lexicography, 1990
- [MCZ+00] Manganaris, S., Christensen, M., Zerkle, D., and Hermiz, K., “A data mining analysis of RTID alarms,” Computer Network 34 (2000), pp.571-577, 2000
- [MG95] Mineau, G.W., Godin, R., “Automatic Structuring of Knowledge Bases by Conceptual Clustering”, IEEE TKDE, 7(5), 824-828, 1995.
- [ML+03] Masuoka, R., Labrou, Y., Parsia, B., and Sirin, E., “The semantic web - Ontology-enabled pervasive computing applications,” Intelligent Systems, IEEE [see also IEEE Expert], Volume: 18 Issue: 5, Sept./Oct. 2003, pp. 68 –72
- [MS01] Maedche, A., and Staab, S., “Ontology learning for the Semantic Web,” Intelligent Systems, IEEE, Volume: 16 Issue: 2, March-April 2001, pp. 72 -79
- [MS03] McClure, S. and Scambray, J., "InfoWorld Security Sweet 16 (ISS16)," http://www.infoworld.com/cgi-bin/displayNew.pl?/security/links/security_iwss16.htm, 2003
- [MSD81] Michalski, R.S., Stepp, R., and Diday, E., "A Recent Advance in Data Analysis: Clustering Objects into Classes Characterized by Conjunctive Concepts", Kanal and Rosenfeld, Progress in Pattern Recognition, North-Holland, 1981.
- [NEG85] Negoita, C.V., “Expert Systems and Fuzzy Systems,” the

Benjamin/Cummings Publishing Corporation, pp.28, 1985

[NF+02] Nikiforou, S., Fink, E., Hall, L.O., Goldgof, D.B., and Krischer, J.P., Knowledge acquisition for clinical-trial selection,” Systems, Man and Cybernetics, 2002 IEEE International Conference on , Volume: 1 , 6-9 Oct. 2002, pp. 66 –71

[NP99] Neumann, P. and Porras, P. A., ”Experience with EMERALD to data,” in Proc. of 1st USENIX Workshop on Intrusion Detection and Network Monitoring, Santa Clara, California, April 1999, pp.73-80, 1999

[NS+01] Noy, N.F., Sintek, M., Decker, S., Crubezy, M., Ferguson, R.W., and Musen, M.A., “Creating Semantic Web contents with Protege-2000,” Intelligent Systems, IEEE, Volume: 16 Issue: 2 , March-April 2001, 60 –71

[PC01] Pal, N.R., and Chakraborty, S., “Fuzzy rule extraction from ID3-type decision trees for real data,” Systems, Man and Cybernetics, Part B, IEEE Transactions on , Volume: 31 Issue: 5 , Oct. 2001, pp. 745 -754

[PET77] Peterson, J.L., “Petri Nets”, ACM Computing Surveys (CSUR), v.9 n.3, 1977

[PHG+01] Preece, A., Hui, K., Gray, A., Marti, P., Bench-Capon, T., Cui, Z., and Jones, D.. “KRAFT: An Agent Architecture for Knowledge Fusion”, International Journal of Cooperative Information Systems, 10, 171-195, 2001

[PN99] Porras, P. A. and Neumann, P. G., “EMERALD: Event monitoring enabling responses to anomalous live disturbances,” <http://www2.csl.sri.com/emerald/concepts.html>, 1999.

[POR92] Porras, P., “STAT – A state transition analysis tool for intrusion detection,” Master’s thesis, Computer Science Department, University of California, Santa Barbara, June 1992.

[POR99] Porras, P. A., “Detecting computer and network misuse through the production-based expert system toolset (P-BEST),” in Proc. of the 1999 IEEE Symposium on Security and Privacy, Oakland, California, May 9-12,1999.

[RB91] Rumbaugh, J. and Blaha, M., et al., “Object-oriented Modeling and Design,” Prentice Hall, 1991

[REI91] Reichgelt, H., “Knowledge Representation, an AI Perspective,” Norwood, New Jersey: Ablex Publishing Corporation, 1991

[RH03] Rafea, Ahmed and Hassen, Hesham, “Automatic knowledge acquisition tool

for irrigation and fertilization expert systems,” Expert Systems with Applications, Vol 24, Issue 1, January 2003, pp 49-57.

[RIC99] Richards, K., “Network based intrusion detection: A review of technologies,” Computer and Security, Vol. 18, pp.671-682, 1999.

[RN95] Russell, S.J., Norvig, P., Artificial Intelligence: Modern Approach, Prentice Hall, 185-216, 1995.

[ROE88] Roesner, H., Expert systems for commercial use. Artificial Intelligence and Expert Systems, pp.35-59, 1988

[ROE99] Roesch, M., “Snort - Lightweight Intrusion Detection for Networks”, Proceedings of the USENIX LISA '99 Conference, Nov. 1999.

[RSC97] Ramaswamy, M., Sarkar, S., Member and Chen, Y.S., “Using Directed Hypergraphs to Verify Rule-Based Expert Systems”, IEEE TKDE, Vol.9, No.2, Mar-Apr, pp.221-237, 1997

[RW02] Rosca, Daniela and Wild, Chris, “Towards a flexible deployment of business rules,” Expert Systems with Applications, Vol. 23 (4), pp. 385-394, 2002

[SAL93] Salzgeber, M. J., et al, “Managing Uncertainty in CLIPS: A System Level Approach,” Proceedings of the 6th Florida Artificial Intelligence Research Symposium, Florida AI Research Society, 1993

[SB75] Shortliffe, E.H. and Buchanan, B.G., “A model of inexact reasoning in medicine,” Mathematical Biosciences, Vol.23, pp.351-379, 1975

[SBA04] Shamsfard, Mehrnoush and Barforoush, Ahmad Abdollahzadeh, “Learning ontologies from natural language texts,” International Journal of Human-Computer Studies, Vol 60, Issue 1, January 2004, pp 17-63.

[SCO03] Sharable Content Object Reference Model, Advanced Distributed Learning, <http://www.adlnet.org/index.cfm>, 2003

[SF02] Souza, M.A.F. de and Ferreira, M.A.G.V., “Designing reusable rule-based architectures with design patterns,” Expert Systems with Applications, Vol. 23 (4), pp. 395-403, 2002

[SG91] Shieh, S. W. and Gligor, V. D, “A pattern-oriented intrusion-detection model and its applications,” in Proc. of IEEE Computer Society Symposium on Research in Security and Privacy, pp 327 –342, 1991.

[SG97] Shieh, S. P. and Gligor, V. D., "On a pattern-oriented model for intrusion detection," in Proc. of IEEE Transactions on Knowledge and Data Engineering,, Volume 9, pp 661 -667, July-Aug. 1997.

[SM86] Stepp, R.E., and Michalski, R.S., "Conceptual Clustering: Inventing Goal-Oriented Classifications of StructuredObjects", R. Michalski, J. Carbonell, and T. Mitchell, Eds., Machine Learning: An A.I. Approach, Kaufmann, 1986.

[SOW00] Sowa, J.F., Knowledge Representation: Logical, Philosophical, and Computational Foundations, Brooks Cole Publishing Co., Pacific Grove, CA, 2000

[SOW84] Sowa, J.F., Conceptual Structures: Information Processing in Mind and Machine, Addison-Wesley, 1984.

[STT99] Su, G. H. and Tseng, S. S. and Tsai, C. J. and Zheng, J. R., "Building an object-oriented and individualized learning environment on the WWW," 7th International Conference on Computers in Education, 1999

[TAK02] Takeda, K., Packet Monster, <http://web.sfc.keio.ac.jp/~keiji/backup/ids/pakemon/index.html>, 2002

[TL89] Thompson, K., and Langley, P., "Incremental Concept Formation with Composite Objects", Proceedings of the 6th Int. Workshop on Machine Learning, Cornell University. Morgan Kaufmann, 1989.

[TL91] Thompson, K. and Langley, P., "Concept formation in structured domains", In D. H. Fisher and M. Pazzani (Eds.), Concept Formation: Knowledge and Experience in Unsupervised Learning, Chap. 5. Morgan Kaufmann Publishers, Inc. 127-161, 1991.

[TL+99] Tsai, J.J.P., Liu, A., Juan, E., and Sahay, A., "Knowledge-based software architectures: acquisition, specification, and verification," Knowledge and Data Engineering, IEEE Transactions on , Volume: 11 Issue: 1 , Jan.-Feb. 1999, pp. 187 -201

[TSA02] Tsai, C.F., "Design and Implementation of New Object-Oriented Rule Base Management System," Master Thesis, Department of Computer and Information Science, NCTU, 2002

[TT02] Tsai, Chang-Jiun and Tseng, S.S., "Building a CAL Expert System based upon Two-phase Knowledge Acquisition," Expert Systems with Applications, Vol. 22 (3), pp. 235-248, 2002

[TTW99] Tsai, C. J. and Tseng, S. S. and Wu, Y. C., "A new architecture of

objected-oriented rule base management system,” Proceeding of Int’l Conf. on TOOLS 31, pp. 200-203, Nanjing, China, 1999

[TUL83] Tulving, E., “Elements of Episodic Memory,” Oxford: Oxford university press, 1983

[TUL73] Tulving, E. and Thomson, D.M., “Encoding specificity and retrieval processes in episodic memory,” Psychological Review, Vol.80, pp.352-373, 1973

[VAR01]M. Vargas-Vera et al., “Knowledge Extraction Using an Ontology-Based Annotation Tool,” Workshop on Knowledge Markup & Semantic Annotation,ACM Press, New York, 2001, pp. 5–12.

[VK98] Vigna, G. and Kemmereer, R. A., “NetSTAT: A network-based intrusion detection approach,” in Proc. of IEEE Computer Security Applications Conference, pp25-34, 1998.

[VSV+01] Visser, U., Stuckenschmidt, H., Vögele, T. and Wache, H., “Enabling Technologies for Interoperability”, Transactions in GIS, 2001.

[WON98] Wong, C.H., GA-Based Knowledge Integration, Ph. D. Dissertation, Department of Computer and Information Science, National Chiao Tung University, 1998

[WOR03] Wordnet, “WordNet Homepage,” cognitive science laboratory, princeton university, <http://www.cogsci.princeton.edu/~wn/>, 2003

[WU00] Wu, X., “Knowledge object modeling,” IEEE Transactions on System. Man, and Cybernetics—Part a: Systems and Humans, Vol.30, NO.2, 2000

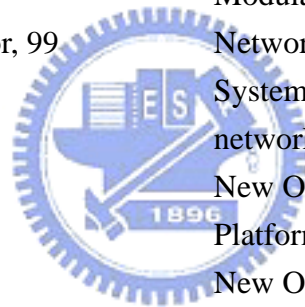
[WU99] Wu, Y.C., “An Approach to Object-oriented Rule Base Management System,” master thesis, Department of Computer and Information Science, National Chiao Tung University, 1999

[WV+01] Wache, H., Vgele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., and Hbner, S., “Ontology-based integration of information - a survey of existing approaches”, Proceedings of the Workshop Ontologies and Information Sharing, IJCAI, 2001

[WW99] Webb, G. and Wells, J., “An Experimental Evaluation of Integrating Machine Learning with Knowledge Acquisition,” Machine Learning, vol. 35, no. 1, 1999, pp. 5–23.

Index

- Abstraction, 27
- Acquire, 37
- Behavior Clustering, 74
- Behavior Clustering Algorithm, 75
- CAL, 108
- Certainty-factor, 36
- CF, 36
- cognitive psychologist, 46
- Computer Assisted Learning, 108
- Coretech Inc, 98
- DAML+OIL, 97
- Data Mining, 19
- DRAMA, 98
- DRAMA Knowledge Extractor, 99
- DRAMA Rule Editor, 99
- Expert System, 23
- expertise, 27
- Extension-of, 39
- greedy growth, 93
- Greedy Growth, 135
- Inter-cluster Semantic Clustering Criterion, 79, 85
- Intra-Cluster Semantic Clustering Criterion, 79, 85
- Intrusion Detection Knowledge Acquisition System, 118
- Intrusion Detection Knowledge Bases Fusion System, 118
- Intrusion Detection Knowledge Mining System, 118
- Java, 104
- KC, 34
- KDDCUP, 132
- Knowledge Acquisition, 19, 51
- Knowledge and Data Engineering Laboratory, 98
- Knowledge Class, 34
- Knowledge Engineering, 18
- knowledge fusion, 79
- Knowledge Management, 23
- Learning, 29
- Learning Content Management System, 109
- Learning Management System, 113
- LMS, 113
- MDE, 121
- Meta Detection Engine, 121
- Modularity, 27
- Network Intrusion Detection Expert System, 117
- network packet, 131
- New Object-oriented Rule Base Platform, iii, 12, 24
- New Object-oriented Rule Model, 32
- NID-ES, 117
- NORBP, 24
- NORM, 32
- Object-Oriented, 16
- ONAD, 121
- Online Network Analyzer/Detector, 121
- Ontology, 20
- Ontology Construction Phase, 95
- Pakemon, 105
- Pattern Explanation, 77
- Pattern Explanation Phase, 66
- Preprocessing Phase, 66
- Procedural knowledge, 46
- Pseudo Rule, 90
- psychology, 28



Reference, 38
Refinement, 50
Relationship Graph, 80
Relationship Graph Construction
Algorithm, 89
Relationship Graph Partitioning
Algorithm, 93
Renumber Sort Algorithm, 68
Reusability, 27
RTE, 113
rule, 35
Runtime Time Environment, 113
SCORM, 111
Sequential Pattern Mining, 76
Sequential Pattern Mining Algorithm,
76
Sharability, 28
Shared Vocabulary Ontology and
Semantic Distance Function, 90
Snort, 105
Strategic knowledge, 46
Structural Succinctness Criterion, 79,
84
TFN, 124
Tribal Flood Network, 124
Trigger, 37
Two-Layer Network Intrusion
Detection System, 118
Two-Layer Pattern Discovering Phase,
66, 73
Uncertainty reasoning, 28
weight, 36

