THESIS APPROVAL

GRADUATE SCHOOL, KASETSART UNIVERSITY

Doctor of Philosophy (Computer Science)

DEGREE

Computer Science	Computer Science
FIELD	**DEPARTMENT**

TITLE: Automatic Thai Legal Ontology Building and Supreme Court Sentences Retrieval Using Ant Colony Algorithm

NAME: Mr. Vi-sit Boonchom

THIS THESIS HAS BEEN ACCEPTED BY

THESIS ADVISOR

(Associate Professor Nuanwan Soonthornphisaj, Ph.D.)

DEPARTMENT HEAD

(Assistant Professor Sirikorn Channual, M.S.)

APPROVED BY THE GRADUATE SCHOOL ON _____

DEAN

(Associate Professor Gunjana Theeragool, D.Agr.)

Copyright by Kasetsart University All rights reserved

THESIS

AUTOMATIC THAI LEGAL ONTOLOGY BUILDING AND SUPREME COURT SENTENCES RETRIEVAL USING ANT COLONY ALGORITHM

VI-SIT BOONCHOM

A Thesis Submitted in Partial Fulfillment of
the Requirements for the Degree of
Doctor of Philosophy (Computer Science)
Graduate School, Kasetsart University
2012

Copyright by Kasetsart University All rights reserved

Vi-sit Boonchom 2012: Automatic Thai Legal Ontology Building and Supreme Court Sentences Retrieval Using Ant Colony Algorithm. Doctor of Philosophy (Computer Science), Major Field: Computer Science, Department of Computer Science. Thesis Advisor: Associate Professor Nuanwan Soonthornphisaj, Ph.D. 102 pages.

Ontology plays an important role in knowledge representation, especially in the domain of information retrieval. However, building ontology remains a challenging problem because it is a time-consuming task for experts. To overcome this problem, an Automatic Thai legal Ontology Building algorithm is proposed in order to reduce the burden of legal experts and to fulfill the criteria for a complete appreciation for users of legal documents. Moreover, users, especially law students, require a set of updated and diversified sentences from the court sentences retrieval process to ensure a comprehensive legal knowledge related to specific keywords. Unfortunately, some of the outdated and non diversified sentences are still yielded. Therefore, a Combination of the Ant colony algorithm and the Ontology algorithm is proposed in order to address such a predicament.

The Automatic Thai legal Ontology Building algorithm can automatically generate seed ontology and expand the ontology with the application of the Thai legal terminology. The expansion process is terminated automatically by the threshold parameter. Moreover, the *weight ontology* is modified to support both algorithms. The concept of ant colony algorithm is applied with the ontology in order to retrieve the updated and diversified sentences. Both algorithms are tested with the Thai court sentences repository. Empirical results demonstrate that the effective ontology should be an embedded ant colony algorithm. The performance of both algorithms measured in terms of the precision, recall, F-measure and diversity value are 0.95, 0.94, 0.94 and 0.40, respectively.

_____ _____ ___ / ___ / ___

Student's signature Thesis Advisor's signature

Copyright by Kasetsart University. All rights reserved

ACKNOWLEDGEMENTS

I would like to grateful thank and deeply indebted to Assoc. Prof. Dr. Nuanwan Soonthornphisaj my thesis advisor for providing the basic, advice, encouragement and valuable suggestion for my thesis. She was the first one who introduced me to the field of artificial intelligence and law and taught me a lot about research methodology. I would sincerely like to thank the other members of my committees– Prof. Dr. Boonserm Kijsirikul and Assoc. Prof. Dr. Kitsana Waiyamai who provided useful comments and finalized my work.

I am heartfelt thank to all the members of the CAL Laboratory who helpful research environment. Moreover, my friend: Miss Chantawan Noisri who helps me prepares for and overcome many obstacles along the way. I express my sincere gratitude to them.

Finally, I would also like to thank the Information Technology and Communication Center, Supreme Court of Thailand for kindly providing the Supreme Court sentences. Moreover, I would like to sincerely thank Thaksin University for providing the financial support of my study. This research was supported by the grant of the Office of the Higher Education Commission, Thailand. Most of all, my appreciations devote to my family for their love and support.

Vi-sit Boonchom

March 2012

Copyright by Kasetsart University. All rights reserved

TABLE OF CONTENTS

Page

Copyright by Kasetsart University. All rights reserved

LIST OF TABLES

Copyright by Kasetsart University. All rights reserved

LIST OF FIGURES

Appendix Figure

Copyright by Kasetsart University. All rights reserved

LIST OF ABBREVIATIONS

IR	=	information retrieval
ATOB	=	Automatic Thai legal Ontology Building
CSR	=	court sentences retrieval
CAO	=	combination of ant colony algorithm and ontology
ACO	=	Ant Colony Optimization
AI and Law	=	Artificial Intelligence and Law
AI	=	Artificial Intelligence
NLP	=	Natural Language Processing
URI	=	Uniform Resource Identifier
MRD	=	Machine-Readable Dictionary
CCT	=	Chinese Classified Thesaurus
LBS	=	Location Based Services
UCC	=	Uniform Commercial Code
OPJK	=	Ontology of Professional Judicial Knowledge
REL	=	Right Expression Languages
AS	=	Ant System
ACS	=	Ant Colony System
MMAS	=	MAX-MIN ant system
TF	=	term frequency
IDF	=	inverse document frequency

Copyright by Kasetsart University. All rights reserved

AUTOMATIC THAI LEGAL ONTOLOGY BUILDING AND SUPREME COURT SENTENCES RETRIEVAL USING ANT COLONY ALGORITHM

INTRODUCTION

Ontology plays a key role as a knowledge representation technique in order to improve the information retrieval (IR) performance in several domains such as medicine (Dridi and Ahmed, 2008), agriculture (Xie *et al*., 2008), e-learning (Henze *et al*., 2004), software engineering (Nicola *et al*., 2009), linguistics (Loukachevitch, 2009), semantic web (Bartalos *et al*., 2007), law (Saravanan *et al*., 2009), etc. Especially, in the law domain, legal ontology is used as a knowledge base for improving the efficiency of the retrieval process.

Although there are a lot of researches done in legal area, the existing legal ontologies can be performed on domain specific and language. There is a lack of studies about the Thai legal ontology. Hence, constructing a Thai legal ontology is becoming attractive task for researchers. Moreover, the main advantage of Thai legal ontology is to fulfill Thai legal knowledge for the users and the legal users can understand the overview of main legal knowledge concepts obtained from the ontology. These main concepts help self-study users to gain complete legal knowledge in the specific domain of law.

Currently, there are several sources used for ontology creation such as thesaurus, WordNet, raw text, specific domain dictionary, etc. One of the most challenging sources for the ontology building is a specific domain dictionary. The advantage characteristics of the specific domain dictionary are correct terms and their relationships, avoiding the synonym, hyponym or ambiguous of terms, and occasional updating by domain expert. There are many specific domain dictionaries which used for ontology construction such as Chinese dictionary (Lee *et al*., 2007), Korean

Copyright by Kasetsart University. All rights reserved

dictionary (Hwang *et al*., 2008), French-English-Malay dictionary (Serasset and Chevallet, 2004), ONTOMO dictionary (Shin *et al*., 2009), etc.

Ontology construction is a tedious job and time-consuming task. Therefore, to facilitate the legal expert, a new algorithm for automatic Thai legal ontology building (ATOB) is proposed. The ATOB algorithm uses a domain specific dictionary called TLlexicon as a resource. The TLlexicon is manually created by the domain expert. In the ATOB algorithm, the TLlexicon and the legal terms which extracted from the Supreme Court sentences are used for automatic seed ontology building and automatic ontology expanding process. The ontology is stored in XML format which supports the *Seed Ontology Building*, *Ontology Expansion* and *Query Expansion* process. The ontology size is depended on the threshold parameter which used in the *Ontology Expansion* process (Boonchom and Soonthornphisaj, 2009, 2010a, 2010b, 2012).

Moreover, law users require a list of relevant, qualified, diversified and updated sentences from the CSR process to ensure a complete legal knowledge related to the keyword. We found that, the relevant and qualified sentences can be obtained from the CSR process which uses the ontology as a knowledge base. However, some of the non diversified and outdated sentences are still retrieved from the CSR process. To overcome this drawback, a concept of the ant colony optimization algorithm (ACO) (Dorigo, 1992) is considered. The ACO algorithm is a meta-heuristic algorithm proposed for solving computational problems which inspired by the behavior of real ants to find good paths through graphs (Dorigo and Stutzle, 2004).

In this research, the state of the art of ant colony algorithm is challenging for the CSR process. The concept of ant colony algorithm can be applied for improving the CSR performance in order to retrieve the diversified and updated sentences. Therefore, the combination of the ant colony algorithm and the ontology (CAO) is proposed. In the CAO algorithm, the concept of ant colony algorithm is applied in the *Query Expansion* process for ordering the good candidate terms which obtained from the ontology by their probability value. Moreover, the concept of ant colony algorithm

Copyright by Kasetsart University. All rights reserved

is also modified for recalculating the pheromone level of the candidate terms in the *Parameters Updating* process and calculating the document score of each related sentence in the *Search* process. The diversified and updated sentences are retrieved by the capacity of ant colony algorithm. Hence, The CAO algorithm has more effect on the CSR efficiency.

The contributions of this research are

1. a new Thai legal ontology building framework using TLlexicon dictionary called ATOB algorithm (Boonchom and Soonthornphisaj, 2009, 2010a, 2010b, 2012).

2. a framework which combine the concept of ant colony algorithm and the ontology for improving the CSR performance called CAO algorithm.

3. a weight ontology which support the ATOB and CAO algorithm.

Copyright by Kasetsart University. All rights reserved

OBJECTIVES

1. Develop the ATOB algorithm for automatic ontology building using the domain specific dictionary.

2. Create a Thai legal ontology for improving the Thai CSR performance.

3. Develop the CAO algorithm in order to improve the Thai CSR performance.

Copyright by Kasetsart University. All rights reserved

LITERATURE REVIEW

This research topic focus on two main approaches: the ATOB and the CAO algorithm. In this section, the background of the Artificial Intelligence and Law (AI and Law), Thai law, ontology and the ACO algorithm are presented for the ease of understanding of the thesis.

1. Artificial Intelligence and Law

1.1 Definition

AI and Law is a subfield of Artificial Intelligence (AI) mainly concerned with applications of both AI and law domain. It is related to automation in the legal domain such as Natural Language Processing (NLP), intelligent databases, data mining, information retrieval and extraction and theories of legal decision making. Currently, works on AI and Law is having impact on the international non-standard logic and argumentation communities (Rissland *et al*., 2003).

1.2 The nature of law

Rissland *et al*. (2003) described the characteristics of legal domain as follow:

1.2.1 Law has diverse categories of knowledge and an abundance of cases, rules, theories, procedures, hierarchies of authority, norms and meta-rules.

1.2.2 In civil code countries like France, Germany, Japan and Thailand, the explicit styles and standards of justification place more emphasis on reasoning with the rules and codes.

Copyright by Kasetsart University. All rights reserved

1.2.3 In the different types of knowledge, there are different types of reasoning, for instance, reasoning with cases alone, rules alone, cases and rules together, etc.

1.2.4 Special repositories of knowledge consist of large collections of cases which available from a great variety of courts.

1.2.5 There are a variety of task orientations such as advocacy, adjudication, advising, planning and drafting, and administration.

1.2.6 Concepts in the law are not black and white with hard edged boundaries between positive and negative instances.

1.2.7 The underlying expectation in the legal domain is that, through the adversarial process, ideally the truth will out.

1.2.8 The law is a very reflective intellectual discipline. It constantly examines and re-examines its underlying methods and missions.

2. Thailand Law

The Kingdom of Thailand is a civil law country with strong common law influences. Thailand law is comprised of primarily statutory law and regulations promulgated by the various government ministries. Moreover, Supreme Court opinions are recorded and published an important basis for the development of Thailand law (Leeds, 2011).

There are four types of court in Thailand law which are: the Constitutional Court, the Courts of Justice, the Administrative Courts and the Military Courts. Especially, the Courts of Justice consist of three levels of courts, i.e. the courts of First Instance, the courts of Appeal and the Supreme Court. In the Supreme Court, the justice has jurisdiction to hear and adjudicate appeals from the Courts of Appeal and

Copyright by Kasetsart University. All rights reserved

from the specialized courts of the Courts of First Instance. The Supreme Court can hear appeals on questions of law and, in certain cases, on questions of fact. The Supreme Court of Justice has original jurisdiction in cases involving the removal or revocation of the right to vote of members of the National Assembly.

In the Kingdom of Thailand, judicial precedent is not binding on lower courts. The Supreme Court of Justice is not bound to follow its own decisions, and lower courts are not bound to follow precedents set by higher courts. In practice, however, the decisions of the Supreme Court of Justice do have significant influence on the Supreme Court of Justice itself and on lower courts. In addition, studying Thai law also needs Supreme Court sentences as a set of precedent cases to fulfill legal knowledge. Moreover, the users can understand the overview of main legal knowledge concepts obtained from the ontology whereas the main concepts help self-study users to gain complete legal knowledge in the domain of specific law

In this research, some of the Supreme Court sentences which related to the Civil and Commercial Code are used as the repository called Thai Legal Corpus.

3. Ontology

3.1 Definition

Ontology definition is a specifying characteristic of the knowledge representation and knowledge engineering in specific areas such as semantic web, systems and software engineering, linguistics, agriculture, law, etc. The term ontology was borrowed from Philosophy to be used in Computer Science and AI (Casellas, 2011). Philosophical ontology is the science of the kinds and structures of objects, properties, events, processes and relations in every area of reality (Smith and Welty, 2001). Moreover, ontology is used in the AI community to refer both the theory of existence and component of knowledge systems. The definition of ontology was defined by many researchers which depend on their aspects and objectives as shown in Table 1.

Copyright by Kasetsart University. All rights reserved

Table 1 The list of definition of ontology

No.	Author	Definition of ontology
1	Gruber (1991)	"Vocabularies of representational terms—classes, relations, functions and object constants—with agreed-upon definitions in the form of human readable text and machine-enforceable, declarative constrains on their well-formed use."
2	Gruber (1993)	"An explicit specification of a conceptualization. The ontology is a systematic account of existence. For knowledge based systems, what exists is exactly that which can be represented. This set of objects, and the describable relationships among them, are reflected in the representational vocabulary with which a knowledge-based program represents knowledge."
3	Gruber (1995)	"An explicit specification of a shared conceptualization of terms and their relationships of a domain of interest by, where terms is a crucial vocabulary of a domain of interest."
4	Guarino (1995)	"A logical theory that gives an explicit, partial account of a conceptualization which is defined as an intentional semantic structure and encoded the implicit rules constraining the structure of a piece of reality. An ontological theory is a designed artifact, a knowledge base of a special kind which can be read, sold or physically shared."
5	Uschold (1996)	"An explicit account or representation of some part of a conceptualization."
6	Kramer *et al*. (1997)	"A vocabulary for context definition. It is a set of terms and their relationships that are used in a domain, denoting concepts and objects."

Copyright by Kasetsart University. All rights reserved

Table 1 (Continued)

No.	Author	Definition of ontology
7	Benjamins *et al*., (1999)	"A shared and common understanding of some domain that can be communicated across people and computers."
8	Missikoff *et al*. (2003), Uschold (1996)	"An explicit and agreed specification about a shared conceptualization which consists of a vocabulary of terms and their relationships."
9	Li *et al*. (2007)	"An explicit specification of a conceptualization where definitions associate concepts, taxonomies, and relationships with human-readable text and formal, machine-readable axiom."
10	Wyner (2008)	"An explicit, formal, and general specification of a conceptualization of the properties of and relations between objects in a given domain. It defines a common vocabulary and organization of information which can be shared, tested, and modified by researchers."
11	Gruber (2009)	"A set of representational primitives with which to model a domain of knowledge or discourse. The representational primitives are typically classes (or sets), attributes (or properties), and relationships (or relations among class members). The definitions of the representational primitives include information about their meaning and constraints on their logically consistent application."

Moreover, the ontology can be defined in short for this thesis as: "*an explicit specification of shared conceptualization which consists of a set of crucial vocabulary of terms and their relationships"*.

Copyright by Kasetsart University. All rights reserved

3.2 Ontology components

The components of ontology were assigned by many researchers in the view of their objectives and applications. There are the examples of components of ontology which created by the researchers as shown in Figure 1.

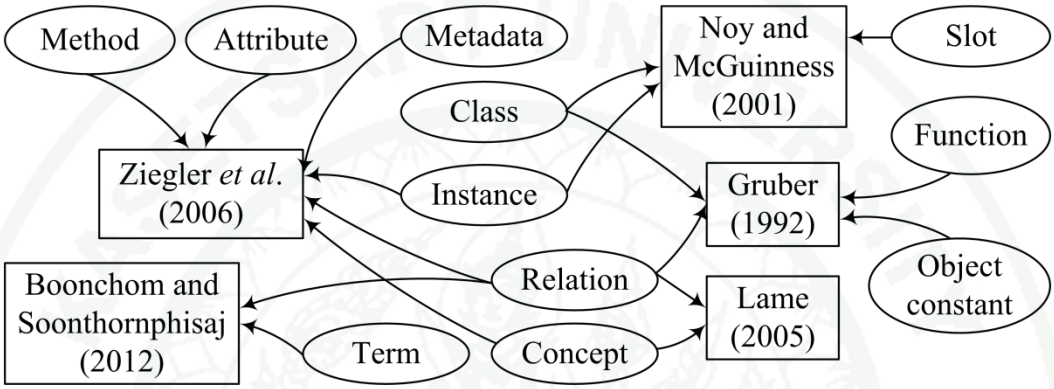


Figure 1 Ontology components

3.2.1 Gruber (1992) identified four kinds of components in the ontology: class, relation, function and object constant.

Ontology ← (*Class, Relation, Function, Object constant*)

where:

1) *Class* is a general idea or concept of many terms in the ontology which derived from specific instances. Moreover, class can be thought of as a collection of individual instances. The notion of class and relation are merged to unify relational and object centered representational conventions. For example, an object *i* is an instance of class *C*, it is denoted by the sentence C_i. Many terms defined in ontology are class which used to organize facts about instances of classes and relationships among classes.

Copyright by Kasetsart University All rights reserved

2) *Relation* is a set of tuples that represents an association among objects in the universe of discourse. Each tuple is a finite, ordered sequence (lists) of objects. Tuples are in the universe of discourse and can be represented as individual objects. For example, IS-A is the relation of R_p, R_c, where R_p is the parent concept and R_c is the child concept. For instance, $IS\text{-}A_{family;marriage}$ is a relationship between parent concept (*family*) and the child concept (*marriage*).

3) *Function* is a special relation which is sets of tuple just like ordinary relations, where the last item of tuple is the value of the function on the first items of the tuple.

4) *Object constant* are used to represent an instance of the object in the world.

3.2.2 Ziegler *et al*. (2006) designed the ontology for a particular purpose. There are six ontology components: concept, attribute, method, relationship, instance and ontological metadata as shown in Figure 2.

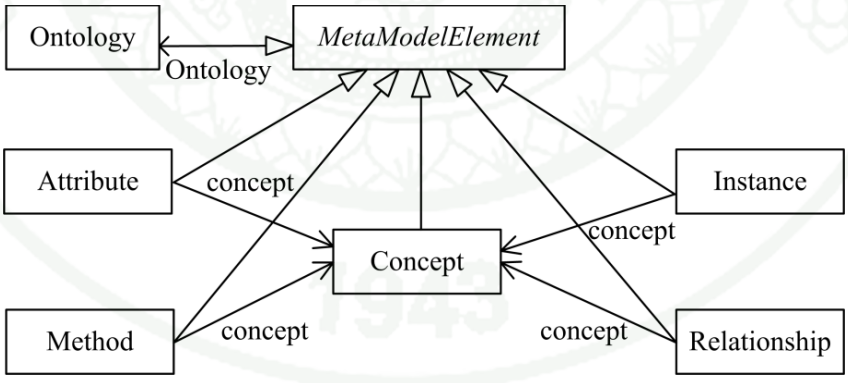


Figure 2 Example of domain specific ontology

Ontology ← (*Concept, Attribute, Method, Relationship, Instance, Metadata*)

Copyright by Kasetsart University. All rights reserved

where:

1) *Concept* is a description of a group of individuals that share common characteristics and occur in the particular universe of discourse. The concept can be described by attributes, methods and relationships. Further, each concept can have direct and indirect super- and sub-concepts, equivalent and antonym concepts, and coordinate concepts (that are situated on the same hierarchy level as the concept itself). For example, ontology structure like {*owl:Class....*}

2) *Attribute* is a property of concepts. Each attribute has a name, documentation, data type, definition and the name of the concept.

3) *Method* is a function which is described by a name, documentation, definition, its parameters, return type, and the name of the concept the method is declared for.

4) *Relationship* is a link between the related concepts.

5) *Instance* is an extension of the particular concept. Each instance has a name and provides concrete incarnations for the attribute values and relationships that are specified in its concept definition.

6) *Metadata* is used to describe the ontology by itself. This includes name, author, date of last modification, (header) documentation, version, copyright, and Uniform Resource Identifier (URI) of the ontology as well as the name of the ontology language. Additionally, each ontology has extensions of all concepts, attributes, methods, relationships, and instances that appear in it.

3.2.3 Noy and McGuinness (2001) proposed three kinds of the components of ontology as:

Ontology ← (*Class, Slot, Instance*)

Copyright by Kasetsart University. All rights reserved

where:

1) *Class* (Concept) is a description of the concept in the domain of the ontologies. For example, a class of wines represents all wines. Specific wines are instances of this class. The *Bordeaux* wine in the glass is an instance of the class of *Bordeaux* wines. A class can have subclasses that represent concepts that are more specific than the *Superclass*. For example, the class of all wines can be divided into red, white and rose wines. Alternatively, a class of all wines can be divided into sparkling and non sparkling wines.

2) *Slot* (Property, Role, Feature or Attribute) describe properties of classes and instances. For example, the *Chteau Lafite Rothschild Pauillac* wine has a full body, it is produced by the *Chteau Lafite Rothschild* winery. There are two slots describing the wine in this example. The slot *body* has the value *full* and the slot *maker* has the value *Chteau Lafite Rothschild winery*. At the class level, instances of the class *Wine* will have slots describing their flavor, body, sugar level, the maker of the wine and so on.

3) *Instance* is the basic objects of each class. For example, Figure 3 shows that, all instances of the class *Wine*, and its subclass *Pauillac*, have a slot maker the value of which is an instance of the class *Winery*. All instances of the class *Winery* have a slot produces that refers to all the wines (instances of the class *Wine* and its subclasses) that the winery produces.

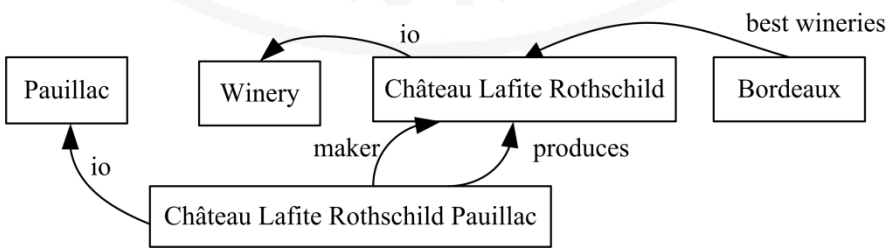


Figure 3 Example of classes, instances and relations among them

Copyright by Kasetsart University All rights reserved

3.2.4 Lame (2005) described the components of legal ontology in two kinds: concept and relation.

Ontology ← (*Concept, Relation*)

where:

1) *Concept* is a collection which labeled and embodied in terms. For instance, breaches of contract or liability are terms that label legal concepts. For example, the concept of *divorce* is referring to branch of *marriage*: A marriage is dissolved by the death of one of the spouses or by lawfully pronounced divorce. So, the divorce is defined as branch of marriage.

2) *Relation* is a semantic link which embedded in among the related of terms. For example, the relations of birth record for people who were born abroad linked to all its components (name, first names, sex, place of the birth, date of the birth, parentage, residence) which related to the general concept of birth record.

3.2.5 Boonchom and Soonthornphisaj (2012) proposed two types of the components in the ontology which are term and their relation.

Ontology ← (*Term, Relation*)

where:

1) *Term* is a set of crucial vocabulary in the domain of interest. For example, there are the legal terms in the Thai family law: สมรส (marriage), หย่า (divorce), หมั้น (engage), etc.

2) *Relation* is a link between two related terms. The type of relation is “IS-A” which used for connecting between them called parent and child term. In case that, a pair of related terms are a synonym or hypernym term. For

Copyright by Kasetsart University. All rights reserved

example, the ผู้จัดการมรดก (administrator of an estate) term is connected to the ทรัพย์ (property) term by “IS-A” relation. It means that, the ผู้จัดการมรดก (administrator of an estate) term is a parent term of the ทรัพย์ (property) term.

3.3 Types of ontology

The typology of ontology can be classified to clarify the revision of ontology. Casellas (2011) classified the general type of legal ontology into four mains aspects: the propose or use of the ontology influence, the subject-matter, the level of generality, and the level of formality of the ontology as shown in Figure 4.

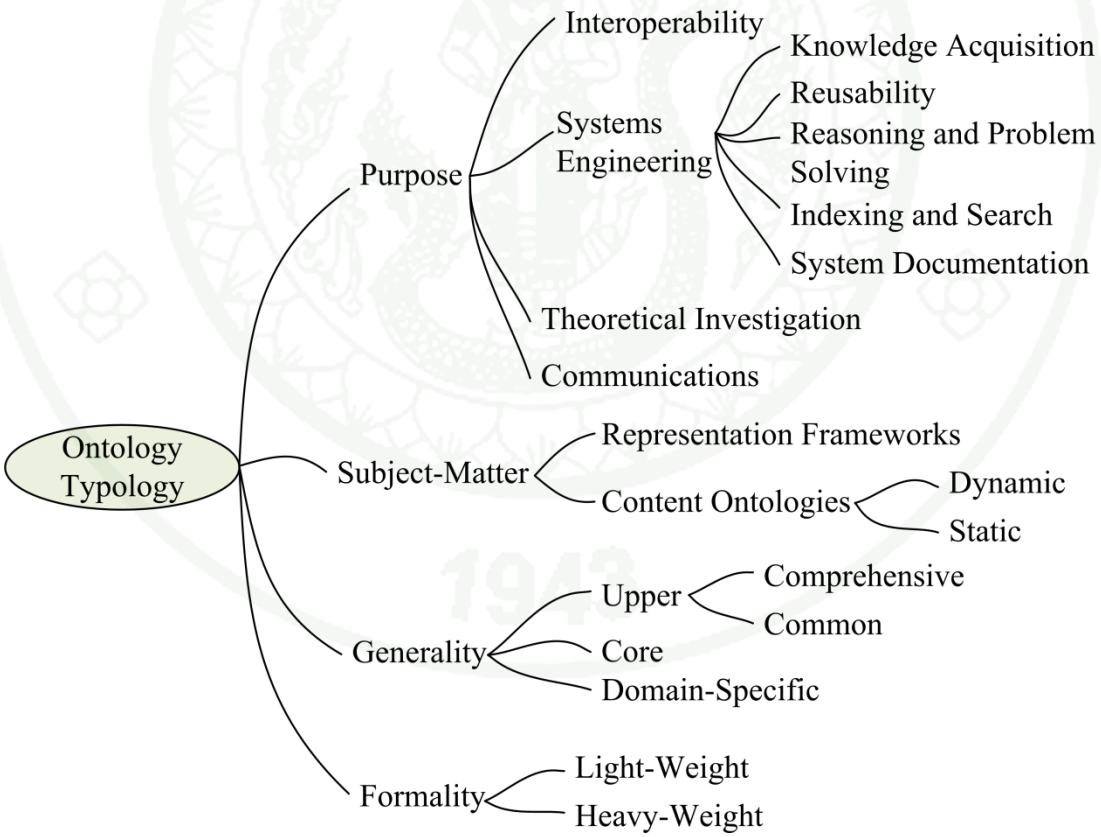


Figure 4 Types of ontology

Copyright by Kasetsart University All rights reserved

3.3.1 Purpose

Purpose is the aim of the construction or of the use of a particular ontology. This dimension may include both the general area of application and the specific intention of the ontology constructed. Taking into account the work of Uschold and King (1995); Uschold and Gruninger (1996); Uschold (1996); Noy and Hafner (1997); Studer *et al*. (1998); Visser and Bench-Capon (1998); Chandrasekaran *et al*. (1999); Valente (2005), ontologies may:

1) enable interoperability between systems.
2) provide benefits for knowledge based or information systems engineering:
    a) aid the process of knowledge acquisition.
    b) offer reusability.
    c) allow reasoning and problem solving.
    d) perform semantic annotation, indexing, search and retrieval.
    e) provide system documentation, within others.
3) conform a theoretical investigation.
4) enable communication (between human agents) through the organization and structuring of knowledge.

3.3.2 Subject-matter

Subject-matter is the type of knowledge that the ontology aims at representing. There are two kinds of subject-matter:

1) Representation frameworks or languages and content ontologies (understood, in general, as including claims regarding some world). Most authors refer to representation frameworks as also representation ontologies or metaontologies (Gruber, 1993; Uschold and King, 1995; Uschold, 1996; Studer *et al*., 1998; Chandrasekaran *et al*. 1999).

Copyright by Kasetsart University. All rights reserved

2) Content ontologies could be further divided into dynamic and static knowledge ontologies.

3.3.3 Generality

Generality is the general level of ontology which usually related to reusability issues. It includes several levels ranging from more abstract, general and independent ontologies towards dependent and specific ontologies (Guarino, 1997, 1998; Borst, 1997). Nevertheless, some issues need to be clarified. First, some authors refer to domain ontologies to describe a level of generality and to describe static knowledge representation regarding a specific domain, at the same time. Therefore, a clear distinction (or a clear terminology) between levels of generality and types of subject-matter is required. Second, and as a consequence, the level of generality and the subject-matter combine towards the distinction of domain, task and method ontologies. Domain, task and method ontologies refer to two different types of knowledge (domain and problem-solving) and may have, therefore, several levels of generality (core or specific). The so-called application ontologies are then specific ontologies which combine static and dynamic knowledge for a specific knowledge area and might become the less reusable of all (Uschold, 1996; Guarino, 1998). Finally, some ontologies might include several levels of generality at the same time. The levels might be distinct and modularized, even reused. A special type of these are large scale general purpose ontologies (Uschold and Jasper, 1999) or universal ontologies, that aim at modeling all existing common-sense or natural language knowledge (some linguistic ontologies, such as WordNet, could be considered universal). Therefore, several levels of generality may be established:

1) Top or upper-level ontologies (Guarino, 1997, 1998). These ontologies might be considered both kinds of upper-level ontologies:

a) Upper comprehensive ontologies (in the sense of describing very general concepts and provide general notions under which all root terms in existing ontologies should be linked).

Copyright by Kasetsart University. All rights reserved

b) Upper common ontologies (providing descriptions of some general notions that may be shared across domains, e.g., the Mereology ontology) (Mizoguchi *et al*., 1995; van Hest, 1995).

2) Core ontologies include top-level concepts of a domain, which allows reusability when formalizing knowledge of that part of the world (van Hest 1995; Valente and Breuker, 1996; Borst, 1997; Studer *et al*., 1998; Breuker, 2003).

3) Domain-specific ontologies (also understood as sub-domain ontologies, as the knowledge they capture is specific of some part of a domain, e.g., Criminal Law).

3.3.4 Formality

Formality and richness of internal structure are the level of formality (or machine-readability) of the constructed ontology. Here the list can be included by Uschold and Gruninger (1996); Uschold (1996); Uschold and Jasper (1999) regarding highly formal, semi-formal, structured informal and highly informal. The revision of existing legal ontologies mostly considers ontologies that include, at least, some formalization. Therefore, the classification will mostly include semi-formal ontologies and highly formal ontologies, which could also be referred to two kinds of formality ontology:

1) Light-weight ontologies.
2) Heavy-weight ontologies (Gomez-Perez *et al*., 2003).

3.4 The example of legal ontology

There are examples of existing ontologies which were built for the legal domain as shown in Table 2.

Copyright by Kasetsart University All rights reserved

Table 2 The example of legal ontology

No.	Author	Resource / Method	Purpose / Application
1.	Kurematsu and Yamaguchi (1997)	Machine-Readable Dictionary (MRD)	The legal ontology in the field of Contracts for the International Sale of Goods
2.	Hu and Du (2007)	WordNet and Chinese Classified Thesaurus (CCT)	The bilingual ontology
3.	Hwang *et al*. (2006)	KorLex 1.5 (Korean WordNet), NLP method	A Korean classifier ontology
4.	Lee *et al*. (2006)	Product information	An operational product ontology system for a government service
5.	Mitre *et al*. (2006)	The evolution of mobile and positioning technologies of Location Based Services (LBS)	An ontology for the Spanish main privacy law
6.	Garcia *et al*. (2007)	Semantic web ontologies for sharing language for copyright representation	A copyright ontology for improving the management of copyright in the Internet

Copyright by Kasetsart University. All rights reserved

Table 2 (Continued)

No.	Author	Resource / Method	Purpose / Application
7.	Breuker *et al*. (2004)	Common-sense knowledge	An ontology for Dutch criminal law in the e-Court European project
8.	Despres and Szulman (2007)	The TERMINAE method	micro-ontologies built from European community directives
9.	Bagby and Mullen (2007)	rules-based	Uniform Commercial Code ontology (UCC)
10.	Casellas (2008), Benjamins *et al*. (2005), Casellas *et al*. (2007), Vallbe (2009), Casellas (2009)	the IURISERVICE system,	Ontology of professional judicial knowledge (OPJK)
11.	Drumond *et al*. (2006, 2007)	the INFONORMA multi-agent recommender system	ONTOLEGIS and ONTOJURIS ontologies in the juridical domain
12.	Nadah *et al*. (2007)	Rights Expression Languages (REL)	An ontology of online licenses
13.	Agnoloni *et al*. (2007, 2009), Francesconi and Tiscornia (2008)	DALOS method and LOIS method with the database of 35,000 concepts in five European languages	Consumer protection ontology

Copyright by Kasetsart University. All rights reserved

3.5 Ontology building

In particular, building ontologies consist of three main approaches: manual ontologies constructing, reusing existing ontologies, and using semi-automatic methods (Antoniou and van Harmelen, 2004). In this part, some of the most relevant methodologies, formal representation languages and tools are presented in order to give an overview of the methodological steps for the construction of legal ontologies. The use of a methodological approach to knowledge-based construction, and thus, to ontology construction, may offer support to successfully reach their goals in time (Sure *et al*., 2006). An ontology engineering methodology considers, generally, several management (e.g., scheduling), development (e.g., conceptualization and formalization), and support activities (e.g., knowledge acquisition and evaluation) (Gomez-Perez *et al*., 2003; Sure *et al*., 2006).

3.5.1 The TOVE project ontologies

Gruninger and Fox (1995) established several steps towards a methodological construction and evaluation of ontologies (based on their experience building the TOVE project ontologies) as shown Figure 5:

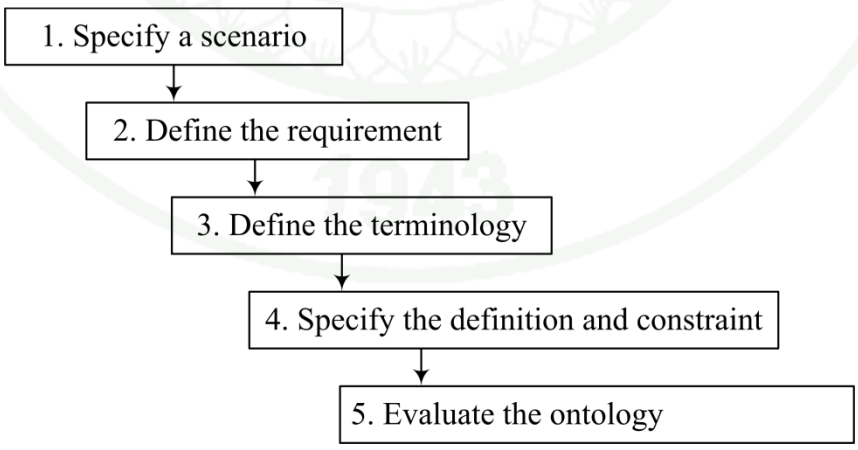


Figure 5 The step for ontology construction by TOVE project

Copyright by Kasetsart University. All rights reserved

Step 1: to specify an scenario that motivates the ontology and its applications,

Step 2: to define the requirements of the ontology as a set of competency questions that the ontology must be able to answer,

Step 3: to define the terminology of the ontology such as its objects, attributes, relations,

Step 4: to specify the definitions and constraints on the terminology and

Step 5: to evaluate the ontology testing the competency questions against completeness theorems.

3.5.2 The Ontolingua ontology construction

Uschold and King (1995); Uschold and Gruninger (1996); Uschold (1996) offered a set of guidelines towards ontology construction and merging. Explicit tool support is given by the Ontolingua Server. These principles heavily influenced the design of most of the more advanced ontology editors (Sure, 2003). Their proposal identified several parts in the ontology building process (see Figure 6)

1) Identifying the propose.
2) Deciding on the level of formality.
3) Building the ontology in several stages: Ontology capture, Ontology coding, and integrating existing ontologies.
4) Evaluation/Revision cycle.

Copyright by Kasetsart University All rights reserved

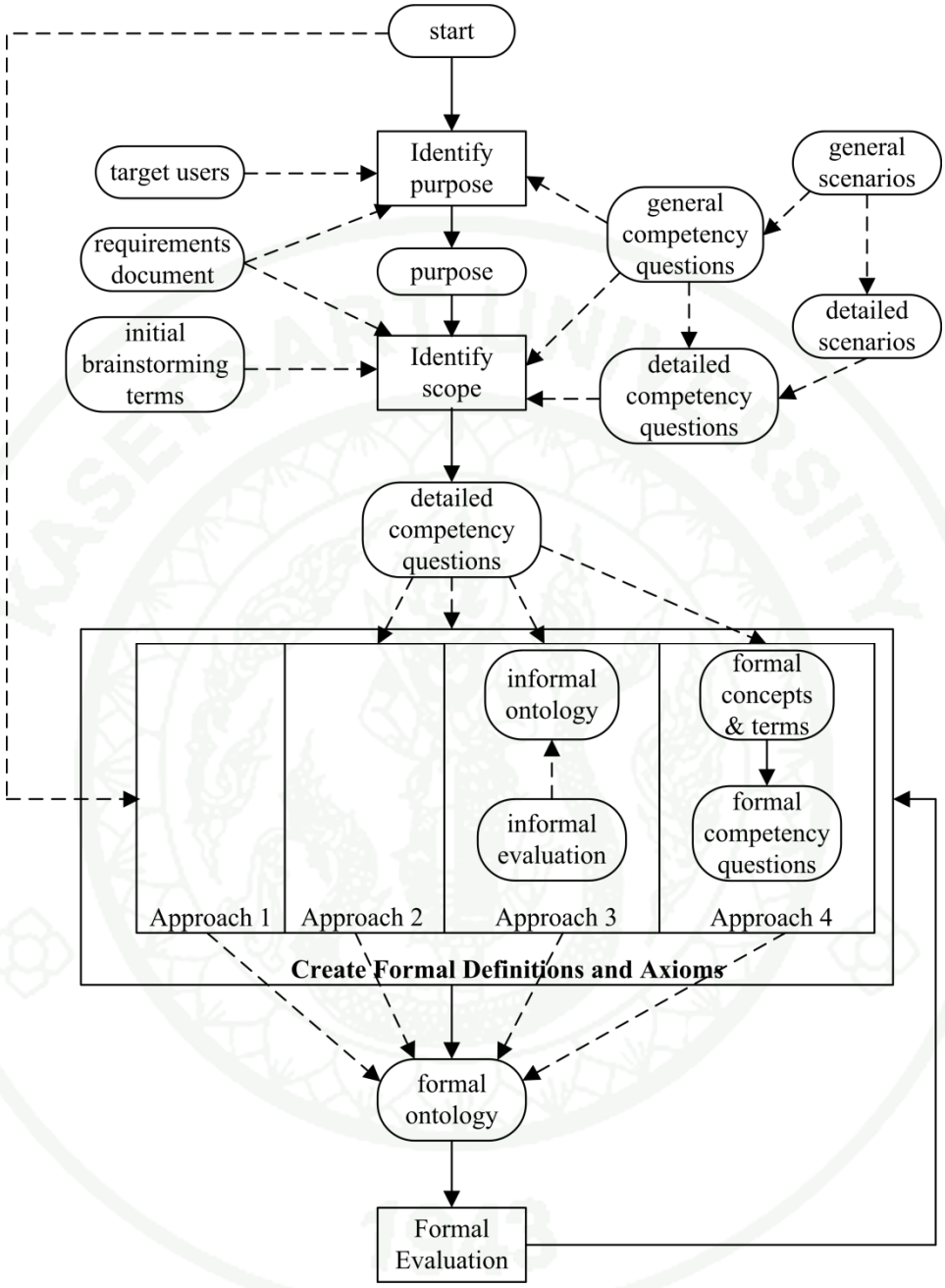


Figure 6 A views on the ontolingua ontology construction step

3.5.3 The KAKTUS methodology

Bernaras *et al*. (1996) developed the KAKTUS methodology which focused on the development of ontologies for particular applications and takes a

Copyright by Kasetsart University All rights reserved

bottom-up approach. The design process takes into account the possibility of reusing (refined and extended) ontologies already developed for the use of different applications in the domain (see Figure 7).

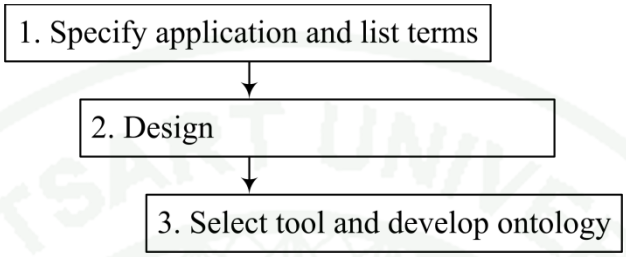


Figure 7 Three steps for KAKTUS ontology construction

1) First, the application is specified and lists of relevant terms and tasks are provided.

2) Second, a preliminary design is made according to the previous lists and specifications, which may include searching for already developed ontologies.

3) Finally, the ontology is refined towards the final design of the application. No specific tools supported this methodology.

3.5.4 The CommonKADS method

The CommonKADS method was tailored for legal knowledge system development based on four major design steps: analysis, conceptual modeling, formal modeling, and implementation (Visser and Bench-Capon, 1997; Visser, 1998). The method offered guidance towards the design of legal knowledge systems. However, they may be regarded as ontology modeling methodological steps. During the analysis phase, the domain knowledge, the sources (cases, articles, statutes, etc.) are identified and the tasks that the legal knowledge system has to perform using that domain knowledge have to be identified. During the conceptual modeling phase, the

Copyright by Kasetsart University. All rights reserved

method (task performing) needs to be described, the domain ontology selected and adapted and knowledge acquired and modeled. These two steps conform the expertise model, where domain knowledge and control knowledge (problem solving knowledge) need to be separated. This step is necessary because legal sources, intuitively modeled as domain knowledge, often contain procedural aspects (which suggest to model them as control knowledge) (Visser and Bench-Capon, 1997). Methodology for the development of legal knowledge base system was proposed (Visser and Bench-Capon, 1997; Visser, 1998; van Kralingen *et al*., 1999) as Figure 8:

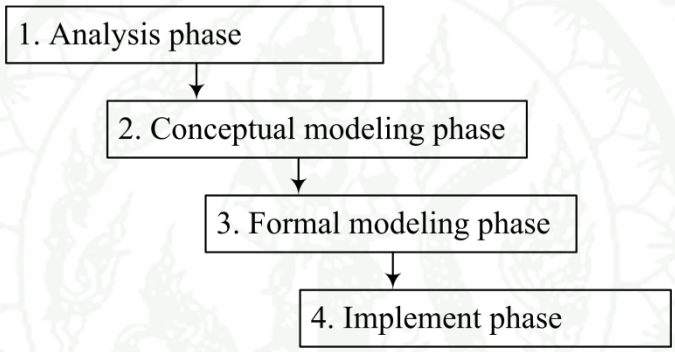


Figure 8 An overview of CommonKADS method for ontology construction

Step 1: Analysis phase which consists of task identification and domain identification.

Step 2: Conceptual modeling phase which consists of method description, domain ontology selection and adaption and knowledge acquisition and modeling.

Step 3: Formal modeling phase which consists of determine boundaries of control and domain knowledge, define control knowledge, create status-specific ontology, formalize domain knowledge and define inference.

Step 4: Implementation phase which consists of select language and platform and implementation.

Copyright by Kasetsart University. All rights reserved

3.5.5 The METHONTOLOGY

METHONTOLOGY is an extensive methodology because this methodology describes the different steps to be taken not only in the conceptualization process of an ontology but also during the ontology development life cycle (Fernandez *et al*., 1997; Fernandez-Lopez, 1999; Fernandez-Lopez and Gomez-Perez, 2002; Gomez-Perez *et al*., 2003). The methodology describes the different steps to be taken in the conceptualization process (see Figure 9).

The METHONTOLOGY development activities are:

1) Specification: establishes informally or formally the purpose and scope of the ontology (why, what use, who are the end users).

2) Conceptualization: organize the knowledge acquired such as build a glossary of terms, classify terms into one or more taxonomies of concepts, define binary relations between the concepts, built the concept dictionary, detail the concept dictionary and define axioms and rules.

3) Formalization.

4) Implementation.

5) Maintenance.

Copyright by Kasetsart University. All rights reserved

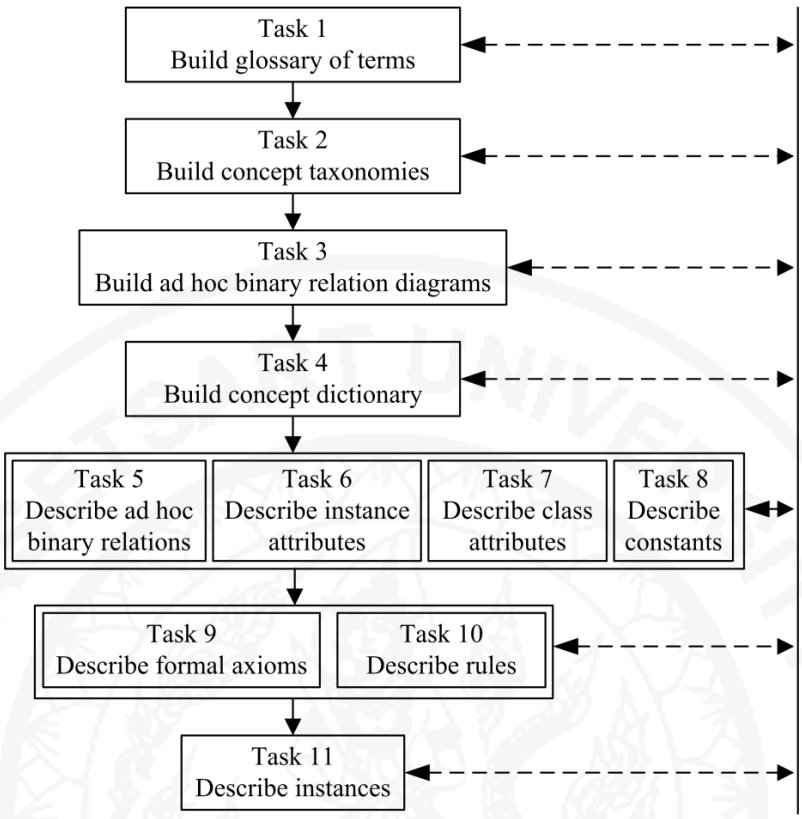


Figure 9 Tasks of the conceptualization of the METHONTOLOGY

3.5.6 The ontology development

Noy and McGuinness (2001) proposed the main states in the ontology creating process (see Figure 10)

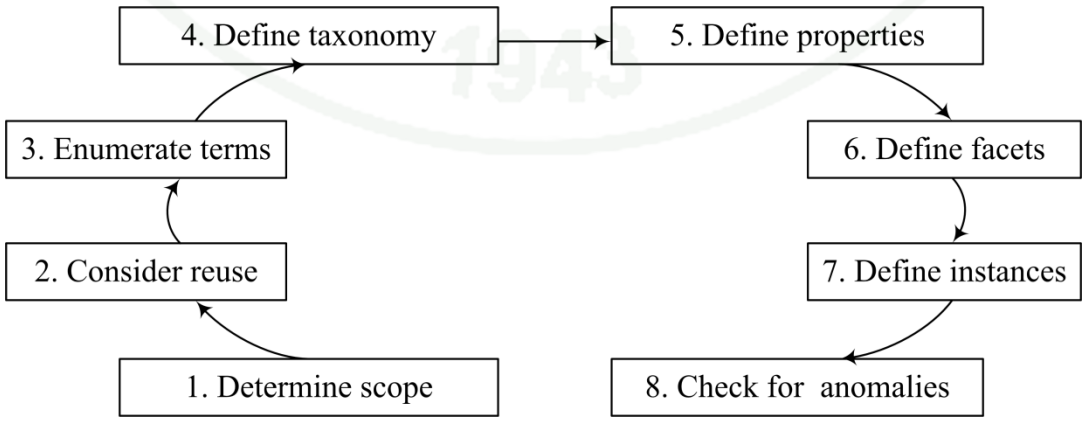


Figure 10 Process for ontology construction by Noy and McGuinness (2001)

Copyright by Kasetsart University All rights reserved

Step 1: the objective of the scope determination is to define a set of data and their structure of the domain to use.

Step 2: consider reuse process makes the ontologies becomes more widely available.

Step 3: enumerate terms process is a first step to write down in an unstructured list all the relevant terms that are expected to appear in the ontology.

Step 4: define taxonomy process, the terms must be organized in a taxonomic hierarchy. It is more efficient/ reliable to do this in a top-down or a bottom-up fashion. The hierarchy is indeed a taxonomic (subclass) hierarchy. In other words, if *A* is a subclass of *B*, then every instance of *A* must also be an instance of *B*.

Step 5: define properties process is often interleaved with the previous step. It is natural to organize the properties that link the classes while organizing theses classes in a hierarchy. For example, the semantics of the *subClassOf* relation demands that whenever *A* is a subclass of *B*, every property statement that holds for instances of *B* must also apply to instance of *A*. Because of this instance, it makes sense to attach properties to the highest class in the hierarchy to which they apply.

Step 6: define facets process changes the ontology from RDF Schema format to the OWL format using the cardinality, required values and relational characteristics.

Step 7: define instances process, the ontology is organized to the sets of instances and it is a separate step to fill the ontology with such instances. The number of instances is many orders of magnitude larger than the number of classes from the ontology.

Copyright by Kasetsart University. All rights reserved

Step 8: check for anomalies process is the use of OWL over RDF Schema to detect inconsistencies in the ontology or in the set of instances that were defined to populate the ontology.

3.6 Ontology construction algorithm

The ontology construction algorithms have been considered as a key performance for knowledge-based applications. The algorithm was developed by many researchers to fulfill their objectives.

3.6.1 The ontology development engine algorithm

Algorithm 1 *Ontology development Engine*

Input: Set of rules(RS) in the form of if antecedent(s) Then Consequent
Output: Suggested domain ontology
Algorithm:

Define Ontology concepts set (OCS) =∅
For every $rule_i \in RS$
Begin-for
1. Map $consequent_i$ to a concept C_i.
2. if $C_i \notin OCS$ then
   Add C_i to OCS
   End-if
3. Define Description Set (DS)=∅
4. for every antecedent of the form ($antecedent_x = category_y$)
   Begin-for
   a. Map $antecedent_x$ to a concept C_x .
   b. Map $antecedent_x\_category_y$ to a concept C_{xy}.
   c. Attach a subsumption relation between C_x and C_{xy}.
   d. Add C_{xy} to DS.
   End-for
5. Describe concept C_i using the intersection logical operator between elements of DS.
6. Map $Rule_i$ To a concept $RuleC_i$.
7. Attach a subsumption relation between $RuleC_i$ and C_i

End-for

A new ontology development algorithm was proposed for building ontology via set of rules generated by rule-based learning system (Ghalayini and

Copyright by Kasetsart University All rights reserved

Kharbat, 2008; Kharbat and Ghalayini, 2009). The proposed algorithm is described in Algorithm 1.

The ruleset is considered as a source input knowledge for developing the Wisconsin Breast Cancer Dataset (WBC) ontology. Note that, WBC is an UCI dataset which has the description of histological images taken from fine needle biopsies of breast masses. WBC consists of 699 test cases, in which every case has nine integer attributes associated with the diagnosis which are *Clump Thickness*, *Uniformity of Cell Size*, *Uniformity of Cell Shape*, *Marginal Adhesion*, *Single Epithelial Cell Size*, *Bare Nuclei*, *Bland Chromatin*, *Normal Nucleoli* and *Mitoses*. The process of applying the proposed algorithm to WBC ruleset is described by walked-through example to a specific rule. Suppose Rule#1 states that:

Rule#1: If ***1>Uniformity of Cell Shape > 4 and***
1 > Bare Nuclei > 4 and
1 > Blend Chromatin > 3 and
Normal Nucleoli = 1 and
Then ***the diagnosis = benign***

Having prepared Rule#1 that describes ***benign*** diagnosis, the algorithm of ontology development starts as follows:

1) The mapping of the consequent of Rule#1 to a ***benign*** concept using Step-1 since it is not included in the ontology concepts set (OCS).

2) The new concept of a ***benign*** is added to **OCS** using Step-2.

3) A new description set (**DS**) is defined as an empty set to accumulate the rule antecedents' definitions using Step-3.

4) The four antecedents in Rule#1, will be transformed using Step-4 as follows:

Copyright by Kasetsart University. All rights reserved

a) The first antecedent of Rule#1 is mapped to a concept of ***Uniformity-of–Cell-Shape***.

b) The antecedent of ***Uniformity of Cell Shape=Low or Mid*** is mapped to a concept of ***Low-Uniformity-of-Cell-Shape*** and to a concept of ***Mid-Uniformity-of-Cell-Shape.***

c) A subsumption relation is attached between the concept of ***Uniformity-of Cell-Shape*** and the sub-concepts of (***Low-Uniformity-of-Cell-Shape and Mid-Uniformity-of-Cell-Shape***) as shown in **Error! Reference source not found.**

d) Add the concepts of ***Low-Uniformity-of-Cell-Shape*** and ***Mid-Uniformity-of-Cell-Shape*** to the Description Set (DS) using the union logical operator. i.e., DS contains: ***Low Uniformity-of-Cell-Shape ⊔ Mid-Uniformity-of-Cell-Shape.***

The sub-steps of Step-4 will be repeated for the following antecedents:

Bare Nuclei=Low or Mid
Bland Chromatin=Low
Normal Nucleoli= Low

Thus, the following concepts and properties will be generated:

a) The concepts of ***Bare-Nuclei, Bland-Chromatin, and Normal-Nucleoli.***

b) The concepts of ***Low-Bare-Nuclei, Mid-Bare Nuclei, Low-Bland-Chromatin, and Low Normal-Nucleoli.***

Copyright by Kasetsart University. All rights reserved

c) A subsumption relation are attached between (1) the concept of ***Bare-Nuclei*** and the sub-concepts of (***Low-Bare-Nuclei*** and ***Mid-Bare-Nuclei***); (2) the concept of ***Bland Chromatin*** and the sub-concept of ***Low-Bland Chromatin***; (3) the concept of ***Normal-Nucleoli*** and the sub-concept of ***Low-Normal–Nucleoli***.

d) DS contains: (***Low-Uniformity-of-Cell-Shape*** ⊔ ***Mid-Uniformity-of-Cell-Shape***), (***Low-Bare-Nuclei*** ⊔ ***Mid-Bare-Nuclei***), ***Low-Bland-Chromatin***, and ***Low-Normal Nucleoli***.

5) The concept of ***benign*** is described using the intersection logical operator between elements of DS as follows:

(***Low-Uniformity-of-Cell-Shape*** ⊔ ***Mid-Uniformity-of- Cell-Shape***) ⊓ (***Low-Bare-Nuclei*** ⊔ ***Mid-Bare-Nuclei***) ⊓ ***Low-Bland-Chromatin*** ⊓ ***Low-Normal-Nucleoli***

6) Rule#1 is mapped to a concept as shown in **Error! Reference source not found.**

7) A subsumption relation is attached between Rule#1 and the concept of a ***benign*** as illustrated in **Error! Reference source not found**.

3.6.2 The automated ontology generation using spatial reasoning

Ontology is widely used in order to facilitate knowledge representation, integration, and reasoning. A vast amount of digitally available information may need to be considered when building ontologies. Some domains extension of the basic relationships could be enhanced further by the analysis of 2D and/or 3D images. Given a collection of 3D image files, utilizes Qualitative Spatial Reasoning (QSR) to automate the creation of an ontology through unambiguous Relational Connection Calculus (RCC) relations.

Copyright by Kasetsart University. All rights reserved

This ontology construction algorithm does not use a seed ontology, instead building the ontology from scratch. The input for Algorithm 2 is a set of objects (where each object has an associated 3D image) and, for each pair of objects in the collection, the RCC-8 relationship that holds between those two objects. The output of Algorithm 2 is an ontology containing: (1) a node for each of the individual objects, (2) nodes representing groupings of these objects classified using *is_a* or *part_of* relationships, and (3) annotative RCC-8 spatial relationships between each pair of objects. Note that, RCC-8 relations for two objects *x* and *y* are listed below (Coalter and Leopold, 2010).

EQ(x, y):	x is equivalent to y.
TPP(x, y):	x is tangentially a proper part of y.
NTPP(x, y):	x is nontangentially a proper part of y.
TPPc(x, y):	y is tangentially a proper part of x.
NTPPc(x, y):	y is nontangentially a proper part of x.
PO(x, y):	x partially overlaps y.
EC(x, y):	x is externally connected to y.
DC(x, y):	x is disconnected from y.

Algorithm 2 *Ontology Construction*

// Initialize the ontology.
Create the root node *Concept*.
As a child node of *Concept*, add a node for each individual object to be included in the ontology, joined to *Concept* by the relationship *part_of*.
Create a node *Synonym* as a child node of *Concept* joined by the relationship *is_a*.
// Handle equivalence relationships.
For each pair of individual objects *x* and *y* where EQ(x, y), do:
    Move node y to be a child of *Synonym* with the relationship *is_a*.
    Add a relationship stating *y is_synonym_of x*.
End for.
// Handle proper-part relationships.
For each pair of individual objects *x* and *y* where either TPP(x, y) or NTPP(x, y), do:
    If neither x nor y is an ancestor of the other in the ontology tree:
        Let n_1 be x.
        If x is a descendant of a group node, let n_1 be the most ancestral group node above x.
        Let n_2 be y.

Copyright by Kasetsart University. All rights reserved

Algorithm 2 (Continued)

If y is a descendant of a group node, let n_2 be the most ancestral group node above y.
If n_1 is not the same as n_2, move n_1 to be a child of y joined by the *part_of* relationship.
End if.
End for.
// Handle inverse proper-part relationships.
For each pair of individual objects *x* and *y* where either TPPc(x, y) or NTPPc(x, y), do:
If neither x nor y is an ancestor of the other in the ontology tree:
Let n_1 be x.
If x is a descendant of a group node, let n_1 be the most ancestral group node above x.
Let n_2 be y.
If y is a descendant of a group node, let n_2 be the most ancestral group node above y.
If n_1 is not the same as n_2, move n_2 to be a child of x joined by the *part_of* relationship.
End if.
End for.
// Handle overlap relationships.
For each pair of individual objects *x* and *y* where PO(x, y), do:
Let p_x be the parent node of node x.
Let p_y be the parent node of node y.
If x and y are each direct children of group nodes:
If p_x and p_y are different nodes
For each node n_c that is a child of p_y:
Move n_c to be a child of p_x with a *part_of* relationship.
End for.
Remove node p_y.
End if.
Else if x is a direct descendant of a group node, but y is not:
Move y to be a child of p_x with a *part_of* relation.
Else if y is directly related to a group node, but x is not:
Move x to be a child of p_y with a *part_of* relation.
Else if neither x nor y is an ancestor of the other:
Create a new group node as a child of *Concept* with a *part_of* relationship. Let this node be n_0.
Let n_1 be x.
If x is a descendant of a group node, let n_1 be the most ancestral group node above x.
Let n_2 be y.
If y is a descendant of a group node, let n_2 be the most ancestral group node above y.
Move n_1 to be a child of n_0 with a *part_of* relationship.
Move n_2 to be a child of n_0 with a *part_of* relationship.

Copyright by Kasetsart University. All rights reserved

Algorithm 2 (Continued)

End if.
End for.
// Handle EC relationships.
For each pair of individual objects *x* and *y* where EC(x, y), do:
Repeat the same logic as that for PO(x, y)
End for.
// Compress group nodes.
Until there is no more compression, do:
For each node x in the ontology tree that has exactly one child node y:
Let p_x be the parent of x.
If x is a group node, move y to be a child of p_x with a *part_of* relation, and then remove node x.
Else if y is a group node, move each node that is a child of y to be a child of x with a *part_of* relation, and then remove node y.
End for loop.
End until loop.
// Add the RCC-8 relations.
For each pair of objects x and y, add to the ontology both the RCC-8 relationship between x and y, R(x, y) and the inverse of the relation $R_i(y, x)$.

4. Ant Colony Optimization

4.1 Definition

In computer science, ACO is a probabilistic technique for solving computational problems which can be reduced to finding good paths through graphs. It is a population-based meta-heuristic that can be used to find approximate solutions of difficult optimization problems based on Ant System (AS) (Dorigo *et al*., 1996). ACO imitates behavior of ant colonies on their task for finding food source. Ants are capable of finding the shortest path between their nest and food source. They communicate through a chemical substance called *pheromone* (Sobecki, 2008). In the real world, each ant randomly seeks for food and leaves pheromone while moving on the chosen path. When they reach the food source, they return to the colony and leave the pheromone on the path. Although as time passes the pheromone will be evaporated, the ants which use the shortest path can be reinforced more rapidly pheromone (Gong and Wang, 2009) and other ants likely use the pheromone trail to be followed. The ants prefer the path with more pheromone trail deposited and the

Copyright by Kasetsart University. All rights reserved

pheromone level increase more rapidly on shortest path (Triay and Pastor, 2010). On the other hand, the pheromone will eventually vanish as time passes for a long period of time (Abachizadeh and Tahani, 2009) and were not reinforced by new ants (Alupoaei and Katkoori, 2004). Moreover, the evaporation pheromone helps to avoid the incorrect path (solution).

4.2 The framework of a basic ACO algorithm

The ant colony algorithm is an essence model of natural swarm intelligence which is inspired by observing the collective behaviors of social insect colonies (Liu *et al*., 2006; Xia *et al*., 2007). Initially, the ants move randomly in the search space for food and lay a pheromone trail on the ground in order to mark some favorable path. It means that the artificial ants explore all solution space and decide to follow the path which depends on the pheromone intensity (Dorigo *et al*., 2006).

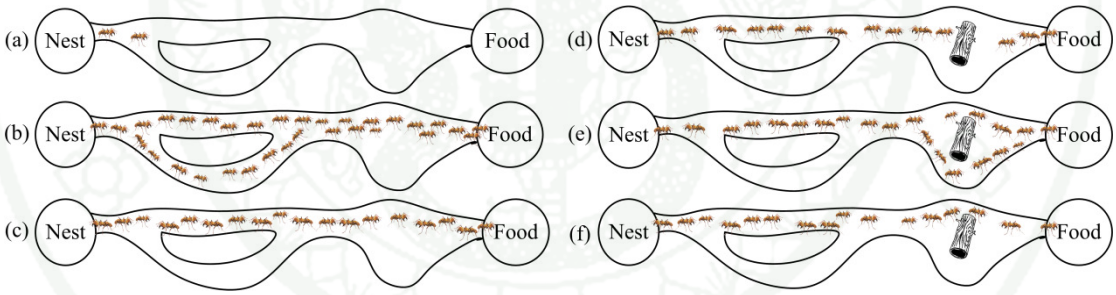


Figure 11 Behaviors of real ants between their nest and food source

Figure 11 shows six main steps of the behavior of ants which communicate through the pheromone trail for finding the shortest path between their nest and food source.

1) Starting from their nest
2) Exploring all possible paths
3) Following a path between their nest and food source
4) Encountering an obstacle of ants
5) Selection of possible paths

Copyright by Kasetsart University. All rights reserved

6) Finding the shortest path

Dorigo (2007) described that artificial ants build a solution to a combinatorial optimization problem by traversing a fully connected construction graph, defined as follows. First, each instantiated decision variable $X_i = v_i^j$ is called a solution component and denoted by c_{ij}. The set of all possible solution components is denoted by *C*. Then the construction graph $G_C(V, E)$ is defined by associating the components *C* either with the set of vertices *V* or with the set of edges *E*.

Informally, an ACO algorithm can be imagined as the interplay of three procedures: *ConstructAntsSolutions*, *UpdatePheromones*, and *DaemonActions* (Dorigo and Stutzle, 2004) (See Algorithm 3).

Algorithm 3 *ACOalgorithm*

```
procedure ACOMetaheuristic
    ScheduleActivities
        ConstructAntSolutions
        UpdatePheromones
        DaemonActions        % optional
    end-ScheduleActivities
end-procedure
```

4.2.*1* *ConstructAntSolutions* manages a colony of ants that concurrently and asynchronously visit adjacent states of the considered problem by moving through neighbor nodes of the problem's construction graph G_C. They move by applying a stochastic local decision policy that makes use of pheromone trails and heuristic information. In this way, ants incrementally build solutions to the optimization problem. Once an ant has built a solution, or while the solution is being built, the ant evaluates the (partial) solution that will be used by the UpdatePheromones procedure to decide how much pheromone to deposit.

4.2.2 *UpdatePheromones* is the process by which the pheromone trails are modified. The trails value can either increase, as ants deposit pheromone on the

Copyright by Kasetsart University All rights reserved

components or connections they use, or decrease, due to pheromone evaporation. From a practical point of view, the deposit of new pheromone increases the probability that those components/connections that were either used by many ants or that were used by at least one ant and which produced a very good solution will be used again by future ants. Differently, pheromone evaporation implements a useful form of forgetting: it avoids a too rapid convergence of the algorithm toward a suboptimal region, therefore favoring the exploration of new areas of the search space.

4.2.3 *DaemonActions* procedure is used to implement centralized actions which cannot be performed by single ants. Examples of daemon actions are the activation of a local optimization procedure, or the collection of global information that can be used to decide whether it is useful or not to deposit additional pheromone to bias the search process from a nonlocal perspective. As a practical example, the daemon can observe the path found by each ant in the colony and select one or a few ants (e.g., those that built the best solutions in the algorithm iteration) which are then allowed to deposit additional pheromone on the components/connections they used.

4.3 Main ACO algorithm

There are three most successful variants: Ant System (Dorigo, 1992; Dorigo *et al*., 1996), Ant Colony System (Dorigo and Gambardella, 1997), and MAX-MIN ant system (Stutzle and Hoos, 2000).

4.3.1 Ant system (AS)

The AS was the first ACO algorithm being proposed in the literature (Dorigo, 1992; Dorigo *et al*., 1996). Its main characteristic is that the pheromone values are updated by all the ants that have completed the tour. Solution components c_{ij} are the edges of the graph, and the pheromone update for τ_{ij}, that is, for the pheromone associated to the edge joining cities *i* and *j*, is performed as follows (see equation 1):

Copyright by Kasetsart University. All rights reserved

$$\tau_{ij} \leftarrow (1-\rho)\bullet\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \quad (1)$$

where $\rho \in (0,1]$ is the evaporation rate, *m* is the number of ants, and $\Delta\tau_{ij}^{k}$ is the quantity of pheromone laid on edge (*i; j*) by the *k*-th ant (see equation 2):

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{1}{L_k} & \text{if ant } k \text{ used edge } (i, j) \text{ in its tour}, \\ 0 & \text{otherwise}, \end{cases} \quad (2)$$

where L_k is the tour length of the *k*-th ant.

When constructing the solutions, the ants in AS traverse the construction graph and make a probabilistic decision at each vertex. The transition probability $p\left(c_{ij} \mid s_k^p\right)$ of the *k*-th ant moving from city *i* to city *j* is given by equation (3)

$$p\left(c_{ij} \mid s_k^p\right) = \begin{cases} \frac{\tau_{ij}^{\alpha} \bullet \eta_{ij}^{\beta}}{\sum_{c_{il} \in N\left(s_k^p\right)} \tau_{il}^{\alpha} \bullet \eta_{il}^{\beta}} & \text{if } j \in N\left(s_k^p\right), \\ 0 & \text{otherwise}, \end{cases} \quad (3)$$

where $N\left(s_k^p\right)$ is the set of components that do not belong yet to the partial solution s_k^p of ant k , and α and β are parameters that control the relative importance of the pheromone versus the heuristic information $\eta_{ij} = 1/d_{ij}$, where d_{ij} is the length of component c_{ij} (i.e., of edge (i, j)).

4.3.2 Ant colony system (ACS)

The first major improvement over the original AS being proposed was ACS (Dorigo and Gambardella, 1997). The first important difference between ACS and AS is the form of the decision rule used by the ants during the construction process. Ants in ACS use the so-called *pseudorandom proportional* rule: the

Copyright by Kasetsart University. All rights reserved

probability for an ant to move from city *i* to city *j* depends on a random variable *q* uniformly distributed over $[0,1]$, and a parameter q_0; if $q \le q_0$, then, among the feasible components, the component that maximizes the product $\tau_{il}\eta_{il}^{\beta}$ is chosen; otherwise, the same equation as in AS is used.

This rather greedy rule, which favors exploitation of the pheromone information, is counterbalanced by the introduction of a diversifying component: the *local pheromone update*. The local pheromone update is performed by all ants after each construction step. Each ant applies it only to the last edge traversed by equation (4).

$$\tau_{ij} = (1-\varphi)\bullet\tau_{ij} + \varphi\bullet\tau_0, \tag{4}$$

where $\varphi \in (0,1]$ is the pheromone decay coefficient, and τ_0 is the initial value of the pheromone.

The main goal of the local update is to diversify the search performed by subsequent ants during one iteration. In fact, decreasing the pheromone concentration on the edges as they are traversed during one iteration encourages subsequent ants to choose other edges and hence to produce different solutions. This makes less likely that several ants produce identical solutions during one iteration. Additionally, because of the local pheromone update in ACS, the minimum values of the pheromone are limited.

Note that, at of the end of the construction process a pheromone update, called *offline pheromone update*, is performed.

ACS *offline pheromone upd*ate is performed only by the best ant, that is, only edges that were visited by the best ant are updated, according to the equation (5).

Copyright by Kasetsart University. All rights reserved

$$\tau_{ij} \leftarrow (1-\rho)\bullet\tau_{ij} + \rho\bullet\Delta\tau_{ij}^{best} \quad (5)$$

where $\Delta\tau_{ij}^{best} = 1/L_{best}$ if the best ant used edge (*i; j*) in its tour, $\Delta\tau_{ij}^{best} = 0$ otherwise (L_{best} can be set to either the length of the best tour found in the current iteration – *iteration best,* L_{ib} – or the best solution found since the start of the algorithm – *best-so-far,* L_{bs}).

4.3.3 MAX-MIN ant system (MMAS)

MMAS ant system is another improvement, proposed by (Stutzle and Hoos, 2000), over the original ant system idea. MMAS differs from AS in that:

1) only the best ant adds pheromone trails, and

2) the minimum and maximum values of the pheromone are explicitly limited (in AS and ACS these values are limited implicitly, that is, the value of the limits is a result of the algorithm working rather than a value set explicitly by the algorithm designer). The pheromone update equation takes the following form (see equation 6).

$$\tau_{ij} \leftarrow (1-\rho)\bullet\tau_{ij} + \Delta\tau_{ij}^{best} \quad (6)$$

where $\Delta\tau_{ij}^{best} = 1/L_{best}$ if the best ant used edge (*i; j*) in its tour, $\Delta\tau_{ij}^{best} = 0$, otherwise, where L_{best} is the length of the tour of the best ant. As in ACS, L_{best} may be set (subject to the algorithm designer decision) either to L_{ib} or to L_{bs}, or to a combination of both.

The pheromone values are constrained between $\tau_{\min}$ and $\tau_{\max}$ by verifying, after they have been updated by the ants, that all pheromone values are within the imposed limits: τ_{ij} is set to $\tau_{\max}$ if $\tau_{ij} > \tau_{\max}$ and to $\tau_{\min}$ if $\tau_{ij} < \tau_{\max}$. It is

Copyright by Kasetsart University. All rights reserved

important to note that the pheromone update equation of MMAS is applied, as it is the case for AS, to all the edges while in ACS it is applied only to the edges visited by the best ants.

The minimum value $\tau_{\min}$ is most often experimentally chosen (however, some theory about how to define its value analytically has been developed) (Stutzle and Hoos, 2000). The maximum value $\tau_{\max}$ may be calculated analytically provided that the optimum ant tour length is known. In the case of the TSP, $\tau_{\max} = 1/\left(\rho \bullet L^*\right)$ where L^* is the length of the optimal tour. If L^* is not known, it can be approximated by L_{bs}. It is also important to note that the initial value of the trails is set to $\tau_{\max}$, and that the algorithm is restarted when no improvement can be observed for a given number of iterations.

4.3.4 Applications of Ant Colony Optimization

In recent years, the interest of the scientific community in ACO has risen sharply. In fact, several successful applications of ACO to a wide range of different discrete optimization problems are now available. The number of successful applications to academic problems has motivated people to adopt ACO for the solution of industrial problems, proving that this computational intelligence technique is also useful in real-world applications as shown in Figure 12.

1) *Applications to NP-Hard Problems*. The usual approach to show the usefulness of a new meta-heuristic technique is to apply it to a number of different problems and to compare its performance with that of already available techniques. ACO has been tested on probably more than one hundred different NP-hard problems. Many of the tackled problems can be considered as falling into one of the following categories: *routing problems* as they arise, for example, in the distribution of goods; *assignment problems*, where a set of items (objects, activities, etc.) has to be assigned to a given number of resources (locations, agents, etc.) subject to some constraints; *scheduling problems*, which in the widest sensor concerned with

Copyright by Kasetsart University. All rights reserved

the allocation of scarce resources to tasks over time; and *subset problems*, where a solution to a problem is considered to be a selection of a subset of available items. In addition, ACO has been successfully applied to other problems emerging in fields such as machine learning and bioinformatics.

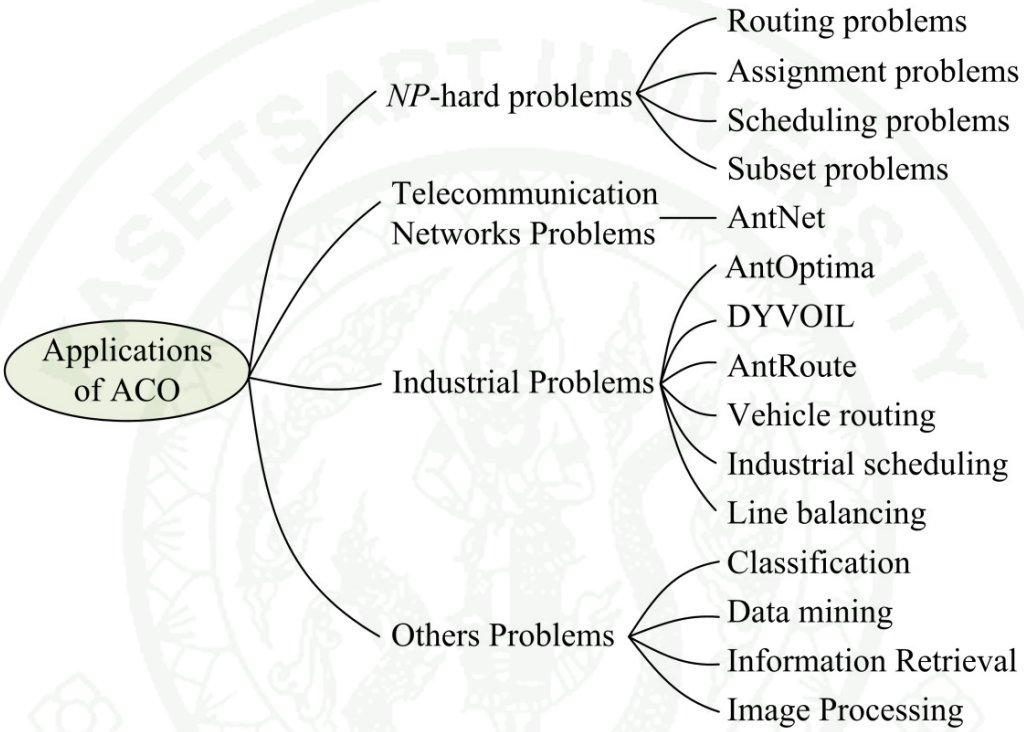


Figure 12 The example of applications of ACO

2) *Applications to Telecommunication Networks*. ACO algorithms have shown to be a very effective approach for routing problems in telecommunication networks where the properties of the system, such as the cost of using links or the availability of nodes, vary over time. ACO algorithms were first applied to routing problems in circuit switched networks (such as telephone networks) (Schoonderwoerd *et al*., 1996) and then in packet-switched networks (such as local area networks or the Internet). A well-known example is *AntNet* (Caro and Dorigo, 1998). AntNet has been extensively tested, in simulation, on different networks and under different traffic patterns, proving to be highly adaptive and robust.

Copyright by Kasetsart University. All rights reserved

3) *Applications to Industrial Problems*. The success on academic problems has raised the attention of a number of companies that have started to use ACO algorithms for real-world applications. Among the first to exploit algorithms based on the ACO metaheuristic is EuroBios (www.eurobios.com). They have applied ACO to a number of different scheduling problems such as a continuous two-stage flow shop problem with finite reservoirs. The problems modeled included various real-world constraints such as setup times, capacity restrictions, resource compatibilities and maintenance calendars.

4) *Others Problems*. The ACO can be applied for other areas such as Classification, Data mining, Information Retrieval, Image Processing, etc.

Copyright by Kasetsart University. All rights reserved

MATERIALS AND METHODS

Materials

1. Computer

The ATOB and CAO algorithm are implemented by using PHP programming language. The experiments are performed under the following computer specification:

1. Intel Core 2 Duo processor 2.00 GHz.
2. RAM 2 GB
3. Hard Disk 250 GB

2. Data

There are two resources needed for the ATOB and CAO algorithm: a set of Supreme Court sentences and a TLlexicon dictionary.

2.1 Supreme Court sentences

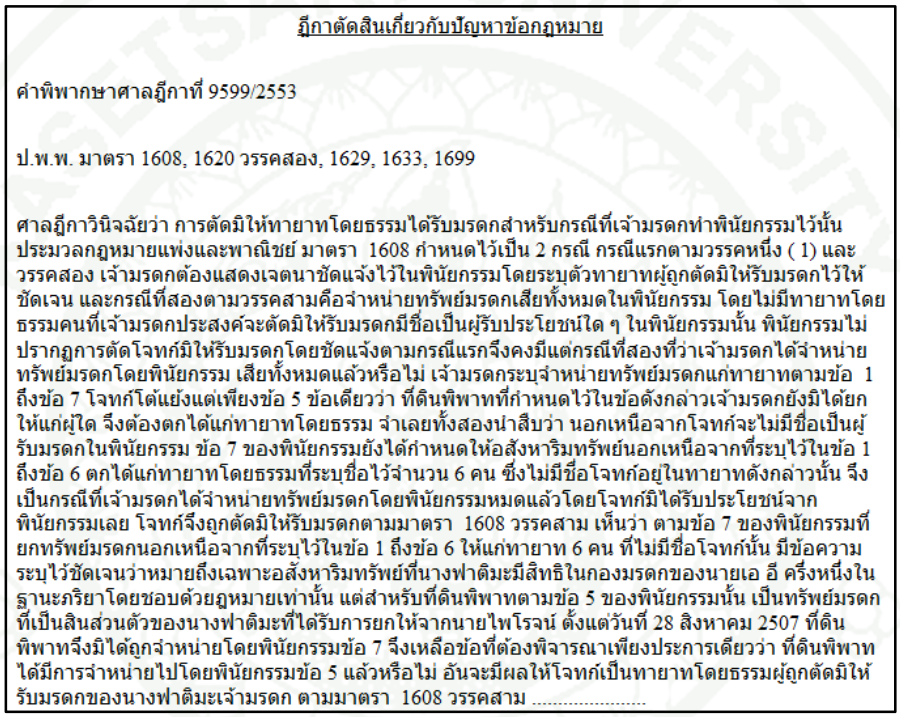
The Supreme Court sentences repository which used in order to test the methodology in this work deals with the domain of Thai law. There are 5,100 completed Court sentences in the corpus which formed in raw text format. The example of raw text is shown in Figure 13. All legal terms are extracted from each sentence and the total number of them in the corpus is 658,457 terms.

2.2 Thai legal dictionary

TLlexicon dictionary is a list of pairs of Thai legal terminology (terms) and their relationships which created manually by the domain expert. TLlexicon covers two areas of Thai Civil and Commercial code, which are succession and family law. The advantages of the specific domain dictionary are correct structure,

Copyright by Kasetsart University. All rights reserved

occasionally updated and suitable for extracting terms and their relationships. Moreover, TLlexicon is used as a source for both a *Seed Ontology Building* and *Ontology Expansion* process. There are 159 terms and 825 relationships used in TLlexicon which extracted from various sources of Thai law such as codified laws, judicial decisions and the expert's experience. Furthermore, the TLlexicon is designed for adding new legal terms and their relationships in the future.



ฎีกาตัดสินเกี่ยวกับปัญหาข้อกฎหมาย

คำพิพากษาศาลฎีกาที่ 9599/2553

ป.พ.พ. มาตรา 1608, 1620 วรรคสอง, 1629, 1633, 1699

ศาลฎีกาวินิจฉัยว่า การตัดมิให้ทายาทโดยธรรมได้รับมรดกสำหรับกรณีที่เจ้ามรดกทำพินัยกรรมไว้นั้น ประมวลกฎหมายแพ่งและพาณิชย์ มาตรา 1608 กำหนดไว้เป็น 2 กรณี กรณีแรกตามวรรคหนึ่ง (1) และวรรคสอง เจ้ามรดกต้องแสดงเจตนาชัดแจ้งไว้ในพินัยกรรมโดยระบุตัวทายาทผู้ถูกตัดมิให้รับมรดกไว้ให้ชัดเจน และกรณีที่สองตามวรรคสามคือจำหน่ายทรัพย์มรดกเสียทั้งหมดในพินัยกรรม โดยไม่มีทายาทโดยธรรมคนที่เจ้ามรดกประสงค์จะตัดมิให้รับมรดกมีชื่อเป็นผู้รับประโยชน์ใด ๆ ในพินัยกรรมนั้น พินัยกรรมไม่ปรากฏการตัดโจทก์มิให้รับมรดกโดยชัดแจ้งตามกรณีแรกจึงคงมีแต่กรณีที่สองที่ว่าเจ้ามรดกได้จำหน่ายทรัพย์มรดกโดยพินัยกรรม เสียทั้งหมดแล้วหรือไม่ เจ้ามรดกระบุจำหน่ายทรัพย์มรดกแก่ทายาทตามข้อ 1 ถึงข้อ 7 โจทก์โต้แย้งแต่เพียงข้อ 5 ข้อเดียวว่า ที่ดินพิพาทที่กำหนดไว้ในข้อดังกล่าวเจ้ามรดกยังมิได้ยกให้แก่ผู้ใด จึงต้องตกได้แก่ทายาทโดยธรรม จำเลยทั้งสองนำสืบว่า นอกเหนือจากโจทก์จะไม่มีชื่อเป็นผู้รับมรดกในพินัยกรรม ข้อ 7 ของพินัยกรรมยังได้กำหนดให้อสังหาริมทรัพย์นอกเหนือจากที่ระบุไว้ในข้อ 1 ถึงข้อ 6 ตกได้แก่ทายาทโดยธรรมที่ระบุชื่อไว้จำนวน 6 คน ซึ่งไม่มีชื่อโจทก์อยู่ในทายาทดังกล่าวนั้น จึงเป็นกรณีที่เจ้ามรดกได้จำหน่ายทรัพย์มรดกโดยพินัยกรรมหมดแล้วโดยโจทก์มิได้รับประโยชน์จากพินัยกรรมเลย โจทก์จึงถูกตัดมิให้รับมรดกตามมาตรา 1608 วรรคสาม เห็นว่า ตามข้อ 7 ของพินัยกรรมที่ยกทรัพย์มรดกนอกเหนือจากที่ระบุไว้ในข้อ 1 ถึงข้อ 6 ให้แก่ทายาท 6 คน ที่ไม่มีชื่อโจทก์นั้น มีข้อความระบุไว้ชัดเจนว่าหมายถึงเฉพาะอสังหาริมทรัพย์ที่นางฟาติมะมีสิทธิในกองมรดกของนายเอ อีกครึ่งหนึ่งในฐานะภริยาโดยชอบด้วยกฎหมายเท่านั้น แต่สำหรับที่ดินพิพาทตามข้อ 5 ของพินัยกรรมนั้น เป็นทรัพย์มรดกที่เป็นสินส่วนตัวของนางฟาติมะที่ได้รับการยกให้จากนายไพโรจน์ ตั้งแต่วันที่ 28 สิงหาคม 2507 ที่ดินพิพาทจึงมิได้ถูกจำหน่ายโดยพินัยกรรมข้อ 7 จึงเหลือข้อที่ต้องพิจารณาเพียงประการเดียวว่า ที่ดินพิพาทได้มีการจำหน่ายไปโดยพินัยกรรมข้อ 5 แล้วหรือไม่ อันจะมีผลให้โจทก์เป็นทายาทโดยธรรมผู้ถูกตัดมิให้รับมรดกของนางฟาติมะเจ้ามรดก ตามมาตรา 1608 วรรคสาม .....

Figure 13 Example of raw text of Supreme Court sentence

The TLlexicon dictionary consists of a set of legal terms called *Term* and their parent called *Superclass*. The node namely *Thing* is used as the root node. For example, there are simple and small legal terms, and their relationships: the ครอบครัว (family) node and the มรดก (succession) node have the same *Superclass* called *Thing* node. Moreover, the ครอบครัว (family) node is a *Superclass* node of both the สมรส

Copyright by Kasetsart University. All rights reserved

(marriage) node and the หมั้น (engage) node. Additionally, the พินัยกรรม (will) node is a child node of the มรดก (succession) node which is shown in Table 3.

Table 3 The example of *Term* and their *Superclass* in TLlexicon

No.	Superclass	Term
1	Thing (root node)	ครอบครัว (family)
2	Thing (root node)	มรดก (succession)
3	ครอบครัว (family)	สมรส (marriage)
4	ครอบครัว (family)	หมั้น (engage)
5	มรดก (succession)	พินัยกรรม (will)
…	…	…

Methods

1. System Overview

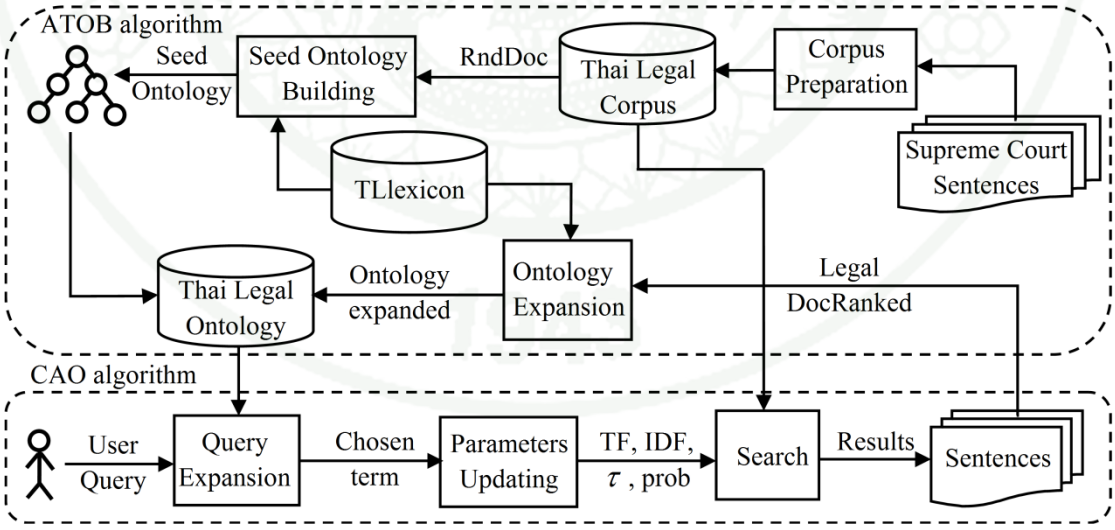


Figure 14 The ATOB and CAO algorithm

The framework of this research is shown in Figure 14 which consists of two main algorithms. The first one is the ATOB algorithm which used for automatic legal

Copyright by Kasetsart University. All rights reserved

ontology construction. There are three processes used in the ATOB algorithm which are the corpus preparation, the automatic seed ontology building and the automatic ontology expansion process. Moreover, in the retrieval process, the CAO algorithm is developed for improving the retrieval efficiency using the combination of the ant colony algorithm and the ontology. Interestingly, the ontology structure is modified by the ant colony algorithm for supporting both ATOB and CAO algorithm called *weight ontology*. These algorithms are explained in the following.

2. ATOB algorithm

There are two legal resources used for automatic ontology building process: TLlexicon and Thai Legal Corpus (see Figure 14). Since a specific domain dictionary has more certain structure and accurate information such as terms and their relationships than thesaurus and raw text, it is the best resource to extract this information for ontology building process. Moreover, the Supreme Court sentences which were stored in the Thai Legal Corpus are raw text. They consist of many important attributes such as sentence number, text, section and legal terms. All of them can be extracted and used in the ontology building process. There are three processes in the ATOB algorithm: *Corpus Preparation, Seed Ontology Building* and *Ontology Expansion* process.

2.1 Corpus preparation process

2.1.1 The Thai Legal Corpus

The Thai Legal Corpus was collected from two resources: the Thai Supreme Court sentences version 1.6 for PC (DEKA 2007) that was collected by the Information Technology and Communication Center, Supreme Court of Thailand and the Supreme Court search engine website (www.supremecourt.or.th). There are 110,982 original Thai Supreme Court sentences in total from two resources published from 1928 to 2009. The 5,100 completed court sentences were extracted from the original court sentences. They are published from 1997 to 2009 and covered Thai civil

Copyright by Kasetsart University. All rights reserved

and commercial law. In this process, all completed court sentences were prepared and stored in the database format as shown in Table 4.

Table 4 An example of sentences in Thai Legal Corpus

docno	dknum	dkyear	text	section	recall
255205690	5690	2552	การที่โจทก์จำเลยจดทะเบียนหย่ากัน เพื่อประโยชน์ในการเสียภาษี…	155 193/6	0
255204921	4921	2552	การที่บุตรของจำเลยทั้งเจ็ดคนทำหลักฐาน…	850 852 **1612** **1613** **1750**	**1**
255205886	5886	2552	โจทก์ใช้ทางพิพาทเป็นทางเข้าออกจากที่ดินของโจทก์…	1382 1387 1390 **1750**	**1**

Table 4 illustrates the important fields of each sentence in the Thai Legal Corpus. The *docno* is the sentence number, which derives from *dkyear* and *dknum*. The *text* includes all decision contents from the judges and the *section* is the section number(s) stated in the code law. The last parameter, *recall*, represents the relevant document status. The *recall* parameter depends on the *section* parameter of each sentence. If at least one section of the sentence is between 1435 and 1755, indicating that the document is either Thai civil and or commercial laws, the *recall* parameter is equal to 1 and it means that this sentence is a relevant document of the civil and commercial law. On the other hand, if the sentence is not a relevant document, the *recall* value is equal to 0 and has no any section between 1435 and 1755.

Consider all characteristics of the court sentences which is shown as an example in Table 4, the *docno* number 255205690 is combined from the *dkyear* number 2552 and the *dknum* number 5690. The final judgment content is "การที่โจทก์จำเลยจดทะเบียนหย่ากันเพื่อประโยชน์ในการเสียภาษี…" (the plaintiff and the

Copyright by Kasetsart University. All rights reserved

defendant got divorced for tax benefits…) as shown in *text*. This sentence was justified according to section number 155 and 193/6. The *recall* is equal to 0 because the section number is not in the scope (1435 to 1755). Therefore, this sentence is not a relevant document. On the other hand, the *recall* parameter of sentence number 255204921 is equal to 1 since three sections of this sentence are in the scope. Consequently, this sentence is a relevant document. In the Corpus Preparation process, the 563 (11.04%) relevant sentences are stored in the corpus.

Moreover, each legal term of each sentence was extracted using the *WordSegmentation* function and stored both the legal terms and their term frequency (TF) in database format as shown in Table 5. There are 658,457 legal terms in total from all court sentences in the corpus. In Table 5, the *docno* parameter refers to the document number, and all legal terms and their frequencies are displayed in database format. The TF parameter is used for calculating the total score of each sentence in the *Search* process of the CAO algorithm.

Table 5 The example of term frequency of each legal term in each sentence

docno	**มรดก (succession)**	**สัญญา (contract)**	**สิทธิ (right)**	**หน้าที่ (duties)**	**โอน (assign)**	**...**
255205886	0	7	0	6	1	...
255202220	**66**	**3**	**12**	**5**	**0**	**...**
255110274	10	24	2	28	14	...
255001099	0	11	0	9	22	...
...	...	...	...	...	...	...

For instance, there are the frequencies of legal terms in sentence number 255202220: มรดก (succession) is equal to 66, สัญญา (contract) is equal to 3, สิทธิ (right) is equal to 12, หน้าที่ (duties) is equal to 5, and have no โอน (assign) term in this sentence.

Copyright by Kasetsart University. All rights reserved

2.1.2 The TLlexicon dictionary

A set of Thai legal terms were obtained from TLlexicon dictionary. There are 159 legal terms and 825 relationships covering two areas of Thai Civil and Commercial law: succession law and family law as shown in Table 3.

2.2 Seed ontology building process

A new process for automatic seed ontology construction is proposed in order to spare experts on the bulk of the job. First of all, this process is started and seed ontology is created automatically using both a set of random sentences and the TLlexicon dictionary as resources. In order to create a seed ontology, the first term should be the child node of *Thing* term. Note that, each node of the legal ontology is assumed to be a legal term of TLlexicon dictionary.

Algorithm 4 *SeedOntologyBuilding*

Input: ThaiLegalCorpus, TLlexicon
Begin
RndDoc=RandomDoc(ThaiLegalCorpus)
For each $Doc_i \in$ RndDoc
Terms ← WordSegmentation(Doc_i)
For each $t_i \in$ Terms
$Superclass_i$←Explore(TLlexicon(t_i))
If $Superclass_i$=='Thing'or NodeInOntology
Then JoinConcept (t_i,Ontology)
If OntologyExpanded==true
Then Stop
End For
End
Output: SeedOntology

In Algorithm 4, given 5,100 completed sentences in Thai Legal Corpus, the 50 court sentences are randomly selected by the *RandomDoc* function. Next, the first random sentence is loaded and each legal term is extracted using the *WordSegmentation* function. Then, the *Superclass* of each term is explored from the TLlexicon using the *Explore* function. If the *Superclass* is *Thing* or a node in the seed

Copyright by Kasetsart University. All rights reserved

ontology, this node will be connected to the ontology using the *JoinConcept* function. When the *OntologyExpanded* function returns true, the meaning is that the seed ontology can be constructed. Next, the *SeedOntologyBuilding* algorithm will be terminated and the seed ontology will be used in the Thai Legal Ontology as an initial ontology with four parameters (Table 6). On the other hand, if the *OntologyExpanded* function returns false, it means that the first random sentence cannot create a seed ontology. After that, the next selected sentence will be loaded and all systems will restart.

Table 6 The four initial parameters in the legal ontology

Superclass	**Term**	**ph**	**count**	**weight(τ)**	**prob**
Thing	มรดก (succession)	100	1	0	0
มรดก (succession)	สัญญา (contract)	100	1	0	0
มรดก (succession)	สิทธิ (right)	100	1	0	0
มรดก (succession)	หน้าที่ (duties)	100	1	0	0
มรดก (succession)	โอน (assign)	100	1	0	0

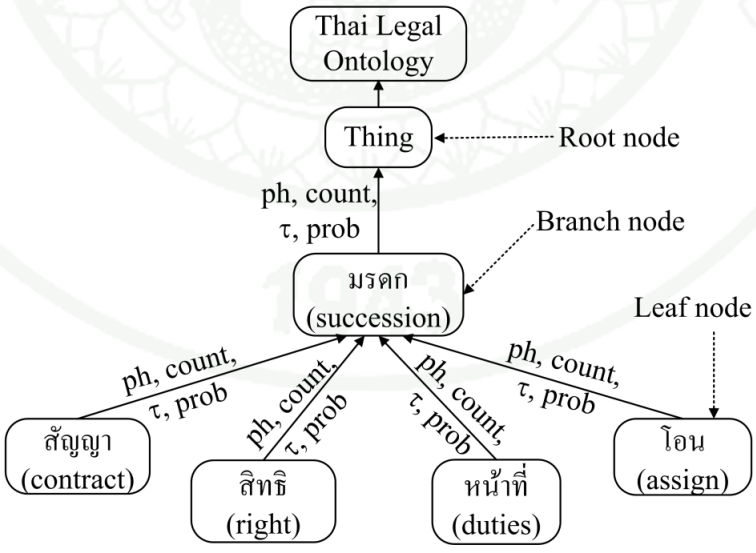


Figure 15 Example of seed ontology with three kinds of node in the ontology

Copyright by Kasetsart University. All rights reserved

Consider Table 3 and Table 5, we found that the first document (document number 255205886) cannot be used to create a seed ontology because it does not has any *root* term (*Thing* node). On the contrary, the seed ontology is constructed by the second document (sentence number 255202220) because it has *'Thing'* node which is a root node of มรดก (succession) term. As a result, the example of the seed ontology is formed using the ATOB algorithm as shown in database structure (see Table 6), tree structure (see Figure 15) and XML document (see Figure 16).

```
<?xml version="1.0" encoding="UTF-8"?>
<TLterms>    <terms>
      <superclass>
            <name>Thing </name>
      </superclass>
      <term>
            <name>มรดก (succession)</name>
            <ph>100</ph>
            <count>1</count>
            <weight>0</weight>
            <prob>0</prob>
      </term>
      <superclass>
            <name>มรดก (succession)</name>
      </superclass>
      <term>
            <name> สัญญา (contract)</name>
            <ph>100</ph>
            <count>1</count>
            <weight>0</weight>
            <prob>0</prob>
      </term>
      ...
</terms>      </TLterms>
```

Figure 16 Example of ontology in the XML document

Copyright by Kasetsart University. All rights reserved

2.3 Ontology expansion process

A new methodology for automatic ontology expansion is proposed using three data sources: the seed ontology obtained from the Thai Legal Ontology, the TLlexicon dictionary and a list of retrieved sentences called *LegalDocRanked* which obtained from the CAO algorithm. The legal terms are extracted from the first sentence of the *LegalDocRanked* and used for expanding the ontology using the *Ontology Expansion* process (see Figure 14). This process is terminated automatically using the threshold parameter.

Through the following example, in the *Ontology Expansion* process, if $term_i$ cannot be found in the ontology, the algorithm will traverse in the TLlexicon and obtain its parent ($Superclass_i$) term. After that, the $Superclass_i$ is explored in the ontology again. Finally, if the $Superclass_i$ is found in the ontology, $term_i$ can immediately be connected to the ontology (see Figure 17). For example, the term 'G' cannot be found in the original ontology, the algorithm explores the TLlexicon and founds that the *Superclass* of term 'G' is term 'C'. So, term 'G' is connected to the ontology through term 'C'.

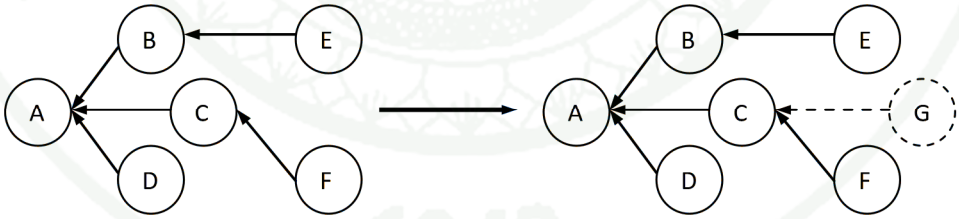


Figure 17 Example of the *Ontology Expansion* process

Algorithm 5 shows the *Ontology Expansion* process, which use a set of retrieved sentences from the CAO algorithm as an input (*LegalDocRanked*). The *status* (threshold) parameter is the percentage of the total number of first-level child nodes of the keyword (query) in the ontology and the total number of first-level child nodes of the keyword in TLlexicon. The *threshold* parameter is set to 0.8. This means

Copyright by Kasetsart University. All rights reserved

that the ontology can be expanded until the number of keyword reach 80% of the total first-level child nodes.

Algorithm 5 *OntologyExpansion*

Input: LegalDocRanked ←{Doc_1, Doc_2,…, Doc_n}, Ontology, status
Begin
If status < 0.8 then
For each $Doc_k \in$ LegalDocRanked
Terms ← *WordSegmentation*(Doc_k)
For each $t_i \in$ Terms
$Superclass_i$←*Explore*(TLlexicon(t_i))
OntologyTraversal(t_i,$Superclass_i$,Ontology)
If *NotFoundLink*(t_i,$Superclass_i$)
JoinConcept(t_i,$Superclass_i$,Ontology)
If (*OntologyExpanded*==true)
Then Stop
Return OntologyUpdated
End.
Output: OntologyUpdated

In case that the *status* parameter of the keyword is less than 0.8, the first document list from the *LegalDocRanked* will be loaded and the *WordSegmentation* function is performed in order to extract legal terms. Each legal term is explored in the TLlexicon dictionary for their *Superclass* using the *Explore* function. Then both the legal term and its *Superclass* are traversed in the ontology again by the *OntologyTraversal* function. If the *NotFoundLink* function returns true, it means that the legal term and its *Superclass* are not joined together in the ontology even if two terms are appeared in the ontology. After that, two terms will be connected using the *JoinConcept* function. Moreover, in case that the *Superclass* is found in the ontology whereas the legal term is not found, the *JoinConcept* function is done by adding the legal term to the ontology. This is, the ontology can be expanded or the *OntologyExpanded* function returns true. Next, the ontology expansion process is terminated and immediately returns the updated ontology to the Thai Legal Ontology.

On the other hand, if the *status* parameter of the keyword is equal to or more than 0.8, the *Ontology Expansion* process will be ignored. Moreover, if the *NotFoundLink* function returns false, it means that both the legal term and its

Copyright by Kasetsart University. All rights reserved

Superclass are already connected in the ontology. If all of the legal terms extracted from the first document cannot be used for expanding the ontology, the *OntologyExpanded* function still returns false. Then the second document in the *LegalDocRanked* is loaded and the *Ontology Expansion* processes will be repeated.

3. CAO algorithm

After the ontology is automatically created by the ATOB algorithm, it is used as a knowledge base in the CSR process. Moreover, all parameters in the ontology i.e. *ph*, *count*, *weight* and *prob* are updated by the CAO algorithm.

The CAO algorithm which combines the ant colony algorithm and Thai legal ontology is proposed for improving the CSR performance. The combination algorithm is applied in three important processes which are *Query Expansion*, *Parameters Updating* and *Search* process (Figure 14).

The first process is the *Query Expansion* which used for calculating the *probability* value of each link between legal terms and their relationships. Moreover, this process is used for proposing the good candidate terms to user and adding the chosen term(s) to his requirement. In the *Query Expansion* process, the query is given by the legal user and traversed through the Thai Legal Ontology (Table 6). Note that, the Thai Legal Ontology is used as a knowledge base and query expansion tool. Moreover, all first level child nodes of the query term are descendent sorted by their *probability* value. The ant colony algorithm has effect on the ranking of candidate terms. The second process is the *Parameters Updating* which used for updating the *ph, count*, *weight* or *pheromone* (τ) and *probability* (*prob*) value of each link between the query term and their candidate terms. The ant colony algorithm is applied for calculating the τ and *prob* value which have effect on the retrieved sentences. Finally, the document score is calculated in the *Search* process. Interestingly, the ant colony algorithm is applied for retrieving the relevant, diversified and updated sentences in this process. The result shows that, the relevant and diversified sentences are still retrieved in the better order list than the Baseline method.

Copyright by Kasetsart University. All rights reserved

Algorithm 6 *CAOAlgorithm*

Input: Query
Ontology ← $P_{(Q,QE_i)}$ using Eq. (7)
If *SearchConcept* (Query,Ontology) **Then**
    CandidateTerms ← *Query Expansion* (Query,Ontology)
    QE ← *ChosenTerm* (CandidateTerms)
        If ($term_i$==QE)
            Parameters Updating (pheromone accumulation) using Eq. (8)
        Else
            Parameters Updating (pheromone evaporation) using Eq. (9)
        IF ($ph_{(Q,QE_i)}$ == 0) **Then**
            DeleteLink ($term_i$,Ontology)
        EndIf
    $DocScore^k$ ← *Search* (Q,QE_i,ThaiLegalCorpus) using Eq. (10)
    LegalDocRanked ← *Sorting* (DocScore)
EndIf.
Output: LegalDocRanked

Algorithm 6 shows the CAO algorithm which is iterated after the query is entered by the user. Each iteration, the probability value of each link of query term in the ontology is updated using equation (7). All candidate terms which related to the query are sorted by the probability value and proposed to the user. In case that the user may choose one or more terms for his query expansion term, the τ of the chosen terms are increased using the equation (8) whereas the τ of the non-chosen terms are decreased using equation (9). Finally, the document score of each related sentences are calculated using equation (10). Then the relevant and updated sentences are stored in the *LegalDocRanked* parameter.

In Table 6, the *Superclass* denotes the root node or the branch node in the legal ontology whereas the *Term* indicates the child node of its *Superclass*. Initially, the *ph* parameter is set to 100. This parameter will be increased, if the *Term* which proposed to the user is chosen for his query expansion term. On the other hand, the *ph* parameter will be decreased, in case that the *Term* is not chosen for the query expansion term. The *Term* and its *Superclass* will be disappeared from the ontology, if the *ph* parameter is equal to 0. It means that, the candidate term is not chosen for query expansion term for long period of time. This situation is compared with the ant

Copyright by Kasetsart University. All rights reserved

behaviors. The pheromone evaporation process helps the users to avoid the incorrect path (query expansion term). Next, the *count* parameter is the amount of time which the *Term* is chosen for query expansion term. Initially, the parameter is equal to 1, however, it will be increased if the *Term* is chosen for the query expansion term. On the contrary, the *count* parameter is ignored if the *Term* is a not chosen term. In the situation of ant, *count* parameter represents the number of ants those choose the same path. Lastly, the τ parameter (*weight ontology*) is the weight of each link between the query term and the candidate terms (the first level child node of the query term). The τ parameter of both chosen and non-chosen term is recalculated when the *Term* is proposed to the user. This parameter will be increased, if the *Term* is chosen for the query expansion term whereas it will be decreased, if the *Term* is ignored for the expansion term. The τ parameter is used as the pheromone trails of real ants which deposited or evaporated by ant.

2.1 Query expansion process

The *Query Expansion* process plays a crucial role in the CAO algorithm for adding term(s) to the original query in order to obtain more refined and relevant results. The aim of this process is to rank the candidate legal term(s) by their probability values and to propose them to the user. Note that, the candidate legal terms are obtained from Thai Legal Ontology (Boonchom and Soonthornphisaj, 2010a, 2010b, 2012). The Thai Legal Ontology is a main knowledge base used for fulfills and gains the complete Thai legal knowledge. The structure of the Thai Legal Ontology is shown in Table 6. In this process, the ant colony algorithm is applied for calculating the probability value of each path between the query term and the candidate legal term(s). Normally, ants can find the shortest path between their nest and food source using the pheromone trail. The ants prefer the path with more pheromone trail deposited and the pheromone level is increased more rapidly on shortest path. The concept of ant colony algorithm shows that, ants can rank the pheromone level of each path by itself. Therefore, the capable of ranking the important path of ant colony algorithm is used for ranking the candidate legal term by their probability value. Figure 18 shows that the candidate legal terms are sorted by

Copyright by Kasetsart University. All rights reserved

the probability value which are CT_4, CT_3, CT_2 and CT_1 respectively. The sorted candidate legal term(s) are proposed to the user for expanding his requirement.

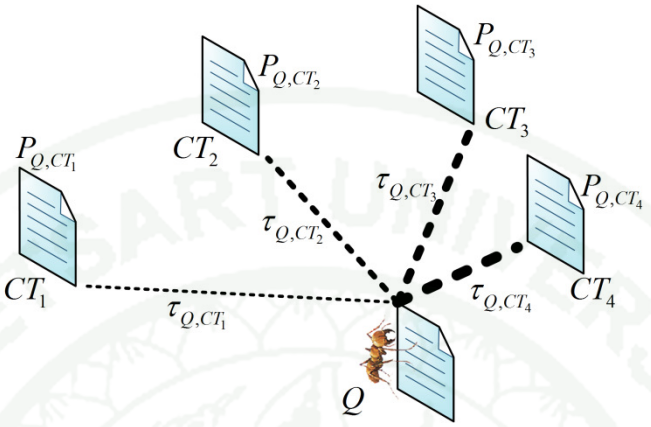


Figure 18 The candidate legal term(s) are ranked by the probability value

In the *Query Expansion* process, users randomly enter a query (Q) to the *Query Expansion* process depends on his requirement. Then the query term is traversed through the ontology. In case that the query term is found in the ontology, this function is performed for determining the type of node for root or branch node. All first level child nodes of the query node are used as the candidate legal terms. The probability value (P) of all paths between the query term and the candidate legal terms are calculated using equation (7).

$$P_{(Q,CT_i)} = \frac{\tau_{(Q,CT_i)} * \lambda_{(CT_i)}}{\sum_{j=1}^{n} \tau_{(Q,CT_i)} * \lambda_{(CT_j)}}, \tag{7}$$

$$\text{where } \tau_{(Q,CT_i)} = \frac{C_{(Q,CT_i)}}{\sum_{j=1}^{n} C_{(Q,CT_j)}}, \qquad \lambda_{(CT_i)} = TF'_{CT_i} * IDF'_{CT_i}$$

where

$P_{(Q,CT_i)}$ Probability value between the query term (Q) and the candidate term$_i$ (CT_i)

$\tau_{(Q,CT_i)}$ Pheromone level between Q and CT_i

Copyright by Kasetsart University All rights reserved

$\lambda_{(CT_i)}$ Importance level of the CT_i

$C_{(Q,CT_i)}$ The count parameter between Q and CT_i

TF'_{CT_i} $\dfrac{term\ frequency\ of\ CT_i\ in\ the\ corpus}{\max\ frequency\ in\ the\ corpus}$

IDF'_{CT_i} $\log\left(\dfrac{the\ total\ number\ of\ the\ documents\ in\ the\ corpus}{the\ number\ of\ documents\ which\ CT_i\ appear}\right)$

The candidate legal terms are descending sorted by their probability value and proposed to the user. It means that, the more related term of the original query term is ranked in the top of the list. Thus, user may select the query expansion term(s) which related to his requirement.

Consider Table 6 and Table 7, suppose that the มรดก (succession) is used as a query term, it is explored through the ontology and it is found in the ontology. All of the first level child nodes of the มรดก (succession) are used as candidate legal terms which are สัญญา (contract), สิทธิ (right), หน้าที่ (duties) and โอน (assign) node (see Table 6). The probability value of each path between the query term and their candidate legal terms are calculated using equation (7). Note that, the maximum frequency is 67,316 legal terms and 5,100 sentences in the corpus. The probability value of $P_{(succession,contract)}$ is equal to 0.442263, $P_{(succession,right)}$ is equal to 0.145169, $P_{(succession,duties)}$ is equal to 0.139625 and $P_{(succession,assign)}$ is equal to 0.272943 as shown in Table 8. Then, the candidate legal terms are sorted by the probability value as follows: สัญญา (contract), โอน (assign), สิทธิ (right) and หน้าที่ (duties) (see Figure 19). It means that the candidate legal term with high probability value is ranked in the top of the list and it is more associated to the query term.

Copyright by Kasetsart University. All rights reserved

Table 7 The example of the corpus which consists of total legal term frequency and the number of document which legal terms occurred

Legal terms	**มรดก (succession)**	**สัญญา (contract)**	**สิทธิ (right)**	**หน้าที่ (duties)**	**โอน (assign)**	**…**
Number of documents which legal terms occurred	708	3222	4196	2161	1585	…

Table 8 The updated parameters in the ontology

Superclass	**Term**	**ph**	**count**	**weight(τ)**	**prob**
Thing	มรดก (succession)	100	1	0	0
มรดก (succession)	สัญญา (contract)	100	1	0.25	0.442263
มรดก (succession)	สิทธิ (right)	100	1	0.25	0.145169
มรดก (succession)	หน้าที่ (duties)	100	1	0.25	0.139625
มรดก (succession)	โอน (assign)	100	1	0.25	0.272943

Moreover, the τ parameter of each arc between the query term and the candidate legal terms are updated as follows: $\tau_{succession,contract}$, $\tau_{succession,right}$, $\tau_{succession,duties}$ and $\tau_{succession,assign}$ are equal to 0.25 (Table 8). Note that the τ parameter is an important component of the ontology which used for calculating the document score and retrieving the relevant sentences in the *Search* process.

A user interface in Figure 19 shows the good candidate legal terms which obtained from the ontology. For example, the มรดก (succession) is used as a query term and all good candidates legal terms are retrieved, ranked by the probability value and proposed to the user.

Copyright by Kasetsart University. All rights reserved

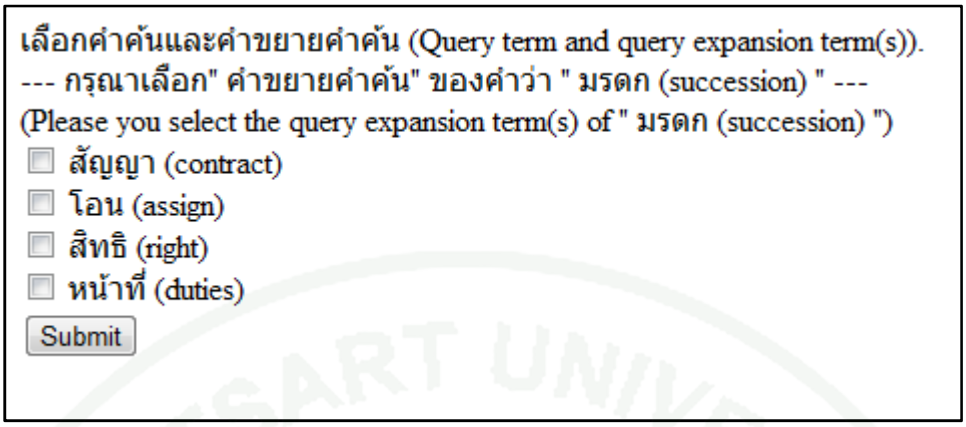


Figure 19 The example of the user interface of the *Query Expansion* process

In case that the query term is not found in the ontology or the query term is the leaf node in the ontology, the *Query Expansion* process is ignored and only one original query term is directly used for calculating the document score in the *Search* process.

2.2 Parameters updating process

The pheromone trails or weight (τ) of each path between original query and all candidate legal terms are recalculated when the candidate terms are proposed to the user. In case of the chosen term, the τ parameter will be accumulated using equation (8). On the contrary, if the candidate legal terms are not chosen term, the τ value will be evaporated using equation (9) (see Table 9).

2.2.1 The accumulation function

In the ant colony algorithm, ants prefer the path with more pheromone trail. Therefore, the $\tau_{(Q,QE_i)}$ parameter which is embedded in the arc between the query term and the query expansion terms (QE_i) are updated using equation (8). In case of the high τ value, it means that the term is chosen for many times by the users and this term is likely to be more related to the query term.

Copyright by Kasetsart University. All rights reserved

$$\tau_{(Q,QE_i)} \leftarrow \tau_{(Q,QE_i)} + \Delta\tau_{(Q,QE_i)} \quad (8)$$

$$\text{where } \Delta\tau_{(Q,QE_i)} = \frac{C_{(Q,QE_i)}}{\sum_{j=1}^{n} C_{(Q,QE_j)}} - \tau_{(Q,QE_i)}$$

where

$\tau_{(Q,QE_i)}$ Pheromone level between query term (Q) and the query expansion term$_i$ (QE_i)

$\Delta\tau_{(Q,QE_i)}$ Amount of accumulation pheromone level between Q and QE_i

$C_{(Q,QE_i)}$ The count parameter between Q and QE_i

2.2.2 The evaporation function

The evaporation pheromone in the ant colony algorithm helps to avoid the incorrect path. Thereby, the pheromone in the link between the query term and the non-chosen term ($\tau_{(Q,CT_i)}$) will be decrease using equation (9).

$$\tau_{(Q,CT_i)} \leftarrow \tau_{(Q,CT_i)} - \left((1-\rho)\tau_{(Q,CT_i)}\right) \quad (9)$$

$$\text{where } \rho = \frac{C_{(Q,CT_i)}}{\tau_{(Q,CT_i)} \sum_{j=1}^{n} C_{(Q,CT_j)}}$$

Table 9 The updated *ph, count, weight* and *prob* parameters in the ontology

Superclass	**Term**	**ph**	**count**	**weight(τ)**	**prob**
Thing	มรดก (succession)	100	1	0	0
มรดก (succession)	สัญญา (contract)	101	2	0.4	0.613291
มรดก (succession)	สิทธิ (right)	99	1	0.2	0.100654
มรดก (succession)	หน้าที่ (duties)	99	1	0.2	0.096809
มรดก (succession)	โอน (assign)	99	1	0.2	0.189246

Copyright by Kasetsart University All rights reserved

Suppose that the สัญญา (contract) is a chosen term for a query expansion term whereas สิทธิ (right), หน้าที่ (duties) and โอน (assign) are not selected. The *ph* parameter of all candidate terms is updated. The *ph* parameter of the chosen term is increased from 100 to 101 whereas the *ph* parameter of the non-chosen term is decreased from 100 down to 99 (Table 9). If the *ph* parameter is equal to 0, it means that the link between *Superclass* and *Term* will be destroyed from the ontology because the candidate term is not chosen for long period of time by the user. Furthermore, the *count* parameter between มรดก (succession) and สัญญา (contract) is increased from 1 to 2. This parameter of the non-chosen term still equal to 1 since this process is ignored for the non-chosen term. Then, the $\tau_{(succession,contract)}$ parameter is accumulated and equal to 0.4 using equation (8) whereas $\tau_{(succession,right)}$, $\tau_{(succession,duties)}$ and $\tau_{(succession,assign)}$ is evaporated and equal to 0.2 using equation (9). Finally, the equation (7) is recalculated for a new probability value (*prob*) of each arc (see Table 10).

2.3 Search process

In case that at least only one candidate legal term is chosen for the query expansion term, three parameters are used for calculating the document score using equation (10) which are the TF, the inverse document frequency (IDF) of both query term and the query expansion term, and the τ parameter between the query and the query expansion term.

$$DocScore^k = \left(\xi_Q^k + \sum_{i=1}^{n} \xi_{Q,QE_i}^k \right) \Delta\tau^k \qquad (10)$$

where $\xi_Q^k = \begin{cases} \left(TF_Q^k * IDF_Q^k\right)10 & if\ TF_Q^k > 0 \\ 0 \end{cases}$

Copyright by Kasetsart University All rights reserved

$$\xi^{k}_{Q,QE_i} = \begin{cases} \left(TF^{k}_{QE_i} * IDF^{k}_{QE_i}\right)\tau_{(Q,QE_i)} & if\ TF^{k}_{QE_i} > 0 \\ 0 \end{cases}$$

$$\Delta\tau^{k} = 1 + \frac{1}{d^{k}}$$

where

$DocScore^{k}$	Document score of document *k*
ξ^{k}_{Q}	Local document score from the query term (Q) of document *k*
ξ^{k}_{Q,QE_i}	Local document score from the query expansion term$_i$ (QE_i) of document *k*
TF^{k}_{Q}	*the number of occurrences of the query term*
IDF^{k}_{Q}	$\log\left(\frac{the\ total\ number\ of\ the\ documents\ in\ the\ corpus}{the\ number\ of\ documents\ in\ which\ Q\ term\ appear}\right)$
$\Delta\tau^{k}$	Up-to-date value of the document *k*

Table 10 The example of legal term frequencies of each sentence

docno/term frequency	**มรดก (succession)**	**สัญญา (contract)**	**สิทธิ (right)**	**หน้าที่ (duties)**	**…**
255202780	8	22	34	8	…
255100850	14	0	7	0	…
255108239	0	12	17	10	…
…	…	…	…	…	…

Suppose that, Table 10 contains the TF of each legal term from each sentence in the repository whereas Table 7 shows the number of documents those have legal term occurred. The TF of the query term (มรดก (succession)) and the query expansion term (สัญญา (contract)) are used for calculating the document score in the *Search* process using equation (10). The final results are descendent sorted in order by the document score as follows: the first rank is the sentence number 255100850 with

Copyright by Kasetsart University All rights reserved

the document score is 180.08, the second listed sentence is number 255202780 with the document score is 140.72 and the last ranked sentence is number 255108239 with the document score is 1.44 (Figure 20).

In case that the query term is not found in the ontology or it is a leaf node in the ontology or the user do not choose any candidate legal term for his query expansion term, the TF and the IDF of only the original query term are used for calculating the document score from the related sentence using equation (10)

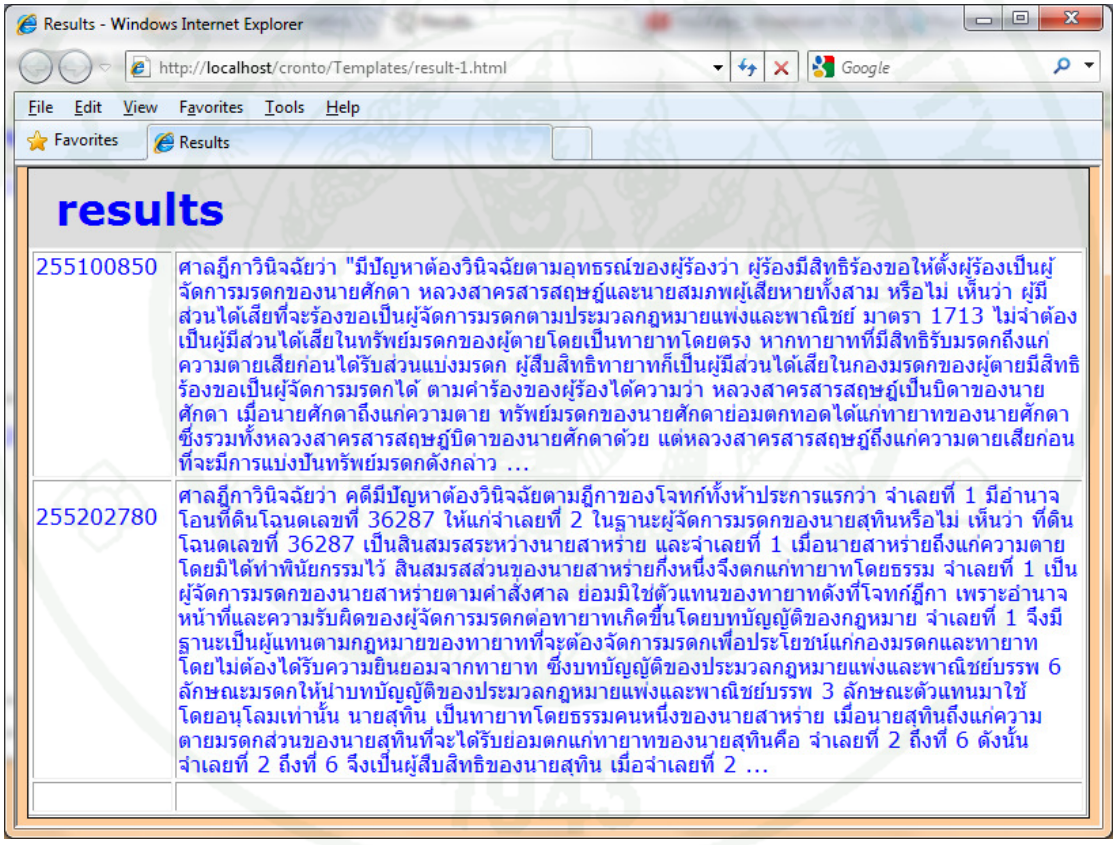


Figure 20 The example of the user interface of the *Search* process

Copyright by Kasetsart University All rights reserved

RESULTS AND DISCUSSION

Evaluation Methods

To evaluate the ATOB algorithm which used for automatic creating the legal ontology and the CAO algorithm which used for improving the CSR performance, the efficiency of these systems were based on test cases in the domain of Thai law. In this study, the performance of the ATOB and CAO algorithm are evaluated using eight queries which are the มรดก (succession), ทายาท (heir), บุตร (child), พินัยกรรม (will), ผู้จัดการมรดก (administrator of an estate), สมรส (marriage), หย่า (divorce) and ผู้สืบสันดาน (descendant) term. All queries are given to twenty six law experts. They randomly pick up the query and both the ATOB and CAO algorithm are applied. In order to evaluate the ATOB algorithm, the inputs obtained from the Corpus Preparation process are assumed to be correct sentences. Moreover, a legal term is assumed as a node in the ontology. The performance of the legal ontology which automatically created by ATOB algorithm is compared with the Thai legal ontology which manually created by the expert. In addition, the ontology created by the ATOB algorithm is used as knowledge base in the CSR process. In order to evaluate the performance of the CAO algorithm in the CSR process, it is compared with the Baseline method which developed by the Supreme Court of Thailand. The standard well-known measurements are used to evaluate the ATOB and CAO algorithm in different aspects such as precision, recall, F-measure and diversity value.

Precision is a value in the [0, 1] range which is the proportion between the number of relevant documents retrieved by the CAO algorithm (true positives) and the total number of retrieved documents (both true positives and false positives). Poor precision value means that users may have to face a large of incorrect information. The value of recall is in [0, 1] range. This value is the ratio of relevant documents retrieved by the CAO algorithm (true positives) over the total number of correct documents in the corpus (both true positives and false negatives). The poor recall

Copyright by Kasetsart University. All rights reserved

means that much correct and good information is missed. However, perfect performance would be 100% precision and 100% recall, it means that all possible relevant documents are retrieved while without any incorrect documents is retrieved. F-measure is a value that varies in the [0, 1] range. It is the harmonic mean of precision and recall. Moreover, all sections of relevant sentences are used for calculating the diversity measure which varies in [0, 1] range. Consider two court sentences, we extract the section numbers which were applied by the judge for each sentences. Therefore set A contains the section numbers appeared in the first rank sentence. Set B contains the section numbers appeared in the second rank sentence. The diversity value is calculated to see the difference between these section numbers. In case of the high diversity value, it means that most retrieved sentences contain different sections in the document. On the other hand, if most of retrieved sentences contain similar sections of law, the diversity value will be low. In general, there are many indicators used for calculating the diversity value. In this research, the Jaccard measure is used in order to investigate the diversity performance of CAO algorithm.

$$precision = \frac{number\ of\ relevant\ sentences\ retrieved\ by\ the\ CAO}{total\ number\ of\ \ sentences\ retrieved\ by\ the\ CAO}$$

$$recall = \frac{number\ of\ relevant\ sentences\ retrieved\ by\ the\ CAO}{total\ number\ of\ \ correct\ sentences}$$

$$F-measure = \frac{2*precision*recall}{precision+recall}$$

$$Jaccard\_diversity(A,B) = 1 - \frac{|A \cap B|}{|A \cup B|}$$

Here is an example for calculating the diversity value, suppose Table 4 is used as retrieved sentences. All sections from the relevant sentences are considered. There are nine sections in total from the relevant sentences whereas there are eight individual sections in total. So, the diversity value is equal to 0.89. It means that, the most relevant retrieved sentences rather contain different sections.

Copyright by Kasetsart University. All rights reserved

Results and Discussion

The experimental results are summarized according to two algorithms: ATOB and CAO algorithm. Note that, ATOB and CAO algorithm are performed consequently (ATOB algorithm is used in the ontology building process whereas the CAO algorithm is used in the information retrieval process).

Table 11 Experimental setup

Set of Experiments	Seed Ontology Building		Retrieval		Ontology Expansion	
	Automatic (ATOB)	**Manual**	**With ant (CAO)**	**With out ant**	**Automatic stop (ATOB)**	**Non-automatic stop**
1) Baseline	-	-	-	-	-	-
2) Set A	✓	**-**	✓	**-**	✓	**-**
3) Set B	✓	-	✓	-	-	✓
4) Set C	✓	-	-	✓	✓	-
5) Set D	✓	-	-	✓	-	✓
6) Set E	-	✓	✓	-	✓	-
7) Set F	-	✓	✓	-	-	✓
8) Set G	-	✓	-	✓	✓	-
9) Set H	-	✓	-	✓	-	✓

In order to find the appropriate setup, several experiments were done (see Table 11). The performance of the ATOB and CAO algorithm were explored in three aspects; Seed ontology, Retrieval and Ontology expansion process. The seed ontology created by ATOB algorithm called *automatic* method and the seed ontology manually built by the expert called *manual* method. Furthermore, in the retrieval process, ant colony algorithm is embedded in the *weight ontology* for improving the retrieval process called *with ant* (CAO) method. In case of *without ant* method, the concept of ant colony algorithm is ignored in the *Query Expansion*, the *Parameters Updating* and *Search* process. Moreover, the ontology expansion process is divided into two techniques; the algorithm can be automatically terminated using the threshold parameter, called *automatic stop* (ATOB) method and the expansion process will be

Copyright by Kasetsart University. All rights reserved

terminated when the first level child nodes of the keyword is completed called *non-automatic stop* method.

To evaluate the performance of seed ontology, the appropriate technique is considered in different aspects (*automatic* (ATOB) vs. *manual*). The effect of ant colony algorithm in the retrieval process (*with ant* (CAO) vs. *without ant*) was examined among these experiments. Furthermore, the ontology expansion process is evaluated to compare between the *automatic stop* (ATOB) and the *non-automatic stop*.

Note that, Baseline method is the keyword search or the keyword matching method (www.supremecourt.or.th). The keyword matching method uses the user query (keyword) to search through the document. The objective of the keyword matching method is to filter and retrieve all of documents which has at least only one matched term in the document. For example, suppose that the "มรดก (succession)" is used as a keyword. The Baseline method used the "มรดก (succession)" for searching through all documents in the corpus. If the "มรดก (succession)" is appeared in the document at least only one time, this document will be retrieved. In addition, the results from the Baseline system are descendent sorted by the sentences number.

The top 30 legal sentences retrieved by the retrieval process in different setup are evaluated to compare with the Baseline method. All experimental finding values are shown in Table 12.

Empirical result shows that set A obtain the best precision, recall and *F*-measure value and diversity value. Notice that, set A, B, E and F give high performance in terms of precision, recall, *F*-measure and diversity because these four experiments apply the ant colony algorithm. Against the Baseline, set C, D, G and H, give poor performance because they are without the ant colony algorithm. This study found that, set A outperforms all experimental setups because this set consists of both

Copyright by Kasetsart University. All rights reserved

ATOB and CAO algorithm. Moreover the advantage of set A is to save processing time and reduce the expert's task for manual seed ontology. Since ATOB algorithm can be used for automatic seed ontology construction, automatic stop ontology expansion and CAO algorithm can be used for retrieving the relevant, qualified, diversified and updated sentences from the CSR process. To conclude, set A which consists of the ATOB and CAO algorithm is the most suitable experimental setup for automatic Thai legal ontology building and improving the Thai CSR performance.

Table 12 The overall performance

Experiment concept / measurement	Precision	Recall	F-measure	Diversity	Nodes	% of nodes	Relations	% of relations
Baseline	0.39	0.47	0.43	0.15	-	-	-	-
Set A	**0.95**	**0.94**	**0.94**	**0.40**	**121**	**76.10**	**653**	**79.15**
Set B	0.91	0.93	0.92	0.37	155	97.48	815	98.79
Set C	0.56	0.57	0.56	0.30	123	77.36	638	77.33
Set D	0.67	0.69	0.68	0.29	157	98.64	816	98.91
Set E	0.91	0.92	0.91	0.36	119	74.84	648	78.55
Set F	0.93	0.94	0.93	0.36	154	96.86	809	98.06
Set G	0.58	0.60	0.59	0.27	125	78.62	671	81.33
Set H	0.75	0.75	0.75	0.33	157	98.74	821	99.52

Considering the number of nodes and relations, there are totally 159 nodes and 825 relations in the dictionary. Set B, D, F and H have more number of nodes and relations than set A, C, E and G because they use the non-automatic stop technique in the ontology expansion process. Nonetheless, the size of ontology has no effect on the ontology performance. The percentage of nodes and relations is a ratio between the number of nodes and relations of each set and the number of nodes and relations of the TLlexicon. Moreover, there are 106 (67.52%) nodes and 564 (68.7%) relations which constructed by all set whereas 51 (32.48%) nodes and 257 (31.3%) relations were built by some set. It means that, the popular nodes and relations are used for ontology construction especially the branch nodes.

Copyright by Kasetsart University. All rights reserved

The performance of the ATOB and CAO algorithm are tested and considered in detail as follow:

1. ATOB algorithm

The ATOB algorithm which is used for automatic ontology building consists of three processes: the *Corpus Preparation*, the *Seed Ontology Building* and the *Ontology Expansion* process. In the results and discussion, both the *Seed Ontology Building* and the *Ontology Expansion* process are tested.

1.1 The Seed Ontology Building

Concerning the methodology for a seed ontology building, there are two methods which are *manual* method and *automatic seed ontology building* method.

The *manual seed ontology building* method is manually created by the domain expert. The seed ontology consists of nine nodes, eight relationships and four initial parameters as shown in Table 13 and Figure 21.

Table 13 Initial seed ontology parameters created by *manual seed ontology building* method

Superclass	**Term**	**ph**	**count**	**weight(τ)**	**prob**
Thing	มรดก (succession)	100	1	0	0
Thing	ครอบครัว (family)	100	1	0	0
Thing	ตาย (dead)	100	1	0	0
ครอบครัว (family)	สมรส (marriage)	100	1	0	0
มรดก (succession)	พินัยกรรม (will)	100	1	0	0
มรดก (succession)	ทรัพย์ (property)	100	1	0	0
มรดก (succession)	ทายาท (heir)	100	1	0	0
ทายาท (heir)	ผู้สืบสันดาน(descendent)	100	1	0	0

Copyright by Kasetsart University All rights reserved

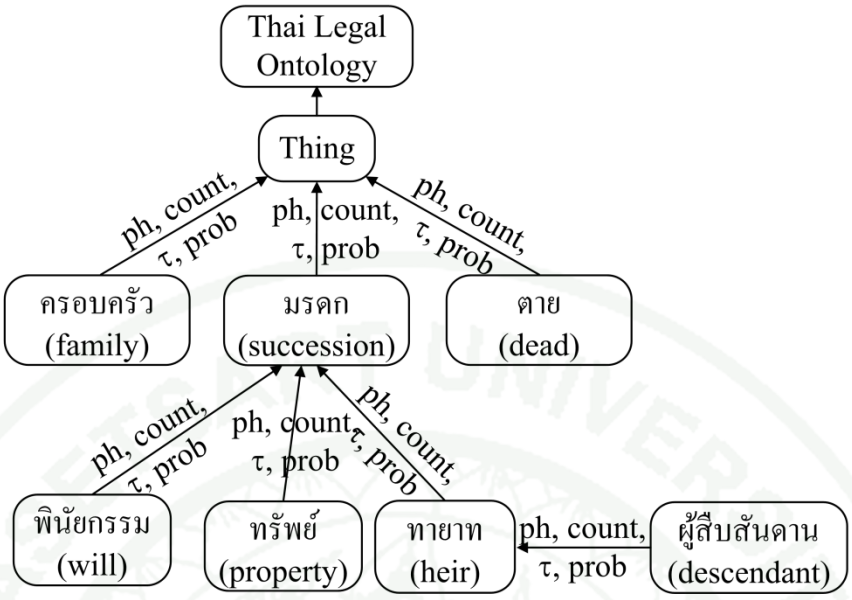


Figure 21 The structure and four parameters of seed ontology created by *manual seed ontology building* method

The *automatic seed ontology building* method is automatically created by the ATOB algorithm. There are 14 set for seed ontology testing which obtained from the ATOB algorithm and shown in Table 14.

Table 14 The seed ontologies which are obtained from the ATOB algorithm

No.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**Avg.**
Nodes	15	6	23	15	8	28	7	14	15	17	7	19	22	20	15
Relations	27	5	44	27	10	54	9	19	30	25	10	44	39	42	28

There are totally 56 nodes and 130 relations which extracted from the random sentences and used for the *automatic seed ontology building* process. The average number of nodes and relations are 15 and 28, respectively. It means that, the seed ontology has an appropriate size. Considering the structure of the test seed ontology, the 11 (19.64%) nodes and 16 (12.31%) relations are used in 7 (50%) set of the seed ontology testing. It means that, the different concept of seed ontologies

Copyright by Kasetsart University. All rights reserved

depend on the selected sentence. In this research, the minimum seed ontology is selected. It consists of six nodes, five relationships and four initial parameters as shown in Table 6 and Figure 15.

In order to investigate the efficiency of the seed ontology, there are two methods for the seed ontology constructing; *automatic* (ATOB) and *manual* method (see Table 15 for detail). The performances obtained from the *automatic* seed ontology construction are calculated from the average performance of set A, B, C and D whereas the performances obtained from the *manual* seed ontology construction are calculated from the average performance of set E, F, G and H.

Table 15 The performance of ontology created by different seed ontology

Experiment	**Method**	**Precision**	**Recall**	***F*-measure**	**Diversity**	**Avg. nodes**	**Avg. relations**
Baseline	-	0.39	0.47	0.43	0.15	-	-
Seed Ontology	Automatic seed ontology building (ATOB)	0.77	0.78	0.77	**0.34**	139	731
	Manual seed ontology building	0.79	0.80	0.80	0.33	139	737

Empirical results show that the performance of Baseline is worse than those of ATOB algorithm and *manual* method. Consider the seed ontology construction, using automatically seed ontology construction outperforms the *manual* ontology construction in terms of diversity. Note that, the precision, recall and *F*-measure of *automatic* (ATOB) and *manual* method are not significant different performance. Considering the diversity value, the retrieved sentences retrieved from the ontology which created by the ATOB algorithm have more varied sections than the retrieved sentences obtained from the ontology which created by the *manual* method. It means

Copyright by Kasetsart University. All rights reserved

that, the ATOB algorithm can be used for creating a better ontology. Furthermore, the average number of nodes and relations which obtain from both the ATOB and *manual* method are not significant difference. Therefore, the experimental results of set A and set E are used as an example and shown in Table 16.

Table 16 The performance of ontology expanded from seed ontology created by ATOB and *manual* method

Seed ontology method	**Number of**					
	Nodes	**Relations**	**The branch nodes**	**The leaf nodes**	**The different nodes**	**The same nodes**
Automatic seed ontology building	**121**	**712**	**60**	**61**	**6**	115
Manual seed ontology building	119	708	60	59	4	

The results of the test of the seed ontology method are summarized in Table 15. The performance of the ontology created by the *automatic seed ontology building* method (ATOB) outperforms the ontology created by the *manual seed ontology building* method in all features. Considering the number of the nodes and relations in the ontology, the seed ontology created by the ATOB algorithm can be expanded to be a comprehensive ontology. The branch node can be used as a query term and it can be expanded by its child node. Moreover, the leaf node is the child node of the ontology and some of them can be connected to be a branch node. Both ontologies expanded from the seed ontology created by the ATOB and *manual* method have 115 the same nodes. Whereas the six nodes are only created in the ontology which expanded from the seed ontology created by the ATOB i.e. จำคุก (imprison), จำหน่าย (sell), บรรลุนิติภาวะ (legal age), ป้า (aunt), ละทิ้ง (abandon) and

Copyright by Kasetsart University. All rights reserved

ลายนิ้วมือ (fingerprint), the four nodes are only created in the ontology which expanded from the seed ontology created by the *manual* method i.e. ยักย้าย (shift), สังหาริมทรัพย์ (movable property), อภัย (forgive) and อสังหาริมทรัพย์ (immovable property). Considering all of the different nodes, the six nodes only created in the ontology which expanded from the seed ontology created by the ATOB are more important than the four nodes in domain of the TF value.

To conclude, the seed ontology should be created using the ATOB algorithm since this technique is to save processing time and reduce the expert's task whereas the performance of both method have no significantly different performance.

1.2 The Ontology Expansion

Table 17 The performance of ontology in different mechanisms

Experiment	**Method**	**Precision**	**Recall**	***F*-measure**	**Diversity**	**Avg. Nodes**	**Avg. Relations**
Baseline	-	0.39	0.47	0.43	0.15	-	-
Ontology Expansion	Automatic stop ontology expansion (ATOB)	0.75	0.76	0.75	0.33	122	635
	Non-automatic stop ontology expansion	0.82	0.83	0.82	0.34	156	815

The concise ontology is an important parameter in the *Ontology Expansion* process of the ontology building process. There are two methods for evaluating the performance of the ontology in different size which are the *automatic stop* (ATOB) and the *non-automatic stop* method. Notice that in the *automatic stop*

Copyright by Kasetsart University All rights reserved

(ATOB) ontology expansion method, the threshold parameter is more than or equal to 0.8 whereas in the *non-automatic stop* ontology expansion method, the threshold parameter is equal to 1.

The performance of the ontology is investigated and shown in Table 17. The value of the *automatic stop ontology expansion* method calculated from the average measured value of set A, C, E and G using the threshold is equal to 0.8 whereas the *non-automatic stop ontology expansion* method is the average measured value from the experimental set B, D, F and H using the threshold is equal to 1.0 or the first level child nodes of the keyword is completed. Although the average number of nodes and relations of *automatic stop ontology expansion* method and the *non-automatic stop ontology expansion* method are significant difference, their performances are no significant difference. The *automatic stop ontology expansion* method is more efficient than the *non-automatic stop* method in sense of the convergence speed, concise of ontology size and total processing time. So the appropriate *Ontology Expansion* method is the *automatic stop ontology expansion* (ATOB) method.

Table 18 The characteristic of ontology created by different mechanisms

Ontology Expansion	**Number of**					
	Nodes	**Relations**	**The branch nodes**	**The leaf nodes**	**The different nodes**	**The same nodes**
Automatic stop ontology expansion (ATOB)	121	712	60	61	0	121
Non-automatic stop ontology expansion	155	819	76	79	34	

Copyright by Kasetsart University. All rights reserved

Moreover, the example of the ontology created by the *automatic stop ontology expansion* method and the *non-automatic stop ontology expansion* method in the *Ontology Expansion* process (set A and set B) are shown in Table 18.

The result in Table 18 shows that, all parameters from the ontology created by the *automatic stop ontology expansion* (ATOB) method less than the ontology created by the *non-automatic stop ontology expansion* method, since the threshold parameter of the ATOB method is set to 0.8 whereas the threshold parameter of the other method is set to 1. However, the performance of the ontology created by both methods are not significantly different.

This study found that, the ontology created by the *automatic stop ontology expansion* (ATOB) method is more efficient than the *non-automatic stop* method because it is more convergence speed, concise of ontology size and total processing time.

2. The CAO algorithm

When a seed ontology is automatically created by the *Seed Ontology Building* process of the ATOB algorithm, the seed ontology is used as a knowledge base in the *Thai Legal Ontology* and then it is used by the CAO algorithm

The CAO algorithm which combines the ant colony algorithm and the ontology is used for improving the retrieval performance, especially the CSR efficiency. Furthermore, the ontology structure is modified by the ant colony algorithm called *weight ontology* and used in the retrieval process. There are two methods for investigating the performance of the retrieval processes which are the *with ant* (CAO) method and the *without ant* method as shown in Table 19.

Copyright by Kasetsart University. All rights reserved

Table 19 The performance of *with ant* (CAO) and *without ant* in the retrieval process

Experi-ment	Method	Precision	Recall	*F*-measure	Diversity	Avg. nodes	Avg. relations
Baseline	-	0.39	0.47	0.43	0.15	-	-
Retrieval	With ant (CAO)	0.93	0.93	0.93	0.37	137	731
	Without ant	0.64	0.65	0.65	0.30	141	737

The empirical results demonstrate that all measurements from the Baseline performed poorly when literal matching is done for retrieval process, due to synonym, hyponym, ambivalence or ambiguous of words. Furthermore, ant colony algorithm extremely effect on the retrieval process. The concept of ant colony algorithm embedded in the *weight ontology* is investigated using two methods which are the *with ant* (CAO) method and the *without ant* method. The *with ant* (CAO) method value is obtained from the average performance of set A, B, E and F whereas the *without ant* method is the average performance from the set C, D, G and H. The result shows that, although the average number of nodes and relations of the *with ant* (CAO) method and the *without ant* method are no significant difference, ant colony algorithm which applied in *weight ontology* has extreme affects on four measured parameters. The *with ant* (CAO) method improved the precision 44.53% compared to the *without ant* and 137.18% compared to the baseline. In addition, the *with ant* (CAO) method improved the recall 42.91% compared to the *without ant* and 98.40% compared to the baseline. Moreover, the *with ant* (CAO) method improved the *F*-measure 43.41% compared to the *without ant* and 115.12% compared to the baseline. Surprisingly, the *with ant* (CAO) method improved the diversity 25.21% compared to the *without ant* and 148.33% compared to the baseline. To conclude that the effective ontology should be embedded concept of the ant colony algorithm.

The CAO algorithm consists of three processes in the retrieval process: the *Query Expansion*, *Parameter updating* and *Search* process. The performance of the CAO algorithm in each process are tested and discussed in detail as follow:

Copyright by Kasetsart University. All rights reserved

2.1 The Query Expansion

The relevant documents are the final goal of the IR task based on a query given by users. So, the CAO algorithm aims to enhance the legal retrieval performance. The ant colony algorithm plays a key role in the *Query Expansion* process which uses the ontology as a knowledge base. The query is explored in the ontology. If the query is found in the ontology and it is a branch node, all of the good candidate terms are sorted and proposed to the user in order to get more refined and relevant results. So, the order of the list of the candidate term is important to the user.

There are two methods for investigating the performance of the *Query Expansion* process in order to sort the candidate terms which are *by the ontology building step* method and *by the probability value* (CAO) method. Given the มรดก (succession) term as a query term, the โอน (assign) is used as a query expansion term. The empirical result demonstrates that the order of the list of candidate terms sorted *by the probability value* (CAO) method outperform the order of the list of candidate terms sorted *by the ontology building step* method as shown in Table 20.

In the experiment, the มรดก (succession) is used as a query and explored through the ontology using the *Query Expansion* process. Assume that the query is found in the ontology and all of the candidate terms are proposed to the user. In *the ontology building step* method, the order of the list of the candidate terms depends on the step of the ontology building process. All candidate terms of the query are the สัญญา (contract), สิทธิ (right), หน้าที่ (duties) and โอน (assign) term. All of them are proposed to the user and ranked by the appearance of node found in the ontology. Suppose that the โอน (assign) is a chosen term for the query expansion term (QE) of the query. It means that, the request from the user needs all sentences which both the query term and the QE play the important role in each document. Fortunately, all performance measure values of the CAO algorithm outperform the Baseline method.

Copyright by Kasetsart University. All rights reserved

Table 20 The order of the candidate terms

Query	Baseline				CAO algorithm						
					Query expansion terms		prob				
	P	R	F	D	Order by the ontology building step	Order by the probability value (CAO)		P	R	F	D
มรดก (succession)	0.37	0.50	0.43	0.13	สัญญา (contract) สิทธิ (right) หน้าที่ (duties) **โอน (assign)**	สัญญา (contract) **โอน (assign)** สิทธิ (right) หน้าที่ (duties)	0.44 0.27 0.15 0.14	0.89	0.93	0.91	0.23

Copyright by Kasetsart University All rights reserved

Since ants can rank the pheromone level of each path by itself, the concept of the ant colony algorithm is applied in order to rank the good candidate terms by their probability value in the *Query Expansion* process and propose to the user.

2.2 The Parameters Updating

In the ATOB and CAO algorithm, the ontology structure is modified using the concept of the ant colony algorithm called *weight ontology*. The *weight ontology* not only has a weight (τ) parameter in the link between parent node and child node, but also has another three parameters embedded in the path which are *ph, count* and *prob* parameter. The performance of the *weight ontology* is tested in two methods: *with ant* (CAO) method and *without ant* method. Note that, equation (8) and (9) are used in the *with ant* (CAO) method whereas both equations are ignored in the *without ant* method. The results is shown in Table 19 and concluded that the ontology which used as a knowledge base should be embedded ant colony algorithm (*weight ontology*).

2.3 The search process

In the court decision process, the relevant, diversified and updated sentences are more interesting than the old sentences in the same content, due to the updated sentence is decided by the updating law. The relevant, diversified and updated sentences are needed and used as precedent cases by the legal users. The relevant and diversified sentences can be retrieved using the *weight ontology* which created by the ATOB algorithm. However, the updated sentences are still challenging problem for the researcher. Thus, the CAO algorithm is proposed in order to retrieve the updated sentences from the corpus.

In order to investigate the performance of the *Search* process, there are two methods for retrieving the updated sentences which are *with ant* (CAO) method and *without ant* method. Note that, the equation (10), TF, IDF and $\Delta\tau$ value (updated

Copyright by Kasetsart University. All rights reserved

value) are used in the *with ant* (CAO) method whereas the *without ant* method use only TF and IDF parameter in the *Search* process.

Table 21 The list of retrieved sentences

Query	Query Expansion	with ant (CAO)	without ant
ผู้จัดการมรดก (administrator of an estate)	เพิกถอน (revoke)	255202220	254106574
		254305141	254100419
		254100419	254007013
		254903560	254401443
		255101839	254903560
		255006391	254001883
		254803039	254106557
		254401443	254302269
		254106557	254004496
		254206804	254106574
		254407449	254102043
		255100850	255006391
		255101840	254105130
		255101128	254101107
		255004276	254503776

Table 21 shows the excellent enhancement of the CAO algorithm for improving the Thai CSR process. The retrieved documents obtained from the CAO algorithm are more up-to-date than the other one, due to the ant colony algorithm can be affected on the rank of the retrieved documents. For instance, the first rank of the list obtained from the CAO algorithm is the sentence number 255202220 which was published in 2552 (2009) whereas the first list obtained from the *without ant* method is the sentence number 254106574 which is published in 2541 (1998). Furthermore, the other orders of the list obtained from the CAO algorithm are more updated sentences than the list obtained from the *without ant* method, although both algorithms used the same query and the query expansion term. This study found that the *Search* process the concept of the ant colony algorithm has an effect on the updated sentences.

Copyright by Kasetsart University. All rights reserved

CONCLUSION AND RECOMMENDATION

Conclusion

Ontology is a very important technique used for sharing, managing the knowledge and improving the IR efficiency. However, the ontology building is a time-consuming task for experts. In this work, the ATOB algorithm is developed for automatic building the Thai legal ontology in order to enhance the CSR performance. The Supreme Court sentences and the domain dictionary are used as resources in the ATOB algorithm. The ATOB algorithm consists of three processes which are the *Corpus Preparation*, the *Seed Ontology Building* and the *Ontology Expansion* process. The *Corpus Preparation* process is used for extracting the complete sentences whereas the *Seed Ontology Building* can be used for creating a seed ontology. Moreover, the *Ontology Expansion* process can be used for extracting the ontology and it can be terminated using the threshold parameter. It means that the ontology size can be assigned by the threshold parameter. Furthermore, the threshold parameter can be used for the convergence speed and total processing time on the ontology building process. The concept of the ant colony algorithm is applied in the ATOB algorithm for creating *weight ontology*. In addition, the *weight ontology* consists of four parameters such as *ph*, *count*, *weight* (τ) and *prob* value which used for supporting the retrieval process.

However, the irrelevant, non diversified and outdated sentences are still the problems in the CSR process. In order to enhance the retrieval efficiency, the CAO algorithm which combines the ant colony algorithm and the ontology is proposed for improving the CSR performance. In this research, the *weight ontology* obtained from the ATOB algorithm is used as a knowledge base in the retrieval process. The CAO algorithm consists of three processes which are the *Query Expansion*, the *Parameters Updating* and the *Search* process and the concept of the ant colony algorithm is applied in each process. In the *Query Expansion* process, the good candidate terms are retrieved from ontology and sorted by the probability value which is the concept of the ant colony algorithm. The *ph*, *count*, *weight* (τ) and *prob* parameter of the query

Copyright by Kasetsart University. All rights reserved

and the query expansion term are updated in the *Parameters Updating* process. The TF, IDF and weight parameter of both query and query expansion term are used for refining the relevant, diversified and updated sentences in the *Search* process.

The empirical results demonstrate that the ontology which obtained from the ATOB and used in the CAO algorithm outperforms the Baseline method. The performance of the ATOB and CAO algorithm can significantly improve the CSR process. The precision was increased by 143.59%, the recall was increased by 100%, the F-measure was increased by 118.6% and the diversity was increased by 166.67%. Moreover, the main contributions obtained from the ATOB and CAO algorithm are the good candidate terms which are ordered by the probability value, the weight and the up-to-date value are modified by the ant colony algorithm for the relevant, diversified and updated sentences. This study found that the ontology should be used as knowledge base in the information retrieval process and the ontology structure should be embedded ant colony algorithm (*weight ontology*). The performance of the ATOB and CAO algorithm is the precision, recall, *F*-measure and diversity which are 0.95, 0.94, 0.94 and 0.40 respectively.

The ATOB, CAO algorithm and *weight ontology* can be extended to other areas of the Thai law and other techniques can be applied for improving the ontology building process and information retrieval efficiency.

Copyright by Kasetsart University. All rights reserved

Recommendation

Although the ATOB algorithm is proposed for automatic ontology building and the CAO algorithm is proposed for improving the CSR process, there are some limitations of this research.

1. The query which disappears in the ontology is ignored for the weight ontology and CAO algorithm.

2. Many query expansions take more processing time for retrieving the relevant, diversified and updated sentences.

Moreover, we plan to extend these contributions to other areas in the near future such as:

1. The domain specific dictionary can be automatically created by machine learning method such as NLP.

2. The weight ontology can be applied for another domain such as medicine area.

3. In the special case, the important sentences can be retrieved in the top of the list using the ant colony algorithm.

Copyright by Kasetsart University. All rights reserved

LITERATURE CITED

Abachizadeh, M. and M. Tahani. 2009. An ant colony optimization approach to multi-objective optimal design of symmetric hybrid laminates for maximum fundamental frequency and minimum cost. **Struct Multidisc Optim** 37: 367–376.

Agnoloni, T., L. Bacci, E. Francesconi, W. Peter, S. Montemagni and G. Venturi. 2009. A two-level knowledge approach to support multilingual legislative drafting. **Frontiers in Artificial Intelligence and Applications** 188: 177-198

Agnoloni, T., L. Bacci, E. Francesconi, P. Spinosa, D. Tiscornia, S. Montemagni and G. Venturi. 2007. Building an ontological support for multilingual legislative drafting, pp. 9-18. *In* **Proceedings of the 2007 conference on Legal Knowledge and Information Systems- JURIX 2007: The Twentieth Annual Conference.** 12-15 December 2007, Leiden, Netherlands.

Alupoaei, S. and S. Katkoori. 2004. Ant Colony System Application to Macrocell Overlap Removal. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems** 12(10): 1118–1122.

Antoniou, G. and F. van Harmelen. 2004. **A Semantic Web Primer**. 1st ed. The MIT Press, England.

Bagby, J. and T. Mullen. 2007. Legal ontology of sales law application to ecommerce. **Artificial Intelligence and Law** 15: 155–170.

Bartalos, P., M. Barla, G. Frivolt, M. Tvarozek, A. Andrejko, M. Bielikova and P. Navrat. 2007. Building an ontological base for experimental evaluation of semantic web applications. **Lecture Note in Computer Science** 4362: 682–692.

Copyright by Kasetsart University. All rights reserved

Benjamins, V. R., P. C. J. Contreras, J. M. L. Cobo and L. Lemus. 2005. Iuriservice: An intelligent frequently asked questions system to assist newly appointed judges. **Law and the Semantic Web- Lecture Note in Artificial Intelligence** 3369: 205–222.

Benjamins, V. R., D. Fensel, S. Decker and A. G. Perez. 1999. $(KA)^2$: building ontologies for the internet: a mid-term report. **International Journal Human-Computer Studies** 51: 687–712.

Bernaras, A., I. Laresgoiti, and J. Corera. 1996. Building and reusing ontologies for electrical network applications, pp. 298-302. *In* **Proceedings of the 12th European Conference on Artificial Intelligence (ECAI'96)**. 11-16 August 1996, Budapest, Hungary.

Boonchom, V. and N. Soonthornphisaj. 2009. Thai succession law ontology building using ant colony algorithm, pp. 27–37. *In* **Proceedings of the 3rd International Workshop on Juris-informatics (JURISIN 2009).** 19-20 November 2009, Tokyo, Japan.

Boonchom, V. and N. Soonthornphisaj. 2010a. Legal ontology construction using ATOB algorithm. **Lecture Note in Business Information System** 57: 268–279.

Boonchom, V. and N. Soonthornphisaj. 2010b. Thai succession and family law ontology building using ant colony algorithm. **Lecture Note in Artificial Intelligence** 6284: 19–32.

Boonchom, V. and N. Soonthornphisaj. 2012. ATOB algorithm: An automatic ontology construction for Thai legal sentences retrieval. **Journal of Information Science** 38 (1): 37-51.

Copyright by Kasetsart University. All rights reserved

Borst, W. N. 1997. **Construction of Engineering Ontologies for Knowledge Sharing and Reuse**. Ph.D. thesis, University of Twente.

Breuker, J. 2003. Managing legal domains: in search of a core ontology for law, pp. 1–36. *In* **Proceedings of the Workshop on Knowledge Management and the Semantic Web at KCAP-2003**. 25 October 2003, Florida, USA.

Breuker, J., A. Valente and R. Winkels. 2004. Legal Ontologies in Knowledge Engineering and Information Management. **Artificial Intelligence and Law** 12: 241–277.

Caro, G. D. and M. Dorigo. 1998. AntNet: Distributed Stigmergetic Control for Communications Networks. **Journal of Artificial Intelligence Research** 9: 317–365.

Casellas, N. 2011. **Legal Ontology Engineering Methodologies, Modeling Trends, and the Ontology of Professional Judicial Knowledge**. 1st ed. Springer.

Casellas, N. 2008. **Modeling legal knowledge through ontologies. OPJK: The Ontology of professional judicial knowledge**. Ph.D. Thesis. Universitat Autonoma de Barcelona.

Casellas, N. 2009. Ontology evaluation through usability measures an experiment with the SUS scale in the legal domain. **Lecture Note in Computer Science** 5872: 594–603.

Casellas, N., P. Casanovas, J.-J. Vallb, M. Poblet, M. Blzquez, J. Contreras, J.-M. Lpez-Cobo and V. Richard. 2007. Semantic enhancement for legal information retrieval: Iuriservice performance, pp. 49–57. *In* **Proceedings of the 11th International Conference on Artificial Intelligence and Law (ICAIL 2007).** 4-8 June 2007, California, USA.

Copyright by Kasetsart University. All rights reserved

Chandrasekaran, B., J. R. Josephson and V. R. Benjamins. 1999. What are ontologies, and why do we need them? **IEEE Intelligent Systems** 14 (1): 20–26.

Coalter, A. and J. Leopold. 2010. Automated ontology generation using spatial reasoning. **Lecture Note in Computer Science** 6291: 428-493.

Despres, S. and S. Szulman. 2007. Merging of legal micro-ontologies from European directives. **Artificial Intelligence and Law** 15: 187–200.

Dorigo, M. 1992. **Optimization, learning and natural algorithms (in italian).** Ph.D. Thesis, Dipartimento di Elettronica, Politecnico di Milano.

Dorigo, M. 2007. **Ant Colony Optimization**. Scholarpedia 2 (3): 1461. Available Source: http://www.scholarpedia.org/article/Ant_colony_optimization, December 2, 2011.

Dorigo, M., M. Birattari and T. Stutzle. 2006. Ant colony optimization artificial Ants as a computational intelligence technique. **IEEE Computational Intelligence Magazine** 28–39.

Dorigo, M. and L. M. Gambardella. 1997. Ant colony system: A cooperative learning approach to the traveling salesman problem. **IEEE Transactions on Evolutionary Computation** 1 (1).

Dorigo, M., V. Maniezzo and A. Colorni. 1996. The ant system: Optimization by a colony of cooperating agents. **IEEE Transactions on Systems, Man, and Cybernetics Part B** 26 (1): 29–41.

Dorigo, M. and T. Stutzle. 2004. **Ant Colony Optimization**. 1st ed. The MIT Press, England.

Copyright by Kasetsart University All rights reserved

Dridi, O. and M. B. Ahmed. 2008. Building an ontology-based framework for semantic information retrieval: application to breast cancer, pp. 1–6. *In* **Proceedings of the 3rd international conference on information and communication technologies: from theory to applications**. 7-11 April 2008, Damascus, Syria.

Drumond, L. R., R. Girardi and A. Leita. 2007. Architectural design of a multi-agent recommender system for the legal domain, pp. 183–187. *In* **Proceedings of the Eleventh International Conference on Artificial Intelligence and Law (ICAIL 2007).** 4-8 June 2007, California, USA.

Drumond, L. R., R. Girardi, A. N. Lindoso and B. L. Marinho. 2006. A Semantic Web based recommender system for the legal domain, pp. 81–83. *In* **Proceedings of the 17th European Conference on Artificial Intelligence (ECAI 2006) Workshop on Recommender Systems.** 28-29 August, 2006, Riva del Garda, Italy.

Fernandez, M., A. Gomez-Perez and N. Juristo. 1997. METHONTOLOGY: From ontological art towards ontological engineering. **AAAI Tech. rept. American Association for Artificial Intelligence SS-97-06.** 8.

Fernandez-Lopez, M. 1999. Overview of methodologies for building ontologies, pp. 4.1–4.13. *In* **Proceedings of the Workshop on Ontologies and Problem-Solving Methods in Conjunction with the 16th International Joint Conference on Artificial Intelligence (IJCAI99).** 2 August 1999, Stockholm, Sweden.

Fernandez-Lopez, M. and A. Gomez-Perez. 2002. Overview and analysis of methodologies for building ontologies. **Knowledge Engineering Review** 17 (2): 129–1564.

Copyright by Kasetsart University All rights reserved

Francesconi, E. and D. Tiscornia. 2008. Building semantic resources for legislative drafting: The dalos project. **Computable models of the law in Lecture Note in Artificial Intelligence** 4884: 56–69.

Garcia, R., R. Gil and J. Delgado. 2007. A web ontologies framework for digital rights management. **Artificial Intelligence and Law** 15: 137–154.

Ghalayini, H. and F. Kharbat. 2008. A new approach for developing ontology from generated ruleset, *In* **Proceedings of the International Arab Conference on Information Technology (ACIT2008).** 16-18 December 2008, Zarqa Private University, Jordan.

Gomez-Perez, A., M. Fernandez-Lopez and O. Corcho. 2003. **Ontological engineering with examples from the areas of knowledge management, e-commerce and the Semantic Web**. 1st ed. Springer.

Gong, W. and S. Wang. 2009. Chaos ant colony optimization and application, pp. 301–303. *In* **Proceedings of the 4th International Conference on Internet Computing for Science and Engineering (ICICSE 2009).** 21-22 December 2009, Harbin, China.

Gruber, T. 1991. The role of common ontology in achieving sharable, reusable knowledge bases, pp. 1–4. *In* **Proceedings of the 2nd International Conference on Principles of knowledge representation and reasoning**. 31 January 1991, California, USA.

Gruber, T. 2009. **Ontology. Encyclopedia of Database Systems**. Available Source: http://tomgruber.org/writing/ontology-definition-2007.htm, December 2, 2011.

Gruber, T. R. 1992. Ontolingua: A mechanism to support portable ontologies. **Knowledge Systems Laboratory, NASA Grant NCC2-537, NASA Grant NAG2-581 and DARPA prime contract DAAAA15-91-C-0104.** 61

Copyright by Kasetsart University. All rights reserved

Gruber, T. R. 1993. A translation approach to portable ontology specifications. **Knowledge Acquisition** 5 (2): 199–220.

Gruber, T. R. 1995. Toward principles for the design of ontologies used for knowledge sharing. **International Journal of Human-Computer Studies** 43: 907–928.

Gruninger, M. and M. S. Fox. 1995. Methodology for the design and evaluation of ontologies, pp. 6.1-6.10. *In* **Proceedings of International Joint Conference on AI 1995, Workshop on Basic Ontological Issues in Knowledge Sharing (IJCAI 1995).** 20-25 August 1995, Quebec, Canada.

Guarino, N. 1995. Formal ontology, conceptual analysis and knowledge representation. **International Journal of Human-Computer Studies** 43 (5-6): 625–640.

Guarino, N. 1997. Semantic matching: Formal ontological distinctions for information organization, extraction, and integration. **Information extraction: A multidisciplinary approach to an emerging information technology** 1299: 139–170.

Guarino, N. 1998. Formal ontology and information systems, pp. 3–15. *In* **Proceedings of the International Conference on Formal Ontologies in Information Systems (FOIS98).** 6-8 June 1998,Trento, Italy.

Henze, N., P. Dolog and W. Nejdl. 2004. Reasoning and ontologies for personalized e-learning in the semantic web. **Educational Technology and Society** 7 (4): 82–98.

Hu, H. and X. Du. 2007. Building bilingual ontology from WordNet and Chinese classified thesaurus. **Lecture Note in Artificial Intelligence** 4798: 649–654.

Copyright by Kasetsart University All rights reserved

Hwang, S., Y. Jung, A. Yoon and H. C. Kwon. 2006. Building Korean Classifier Ontology Based on Korean WordNet. **Lecture Note in Artificial Intelligence** 4188: 261–268.

Hwang, S., A. Yoon and H. C. Kwon. 2008. Semantic representation of Korean numeral classifier and its ontology building for HLT applications. **Lang Resources and Evaluation** 42: 151–172.

Kharbat, F. and H. Ghalayini. 2009. New algorithm for building ontology from existing rules: a case study, pp. 12-16. *In* **Proceedings of the International Conference on Information Management and Engineering (ICIME2009).** 3-5 April 2009, Kuala Lumpur, Malaysia.

Kramer, R., R. Nikolai and C. Habeck. 1997. Thesaurus federations: loosely integrated thesauri for document retrieval in networks based on Internet technologies. **International Journal of Digital Libraries** 1: 122–131.

Kurematsu, M. and T. Yamaguchi. 1997. A Legal Ontology Refinement Support Environment Using a Machine-Readable Dictionary. **Artificial Intelligence and Law** 5: 119–137.

Lame, G. 2005. Using NLP Techniques to Identify Legal Ontology Components: Concepts and Relations. **Lecture Note in Artificial Intelligence** 3369: 169–184.

Lee, C. S., Y. F. Kao, Y. H. Kuo and M. H. Wang. 2007. Automated ontology construction for unstructured text documents. **Data and Knowledge Engineering** 60: 547–566.

Lee, T., I. hoon Lee abd Suekyung Lee, S. goo Lee, D. Kim, J. Chun, H. Lee and J. Shim. 2006. Building an operational product ontology system. **Electronic Commerce Research and Applications** 5: 16–28.

Copyright by Kasetsart University. All rights reserved

Leeds, J. 2011. **Introduction to the Legal System and Legal Research of the Kingdom of Thailand.** Available Source: http://www.nyulawglobal.org/Globalex/Thailand1.htm, December 4, 2011.

Li, X., J. Zhang and T. Huang. 2007. A standardized learning resources retrieval system based on ontology matching. **Lecture Note in Computer Science** 4469: 411–421.

Liu, X., H. Qi and Y. Chen. 2006. Optimization of special vehicle routing problem based on ant colony system. **Lecture Note in Business Information Systems** 1441: 1128–1233.

Loukachevitch, N. 2009. Concept formation in linguistic ontologies. **Lecture Note in Artificial Intelligence** 5662: 2–22.

Missikoff, M., P. Velardi and P. Fabriani. 2003. Text mining techniques to automatically enrich a domain ontology. **Applied Intelligence** 18: 323–340.

Mitre, H. A., A. I. Gonzlez-Tablas, B. Ramos and A. Ribagorda. 2006. A legal ontology to support privacy preservation in location-based services. **Lecture Note in Computer Science** 4278: 1755–1764.

Mizoguchi, R., Y. Tijerino and M. Ikeda. 1995. Task analysis interview based on task ontology. **Expert Systems with Applications** 9 (1): 15–25.

Nadah, N., M. D. de Rosney and B. Bachimont. 2007. Licensing digital content with a generic ontology: Escaping from the jungle of rights expression languages, pp. 65–69. *In* **Proceedings of the Eleventh International Conference on Artificial Intelligence and Law (ICAIL 2007).** 4-8 June 2007, California, USA.

Copyright by Kasetsart University. All rights reserved

Nicola, A. D., M. Missikoff and R. Navigli. 2009. A software engineering approach to ontology building. **Information Systems** 43: 258–275.

Noy, N. F. and C. D. Hafner. 1997. The state of the art in ontology design a survey and Comparative review. **AI Magazine** 18 (3): 53–74.

Noy, N. F. and D. L. McGuinness. 2001. Ontology development 101: A guide to creating your first ontology. **Stanford Knowledge Systems Laboratory and Stanford Medical Informatics SMI-2001-0880**, 25.

Rissland, E. L., K. D. Ashley and R. Loui. 2003. AI and Law: A fruitful synergy. **Artificial Intelligence** 150: 1–15.

Saravanan, M., B. Ravindran and S. Raman. 2009. Improving legal information retrieval using an ontological framework. **Artificial Intelligence and Law** 17: 101–124.

Schoonderwoerd, R., O. Holland, J. Bruten and L. Rothkrantz. 1996. Ant-based load balancing in telecommunications networks. **Adaptive Behavior** 5 (2): 169–207.

Serasset, G. and J. P. Chevallet. 2004. Simple Translations of Monolingual Queries Expanded Through an Association Thesaurus X-IOTA IR System Used for CLIPS Bilingual Experiments. **Lecture Note in Computer Science** 3237: 242–252.

Shin, I., T. Kawamura, H. Nakagawa, K. Nakayama, Y. Tahara and A. Ohsuga. 2009. ONTOMO: Development of Ontology Building Service Evaluation of Instance Recommendation Using Proper Noun Extraction. **Lecture Note in Artificial Intelligence** 5925: 143–158.

Copyright by Kasetsart University All rights reserved

Smith, B. and C. Welty. 2001. Ontology: Towards a New Synthesis, pp. 1-8. *In* **Proceedings of the International Conference on Formal Ontology in Information Systems (FOIS'01)**. 17-19 October 2001, Ogunquit, Maine, USA.

Sobecki, J. 2008. Ant Colony Metaphor Applied in User Interface Recommendation. **New Generation Computing** 26: 277–293.

Stutzle, T. and H. H. Hoos. 2000. MAX-MIN Ant System. **Future Generation Computer Systems** 16: 889–914.

Studer, R., V. R. Benjamins and D. Fensel. 1998. Knowledge Engineering: Principles and Methods. **Data Knowledge Engineering** 25 (1-2): 161–197.

Sure, Y. 2003. **Methodology, tools and case studies for ontology based knowledge management**. Ph.D. Thesis. Fakultat fur Wirschaftwissenschaften der Universitat Fridericiana zu Karlsruhe.

Sure, Y., C. Tempich and D. Vrandecic. 2006. **Semantic web technology: Trends and research in ontology-based systems**. 1st ed. Wiley, England.

Triay, J. and C. C. Pastor. 2010. An ant-based algorithm for distributed routing and wavelength assignment in dynamic optical networks. **IEEE Journal on Selected Areas in Communications** 28 (4): 542–552.

Uschold, M. 1996. Building ontologies: Towards a unified methodology, pp. 1–18. *In* **Proceedings of Expert Systems the 16th Annual Conference of the British Computer Society Specialist Group on Expert Systems.** 16-18 December 1996, Cambridge, UK.

Uschold, M. and M. Gruninger. 1996. Ontologies: Priciples, Methods and Applications. **Knowledge Engineering Review** 11 (2): 93–155.

Copyright by Kasetsart University All rights reserved

Uschold, M. and R. Jasper. 1999. A framework for understanding and classifying ontology applications, pp. 1–12. *In* **Proceedings of the IJCAI-99 Workshop on Ontologies and Problem-Solving Methods (KRR5).** 31 July – 6 August 1999, Stockholm, Sweden.

Uschold, M. and M. King. 1995. Towards a methodology for building ontologies, pp. 1–13. *In* **Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing (IJCAI95)**. 20-25 August 1995, Quebec, Canada.

Valente, A. 2005. Types and roles of legal ontologies. **Law and the Semantic Web- Lecture Note in Artificial Intelligence** 3369: 65–76.

Valente, A. and J. Breuker. 1996. Towards principled core ontologies. *In* **Proceedings of the 10th Knowledge Acquisition Workshop (KAW 96).** November 1996, Alberta, Canada.

Vallbe, J. J. 2009. **Facing uncertainty: Institutional and cognitive constraints in Spanish junior judges decision-making**. Ph.D. Thesis. Universitat de Barcelona, Barcelona.

van Hest, G. A. C. M. 1995. **The role of ontologies in knowledge engineering**. Ph.D. Thesis. University of Amsterdam.

van Kralingen, R. W., P. R. Visser and T. J. Bench-Capon. 1999. A principled approach to developing legal knowledge systems. **International Journal of Human-Computer Studies** 51 (6): 1127–1154.

Visser, P. R. S. 1998. Implicit assumptions in legal knowledge systems. pp. 1-10. *In* **Proceedings of the 13th BILETA Conference: The Changing Jurisdiction**. 27-28 March 1998, Trinity College, Dublin, UK.

Copyright by Kasetsart University. All rights reserved

Visser, P. R. S. and T. J. M. Bench-Capon. 1997. A comparison of two legal ontologies, pp. 37–45. *In* **Proceedings of the 1st International Workshop on Legal Ontologies (LEGONT 97).** July 1997, Melbourne, Australia.

Visser, P. R. S. and T. J. M. Bench-Capon. 1998. A Comparison of four ontologies for the design of legal knowledge systems. **Artificial Intelligence and Law** 6: 27–57.

Wyner, A. 2008. An ontology in OWL for legal case-based reasoning. **Artificial Intelligence Law** 16: 361–387.

Xia, N., J. Jiang, M. Qi, C. Yu, Y. Huang and Q. Zhang. 2007. A WSN coalition formation algorithm based on ant colony with dual-negative feedback. **ICCS 2007 Part IV- Lecture Note in Computer Science** 4490: 1139–1146.

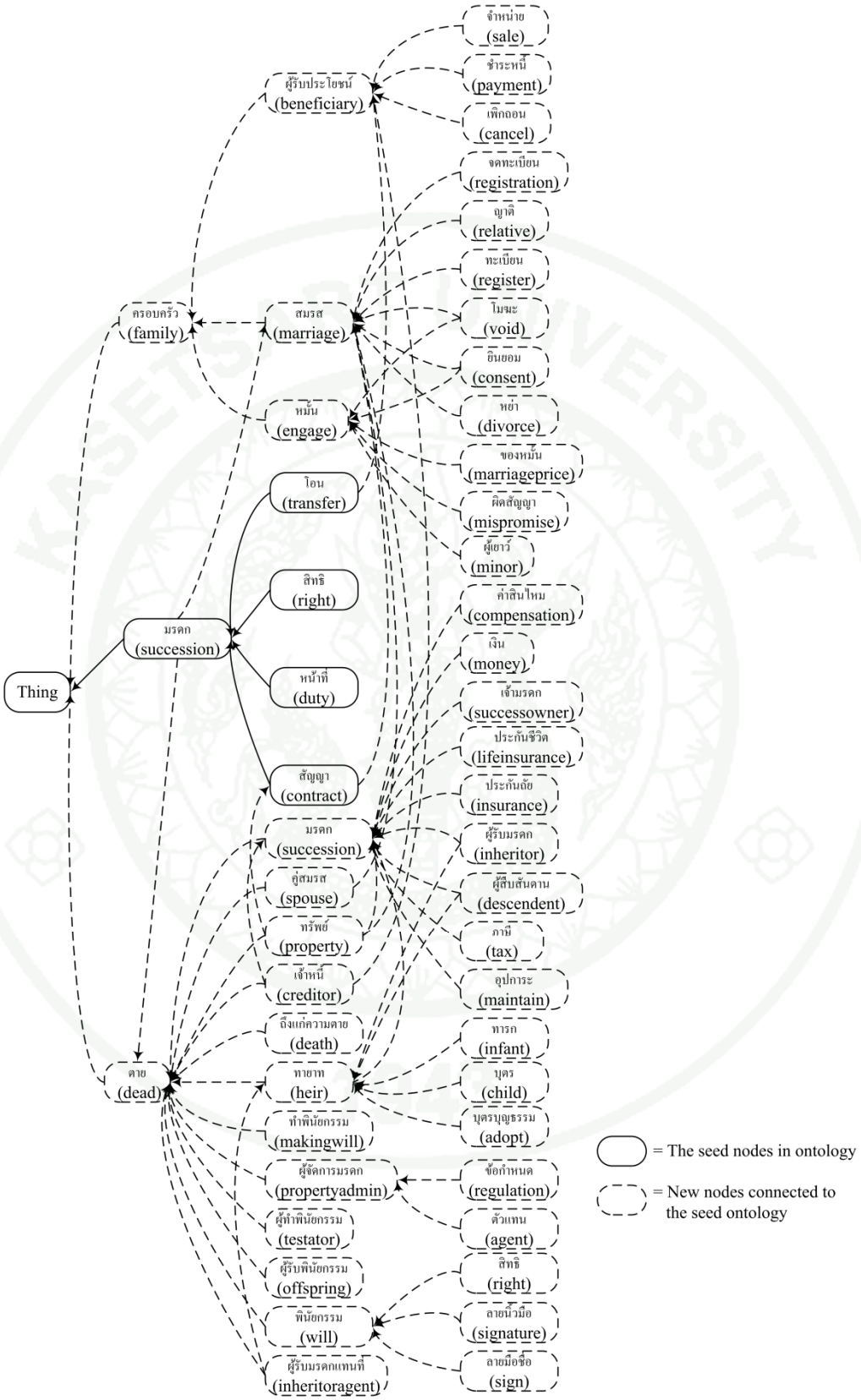
Xie, N., W. Wang and Y. Yang. 2008. Ontology-based agricultural knowledge acquisition and application. **Computer and Computing Technologies in Agriculture** 1: 349–357.

Ziegler, P., C. Kiefer, C. Sturm, K. R. Dittrich and A. Bernstein. 2006. Detecting similarities in ontologies with the SOQA-SimPack toolkit. **Lecture Note in Computer Science** 3896: 59–76.

Copyright by Kasetsart University. All rights reserved

APPENDIX

Copyright by Kasetsart University All rights reserved



Appendix Figure 1 The example of the Thai legal ontology

Copyright by Kasetsart University All rights reserved

CURRICULUM VITAE

NAME : Mr. Vi-sit Boonchom

BIRTH DATE : August 29, 1975

BIRTH PLACE : Pattini, Thailand

EDUCATION :

YEAR	INSTITUTE	DEGREE/DIPLOMA
2000	RMUTP.	B.Ed (Computer Engineering)
2004	KMUTT.	M.I.Ed (Computer and IT)

POSITION/TITLE : Lecturer

WORK PLACE : Faculty of Science, Thaksin University

SCHOLARSHIP/AWARDS : Higher Education Commission 2008-2011

Copyright by Kasetsart University. All rights reserved