Information Transformation and Automated Reasoning for Automated Compliance Checking in Construction

J. Zhang[1] and N. M. El-Gohary[2]

[1]Graduate Student, Department of Civil and Environmental Engineering, University of Illinois at Urbana-Champaign, 205 North Mathews Ave., Urbana, IL 61801; PH (217) 607-6006; FAX (217) 265-8039; email: jzhang70@illinois.edu
[2]Assistant Professor, Department of Civil and Environmental Engineering, University of Illinois at Urbana-Champaign, 205 North Mathews Ave., Urbana, IL 61801; PH (217) 333-6620; FAX (217) 265-8039; email: gohary@illinois.edu

ABSTRACT

This paper presents a new approach for automated compliance checking in the construction domain. The approach utilizes semantic modeling, semantic Natural Language Processing (NLP) techniques (including text classification and information extraction), and logic reasoning to facilitate automated textual regulatory document analysis and processing for extracting requirements from these documents and formalizing these requirements in a computer-processable format. The approach involves developing a set of algorithms and combining them into one computational platform: 1) semantic machine-learning-based algorithms for text classification (TC), 2) hybrid syntactic-semantic rule-based algorithms for information extraction (IE), 3) semantic rule-based algorithms for information transformation (ITr), and 4) logic-based algorithms for compliance reasoning (CR). This paper focuses on presenting our algorithms for ITr. A semantic logic-based representation for construction regulatory requirements is described. Semantic mapping rules and conflict resolution rules for transforming the extracted information into the representation are discussed. Our combined TC, IE and ITr algorithms were tested in extracting and formalizing quantitative requirements in the 2006 International Building Code, achieving 96% and 92% precision and recall, respectively.

INTRODUCTION

Manual regulatory compliance checking of construction projects is costly, time-consuming, and error-prone. Automated compliance checking (ACC) is expected to reduce the time, cost, and errors of compliance checking. Previous research (e.g. Tan *et al*. 2010, Eastman *et al*. 2009, Lau and Law 2004, Garrett and Fenves 1987) and software development efforts (e.g. Solibri 2011) have undoubtedly paved the way for ACC in the architectural, engineering, and construction (AEC) industry. However, these efforts are limited in their automation and reasoning capabilities; existing ACC systems require manual effort for extracting requirements from textual regulatory documents (e.g. codes) and encoding these requirements in a computer-processable format. To address this gap, the authors are proposing a new approach for ACC (Zhang and El-Gohary 2012). It utilizes semantic modeling, semantic Natural Language Processing (NLP) techniques (including text classification and information extraction), and logic reasoning to facilitate automated textual regulatory document analysis (e.g. code analysis) and processing for extracting requirements from these documents and formalizing these requirements in a computer-processable format. The approach involves developing a set of algorithms and combining them into one computational platform: 1) semantic machine-learning-based algorithms for text classification (TC), 2) hybrid syntactic-semantic rule-based algorithms for information extraction (IE), 3) semantic rule-based algorithms for information transformation (ITr), and 4) logic-based algorithms for compliance reasoning (CR). In this paper, we focus on presenting our algorithms for ITr.

BACKGROUND

Natural language processing (NLP) is a field of artificial intelligence (AI) that is intended to enable computers to analyze and process natural language text or speech in a human-like manner. Examples of NLP techniques include tokenization, part-of-speech (POS) tagging, named entity recognition, and co-reference resolution etc. (Marquez 2000). Information extraction (IE) is a subfield of NLP that aims at extracting targeted information from text sources to fill in pre-defined information templates. In our proposed ACC approach, we utilize NLP techniques because construction codes and regulations are represented in unstructured text format. NLP techniques will facilitate the analysis and processing of these codes and regulations for extraction and formalization of requirements/rules.

PROPOSED AUTOMATED COMPLIANCE CHECKING APPROACH

The authors are proposing a six-phase, iterative approach for extracting requirements from textual regulatory documents and formalizing these requirements in a computer-processable format (Figure 1).

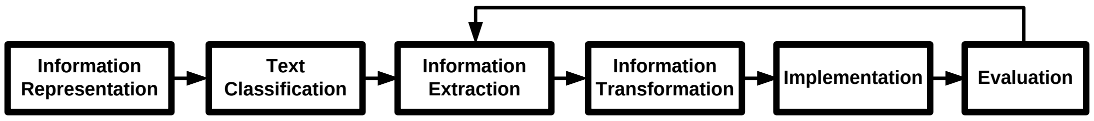


Figure 1. Proposed approach.

Phase 1 – Information representation. We represent the requirements in construction regulations as first order logic-based axioms. Our representation is limited to Horn-Clause-type axioms to facilitate further reasoning using logic programs (logic programming can only represent sentences of the form of a Horn Clause). "Horn clause is a disjunction of literals of which at most one is positive." All horn clauses can be represented in rules that have one or more antecedents (i.e. left-hand sides) that are conjoined (i.e. combined using 'and operator'), and a single consequent (i.e. right-hand side) (Russell and Norvig 2010). Each horn clause represents one requirement. Its right-hand side indicates what this requirement is about. Its left-hand side is consisted of one or more predicates. A predicate is consisted of a predicate symbol and one or more arguments in parenthesis following the predicate symbol. Each predicate organizes information instances corresponding to one or more concepts and/or relations. The horn clauses representation is the target format for our information transformation (ITr) process. The input representation to ITr is the tuple format which results from IE. Each piece in the tuple is an "information element". Each extracted word or phrase recognized corresponding to an "information element" is an "information instance". An example illustrating input and output formats of ITr is shown in Table 1.

Table 1. A transformation example.

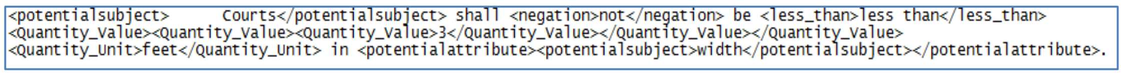
Information Extraction (Output: Tuple Format)					
Requirement Sentence	Subject	Compliance Checking Attribute	Comparative Relation	Quantity Value	Quantity Unit/Reference
Courts shall not be less than 3 feet in width.	court	width	not less than	3	feet
Information Transformation (Output: Logic Clause)					
Generated logic clause	compliance_width_of_court(Court) :- width(Width), court(Court), has(Court,Width), greater_than_or_equal(Width,quantity(3,feet)).				

Phase 2 - Text classification. Text classification (TC) aims at recognizing the relevant sentences from a text corpus. Relevant sentences are the sentences that contain the type of information that need to be extracted and transformed into logic clauses. This phase saves unnecessary processing of irrelevant sentences in later phases. It also avoids extraction and transformation errors caused by irrelevant sentences. The presentation of our TC algorithms and results is outside the scope of this paper. For further details on the authors' work in the area of TC, the reader is referred to Salama and El-Gohary 2013.

Phase 3 - Information extraction. Information extraction (IE) aims at recognizing the words and phrases in the relevant sentences that carry target information, extracting these information, and filling these information into pre-defined information templates. Target information is the needed information for constructing logic clauses that describe requirements in construction regulations. IE consists of feature generation, target information analysis, and development of extraction rules. Both syntactic (i.e. related to syntax and grammar) and semantic (i.e. related to context and meaning) features are used for IE. The presentation of our IE algorithms and results is outside the scope of this paper. For further details on the authors' work in the area of IE, the reader is referred to Zhang and El-Gohary 2012.

Phase 4 - Information transformation. Information transformation (ITr) aims at transforming the extracted information into logic clauses. ITr algorithms are developed using semantic mapping rules, and conflict resolution rules. The semantic mapping rules define how to process the information instances according to their semantic meaning. The semantic meaning of each information instance is defined by the concept or relation it is associated with. (e.g. 'subject' defines the semantic meaning for 'court' in the example in Table 1, i.e. it defines that 'court' is the 'subject' of compliance checking). For example, one semantic mapping rule could be "If both 'subject' and 'attribute' information instances exist for an information tuple, then generate a fresh variable with the 'subject' information instance being the predicate symbol, generate another fresh variable with 'attribute' information instance being the predicate symbol, and a relationship 'has' with the two arguments filled by the two variables". According to this semantic mapping rule, horn clause disjoints court(Court), width(Width), and has(Court,Width) will be generated for the statement "Courts shall not be less than 6 feet in width", since for this requirement sentence, "court" is recognized as 'subject' information instance and "width" is recognized as 'attribute' information instance. Conflict resolution rules resolve conflicts between information elements. For example, one conflict resolution rule could be "The information instance indicating 'comparative relation' should appear before its corresponding information instance indicating 'quantity value', and it should be the nearest one to its 'quantity value'". According to this conflict resolution rule, for the requirement sentence "The openings therein shall be a minimum of 1/8 inch and shall not exceed 1/4 inch", the two 'comparative relation' information instances - ‘minimum’ and ‘not exceed’, will be coupled with the correct ‘quantity value’ information instances - ‘1/8’ and ‘1/4’, respectively.

Phase 5 - Implementation. This phase aims at implementing our algorithms for TC, IE, and ITr into one computational platform. For logic clause representation, we chose the representation of Prolog to facilitate future compliance reasoning (CR). Prolog is an approximate realization of the logic programming computation model on a sequential machine (Sterling and Shapiro 1986). We used the syntax of B-Prolog. B-Prolog is a Prolog system with extensions for programming concurrency, constraints, and interactive graphics. It has bi-directional interface with C and Java (Zhou 2012). We utilized two types of logic statements in B-Prolog syntax: facts, and rules. A rule has the form: “H :- B1, B2, …, Bn. (n>0)”. H, B1, …, Bn are atomic formulas. H is called the head and the right-hand side of ‘:-’ is called the body of the rule. A fact is a special kind of rule whose body is always true (Zhou 2012). To build the ground for quantitative reasoning, we develop a set of built-in rules for our logic clause representation. To prevent non-termination of deduction process, the “cut” operator in Prolog is utilized to remove choice points from alternative clauses to the left of the “cut”. TC and IE are implemented using GATE (General Architecture for Text Engineering) tools (Univ. of Sheffield 2011). GATE has a variety of built-in tools for a variety of text processing functions (e.g. tokenization, sentence splitting, POS tagging, gazetteer compiling, morphological analysis, Java Annotation Patterns Engine, etc.). For ITr, the semantic mapping rules and conflict resolution rules are implemented in Python programming language (v3.2.3). The “re” module (i.e. regular expression module) in Python is utilized for pattern matching, so that each extracted information instance could be used for subsequent processing steps based on their information element tags (example tags are shown in Figure 3).



```
<potentialsubject>      Courts</potentialsubject> shall <negation>not</negation> be <less_than>less than</less_than>
<Quantity_Value><Quantity_Value><Quantity_Value>3</Quantity_Value></Quantity_Value></Quantity_Value>
<Quantity_Unit>feet</Quantity_Unit> in <potentialattribute><potentialsubject>width</potentialsubject></potentialattribute>.
```

Figure 3. An example sentence with recognized information element tags.

Phase 6 - Evaluation. This phase aims at evaluating the combined result of TC, IE, and ITr using precision (P), recall (R), and F-measure (F), where P = correct logic clause elements produced / total logic clause elements produced, R = correct logic clause elements produced / total logic clause elements ought to be produced, and F= 2PR/(P+R). A logic clause element is a predicate symbol or a predicate argument for a logic clause. For example, for the predicate court(C), ‘court’ is a logic clause element, and ‘C’ is also a logic clause element.

PRELIMINARY EXPERIMENTAL RESULTS AND ANALYSIS

The proposed approach was tested on quantitative requirements in the 2006 International Building Code (ICC 2006). Chapter 12 was randomly selected for testing. A quantitative requirement is a rule that defines the relationship between a quantitative attribute of a subject and a specific quantity. The preliminary experimental results are shown in Table 2.

Table 2. Preliminary experiment results

	Subject	Compliance Checking Attribute	Comparative Relation	Quantity Value	Quantity Unit/Reference	Total
Number of logic clause elements in gold standard	233	163	67	76	132	671
Total number of logic clause elements generated	225	156	69	76	119	645
Number of logic clause elements correctly generated	210	151	63	75	119	618
Precision	0.93	0.97	0.91	0.99	1.00	0.96
Recall	0.90	0.93	0.94	0.99	0.90	0.92
F-Measure	0.92	0.95	0.93	0.99	0.95	0.94

To conduct our experiment, we have developed and used a small-size ontology to assist in the recognition and extraction of construction domain concepts and relations. We used the built-in ontology editor in GATE for ontology development. For TC and IE, we used ANNIE (A Nearly-New Information Extraction System) in GATE for POS tagging, and gazetteer compiling; and we used JAPE (Java Annotation Patterns Engine) transducer for text classification and for writing information extraction rules. When conducting our IE, five information elements were recognized: ‘subject’, ‘compliance checking attribute’, ‘comparative relation’, ‘quantity value’, and ‘quantity unit’ or ‘quantity reference’. A ‘subject’ is a ‘thing’ (e.g. building object, space, etc.) that is subject to a particular regulation or norm. A ‘compliance checking attribute’ is a specific characteristic of a ‘subject’ by which its compliance is assessed. A ‘comparative relation’ is a relation commonly used for comparison such as greater_than_or_equal, less_than_or_equal, or equal_to, etc. A ‘quantity value’ is the value that quantifies the requirement. A ‘quantity unit’ is the unit of measure for the ‘quantity value’. A ‘quantity reference’ is a reference to another quantity (a quantity value and its corresponding unit). Ten and 11 semantic mapping rules and conflict resolution rules were developed, respectively.

The preliminary experimental results show more than 90% performance (using measures of P, R, and F) for all information elements. This indicates the potential effectiveness of our approach. Yet, since a 100% performance was not achieved, we conducted an error analysis to identify the sources of error. For example, in one case, the cause for not producing a logic clause element that ought to be produced was the use of uncommon expression structures in the text: in the sentence "…has as a longer side at least 65 percent open and unobstructed", the "quantity reference" was not produced because the adjective "65 percent open and unobstructed" is not a commonly-used expression structure for "quantity reference". For future improvement, to prevent/reduce the detected errors, we propose two ways of adapting and refining our algorithms. First, enhance our ontology-based deductive reasoning to better detect concepts that are implicit in the text (e.g. the 'compliance checking attribute' 'height' was implicit in the part of sentence "ventilators… at least 3 feet above eave or cornice vents"). Second, add pointer-word resolution rules to avoid missing relations between concepts connected by pointer-words (e.g. the pointer-word 'thereof' in the part of sentence "Any room … in two thirds of the area thereof" connects the 'compliance checking attribute' 'area' with the 'subject' 'room').

CONCLUSION AND FUTURE WORK

In this paper, the authors proposed a six-phase iterative approach for extracting requirements from textual regulatory documents and formalizing these requirements in a computer-processable format. The approach combines our text classification (TC), information extraction (IE), information transformation (ITr), and compliance reasoning (CR) algorithms into one computational platform. The paper focuses on presenting and discussing our information transformation algorithms, which transforms the extracted information, into logic-based representation ready for compliance reasoning. Our approach was tested in extracting and formalizing quantitative requirements in Chapter 12 of the 2006 International Building Code. A precision, recall, and F-measure of 96%, 92%, and 94%, respectively, were achieved. The preliminary experimental results show that our proposed approach is promising. For further improvement, we conducted an error analysis. As part of future/ongoing work, the authors will adapt and refine our algorithms to prevent/reduce the detected errors. Our future work on automated compliance checking will also explore automated IE and ITr from other types of construction documents (e.g. contract specifications).

ACKNOWLEDGEMENT
This material is based upon work supported by the National Science Foundation under Grant No. 1201170. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

Eastman, C., Lee, J., Jeong, Y., and Lee, J. (2009) "Automatic rule-based checking of building designs." *Autom. Constr., 18(8*), 1011-1033.

Garrett, J., and Fenves, S. (1987). "A knowledge-based standard processor for structural component design." *Eng. with Comput.*, 2(4), 219-238.

International Code Council (ICC). (2006). "2006 International Building Code." <http://publicecodes.citation.com/icod/ibc/2006f2/index.htm> (Feb. 05, 2011).

Joshi, A. (1991). "Natural Language Processing." *Science, 253*, 1242-1249.

Lau, G., and Law, K. (2004). "An information infrastructure for comparing accessibility regulations and related information from multiple sources." *Proc., 10th Intl. Conf. Comput. Civ. and Build. Eng.*, Weimar, Germany.

Marquez, L. (2000). "Machine learning and natural language processing." *Proc., "Aprendizaje automatico aplicado al procesamiento del lenguaje natural".*

Russell, S., and Norvig, P. (2010). *Artificial intelligence – a modern approach (third edition),* Pearson Education, Inc., Upper Saddle River, New Jersey.

Salama, D., and El-Gohary, N. (2013). "Automated semantic text classification for supporting automated compliance checking in construction". *J. Comput. Civ. Eng., In Review.*

Solibri. (2011). "Solibri Model Checker." <http://www.solibri.com/solibri-model-checker.html> (July 15, 2011).

Sterling, L., and Shapiro, E. (1986). *The art of Prolog: advanced programming techniques,* MIT Press, Cambridge, Massachusetts, London, England.

Tan, X., Hammad, A., and Fazio, P. (2010) "Automated code compliance checking for building envelope design." *J. Comput. Civ. Eng.*, 24(2), 203-211.

University of Sheffield. (2011). "General architecture for text engineering." <http://gate.ac.uk/> (Feb. 05, 2011).

Zhang, J., and El-Gohary, N. (2012). "Extraction of construction regulatory requirements from textual documents using natural language processing technieques." *Proc., Comput. Civ. Eng.*, ASCE, Reston, VA, 453-460.

Zhou, N. (2012). "B-Prolog user's manual (version 7.7): Prolog, agent, and constraint programming." Afany Software. <http://www.probp.com/manual/manual.html> (Nov. 19, 2012).