A SEMI SUPERVISED SEMANTIC ANALYSIS OF NATURAL LANGUAGE CONSTRAINTS USING DRT AND MARKOV LOGIC

Imran Sarwar Bajwa[1], Sadaf Iqbal[2]
[1] Department of Computer Science& IT, The Islamia University of Bahawalpur
[2] Department of Computer Science, NCBA&E, Lahore
imran.sarwar@iub.edu.pk, sadafiqbal516@yahoo.com

ABSTRACT: *In this paper a framework is presented to automate the process of deep semantic analysis of constraints that are written in natural language. Constraints are typically specified in Object Constraint Language (OCL) to complete software models. A couple of approaches have been presented to automatically generate OCL constraints for a particular UML class model. Existing approaches are not much accurate because most of the English constraints are based on composite sentence and it is not possible to extract required information from NL constraints without discourse analysis. We present a novel approach for deep semantic analysis in order to identify the relations between discourse and the emergence of syntactic structure and the relations between text (discourse) and context. The C&C Boxer is a common tool used for DRT analysis. However, it is found after experimentation that tools like C&C Boxer tools perform better if more accurate semantic labelled data is provided. In this paper, the C&C Boxer tool is tested with conventional semantic role labelling and Markov Logic based semantic role labelling and a clear improvement of results is achieved. The results show that discourse analysis greatly improves the accuracy of semantic analysers in order to extract correct information from requirement specifications and interpret constraints.*

Keywords: Discourse Analysis, Natural Language, Deep Semantic Analysis, Expectation-Maximization

1. INTRODUCTION

A constraint is a statement of restriction that limits the range of acceptable values of a variable. Constraints can be general or on opposite side very specific related to an object. Constraints are contained in software requirement specification of a user system. Software requirement specification also contains non-functional requirements. Constraints are imposed on design and implementation by non-functional requirements [1]. Software requirements are usually specified in any natural language sentences such as English. Constraints (see Figure 1.1) may be written in the form of a single sentence or in the form of composite sentences e.g.

Table 1.1: Examples of English constraints

Each **Person** has a *firstName* and a *lastName*, a *birthDate* and an *age*. Each **Person** could be married or not, specified by the boolean value *is Married*. Each **Person** could have a **Grade** specified by the attribute *grade* and be *supervised* by one or no supervisor who is also an instance of the class **Person**. Furthermore, a Person could be the *supervisor* of between zero and n **Person**s. (Dresden-ocl.sourceforge.net, 2015)

Constraints are implemented in design phase to validate an object so that an object may always be in a consistent state. However development of such constraints manually is an error prone process. In model driven engineering (MDE) a model [2] is a considered as the basic building block from which instances (Objects) can be constructed. Models are usually specified graphically by using graphical notations like Unified Modeling Language (UML). Graphical syntax generally allows rough specification. In Meta Model Facility (MOF) models are referred as classes [3]. The instances of a class may grow while not all instances satisfy the semantics of the model therefor, not valid whether being syntactically valid. Therefore, constraints are imposed on design of models which restricts the number of possible instances. These constraints may be legal restrictions, company policies, grant for privileges or may be technical restriction on models. A pattern based approach presented by Wahler *et al*. [21] to create complex constraints and parameterized them in a CASE tool. For concisely maintenance of UML/OCL (Object Constraint Language) specification this research [21] is an important contribution. The approach presented uses logical structure and classification of patterns into an atomic and composite pattern in an abstract way. However future work indicates complexity in inherit constraints [21] due to lack of deep semantic analysis (Discourse Analysis). Object Constraint Language (OCL) is used to translate English specification of constraints to formal specification along with UML which is used to model real world requirements. In order to automate the software engineering tasks NL constraints specification plays a key role. Work has done on shallow semantic and deep semantic analysis of NL constraints. Shallow semantics [8, 9] fails to find relations between composite sentences for this deep analysis/discourse analysis is used.

In last two decades, a research domain of automated software engineering has emerged. Various approaches have been presented to automatically analysis natural language software requirement specification constraints and generate UML class diagram such as UML class models [7, 13, 23]. Since, OCL based constraints are integral part of typical UML class model. A couple of approaches have been presented to automatically generated OCL constraints for a particular UML class model. A pattern based approach was introduced [21] to automatically generated OCL constraints. Similarly another approach was presented [1] to automatically generated OCL constraints from NL software requirement specifications [10]. In this research, NL processing techniques were used to automatically analyzed NL software constraints and model that transform that information to OCL through SBVR. A couple of case studies were also resolved by such approaches however, it is mentioned in the future work of "Bajwa" [2] that the performed semantic analysis is incomplete due to absence of discourse analysis. Various

researches [5-8] have shown that discourse analysis can improve accuracy of semantic analysis. Since most of the NL constraints are based on composite sentences. It is not possible to extract required information from NL constraints without discourse analysis. Here, discourse analysis becomes an important requirement of semantic analysis of constraints especially the NL constraints are based on composite sentences.

2. Background and Related Work

Software engineering tasks are automated by using case tools. This automation requires the correct analysis of software requirements which are generally written in natural language. Natural language is ambiguous and non-consistent due to its flexible syntax. First these requirements need to be written in formal some formal language with controlled syntax to reduce ambiguity so that models can be accurately generated. Some researchers [4, 14] proposed Natural Language Processing (NLP) techniques for refining the software requirements and get their accurate semantics. A lot of work is done on shallow semantic and deep semantic of natural language. Text annotation techniques are used for shallow semantic analysis and Discourse representation theory (DRT) for deep semantic analysis [3, 5]. These all techniques together with shows a great accuracy but still leaves gap for research as model consistency remains incomplete without constraints. The limitations imposed on the design should be correctly reflected in models for consistent and accurate design. To incorporate constraints in models different methods and techniques are propose and successfully implemented [1]. But all these methods and techniques involve intermediary representation, a formal representation (such as controlled English or SBVR) before translating requirements into models. Recent research [3, 4, 5, 6] in the field of NLP and DRT makes it possible to do analysis task under a single format. However this work still needs attention because constraints are not translated into models. Research shows that DRT can be helpful to analyse constraints and resolve reference objects on which constraints are imposed. Efficiency and performance is achieved by developing API for two different parsers for Rhetorical Discourse Theory (RDT) [18]. Features of first parser based on dependency syntax, produced by a fast shift-reduce parser. Second parser is used to resolve constituent and dependency syntax and co-reference information using Stanford CoreNLP toolkit [2].

2.1 Discourse Analysis

Discourse analysis [12] is a phenomenon that is beyond the scope of what we get from interpretation of individual sentences. It tries to find what hearer actually infers from these sentences. Sentences are building blocks for discourse structure. Normally people speak freely. They talk about one subject then go out of the way and change subject altogether. They hardly follow formal constraints. Their conversation is determined by the need of speaker and its perception of listener [22]. A discourse is a linguistic phenomenon which reflects speakers thought. In oral communication face expressions and body gestures help to reflect ideas of speakers and listener can follow the focus of conversation easily. But in written communication it is hard to identify what is discourse about? Therefore, a discourse structure helps to interpret the speaker's idea to listener. DRSs are mental representations of discourse. A Discourse Representation Structure (DRS) [18] has two parts, A Discourse Referent (DR) and the set of conditions. DRs are variables to represent entities of discourse and conditions are logical connections between entities. Discourse entities can be denoted in four different ways [22].

- Names
- Definite descriptions ('the tall woman')
- Indefinite descriptions ('a tall woman')
- Pronouns

Definite descriptions are expressions to denote unique or specific things. 'The' is definite article. Indefinite descriptions are expressions are used to denote member of a class of entities. 'A' and 'An' are indefinite articles. DRS condition may be atomic or complex. An atomic case it consist of a predicate and suitable number of discourse referents whereas in complex case it combines two DRSs [11]. Following is syntax of DRS by Geurts& Beaver [11].

- A DRS **K** is a pair $< U_{\mathbf{K}}, Con_{\mathbf{K}}>$, where $U_{\mathbf{K}}$ is a set of discourse referents, and $Con_{\mathbf{K}}$ is a set of DRS-conditions.
- If P is an n-place predicate, and $\mathbf{x_1}, \ldots, \mathbf{x_n}$ are discourse referents, then $\mathbf{P(x_1, \ldots, x_n)}$ is a DRS-condition.
- If **x** and **y** are discourse referents, then $\mathbf{x = y}$ is a DRS-condition.
- If **K** and **K′** are DRSs, then $\neg\mathbf{K}$, $\mathbf{K \Rightarrow K'}$, and $\mathbf{K \vee K'}$ are DRS-conditions.
- If **K** and **K′** are DRSs and **x** is a discourse referent, then $\mathbf{K(\forall x)K'}$ is a DRS-condition.

Used Framework

The used framework is depicted in Figure. The presented approach incorporates the Discourse Representation Theory (DRT) to identify the discourse structures and finding relationships among the identified discourse structures.

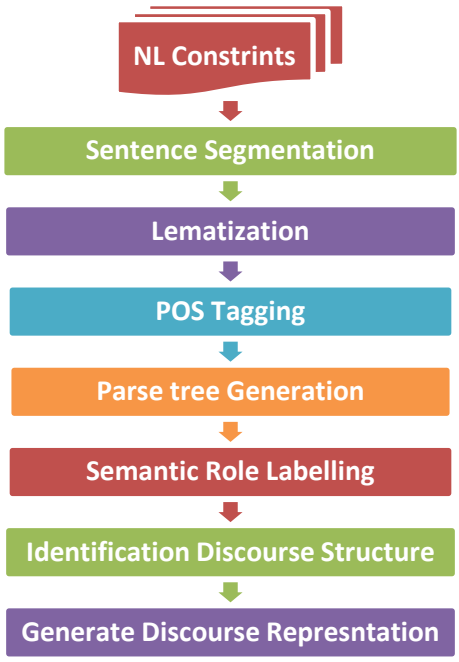

Figure 3.1: Used Framework for Discourse Analysis of NL Software Constraints

" Every Student is a Person Who has a matriculation number and a matriculation date . "

3.1 Sentence Splitting

Sentence splitting is the process of dividing a string of text into meaningful component sentences by identifying sentence boundaries or punctuation marks. Sentence splitting is a deterministic consequence of tokenization. Typically a sentence boundary is determined when a (. , ! ?) character is found which is not grouped with other character. A simple way to find a boundary is looking for a period (.) followed by space which followed by a capital letter. However, Mark a sentence boundary is not as simple. A period may or may not imply sentence boundary. A full stop character may use for abbreviation (E.g.; "Dr. Sarwar ". Here Dr. is not a sentence itself. It is a part of the sentence.). Similarly a single quote or brackets may be a part of the sentence.

3.2 Lemmatization

Tokenization is the process of breaking a stream of text into meaningful elements called tokens. A token may refers as a word, phrase or a symbol. A tokenizer is a program that takes text stream as input and breaks it in tokens [13]. Normally a tokenizer uses some heuristics to identify a token. Such as

- Space character, punctuation marks is used to identify a token.
- A Specific Sequence of characters (flag).
- Explicit definition of dictionary

Example 3.1

A constraint written in English "Every Student is a Person who has a matriculation number and a matriculation date." Is tokenized as:

Process of identifying base form of a word is called lemmatization. In natural languages such as English words are appeared in different inflected forms to reflect meanings according to tense, mode, number or gender etc. E.g., Move can be used as move to, moved, moves, moving. In this process the inflected forms of words [14] and parts of speech are grouped together so that they can be analyzed as a single term. E.g.; "We are having a trouble." In this sentence having is inflected form of verb 'have'. This inflection is due to the tense.

3.3 POS Tagging

POS (Part of Speech) tagging is one of the annotation process in which a word is marked up according to its grammatical definition and context [15]. Each word in text corresponding to a particular part of speech(Noun, verb, adjective etc.) with respect to its relationship with adjacent words in a sentence.

Example 3.2 POS tagging of a sentence.

Tags	Tokens
DT	Every
NNP	Student
VBZ	is
DT	a
NNP	Person
WP	who
VBZ	has
DT	a
NN	matriculation
NN	number
CC	and
DT	a
NN	matriculation
NN	date
.	.

3.4 Parse Tree Generation

Parse tree is an ordered tree that represents syntactical structure of a sentence. It is generated by orderly assigning syntactic constituents by analyzing grammatical constituents, part of speech and syntactic relations. A constituency based parse tree generated from phrase structure grammar. It

distinguishes between terminal and non-terminal. Tree structure [16] generally has a root node, interior nodes and leaf nodes. Root node is a node that doesn't dominated by any node, interior nodes(branch nodes) are mother nodes that connects to two or more daughter nodes and leaf nodes are ending nodes that doesn't dominate other nodes in a tree.

Figure 3.2 shows a parse tree for phrase "Every Student is a Person who has a matriculation number and a matriculation date."is:

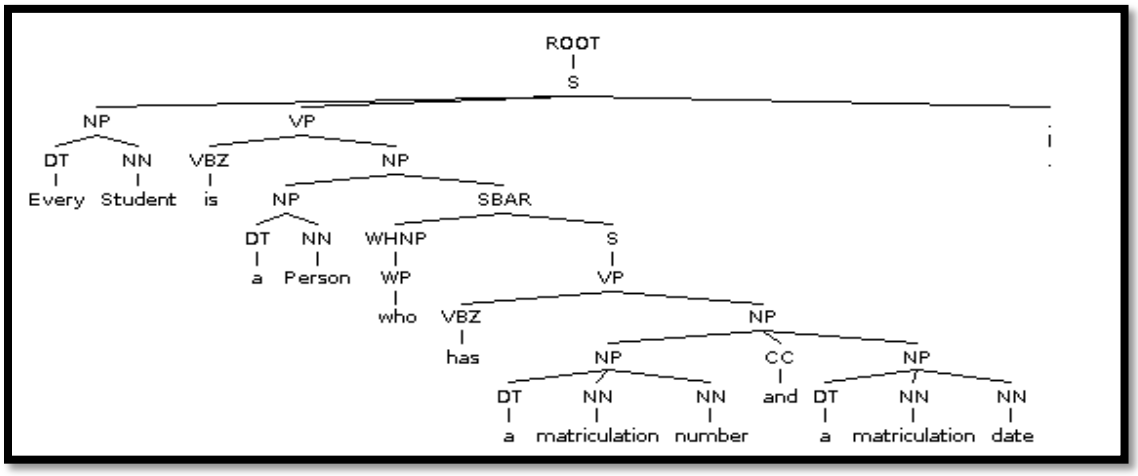

Figure 3.2: Parse tree generation

Here Noun Phrase (NP) and Verb Phrase (VP) are grammatical units. Phrase construction is an intermediary process between identifying POS (Part Of Speech) in a sentence and generation of full parse tree [18-19]. This process is known as chunking. In chunking a sentence is divided into sub constituent phrases (noun, verb or prepositional phrases).

3.5 Semantic Role Labelling

In semantic role labeling role of each term in a phrase is assigned to make sense of meaning of a sentence. It is a higher level of extraction than a parse tree. Some sentences may have same semantic meanings but different syntactic forms.

For example

"Every Student is a Person who has a matriculation number and a matriculation date."

"A person having a matriculation number and a matriculation date is a Student."

Since, in these both sentences, there are same semantic role but different syntactic forms and such complex role labeling with conventional semantic role labelers provides less accuracy. However, in our approach we propose the use of Markov Logic [16] to label semantic roles by determining features and their weights of each input text. Mostly features like predicate, path, constituent type, position and head word are easily extracted from syntactic parse of a sentence. For classification of semantic roles, Markov Logic represents the features of the input data in terms of nth joint distribution as shown in equation (1):

$$P(X=x) = \frac{1}{Z}\prod_{k}\phi_k(x_{\{k\}}) \tag{1}$$

Now, the joint distribution of a model is mapped as a set of variables i.e. $X \in (X_1, X_2, \ldots, X_n)$. Typically, in a network of Markov Logic, a set of pairs (F_i , w_i) represent a predicate where a predicate in First Order Logic (FOL) is represented by F_i and a real number depicts w_i that is weight of the predicate/formula. In the used approach, the weights are updated by using equation (2) that is based on statistical relational learning approach and is incorporated by combining probability with the traditional first-order logic. Here, a typical MLN (Markov Logic Network) [16] with a set of weights and formulas can be represented as below:

$$P(X=x) = \frac{1}{Z}\exp\left(\sum_{j} w_j f_j(x)\right) \tag{2}$$

$$w = w + D^{-1}\, g \tag{3}$$

The weights of the formulas are dynamically updated by using diagnolized Newton Method [15]. Here, the weight update formula is shown in equation (3). The classification of output is used in next phase for identification of discourse structures.

3.6 Identification of Discourse Structure

Each language has its own text structure. Discourse analysis is done to identify discourse structure in text. Models are generated to identify discourse elements. A representation for discourse structure consists of discourse referent and discourse condition.Discourse representation proposed a set of relations (such as evidence, concession, justification, background, circumstance, etc.). These can be further refined, extended, and applied to corpus annotations. Identification of discourse relation is done by using Penn Discourse Tree Bank (PDTB) having millions of word corpus annotated with discourse [17]. PDTB annotation scheme does not assume a particular structure. PDTB annotation involves span detection for explicit and implicit discourse connectives ("but", "because", "when", etc.) and disambiguation of discourse connectives. The PDTB annotation scheme has three levels of

granularity: four classes at the topmost level - "Comparison", "Contingency", "Temporal" and "Expansion" that are further expanded. Annotator can choose among three levels. .7 Generate Discourse Representation Structure

Discourse Representation Structure (DRS) is a box like structure which contains discourse referent and conditions. Semantic representation of meanings of sentence used grammatical and semantic attachment rules. According to grammar rule sentence is split into subject, predicate and object [20].

Example 3.2 Illustration of semantic operators

Every **Student** is a **Person** who has a matriculation number and a matriculation date.

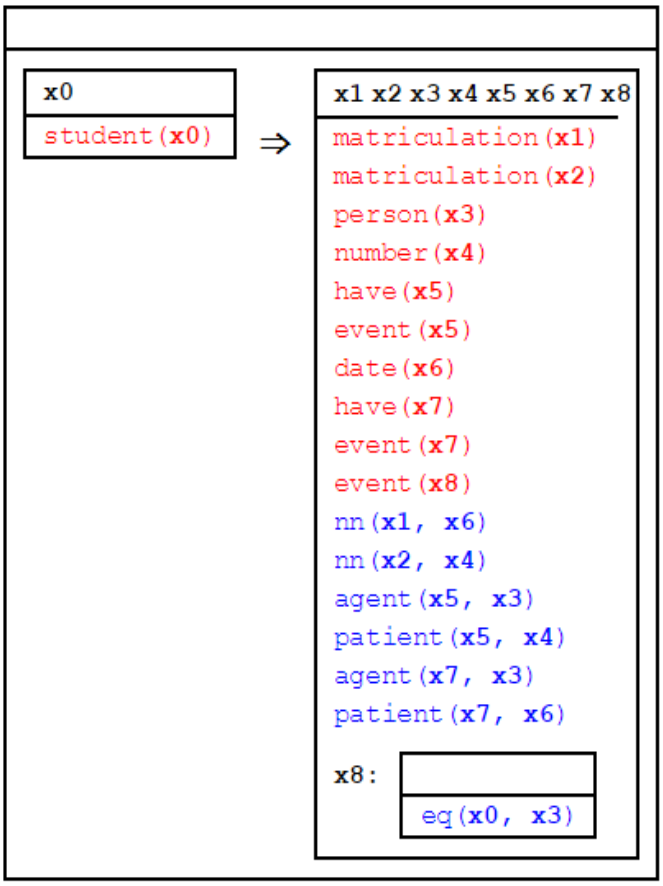

Figure 3.2: An Example of DRS Representation (Box notation)

3. EXPERIMENTS AND RESULTS

Following are a few examples solved the used approach and Table 4.1 and Table 4.2 show the detailed parsing of the given examples.

A customer may place an order only if he holds an account. An order must not be shipped if the outstanding balance of the account exceeds the credit authorization of the customer.

A project team member must not be rotated off the project until finished.

Table 4.1.: Detailed parsing of input example given above

ID	Form	PLemma	PPOS	PHead	PDeprel	IsPred	Pred	Args: place.01	Args: order.01	Args: hold.01	Args: account.01	Args: ship.01	Args: balance.02	Args: exceed.01	Args: authorization.01
1	A	A	DT	2	NMOD	_	_	_	_	_	_	_	_	_	_
2	customer	customer	NN	3	SBJ	_	_	A0	A0	_	_	_	_	_	_
3	may	May	MD	0	ROOT	_	_	AM-MOD	_	_	_	_	_	_	_
4	place	place	VB	3	VC	Y	place.01	_	_	_	_	_	_	_	_
5	an	An	DT	6	NMOD	_	_	_	_	_	_	_	_	_	_
6	order	order	NN	4	OBJ	Y	order.01	A1	A1	_	_	_	_	_	_
7	only	Only	RB	8	ADV	_	_	_	_	_	_	_	_	_	_
8	if	If	IN	4	ADV	_	_	AM-ADV	_	_	_	_	_	_	_
9	he	He	PRP	10	SBJ	_	_	_	_	A0	A0	_	_	_	_
10	holds	Hold	VBZ	8	SUB	Y	hold.01	_	_	_	_	_	_	_	_
11	an	An	DT	12	NMOD	_	_	_	_	_	_	_	_	_	_
12	account	account	NN	10	OBJ	Y	account.01	_	_	A1	_	_	_	_	_
13	.	.	.	3	P	_	_	_	_	_	_	_	_	_	_
14	An	An	DT	15	NMOD	_	_	_	_	_	_	_	_	_	_
15	order	order	NN	16	SBJ	_	_	_	_	_	_	A1	_	_	_
16	must	must	MD	3	DEP	_	_	_	_	_	_	AM-MOD	_	_	_
17	not	Not	RB	16	ADV	_	_	_	_	_	_	AM-NEG	_	_	_
18	be	Be	VB	16	VC	_	_	_	_	_	_	_	_	_	_
19	shipped	Ship	VBN	18	VC	Y	ship.01	_	_	_	_	_	_	_	_
20	if	If	IN	19	ADV	_	_	_	_	_	_	AM-ADV	_	_	_
21	the	The	DT	23	NMOD	_	_	_	_	_	_	_	_	_	_
22	outstanding	outstanding	JJ	23	NMOD	_	_	_	_	_	_	_	_	_	_
23	balance	balance	NN	27	SBJ	Y	balance.02	_	_	_	_	_	_	A0	_
24	of	Of	IN	23	NMOD	_	_	_	_	_	_	_	A1	_	_
25	the	the	DT	26	NMOD	_	_	_	_	_	_	_	_	_	_
26	account	account	NN	24	PMOD	_	_	_	_	_	_	_	_	_	_
27	exceeds	exceed	VBZ	20	SUB	Y	exceed.01	_	_	_	_	_	_	_	_
28	the	the	DT	30	NMOD	_	_	_	_	_	_	_	_	_	_
29	credit	credit	NN	30	NMOD	_	_	_	_	_	_	_	_	_	A1
30	authorization	authorization	NN	27	OBJ	Y	authorization.01	_	_	_	_	_	_	A1	_
31	of	Of	IN	30	NMOD	_	_	_	_	_	_	_	_	_	A0
32	the	The	DT	33	NMOD	_	_	_	_	_	_	_	_	_	_
33	customer	customer	NN	31	PMOD	_	_	_	_	_	_	_	_	_	_
34	.	.	.	3	P	_	_	_	_	_	_	_	_	_	_

Table 4.2.: Detailed parsing of input example given above

ID	Form	PLemma	PPOS	PHead	PDeprel	IsPred	Pred	Args: team.01	Args: member.01	Args: rotate.02	Args: finish.01
1	A	A	DT	4	NMOD	_	_	_	_	_	_
2	project	Project	NN	3	NMOD	_	_	A1	_	_	_
3	team	team	NN	4	NMOD	Y	team.01	_	A1	_	_
4	member	member	NN	5	SBJ	Y	member.01	_	_	A1	_
5	must	must	MD	0	ROOT	_	_	_	_	AM-MOD	_
6	Not	not	RB	5	ADV	_	_	_	_	AM-NEG	_
7	Be	be	VB	5	VC	_	_	_	_	_	_
8	rotated	rotate	VBN	7	VC	Y	rotate.02	_	_	_	_
9	Off	off	IN	8	ADV	_	_	_	_	A2	_
10	The	the	DT	11	NMOD	_	_	_	_	_	_
11	project	project	NN	9	PMOD	_	_	_	_	_	_
12	until	until	IN	8	TMP	_	_	_	_	AM-TMP	_
13	finished	finish	VBN	12	PMOD	Y	finish.01	_	_	_	_
14	.	.	.	5	P	_	_	_	_	_	_

The developed tool, for Semantic annotation of entities in business rule, takes business rule text document as an input then performs different steps on it. The objective of this research is to annotate the entities in business rules. Our plan for future research is that to increase the performance and accuracy of this tool. Our future research will go through Semantic annotation in other Business Rules.

Table 4.3: Results of Semantic role labelling and discourse representation with and without Markov Logic

Category	Total Roles	Correct Roles	Missed Roles	Incorrect Roles	Recall	Precision
Semantic Role labelling without Markov Logic	36	30	2	3	86.11%	91.17%
Semantic Role labelling with Markov Logic	36	33	1	2	91.66%	93.93%
Discourse Representation without Markov Logic	20	16	1	3	80.33%	84.21%
Discourse Representation with Markov Logic	20	18	0	2	90.00%	90.90%

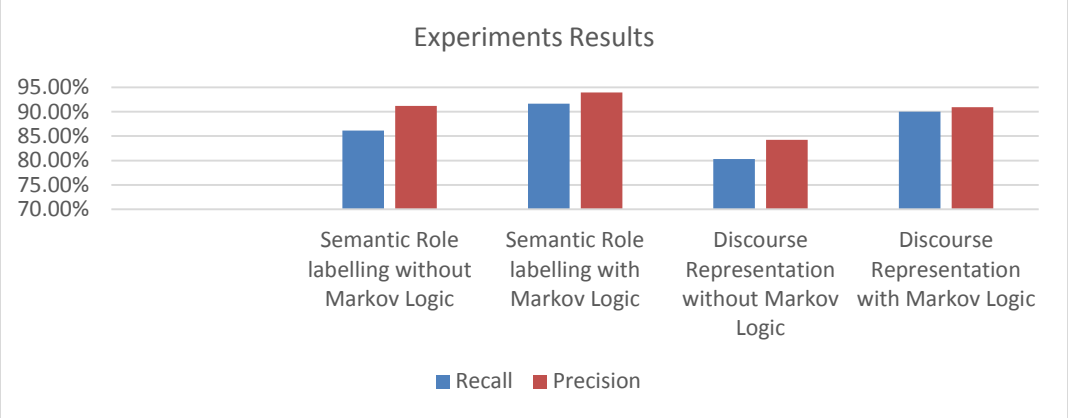

Figure 4.1: Results of experiments

The results shown in Table 4.3 and Figure 4.1 clearly mention that the recall and precision of Markov Logic based semantic role labelling and discourse representation is higher than the conventional approaches used for the same task.

4. CONCLUSION

The approach used for semantic analysis of NL constraints, generates a discourse representation structure after identifying the relation between text and context. The used approach first annotates the given constraints by using natural language techniques for semantic analysis. In step by step process, sentences are split in given text constraints then tokenized the phrases. After tokenization POS are marked according to their grammatical definition and semantic roles are labelled. After tagging parse tree is generated. By analysing tree grammatical relationships are represented in the form of dependencies. After this discourse relations are identified by using DRT and generated a discourse representation structure which finally mapped to FOL. It helps to resolve many implicit, explicit and hidden relations in discourse. The presented approach is also implemented and is available online as C&C Boxer. A set of experiment is conducted in last section. We performed discourse analysis constraints on different real life examples. Our methodology frame work can greatly improve the efficiency of analysers to automatically translate the software constraints into UML models and can be incorporated as an integral part. The used methodology framework is a domain specific solution which helps to automatically analyse natural language constraints of software requirements by using Discourse Analysis. However performed discourse analysis is limited to resolve basic anaphoric relations and discourse markers. It is unable to resolve complex relations in multiple sentences.

5. REFERENCES

[1] Bajwa, I., Lee, M., & Bordbar, B. (2012). Translating Natural Language Constraints to OCL. Journal of King Saud University - Computer And Information Sciences, 24(2), 117-128. Doi:10.1016/J.Jksuci.2011.12.003

[2] Bajwa, I.S., Lee, M.G., 2011a. Transformation Rules For Translating Business Rules To OCL Constraints, 7th European Conference On Modelling Foundations And Applications (ECMFA 2011). Birmingham, UK. Jun 2011. Pp:132-143, Birmingham, UK

[3] Basile, V., Bos, J., Evang, K., & Venhuizen, N. (2012). Developing A Large Semantically Annotated Corpus. In LREC (Vol. 12, Pp. 3196-3200).

[4] Boyd, N. (1999). Using Natural Language In Software Development. Journal Of Object-Oriented Programming, 11(9), 45-+.

[5] Bunt, H. (2015, April). On The Principles Of Interoperable Semantic Annotation. Inproceedings Of The 11th Joint ACL-ISO Workshop On Interoperable Semantic Annotation (Pp. 1-13).

[6] CRIBLE, L., & ZUFFEREY, S. (2015, April). Using A Unified Taxonomy To Annotate Discourse Markers In Speech And Writing. In Proceedings 11th Joint ACL-

ISO Workshop On Interoperable Semantic Annotation (Isa-11) (P. 14).

[7] Bajwa, I. S., Lee, M. G., & Bordbar, B. (2011, March). SBVR Business Rules Generation from Natural Language Specification. In *AAAI spring symposium: AI for business agility* (pp. 2-8).

[8] Dresden-Ocl.Sourceforge.Net,. (2015). The University Model - An UML/OCL-Example. Retrieved 3 January 2015, From Http://Dresden-Ocl.Sourceforge.Net/Usage/Ocl22sql/Modelexplanation.Html

[9] Gabbay, D., & Guenthner, F. (2011). Handbook Of Philosophical Logic. Dordrecht: Springer.

[10] Gast, V., Bierkandt, L., & Rzymski, C. (2015, April). Creating And Retrieving Tense and Aspect Annotations With Graphanno, A Lightweight Tool For Multi-Level Annotation. In Proceedings 11th Joint ACL-ISO Workshop on Interoperable Semantic Annotation (Isa-11) (P. 23).

[11] Maier, E. (2016). Attitudes and Mental Files in Discourse Representation Theory. *Review of Philosophy and Psychology*, 1-18.

[12] Hoek, J., & Zufferey, S. (2015, April). Factors Influencing the Implication Of Discourse Relations Across Languages. In Proceedings 11th Joint ACL-ISO Workshop on Interoperable Semantic Annotation (Isa-11) (P. 39).

[13] Ilieva, M., & Boley, H. (2008). Representing Textual Requirements as Graphical Natural Language For UML Diagram Generation. In SEKE (Vol. 8, Pp. 478-483).

[14] Juristo, N., & Moreno, A. M. (1997). Object Oriented Modeling Focused On A Linguistic Approach. In 22nd Annual NASA Software Engineering Workshop.

[15] Lawless, A. S., Nichols, N. K., Boess, C., & Bunse-Gerstner, A. "Approximate Gauss–Newton methods for optimal state estimation using reduced-order models", *International journal for numerical methods in fluids*, *56*(8), 1367-1373, 2008.

[16] Richardson, M., & Domingos, P. "Markov Logic Networks", *Machine learning*, *62*(1-2), 107-136, 2006.

[17] Signalprocessingsociety.Org,. (2015). SLTC Newsletter : February 2011 : Automatic Identification Of Discourse Relations In Text - IEEE Signal Processing Society. Retrieved 22 February 2015, From Http://Www.Signalprocessingsociety.Org/Technical-Committees/ List/Sl-Tc/Spl-Nl/2011-02/Discourse-Relations/

[18] Surdeanu, M., Hicks, T., & Valenzuela-Escárcega, M. A. (2015). Two Practical Rhetorical Structure Theory Parsers. Proceedings Of The North American Chapter Of The Association For Computational Linguistics (NAACL): Software Demonstrations.

[19] Svn.Ask.It.Usyd.Edu.Au,. (2015). Boxer - C&C Tools - Trac. Retrieved 21 February 2015, From Http://Svn.Ask.It.Usyd.Edu.Au/Trac/Candc/Wiki/Boxer

[20] Prasad, R., & Bunt, H. (2015, April). Semantic relations in discourse: The current state of ISO 24617-8. In *Proceedings 11th Joint ACL-ISO Workshop on Interoperable Semantic Annotation (ISA-11)* (pp. 80-92).

[21] Wahler, M., Koehler, J., & Brucker, A. (2007). Model-Driven Constraint Engineering. Electronic Communications Of The EASST, 5(1863-2122).

[22] Www-Rohan.Sdsu.Edu,. (2015). Discourse. Retrieved 11 February 2015, From Http://Www-Rohan.Sdsu.Edu/~Ling354/Discourse.Html#Discourse Segments

.