Can Rules of Inferences Resolve Coordination Ambiguity in Natural Language Requirements Specification?

[1]Sri Fatimah Tjong and [2]Daniel M. Berry
[1]*University of Nottingham, Malaysia Campus, Malaysia*
[2]*University of Waterloo, Waterloo, ON, Canada*
[1]kcx4sfj@nottingham.edu.my, [2]dberry@uwaterloo.ca

Abstract

Coordination ambiguity is the ambiguity that occurs from the use of coordinators such as and *and* or*. The first author's previous work tried to resolve coordination ambiguity in natural language (NL) requirements specifications (RSs) by using language patterns derived from rules of logic (RLs). However, a reevaluation of the language patterns show that RLs cannot fully resolve coordination ambiguity in a RS because the coordinators are not always truth functional, due to incompatible interpretations that a human ascribes to the ambiguous RS. Furthermore, human perceptions in the interpretation of the coordinators in a RS vary incredibly widely. The implications of this observation on requirements elicitation are discussed.*

1 Introduction

Ambiguity in natural language (NL) is a major problem in scientific disciplines and in natural language requirements specifications (RSs). A statement is *ambiguous* when it has more than one distinct meaning. An ambiguous RS can have hazardous consequences for its software development project as a whole, in which wrongly implemented requirements cause high costs for rework, delayed product releases, failure, or more than one of these [17, 18, 19].

This paper addresses coordination ambiguity in RS [5] as a challenging issue. Coordination ambiguity is the ambiguity that occurs from the use of coordinators such as and and or. Coordination ambiguity is a very common form of syntactic ambiguity in RS, because of the frequent need to use and and or in a RS. This paper focuses on the coordinators and and or, which account for approximately 3% of the words in the British National Corpus (BNC) [5] and are also the most common causes of coordination ambiguity. Words and phrases of all types can be coordinated. The external modifier can be a word or phrase of almost any type and can appear either before or after the coordinator [7]. Consider the phrase secure hardware and software. It can be parsed as either:

a. (secure hardware) and (software)
b. (secure (hardware and software))

Identifying which two modifiers are conjoined by coordinator and is necessary in order to arrive at a correct interpretation of the phrase. Understanding the phrase according to the structure (a) leads to only the hardware's being secure. However, understanding the phrase according to the structure (b) leads to both the hardware's and software's being secure. The reading (a) is a coordination last reading, and the reading (b) is a coordination first reading. As in many situations, it is possible that context will disambiguate. A variation of this phrase suffering even more ambiguity is:

secure hybrid hardware and software

It can be parsed as any of:

(secure hybrid hardware) and (software)
(secure hybrid (hardware and software))
(secure (hybrid hardware)) and (software)
(secure ((hybrid hardware) and software))

The first author's previous work [1, 2] presented a set of language patterns based on rules of logic (RLs) that were meant to resolve coordination ambiguity resulting from the use of the coordinators that are the logical operators and and or. The language patterns were seemingly able to remove coordination ambiguity by simplifying according to RLs. However, later studies showed that disambiguation of coordinators and and or by RLs is not plausible because human perceptions of the interpretations of the coordinators in a RS can vary widely. In addition, coordinators are not always found to be truth functional, because the state-

ment resulting after simplification by RLs has interpretations that are incompatible with those of the original ambiguous statement.

Hence, this paper argues that RLs cannot resolve coordination ambiguity. RLs are unable to consider (1) domain-specific information, (2) where the coordinator occurs, (3) what parts of speech occur on both sides of the coordinator, and (4) any other factors that are relevant in disambiguating the use of the coordinator. Determining how these various elements interact is challenging. Thus, research has been going on to find simpler ways to write less ambiguous requirements statements (RStats) that avoid ambiguous use of coordinators [1, 2, 3, 4, 5, 6].

Throughout this paper, the RStats given are obtained from studies on several sets of industrial strength RSs [4]. Text from a RStat or other example is typeset in a sans serif typeface, and this text may end with punctuation.

Section 2 describes past work on coordination ambiguity in NL and NL RSs. Section 3 shows why RLs are unable to resolve coordination ambiguity. Section 4 discusses the implications on requirements elicitation of the observations of Section 3. Section 5 discusses the role of guiding rules in *avoiding* ambiguity and future work. Section 6 concludes the paper.

2 Past Work

Coordinators such as and and or are potential sources of syntactic ambiguity in NLs and in NL RSs. Since their interpretations directly affect the meaning of the text, their disambiguation is critical in order to understand the text precisely. However, they have always been problematic because they are not always used in truth-functional ways. When one of these coordinators is the main connective in a compound sentence, the truth value of the compound sentence does not depend in all cases on solely the coordinator and the truth values of the component sentences [8].

Research has been done on resolving the coordination ambiguity in NL RSs [1, 2, 5, 6, 13]. Work on coordination ambiguity in NL has focused on matching patterns of coordinators [8, 9, 10].

Agarwal and Boggess [10] identified a coordinator by matching part of speech (POS) and semantic tags of the words modified by the coordinator. Resnik [12] proposed a semantic-similarity approach to disambiguate coordinators that involves nominal compounds. Goldberg [11] applied a coocurrence-based probabilistic model to determine the attachments of ambiguous coordinators in phrases, using unsupervised learning.

Chantree *et al*. [5] presented a binary classifier for coordination ambiguity. They collected a dataset of ambiguous phrases from a corpus of requirements specifications, and a collection of associated human judgements on their interpretations. Their model was based on word distribution information obtained from BNC.

Fuchs, Schwitter, and Schwertel [13, 14] proposed a restricted NL called Attempto Controlled English (ACE). In ACE, coordinators are *defined* to be truth functional. ACE thus avoids the inherent ambiguity of unrestricted NL. Specifications written in ACE can be translated into formal logic.

Some works define language patterns that transform an unstructured NL RStat into a structured NL RStat or even into a formal specification [e.g. 15, 16].

Tjong collected several sets of industrial-strength RSs in order to observe patterns of NL RStats commonly found in RSs. Based on the observations, she produced some general yet standardised language patterns using RLs. The suggested language patterns were applied to rewrite the RStats that contained coordination ambiguities. She produced also some guiding rules that help avoid ambiguities [1, 2, 3, 4].

3 Reevaluation of Coordinator Patterns

This section shows a number of ways that and and or can be used in manners that invalidate ambiguity remedies that are based on RLs.

3.1 Ambiguity of the and Coordinator

The conjunctive coordinator and is very ambiguous because a phrase *A* and *B* can express a number of meanings other than the logical conjunction of *A* and *B*:

- *B* follows *A* chronologically, e.g., The computer stops all programs and shuts down.,
- *B* follows *A* logically, e.g., The system shall timestamp and record the initiation and completion of all tasks. (Here, the recording happens as a result of the timestamping.),
- *B* is in contrast to *A*, in which case and is often replaced by but, e.g., The lift should not be stopped from fast mode but should always be switched to slow mode for at least 1 second before stopping.,
- *B* is a surprise given *A*, in which case and is often replaced by yet, e.g., She is only 13; yet, she is already a university student., and
- *B* is dependent on *A*, e.g., Score well and you will win this game!.

For an example of how strict application of a RL-based rule can upset the meaning of a sentence, con-

sider the RL that if *A* and *B* then *A* and *B*, i.e., from the logical conjunction *A* and *B*, one is allowed to say each of *A* and *B*. There are RL-based rules that transform a Rstat of the form

Subject Verb Phrase1 and *Phrase2*.

into two Rstats:

Subject Verb Phrase1.
and
Subject Verb Phrase2.

For

E0: The system shall provide input and output.,

the transformation is valid. However, for

E1: The message is placed in the Outbox and marked as queued.,

the transformation to two Rstats

E2.1: The message is placed in the Outbox.

and

E2.2: The message is marked as queued.

loses the temporal connection between the actions. Transforming

E3: For logging and chat tools, the system shall allow for 'Undo' and 'Redo' functions.

into

E4.1: For logging and chat tools, the system shall allow for 'Undo' function.

and

E4.2: For logging and chat tools, the system shall allow 'Redo' function

loses the dependency relationship between the 'Undo' and 'Redo' actions. In many a system, the actions 'Undo' and 'Redo' are inseparable to the extent that when a system supports the 'Undo' action, it supports also the 'Redo' action. Rewriting

E5: Employees and supervisors are compensated and retained based on their performance and contribution to mission.

as

E6.1: Employees and supervisors are compensated based on their performance and contribution to mission.

and

E6.2: Employees and supervisors are retained based on their performance and contribution to mission.

breaks the relationship of compensation and retention if the intent of E5 is to say that after being compensated, employees and supervisors are also retained and are not dismissed or do not quit. Likewise, blindly transforming

E7: If a search is done and one match is found, the search and search-all-names command buttons are not enabled.

into

E8.1: If a search is done and one match is found, the search command button is not enabled.

and

E8.2: If a search is done and one match is found, the search-all-names command button is not enabled.

causes the misconception that disabling either the search command button or the search-all-names button is sufficient when a match is found. However, the intent of E7 may be the disabling both of the search command buttons after a match is found. That is, the intent of E7 is to prevent user's clicking or misclicking either search button and thus requesting the system to search based on the previously returned search result.

The use of an and to pair nouns can convey that the elements of a set are to be considered together, e.g, as in Each user shall enter his user id and password in order to login to the system. However, and can convey also that each element of the set is to be considered separately, e.g., as in Submission of data shall be supported through web based interactions and file submission. Thus, an and can cause problems regardless of where it is positioned in its sentence.

3.2 Uncertainty of the or Coordinator

The coordinator or has two ambiguities. First, when several elements are joined with an or, it is unclear whether all the elements in the disjunction are to be attributed (1) to one coordinator, (2) to the other, e.g., as in He is to pay a $500 fine or to spend ten days in jail. or (3) to both of them e.g., as in The stocks are obtainable at Walmart or Carrefour., which implies that both Walmart and Carrefour have the stocks and one is able to obtain the stocks at either or both places. Second, when an or occurs in the context of a negation, the or can be interpreted as and [1, 2, 14], e.g., She doesn't speak Spanish or French.

In English, the disjunctive coordinator, or, carries two interpretations:

- Inclusive or: a disjunction is true when at least one of its disjuncts is true; a disjunction is false only when all of its disjuncts are false.
- Exclusive or: a disjunction is true when at exactly one of its disjuncts is true; a disjunction is false when more than one disjunct is true or when all of its disjuncts are false.

By contrast, Latin has two different disjunctive words, vel, for inclusive or, and aut, for exclusive or.

As an example of the first ambiguity, the sentence

> **E9:** If the user requests to close a window or exit the system after making uncommitted changes to a screen, then the system shall prompt the user to commit or cancel those changes.

has two disjunctions. More likely, the first is inclusive and the second is exclusive, but only the customer can say for sure.

A negated or derives a conjunctive interpretation as in

> **E10:** No data or log files will be deleted by the system without immediate or prior approval by an operator or other appropriate personnel.

E10 means to say the system shall not delete data and shall not delete log files without approval from either an operator or other appropriate personnel. We assume that obtaining approval from either an operator or appropriate personnel is sufficient to authorise the deletion, but only the customer can say for sure.

There is yet another use of or, to introduce an alternative, as in:

> **E11:** If the input is correct, then the system shall calculate the temperature, or if not, then the system shall issue an error message.

In this case, the or is not truth functional, and it and the if not, then should be replaced by Otherwise,, by Alternatively,, or by only the If not, then, e.g.:

> **E12:** If the input is correct, then the system shall calculate the temperature. Otherwise, the system shall issue an error message.

3.3 Reevaluation of the Combination of and and or

The occurrence of both and and or in one sentence can complicate the sentence and introduce yet more ambiguity to the sentence, based on the difficulty of determining precedences of the coordinators. For example,

> **E13:** Aircraft that are non-friendly and have an unknown mission or have the potential to enter restricted airspace within 5 minutes shall raise an alert.

E13 has two possible parses:

> **E14.1:** Aircraft that ((are non-friendly) and (have an unknown mission)) or (have the potential to enter restricted airspace within 5 minutes) shall raise an alert.

and

> **E14.2:** Aircraft that (are non-friendly) and ((have an unknown mission) or (have the potential to enter restricted airspace within 5 minutes)) shall raise an alert.

Only the customer can describe the intent, and only after the intent is known can any RLs be applied to carry out simplifications based on commutativity, associativity, and distributivity of logical operators.

4 Implications of Observations for Requirements Elicitation

The observations of Section 3 have an important implication on the process of requirements elicitation. Basically, a requirements analyst must search for all instances of and and or in a RS. She must examine each and if there is any chance that the use is not truth func-

tional, she must ask questions of the client to determine his intent for the instance. An instance that is not truth functional should be changed to another word or phrase that more accurately indicates the instance's intent. The goal of the analyst is that when she is finished examining the RS, each remaining instance of and or or is truth functional and can be subjected to RL-based rules. Specifically, for each phrase *A* and *B*, the analyst and customer must determine if it is one of the following:

- *B* follows *A* chronologically, in which case the and should be changed to and then,
- *B* follows *A* logically, in which case the and should be changed to and therefore,
- *B* is in contrast to *A*, in which case the *A* and *B* should be split into two sentences capturing *A* and *B*, and the *B* sentence should be changed to begin with However,,
- *B* is a surprise given *A*, in which case the *A* and *B* should be changed into *B* in spite of the fact that *A*., and
- *B* is dependent on *A*, in which case the and should be changed to and thus,.

If the phrase *A* and *B* is not one of these, then it can be safely assumed that the and is truth functional.

Equally specifically, for each phrase *A* or *B*, the analyst and customer must determine if *B* is an alternative to *A*, in which case the *A* or *B* should be split into two sentences capturing *A* and *B*, and the *B* sentence should be changed to begin with Alternatively, Otherwise, or If not, then. If the phrase *A* or *B* is not of this form, then it can be safely assumed that the or is truth functional. Then the analyst and customer must examine each remaining phrase *A* or *B* to classify its or as inclusive or exclusive. If

- *A* or *B* is inclusive, then *A* or *B* should be changed to *A* or *B* or both or *A*, *B*, or both,
- *A* or *B* is exclusive, then *A* or *B* should be changed to either *A* or *B*.

Finally, the analyst and customer must determine for any phrase involving more than one coordinator the precedences of each so that the scope of each coordinator is known. Then it will be possible to use RLs to work with the coordinators.

5 Guiding Rules and Future Work

Since RL-based rules will never be able to resolve coordination ambiguity, perhaps a better suggestion is to avoid introducing ambiguities during writing by using disambiguation guiding rules such as those we described previously [3]. A guiding rule is an instruction describing an ambiguous language use pattern with a suggestion for replacing that ambiguous language use with a less ambiguous way to say what is intended. For example, one guiding rule is:

> Avoid writing *S* containing X and/or Y. Instead, write X, Y, or both.,

where "S" is a variable standing in for "any sentence". One may use the guiding rules also to drive inspections for ambiguities in a RS. Guiding rules can be written to capture the transformations described in Section 4, and doing so is the subject of future work.

Space limitations preclude a full discussion of guiding rules and their use. Additionally, References 1 through 4 present all the guiding rules the first author has found, gives examples of the use of each, and discusses their strengths and weaknesses. Among the weaknesses is that no list of guiding rules can be complete, because new sources of ambiguity are discovered all the time. We do hope that the rate of discovery of new sources of ambiguity will eventually taper off. Another weakness of guiding rules is that applying them requires human judgement, because no rule is always applicable.

Therefore, the future work is to develop guiding rules to avoid coordination ambiguity and to continue to find other sources of ambiguity that can be avoided by guiding rules.

6 Conclusion

A reevaluation of language patterns shows that language patterns derived from RLs cannot truly resolve coordination ambiguity. Coordinators are not always used in truth-functional ways. The interpretation of the result of a transformation based on RLs sometimes differs from that of the original. Many a use of a coordinator serves to convey relationships other than conjunction or disjunction, such as temporal ordering and logical implication or exclusion. Such uses must be identified and made explicit before any RL-based patterns can be applied. Guiding rules are an alternative to language patterns and can be used to identify and make explicit the non-truth-functional uses of coordinators. Guiding rules can be used also to drive inspections.

References

[1] S. F. Tjong, N. Hallam, M. Hartley, "Improving the quality of natural language requirements specifications through natural language requirements patterns", In *IEEE International Conference on Computer and Information Technology*, Korea, 2006

[2] S. F. Tjong, *Elaborated natural language patterns for requirements specifications*. Technical Report, Faculty of Engineering & Computer Science, University of Nottingham, 2006, http://sepang.nottingham.edu.my/~kcx4sfj/TRII.pdf

[3] S. F. Tjong, M. Hartley, D. M. Berry, Extended Disambiguation Rules for Requirements Specifications, *10th Workshop on Requirements Engineering*, Toronto, Canada 2007
[4] S. F. Tjong, Avoiding Ambiguities in Requirements Specifications, PhD Dissertation, Faculty of Science, University of Nottingham, February 2008
[5] F. Chantree, A. Willis, A. Kilgarriff & A.D. Roeck, *Detecting dangerous coordination ambiguities using word distribution*, Recent Advances in Natural Language Processing, Springer, 2006
[6] F. Chantree, B. Nuseibeh, A.D. Roeck, A. Willis, *Nocuous ambiguities in requirements specifications*, Technical Report, Department of Computing, Faculty of Mathematics and Computing, The Open University, 2005
[7] A. Okumura & M. Kazunori, "Symmetric pattern matching analysis for English coordinate structure", *Proceedings of the 4th Conference on Applied Natural Language Processing*, Stuttgart, Germany, 41-46, 1994
[8] S. M. Cohen, *Introduction to Logic, Chapter 3: The Boolean Connectives*, University of Washington, 2004
[9] M. Haspelmath, *Coordinating constructions*, John Benjamins B.V., 2004
[10] R. Agarwal and L. Boggess, "A simple but useful approach to conjunct identification", In *Proceedings of ACL* 1992, pp. 15-21
[11] M. Goldberg, "An unsupervised model for statistically determining coordinate phrase attachment", In *Proceedings of ACL* 1999, pp. 610-614
[12] P. Resnik, "Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language", *Journal of Artificial Intelligence Research*, 1999, 11:95-130
[13] N. E. Fuchs and R. Schwitter, "Specifying logic programs in controlled natural language", In *CLMLP'95, workshop on Computational Logic for Natural Language Processing*, 1995
[14] U. Schwertel, "Controlling plural ambiguities in Attempto Controlled English", In *Proceedings of the Third International Workshop in Controlled Language Applications (CLAW)*, 2000, Seattle, WA, USA
[15] E. Kamsties, Surfacing Ambiguity in Natural Language Requirements, PhD Dissertation, Fachbereich Informatik, Universität Kaiserslautern, Kaiserslautern, Germany, also Volume 5 of PhD Theses in Experimental Software Engineering, Fraunhofer IRB Verlag, Stuttgart, Germany, 2001
[16] C. Denger. High Quality Requirements Specifications for Embedded Systems through Authoring Rules and Language Patterns, MSc Thesis, Fachbereich Informatik, Universität Kaiserslautern, Kaiserslautern, Germany, 2002
[17] W. van Rossum, "The Implementation of Techologies in Intensive Care Units: Ambiguity, Uncertainty and Organizational Reactions", Research Report 97B51, Research Institute SOM (Systems, Organisations and Management), University of Groningen, Groningen, the Netherlands, 1997
[18] S. W. Sussman and P. J. Guinan, "Antidotes for High Complexity and Ambiguity in Software Development", *Information and Management,* 1999, 36: 23-35
[19] D. C. Gause and J. M. Weinberg, *Exploring requirements: quality before design*, Dorset House, New York, NY, USA, 1989