

Developing an Algebra Sublanguage for Tutorial Dialogues

Van Uy Truong¹, Michael Glass² and Jung Hee Kim¹

¹North Carolina A&T State University, Greensboro, NC

²Valparaiso University, Valparaiso, IN

vutruong@ncat.edu, michael.glass@valpo.edu, jungkim@ncat.edu

Abstract

In this study we examine the specialized sublanguage for discussing, teaching, and solving algebra problems as found in transcripts of tutorial dialogues. In this sublanguage some verbs are constrained to apply to certain objects in the domain of discourse. The syntagmatic roles of the verb complements are typically algebraic, e.g. the complements to the verb *multiply* refer to operands and methods, while *simplify* refers to expressions. We use case frames to describe the verbs and their special sublanguage complements. From case frames we create grammar and lexicon for the Link Grammar Parser, enabling the parsing of dialogues containing mixed ordinary English and algebra tutorial English.

Introduction

When teachers and students converse about algebra they employ a specialized mathematical sublanguage that is not ordinary English. We are studying this algebra tutorial sublanguage with the goal of constructing language understanding facilities for the Wooz-2 intelligent tutoring assistant [Glass et al. 2007]. From a corpus of 48 algebra transcripts collected by previous work [Chae et al. 2005], we identified usages that were specific for this sublanguage. We created case frames to describe the sublanguage verbs and their complements in a formal manner. Finally we have been converting the case frames to Link Grammar rules for the Link Parser [Sleator and Temperley 1995].

Table 1 contains sentences from typical algebra tutoring transcripts that illustrate different ways that the concept of multiplying two numbers can be expressed in dialogue, using the words *multiply* and *times* and the operator “*”. In sentence 1) the complements to the verb *multiply* each refer to an operand to be multiplied. But sentence 2) contains a

- | | |
|----|---|
| 1) | Then multiply the 6 times 4. |
| 2) | Multiply exponents. |
| 3) | You can just multiply each term including $3/2$ by the LCD. |
| 4) | 12 times 5 is 60. |
| 5) | $12n=12*10$ and I am not sure but I think you should add 5 to $12*10$. |

Table 1: Examples of Language for Multiplication

single plural complement *exponents* that refers to an undetermined number of operands which will be multiplied together. The plural complements to the verb *multiply* in sentences 2) and 3) do not play the same algebraic role, since the *exponents* of 2) are to be all multiplied together while *each term including $3/2$* from 3) will not be all multiplied together. Words such as *multiply*, *times*, *add* and *is* can be used in place of (and sometimes mixed with) “*”, “+”, and “=” symbols, as sentences 4 and 5 hint. But the word versions of mathematical operations are not completely interchangeable with the symbols, as we have no attested examples of sentences like ** the 6 by 4*.

In this work we describe this sublanguage using the formalism of case frames, where each frame identifies one word that has specialized usage (usually a verb) along with its complements. Formally describing this aspect of the sublanguage is needed in order to exploit computer language understanding capabilities.

Background

The Differential Equations tutorial dialogue project [Glass et al. 2007] is developing an extension of the Wooz software for computer-mediated mathematics tutoring [Kim and Glass 2004]. Wooz provides web-delivered communication tools—a chat window, an equation editing window with mathematics typing widgets, equation visualization applets, a library of problems—so that a student and a tutor can engage in problem-solving tutorial dialogues from separate locations. For some algebra problems, Wooz also provides to the tutor a browsable structured set of tutorial goals. For example a goal might be “have the student check the sign of the coefficients.” Along with each goal there are examples of suitable tutorial language for achieving those goals.

These goals and associated language, which include a number of hints and tutoring tactics, were obtained by observing expert tutors at work. There are forty eight transcripts of algebra tutoring, with around one hour of typed (keyboard-to-keyboard) tutoring each. We are also developing a corpus of differential equation transcripts.

While the Wooz set of tutorial goals is passively displayed, they are merely available so the tutor can browse them and insert sentences into the dialogue. Wooz-2 will endeavor to actively follow the tutorial conversation in progress. The Wooz-2 tutor will not be a fully automated

Case Roles in Syntactic Order	Example Sentence
Objective	The door opened.
Agent, Objective	John opened the door.
Instrument, Objective	This key opens the door.
Objective, Instrument	The door opened with a key.
Agent, Objective, Instrument	John opened the door with a key.

Table 2: Case Frames for Verb *open* [Gibbon 1997].

intelligent tutoring system. It is in the service of having the computer follow and understand the tutorial conversation that we analyze the algebra sublanguage.

A sublanguage is specialized form of a natural language that is used within a particular domain or subject matter, in particular “sets of sentences whose lexical and grammatical restrictions reflect the restricted sets of objects and relations found in a given domain of discourse” [Kittredge and Lehrberger 1982, p. 2]. Sublanguages may or may not be strict subsets of English, both the semantics of the domain and linguistic conventions of the sublanguage control what is acceptable. For example, in biochemistry it is not acceptable to say *hydrochloric acid was washed in polypeptides* but it is permissible for polypeptides to be washed in hydrochloric acid [Harris and Mattick 1988]. In our algebra sublanguage not every noun phrase can be a complement of *multiply*. It is OK to *multiply binomials* and *multiply exponents*. But although it is admissible in normal English to *multiply problems* and for *problems to multiply*, these are not admissible in the algebra sublanguage.

Since domain-dependent knowledge is embedded within the grammar and selectional restrictions of a sublanguage, writing systems to precisely extract relevant domain information and skip over less relevant text can be enhanced by first analyzing the sublanguage. This was the approach used for processing queries for government statistical information [Liddy and Liddy 2001] and extracting information from trouble tickets [Liddy et al. 2006].

Today, some successful sublanguages are pharmacology reports, weather reports, technical manuals, cooking recipes, patents, stock market reports, patient medical histories, legal documents, university catalogs, journal abstracts, life insurance applications, and web pages [Liddy et al. 2006].

Case grammar [Fillmore 1968] combines semantic roles of verbs with syntactic position. The case frame for a verb describes the collection of roles that must be present as complements. For example, case frames for a verb *open* are illustrated in Table 2. Without committing to particular theories of lexical semantics, we can use the formalism of case frames for describing the semantically-derived restricted verb complements of sublanguages.

The Link Grammar Parser (LGP) [Sleator and Temperley 1995; Lafferty et al. 2009] is convenient in that it uses a grammar formalism that is similar to case frames. Thus it is suitable for parsing our sublanguage text that has been analyzed according to case frames. Its lexicon contains “links”

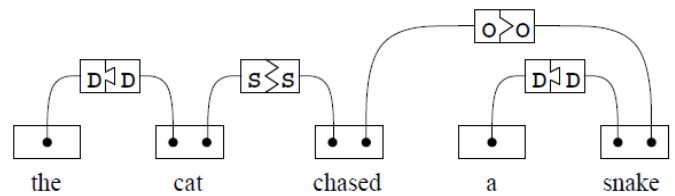


Figure 1: Simplified Link Grammar Parse

for most words that identify the role relationships and the relative syntactic positions of words that are joined in the same surface constituent, as illustrated in Figure 1. In this very simplified example the verb *chased* specifies an S (subject) link to the left and an O (object) link to the right. The head nouns from both these complements join up with the verb through the S and O links. Each head noun and determiner are further joined by D (determiner) links. In the Link formalism, crossed links—representing, roughly speaking, fragmented non-contiguous constituents—are signs of ungrammaticality.

Notice that the Link grammar has no representation for deep structure constituents. It represents surface structure in its formalism. Case frame theory explicitly evolved from transformational grammar, where the several roles of a case frame were posited to be deep syntactic constituents. Because deep structure is transformed into surface structure, the roles in a case frame can be expressed as number of surface orderings of constituents. In our work we write case frames that represent surface orderings only, in order to create Link Parser lexical entries that have the surface structure links.

The Link parser is mature. It comes with a dictionary that has up to 60,000 word forms. It performs well especially in its ability to guess unknown words. The Link Parser is adequately fast to parse typed dialogues in real-time.

Sublanguage Analysis using Case Frames

A typical short segment of tutorial dialogue that uses *multiply* in several ways is shown in Figure 2. This dialogue fragment illustrates a very common tutorial goal, making sure the student knows what operation will be performed. Generally the tutor achieves this goal by eliciting the name of the operation from the student.

In these dialogues an equation to be solved, along with more equations introduced while solving the problem, are frequently displayed in a separate window. They are visible all the time. Many of the referring expressions are referring to parts of these equations. Thus when the teacher asks “what operation do you need to apply here?” the student is looking at two binomials side-by-side in the equation window, waiting to be multiplied together.

From the dialogues, we manually identified a number of verbs and operations that serve specialized functions in the algebra sublanguage, shown in Table 3. They all receive special case frames.

By way of illustration we have created case frames for the verb *multiply* as illustrated in Table 4.

Tu: James, What operation do you need to apply here??
 St: Factoring.
 Tu: No, James, not factoring, the expression is already factored. What is the opposite of factoring?
 St: Distributing.
 Tu: You can say distributing. But we are multiplying here. Now, what process do we use when we are multiplying two binomials.
 St: Foil.
 Tu: Excellent. You use FOIL. What does FOIL stand for?
 St: First Outer Inner Last.
 Tu: Good. Now go ahead and use FOIL to multiply the two binomials together.

Figure 2: Dialogue From Transcript T1.

multiply	divide	add	subtract	times
cancel	change	check	combine	compare
determine	distribute	factor	find	plug
raise	reduce	square root	simplify	square
substitute	solve	* + - / ^	=	

Table 3: Verbs in the Algebra Sublanguage

Because the case frames are created for our special domain, the syntagmatic roles of the verb complements are not the traditional ones of “agent,” “object,” “beneficiary,” and so on. Instead we have roles in algebraic manipulations, such as “op” (for operand), “process” (method of solution), and “bottom op” (the bottom of a fraction being operated on). A verb such as *solve* can have complements such as problem to be solved and equation to be solved.

Case 1 of *multiply* represents a common imperative, where the verb has no complements in the surface structure. The objects to be multiplied are present in the dialogue context. Case 2 combines several items with a single plural noun phrase that will be all multiplied together. Case 3 names two operands for a dyadic multiplication with two singular noun phrases. Case 4 specifies several dyadic multiplications, using a plural noun phrase and a singular noun phrase. In case 5, the process for multiplying is specified. The “process” role is similar to to Fillmore’s “instrument” role (the “key” in the Table 2 example). Case 6 specifies the top and bottom of a fraction. It is similar to a plural noun phrase, but this semantic case occurs frequently, using this syntactically frozen form, so we think operating on the top and bottom of a fraction is likely a special construction in the algebra sublanguage.

Sublanguage Parsing with Link Parser

Figure 3 shows Link Parser structures generated from parsing two sample sentences in the transcript.

LGP does a good job parsing a majority of the ordinary text in our transcripts, Figure 3a) shows an example of LGP parsing *what is the operation here?* This sentence appears to be ordinary English. Although it occurs frequently in our dialogues, and we have classified it as part of our sublan-

1	(No complements) 1. But we are multiplying here 2. Go ahead and multiply now.
2	plural-op 1. Now, what process do we use when we are multiplying two binomials. 2. Now do the multiplication for each numerator 3. Multiply exponents
3	op, by-op 1. Then multiply the 6 times 4. 2. Multiply 5/2 by 1/2.
4	plural-op, by-op 1. Yes, so I multiply everything by the reciprocal of the first one right? 2. You can just multiply each term including 3/2 by the LCD then distribute (3/2 * LCD).
5	process, plural-op 1. Now go ahead and use FOIL to multiply the two binomials together.
6	top-op, bottom-op, by-op 1. Okay, multiply top and bottom by (x - 4). 2. multiply top and bottom by that factor.

Table 4: Case Frames for *multiply*

guage because *operation* is a special concept in the algebra domain, there are no violations the norms of ordinary English and there are no special roles that we need to extract. From LGP, parts of speech have been tagged correctly and punctuation marks have been assigned reasonable roles. The main verb *is* has been identified and properly joined to its two complements *what* and *operation*. The sentence has been properly identified as a question.

Figure 3b) on the other hand, shows a failed parse when trying to parse *so what is y - 5ly?*, a sentence that violates normal English rules. The variable *y* is ignored and the hyphen is misidentified as a colon or semi-colon rather than an arithmetic operator. The expression *5ly* is guessed by the system to be a noun, perhaps a reasonable guess for English purposes but wrong for discussing algebra where *y* and *5ly* should be tagged as operands.

By contrast, Figure 3c) illustrates our enhancements to the LGP grammar. It shows a useful parse of a sentence within

our algebra sublanguage: *so I multiply each and every term by 18?* The constituents within the sentence are correctly identified and linked. The ALG link from the LEFT-WALL to the main verb *multiply* identifies the sentence as being in the algebra sublanguage. The lexical entry for the word *multiply* calls for this link. It also calls for links to the various verb complements as specified by the applicable case frame. The case frame that matches this sentence is: “multiply” + plural-op + by-op, in that order. Thus we have created a COP2 link to join the main verb to the plural-op, while the OO link joins in the by-op. Slightly less useful are the ambiguous parses of the plural NP (both shown): one identifies *each term* and one identifies *every term* as the plural operand.

Thus we see that case frames written according to the surface structure of the sentence can readily be mapped into link grammar rules.

Equations Embedded in Language

However the case frames are not enough, we also need to parse embedded equations of the sort in Figure 3b). When we enhance the Link Parser Grammar for this purpose we define a set of equation-embedded constituents (rather like a new part of speech) represented by `<equation>`.

Several types information are needed for `<equation>` constituents: we define new link labels, we define rules guiding the construction of these links, and we define new lexical entries where the words require the new links. The new link names created for `<equation>` are as follows. Note that “Case” represents the head of the phrase.

Word Pair	Link
Case to <code><equation></code>	CEQ
<code><equation></code> to <code><process></code>	EQP
<code><equation></code> to <code><for op></code>	EQF
<code><equation></code> to Case	EQC
<code><plural op></code> to <code><equation></code>	POE
<code><op></code> + <code><equation></code>	OEQ

Some of the rules for link construction are:

1. Case + `<equation>`
This is a CEQ link to the left
2. `<equation>` + Case
This is an EQC link to the left
3. Case + `<plural op>` + `<equation>`
This is a POE link to the left
4. Case + `<for op>` a `<equation>`
A CQF link to the right
5. Case + `<equation>` + `<process>`
A CEQ link to the left and EQP to the right

Finally we make a syntactic pattern for `<equation>` class words as follows.

```
<equation>:
  POE- or CEQ- or (CQF- & EQP+) or
  (CEQ- & EQF+) or EQC-;
```

In this notation, POE- means that `<equation>` class words can form a POE link to the left. The clause (CEQ- & EQP+) means a CEQ link to the left and an EQP link to the right.

There are other constraints applied which are not shown here, such as a possible link to LEFT-WALL that is the leftmost constituent of every sentence.

Conclusions

The formalism of case frames gives us a way to describe the algebra sublanguage domain-dependent verb and equation usages attested in tutorial dialogue. The case frame description of a verb’s specialized syntax can be translated into instructions for the Link Grammar Parser with relative ease. We can also write link grammar rules for embedded equations. In the future, since case-frame derived grammar rules assign case roles to constituents of the sentence, the case frames will aid in producing semantic representations of the utterances.

Acknowledgment

This work is supported by the REESE program of the National Science Foundation, under awards 0634049 to Valparaiso University and 0633953 to North Carolina A&T State University. The content does not reflect the position or policy of the government and no official endorsement should be inferred.

References

- Hyeun Me Chae, Jung Hee Kim, and Michael Glass. Effective behaviors in a comparison between novice and expert algebra tutors. In *Proceedings of the Sixteenth Midwest AI and Cognitive Science Conference*, pages 25–30, 2005.
- Charles Fillmore. The case for case. In E. Bach and R. T. Harms, editors, *Universals in Linguistic Theory*, pages 1–88. Holt, Reinhart, and Winston, 1968.
- Dafydd Gibbon. *Semantics: The meaning of verbs*, 1997. URL <http://coral.lili.uni-bielefeld.de/Classes/Summer97/VerbSem/vsemss97/vsemss97.html>. Notes for a summer 1997 class at the University of Bielefeld.
- Michael Glass, Jung Hee Kim, Karen Allen, and Kathy Cousins-Cooper. Towards Wooz-2: Supporting tutorial dialogue for conceptual understanding of differential equations. In *Proceedings of the Eighteenth Midwest AI and Cognitive Science Conference*, pages 105–110, 2007.
- Zellig Harris and Paul Mattick. Science sublanguages and the prospects for a global language of science. *Annals of the American Academy of Political and Social Science*, 495:73–83, 1988.
- Jung Hee Kim and Michael Glass. Evaluating dialogue schemata with the wizard of oz computer-assisted algebra tutor. In James C. Lester, Rosa Maria Vicari, and Fábio Paraguaçu, editors, *Intelligent Tutoring Systems, 7th International Conference, Maceió, Brazil*, volume 3220 of *Lecture Notes in Computer Science*, pages 358–367. Springer, 2004.

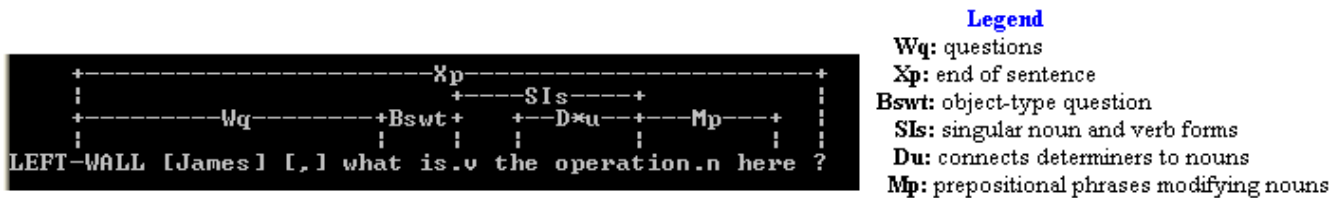
Richard Kittredge and John Lehrberger, editors. *Sublanguage: Studies of language in restricted semantic domains*. Walter de Gruyter, 1982.

John Lafferty, Daniel Sleator, and Davy Temperley. Link grammar web page, 2009. URL <http://www.link.cs.cmu.edu/link/>.

Elizabeth D. Liddy and Jennifer H. Liddy. An NLP approach for improving access to statistical information for the masses. In *Proceedings of the Federal Committee on Statistical Methodology 2001 Research Conference*, 2001.

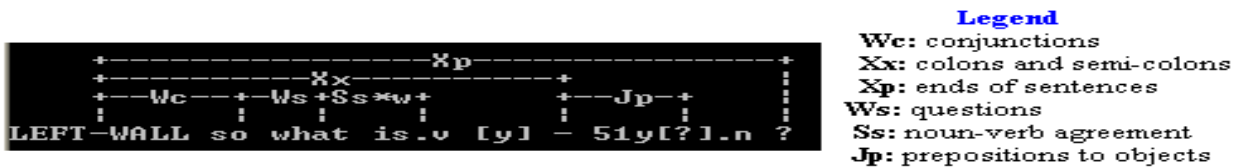
Elizabeth D. Liddy, Svetlana Symonenko, and Steven Rowe. Sublanguage Analysis Applied to Trouble Tickets. In *19th International FLAIRS Conference*, 2006.

Daniel Dominic Sleator and David Temperley. Parsing English with a link grammar. *CoRR (Computing Research Repository)*, abs/cmp-lg/9508004, 1995.



Legend
Wq: questions
Xp: end of sentence
Bswt: object-type question
SIs: singular noun and verb forms
Du: connects determiners to nouns
Mp: prepositional phrases modifying nouns

a) Parse of “James, what is the operation here?” using regular Link grammar



Legend
Wc: conjunctions
Xx: colons and semi-colons
Xp: ends of sentences
Ws: questions
Ss: noun-verb agreement
Jp: prepositions to objects

b) Parse of “so what is y-51y?” using regular Link grammar



Linkage 1
Legend
ALG: left-wall to a case word
Xp: end of sentence
COP2: link a case or operation to plural operand
OO: Linking a plural operand to a single operand
IDQ and **IDJ**: multiword or compound word

Linkage 2

c) Two proposed parses of “So I multiply each and every term by 18?” using enhanced grammar.

Figure 3: Link Grammar Parser Results