Chapter 11

Generating English paraphrases from logic

Dan Flickinger
Stanford University

A set of semantic transfer-based paraphrase rules is developed and applied using an existing broad-coverage grammar and efficient generator to implement the automatic production of a wide variety of English sentences from input propositions in quantifier-free first-order logic within the "blocks language" of Tarski's World, a component of the course Language, Proof and Logic.

1 Introduction

A student learning to express propositions in formal logic is typically presented with practice exercises where a proposition is expressed in a natural language such as English, and the task is to construct the corresponding logical form. In an automated instructional system, it would be desirable to be able to construct a new target logical form on demand, and automatically generate from this expression a natural language sentence to present to the student as a new exercise. Similarly, it would be helpful to be able to take the student's incorrect attempt at a solution, and generate a natural language sentence to show the student what the proposed solution actually says, as one way of prompting repair. Since the mapping between natural language and logic can vary in transparency even for relatively simple logical systems, it would be convenient to generate, in fact, a diverse set of annotated paraphrases so that the instructional software could select a paraphrase appropriate for the student's current level of proficiency.

One widely used instructional software system for teaching first-order logic (FOL) is provided as part of the textbook *Language, proof and logic* by Barker-Plummer, Barwise & Etchemendy (2011). One component of this software is called Tarski's World, which defines a "blocks language" used for many of the exercises in the book, with fewer than twenty predicates, varying in arity from one to three arguments, along with boolean and conditional connectives, and quantifiers. Some of these exercises present the student with an English language proposition such as "**b** is a cube" to be

translated into this blocks language, where the software system can automatically evaluate the correctness of the student's submitted solution. For the first stage of developing an English generator from FOL propositions in Tarski's World, quantifiers have been excluded, but otherwise the full range of expressive variation illustrated in the textbook is included.[1]

The objective for the present study was to produce an accurate and robust implementation of an English generator which can take as input any well-formed FOL expression (excluding quantifiers) within the blocks language of Tarski's World, and produce a rich set of English paraphrases each of which translates back to the input FOL, and ideally only to that FOL. The next section describes the grammar-based method employed in this implementation, including descriptions of the existing resources adapted for this task. Section 3 presents the set of paraphrase rules and the declarative formalism used to define them, along with examples of output from the generator. The final two sections include a discussion of how the generator's output has been evaluated to date, and provide some larger context for this effort in related work on generation from logic or other formal languages.

2 Method

In the approach adopted here, the generation system accepts an input proposition in FOL, converts it to the grammar-specific semantic representation, and then applies a set of paraphrase rules to produce alternate semantics, each of which is then presented to the generator itself to produce a set of English sentences which realize that semantic representation. Each output sentence carries an annotation identifying which if any of the paraphrase rules were applied to produce its semantics, so the instructional system can make a more informed choice about which of the alternative English outputs to present to the student based on prooperties of the exercise or of the student's proficiency.

2.1 Component resources

The FOL-to-English implementation draws on two substantial pre-existing resources developed within the DELPH-IN consortium (`www.delph-in.net`): the ACE parser and generator (`moin.delph-in.net/AceTop`) developed by Woodley Packard, and the English Resource Grammar (ERG: Flickinger 2000; 2011), a linguistically rich broad-coverage grammar implementation within the framework of Head-driven Phrase Structure Grammar (HPSG: Pollard & Sag 1994). The ACE engine is a more efficient re-implementation of the chart parser and generator of the LKB (Copestake 2002; Carroll et al. 1999), applying the rules of a grammar such as the ERG either to an input sentence in order to output semantics (parsing), or to an input semantic representation in order to output sentences (generating). Both of these general-purpose

[1] The "blocks language" predicates are *Cube, Tet, Dodec, Small, Medium, Large, Smaller, Larger, LeftOf, RightOf, BackOf, FrontOf, SameSize, SameShape, SameRow, SameCol, Adjoins, =, Between.*

resources were used essentially unchanged for this task, apart from minor lexical additions for the Tarski's World domain.

2.2 FOL to MRS

The semantic framework adopted within the ERG is called Minimal Recursion Semantics (MRS: Copestake et al. 2005), so an input FOL proposition for this task is first automatically converted to a 'skeletal' MRS by a combination of a simple reformatting utility[2] and then a small set of declarative MRS-to-MRS rules normalize to grammar-internal predicate names, and assign default values for tense/aspect/mood on verbal predications, and person/number plus definiteness for nominal predications. This fully-specified MRS can be given as input to the generator, which will produce one or more English sentences which realize the MRS. Here is a simple example of these three basic steps, before we turn to paraphrases:

FOL: `Large(a)&Large(b)`

Skeletal MRS:

```
LTOP: h1
INDEX: e1
RELS: < [ "name" LBL: h3 ARG0: x1 CARG: "A" ]
        [ "large" LBL: h4 ARG0: e2 ARG1: x1 ]
        [ "name" LBL: h5 ARG0: x2 CARG: "B" ]
        [ "large" LBL: h6 ARG0: e3 ARG1: x2 ]
        [ "and" LBL: h2 ARG0: e1 L-INDEX: e2 R-INDEX: e3 ] >
```

Full MRS:

```
LTOP: h20
INDEX: e13 [SORT: collective SF: prop TENSE: pres PERF: -]
RELS: <
 [ named LBL: h5 ARG0: x10 [PERS: 3 NUM: sg] CARG: "A" ]
 [ named LBL: h9 ARG0: x11 [PERS: 3 NUM: sg] CARG: "B" ]
 [ proper_q LBL: h2 ARG0: x10 RSTR: h3 BODY: h4 ]
 [ proper_q LBL: h6 ARG0: x11 RSTR: h7 BODY: h8 ]
 [ _large_a_1 LBL: h18 ARG0: e14 [SF: prop TENSE: pres PERF: -]
                       ARG1: x10 ]
 [ _large_a_1 LBL: h19 ARG0: e15 [SF: prop TENSE: pres PERF: -]
                       ARG1: x11 ]
 [ _and_c LBL: h12 ARG0: e13 L-INDEX: e14 R-INDEX: e15
                             L-HNDL: h16 R-HNDL: h17 ] >
HCONS: < h3 qeq h5 h7 qeq h9 h16 qeq h18 h17 qeq h19 >
```

[2] Thanks to Aaron Kalb for the Python script.

English realizations from the generator:

a *is large and* ***b*** *is large.*
a *is large, and* ***b*** *is large.*

In this example, the constants ***a*** and ***b*** are mapped to "name" elementary predications (EPs) in the 'skeletal' MRS; the one-place predicate `Large` is mapped to a one-place EP (plus the inherent ARG0 event variable); and the conjunction symbol is mapped to a two-place EP with its arguments the ARG0 values of the two "large" EPs. The full MRS in this simple example still contains in its RELS list these five EPs normalized with grammar-internal predicate names, with default values filled in for tense, number, etc., and with quantifier EPs added to bind each of the two "named" EP ARG0 instance variables, to satisfy general constraints on MRS well-formedness.[3]

2.3 MRS to MRS for paraphrases

In order to produce non-trivial paraphrases of the sentences that correspond to this 'literal' mapping from FOL to MRS, a set of MRS-to-MRS mapping rules have been developed, and some or all of these can be applied to the original MRS to produce a (potentially large) set of alternate MRSs, each of which can be given as input to the generator for realization into English sentences. These MRS-to-MRS rules employ a rule specification formalism originally developed by Stephan Oepen for machine translation within the LOGON research project (Lønning et al. 2004), where the formalism allows each 'translation' rule to specify one (often partial) MRS expression as *input* and another MRS expression as *output*, along with optional positive and negative constraints on other elements within the full MRS being 'translated'. For the present task, the rule set serves to map one English MRS to one or more output English MRSs that may be realized by the generator as paraphrases of sentences realized for the input MRS.

To continue with the above example, the paraphrase rules include ones for mapping a conjunction of two clauses with a common subpart into a single clause with either a conjoined noun phrase or a conjoined verb phrase. Thus the original MRS realized as ***a*** *is large and* ***b*** *is large* can be mapped via these rules into an MRS which is realized as follows:

a *and* ***b*** *are large.*
Both ***a*** *and* ***b*** *are large.*

Similarly, the rule set includes mappings that can reverse the order of the conjuncts, to produce MRSs that will be realized as follows:

b *is large and* ***a*** *is large.*
b *is large, and* ***a*** *is large.*

[3] The full MRS includes, in addition to the outermost label LTOP and event variable INDEX, a set of scope constraints in HCONS which are not relevant for this discussion, but which will be essential as this work is extended to accommodate FOL expressions with quantifiers.

b *and* ***a*** *are large.*
Both ***b*** *and* ***a*** *are large.*

Examples of the paraphrase rules themselves are presented in the next section.

3 Paraphrase rules

Inspection of the exercise example sentences in the LPL textbook reveals a number of sources of linguistic variation in the English expression of propositions within the blocks language for Tarski's World, including the following:

- coordination/aggregation;
  - nominal phrases (***b*** *is in front of a cube and a tetrahedron*);
  - predicates (***b*** *is a cube and is large*);
- negation (*It is not the case that* ***a*** *and* ***b*** *are large*);
- pronouns (*If* ***b*** *is a cube, it is large*);
- partitives (***a*** *and* ***b*** *are large, and both of them are cubes*);
- VP ellipsis (*If* ***b*** *is large, then* ***c*** *is*);
- sentence connectives (*if and only if, just in case, unless*);
- adjectives as pre-modifiers (***b*** *is a large cube*);
- adverb addition (*If* ***a*** *is large,* ***b*** *is also large*);
- reordering (***a*** *and* ***b*** *are cubes* – ***b*** *and* ***a*** *are cubes*).

Implementation of the MRS-to-MRS mapping to enable these variations resulted in a set of 143 paraphrase rules,[4] defined as a hierarchy of types within the LOGON-based formalism for semantic transfer, which is supported by the ACE engine.

An example of one of these paraphrase rule types accommodates the generation of sentences with verb-phrase ellipsis, as in the following alternation:

b *is large and* ***c*** *is large*
b *is large and* ***c*** *is*

The following rule applies to an input MRS where two event EPs have the same predicate name (the value of the PRED attribute in an EP), identifying in the CONTEXT attribute the first EP which will survive the rule's application unchanged, and in the INPUT attribute the second EP (and possibly additional EPs for that second verb phrase) which will be replaced by the ellipsis EP in the OUTPUT attribute.

[4] These rule definitions are included in the most recent version of the open-source ERG, within the subdirectory 'openproof', available at `www.delph-in.net/erg`.

```
basic_vp_ellipsis_gpr := monotonic_mtr &
[ CONTEXT [ RELS  < [ PRED #pred, ARG0 event ] > ],
  INPUT   [ RELS  < [ PRED #pred, LBL #h1,
                      ARG0 #e2 & event,
                      ARG1 #x3 & ref-ind ], ... > ],
  OUTPUT  [ RELS  < [ PRED ellipsis_ref, LBL #h1,
                      ARG0 #e2,
                      ARG1 #x3 ] > ] ].
```

This output ellipsis EP preserves the inherent argument (ARG0) and the external (subject) argument (ARG1) of the deleted EP, to ensure the correct tense and agreement for the form of the verb *be* which will realize this ellipsis EP in the sentences generated from the resulting MRS.

The set of paraphrase rules is partially ordered to ensure that certain desired feeding relationships among the rules are enabled and that unwanted ones are prevented. If no restrictions on rule applicability are imposed when the paraphase generator is invoked for an MRS, all of the rules are applied exhaustively and iteratively until no further application of any rule is possible, resulting in an often large set of derived MRSs. Each of these can be presented as input to the English generator, which can realize one or all of the sentences licensed by the ERG for that MRS. Invocation of the paraphrase engine can include positive or negative requirements on which paraphrase rules to apply, for example requiring only coordination variants that preserve the original order of the conjuncts, or excluding variants with pronouns or ellipsis.

4 Evaluation and discussion

The initial benchmark for this implementation was to provide sufficient MRS-to-MRS paraphrase rules in order to successfully generate the full range of sentence variants observed in the LPL textbook, exhibiting rich combinations of the linguistic phenomena described above. The present rule set accomplishes this task for a set of 77 FOL expressions drawn from the book, including the following example along with a few of the English paraphrases generated by the system:

`Larger(`**`a`**`,`**`c`**`)&Larger(`**`e`**`,`**`c`**`)&¬Large(`**`a`**`)&¬Large(`**`e`**`)`

a *and* ***e*** *are both larger than* ***c****, and neither of them is large.*
a *is larger than* ***c*** *and* ***e*** *is larger than* ***c****, but neither of them is large.*
e *is larger than* ***c*** *and* ***a*** *is larger than* ***c****; moreover,* ***e*** *isn't large, and it's not the case that* ***a*** *is large.*

But missing from the current 4,448 distinct sentences generated by the system for this one FOL is the following, which should be expected from seeing the first two outputs above.

a *and* ***e*** *are both larger than* ***c****, but neither of them is large.*

This illustrates the need for further adjustments either to the partial order imposed on the existing rule set, or tuning of the input conditions for one or more of the rules, to enable the substition of *but* for *and* where the first conjunct is a sentence with a conjoined subject, and the second conjunct contains a partitive subject noun phrase with a pronoun anchored to that conjoined subject.

A more interesting evaluation of this paraphrase generator would involve a live study with students in the LPL course, to see if either on-the-fly generation of new exercises or rephrasing in English of incorrect answers could be shown to correlate with improved learning outcomes. A carefully controlled study of this type would be non-trivial to design and to carry out, but the generator in its current state should provide the functionality required for the FOL-to-English component of such a study.

5 Related work and next steps

The study of generation of natural language from logic dates back at least to the rule-based method of Wang (1980), which focused on problems involving quantifiers, unlike the present study, but developed a similarly decomposed approach to the translation task. Another rule-based approach is the semantic-head-driven method of Shieber et al. (1990), which is concerned primarily with the generation algorithm itself, and with the treatment of quantification, defining grammar-specific rules both for semantics-to-semantics mappings, and for semantics to surface forms; again the issue of paraphrasing is not addressed beyond variations due to lexical choice. More recent work such as that of Lu & Ng (1990) explores the use of statistical methods to generate English sentences from expressions in typed lambda calculus, including both log-linear and generative models. Both of these latter approaches also included application to multiple languages, which should also be possible with the present method, given an MRS-based grammar of another language, but no work has as yet been done in this direction. Another more recent approach by Kutlak & van Deemter (2015) enables improved generation in English output by defining simplification rules at the logic-to-logic level, rather than at the grammar-specific semantics level, but again no attention is given to paraphrasing.

An approach more closely aligned to the present one is the relatively early work of de Roeck & Lowden (1986), which identifies the important and vexing issue of how to minimize ambiguity in the natural language output of an automated generation system. This remains a challenge for the present approach, as can be seen by the following paraphrase currently and unfortunately generated by the system from an FOL with negation:

¬Large(**a**)&¬Large(**b**)
*It is not the case that **a** is large and **b** isn't large.*

This sentence makes a locally reasonable substitution for the negation predicate in the first conjunct clause, but creates an unwanted ambiguity where the second conjunct could be interpreted as within the scope of *It is not the case that...* Additional

unwanted ambiguity emerges at many other points in the mapping space encompassed by the rules defined for the present system, including confusable antecedents of pronouns and of elided verb phrases. Clearly such ambiguity needs to be minimized, and has been reduced to a large degree by careful manual adjustment to input conditions for the rules, but the development of a more systematic method of detecting and avoiding such unwanted ambiguity remains for future research.

Acknowledgments

I am grateful to David Barker-Plummer and his student Aaron Kalb of the Openproof project at Stanford's Center for the Study of Language and Information for presenting me with this challenge, and for their constructive critique during the development of the paraphrase generator. I am also grateful to the project for funding which supported the work. On a grander scale, I am indebted to John Nerbonne for illuminating the rich complexity of the mapping between natural language and logic during the years that we worked together in the Natural Language Project at Hewlett-Packard Laboratories.

References

Barker-Plummer, David, Jon Barwise & John Etchemendy. 2011. *Language, proof and logic.* 2nd edn. Stanford, CA: CSLI Publications.

Carroll, John, Ann Copestake, Daniel Flickinger & Victor Poznanski. 1999. An efficient chart generator for (semi-)lexicalist grammars. In *Proceedings of the 7th European Workshop on Natural Language Generation*, 86–95. Toulouse, France.

Copestake, Ann. 2002. *Implementing typed feature structure grammars.* Stanford, CA: CSLI Publications.

Copestake, Ann, Dan Flickinger, Carl J. Pollard & Ivan A. Sag. 2005. Minimal recursion semantics: an introduction. *Research on Language and Computation* 3(4).

de Roeck, A. N. & B. G. T. Lowden. 1986. Generating english paraphrases from formal relational calculus expressions. *Proceedings of the 11th Conference on Computational Linguistics, COLING 1986.* 581–583.

Flickinger, Dan. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering* 6 (1) (Special Issue on Efficient Processing with HPSG). Dan Flickinger, Stephan Oepen, J. Tsujii & Hans Uszkoreit (eds.). 15–28.

Flickinger, Dan. 2011. Accuracy vs. robustness in grammar engineering. In Emily M. Bender & Jennifer E. Arnold (eds.), *Language from a cognitive perspective: grammar, usage, and processing*, 31–50. Stanford: CSLI Publications.

Kutlak, Roman & Kees van Deemter. 2015. Generating succinct English text from FOL formulae. *Proceedings of the First Scottish Workshop on Data-to-Text Generation.*

Lønning, Jan Tore, Stephan Oepen, Dorothee Beermann, Lars Hellan, John Carroll, Helge Dyvik, Dan Flickinger, Janne Bondi Johannessen, Paul Meurer, Torbjørn Nordgård, Victoria Rosén & Erik Velldal. 2004. LOGON. A Norwegian MT effort. In *Proceedings of the Workshop in Recent Advances in Scandinavian Machine Translation*. Uppsala, Sweden.

Lu, Wei & Hwee Tou Ng. 1990. A probabilistic forest-to-string model for language generation from typed lambda calculus expressions. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing* 16 (1). 305–42.

Pollard, Carl & Ivan A. Sag. 1994. *Head-driven phrase structure grammar* (Studies in Contemporary Linguistics). Chicago, IL & Stanford, CA: The University of Chicago Press & CSLI Publications.

Shieber, Stuart, Gertjan van Noord, Fernando C. N. Pereira & Robert C. Moore. 1990. Semantic-head-driven generation. *Computational Linguistics* 16 (1). 305–42.

Wang, Juen-tin. 1980. On computational sentence generation from logical form. *Proceedings of the 8th Conference on Computational Linguistics, COLING 1980*. 405–411.