Translating a detailed linguistic semantic representation into Horn clause logic

Jean-François Delannoy [1] **Riverson Rios** [1,2]

[1] Department of Computer Science
University of Ottawa
Ottawa, Ontario
K1N 6N5 Canada
tel: +1 (613) 564-5427 fax: +1 (613) 564-9486
e-mail: delannoy@csi.uottawa.ca, riverson@csi.uottawa.ca

[2] on leave from Universidade Federal do Ceará, Brazil

Abstract

From the semantic representation produced by TANKA, a domain-independent, knowledge-scant, and interactive natural language analyzer, we construct Horn clauses which represent the facts or rules found in technical texts, specifically legal texts. This article describes the conversion process and the way it takes into account the detailed structure returned by the analyzer, a precise syntactic-semantic description of the sentence, and in particular at the three levels of noun phrase semantics, case structures and inter-clausal relations.

1. Introduction

Knowledge acquisition from text is an expanding domain of study (Banff 94, Britton&Black 85, Delannoy et al 93, ICCS93, MUC-4, MUC-5, Rousselot 94 – among others). Contrary to approaches which go for semantics directly, or which apply statistical methods with little regard for the linguistic aspect, we believe that it is both justifiable and rewarding to start from the text itself, analyze it thoroughly, and acquire all knowledge from the text and from lazy interaction with the user (that is, when that knowledge is needed).

This paper deals with the conversion of a linguistically motivated semantic structure into a subset of classical logic (Horn clauses), that is, with the conversion from *linguistic semantics* (which Frawley 92 defines as: "literal, decontextualized, grammatical meaning") to one which is more remote from the textual description, and more appropriate for a rule-based expert system.

Brazilian Symposium on Artificial Intelligence (SBIA), Fortaleza, Ceara, Brazil, October 1994

2. Presentation of the system: TANKA and MaLTe

TANKA (Text ANalysis and Knowledge Acquisition – Figure 1) is a natural language analyzer consisting of two main components:

- the parser (DIPETT) is general and strictly syntactic. It can be used in batch mode or interactively – but with deliberately very little interaction, reduced to choosing between idiomatic and non-idiomatic reading of compound terms.
- the semantic analyzer (HAIKU) is more interactive: the user can enter a case structure, or noun-phrase structure; or he can validate the guess which the system has made via a case-based reasoning mechanism: the analyzer maintains tables of syntactic patterns and case patterns encountered in previous interactions, so as to be able to propose one or several options if the context is the same, or even only similar, to one or several past contexts.

A complete description of TANKA is provided in Delisle 94.

The aim of TANKA and MaLTe (Machine Learning from Text) is to achieve knowledge acquisition from technical texts in general, without forsaking the complexity of linguistic analysis and without first encoding a lot of encyclopedic knowledge as in CYC (Lenat & Guha 1990), or domain knowledge as in MUC. On the contrary it is a *knowledge-scant*, *interactive* and *open-ended* (evolutive) system where knowledge is added through user interaction when needed, self-observation (case-based reasoning – Delisle et al. 93b) and machine learning. The idea of designing such a *trainable* system is to strike a balance between systems which are totally pre-designed, and blind statistics-based systems. (See Delisle et al. 93a). Regarding its aims and characteristics, TANKA is not a query translator to logic or SQL, or a news-story understander, nor is it the illustration or implementation of one particular linguistic theory. It is a tool for knowledge acquisition.

TANKA has separate syntax and semantic modules to keep the parser portable, while the semantic analysis has to be interactive and hence, to some extent, domain- and user-related. The semantic module produces an output which is justified by linguistic criteria, plus disambiguation obtained from user choices reflecting personal or domain-oriented preferences - but it still a general-purpose output. The output at this stage is called the *protonetwork*; from which TANKA proceeds with the construction of a semantic network.

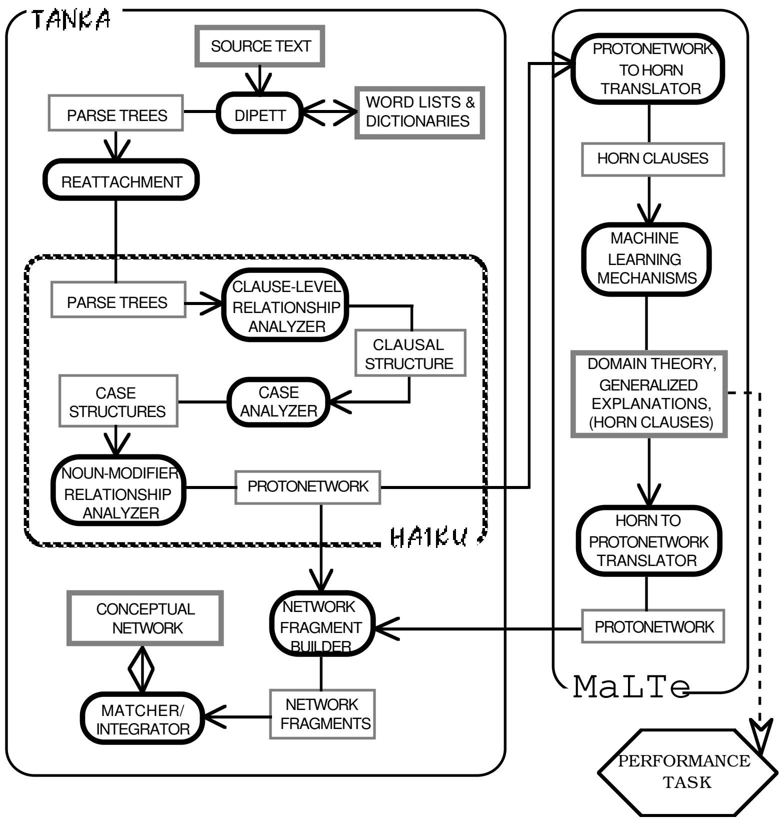

Figure 1: Architecture of TANKA and MaLTe

It is also from the protonetwork that MaLTe operates. If first converts the protonetwork into Horn clauses; this is the object of the rest of this article. MaLTe then applies machine learning to the Horn clauses, with special interest for the semantic representation of examples as commonly found in technical texts. Human readers exploit such examples via appropriate (operational) generalization. MaLTe simulates this by applying such machine learning techniques as:

- generalization/abstraction
- explanation-based learning (EBL; see Delannoy et al. 93)
- and interactive input of new rules when the EBL mechanism finds that there may be a missing link to complete a logical proof (developed by Louis Massey).

To describe more precisely how TANKA and MaLTe fit together: HAIKU takes the output of the parser and rearticulates the clauses and case structures. The output of HAIKU is the accretion of syntactic and semantic constructs:

- noun phrases are mapped to binary Noun Modifier Relationships (NMRs) applied to their central concept (whose naming is generally done based on the head noun);
- clauses map to case structures in HAIKU, and to logical terms in the translation;

- finally, inter-clausal relations (CLRs: Clause-Level Relationships) represent the logical articulation of the clauses.

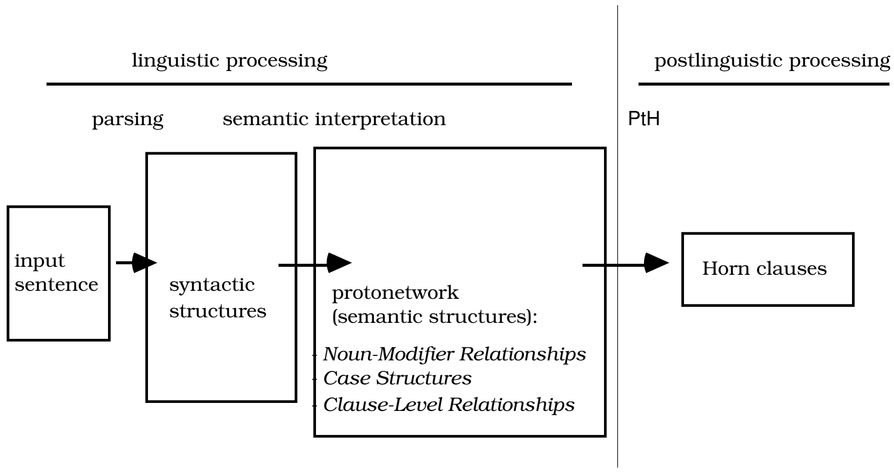

Figure 2: Processing flow

3. Principles and implementation of the PtH translator

3.1 General method

Looping over clauses, the semantic analyzer maps the noun phrases of a clause into NMR relations applied to an identifier associated to the head noun. The NMR analyzer reanalyzes each noun phrase parse tree to retrieve modifiers (adjectives, pre- and post-modifiers of the head noun, and noun-phrase modifiers) using a ‘grammar’ of trees. The logical translation of the noun phrases is passed on to the analysis of Case Structures, and from there to the translation of the CLRs. Tables of conversion mapping syntactic patterns and semantic output are used to allow case-based reasoning in later interaction.

3.2 Example

Consider the following input sentence:

"Jim and his family are deemed residents of Canada because Jim serves in the armed forces."

The parse tree associated with that sentence is the following, in a simplified form. Comments are preceded by "%".

```
minimal_pt(4,
  parse_tree_decla_or_imper( compound_sentence(
        structure(head_main_clause(
              declarative(statement( predicate(regular(                    % the first clause
                    verb(deem, tense([present_simple]), neg(yes), trans(tr))), voice(passive),     % verb
                    complement(svoo([                                          % complement
                          entity(head_noun(noun(resident)), number(pl),
                          np_postmodifiers([conj_pps(...,  [pp(of, entity( proper_noun(canada),
                                            number(sg3), tokens([canada])), tokens([of, canada])), ...
                                noun_modifiers(pre_modif(conj_noun_pre_mod(implicit_and, [income, tax]))))), ...
   …,
                                conj_nps(and, pl, [entity(proper_noun(jim), number(sg3), ...
              declarative(simple_sentence(structure(single_main_clause(
              declarative(statement(subj(entity(ref(pers_pron(he)), number(sg3), tokens([he]))),
                          predicate(regular(verb(serve, tense([present_continuous]), neg(yes), ...voice(active),
                          ...complement(svo([pp([[abroad, tokens([abroad])], in],
                    ...entity(determinatives(deter(the)), attributes([attrs(adj(arm, pos)), ...,
   end_of_input(period))).
```

The semantic analysis carried out by HAIKU produces a description structured as follows:

- input string: the original input;
- updated parse tree, after pronoun resolution;
- logical clausal structure: the articulation of clauses at the sentence level;
- the "symbolic a-list": the coordination structure of the sentence, an association list of clause references (in the form *statement#*) and clause parse trees, and an associations list of entity references and entity parse trees. ("entities" are in fact noun phrases).
- clause-level relationships: logical relations between clauses;
- case structure: the relation of a verb and its arguments;
- noun-modifier relationships: the semantic relations of the heads of noun phrases and their modifiers.

For the example above, Haiku produces the following output (here in a simplified form):

```
semantic_structure(4,[logical_input(
'INPUT STRING'
      ([[jim,and,his,family,are,deemed,residents,of,canada,for,income,tax,purposes,because,he,is,serving
                                          ,abroad,in, the,armed, forces]]),
'UPDATED PARSE TREE'
   (parse_tree_decla_or_imper(compound_sentence(structure(head_main_clause(declarative('
    *statement10*')),next_main_clause('*coord3*',declarative(*statement11*...
'LOGICAL CLAUSAL STRUCTURE'
  ([top_structure([['*statement10*','*coord3*','*statement11*']])]),
'SYMBOLIC A-LISTS' ([
coordinators([['*coord3*', coord(because, nil)]]),            % coordinators
clauses([                              % association list of clauses references and clause parse trees
```

```
    ['*statement10*', statement(predicate(regular(verb(deem, tense([present_simple]), neg(yes), trans(tr))),
    voice(passive), complement(svoo([entity(head_noun(noun(resident)), number(pl),
    np_postmodifiers([conj_pps(implicit_and, [pp(of, entity(proper_noun(canada), number(sg3), tokens([canada]))
    ...,                        % the parse tree of the first clause
    ['*statement11*', statement(subj(entity_pro(proper_noun(jim), number(sg3), ..., predicate(regular(verb(serve,
    tense([present_continuous]), neg(yes), trans(tr_intr))), voice(active), complement(svo([pp([[abroad,
    tokens([abroad])], in], entity(determinatives(deter(the)), ... tokens([the, armed, forces])),...,
                    % the parse tree of the second clause
'CLAUSE-LEVEL RELATIONSHIPS'
([clr_structure(enab,['*statement11*','*statement10*'])]),  % [the semantic of] the first clause    enables
                                                            % [the semantic of] the second
'CASE STRUCTURES'
([clauses([
['*statement10*', case_structure(
    verb(be, tense([present_simple]), neg(yes), trans(stat)),
    nil,                                 % the first clause is stative : it has no case markers,
    nil,                                 % and no case is assigned.
    conj_nps(and, pl, [entity(proper_noun(jim), number(sg3),
    entity(determinatives(deter(his)), head_noun(noun(family)), number(sg3), …
    entity(head_noun(noun(resident)), number(sg3),
   np_postmodifiers([conj_pps(implicit_and, [pp(of, entity(proper_noun(canada), number(sg3),
    tokens([canada])), ...,
['*statement11*',
    case_structure(verb(serve, tense([present_continuous]), neg(yes), trans(tr_intr)),
            psubj-in,                    % The case markers for this clause are    psubj  and in
            agt-benf,                    % and the case pattern consists of    agt and benf
            entity_pro(proper_noun(jim), number(sg3), tokens([jim])), [pp([[abroad, tokens([abroad])], in], ...
'NOUN MODIFIERS RELATIONSHIPS'
[   [nmr_struc(attributive, resident, canada)],
    [nmr_struc(manner, resident, income_tax_purposes)],
    [nmr_struc(specification, purposes,income_tax), ...] ]
....)]).
```

The updated parse tree as well as the logical clausal structure are generated by DIPETT. The rest is output by HAIKU.

3.3 Noun-phrase semantics

NMRs are terms based on a list of relations (see Table 2).

verbA' is a verb corresponding to the noun (as assigned by the user with the help of the Collins on-line dictionary)

The decision between one of the two relations for the appositive case is made according to criteria of definiteness of the determiners. User confirmation is required in all cases.

The reason why many NMRs map to predicate 'relation' is because most involve two entities. On the other hand, when a non-noun modifies a noun, it is more likely to be mapped to 'property' by the noun-phrase part of HAIKU, and from this, to predicate 'prop' in our module.

Noun-Modifier Relations	Logical term
nmr_struc(agentive, A,B)	act(verbA,[agt],B).
nmr_struc(appositive, A,B)	isa(A,B) or ako(A,B).
nmr_struc(benefactive, A,B)	rel(Label, A,B).
nmr_struc(causative, A,B)	act(verbA,[agt],B]) :- act(verbC,[agt],D])
nmr_struc(classificatory, A,B)	isa(A,B) or ako(A,B).
nmr_struc(compositional, A,B)	rel(compositional, A,B).
nmr_struc(destination, A,B)	rel(destination, A,B).
nmr_struc(humanrel, A,B)	rel(humanrel,A,B).
nmr_struc(instrumental, A,B)	rel(instrumental, A,B).
nmr_struc(locative, A,B)	rel(locative, A,B).
nmr_struc(objective, A,B)	rel(objective, A,B).
nmr_struc(possessive, A,B)	rel(possessive, A,B).
nmr_struc(productive, A,B)	rel(productive, A,B).
nmr_struc(property, A,B)	prop(A,B).
nmr_struc(purposive, A,B)	rel(purposive, A,B).
nmr_struc(source, A,B)	rel(source, A,B).
nmr_struc(temporal, A,B)	rel(temporal, A,B).
nmr_struc(topic, A,B)	rel(topic, A,B).

Table 1. Translation table between Noun Modifier Relations and assertable terms, for the PtH module

3.4. Case-structure semantics

The case analyzer (Barker et al. 93; Delisle et al. 93) returns a case pattern corresponding to the case-marker pattern, so as to give a semantic description to the relation of the verb and its arguments. It is initially assigned by the user, and the mapping is kept in the memory of the system, which will be able to propose it for later processing of the same verb with similar case-marker patterns.

With *stative* clauses (i.e. "be" and verbs of state, often used in technical texts for definitions), HAIKU assigns no case structure, as is generally done in linguistic case theory (but the NMR analyzer will have returned an *isa* or *ako* relation).

With *non-stative* clauses, HAIKU uses the verb and case pattern, using functor “act”. In the example, case analysis construct the term:

```
act(serve, [agt, benf], [jim, armed_forces]).
```

In that example, the case-marker pattern *psubj_pobj* has been mapped to the case pattern *[agt, benf]*.

3.5. Clause-level relationships

The Clause-level analyzer takes markers and assigns the appropriate relation interactively (the user has to confirm; the choice of the system is correct in more than 80% of the cases in our measurements done on a sample from the Canadian Tax Guide). The treatment applied then by the Horn-clause translator to inter-clausal relations is based on the rules of conversion shown in Table 2:

Name of Relation	**Logical correspondence (in Prolog)**	**Remarks**
entailment	B:-A	that is, given A, B can be proved
enablement	B:-A	
prevention	¬B :- A	the use of a negative predicate head is possible in Quintus Prolog
entailment (or en-ablement) associated with prevention	C :- A, ¬B	an exception overrides a positive rule (as in “you can deduct childcare expenses if they were paid to your neighbor, unless he or she is a relative” .
detraction	A, B	discarding the info that normally A could prevent B (this would require modality)
temporal prece-dence	A, B	temporal relations are not taken into account, so far
cotemporality	A, B	
conjunction	A, B	
disjunction	¬(A and B)	for non-exclusive OR
	¬(A and B) and ¬(B and A)	for exclusive OR

Table 2: Correspondences between inter-clausal relations and logical rules

3.6. PtH: Logical translation

Finally, in the PtH module, the NMRs are mapped to logical facts (using the table represented in Table 1), and the CLRs to logical rules based on terms constructed from the Case patterns.

The example of Section 3.2 translates into the following Horn clauses:

```
isa(family1, family).
```

```
isa(armed_forces1, armed_forces).
isa(jim, deemed_resident_of_canada) :-
   act(serve, [agt, benf], [jim, armed_forces1]).
isa(family1, deemed_resident_of_canada) :-
   act(serve, [agt, benf], [jim, armed_forces1]),
   rel(humanrel, family1, jim).
```

These clauses will allow to prove that Jim and his family are deemed residents, from the knowledge that he is serving in the armed forces. Of course this is strictly local to the example, and has to be generalized by machine learning to any persons and families in similar situations, up to a user-defined level of generality in the hierarchy of occupations - for instance, civil servants.

4. Discussion, conclusion

As said above, the translation of the first semantic representation, as produced by the analyzer, comprises the three levels of inter-clausal relations (with tables), case structures, and noun-modifier relations (using a table plus the result of the reanalysis of part of the case structures).

The sophistication and cost of this method for obtaining a logical translation of a text are justified by a more principled processing, better portability, and versatility of the acquisition tool. The complexity of the logical semantic constructions is not limited in principle because the semantics is not accommodated into pre-defined frames.

This is a way to take full account of the linguistic structures of technical texts, for which the syntax is generally most indicative of the meaning, and to widen the range of potential applications of the output. In short, the linguistic semantics (protonetwork) is anchored in the text; the first translation into Horn clauses is more domain-dependent (and made so by user intervention); machine learning optimizes the contents of this translation, with considerations of both the domain and the application task. An interesting task for the Canadian Tax Guide is the production of expert advice on eligibility for deduction.

A prototype of TANKA and MaLTe including the translator has been developed in Quintus Prolog on a Sun Sparc workstation.

Further work will address the *learning* of the PtH process: case-based reasoning will retrieve a situation with a sufficient structural similarity and propose, *mutatis mutandis*, a plausible target Horn clause. Techniques considered for this involve both symbolic and statistical methods (Mercer 1993). Another topic is the evaluation of the *decrease in user interaction*. The burden on the user should be high initially, but should decrease not only quantitatively but also qualitatively (tending to mere confirmation of proposed choice, rather than value entry).

Acknowledgements

This work is supported by NSERC in Canada and by CAPES in Brazil.

Thanks for advice, discussion and graphical material to all of the TANKA/MaLTe team: Stan Matwin, Stan Szpakowicz, Ken Barker, Terry Copeck, Cao Feng, Sylvain Delisle, and Louis Massey.

References

Banff 94: *Proceedings of the Banff Workshop on Knowledge Acquisition (1994), workshop on Acquisition from text*

Barker K., Copeck T., Delisle S., Szpakowicz S. (1993): "An Empirically Grounded Case System". TR-93-08, Department of Computer Science, University of Ottawa.

Britton B.K., Black J.B.(1985): *Understanding Expository Text. A Theoretical and Practical Handbook for Analyzing Expository Text,* Lawrence Erlbaum Associates

Copeck T., S. Delisle & S. Szpakowicz (1992): “Parsing and Case Analysis in TANKA”, *Proceedings of the 15th Annual Conference on Computational Linguistics* (COLING-92), Nantes, July 1992, pp. 1008-1012.

Delannoy, C. Feng, S. Matwin. S. Szpakowicz (1993): “Knowledge Extraction from Text: Machine Learning for Text-to-Rule Translation”, *Proceedings of the Machine Learning and Text Analysis Workshop, European Conference on Machine Learning (ECML-93),* Vienna, pp. 1-7.

Delisle S.(1994): *Text Processing without a priori domain knowledge: semi-automatic linguistic analysis for incremental knowledge acquisition*, PhD, University of Ottawa, Canada

Delisle S., K. Barker, JF Delannoy, S. Matwin, S. Szpakowicz (1994): “From Text to Horn Clauses: Combining Linguistic Analysis and Machine Learning", *Canadian Artificial Intelligence Conference* 1994

Delisle S., Barker K., Copeck T., Szpakowicz S. (1993): "Interactive Semantic Analysis of Technical Texts". submitted to *Computational Intelligence* (in revision); an earlier version is available as TR-93-13, Department of Computer Science, University of Ottawa

Delisle S., T. Copeck, S. Szpakowicz and K. Barker (1993b): "Pattern Matching for Case Analysis: A Computational Definition of Closeness". in O. Abou-Rabia, C. K. Chang and W. W. Koczkodaj (eds.) *Proc ICCI-93*, pp. 310-315.

Frawley, W. (1992): *Linguistic Semantics*, Lawrence Erlbaum Associates

ICCS93: Proceedings of the International Conference on Conceptual Structures, Quebec City, Canada, August 1993

Lenat D.B., Guha R.V. (1990): *Building Large Knowledge-Based Systems*, Addison-Wesley

Mercer L.: “Computational Linguistics Using Large Corpora”, *Computational Linguistics*, vol. 19, Number 1, March 1993

MUC-4: *Proceedings of the Fourth Message Understanding Conference*, 1992, Morgan-Kaufmann

MUC-5: *Proceedings of the Fifth Message Understanding Conference*, 1993, Morgan-Kaufmann

Rousselot F, ed. (1994): *Atelier d’Ingénierie des Connaissances et des Données (AICD)*, Strasbourg, France, January 1993