Controlling logical scope in text generation*

Richard Power
ITRI, University of Brighton
Lewes Road, Brighton BN2 4AT, UK
Richard.Power@itri.bton.ac.uk

Abstract

Many applications of Natural Language Generation employ object-oriented knowledge representation formalisms such as LOOM. Each entity in the domain is represented by an object belonging to a specified class and possessing attributes suitable for that class. From a logical viewpoint, domain entities correspond to existentially quantified variables, but since LOOM and kindred formalisms provide no scoping mechanisms, it has to be assumed that all variables have wide scope. For technical material including conditional, negative or modal contexts this is a serious limitation. We suggest a simple extension to standard object-oriented formalisms which allows encoding of logical scope, and show how logically complex material can be edited using the 'WYSIWYM' approach.

1 Introduction

Many Natural Language Generation (NLG) applications (Paris et al., 1995; Bateman, 1996) encode the content of the generated texts in an object-oriented knowledge representation formalism, such as LOOM (MacGregor and Bates, 1987). In LOOM and kindred languages, a distinction is made between the 'T-box', which defines the classes to which objects may belong, and the 'A-box', which describes the properties of specific entities belonging to these classes. The T-box serves as an ontology that determines the semantic coverage of the system; the A-box represents the content of a generated text, or perhaps a wider body of knowledge from which this content is drawn.

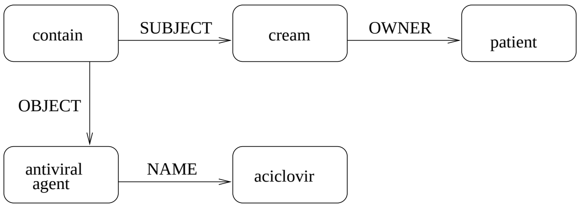

Figure 1: A-box for simple assertion

The semantic network in figure 1 shows a simple A-box underlying the sentence [1]:

> Your cream contains an antiviral agent called aciclovir

Each node of the network represents an entity; the node label represents the class of the entity; outgoing arcs represent its properties. In a language like LOOM, the A-box would be encoded through a set of assertions:

```
(tell (contain u1)
      (subject u1 u2)
        (cream u2)
        (owner u2 u3)
```

*This work is supported by the Engineering and Physical Sciences Research Council (EPSRC) Grant L77102.

[1]Throughout the paper, our examples will be taken from the domain of Patient Information Leaflets, the instruction slips inside medicine packets.

```
        (patient u3)
    (object u1 u4)
      (antiviral-agent u4)
      (name u4 u5)
        (aciclovir u5))
```

How should such a set of assertions be interpreted logically? The usual assumption is that the 'entities' `u1`, `u2` etc. (or the nodes in figure 1) should be treated as existentially quantified variables with wide scope. Under this assumption, the content of the A-box could be represented by a single predicate calculus formula:

$$[\exists u_1 \exists u_2 \exists u_3 \exists u_4 \exists u_5\ contain(u_1) \\ \& subject(u_1, u_2)\ \&\ cream(u_2) \\ \& owner(u_2, u_3)\ \&\ patient(u_3) \\ \& object(u_1, u_4)\ \&\ antiviral(u_4) \\ \& name(u_4, u_5)\ \&\ aciclovir(u_5)]$$

So far so good, but what happens when an assertion contains an variable with narrow scope? This happens often in instructional domains such as patient information leaflets, in which conditional, negative and modal contexts abound. Consider for example the instruction

> If you develop a rash, you should consult your doctor.

for which a plausible logical form is

$$[\exists p \exists d\ doctor(p, d)\ \& \\ [[\exists r\ rash(r)\ \&\ develop(p, r)] \rightarrow \\ \Box consult(p, d)]]$$

The rash r cannot be assigned wide scope here without implying that it is a 'real' entity, like the patient and the doctor, rather than a hypothetical entity bound to the antecedent of the conditional.

Attempts have been made to generate such sentences from a LOOM knowledge base. The PENMAN Generalized Upper Model (Bateman et al., 1995) allows abstract entities representing logical relations, by providing classes equivalent to the logical operators of `implication` and `obligation` (the concept

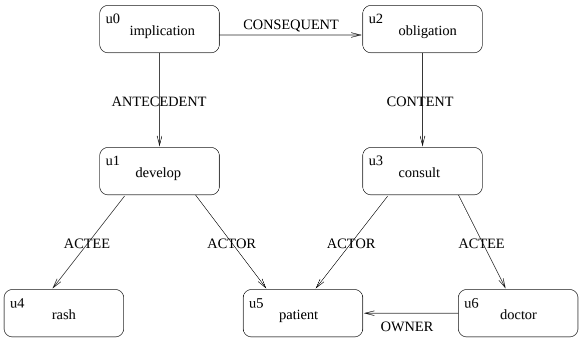

Figure 2: A-box for conditional instruction

names actually used in the Upper Model are LOGICAL-CONDITION and NECESSITY). An A-box that approximately represents the meaning of the conditional instruction can therefore be defined, as in figure 2. However, this A-box fails to indicate that the `patient` and `doctor` entities have wide scope while the `rash` entity has narrow scope.

This paper has two aims. First, we propose a simple method by which an object-oriented knowledge formalism like LOOM can be augmented in order to represent scope distinctions. Secondly, we show how the augmented knowledge representation can be edited by the 'WYSIWYM' method (Power and Scott, 1998; Scott et al., 1998), so allowing authors who are not logicians or knowledge engineers to encode logically complex knowledge.

2 Logical context

Since the beginning of modern logic, network diagrams have been employed as convenient representations of logical form with implicit existential quantification. Pierce's Existential Graphs (Roberts, 1973) were introduced in 1897; according to Sowa (1995) they are formally equivalent to Discourse Representation Structures (DRSs) (Kamp, 1981). In both notations, scoping assignments are made by binding entities and relationships to *logical contexts*, represented by 'oval enclosures' in Existential Graphs and by 'boxes' (or DRSs) in Discourse Representation Theory (DRT). Sowa's more elaborate Conceptual Graph notation represents logical contexts through complex 'proposition' nodes which enclose sub-networks. Hendrix (1975) has proposed a similar notation in which the nodes and arcs of a semantic network are assigned to 'spaces', but since he allows these spaces to overlap, his diagrams cannot always be drawn unambiguously on a plane.

All these representations are capable of expressing scope distinctions, so the choice among them is dictated mainly by convenience. We will adopt here the DRT framework, which has been employed in much recent work in linguistics. Figure 3 shows a DRT diagram for the sentence 'If you develop a rash you should consult your doctor'. For readers unfamiliar with DRT, the conventions of the diagram will be explained briefly. It contains four *boxes* labelled $K_0..K_3$, five *discourse referents* (u_1, u_3, u_4, u_5, u_6) each assigned to the *universe* of a box, ten *simple conditions* in predicate-argument form, and two *complex conditions* based on the logical operators of implication ($\Rightarrow$) and necessity ($\Box$). Intuitively, each box represents a logical context involving some entities (discourse referents) that belong to its universe. The outer box K_0 represents the context affirmed by the author of the patient information leaflet; it includes the patient u_5, the patient's doctor u_6, three simple condition presenting properties of these referents, and a complex condition presenting the conditional instruction. The instruction in its turn contains a box K_1 describing a *hypothetical* context in which the patient develops a rash. A crucial idea in DRT is that a referent may be cited only in logical contexts that are *accessible* from the box to which it is bound. For instance the rash u_4, which is bound to the hypothetical context K_1, could occur within a simple condition in any of the boxes subordinated to K_1 (i.e. K_2 and K_3), but not in the outer box K_0. Intuitively, to cite u_4 within K_0 would imply that the rash definitely existed, in which case u_4 should have been bound to K_0, not to K_1.

3 Scoped semantic network

What is the minimum information that need be added to the A-box in figure 2 in order to represent fully the logical form of the conditional instruction, as shown by the DRS in figure 3? This question can be addressed by seeking a procedure for interpreting a semantic network like figure 2 as a DRS, and noting exactly where this procedure stalls.

The *first* step in interpreting figure 2 as a DRS is to distinguish nodes that express simple conditions from ones that express complex conditions. This is straightforward: the T-box classification should distinguish logical operators like `implication` and `obligation` from non-logical entities like actions or physical objects, so in figure 2 we can conclude that u_0 and u_2 express complex conditions while all other nodes express simple conditions.

The *second* step is to pair some nodes with DRT boxes; this can be done by applying two rules.

- The root of the network is paired with the outer box K_0.
- If a node expresses a complex condition, its direct descendents (i.e. the values of its attributes) are paired with boxes.

In figure 2, these rules yield four boxes: K_0 is paired with u_0, the root of the network; K_1 and K_2 are paired with u_1 and u_2, the direct descendents of the implication u_0; and K_3 is paired with u_3, the direct descendent of the obligation u_2.

The *third* step is to place each complex condition in the correct box, so obtaining the box structure in figure 3. Again two rules suffice:

- If a node u_A expressing a complex condition is paired with a box K_A, the complex condition should be placed in K_A.

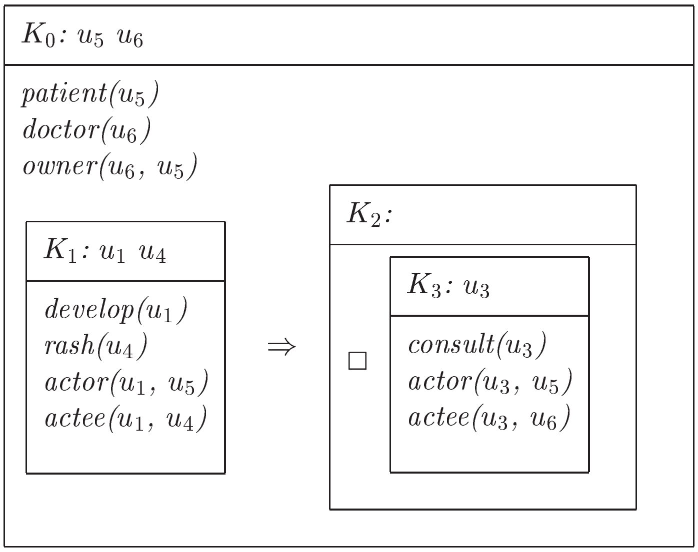

Figure 3: DRS for a conditional instruction

- If u_A is not paired with a box, we find the nearest ancestor u_B that is paired with a box K_B, and place the complex condition expressed by u_A in K_B. (There must be a unique nearest ancestor, since a network is ill-formed if there are two paths leading from the root node to a node expressing a complex condition.)

Applying these rules to figure 2, we can place the implication u_0 in K_0, and the obligation u_2 in K_2, the consequent of the implication.

The *fourth* step is to bind each referent represented by a node in figure 2 to the correct box. We can ignore the nodes expressing complex conditions (u_0 and u_2), since abstract entities of this kind are usually omitted from DRT diagrams. Among the other nodes, we can distinguish nodes paired with boxes (u_1 and u_3) from the rest (u_4, u_5 and u_6). The former will be called *fixed referents*, the latter *floating referents*. The scope of a fixed referent is determined: it must be bound to the box it is paired with. Thus u_1 must be assigned to the universe of K_1, and u_3 must be assigned to the universe of K_3. For floating referents there may be a choice. By applying DRT accessibility rules, a set of *potential scopes* can be calculated for each floating referent; if this set contains more than one box the scope of the network is underspecified, and so the network fails to determine a unique DRS.

We now have an answer to our original question: to interpret an A-box like figure 2 as a DRS, *each floating referent must be assigned a scope.* Since fixed referents are paired with boxes, this can be done by linking each floating referent to a fixed referent that belongs to its set of potential scopes; we will call the result a 'scoped semantic network'. In figure 4, scope assignments are shown by dotted arcs; as an aid to visualization, floating referents are represented by ovals and fixed referents (paired with boxes) by rectangles.

Having assigned each referent to a box, the *fifth* and final step is to place the simple conditions in the correct boxes. Simple conditions are of two kinds, assertions of class membership such as *consult*(u_3) and attributions such as *actor*(u_3, u_5), the former expressed by nodes in the network and the latter by arcs. We assume that if a referent u is bound to a box K, then all simple conditions de-

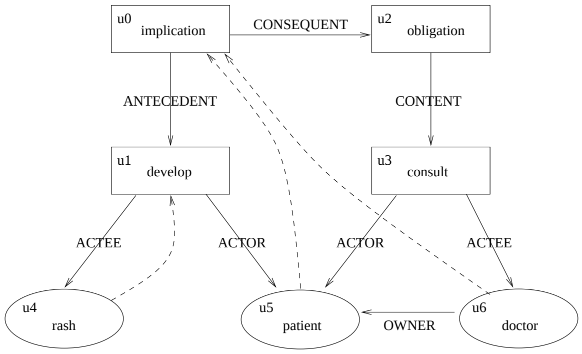

Figure 4: A-box with scope assignments

scribing properties of u (i.e. its class and its attributes) should be placed in K. Thus $actor(u_3, u_5)$ is placed in K_3 because it describes a property of u_3, which is bound to K_3. In general, any simple condition of the form $r(u_A, u_B)$ is placed in the box to which u_A is bound, even if u_B is bound to a different box.

It might seem that this assumption is too strong. By comparison, Hendrix (1975) binds each arc as well as each node to a logical context; consequently, a partioned semantic network would allow the simple conditions $consult(u_3)$, $actor(u_3, u_5)$ and $actee(u_3, u_6)$ to be bound to three different boxes, whereas a scoped semantic network constrains them to appear in the same box. However, this apparent limitation is actually a useful simplification in an A-box that *reifies* relations like `develop` and `consult`. In a logic textbook, the three simple conditions in K_3 would typically be collapsed into a single formula $consult(u_5, u_6)$; in DRT the notation $u_3 : consult(u_5, u_6)$ is sometimes used if there is a subsequent reference to the consulting event itself. Obviously this single condition cannot be distributed among several boxes, so our assumption causes no loss of expressive power compared with the normal DRT notation.

4 Scope constraints

In WYSIWYM editing an A-box is built top-down starting from the root node. Having decided that the root node should be (say) an entity of type `implication`, the author is presented with a feedback text indicating that an antecedent and a consequent must be specified:

> If **this situation** holds, **this consequence** results.

The phrases in bold face are mouse-sensitive anchors marking unfilled attributes; by clicking on an anchor, the author can view the classes to which a suitable filler may belong, and choose one of them to create the filler. Every time a choice is made, the feedback text is updated and new editing options are offered. By successive choices on anchors, the author can build up a complete network such as figure 2. However, as we have seen, for logically complex material a standard A-box such as figure 2 is insufficient. To define the desired meaning completely, the author must somehow assign scopes to the floating referents, as in figure 4.

The control of scope during editing raises several tricky problems:

1. When should the options be presented? Should a scoping decision be required whenever a new entity is introduced, or should the system allow the scope to be left temporarily unspecified?

2. How can the decision be presented in a way that non-logicians can understand?

3. How should the editing operations of Cut and Paste affect scoping assignments? Suppose for example that in figure 2 u_5 was assigned the scope u_1, what would happen if either u_1 or u_5 were removed from the main network by a Cut operation?

4. How should the system react if the author attempts an editing operation that

would be illegal under the current scoping assignments? Should it forbid the operation, or offer the operation on condition that some scopes are reassigned?

As an illustration of the last problem, suppose that the author has constructed the network in figure 2, except that the `actor` arc from u_1 has not yet been directed to the `patient` node u_5; and suppose further that the author (probably by mistake) has bound u_5 to the local context K_3. The resulting feedback text might be

> If **this person** develops a rash, a patient should consult his/her doctor.

In WYSIWYM editing, an existing entity can be copied into an unfilled attribute by selecting a phrase denoting the entity (e.g. 'the patient'), choosing 'Copy' from the pop-up menu, selecting the anchor denoting the unfilled attribute (**this person**) and choosing 'Paste' (van Deemter and Power, 1998). Allowing this operation would install the patient as the `actor` of the `develop` action u_1; unfortunately it would also violate DRT scoping rules, since a simple condition $actor(u_1, u_5)$ in K_1 would now cite a referent u_5 bound to a box (K_3) that is inaccessible from K_1.

All four problems listed above can be solved (or at least alleviated) by introducing a further simplifying assumption:

> Every floating referent must either be bound to the outer box K_0, or must be interpreted *in situ* (i.e. bound to the narrowest of its potential scopes).

With this assumption, the author need no longer provide an exact scoping decision; it suffices to distinguish 'Global scope' (assignment to K_0) from 'Free scope'. By adopting free scope as a default, all we require from the author is a decision over which floating variables should be bound to K_0. Such a decision cannot violate accessibility constraints, since the outer box is always among the potential scopes of a floating referent. It can also be presented in a way more familiar to non-logicians, as a distinction between 'real' entities, which (according to the text) definitely exist, and 'hypothetical' entities, which might or might not exist. Referents that have been cut to the buffer can retain their global/free status, and interpreted accordingly if they are pasted back into the main network. Finally, in the situation described above, the `patient` referent u_5, with its scope left free, could be copied without inconsistency into the `actor` attribute on u_1; by removing K_2 and K_3 from its set of potential scopes, this operation would in effect change the exact scope assigned to u_5 from K_3 to K_1.

5 Controlling scope during editing

We now show in detail how an author might construct the scoped network in figure 4 (equivalent to the DRS in figure 3) by WYSIWYM editing. The method described above has been implemented fully in the ICONOCLAST system, which generates patient information leaflets; for simplicity we assume that the system has been configured to generate a single instruction only rather than a complete leaflet. On starting a new A-box, the author sees the feedback text

> Follow **this instruction**.

implying that the root of the network should be some kind of instruction; of course the aim of WYSIWYM to present such information in ordinary English (or ordinary French, etc.), avoiding technical terms like 'network' or 'A-box'. Opening the anchor yields a list of instruction types

action
conditional
procedure
warning
...

from which the author should select 'conditional'; probably these alternatives could be worded in a way that was more accessible to the intended users, but we will not pursue this issue here. The A-box is now updated by creating an `implication` entity with undefined antecedent and consequent, and the feedback text is regenerated to show the new editing options:

> If **this situation** holds, **this consequence** results.

The anchors may be developed in any order, but we assume that the author clicks first on **this situation** [2]

allergic
develop
pregnant
...

and chooses a situation of type `develop`, thus specifying the node u_1 in figure 4. Since it descends directly from a complex condition, u_1 is a fixed referent, so there is no need to offer the author an editing operation for controlling its scope. The feedback text changes to

> If **this person** develops **this symptom**, **this consequence** results.

whereupon the author might click on **this person**

doctor
nurse
patient
pharmacist
...

and choose 'patient', thus specifying u_5 (figure 4). This time, the newly specified node descends from a simple condition, so it represents a floating referent. The current network implies three boxes, linked to the nodes u_0 (the root), u_1 (the antecedent of a complex condition) and u_2 (the consequent); by

[2]To save space, just a few alternatives are shown.

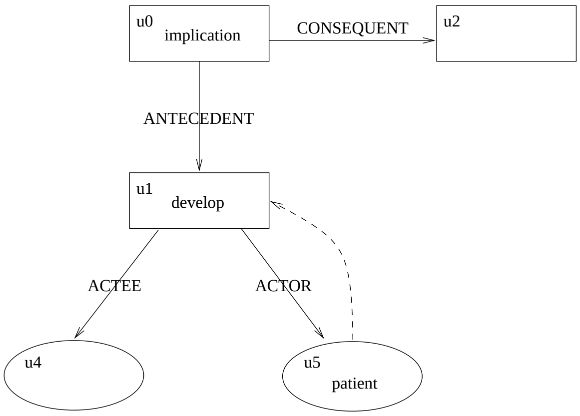

Figure 5: Scoped network during editing

DRT rules, u_5 could be bound either to K_0 or K_1 (represented in the network by u_0 and u_1 respectively). However, since a newly introduced referent has free scope, it is initially bound to u_1, the narrowest of its potential scopes. The resulting network (figure 5) is presented through an updated feedback text

> If a patient develops **this symptom**, **this consequence** results.

in which the indefinite description 'a patient' suggests that the instruction is addressed to *any* patient, not specifically to the patient reading the leaflet. Since this is not the desired meaning, the author can now click on 'a patient':

Cut
Copy
Global scope

When a text span expresses a node of specified type (i.e. it is not an anchor), the associated menu always offers the options 'Cut' and 'Copy'; in addition, for a floating referent, it offers 'Global scope' if the referent is currently unconstrained, and 'Free scope' if it is constrained. (Again, we will eventually word this option in a more accessible way.) Selecting 'Global scope' modifies figure 5 by redirecting the dotted arc to u_0, thus binding

u_5 to the outer box K_0 in the corresponding DRS. Making the reasonable assumption that a patient entity in K_0 denotes the reader, the system can now update the feedback text as follows:

> If you develop **this symptom**, **this consequence** results.

To complete the antecedent, the author clicks on **this symptom** and chooses 'rash'.

> If you develop a rash, **this consequence** results.

This time, the default scope assigned to the newly specified referent u_4 is appropriate, so the author develops the consequent by choosing 'obligation' and then 'consult':

> If you develop a rash, **this action** should be performed.
>
> If you develop a rash, **this person** should consult **this person**.

At this point, an inexperienced author might err by clicking on the first anchor **this person** and selecting 'patient'. The resulting feedback text reveals the nature of the error:

> If you develop a rash, another patient should consult **this person**.

a new patient has been introduced instead of referring back to the original one. To undo the mistake, the author should select 'Cut' on the phrase 'another patient', so reverting to the previous feedback text, and then click on 'you'

Cut
Copy
Free scope

and select 'Copy'. (Note that the rescoping option is now 'Free scope' rather than 'Global scope', since the scope of the `patient` referent u_5 was constrained earlier.) Clicking on the anchor **this person** will now yield a menu that includes a 'Paste' option in addition to the list of permitted types:

Paste
doctor
nurse
patient
pharmacist
...

The result of selecting 'Paste' is to install u_5 (the copied referent) as the value of the `actor` attribute on u_3 (figure 4).

> If you develop a rash, you should consult **this person**.

It remains to define the person that should be consulted, by choosing 'doctor' and then pasting in the patient again:

> If you develop a rash, you should consult a doctor of **this person**.
>
> If you develop a rash, you should consult a doctor of yours.

The doctor referent still has free scope, so it is bound to its local context u_3 (paired with the DRT box K_3). However, the intended meaning is not just any doctor that the patient might have, but his/her *actual* doctor. The author should therefore click on 'a doctor of yours' and select 'Global scope' to complete the network in figure 4.

> If you develop a rash, you should consult your doctor.

6 Conclusions

We have argued three main points:

- Knowledge bases in NLG applications often fail to represent the scopes of existentially quantified variables.
- The standard A-box used in LOOM and similar languages can represent logically complex propositions if each 'floating referent' is bound by a scoping relation to an accessible 'fixed referent'; the resulting network can be interpreted as a DRS.

- The task of constructing a logically complex proposition can be conveniently simplified, for users who are not trained in logic or knowledge engineering, by requiring only a decision between 'global' and 'free' scope, with free scope serving as the default.

These proposals have been implemented in ICONOCLAST, a system that uses WYSIWYM editing in order to define the content of patient information leaflets. An important task for future research is to evaluate whether the system can be used reliably by our intended authors — domain experts in the pharmaceutical industry. In particular, we need to confirm that non-logicians can distinguish global scope from free scope, and to find out how this distinction can be most clearly presented.

References

Bateman, J. (1996). KPML: The KOMET-Penman (Multilingual) Development Environment. Technical report, Institut für Integrierte Publikations- und Informationssysteme (IPSI), GMD, Darmstadt. Release 0.9.

Bateman, J., Magnini, B., and Fabris, G. (1995). The generalized upper model knowledge base: organization and use. In *Conference on Knowledge Representation and Sharing*, Twente, The Netherlands.

Hendrix, G. (1975). Expanding the utility of semantic networks through partioning. In *Fourth International Joint Conference on Artificial Intelligence*, pages 115–121, Tbilisi.

Kamp, H. (1981). Events, discourse representations, and temporal references. *Languages*, 64:39–64.

MacGregor, R. and Bates, R. (1987). The LOOM knowledge representation language. In *Proceedings of the Knowledge-Based Systems Workshop,* St. Louis, April 21–23.

Paris, C., Vander Linden, K., Fischer, M., Hartley, A., Pemberton, L., Power, R., and Scott, D. (1995). A support tool for writing multilingual instructions. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 1398–1404, Montreal, Canada.

Power, R. and Scott, D. (1998). Multilingual authoring using feedback texts. In *Proceedings of the 17th International Conference on Computational Linguistics and 36th Annual Meeting of the Association for Computational Linguistics*, pages 1053–1059, Montreal, Canada.

Roberts, D. (1973). *The Existential Graphs of Charles S. Pierce.* Mouton, The Hague.

Scott, D., Power, R., and Evans, R. (1998). Generation as a solution to its own problem. In *Proceedings of the 9th International Workshop on Natural Language Generation*, pages 256–265, Niagara-on-the-Lake, Canada.

Sowa, J. (1995). Syntax, semantics and pragmatics of contexts. In *Third International Conference on Conceptual Structures*, Santa Cruz, USA.

van Deemter, K. and Power, R. (1998). Coreference in knowledge editing. In *Proceedings of the COLING-ACL workshop on the Computational Treatment of Nominals*, pages 56–60, Montreal, Canada.