Text Generation in a Dynamic Hypertext Environment

Maria Milosavljevic, Adrian Tulloch and Robert Dale
Microsoft Institute
Macquarie University
65 Epping Road, North Ryde NSW 2113
{*t-mariam, t-atullo, rdale*}*@microsoft.com*

Abstract

This paper describes PEBA-II, a working natural language generation system which interactively describes animals in a taxonomic knowledge base via the production of World Wide Web pages. Our aim is to construct a natural language document generation system with real practical applicability: to this end, the system reconstructs and combines a number of existing ideas in the literature in a novel way, and proposes a solution to the problem of breadth of coverage that is based on a pragmatic approach to knowledge representation and linguistic realisation. The system embodies the following features:

- *a reconstruction of some of the core ideas in schema–based text generation [McKeown 1985], applied to the generation of hypertext documents;*
- *the principled use of a phrasal lexicon to ease surface generation, in concert with a knowledge base whose elements may correspond to pre–compiled collections of atomic units;*
- *a user model and discourse model that permit interesting variations in the texts produced.*

We describe each of the above aspects of the existing system in some detail, and point to a number of interesting research directions it opens up.

Keywords natural language processing, natural language generation, hypertext.

1 Introduction

A common complaint about existing multimedia resources is that they have a single audience model hard-wired, and effectively try to be all things to all people. Ultimately, we need computational mechanisms that can retrieve information from various sources and summarise and tailor the presentation of this information to meet the needs of particular users. As a small step towards this goal, this paper describes PEBA-II, a natural language generation system which interactively describes entities in a taxonomic knowledge base via the dynamic generation of hypertext documents (in fact, World Wide Web pages). In PEBA-II, the information provided to the user varies depending upon the context of use: at this stage, the system produces different texts for novice and expert users, but one might also imagine different descriptions for adults as opposed to children, for people in a hurry as opposed to those who are undertaking leisurely browsing, for those who have read the information before but want a refresher as opposed to those who are viewing it for the first time, and, eventually, for users who want the information in different languages.

In this paper we describe a number of the key ideas that underlie our current work:

- the knowledge representation used: this represents the information we need to convey at a level of abstraction that is appropriate to our particular task;
- the natural language generation techniques used: in order to structure the information for presentation, we use a variation on McKeown's [1985] schemas as a way of specifying discourse structure, but extended to produce hypertext documents; and
- the phrasal lexicon and surface realisation: to overcome problems of both speed and coverage, we extend an existing linguistic realisation component with the use of a phrasal lexicon.

Each design decision aims to bridge the gap between theory and real world applications, allowing the construction of a resource that has real practical value while still allowing scope for theoretically interesting developments in the field.

2 The PEBA-II Architecture

PEBA-II adopts a fairly traditional natural language generation system architecture in that it consists of two distinct components corresponding to two stages of the generation process: a text planning component which determines what content has to be expressed and how this is to be organised, and

Proceedings of the 19th Australasian Computer Science Conference, Melbourne, Australia, January 31–February 2 1996.

a linguistic realisation component that realises the parts of this text plan as natural language expressions. To these components we add a third which must ultimately play a role in any system which does more than generate disembodied texts: a document renderer, which carries out the work required to realise the generated text in some medium. The overall architecture of the system is shown in Figure 1.

Figure 1: A fragment of the knowledge base

The text planner begins with some communicative goal provided by the user and, taking account of the available linguistic resources and contextual constraints, produces a discourse plan that satisfies this goal. This discourse plan consists of a collection of individual sentence plans organised in a coherent fashion: Section 4.2 discusses this process in more detail and provides examples.

The linguistic realiser we use is Elhadad's [1992] FUF, combined with a small unification-based grammar of English we have developed for our domain. This takes the sentence plans that make up the discourse plan and realises each as an English natural language sentence. An important element of our approach here is the use of a phrasal lexicon. This realisation process and the role played by the phrasal lexicon is described in more detail in Section 5.

The document renderer in the current version of the system is any Web browser such as Mosaic or NetScape, but could equally well be some other component which translates document structuring commands into a visible form. In the current system the document structuring commands used are a subset of HyperText Markup Language (HTML).

The content of the generated texts is derived from a knowledge base of facts about animals, which in the current version of the system has been hand-constructed from an analysis of existing encyclopaedia articles about animals. The content of this knowledge base and the methodology adopted in constructing it are described in Section 3.

In operation, the user guides the system's processing by selecting hypertext tags which are used to indicate new discourse goals for the text planning component; each goal results in the generation of a Web page which contains a number of hypertext tags that correspond to a range of further discourse goals the user can choose to pose to the system; this results in an dynamic text planning enterprise where the user decides what information she would like to see on the next page generated.

3 Knowledge Representation

3.1 Choosing a Starting Point

An important methodological question in natural language generation is that of what the input to the process should be. Most existing work adopts one of the following solutions to this question:

- Use a representation which already exists for independent reasons: for example, we might want to generate text from an underlying representation used by an expert system or a CAD system.
- Choose a representation which is intended to correspond in some way to human mentalese; many approaches based on semantic primitives fall into this camp.
- Choose an input representation that makes semantic distinctions whose surface realisations the researcher wants to explore: an example of this would be the use of a notion of focus in the underlying representation in order to motivate choice between different sentential forms such as active and passive, or the use of pronominalisation.

To these we add a fourth alternative: we use an underlying representation which

- makes precisely those distinctions that are relevant for the range of texts we intend to generate; and
- is no more abstract than is required for the inference processes we need to perform over the representation.

This is essentially the same reasoning as adopted in interlingual approaches to machine translation; an argument for the adoption of this methodology in natural language generation is provided in Dale *et al* [1994]. This line of thinking is driven by the observation that many approaches to knowledge representation require extremely complex models which are very costly to construct. Such representations are necessary where their host systems need to be able to make arbitrary inferences, but the cost of constructing representations of this depth typically results in a loss of breadth of coverage. Our concern, on the other hand, is to provide as broad a coverage of the domain as is feasible, and perhaps ultimately to construct the requisite knowledge bases by semi-automated means. This is within the bounds of possibility because the range of inferences required of our system is relatively limited: the twin tasks PEBA-II addresses, of describing entities and comparing entities, do not require very sophisticated reasoning.

The most important outcome of this approach to the content of a knowledge representation is that the vocabulary of the representation language we use contains not only familiar elements for simple entities, properties and relations, but also higher-level semantic objects that correspond to precompiled constructions of these more atomic parts. Quite apart from the fact that this removes a great deal of unnecessary 'generation from first principles' and assists us in constructing a practical NLG system, it complements very neatly our use of a phrasal lexicon at the linguistic level, as described in Section 5.

3.2 The Taxonomic Knowledge Base

Our knowledge representation is essentially a semantic network of the Linnaean animal taxonomy whose principal nodes are animal classes and whose arcs, represented using ako links, indicate subset and superset relationships between these classes. An example fragment of the knowledge base hierarchy is shown graphically in Figure 2. Concepts in the knowledge base are paired with semantic and syntactic structures in a phrasal lexicon, described further in Section 5. As is generally the case in natural language generation systems, some of these concepts correspond to single words; however, many concepts correspond to entire phrases. This both simplifies the process of knowledge base construction, and at the same time provides a level of abstraction that is suitable for our purposes. Of course, choosing the right level of granularity is not always a simple task, because it requires some prediction of the degree of decomposition of elements that is likely to be required by as-yet-unforeseen requests to the generator.

The animal hierarchy allows us to infer relationships between animals and animal classes and to describe these. It also allows for inheritance of features so that, for example, we may assume that all the subtypes of the Mammal produce milk (unless they have some counter clause). The hierarchy forms the main backbone for hypertext generation, as will be seen later in Section 4.

Figure 2: An example knowledge base hierarchy.

Each node in the hierarchy serves as a location off which properties of the entity in question can be hung. There are two types of properties in the knowledge base. The distinguishing-characteristic (DC) clauses single out the important property that indicates how one subtype of a node is distinguished from others (and thus justifies the taxonomic distinction); for example, from Figure 2, the characteristic that distinguishes the Monotreme from all other Mammals is that it lays eggs. The hasprop clauses enumerate the known properties of an entity, as shown in Figure 3.

A substantial analysis of encyclopaedia articles about animals indicated that information in this domain tends to fall into a number of categories, such as *naming*, *physical properties*, *social behaviour*, *diet* and *lifespan*. Although these categories are not always mutually exclusive, they provide a useful tool for imposing structure on the vast quantity of information we are faced with. This analysis provides us with another taxonomy in which we encode relationships between properties: for example, physical properties may be size-related (e.g., *weight*) or body-parts (e.g., *nose*); body-parts may be internal (e.g., *skeleton*, *heart*) or external (e.g., *body-covering*, *limbs*), and so on. These relationships play a role in determining precisely what properties of entities can be used when the user requests that two animals be compared.

A fragment of the resulting knowlede base is shown in Figure 3. The entire knowledge base so far contains 1137 clauses describing 401 classes.

```
(hasprop Echidna
   (linean-classification Family))
(distinguishing-characteristic Echidna
    Monotreme (body-covering sharp-spines))
(hasprop Echidna
   (nose long-snout))
(hasprop Echidna
   (social-living-status lives-by-itself))
(hasprop Echidna
   (diet eats-ants-termites-earthworms))
(hasprop Echidna
   (activity-time active-at-dusk-dawn))
(hasprop Echidna
   (colouring browny-black-coat-paler-spines))
(hasprop Echidna
   (lifespan lifespan-50-years-captivity))
```

Figure 3: A fragment of the knowledge base

4 Text Planning and Hypertext

4.1 Generating Hypertext

Multimodal text generation seeks to integrate non-linguistic information, such as graphics, animation, sound or hypertext, into the communication process. A notable previous experiment of this kind is Reiter *et al*'s [1992] IDAS system, which dynamically generates on-line hypertext documentation of electronic equipment, tailored to the user's task type and experience. IDAS is designed to generate small pieces of text, relying on the user to guide the generation process using the hypertext facility, and relieving the system from having to reason more deeply about the user's needs. This idea is adopted wholesale in PEBA-II, except that the level of discourse planning at which the user interacts is such that we generate longer texts in each interaction. The full potential of this symbiosis between user and machine has yet to be explored.

Hypertext interfaces have penetrated many information presentation areas including the World Wide Web, on-line documentation, help systems, kiosk informational systems and CD-ROM encyclopaedias. Adding text generation to these environments promises substantial leverage for a variety of reasons:[1]

User Modeling: Automatic text generation can tailor hypertext documents to the user's knowledge, task type or the current context. Apart from tailoring the textual content itself, another form of tailoring involves offering links to related areas that might be of interest to a particular user.

Source Material Construction Cost: Any documentation, whether hypertext or not, is time consuming to produce. Whenever the same information is likely to be reused again and presented in different forms, system construction effort can be reduced substantially by constructing one knowledge base in combination with natural language techniques to realise this information in different ways. For example, in our domain, we cannot determine ahead of time which two animals the user might like to compare. To store every possible comparison of 100 animals would require 4950 separate documents to be written, a task which is probably not worth considering; however, by making use of an underlying knowledge base that contains information on each of the 100 animals, NLG techniques allow any one of these comparisons to be generated on demand.

Maintenance: Updating our knowledge base is simpler than maintaining pre-prepared texts, particularly the comparative texts. A change to the knowledge base propagates to all those documents about an entity. The grain-size of our knowledge representation means that it is also very cheap to add quantities of information, as will be seen in Section 5: the more abstract nature of most conventional approaches to knowledge representation results in more time-consuming knowledge base construction.

Discourse History: We can establish a discourse history, not only of the generated text, but also from the visited sites for a hypertext network. Such a strategy allows the text generated to be based on what the user has previously seen.

Interestingly, the benefits of marrying natural language generation and hypertext do not accrue to only one party to the relationship: using hypertext can add many benefits to text generation too.

Structure: Using hypertext imposes some (navigational) structure on text, even on unstructured information.

Discourse planning: The hypertext interface allows the user to perform high-level discourse planning and allows for generating less text than we would normally without this facility. This interactive capability begins to break down the distinction between monologue and dialogue in text generation.

Related information: We can provide links to more detail, elaborations or related information without actually generating the text unless the user wants to see it.

Browsing capability: The user can browse if she is unsure exactly what she wants to know. We don't need a query interface with the attendant problems of robust natural language analysis; it is

[1]Many of these advantages have been noted in previous literature: see in particular Reiter *et al* [1992] and Moore [1995].

obvious what the user may find out more about, so there is less likelihood of user frustration.

Using the World Wide Web as the basis for hypertextual generation provides a number of additional capabilities that demonstrate the usefulness of the technology:

Audience: Providing documentation on the WWW allows for as wide an audience as one would like.

Interface: The WWW viewers provide a flexible interface, so the text generation system doesn't require a separate display module to be developed.

Multi-modal Display: The interface has a built-in multimodal display.

Easy Navigation: The user can go down a path searching for information and can always return back to base. So, for instance, if the user selects a hypertext link on one page, she can always come back and select on any other link on that same page to follow a different path.

4.2 Using Schemas

In broad terms, the process of text planning involves taking some (often pre-determined) collection of informational elements to be presented to a user, and imposing upon this set of elements a structure which provides the resulting text with fluency and cohesion. Within the literature on text planning, there are two main approaches to this task: schema-based approaches which, following the work of McKeown [1985], use what are effectively discourse grammars to produce texts that meet predefined and empirically recurring patterns; and approaches based on Rhetorical Structure Theory (sc rst; Mann and Thompson [1987]), which attempt to build a text dynamically using planning operators that specify how fragments of a text can be pieced together in a coherent fashion (Hovy's [1991] work gives a good example of this approach).

The range of texts we are interested in generating is sufficiently invariant that the schema-based approach makes most sense.[2] Schemas essentially provide paragraph templates of pre-defined structure, content and order: for example, we can formulate a standard way to describe an animal which includes giving information about its name and taxonomy, distinguishing features, habitat, size and weight, followed perhaps by an example. In the general case such techniques are too rigid for fluent text production; however, some variation comes from the differing kinds of information available on any given animal, and the remaining elements of uniformity themselves have some value in an instructional context.

[2]We do, however, intend to use our domain as a way of exploring the idea that schemas are effectively compiled collections of RST relations: this claim has been made several times in the literature but never adequately explored.

McKeown's [1985] work describes four schemas which she saw as useful for describing information in a naval database; these she called **Identification**, **Constituency**, **Attributive** and **Compare and Contrast**. Each schema provides a set of ordering constraints over a pattern of RHETORICAL PREDICATES in such a way that the resulting text is fluent and coherent; each rhetorical predicate is effectively a representation of a speech act type, defined in such a way as to provide an interface to the underlying knowledge representation. Thus, for example, McKeown's **Evidence** rhetorical predicate knows how to find information in the database that would constitute evidence.

From the outset, our aim has been to produce hypertext documents; this, and the slightly different characteristics of our domain, mean that the schemas we need are a little different from those used in McKeown's work. For our purposes, two schemas—which we call **Identify** and **Compare and Contrast**—suffice; our **Identify** schema, since it produces a hypertext document from which the user can request further detail, effectively conflates McKeown's **Identification** and **Constituency** schemas.

4.2.1 The Compare and Contrast Schema

The animal **Compare and Contrast** schema is represented by the following grammar rules,where each terminal symbol in the grammar corresponds to a rhetorical predicate:

$$\begin{aligned}&\mathsf{CompareAndContrast} \longrightarrow\\ &\quad \mathsf{LinnaeanRelationship\ CompareProperties}\\ &\mathsf{CompareProperties} \longrightarrow\\ &\quad \mathsf{CompareProperty\ CompareProperties}\\ &\mathsf{CompareProperties} \longrightarrow \phi\end{aligned}$$

This grammar is implemented within the system as an augmented transition network. The schema first identifies how two animals are related in the animal taxonomy, and then proceeds to compare their properties. In the schema, we use the underlying categorisation of properties into topics to permit appropriate comparisons to be drawn; a taxonomy of properties allows us to determine that, for example, **height** and **length** are both measurements of size and so can be usefully mentioned together.

The **LinnaeanRelationship** rhetorical predicate generates how the animals are related according to the Linnaean animal taxonomy. This relationship is determined by traversing the hierarchy upwards from each animal until a common ancestor is found. The subtypes of this common ancestor to which the animals belong form the main basis for generating the relationship.

Using corpus-based categories, the **CompareProperty** rhetorical predicate searches for related properties for the selected animals. As discussed earlier, these categories were defined based on the sorts of properties which are typically used to describe animals in encyclopedic texts.

Figure 6 shows a Web page created by the **Compare and Contrast** schema. The underlined entities are clickable hypertext which indicate new discourse goals for the text generation system.

The text plan underlying the Web page shown in Figure 6 is shown in Figure 4. The plan consists of a sequence of speech act specifications, each of which specifies the propositional content of the speech act in terms of a set of semantic elements. Each speech act specification is then used to construct a sentence plan, from which a surface sentence is produced, as described in Section 5.

4.2.2 The Identify Schema

The animal identification schema, **Identify**, is expressed as an augmented transition network in Figure 5. It essentially dictates that, in order to identify a particular node in our taxonomy, we first provide some information about naming; then mention each of the subtypes; and then list known properties of the entity in question.

Figure 5: The Identify Schema.

Name-Entity, **Name-Subtype** and **Describe-Property** are rhetorical predicates that know how to locate the appropriate information to satisfy these goals, and then use this to construct input specification for the realisation component.[3]

The **Name-Entity** rhetorical predicate generates the naming information for an animal. **Name-Subtype** outputs any subtypes for the current node in hypertext so that the user may request definitions of these. **Describe-Property** generates a sentence for each property of the current node. It currently uses a simple pronominalisation algorithm to avoid repeating the animal's name for each sentence. Unlike the **Name-Entity** predicate, the **Name-Subtype** and **Describe-Property** predicates are optional, since the node being described may be a leaf and/or may not have any properties assigned to it.

Figure 7 shows the World Wide Web page created by this schema for the Echidna. Again, the underlined words in these texts are clickable hypertext queries which will invoke the construction of a new page that provides further information about the animal in question, thus allowing the user to interactively interrogate the taxonomy. The user may traverse up the Linnaean animal hierarchy by clicking on the supertype (the Monotreme in this example), and traverse down via the subtypes (the short and long-beaked Echidnas here).

[3]The rhetorical predicates are roughly similar to McKeown's **Identification**, **Constituency** and **Attributive** predicates, but sufficiently different that we have chosen distinct names.

4.3 Variations for Naive and Expert Users

The construction of these texts is parameterised by a simple user model that distinguishes between naive and expert users. This is currently implemented automatically in two ways: via animal naming protocols, and the notion of the Linnaean animal taxonomy. Each animal class (or node) always has a Linnaean name attached to it, often has a true name and sometimes a common name. In each case, different conventions are used for naming animals, thus motivating the difference between the following examples:

1. The Peludo, also known as the six-banded Armadillo, is a type of Armadillo which has six flexible bands.

2. Euphractus sexcinctus, also known as the Peludo, is a member of the Euphractus Genus which has six flexible bands.

In (1) above, generated for the novice user, the Linnaean name for a node would never be used unless there is no true name. The text in (2) was produced for the expert user employing scientific Linnaean naming.

Note that in the description produced by the **Identify** schema, information is provided about the superordinate and subordinate nodes in the taxonomy. The user model plays a role here too, since what is considered to be the superordinate or subordinate of a given node depends on who the reader is: for an expert audience, we use the full Linnaean taxonomy, whereas for non-experts we use a more restricted taxonomy which ignores some of the more technical distinctions; thus, (1) above indicates that the Peludo is a member of the Armadillo family, ignoring the intermediate genus mentioned in (2); the list of subtypes provided will also be different in each case. This non-expert taxonomy is derived automatically by examining the topology of the Linnaean taxonomy. Each WWW page indicates the current user level and allows the user to switch between them.

The system also makes use of a simple discourse history to constrain subsequent output; at the moment this is limited to the use of a simple pronominalisation algorithm, although we intend to inte-

```
((schema-type identify)
 (constituents
     ~(((speech-act-type name-entity)
        (content ((primary-name ((cat np) (sem echidna) (name-type name)))
                  (secondary-name
                        ((cat np) (sem echidna) (name-type common-name)))
                  (supertype ((cat np) (sem monotreme) (name-type name)))
                  (relationship ((sem is-a-type-of)))
                  (distinguishing-characteristic
                        ((cat vp) (sem sharp-spines))))))
       ((speech-act-type list-subtypes)
        (content ((head ((cat np) (sem echidna) (name-type name)))
                  (arguments
                   ~(((cat np) (sem short-beaked-echidna) (name-type name))
                     ((cat np) (sem long-beaked-echidna)
                               (name-type name)))))))
       ((speech-act-type describe-property)
        (content ((name ((cat np) (sem echidna) (name-type name)))
                  (property ((cat vp)
                             (sem lifespan-50-years-captivity))))))
       ((speech-act-type describe-property)
        (content ((name ((cat np) (sem echidna) (name-type name)))
                  (property ((cat vp)
                         (sem browny-black-coat-paler-coloured-spines))))))
       ((speech-act-type describe-property)
        (content ((name ((cat np) (sem echidna) (name-type name)))
                  (property ((cat vp) (sem active-at-dusk-dawn))))))
       ((speech-act-type describe-property)
        (content ((name ((cat np) (sem echidna) (name-type name)))
                  (property ((cat vp)
                             (sem eats-ants-termites-earthworms))))))
       ((speech-act-type describe-property)
        (content ((name ((cat np) (sem echidna) (name-type name)))
                  (property ((cat vp) (sem lives-by-itself))))))
       ((speech-act-type describe-property)
        (content ((name ((cat np) (sem echidna) (name-type name)))
                  (property ((cat vp) (sem long-snout))))))))
```

Figure 4: The discourse plan for a Compare and Contrast page.

grate more sophisticated models of referring expression generation (see Dale [1992] and Dale and Reiter [1995]). In future work we intend to take advantage of the hypertextual mode of presentation to display suitably modified text on subsequent requests for the description of a given node. If the user has visited a particular node in the hierarchy, then we might describe an animal by comparing it to other known animals or by using concepts with which the user is familiar. For example, if a child has been told about the dog, and subsequently asks about the cat, we might actually choose to describe the cat by comparing it to the concepts the child knows about the dog (see [Milosavljevic 1996]).

5 Surface Realisation and Phrasal Lexica

5.1 Sentence Plans

As indicated above, the discourse plan consists of a series of speech act specifications. Each speech act specification is then used as input to a sentence planning process, which determines how to arrange this information in a sentence. A typical sentence plan is shown below:

```
((cat s)
  (sem ((proc is-a-kind-of)
        (carrier ((sem echidna)
                  (cat np)
                  (head ((cat n)
                         (lex ``Echidna'')) 
                  (det ((lex ``the''))))
        (domain ((sem monotreme)
                 (cat np)
                 (lex ``Monotreme''))))))
```

This specifies the semantic content to be realised in a sentence, and provides some directives to the realisation component as to how this should proceed: so, the gross syntactic strucure is already specified, as are some of the lexical items. This mixing of levels is deliberate: in line with much current thinking about the interactions between the stages of the generation process, our aim is to move towards a blackboard-like architecture where each stage of the system can impose constraints on the output, thus moving away from the simpler two-stage architecture we currently use.

Realisation in FUF proceeds by a process of unifying the grammar against the input sentence specification, followed by a linearisation phase that walks around the fully specified semantic–syntactic structure to produce a sequence of words complete with appropriate morphology.

5.2 Using Phrasal Lexica

A major part of the natural language generation task is mapping from knowledge base entities or semantic elements to surface syntactic and lexical phenomena. This mapping is often thought of as going from atoms in one vocabulary to atoms in another vocabulary; relational elements in the semantic representation map to syntactic frames, and arguments to the fillers of slots in those frames.

We take the view that knowledge abstraction is often more laborious than it needs to be, and that the atoms should often be much larger chunks. Mirroring our decision to use complex knowledge elements in our knowledge representation, it turns out to be convenient to also use multi-word structures in our lexicon which provide realisations of these properties. The lexical entry for the verb phrase corresponding to the semantic element 30cm-long is as follows:

$$\left[\begin{array}{ll} \text{sem:} & \text{30cm-long} \\ \text{orth:} & \text{"is 30cm long"} \\ \text{syn:} & \left[\begin{array}{ll} \text{cat:} & \text{vp} \\ \text{agr:} & \left[\text{number: singular}\right] \end{array}\right] \end{array}\right]$$

Note here that all we have done is draw a direct correspondence between a semantic chunk from the knowledge base and the very specific linguistic chunk that is used to realise it. This representation also permits some factoring of information to allow number agreement to be realised appropriately; in the following example, the number of the verb will be derived from the number of the subject noun phrase, but we can still directly store the predicate as an orthographic string:

$$\left[\begin{array}{ll} \text{sem:} & \text{30cm-long} \\ \text{syn:} & \left[\begin{array}{ll} \text{cat:} & \text{vp} \\ \text{agr:} & \left[\text{number: Number}\right] \\ \text{verb:} & \left[\begin{array}{ll} \text{syn:} & \left[\begin{array}{ll} \text{cat:} & \text{v} \\ \text{lex:} & \text{"be"} \\ \text{number:} & \text{Number} \end{array}\right] \end{array}\right] \\ \text{object:} & \left[\begin{array}{ll} \text{syn:} & \left[\text{cat: np}\right] \\ \text{orth:} & \text{"30cm long"} \end{array}\right] \end{array}\right] \end{array}\right]$$

In PEBA-II, knowledge base facts are used for two reasons: to describe animals and to compare animals. A property is only decomposed if the elements of that decomposition are likely to be required to be for a particular generation purpose; otherwise, it is correlated directly with a phrase in the lexicon. For example, we might need to decompose the content of the sentence *The Echidna has a long snout and sharp spines* into two separate properties so that we can compare its *nose* and *body covering* (using the corpus-based categories described earlier) separately with those of other animals. However, for our needs, we do not need to decompose the content of *The Apar can roll itself into a ball for defense*, since we only need to know that this property is a defense mechanism (*predator-behaviour*) which we can compare directly with that of other animals; this means we can often avoid very tricky representational questions.

We are exploring other kinds of abstractions that provide broader coverage in this way; in particular, we are interested in determining what kinds of abstractions are required in order for the generator to make appropriate use of ellipsis, conjunction, and nominalisation.

The use of phrasal lexical items of this kind has a two specific advantages:

Reuse and Efficiency: If we repeatedly realise a semantic element in the same way, it is better to remember this and avoid rebuilding the surface form each time.

Idioms: We have no alternative but to store complete chunks of this kind where a given phrase, such as *kick the bucket*, does not have a compositional semantics.

In effect, we are applying the arguments first put forward by Becker for the use of a lexicon that contains phrases as well as atomic lexical elements [Becker 1975]. Some of these ideas have been explored in computational models by Kukich [1983] and Hovy [1988], although to our knowledge no previous work uses observations within a unification-based approach to linguistic realisation. Our approach to both knowledge representation and surface realisation is thus driven by the same minimalist principle: only decompose the knowledge representation and construct surface forms from atomic elements when necessary. Note that this is not just something we can do when the lexicon is constructed: we can also dynamically cache such structures as they are built from first principles during the generation process. We are pursuing this idea by integrating the idea of a 'persistent chart' into an existing linguistic realisation component to produce a more efficient generator (see Tulloch and Dale [1995]).

An additional argument for the use of such chunkings of linguistic material is that they may be easier to mine automatically from existing textual resources. The feasibility of this remains to be determined by future work.

6 Conclusions and Future Work

In this paper, we have presented PEBA-II, a text generation system that dynamically generates descriptions of animals in an interactive hypertext environment. A major advantage gained by embedding text generation within a hypertext environment is that it allows the user to perform high-level discourse planning and thus reduces some of the burden on the text planner. PEBA-II further introduces user modeling into the WWW hypertext environment, allowing the production of different texts for different users. The use of a phrasal lexicon also allows us to make precisely those distinctions that are necessary to generate descriptions and comparisons of animals.

In future work, we intend to focus on the following issues:

- the integration of RST-style text planning, to allow a proper comparison of the two main approaches to text planning;
- further extensions of the use of the phrasal lexicon in order to determine the most useful abstractions to make use of in such a mechanism;
- automatic construction of the phrasal lexicon from existing encyclopedia articles about animals;
- more sophisticated user modeling, integrated with discourse modeling in such a way as to explore the issues that arise in hypertext generation; and
- the addition of elements of pictures and other graphical devices to produce a fully multimodal generation system.

Each of these directions provides scope for taking natural language generation techniques from the laboratory into practical applications.

Acknowledgements

We would like to thank Mark Dras and Mark Lauer of the NLP Unit at the Microsoft Institute for many discussions related to this work, and Chris Mellish and Ehud Reiter for motivating much of our initial thinking on the construction of practical NLG systems.

References

Becker J.D. [1975] The Phrasal Lexicon. In *Proceedings of the Conference on Theoretical Issues in Natural Language Processing*, Cambridge, MA, pp. 70–77.

Dale, R [1992] *Generating Referring Expressions.* MIT Press, Cambridge, MA.

Dale R., Finkler W., Kittredge R., Lenke N., Neumann G., Peters C. and Stede M. [1994] Lexicalisation and Architecture. In Hoeppner W., Horacek H. and Moore J. (eds), *Principles of Natural Language Generation*, Dagstuhl Seminar Report, pp. 30–38.

Dale, R and Reiter, E [1995] Computational Interpretations of the Gricean Maxims in the Generation of Referring Expressions. *Cognitive Science*, **19**(2), pp. 233–263.

Elhadad M. [1992] *Using Argumentation to Control Lexical Choice: A Functional Unification Implementation.* PhD Thesis, Columbia University.

Hovy E.H. [1991] Approaches to the Planning of Coherent Text. In Paris C.L., Swartout W.R. and Mann W.C. (eds), *Natural Language in Artificial Intelligence and Computational Linguistics*, Kluwer, Boston, pp. 83–102.

Hovy E.H. [1988] Generating Language with a Phrasal Lexicon. In McDonald D.D. and Bolc L. (eds), *Natural Language Generation Systems*, Springer-Verlag, New York, pp. 353–384.

Kukich K. [1983] Design of a Knowledge-Based Report Generator. In *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, Cambridge, Mass.

McKeown, M. [1985] *Text Generation.* Cambridge: Cambridge University Press.

Mann W. and Thompson S. [1987] *Rhetorical Structure Theory: A Theory of Text Organisation.* Technical Report RS–87–190, USC/Information Sciences Institute, Marina Del Rey, Ca.

Moore, J [1995] *Participating in Explanatory Dialogues.* MIT Press, Cambridge, MA.

Milosavljevic M. [1996] Introducing New Concepts Via Comparison: A New Look at User Modeling in Text Generation. In *Proceedings of the Fifth International Conference on User Modelling, Doctoral Consortium*, Hawaii.

Reiter E., Mellish C. and Levine J. [1992] Automatic Generation of On-Line Documentation in the IDAS Project. In *Proceedings of the Third Conference on Applied Natural Language Processing*, Trento, Italy, pp. 64–71.

Tulloch, A and Dale, R. [1995] Speeding up Linguistic Realisation Using a Cache. Poster at *the Eighth Australian Joint Conference on Artificial Intelligence (AI'95)*, Canberra, Australia.

Figure 6: A Compare and Contrast Schema WWW Page.

Figure 7: An Identification Schema WWW Page.