User Modeling and User-Adapted Interaction (2005) 15: 135–168 © Springer 2005
DOI 10.1007/s11257-004-5637-6

Tailoring Automatically Generated Hypertext

KALINA BONTCHEVA and YORICK WILKS
Department of Computer Science, University of Sheffield, 211 Portobello Street, Sheffield S1 4DP, UK. e-mail: {kalina}@dcs.shef.ac.uk

(Received: 31 October 2003; accepted in revised form 15 October 2004)

Abstract. This paper describes an approach for tailoring the content and structure of automatically generated hypertext. The implemented system HYLITE+ is based on applied Natural Language Generation (NLG) techniques, a re-usable user modelling component (VIEWGEN), and a flexible architecture with module feedback. The user modelling component is used by the language generation modules to adapt the hypertext content and links to user beliefs and preferences and to the previous interaction. Unlike previous adaptive NLG systems, which have their own, application-specific user models, HYLITE+ has re-used a generic agent modelling framework (VIEWGEN) instead. Apart from avoiding the development costs of a new model, this also enabled a more extendable system architecture. Another distinct feature of our approach is making NLG techniques adaptable by the user, i.e., providing users with control over the user model and the hypertext adaptivity.

Key words. adaptive language generation system, dynamic hypertext, hypertext, user modelling

1. Introduction

When people talk to each other or write texts, they constantly adapt their language and information content to make themselves understandable, relevant, and interesting. Some of the main factors influencing these choices are the context and medium of communication, the goals they are trying to achieve, and also who their listeners or readers are.

Similarly, computer conversational agents and text generation systems need to provide user adaptivity. In the terms adopted here, they need to maintain a model of the user and be able to adapt their behaviour according to the state of this model.

Research in user and belief modelling has led to a wide range of approaches for modelling and reasoning with user-related information. Some of the most powerful models (e.g., Ballim and Wilks, 1991; Kobsa and Pohl, 1995) can represent agents' beliefs, goals, intentions and other propositional attitudes, including beliefs about other agents' beliefs (e.g., a parent's belief that her child believes Santa Claus is real). Such nested models have already been used in language understanding (e.g., Kobsa, 1988; Taylor et al., 1996), Natural Language Generation (NLG) (e.g., Paris, 1993; Zukerman and McConachy, 1995; Moore, 1995), and some adaptive

hypertext systems (e.g., Ardissono and Goy, 1999; Kobsa et al., 1997; Brusilovsky, 2001).

More specifically, in the area of NLG, researchers have investigated ways to tailor automatically-generated texts to user goals and characteristics (e.g., Paris, 1993; Zukerman and McConachy, 1995). The bulk of this work has focused on tailoring generated text, but more recently NLG methods have been applied to the problem of generating *dynamic hypertext* (Dale et al., 1998). In *dynamic hypertext*, page content and links are created on demand from some formal knowledge representation using NLG techniques. The generated hypertext is often adapted to user preferences and characteristics and the interaction context. The key difference from non-NLG-based adaptive hypertext systems (e.g., AHA! De Bra and Calvi, 1998, ELM-ART Brusilovsky et al., 1996, KN-AHS Kobsa et al., 1997) is that there are no pre-existing, human-authored hypertext documents or text snippets, instead both the content and the links are generated on the fly. In contrast, the choices in adaptive hypertext systems are somewhat limited by the pre-authored text content. For further details on adaptive hypertext systems and approaches see (Brusilovsky, 1996, 2001).

Previous dynamic hypertext systems, such as ILEX (Knott et al., 1996) and PEBA-II (Milosavljevic et al., 1996), have implemented their own, application-specific user models. One of the novel aspects of this work is in the choice to reuse a generic agent modelling framework (VIEWGEN) instead (Ballim and Wilks, 1991). Apart from avoiding the development costs of a new model, this also enabled the creation of a more extendable system architecture (Bontcheva and Wilks, 2001), which supports easier porting to new domains and applications (e.g., report generation from Semantic Web ontologies (Bontcheva and Wilks, 2004), intelligent knowledge management[1]). As argued by Brusilovsky Brusilovsky (2001), such modularity and re-use are much desired in adaptive hypertext systems and one way of achieving that is by using generic user/learner models, such as BGP-MS (Kobsa and Pohl, 1995) and VIEWGEN.

Another distinct feature of our approach is the use of results from hypertext usability studies, user trials with similar software products, and walkthroughs during system design (see Bontcheva, 2001). This led to making HYLITE+ adaptable, in addition to being adaptive, thus becoming the first dynamic hypertext system to provide users with control over the NLG modules.

This paper focuses on the user modelling approach and its role in generating dynamic hypertext. We first introduce the HYLITE+ architecture (Section 2). The role of the user modelling framework in the generation process is presented in Section 3. Section 4 focuses on hypertext adaptivity, both in terms of content and link personalisation. Adaptability is detailed in Section 5, followed by Section 6 which describes the system evaluation. The paper concludes by positioning HYLITE+ with

[1]For information on the SEKT project see http://gate.ac.uk/projects/sekt/.

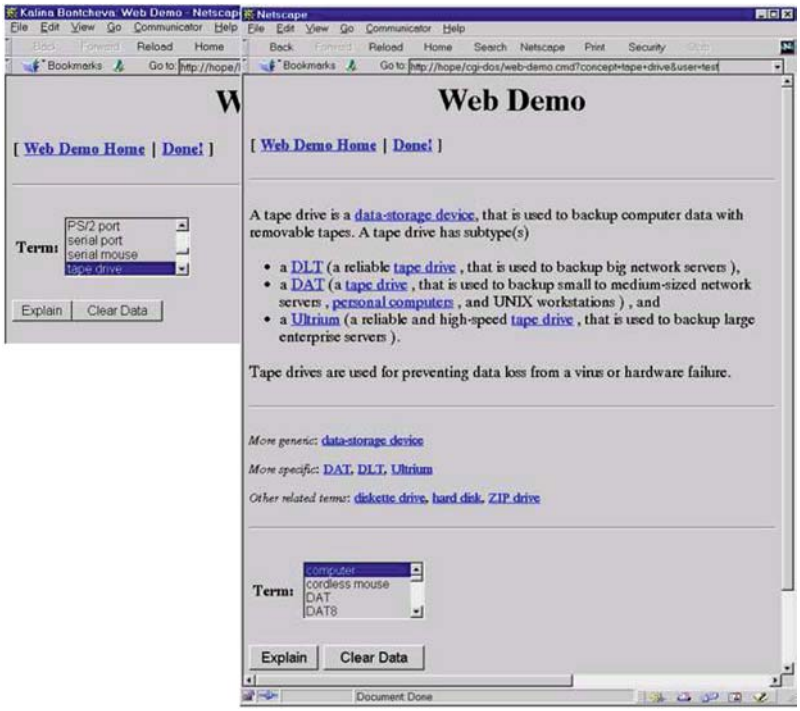

Figure 1. The system home page (left) and an explanation generated by the adaptive system (right). The different options of the user model can be modified by the user via hypertext links provided at the bottom of the generated text (not visible here).

respect to some relevant previous work in NLG and adaptive hypertext (Section 7) and discussing some future directions.

2. System Overview

HYLITE+ is a *dynamic hypertext system* that generates encyclopaedia-style explanations of terms in two specialised domains: chemistry and computers. The user interacts with the system in an ordinary Web browser (e.g., Netscape, Internet Explorer) by specifying a term they want to look up. The system generates a hypertext explanation of the term; further information can be obtained by following hypertext links or specifying another query (see Figure 1).

Since users expect real-time interaction when browsing hypertext, efficient and robust applied NLG techniques are typically used for dynamic hypertext generation. For instance, ILEX (Knott et al., 1996) uses a combination of canned stories and templates; EXEMPLARS (White, 1998) is rule-based; and PEBA (Milosavljevic et al., 1996) uses text schemas (McKeown, 1985) and a phrasal lexicon. The input to these systems is typically a domain knowledge base or a database, representing objects and relationships between them.

Similar to these systems, the NLG architecture of HYLITE+ consists of NLG modules organised in two main stages: (i) content organisation, which includes modules that select the semantic content of the text and organise it in a coherent way; and (ii) *surface realisation* modules that generate sentences from the semantic representation (i.e., conceptual graphs).

The system uses an existing text organisation module (Bontcheva, 1997), which is based on high-level discourse patterns similar to text schemas (McKeown, 1985). The organisational patterns effectively impose an ordering on the selected propositions, based on their relations and concepts. Although schema-based text organisation has some known limitations (see Moore and Paris, 1993), it was chosen for its computational efficiency and because it has been shown to work well for generating encyclopaedia-style explanations (Paris, 1993).

The most frequently used organisational pattern is for *entity descriptions* (i.e., concepts inheriting from ENTITY in the hierarchy) and has some similarities with the identification schema in (McKeown, 1985) and the constituency schema in (Paris, 1993). In HYLITE+ entities are defined by their *supertype(s) or type definitions, examples, characteristics, constituents* and *processes/events*. If the entity has several synonymous terms, the term from the user query is used throughout the explanation and the rest are given in brackets when the entity is first introduced.

HYLITE+ uses the EGEN surface realiser (Bontcheva, 1997), adapted from German to English. The realisation component receives as an input the text plan which has pointers to the propositions that have to be verbalised. The output is a textual explanation in HTML format which is displayed by the Web browser.

3. The Role of the User Model

Unlike previous adaptive NLG systems which have their own, application-specific user models, our dynamic hypertext system has re-used a generic agent modelling framework (VIEWGEN) instead (Ballim and Wilks, 1991).

In terms of modelling user beliefs VIEWGEN is somewhat similar to extended overlay models (see Holt et al., 1993), because it represents both user beliefs that overlap with the system's own beliefs and also wrongly believed facts. However, the nested belief models in VIEWGEN are most similar to the approach in BGP-MS (Kobsa and Pohl, 1995). The main difference between VIEWGEN and BGP-MS is that the former has no mutual belief operator and instead uses a belief ascription mechanism.[2] While, in principle, arbitrary levels of nesting could lead to complex reasoning and lower efficiency in comparison to simpler models (e.g., extended overlay), in practice it has been shown that only two levels of nesting are typically sufficient to model beliefs in multi-agent systems (Taylor et al., 1996).

From an application's point of view, the main benefit from using VIEWGEN comes from its reasoning mechanisms for update and maintenance of the user

[2]For further comparison see (Ballim, 1992).

model. For example, the topic environments provide an easy way to access all propositions relevant to a given topic.

Finally, VIEWGEN's nested structures also support the storage of users' individual preferences as part of their environments, in the same way as their beliefs. The pros and cons of this approach are discussed further in Section 7.

In this section we will detail the role of the user modelling framework in the generation process and its relation to the domain knowledge and the discourse history.

3.1. THE USER MODEL AND THE DOMAIN KNOWLEDGE BASE

The user model and the domain knowledge base are the basis for the generated adaptive hypertext explanations. HYLITE+ uses the VIEWGEN user modelling framework to access and update user beliefs as the interaction progresses. The propositions believed by each user are encoded as conceptual graphs[3] (CG) (Sowa, 1984) and the CG reasoning mechanisms are used to detect whether or not the user already holds a given belief. Each CG has a unique identifier which is used to reference it from the VIEWGEN belief environments. The CGs are stored only once – in the knowledge base itself.

VIEWGEN distinguishes the beliefs of each user by putting them in separate environments, including one for the system itself which contains the domain ontology. The motivation for modelling domain facts as system beliefs is that the system can treat them in the same manner as user beliefs and also deal with incomplete domain knowledge (i.e., assume that the system's knowledge is potentially incomplete).

The separation between system and user belief environments also enables the modelling of discrepancies between propositions believed by the system and those believed by the user. One example of such discrepancies is taxonomic knowledge, where the system has detailed domain taxonomy, while the user has only a partial and/or incorrect representation. Such discrepancies might come from common misconceptions encoded as stereotypes or be specific to the given user, in which case they are stored exclusively in their belief space. Misconceptions from stereotypes are ascribed dynamically to all users who conform to that stereotype.

In this way, differences between the domain ontologies of different agents/users are easily encoded (see Figure 2). VIEWGEN's operations take such differences into account and can reason correctly about the user's beliefs when these are affected by differences between their and the system's ontologies. For example, when determining the beliefs of a chemistry expert about photographic emulsion (Figure 2 left), VIEWGEN also considers propositions inherited from gels and sols. By default

[3]Conceptual graphs are a type of semantic network. In the notation used here concepts are written in square brackets and relations in round ones. Extra information (e.g., whether a concept is familiar to the user, as determined on the basis of the user model) can be associated with concepts, relations or entire graphs using feature structures.

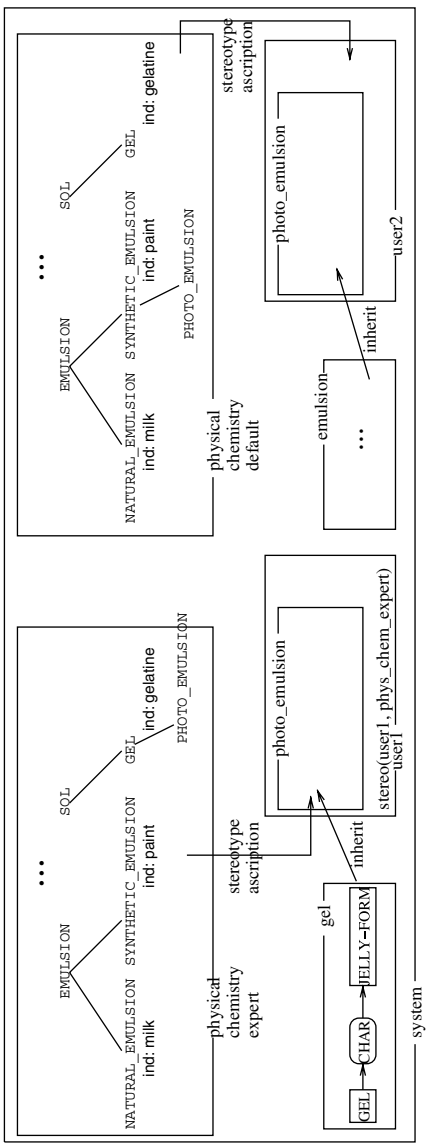

Figure 2. An example of representation of conflicting information. There are two stereotypes (top half): one for chemistry experts and one for laymen. There are also two belief environments: one for `user1` (chemistry expert) and one for `user2` (layman). The rectangles with names like gel and emulsion exemplify VIEWGEN topic environments and some of their context.

if the user has only general knowledge about a concept, VIEWGEN takes into account the default stereotype and includes beliefs inherited from emulsion instead (Figure 2, right).[4] In general, because VIEWGEN determines the user's beliefs separately for each concept, it is possible to inherit expert beliefs on some concepts (e.g., computer-related ones) while also inheriting layman beliefs on others (e.g., chemistry-related ones).

3.2. INTERFACE TO THE USER MODELLING FRAMEWORK

The communication between VIEWGEN and the generation modules is carried out using a well-defined interface consisting of a small set of predicates. The interface can easily be mapped to a client-server one, so the user modelling framework can run on a separate server and/or be replaced with a different user modelling component. The main operations are:

- `getPropositions` – returns all propositions believed by the given user (can be restricted to a given topic (e.g., microcomputer)).
- `getTopics` – returns all topics about which a given user has beliefs.
- `setPropositions` – adds the given propositions to the user model.
- `isBelieved` – true when the user believes the given proposition. It returns false if the proposition is not in the user model. Disbelieved propositions are not handled at present.

As is evident from these predicates, the generator has the main initiative in the communication with the user model, including the decisions as to when and what to add to VIEWGEN. However, VIEWGEN independently performs a number of tasks needed for maintaining consistency of the model, transparently to the application (e.g., checking for duplicate or subsuming beliefs).

The use of the various VIEWGEN operations (e.g., use of stereotypes, belief update) is also controlled by the application. For example, if stereotypes are switched off, `getPropositions` only returns the propositions already present in the user's belief space but not those which can be ascribed from stereotypes. In this way, the application not only controls when to use and update the user model but can also configure to an extent the operation of VIEWGEN itself. The available configuration parameters are shown in Table I with their default values. These parameters are used by HYLITE+ to support adaptability (see Section 5), i.e., allow users to control what personal information is stored and how it is used by the system.

3.3. DISCOURSE HISTORY AND THE USER MODEL

The other module that plays an important role in the generation of personalised hypertext explanations is the *discourse history*, which records information about the

[4]The reason for that is because laymen tend to think that photographic emulsion is an emulsion due to the occurrence of the word 'emulsion' in the term.

Table I. VIEWGEN configuration parameters

Parameter	Default value	Purpose
`consultUserBeliefs`	true	If true, HYLITE+ accesses the propositions in the user's belief space in order to adapt the generated hypertext. If false, then HYLITE+ operates without a user model.
`updateUserBeliefs`	true	If set to false, VIEWGEN does not record or update user beliefs; however, existing ones will still be used if `consultUserBeliefs` is enabled. This parameter, in conjunction with `consultUserBeliefs`, allows the user to switch off VIEWGEN.
`stereotypesUsed`	true	Stereotypes model the beliefs of classes of users (e.g., chemistry experts) and, when this variable is set to `true`, applicable stereotypes are consulted to derive additional user beliefs (see Section 3.4).
`inheritanceUsed`	true	Inheritance (when enabled) is used to present information inherited from a parent concept in the domain taxonomy (see also Section 5).

user–system interaction. The discourse history is maintained and updated independently of VIEWGEN. Clearly in our implementation there is some overlap in the information in the Discourse History (DH) and the belief environments, e.g., the facts explained so far. However, the purpose for which each of these sources is used differs: the DH contains the interaction information and is used in linguistic and formatting choices, whereas the agent environments contain only the set of propositions believed by the user and are used primarily to tailor the explanation content. The main role of the DH is to allow the system to have a notion of time, e.g., to obtain the number of pages since a given fact was presented to the user.

In addition, the DH also contains application-specific data such as whether the user reached an explanation through a hypertext link or a search, and how long they spent reading each explanation. This data is undoubtedly useful for a hypertext generation system but is of limited value to any other application, so it is only stored locally in the DH. Nevertheless, inferences about user beliefs can be made on the basis of this data and stored in the relevant part of VIEWGEN.

In the user modelling field, the complex relationship between DH and user models has been the subject of several debates (e.g., Schuster et al., 1988; Moore, 1995). The view taken here is similar to Wahlster's argument in (Schuster et al.,

1988), namely that the two components should be kept separate despite the fact that they process similar input and are used by the same application. The overlapping information represented separately in both these resources also reduces the dependence between the NLG application and the user model.

In common with other interactive NLG systems (e.g., Moore, 1995), the DH stores information about *user utterances* (in our case requests for explanation on a given topic), *focused objects*, and *text plans* of all system explanations. The text of the generated explanation is not preserved in the DH since the system can derive all necessary information from the stored text plans.

During the user interaction, the system constantly updates the DH by first adding the user's request, followed by information from the content selection and text organisation phases. Consequently, the NLG algorithms and also VIEWGEN can obtain information about:

- the last explained concept;
- whether a concept was previously explained and how many pages ago;
- the content and structure of the last explanation;
- the content, structure and recency of the explanation of a given concept.

Based on this information, HYLITE+ can decide, for example, whether to insert a link to a previously explained concept, depending on how recently it was explained. However, the decision about whether a given concept or a fact is known by the user is based on the user model (unless it has been disabled by the user).

3.4. POPULATING AND UPDATING THE USER BELIEF ENVIRONMENTS

The application obtains information from the user registration, e.g., native language, age range, background, interests, and job information. The correctness of this information is not crucial as users can explicitly modify the system's behaviour (see Section 5), so the system can be used both by casual users and by registered users, e.g., students, who want to use the system on a regular basis. It is also possible to provide such user information externally, from a third-party application, if HYLITE+ is used within a bigger system, e.g., in the MIAKT project (see Section 8). If the user does not provide any information, then the system assumes a naive user with empty belief spaces.

This user information is used to trigger some generic stereotypes. For example, unless the user has stated that they have a chemistry degree and/or job, the system will not assign them a chemistry-expert stereotype. By default, total novices have key terms explained to them, if page length and formatting permit this.

After each new page is viewed, the system updates dynamically the user belief environment in VIEWGEN by adding the identifiers of all facts from that page. For each of these facts VIEWGEN automatically checks whether they are subsumed by other propositions, whose identifiers are already present in the belief space and only adds those that are new. The proposition identifiers are added to the topics

corresponding to the concept explained in the current page and to all other environments to which they are relevant.

For example, when the graph in Figure 3 about dispersion and waste water is used during an explanation of dispersion, its identifier is added first to the dispersion topic environment. Afterwards, since it is also relevant to waste water, its identifier is also added to the waste water topic.[5] In such a way, if the user requests an explanation of waste water later, information about this already known fact will be present in their belief space and the system's response can be tailored to reflect this (e.g., by inserting phrases like 'besides' – see Section 4.4).

3.5. CONSULTING THE UM DURING GENERATION

When the user requests information about a topic (e.g., emulsion), HYLITE+ first determines the set of propositions to be verbalised, i.e., the explanation content. The relevant propositions are obtained from VIEWGEN's system environment for the given topic (e.g., emulsion). If no such environment exists already, then HYLITE+ uses CG reasoning operations to find all graphs in the KB which contain the query concept or a sub-concept of it. As a result of that, VIEWGEN also creates a new environment for the topic and stores the results there, so they can be used there in subsequent requests. Effectively this leads to the creation of a separate VIEWGEN environment for each already explained concept.

Given the set of relevant propositions selected for generation, the generator first establishes whether any of them are already believed by the user. VIEWGEN is queried about each proposition and all already believed propositions are annotated as such. After all believed propositions are identified, the system also checks for any partially known propositions. This is necessary because the generator frequently uses CG reasoning operations to present only part of a given conceptual graph (to make the explanation shorter). For example, one of the graphs extracted for an explanation of the DISPERSION concept is:[6]

(cg1) [DISPERSION] <– (CHAR) <– [WASTE_WATER_WTH_PART].
:fs(derived_from: graph_id)

This is in fact a subgraph of the following:

(cg2) [WASTE_WATER_WTH_PART] –> (CHAR) –> [CONCENTRATION]
(CHAR) –> [RESISTANCE]
(CHAR) –> [DISPERSION].

[5]However, on Figure 3 we chose to show the graphs themselves, instead of their identifiers, in order to make it clearer to the reader what are the contents of each topic environment.
[6]fs() is used to encode features which can be associated with concepts, relations and graphs. In this example, the feature specifies which graph was used in the derivation of the graph it is associated with.

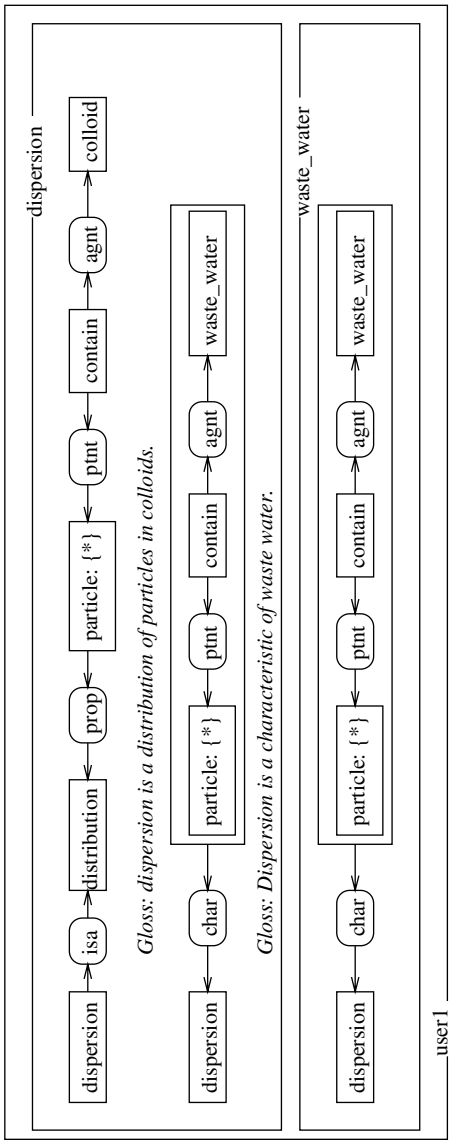

Figure 3. User's beliefs after an update following an explanation of dispersion. Before the explanation, the waste water topic environment was empty. Now the waste water environment is updated with the already said proposition.

Because the focus in this case is the term 'dispersion', cg2 will be verbalised as 'Dispersion is a characteristic of *waste water* that contains particles'. Subsequently, if the user clicks on the link for waste water, HYLITE+ needs to generate an explanation of that concept. At that point, cg1 is extracted as one of the graphs relevant to WASTE_WATER. When VIEWGEN is consulted, it returns that cg1 is not already believed by the user, since the user's belief environment only contains graph cg2.

However, the system checks each graph returned by the user model for a derived_from feature. If found, it checks whether the graph, from which the believed graph has been derived, is among those selected for explanation. In this case it annotates each such parent graphs (e.g., cg1) as partially known.

In our example, at this step graph cg2 will be identified as already believed by the user and also derived from cg1. Consequently, cg1 is annotated as partially known by adding *fs(um_state: partially_known)* to the graph. Also, the already verbalised concepts and relations are annotated as already believed by adding *fs(um_state: known)* to them. The result is:

[WASTE_WATER_WTH_PART] –> (CHAR) –> [CONCENTRATION]
(CHAR) –> [RESISTANCE]
(CHAR) *:fs(um_state: known)* –> [DISPERSION].
:fs(um_state: partially_known)

In other words, a new feature um_state is added to the graph and also to all relations which occur in the already explained graph. This information is then used by the generator to choose a phrasing which conveys the fact that dispersion is previously seen, e.g. 'as well as', which also makes the generated hypertext more coherent (see Section 4.4).

The next step is to establish whether there are contradictions between the system knowledge (i.e., the selected propositions) and the user beliefs. Therefore, all graphs returned as believed by the user (via a stereotype or from their own environment) are compared against all selected propositions using CG reasoning. If such a contradiction is detected, both system and user beliefs are added to the propositions to be conveyed, e.g. *bel(user, isa(photo_emulsion, synthetic_emulsion))* and *bel(system, isa(photo_emulsion, gel))*. In this example, the contradiction comes from the fact that the superconcepts of photo_emulsion are incompatible. In other words, the system's type hierarchy (i.e., a chemistry expert) specifies the supertype as gel, which is a type of sol (see Figure 2), whereas the default stereotype that applies for naive users has the supertype as emulsion.

Such incorrect beliefs are explained by providing them in parallel with the correct fact. In the photographic emulsion example, the system generates the following sentence as part of the explanation:[7]

[7]The 'a common error is ... ' expression is used to express the uncertainty of this assumption which has been made on the basis of a stereotype. The generator can also deal with incorrect beliefs of individual users.

A common error is to believe that photographic emulsion is emulsion, whereas, in fact, it is a gel.[8]

In the final step, VIEWGEN is consulted about concepts that appear in the selected propositions. If no user beliefs can be obtained about these concepts, then the generator flags them as unknown by adding a `(um_state: unknown)` feature.

4. Generating Adaptive Hypertext

Previous work on hypertext usability (e.g., Nielsen, 2000) and our own subject-based experiments (Bontcheva, 2001) were used to derive a set of requirements, the most important among which were:

- generation of well-structured hypertext with meaningful links;
- use of formatting that facilitates skimming (e.g., bullet lists);
- use of different colours for links to visited pages and links to new material;
- (short) clarifying information for important unknown terms.

This section details how these issues were addressed as part of the adaptive hypertext generation process in HYLITE+ and compares our approach to other dynamic hypertext systems.

4.1. CHOOSING FORMATTING

Hypertext usability studies (Nielsen, 2000) have shown that formatting is very important since it improves the readability of hypertext. Bullet lists and font size in particular facilitate skimming by making important information more prominent. Therefore this work has focused on generating these two types of formatting, especially due to the connection between formatting and some adaptivity techniques (see Section 4.5).

The browser default font size is used in the explanation body while the list of links to other relevant pages is generated in a smaller font size to keep the main text more prominent. All font sizes are relative, not absolute, so text size can be controlled from the browser.

Bullet lists received most attention because they are used much more frequently and even very concise texts can benefit from them. In addition, even if no other type of formatting is used, lists substantially improve the readability and user preference for such pages (see the study discussed in Nielsen, 2000, p. 105). The use of bullet lists in HYLITE+ is determined as a first step of the surface realisation process, on the basis of the fully fledged text plan.

The semantic aggregation stage joins all propositions which share the same focused entity and main relation, so the resulting more complex propositions can

[8] Emulsion was modelled in the generator's lexicon as a mass noun, which is the reason why there is no indefinite article in front of emulsion and photographic emulsion.

have three or more entities that need to be enumerated in the same sentence. For example, the following complex proposition appears in the text plan of a colloid explanation:

```
[COLLOID :fs(focus, true)] <- (ISA) <- [AEROSOL]
                           - (ISA) <- [FOAM]
                           - (ISA) <- [EMULSION]
                           - (ISA) <- [SOL].
```

The formatter examines each proposition in the text plan to determine if the focused entity participates in the same relation with three or more different concepts.[9] Such propositions are annotated for bullet list formatting if the repeating relations can be realised as verbs (e.g., ISA, PART_OF). No HTML markup is generated at this stage. Instead, the formatting choice is stored as annotation of the whole graph:

```
[COLLOID :fs(focus, true)] <- (ISA) <- [AEROSOL]
                           - (ISA) <- [FOAM]
                           - (ISA) <- [EMULSION]
                           - (ISA) <- [SOL]. : fs(format,ul)
```

This information is used later by the grammar to generate the HTML tags at the same time as the text itself. This separation allows the formatter (or a later module) to change this choice if it is not appropriate, e.g., to avoid overusing lists.

Such a distinction is also made in the RAGS project (Cahill et al., 1999), which has a special data type called *abstract document representation* used to represent information about ordering and layout (e.g., section, title, list-item). HYLITE+ currently does not have separate data types as in RAGS, instead all the information is encoded as annotations (features) on the relevant graph or concept. However, in future work we are considering the use of such data types because they also allow encoding of formatting information that spans more than one proposition (graphs in our case).

4.2. CHOOSING LINKS

HYLITE+ generates links for domain concepts which lead to separate explanation pages. In this way instead of having one long article discussing many related concepts (as is frequently the case in existing online encyclopaedias), the material can be broken down into several short pages.

There are three alternatives for generating links for terms that occur more than once in a page:

1. Provide a hypertext link for every occurrence;
2. Provide a hypertext link at the first occurrence of this term on this screen (i.e., if the previous link for this term is not visible on the screen, then provide the link again);
3. Provide a hypertext link only at the first occurrence of the term.

[9] The number was determined on the basis of the user preferences in the mockup experiment (Bontcheva, 2001).

User feedback during the experiment with existing encyclopaedia (Bontcheva, 2001) suggested that some users expect to have a link for each occurrence of a term, while others would prefer hypertext with less links, so the second alternative is better for such users. The third alternative is effectively the same as the second one for texts that fit on one page, but the second one is preferred for longer articles so that the user does not need to scroll the page.

To accommodate these preferences, the system has a parameter which users can set depending on their preference for a link strategy. By default the system provides a link for each appearance of the term.

Since the current knowledge base was built for experimental purposes, the data provided for some concepts is very sparse, sometimes just a definition, super- and sub-concepts. In the case where HYLITE+ has already provided the concept definition in the current page (e.g., as clarifying information in parenthesis – see Section 4.5), most of the information about this concept is already shown, so no link to a separate page is generated.

So far we have discussed links that occur within the generated explanation. Another useful type are links that point to relevant material, provided after the main text. As suggested by our user study (Bontcheva, 2001), they are often considered useful by users, as they suggest other pages which might be of interest, and help the user choose among the many alternatives. Their utility can be improved still further by grouping them in categories like More general, More speci c, Also related.

In HYLITE+, related topic links are generated based on the type hierarchy. The supertypes, subtypes, and sister concepts of the query concept are included respectively as More general, More speci c, and Also related (for an example see the right page in Figure 1). Initially, only the sister concepts were suggested as related topics because links to the supertypes and subtypes are already provided in the main explanation. However, the mockup experiment showed that almost all users preferred to see them all, regardless of their occurrence in the text. In the system evaluation (Section 6), one of the users commented that the more generic/specific links are desirable because they reinforce the points made in the text and make the term hierarchy more explicit.

In addition, users rely strongly on visual distinctions between links to visited and unvisited pages (Nielsen, 2000), so it is important that dynamic hypertext generation systems always use the same URL for the same page to maintain consistent link colouring. If such a distinction is indeed maintained, there is less of a need to adapt the page opening as well because the information that this page has already been visited is conveyed by the link colour. Therefore, our main effort in HYLITE+ implementation was focused on generating consistent URLs.

4.3. RETURNING TO PREVIOUSLY VISITED PAGES

Previous work on dynamic hypertext (Dale et al., 1997) has argued for treating hypertext generation in a similar way to dialogue generation, due to the dialogue-like nature of hypertext browsing. However, despite such similarities, one of the main differences between them is that in hypertext users can return to previously visited pages by using the Back browser button or a link. Therefore, due to strong user expectations, conditioned by traditional hypertext interaction, an intelligent information system (including a dynamic hypertext one) should avoid adapting the content and layout of already visited pages. Consequently, HYLITE+ does not change the page content when the `Back` button is used (unlike some other dynamic hypertext systems, e.g., Dale et al., 1998).

4.4. PRESENTING ALREADY SAID FACTS

As discussed in (Dale et al., 1998), coherence between pages can be improved by generating texts that take into account already said facts. For example, in the PEBA system, if the user followed a link to marsupial from a page about kangaroos, then the opening of the marsupial page is adapted accordingly:

> Apart from the Kangaroo, the class of Marsupials also contains the following subtypes... (Dale et al., 1998, p. 129)

In our system, we looked into the broader problem of contextualising the already presented material regardless of its position on the page.

In order to achieve this, HYLITE+ obtains information from VIEWGEN during content selection and annotates all partially known facts (see Section 3.5). This information is used during surface realisation to choose an appropriate cue word and sentence structure. For example, if the user has already seen the page about dispersion (which among other things states that dispersion is a characteristic of polluted water), then this information would be annotated as known in the graph about polluted water:

```
[WASTE_WATER] –> (CHAR) –> [CONCENTRATION]
                (CHAR) –> [RESISTANCE]
                (CHAR) :fs(um_state: known) –> [DISPERSION].
    : fs(um_state: partially_known)
```

The grammar takes into account such known sub-facts and tries to generate a sentence using phrases like besides and apart from. For our example, the result would be:

> Besides dispersion, other characteristics of polluted water are concentration and resistance.

If the grammar fails to use such constructions, the default rule is applied which ignores the `known` annotation and produces the neutral sentence:

Polluted water is characterised by dispersion, concentration, and resistance.

However, and unlike normal dialogue, hypertext is often skimmed instead of being read fully, so it is more likely that users do not remember all the facts from a previous page. Therefore, the use of such contextualising phrases becomes less appropriate as the number of intermediate pages increases. This conjecture was confirmed in our mockup experiment, where all users preferred the contextualised hypertext when the fact also appeared in the previous page (0 intermediate pages), but that number decreased to 50% when there were 3 intermediate pages (Bontcheva, 2001). Consequently, HYLITE+ uses the discourse history in conjunction with the annotations from the user model to determine whether or not it is appropriate to use this particular grammar rule.

Another case is when the whole graph has already been presented in an earlier explanation, e.g., the user has already been told all the characteristics of polluted water. One approach is to treat this as a repetition and remove the sentence from the explanation (e.g., BLAH, Weiner, 1980). However, the predominantly skimming behaviour of hypertext readers, combined with the text genre (encyclopaedia), makes this approach inappropriate because the users expect to see the complete information about each topic. Therefore, facts are always included in the explanation and the grammar can use a phrase like As you probably remember or As previously discussed to indicate that the material has been seen before (provided that the user has not disabled the use of such phrases – see Section 5).

4.5. CLARIFYING UNKNOWN TERMS

As discussed in Section 3.5, the generator uses information from the user model to detect and annotate unknown concepts. During surface realisation, hypertext links leading to separate explanation pages are added for these concepts. In addition, some parenthetical information is considered for unknown concepts in definitions, parts and subtypes.

The system mockup experiments (Bontcheva, 2001) showed that users prefer two types of concise parenthetical information:

– brief term definitions;
– a familiar superconcept.

4.5.1. *Generating Term Definitions*

Let us assume a user who has looked up computer programs and then followed a link to personal computer; the user has not specified preferences for the type of clarifying information, so definitions can be provided where appropriate. Following this request, the content planner passes the following facts for realisation (the graph describing all parts is truncated here):

[PCISA] –> [MICRO_COMP *:fs(um_state:unknown)*].

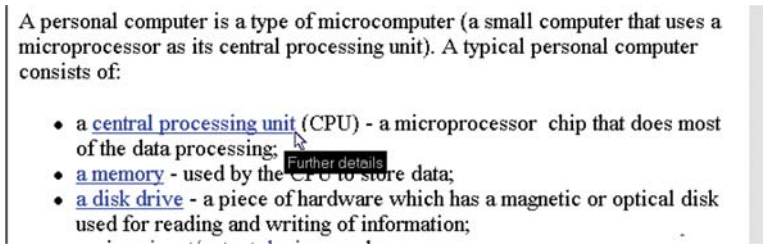

Figure 4. Example of clarifying definitions integrated in the main text.

[PC] <– (PART_OF) <– [CPU *:fs(um_state:unknown)*]
– (PART_OF) <– [MEMORY *:fs(um_state:unknown)*]
– (PART_OF) <– [HDD *:fs(um_state:unknown)*]
– (PART_OF) <– [MONITOR]...

First, the realiser determines the explanation formatting (see Section 4.1). In this case, a bullet list is chosen to enumerate all parts. Then it starts generating text for the first graph. Because the newly introduced supertype (microcomputer) is unknown, but important for the understanding of the text, the system decides to provide some extra material about it. The action corresponding to generating a new definition is `define(micro_comp)`, which is passed as a parameter to the recursively called generator. The generator parameters are also set for very short texts with syntactic preference for noun phrases. The resulting text – a small computer that uses a microprocessor as its central processing unit – is evaluated for length[10] and provided in brackets (see Figure 4).

Similarly for all unknown parts of the PC, the generator is called recursively to produce their definitions (see Figure 4).

In order to determine the appropriate formatting for the new material, the system has a set of rules which examine the type of proposition (e.g., `definition`, `part_of`, `attribute`) and the formatting of the main text. At present, new material which is to be integrated in unformatted text is put in parenthesis. Since such text disturbs the flow of the main sentence, parenthetical definitions are only provided *once per sentence*, usually for the most important concept in the proposition (e.g., the supertype). For other unknown concepts in the same sentence only a hypertext link is generated.

If more than one parenthetical definition needs to be generated within the same page (see Figure 4), a separate recursive call to the generator is made for each one. On one hand this might sometimes lead to the generation of less fluent text. In order to minimise this risk, HYLITE+ uses an entity stack which provides information about all previously defined concepts. On the other hand, this approach has

[10]Based on the user perceptions from the mockup studies the length restriction was set to a maximum of 15 words. More empirical studies are needed to establish if this is indeed an acceptable length.

the advantage of simplifying the generation process, thus making it more efficient (Bontcheva and Wilks, 2001).

The reason for the somewhat circular nature of the generated explanation in this particular example, i.e., the use of CPU as part of the definition of microcomputer, is an artefact of the contents of the knowledge base itself (in this case, the knowledge was obtained from a glossary of computer terms). Figure 1 shows a different style of parenthetical definitions, due to the different underlying knowledge. In general, these examples show that the quality of the generated content depends to a large degree on the domain knowledge available in the system.

4.5.2. *Providing a Familiar Supertype*

The other type of clarifying information currently provided for unknown terms is familiar supertypes in parenthesis. Since this kind of information can be obtained directly from ViewGen, the generator does not need to carry out planning for the new material. Instead, the surface realiser uses a (series of) `isBelieved` request(s) to obtain from ViewGen information whether a supertype is already familiar to the user.

A supertype could be familiar from a previously visited page, in which case it will be present in the user's ViewGen environment. For example, if the user has visited the page about colloid before looking up gel, then `COLLOID` would be selected as the familiar supertype of `SOL` (which would be an unknown concept in the gel explanation). In contrast, if colloid is unknown, then the familiar supertype of `SOL` would be selected by the default mechanism (described next).

When no supertype is present in the user's environment, default reasoning is used to obtain a familiar supertype based on common world knowledge (modelled in ViewGen as stereotypes that apply to all users). Since the KB taxonomy has a very generic upper half with concepts like `OBJECT` and `DEVICE`, a commonly known supertype can usually be found. Following this default approach, the familiar supertype of `COLLOID` is determined to be `MIXTURE`. In this case, the generated sentence would be:

> Gel is a colloid (mixture) of solid particles in a liquid.

The related problem of generating object descriptions using content words familiar to the user has been addressed by (Reiter, 1990). His goal was to generate descriptions that are accurate and free from false implicature, based on a domain taxonomy and a user model. The difference is that terminological explanations are oriented towards users who try to familiarise themselves with the domain terms, so HYLITE+ needs to provide both the unknown word and the familiar supertype.

This example above also shows that the chosen way of lexicalising the familiar supertype in parenthesis might mislead users into thinking that mixture is another name for colloid. One way of making the supertype–subtype relation more explicit is to include 'a kind of'. An alternative is to use the supertype in the explanation

and provide the term in the parentheses instead, e.g., 'Gel is a mixture (called colloid) of solid particles in a liquid'. In future work, we will experiment with these alternatives or, perhaps even better, allow the users to choose which one they like best as part of HYLITE's adaptability features.

5. Adaptability: Giving Users Control

The results from our mockup experiments (Bontcheva, 2001) showed that the participants had widely different opinions for some adaptivity features. For example, one of the users found the use of phrases like as you have already seen and as you probably remember too patronising and would want to disable their use, although (s)he liked the other adaptivity ideas (e.g., parenthetical definitions). The fact that none of the other users disliked these phrases shows how individual these preferences can be.

Inspired by other adaptive hypertext systems, which offer adaptability (see Section 7), we implemented HYLITE+ so that the user has control over the user model and the adaptivity features via the generated hypertext pages. The available options are:

– User modelling:
  - whether or not the system *updates their* VIEWGEN *model*;
  - whether or not the system *uses their* VIEWGEN *model*;
  - whether or not the system provides *information inherited* from more generic concepts;
  - whether or not the system uses *stereotypes*;
– Adaptivity features
  - whether or not the system generates *parenthetical definitions*;
  - whether or not the system presents *familiar supertypes* in brackets;
  - whether or not the system provides *links to related material*;
  - whether or not the system uses *contextualising phrases* like besides.
– *Disable all* adaptivity and user modelling.

When an adaptivity feature is disabled, this affects the generation algorithms. For example, if parenthetical definitions are disabled, but familiar supertypes are still enabled, then only the latter will be generated for unfamiliar terms, because the rules for the generation of the former will not fire.

The adaptability also proved useful during the development and testing of the generation algorithms, because it offered control over the corresponding functionality. In this way, it was possible to examine the influence of each of these features on the generated hypertext.

The user is given control over these parameters by including at the bottom of each page a set of links pertaining to adaptations in the generated text (e.g.,

'disable parenthetical definitions', 'enable links to related pages'). The links correspond to the parameters of the user model and adaptivity features discussed above. This approach is similar to that of some other adaptive hypertext systems (see Section 7).

6. Evaluation

We carried out a formative evaluation of HYLITE+ in order to assess the acceptability and utility of the adaptive features. The users interacted with two versions of our system: a baseline one and the adaptive one. The two versions were, in fact, the same system with the user model and adaptivity features switched off to form the non-adaptive baseline. In this way, we ensured that the same information was available in both systems. Also, this approach minimises the potential influence of different systems' response times on the experiment results, because both versions generate hypertext on the fly and have similar performance.[11]

The initial system page, which was identical in the two versions, contained an alphabetical list of topics, relevant to the experimental tasks (see Figure 1, left). The participants could request an explanation on a given topic by selecting it and clicking **Explain**. After reading the generated page, they could obtain further information by following links, navigating back and forward with the browser buttons, or by selecting a new term in the topic list (see Figure 1, right).

6.1. METHODOLOGY

Hypermedia applications are often evaluated with respect to: *interface look and feel*, *representation of the information structure*, and *application-specific information* (Wills et al., 1999). The information structure is concerned with the hypertext network (nodes and links) and navigation aids (e.g., site maps, links to related material, index). The application-specific information concerns the hypermedia content – text, images, etc. For our system there is no need to evaluate the interface, since HYLITE+ generates HTML and uses Web browsers as rendering tools. Therefore, the evaluation efforts were concentrated on the information content and navigational structure of the generated hypertext.

In this paper we discuss the following measures:[12]

- information content measures: average time to complete each task.
- navigational structure measures: average time per page visited; number of links followed; usage of the browser Back button; and usage of the system's topic list to find information.

[11] The ILEX system, for example, used pre-generated static pages as a baseline and the study reported that the difference in the two systems' response times might have influenced some of the results (Cox et al., 1999).

[12] For detailed information on the other measures used in this evaluation see Bontcheva (2001).

The experiment had a *repeated measures*, *task-based* design (also called within-subjects design), i.e., the same users interacted with the two versions of the system, in order to complete a given set of tasks.[13] The tasks were of three types: browsing, problem-solving, and information location.

The tasks were designed to be as realistic as possible for computer users who frequently need to make their own decisions on buying hardware for their PCs.[14] The particular topics for the browsing and problem-solving tasks were chosen so that even most computer-literate people would not have sufficient prior knowledge about them. For example, the two problem solving questions were to use the system in order to choose the cheapest tape drive suitable for backing up 10GB/30GB of data. Figure 1 shows the hypertext page generated by the system for a novice user, when they ask for information on tape drives.

The tasks were completed successfully by eight participants with a computer science background – three male and five female. Prior to the experiment, they were asked to provide some *background information* (e.g., computing experience, familiarity with Web browsers, and electronic encyclopaedia) and to fill in a *multiple choice pre-test*, that diagnosed their domain knowledge, in order to separate them into two categories: novice and more advanced users. The users were assigned randomly to two groups: adaptive-first and non-adaptive first. However, care was taken to have the same number of novice and advanced users in each of these groups. After completing the first three tasks, the users swapped systems for the remaining three tasks.

At the end the participants were asked to fill in a *questionnaire* and participate in a *semi-structured interview* discussing their experience with the two systems, in order to elicit problems and provide qualitative assessment of the implemented adaptivity techniques.

6.2. RESULTS

Due to the small sample size and the differences in users' prior domain knowledge and browsing styles, the results obtained could not be used to derive a statistically reliable comparison between the measures obtained for the adaptive and the non-adaptive versions. Nevertheless, the evaluation led to an innovative finding, namely that users' reading behaviour needs to be taken into account in future experiments, as it has a major impact on the performance measures. Next we will present the data analysis both with respect to content and navigation measures and justify this conclusion.

6.2.1. *Importance of Users' Reading Behaviour: Skimmers vs Readers*

One of the quantitative measures derived from the interaction logs was *average (mean) time per task*. The results showed (see Table II that the users who

[13]The design of the tasks follows the design used in the evaluation of two other adaptive hypermedia applications – PUSH (Höök, 1998) and (Wills et al., 1999).

[14]The task design was based also on articles and buyers' guides from magazines such as *What PC*.

Table II. Mean and standard deviation of task completion time (in seconds)

Task		Seconds to complete non-asaptive	Seconds complete adaptive
Browse ZIP drive	Mean	194.50	129.00
	Std. Deviation	86.87	59.67
Choose 'l0/1 GB tape drive'	Mean	214.50	193.75
	Std. Deviation	93.49	81.33
Locate mouse details 1	Mean	260.25	158.25
	Std. Deviation	78.93	75.74
Browse PCMCIA cards	Mean	113.75	133.75
	Std. Deviation	39.91	61.82
Choose 30/50 GB tape drive	Mean	102.75	135.50
	Std. Deviation	42.92	55.16
Locate mouse details 2	Mean	87.75	162.25
	Std. Deviation	42.46	78.13
Total	Mean	162.25	152.08
	Std. Deviation	89.08	654.52

The rows of the table show the mean and standard deviation for each of the six tasks and the total. The rows called 'Browse...' are for the browsing tasks, 'Choose...' are the problem-solving tasks, and 'Locate...' are for the information location tasks. The thick line shows where the participants swapped systems.

completed the first three tasks with the non-adaptive system took longer on average, with the biggest differences for the browsing and information location tasks. Since the participants swapped systems for the next three tasks and were already familiar with the hyperspace, the average times for both the adaptive and non-adaptive versions are naturally lower.

Unfortunately, these results cannot be taken as an indication that, for example, the adaptive system might be more helpful when the user is not familiar with the hyperspace. This is because we discovered *post hoc* an important user characteristic which influenced the task completion times, but no previous evaluations of adaptive hypertext systems have controlled for it. The problem comes from the fact that even when asked to locate particular information, some users still read the hypertext pages thoroughly, instead of just skimming them to find only the facts relevant to the task.[15]

Although we were aware that previous studies of people browsing hypertext (e.g. Nielsen, 2000) have differentiated two types, *skimmers* and *readers*, we did not expect that this distinction would be important when the users perform information location tasks. Unfortunately, not only does the difference appear to remain, regardless of the task, but it also happened that when the participants were assigned to the systems, all three 'readers' were assigned to the same systems, i.e., the non-adaptive for the first three tasks, then the adaptive one for the remaining three. This explains why the results obtained for the two groups are so different.

[15]This readers/skimmers difference was discovered by comparing the mean time each participant spent per visited hypertext page.

Consequently, the results for mean time per task need to be interpreted by taking into account this group effect (3 readers and 1 skimmer *versus* 4 skimmers). As can be seen in Table II, both groups were faster when they interacted with the second system (adaptive for the first group and non-adaptive for the second group), because they were familiar with the hyperspace. The group effect also explains why the mean times per task are consistently higher for the non-adaptive system in the first three tasks and higher for the adaptive system in the second set of three tasks. The readers group (non-adaptive first) was always much slower than the other group, which consisted only of skimmers.

This group effect will become apparent again in other measures discussed below. Its impact on the evaluation results show that it is very important to control for the differences in user characteristics and behaviour, which influence the subjects' task performance much more than the two different experimental conditions, i.e., adaptive versus non-adaptive system. In order to avoid this problem in future experiments, we intend to control for users' reading behaviour prior to the experiment, just as we did for domain knowledge.

The questionnaire showed that the users did not have problems locating information, neither did they find the texts too long. The results also showed that most users (75%) found the additional information in the adaptive system useful, while the rest were neutral. Overall, the participants did not have problems understanding the generated explanations.

6.2.2. *Hypertext Navigation Measures*

The measures related to hypertext navigation are the use of links to related pages, navigation browser buttons and the topics list. The statistics of the use of these navigation aids in the non-adaptive and adaptive versions are shown in Table III.

In both systems, the subjects used links as their primary way to navigate the hyperspace (Table III, first 2 columns). The browser Back button (Table III, middle 2 columns) and the topics list (Table III, last 2 columns), however, were used much more frequently in the non-adaptive system, e.g., the topic list was used, on average, twice as often. The topic list was used mostly in the two 'locate mouse details' tasks to access directly information about particular types of mouse (e.g. wheel, optical) – see Table III.

The use of the topic list in the non-adaptive system is due to the fact that the links to related pages were only available in pages generated by the adaptive system.

The difference in the use of the browser Back button between the two versions seems to be due mostly to the different ways in which the participants navigated the hyperspace. In the non-adaptive version, they would often start from a topic, e.g., tape drive, then follow a link to the first subtype, e.g., DLT, then return to the previous page, explore the next subtype, etc. In other words, the most common navigation pattern was similar to a depth-first traversal of the domain taxonomy.

Table III. Statistics of the use of navigation features by task. The thick lines shows where the participants swapped systems

Task		Links used non-adaptive	Links used adaptive	Back button used adaptive	Back button used non-adaptive	Topics list used adaptive	Topics list used non-adaptive
Browse ZIP drive	Mean	3.25	4.25	3.25	3.25	0.25	0.00
	Std. Dev.	1.26	2.36	1.26	2.06	0.50	0.00
Choose '10/1	Mean	7.75	7.25	6.50	5.25	0.00	0.25
GB tape drive'	Std. Dev.	5.68	2.99	3.32	1.50	0.00	0.50
Locate mouse	Mean	5.67	6.00	4.67	2.33	2.33	1.00
Details 1	Std. Dev.	2.52	4.58	1.53	2.52	1.15	1.00
Browse	Mean	4.33	3.50	2.33	1.25	0.33	0.00
PCMCIA cards	Std. Dev.	0.58	0.58	2.31	1.89	0.58	0.00
Choose 30/50	Mean	5.33	4.33	5.00	1.00	0.00	0.00
GB tape drive	Std. Dev.	0.58	1.53	1.00	1.73	0.00	0.00
Locate mouse	Mean	4.00	7.00	3.33	2.75	2.00	1.00
Details 2	Std. Dev.	3.46	5.35	3.21	2.75	1.73	1.41
Total	Mean	5.10	5.41	4.25	2.73	0.75	0.36
	Std. Dev.	3.14	3.28	2.47	2.35	1.21	0.79

On the other hand, the additional information and the extra links to related material in the adaptive system changed the subjects' navigation pattern. The interaction logs showed an increased use of 'horizontal' taxonomic links, i.e., users would often jump from one subtype straight to the next, without going back to the 'parent' node. This change of behaviour was observed for more than half of the participants and accounts for the reduced number of repeated visits to the same page, and hence the reduced number of visited pages in the adaptive system.

However, some of these results might again have been influenced by the difference between the two groups: readers (non-adaptive first) and skimmers (adaptive first). The use of the Back button by the readers group changes substantially when they move to the adaptive system (e.g., from 3.25 for task 1 to 1.25 for task 4). Unlike them, the behaviour of the skimmers group hardly changes between the two systems (e.g., 5.25 for task 2 and 5.00 for task 5). Therefore, it is hard to judge to what degree the difference is due to the readers/skimmers effect or to the different treatment they received, i.e., which system they interacted with first. The reason for this is that possibly the people who read the pages more carefully could remember better which of the other terms present on the previous page were worth exploring and jump straight to them, i.e., benefit from the extra links. Unlike them, the skimmers needed to go back to the previous page to remind themselves of the other terms, so they could choose where they wanted to go next. This conjecture is also supported by the difference in the total number of pages visited by the two types of users.

The questionnaire results showed that none of the users felt disoriented in any of the systems and the majority had no problems finding information. When deciding which links to follow in the non-adaptive system, some of the novice users reported problems with unfamiliar terminology (25% of all users). In addition, half of the participants responded that there were not enough links in the non-adaptive pages. 37.5% of the users also felt that they had to visit too many pages in the non-adaptive system in order to find the information needed. For the adaptive system this number was down to 12.5% and all the other users actively disagreed with that statement. The majority of the users also felt that the extra information and links provided in the adaptive version made it easier for them to choose which link to follow next.

7. Related Work

In the context of previous work on text generation, the HYLITE+ approach is novel because it re-uses a generic user modelling framework (i.e., general purpose UM shell) – VIEWGEN, which can be shared with other parts of the system, e.g., dialogue planning and understanding. In comparison, other NLG systems which employ user models in order to generate tailored output (Moore, 1995; Knott et al., 1996; Milosavljevic et al., 1996; Ardissono and Goy, 2000) tend to use custom-implementations where facts from the generator's knowledge base can be marked

as known by the user. Differences and conflicts in user and system belief are often not represented with a few exceptions (McCoy, 1988). Typically, the representational power and structure of the user model are chosen to be sufficient for the task at hand. In contrast, a generic user modelling framework can support both simpler, overlay-style models and more complex ones, including user goals and plans. This flexibility makes it easier to re-target HYLITE+ to new applications, without need for further implementation or fine-tuning of the user modelling mechanism.

Unlike previous NLG-based approaches, other adaptive hypermedia systems have used such generic user modelling frameworks successfully in a variety of applications, thus proving their advantage. The most relevant system of this kind is KN-AHS (Kobsa et al., 1997) which uses the general purpose user modelling shell system BGP-MS (Kobsa and Pohl, 1995). In terms of representational power, BGP-MS is very similar to VIEWGEN: both frameworks model domain knowledge (BGP-MS uses a KL-ONE-based formalism), support stereotypes, individual user models, and are capable of representing conflicts. The main difference between HYLITE+ and KN-AHS is that HYLITE+ employs language generation techniques to produce the hypertext content and links automatically, while KN-AHS uses human-written texts. Both HYLITE+ and KN-AHS use well-defined protocols to communicate with their user modelling shells. This effectively de-couples the systems from the concrete modelling frameworks and makes it possible to replace them with other UM shells (e.g., UM toolkit, Kay, 1995) as long as they offer similar functionality. Other adaptive systems with open user models, i.e., models that can be shared between applications are AHA! (De Bra and Calvi, 1998) and Interbook (Brusilovsky and Schwarz, 1997).

Another difference between KN-AHS and HYLITE+, on one hand, and adaptive systems like AHA! (De Bra and Calvi, 1998) and Metadoc (Boyle and Encarnaçion, 1994), on the other, is in the richness of their knowledge representation formalisms. The latter class of systems tends to model only domain concepts, whereas the former ones support richer structures, e.g., concept hierarchies, relations, and reasoning operations such as inheritance. Interbook (Brusilovsky and Schwarz, 1997), for example, uses domain relations like *is-a* and *part-of*, in combination with the student model, to personalise the sequence of concepts to be learned. Similarly, HYLITE+ exploits the type hierarchy to recommend links to related pages.

Another relevant area is research on open learner models, i.e., providing users with control over their models. HYLITE+ does not address the issue of externalising the user model and allowing users to change its content. This area has been the focus of research in some adaptive tutoring systems. For example, in the ELM-ART adaptive hypertext system (Weber and Brusilovsky, 2001) users can specify which course units are already known to them. Interactive student diagnosis is illustrated in STyLE-OLM (Dimitrova, 2003) which implements a novel student modelling approach that enables a learner to inspect and discuss the content

of their model. As a result, an advanced and reliable learner model is extracted, which can be used by an intelligent tutoring system to adapt to the needs of each individual student. The system used conceptual graphs as the formalism for showing and discussing the content of the learner model, thus demonstrating that CGs can be understood by non-specialist users.

The adaptivity techniques in HYLITE+ fall into two main categories: content adaptation and link adaptation (following Brusilovsky, 1996). The adaptive generation of explanations for unknown concepts in HYLITE+ is similar to conditional inclusion of content in systems such as AHA! (De Bra and Calvi, 1998) (through conditional fragments), KN-AHS (Kobsa et al., 1997), and Metadoc (Boyle and Encarnaçion, 1994) (via stretchtext). However, our NLG-based approach does not require authoring efforts for all possible fragments. It can also choose a different order for presenting the page content, which would be harder to achieve in 'traditional' AH systems where all variants need to be pre-written.

HYLITE+ performs link adaptation in two main ways: link removal/hiding and inclusion of links to other relevant pages. Link removal occurs when all content about a given concept has already been included in the current page (e.g., clarifying information in parenthesis), as a result of content adaptation. Similar features exist in adaptive systems like AHA! (De Bra and Calvi, 1998) and ELM-ART (Brusilovsky et al., 1996), which also provide direct guidance, i.e., they determine which page to recommend to be visited next (typically by having a 'next' button).

As already argued above, the main difference between HYLITE+ and other dynamic hypertext systems comes from the use of a generic user modelling framework. In addition, in HYLITE+ we focus mainly on the generation of clarifying information and contextualised presentation of already known facts. The focus of PEBA-II (Milosavljevic et al., 1996) is on generation of comparisons – work which is complementary to ours. The ILEX system (Knott et al., 1996) studies the similarities between dialogue (or conversation) and hypertext navigation. An interesting aspect of this work is that it suggests the replacement of the traditional 'hypertext as spatial navigation' metaphor with a more conversational one. However, as discussed by its authors, it is unclear whether changing already visited pages would not be confusing to the users. Therefore, in HYLITE+ we decided only to modify new pages, while allowing users to use the Back browser button and keeping already seen pages intact.

Another relevant system is ARIANNA (Carolis et al., 1998), which generates medical guidelines, including explanations aimed at teaching health care staff. Therefore, as argued by the authors themselves, their style and function are different to explanations meant to inform users (like those generated in HYLITE+). Another distinguishing aspect is that ARIANNA uses human-written canned text associated with domain concepts instead of a surface realiser. The disadvantage of this approach is that it cannot perform adaptation at sub-sentence level, as HYLITE+ does when presenting already known facts.

With respect to using HCI techniques to design adaptivity in dynamic hypertext systems, the MIGRAINE system (Buchanan et al., 1995) is relevant here, because it used comprehensive ethnographic studies in order to design the tailoring strategies for its patient explanation sheets. In contrast, in HYLITE+ we used cheaper techniques like mockups and walkthroughs (Bontcheva, 2001), due to the much smaller scope of our project. Another difference is in the generation techniques: MIGRAINE uses text planning, whereas we used schemas due to their efficiency.

Finally, an important distinguishing aspect of HYLITE+ in comparison to all dynamic hypertext systems is in its support for *adaptability* as part of the NLG algorithms. Previous work in adaptive hypertext has shown that providing users with control over aspects of the system's adaptive behaviour is a feasible approach. For example, AHA! (De Bra and Calvi, 1998) offers users control over the link colour scheme through a form and over link underlining via the browser settings (as does HYLITE+). Interbook (Brusilovsky and Schwarz, 1997) goes further by providing users with means to enable or disable some interface features via hypertext links provided at the bottom of the pages. The changes take effect immediately.

Adaptability in HYLITE+ is implemented in a similar way (see Section 5). However, unlike Interbook, our system does not modify the current page, but applies the settings only for subsequently generated pages, in order to maintain consistency.

On the other hand, changing the current page might have the advantage of helping the user to understand better the system behaviour and how it can be controlled. This issue was studied in detail in the Tutor3 system (Czarkowski and Kay, 2003). It offers a special link 'How was this page adapted to you' and enables the user to see the different adaptations applied to the page (in this case mostly insertions and deletions of content). The evaluation showed that, once users became aware of the scrutiny features, they could understand how to control the system's adaptive behaviour.

8. Conclusions and Future Work

This work belongs to the broad area of building intelligent systems that adapt their behaviour according to the characteristics and preferences of their users. In particular, this research addressed the problem of generating hypertext explanations, tailored to the user and to the interaction context. As a result of this work, we developed an efficient and modular approach to adaptive hypertext generation, which can be used in a number of applications, ranging from online information systems to e-mentoring. Since some of these applications require powerful models of the user, we incorporated in our algorithms a generic agent modelling framework (VIEWGEN), which can support both simpler, overlay-style models and more complex ones, including user goals, interests, etc. Conceptual Graphs (Sowa, 1984) were used for knowledge representation both in the user modelling and the NLG components. However, VIEWGEN has been used successfully with other formalisms,

e.g., first-order logic (Lee and Wilks, 1996), so our user modelling approach is independent from CGs.

Another strand of this research has been concerned with designing and implementing adaptivity techniques which are acceptable and useful in the context of dynamically generated explanations. Following the classification in (Brusilovsky, 1996), we can distinguish two main types of adaptivity: *adaptive presentation* and *adaptive navigation support*. In HYLITE+, we addressed both categories: in adaptive presentation we focused on formatting, already known facts, and additional information. Adaptive navigation support was addressed via link colours and the suggestion of links to related material.

The core assumption of our adaptivity approach is that users are assumed to have seen facts that have been presented to them and this is reflected in their user models. However, this information is not used to omit information, but mainly to determine how the page should be tailored to the context and the user.

The adaptive explanation techniques developed in this research are complementary to some of those from previous work on dynamic hypertext and user-adapted explanations:

- generation of comparisons (e.g., PEBA, Dale et al., 1998);
- presentation of different content, based on users' interests (e.g., ILEX, Knott et al., 1996);
- explaining object-related misconceptions (e.g., McCoy, 1989);
- concept examples in descriptions (e.g., Mittal and Paris, 1993).

Our task-based evaluation was aimed at assessing the usability of the generated adaptive hypertext and also indicate potential problems with the adaptivity techniques. The sample size (8 users) was chosen to be sufficient from a usability perspective, where only a small number of people are needed to identify problems with a web site (Starling, 2002). The results showed that the participants found the adaptive system easy to use and the generated hypertext intuitive to navigate. The generation techniques were sufficiently fast, so users did not have to wait for the system response. All users felt comfortable with both the adaptive and the non-adaptive versions. None of them stated that either system was unusable. In addition, the majority preferred the adaptive system, where they also felt it was easier to perform the tasks.

From a statistical point of view the sample size is too small to derive a statistically reliable comparison between the performance measures obtained for the adaptive and the non-adaptive versions, but the quantitative results and the qualitative feedback are sufficiently encouraging to suggest that adaptive generation of additional information in hypertext is of benefit to some types users (e.g., helps their navigation in information seeking tasks). The evaluation also led to an innovative finding, namely that users' reading behaviour needs to be taken into account in future experiments, as it has a major impact on the quantitative measures.

At present, we are investigating the reusability and portability of HYLITE+ to new domains and applications. For instance, work in the MIAKT project (http://www.aktors.org/miakt) is focused on automatic generation of textual descriptions from the semantic information associated with a medical case – patient information, medical procedures, X-rays, etc. Research in SEKT (http://sekt.semanticweb.org) is concerned with generation of textual summaries from Semantic Web ontologies, tailored to the user's profile and device. Another potential application is intelligent tutoring systems, e.g. learning agents (Goodman et al., 2004).

One particularly promising extension of this work is concerned with modelling user interests. As demonstrated in the ILEX hypertext generation system (Knott et al., 1996), user interests play an important role in the selection of relevant content. Interests can be specified by the user as keywords or assigned from a stereotype. There are also approaches which automatically identify and track interests by monitoring users' e-mail and Web activities (e.g., Crabtree and Soltysiak, 1998). We envisage that adding support for interests in VIEWGEN will be straightforward, by introducing a new type of attitude, called *interest*, in a similar way to which VIEWGEN has been extended previously to handle user goals (Lee and Wilks, 1996).

Another strand of future work would be to make VIEWGEN's belief model a probabilistic one (e.g., Jameson, 1995). Recent work on applying such models to argument interpretation (Zukerman and George, 2004) has demonstrated encouraging results, based on both synthetic and user-based evaluation experiments.

Acknowledgements

This work is being supported in part by the UK Engineering and Physical Sciences Research Council as part of the MIAKT project (Grant GR/R85150/01) and also the EU-funded project SEKT. Previous support included a University of Sheffield Ph.D. fellowship and an Overseas Research Students Award. The authors also wish to thank the three anonymous referees and the editors of the special issue whose comments and suggestions helped us improve this paper.

References

Ardissono, L. and Goy, A.: 1999, Tailoring the interaction with users in electronic shops. In: *Proceedings of the 7th Conference on User Modeling*. Banff, Canada: Springer Verlag, 35–44.

Ardissono, L. and Goy, A.: 2000, Dynamic generation of adaptive web catalogs. In: P. Brusilovsky, O. Stock, and C. Strapparava (eds.): *Adaptive Hypermedia and Adaptive Web-Based Systems*: Berlin, Heidelberg: Springer Verlag, 5–16.

Ballim, A.: 1992, Viewfinder: A framework for representing, ascribing and maintaining nested beliefs of interacting agents. Ph.D. thesis, Federal Polytechnic of Lousanne.

Ballim, A. and Wilks, Y.: 1991, Beliefs, stereotypes and dynamic agent modelling. *User Modelling and User-Adapted Interaction* **1**, 33–65.

Bontcheva, K.: 1997, Generation of multilingual explanations from conceptual graphs. In: R. Mitkov and N. Nicolov (eds.): *Recent Advances in Natural Language Processing: Selected Papers from RANLP'95*, Vol. 136 of *Current Issues in Linguistic Theory (CILT)*. Amsterdam/Philadelphia: John Benjamins, 365–376.

Bontcheva, K.: 2001, Generating adaptive hypertext explanations. Ph.D. thesis, University of Sheffield.

Bontcheva, K. and Wilks, Y.: 2001, Dealing with dependencies between content planning and surface realisation in a pipeline generation architecture. In: *Proceedings of the International Joint Conference in Artificial Intelligence (IJCAI'2001)*. Seattle, WA.

Bontcheva, K. and Wilks, Y.: 2004, Automatic report generation from ontologies: the mia-kt approach. In: *Ninth International Conference on Applications of Natural Language to Information Systems (NLDB'2004)*.

Boyle, C. and Encarnaçion, A. O.: 1994, MetaDoc: An adaptive hypertext reading system. *User Modelling and User-Adapted Interaction* **4**(1), 1–19.

Brusilovsky, P.: 1996, Methods and techniques of adaptive hypermedia. *User Modelling and User-Adapted Interaction* **6**(2–3), 87–129. Special issue on Adaptive Hypertext and Hypermedia.

Brusilovsky, P.: 2001, Adaptive hypermedia. *User Modelling and User-Adapted Interaction* **11**(1–2), 87–110.

Brusilovsky, P. and Schwarz, E.: 1997, User as student: towards an adaptive interface for advanced Web-based applications. In: A. Jameson, C. Paris, and C. Tasso (eds.): *Proceedings of 6th International Conference on User Modeling*. Sardinia, Italy: Springer, 177–188.

Brusilovsky, P., Schwarz, E., and Weber, G.: 1996, ELM-ART: An intelligent tutoring system on world wide web. In: C. Frasson, G. Gauthier, and A. Lesgold (eds.): *Intelligent Tutoring Systems: Proceedings of the Third International Conference*. Berlin: Springer, 261–269.

Buchanan, B. G., Moore, J. D., Forsythe, D. E., Cerenini, G., Ohlsson, S. and Banks, G.: 1995, An intelligent interactive system for delivering individualized information to patients. *Artificial Intelligence in Medicine* **7**, 117–154.

Cahill, L., Doran, C., Evans, R., Mellish, C., Paiva, D., Reape, M., Scott, D. and Tipper, N.: 1999, Towards a reference architecture for natural language generation systems. Technical report, ITRI-99-14, University of Brighton. Available at http://www.itri.brighton.ac.uk/projects/rags/.

Carolis, B. D., de Rosis, F., Andreoli, C., Cavallo, V., and Cicco, M. L. D.: 1998, The dynamic generation of hypertext presentations of medical guidelines. *The New Review of Hypermedia and Multimedia* **4**, 67–88.

Cox, R., O'Donnell, M. and Oberlander, J.: 1999, Dynamic versus static hypermedia in museum education: an evaluation of ILEX, the intelligent labelling explorer. In: S. Lajoie and M. Vivet (eds.): *Artificial Intelligence in Education: Open Learning Environment: New Computational Technologies to Support Learning, Exploration and Collaboration*. Amsterdam: IOS Press, 181–188.

Crabtree, B. and Soltysiak, S.: 1998, Identifying and tracking changing interests. *International Journal of Digital Libraries* **2**(1), 38–53.

Czarkowski, M. and Kay, J.: 2003, How to give the user a sense of control over the personalization of AH?. In: *Workshop on Adaptive Hypermedia and Adaptive Web-Based Systems, AH2003*. http://wwwis.win.tue.nl/ah2003/proceedings/paper11.pdf.

Dale, R., Milosavljevic, M., and Oberlander, J.: 1997, The web as dialogue: the role of natural language generation in hypertext. In: *Working Notes of Natural Language Processing for the World Wide Web*. AAAI-97 Spring Symposium Series, 35–43.

Dale, R., Oberlander, J., Milosavljevic, M., and Knott, A.: 1998, Integrating natural language generation and hypertext to produce dynamic documents. *Interacting with Computers* **11**, 109–135.

De Bra, P. and Calvi, L.: 1998, AHA! an open adaptive hypermedia architecture. *The New Review of Hypermedia and Multimedia* **4**, 115–139.

Dimitrova, V.: 2003, STyLE-OLM: Interactive open learner modelling. *International Journal of Artificial Intelligence in Education* **13**(1), 35–78.

Goodman, B. A., Linton, F. N., Gaimari. R. D., Hitzeman, J. M., Ross, H. J. and Zarrella G.: 2004, Using dialogue features to predict trouble during collaborative learning. *User Modelling and User-Adapted Interaction*. This volume.

Holt, P., Dubs, S., Jones, M., and Greer, J.: 1993, The state of student modeling. In: J. Greer and G. McCalla (eds.): *Student Modeling: The Key to Individualized Knowledge-Based Instruction*. Springer, 3–38.

Höök, K.: 1998, Evaluating the utility and usability of an adaptive hypermedia system. *Knowledge-Based Systems* **10**, 311–319.

Jameson, A.: 1995, Logic is not enough: why reasoning about another person's beliefs is reasoning under uncertainty. In: A. Laux and H. Wansing (eds.): *Knowledge and Belief in Philosophy and Artificial Intelligence*. Berlin: Akademie Verlag, 199–229.

Kay, J.: 1995, The UM toolkit for cooperative user modelling. *User Modelling and User-Adapted Interaction* **4**(3), 149–196.

Knott, A., Mellish, C., Oberlander, J., and O'Donnell, M.: 1996, Sources of flexibility in dynamic hypertext generation. In: *Proceedings of the 8th International Workshop on Natural Language Generation (INLG'96)*, 151–160. http://acl.ldc.upenn.edu/w/w96/w96-0416.pdf.

Kobsa, A.: 1988, A taxonomy of beliefs and goals for user models in dialog systems. In: A. Kobsa and W. Wahlster (eds.): *User Models in Dialog Systems*. Berlin: Springer Verlag, Symbolic Computation Series.

Kobsa, A., Nill, A., and Fink, J.: 1997, Hypertext and hypermedia clients of the user modelling system BGP-MS. In: M. Maybury (ed.): *Intelligent Multimedia Information Retrieval*. MIT Press.

Kobsa, A. and Pohl, W.: 1995, The user modelling shell system BGP-MS. *User Modelling and User-Adapted Interaction* **4**(2), 59–106.

Lee, M. and Wilks, Y.: 1996, An ascription-based approach to speech acts. In: *Proceedings of the 16th Conference on Computational Linguistics (COLING-96)*. Copenhagen.

McCoy, K. F.: 1988, Reasoning on a dynamically highlighted user model to respond to misconceptions. *Computational Linguistics* **14**(3), 52–63.

McCoy, K. F.: 1989, Generating context-sensitive responses to object-related misconceptions. *Artificial Intelligence* **41**(2), 157–195.

McKeown, K. R.: 1985, *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge, England: Cambridge University Press.

Milosavljevic, M., Tulloch, A., and Dale, R.: 1996, Text generation in a dynamic hypertext environment. In: *Proceedings of the 19th Australian Computer Science Conference*. Melbourne.

Mittal, V. O. and Paris, C. L.: 1993, Automatic documentation generation: the interaction between text and examples. In: *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI'93)*. Chambery, France.

Moore, J. D.: 1995, *Participating in Explanatory Dialogues*. Cambridge, MA: MIT Press.

Moore, J. D. and Paris, C. L.: 1993, Planning texts for advisory dialogs: capturing intentional and rhetorical information. *Computational Linguistics* **19**(4), 651–694.

Nielsen, J.: 2000, *Designing Web Usability: The Practice of Simplicity*. New Riders Publishing.
Paris, C. L.: 1993, *User Modelling in Text Generation*. London: Francis Pinter Publishers.
Reiter, E.: 1990, Generating descriptions that exploit a user's domain knowledge. In: R. Dale, C. Mellish, and M. Zock (eds.): *Current Research in Natural Language Generation*. London: Academic Press, 257–285.
Schuster, E., Chin, D., Cohen, R., Kobsa, A., Mörik, K., Sparck Jones, K. and Wahlster, W.: 1988, Discussion section on the relationship between user models and discourse models. *Computational Linguistics* **14**(3), 79–103.
Sowa, J.: 1984, *Conceptual Structures: Information Processing in Mind and Machine*. Addison Wesley.
Starling, A.: 2002, Usability Testing in Practice. Technical report. http://wdvl.internet.com/Authoring/Design/UsabilityTesting/.
Taylor, J., Carletta, J., and Mellish, C.: 1996, Requirements for belief models in cooperative dialogues. *User Modelling and User Adapted Interaction* **6**.
Weber, G. and Brusilovsky, P.: 2001, ELM-ART: an adaptive versatile system for web-based instruction. *International Journal of Artificial Intelligence in Education* **12**(4), 351–384.
Weiner, J. L.: 1980, BLAH, a system which explains its reasoning. *Artificial Intelligence* **15**, 19–48.
White, M.: 1998, Designing dynamic hypertext. In: *2nd Workshop on Adaptive Hypertext and Hypermedia*. Held in conjunction with Hypertext'98, Pittsburgh, USA.
Wills, G. B., Heath, I., Crowder, R., and Hall, W.: 1999, User evaluation of an industrial hypermedia application. Technical report, M99/2, University of Southampton. http://www.bib.ecs.soton.ac.uk/data/ 1444/html/html/.
Zukerman, I. and George, S.: 2004, A probabilistic approach for argument interpretation. *User Modelling and User-Adapted Interaction*. This volume.
Zukerman, I. and McConachy, R.: 1995, Generating explanations across several user models: maximizing belief while avoiding boredom and overload. In: *Proceedings of 5th European Workshop on Natural Language Generation (EWNLG-95)*.

Author's vitae

Kalina Bontcheva received her master degree in Computer Science from the University of Sofia, and her Ph.D. in Computer Science from the University of Sheffield. Since 1999 she has been a research fellow (promoted from research associate) on a number of European and UK-funded projects in the areas of natural language generation, information extraction, and knowledge technologies. Her contribution is based on work carried out as part of her Ph.D. research as well as current projects.

Yorick Wilks is Professor of Computer Science at the University of Sheffield, head of the Natural Language Processing group, and director of the Institute for Language, Speech and Hearing (ILASH). His research interests include information extraction, dialogue systems, and machine translation. He is member of the UKs Engineering and Physical Sciences Research Council College of Computing and a fellow of the American Association for Artificial Intelligence and the European AI Association.