A Knowledge Based Approach to Support Learning Technical Terminology*

Vania Dimitrova[1], Darina Dicheva[2], Paul Brna[1], John Self[1]

[1]*Computer Based Learning Unit, University of Leeds, Leeds LS2 9JT UK*
E-mail: {J.A.Self, P.Brna, V.Dimitrova}@cbl.leeds.ac.uk

[2]*Faculty of Mathematics and Informatics, Sofia University*
5 James Bouchier St.1164 Sofia, Bulgaria
E-mail: darinad@fmi.uni-sofia.bg

To appear in the Proceedings of the Third Joint Conference on Knowledge Based Software Engineering, Slovakia, September, 1998

Abstract. In this paper we discuss the structure of a domain in a terminological area and some aspects related to using it for computer-based intelligent instruction. The most significant aspect of the suggested terminological model lies in its effective applicability for tutoring. We consider two basic instructional issues: (1) organising the pedagogical process and guiding instruction; (2) building interactive student models. The first issue concerns the work done in the ITELS project whereas the second one presents a research proposal to develop an interactive diagnostic tutor to help students learn technical terminology. We argue that a conceptual graphs terminological knowledge base is useful to fulfil the expert model requirements of a traditional intelligent tutoring system. It provides knowledge allowing the system to take strategic decisions about the instruction including its focus and content. Moreover, conceptual graphs can be used for the design of an interactive diagnostic tutor. They provide a graphical external representation of the student model as well as a medium for system-learner communication and thus appear useful for solving some problems in collaborative diagnosis.

1. Introduction

Most of the presently developed terminological knowledge bases support the user's exploration of the subject area and function as terminological dictionaries and domain encyclopaedias (c.f. [9], [14]). Typically they generate explanations from the terminological knowledge base (KB). However, they do not offer instruction and generally are not concerned with any educational aspects. An intelligent tutor, based on its expert model, should be able to take strategic decisions on the instructional focus and content. It should incorporate appropriate mechanisms for extracting knowledge from the KB necessary for completing the main pedagogical activities, including: answering questions, suggesting feedback, generating learning materials, diagnosing the student's current knowledge and adapting the instruction to the learner.

In this paper we address some aspects related to the use of a terminological KB for computer-based intelligent instruction. We illustrate our ideas on using a Conceptual Graphs (CGs) terminological knowledge base which we have designed and built as part of ITELS - an intelligent tutoring system aimed at helping Bulgarians to learn Computer Science terminology in English [8].

The most significant aspect of the suggested terminological model lies in its effective applicability for tutoring. We consider two basic instructional issues:

- organising the pedagogical process and guiding instruction;
- building interactive student models.

We argue that a CGs KB is useful to fulfil the expert model requirements of the traditional intelligent tutoring system. Moreover, CGs can be used as a graphical external representation of the learner model as well as a medium for system-learner communication in an interactive diagnostic tutor where the learner inspects and changes the student model.

As far as our approach concerns building a terminological knowledge base and using it for educational purposes, in the next section we outline the main characteristics of a terminological domain and formalisms used for representing terminological knowledge. Section 3 describes our model for representing terminological knowledge which is based on conceptual graphs. The discussion in section 4 relates to the use of the terminological KB for intelligent instruction. In the last section we point out some directions for future work.

2. Terminological Knowledge Bases

2.1. Domain content

Galinski and Budin define 'terminology' as a structured set of concepts and their designations in a particular subject field [11] which suggests connecting terminology to conceptualisation and its explicit specification - *ontology* [12]. Fridman and Hafner discuss three different levels in the ontology content [10]:

- taxonomy (*is-a* relations);
- internal concept structure and relations between concepts;
- presence or absence of explicit axioms.

It is commonly agreed that the ontology of a domain should include a proper representation of concept type taxonomy explicating hierarchical relations between the main classes of concepts. Some ontology designers, however, disagree on the categorisation of the information presented in the taxonomy, mainly in relation to its interpretation and density. As a result, there are different approaches to concept organisation, which can be summarised in two basic trends – inclusion of all concept types in one taxonomy or having a number of small taxonomies (for more details see, for example, [11]).

An ontology typically includes more than just a taxonomy of concept types. Concepts have properties and roles associated with them and relations that link concepts to each other. Concept properties are commonly represented as the slots of concept frames whereas the roles and relations between concepts are represented by different declarations where individuals are related to classes or to each other. A deeper representation should be able to distinguish among different categorisations and to elucidate the category-subcategory distinction. Besides, there are some constraints on the values of the concepts' properties and roles which have to be represented. For these purposes sets of axioms are usually used. The axioms could be presented either implicitly in the application code or explicitly (e.g. by using first order logic or some extended logic, such as defaults).

To summarise, when designing a terminological KB one should consider formalisms which provide convenient tools for building type hierarchies and for representing both mutual relations between different types and relations between various instances. Conceptual graphs, with their formal structures and operations, appear to be a suitable formalism for constructing terminological KBs. Moreover, they are logically equivalent to first order predicate calculus [16], thus allowing for a powerful explicit representation of the basic axioms in the domain. Next in this section we discuss different formalisms for

representing and processing terminological knowledge and point out the advantages of conceptual graphs.

2.2. Representation of terminological knowledge

Representation of terminological knowledge is a central issue in terminological knowledge-based systems. Terminological knowledge representation languages inherit their general characteristics from KL-ONE [3]. They comprise two distinct components: a *general schema* concerning the classes of individuals and an *instantiation part* containing affirmations relating individuals to classes or individuals to each other [5]. These languages allow the building of semantic models which enable retrieving information from the terminological KB by employing a deductive process. In this way some basic problems of terminological knowledge processing are solved.

Recently several natural language projects have employed conceptual graphs to represent terminological knowledge bases (c.f. [1], [18]). The last authors comment on the usefulness of conceptual graphs as a formalism for providing subject area knowledge for linguistic systems and argue that conceptual graphs provide formal structures and operations suitable for representing and processing terminological knowledge.

Similar to KL-ONE, CGs can be considered as a terminological representation formalism which allows concept descriptions based on using necessary and sufficient conditions as well as structuring these concepts in a hierarchy. However, in addition CGs provide finer points for natural language processing by keeping syntactic clarity, and supplying particularly more powerful expressiveness about quantifier processing [2]. The latter concerns the individual-generic distinction which is provided by flexibility of the references. CGs also improve semantic power by including a representation of the context.

We use conceptual graphs as the formalism for building a terminological knowledge base. This representation of the domain knowledge allows for the implementation of intelligent instruction, based on diagnosing the student's knowledge and providing adaptive feedback. The structure of our knowledge base is described in section 3.

3. An Approach to Representing Terminological Knowledge: the Conceptual Graphs Model

The terminological knowledge base we present can be easily adapted to support learning in different subject areas. We have chosen the area of Computer Science for exemplifying our ideas. The exposition in this section follows that in section 2 describing the main characteristics of our domain together with the representation of terminological knowledge.

The domain knowledge is classified into topics and each topic comprises a set of subject area terms and the relations between them. The model for representation of terminological knowledge is based on conceptual graphs [17]. A conceptual graph is a kind of semantic network which comprises two types of nodes - *concepts* and *conceptual relations* - connected by directed arcs. An example of a conceptual graph representing the relations between the terms COMPILER, OBJECT PROGRAM, SOURCE PROGRAM and OBJECT LANGUAGE is shown in Figure 1. A CG editor supports the knowledge engineer in building knowledge bases in the chosen subject area assuming that he/she is responsible for the KB's completeness and consistency.

Concepts are the smallest units of the CG terminological KB. Each concept from the subject area is represented by its *concept type* and its *referent*. The former determines the typical characteristics of the concept whereas the latter helps the particular instantiation of a concept to be clearly differentiated. Conceptual relations indicate that there are certain dependencies between concepts' meanings. Sowa suggests a set of basic conceptual

relations and these are commonly used in conceptual graphs KBs [17].

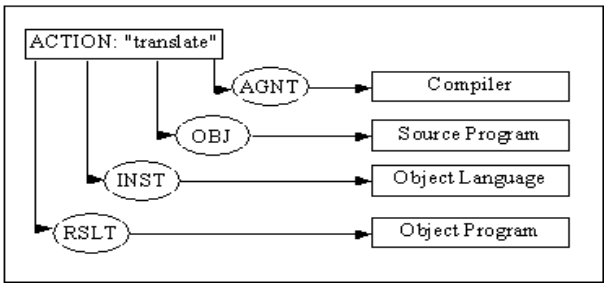

Figure 1. An example of a conceptual graph (AGNT is a denotation for *agent*, OBJ - for *object*, INST - for *instrument*, and RSLT - for *result*).

However, our experience (see section 4.2 when we mention the experiment we carried out) showed that students could not distinguish some of Sowa's basic relations (e.g. *characteristic* - *attribute*) and mixed the use of others (e.g. *agent* with *initiator* and *object*). No deep investigation is known of what would make an optimal set of basic conceptual relations which would avoid the relations' ambiguity and allow people both to express clearly their thoughts in the CG language and to understand information presented by CG. This motivated us to provide the knowledge engineer with certain flexible tools allowing him to define for himself the set of basic conceptual relations.

The type presents *typicality*. It is a denotation for a set of individuals, for example OBJECT-ORIENTED LANGUAGE = {SMALLTALK, SIMULA, C++, …}. In our KB concept types are organised in a hierarchy according to their level of generality. The hierarchy is build as a lattice where each concept type is linked with its supertypes (*genera*). For example, LOGIC PROGRAMMING LANGUAGE has a link to its *genus* DECLARATIVE LANGUAGE and is a *genus* for LOGIC OBJECT-ORIENTED LANGUAGE which is linked to its *genera* LOGIC PROGRAMMING LANGUAGE and OBJECT-ORIENTED LANGUAGE. The hierarchy permits information inheritance. Concepts which belong to the same type inherit all characteristics of this type and from its supertypes. The knowledge engineer can add new basic types to the concept hierarchy by determining their supertypes.

The relations between concepts are represented in the model by conceptual graphs. Our CG editor provides the knowledge engineer with a graphical interface allowing him to easily create/edit/delete conceptual graphs divided into conceptual relations.

Building a CGs KB we can define new types of concepts and relations in terms of the basic ones [17]. Following Sowa's model we define a new type specifying its *genus* and a defining graph (*differentia*) that allows the new type to be distinguished from the genus. Figure 2 shows the definition of the new type FUNCTIONAL LANGUAGE. When adding the new type FUNCTIONAL LANGUAGE the knowledge engineer specifies its *genus* - DECLARATIVE LANGUAGE - and links the definition to the CG presenting their *differentia*. New conceptual relations can also be defined by constructing conceptual graphs using basic relations. Figure 2 shows the definition of the new relation INIT (initiator) which could be used for explaining the difference between *agent* and *initiator* (we mentioned already that the learners failed in distinguishing these relations).

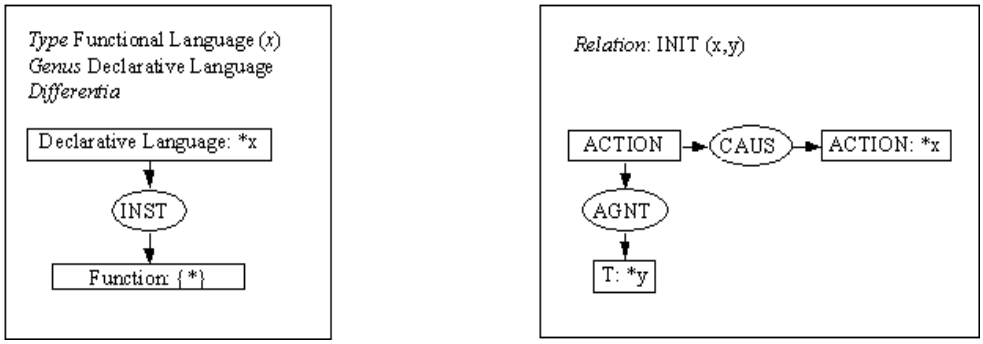

Figure 2. Defining the new concept type *Functional Language* and the new conceptual relation INIT - *initiator* (INST is a denotation for *instrument*, AGNT - for *agent* , and CAUS - for *cause*. T is an abstract type which is a supertype of every type.)

Conceptual graphs provide a convenient mechanism for extracting information from the KB. Basic conceptual operations *copy, restrict, join, simplify* [17] allow new conceptual graphs to be derived from the CGs in the knowledge base. Conceptual graphs operations supply an *inference mechanism* for retrieving specific information from the KB. With *projection mapping* [17] information related to specific properties of the concepts (e.g. attributes, characteristics, etc.) can be extracted from a CG. Conceptual operations use for retrieving information in an intelligent tutor for teaching terminology is discussed in the next section.

4. Using the Terminological KB for Instruction

The terminology model, used as an expert model in an intelligent tutoring system, supplies knowledge of how to teach technical vocabulary and to adapt the instruction to the learner. In this section we discuss two issues: extracting information from the terminological KB for instruction planning, and using the CGs model for collaborative diagnosis. The ideas related to the first issue were exemplified in the ITELS project whereas those related to student modelling motivate us to use conceptual graphs as a powerful knowledge source in the design of an interactive diagnostic tutor.

4.1. Instructional planning

An intelligent tutor should be able to take strategic decisions on the instruction including its focus and content. In the context of terminology the focus relates to determining terms to be taught next and the content – to generating feedback and teaching materials (e.g. exercises). The key idea used for instructional planning in ITELS is the conception of *term similarity* [8]. The following *hierarchical similarities* are included: supertype, subtype, terms with common parents. *Relationally similar* are those terms which are in the same CG or in a graph that could be obtained from the CG knowledge base by using the operations *restrict*, *join*, and *simplify* (e.g. TRANSLATION, COMPILER, OBJECT PROGRAM, SOURCE PROGRAM, and OBJECT LANGUAGE, see Figure 1). We assume that two terms are *near* if they are *similar* hierarchically or relationally and *far* otherwise.

- Selecting terms to be taught

Deciding which term to be focussed on next depends on the way in which the domain knowledge base is to be traversed during a tutorial session. For each selected course topic ITELS maintains a list of terms to be taught/exercised. This list initially contains all key terms of the topic and is continuously updated according to the student's answers. If the student suggests an erroneous term which is *near* to the correct answer all terms that are *similar* simultaneously to both terms get included in the list. If the erroneous and the correct terms are found to be *far,* the system assumes that there is a basic misunderstanding of their semantics and therefore it is necessary that the key terms similar to each of them be exercised.

- Generating exercises

For exercising each term from the term list, the system selects an appropriate learning block. ITELS allows the teacher to create teaching materials in advance. However, when the learner has persistent difficulties with some terms it could happen that the system cannot find an appropriate new exercise. The terminological KB could be then used as a source of knowledge for generating exercises. New exercises can be generated either directly from existing CGs or from CGs obtained by *join*, *restrict*, and *simplify* operations. The system uses templates for constructing *multiple choice* and *fill-in-the-gap* exercises.

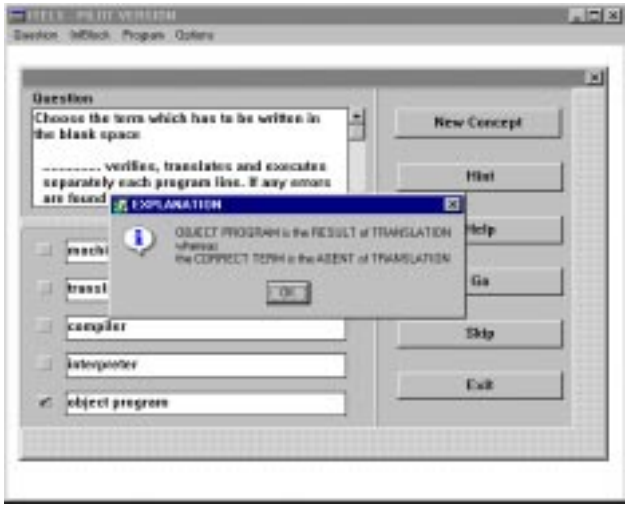

Figure 3. Feedback generated in ITELS

- Answering questions and providing feedback

The terminological KB is a good source of information for answering questions and for generating explanation (e.g. hints). In order to help, the system can supply information about all *similar* terms to the correct answer. It is feasible to do so if the student is *near* to the correct answer. In such a case the system first finds out whether the two terms - the correct one and the one suggested by the student - are *similar* and then explains their difference. The system uses *projection mapping* for generating explanations about the specific relationships between the concepts. Figure 3 shows an example of the explanation generated in ITELS when the student has suggested the term OBJECT PROGRAM instead of the correct term COMPILER. When the terms are *hierarchically near* the difference between a concept type and its *genus* could be explained by using the *differentia* from the type definition. For example, the difference between DECLARATIVE LANGUAGE and FUNCTIONAL LANGUAGE will be explained by the following sentence generated in accordance to the definition shown in Figure 2: *A functional language is a Declarative Language which operates with functions*.

4.2. Interactive diagnosing

Intelligent instruction requires adaptability, i.e. tailoring problems and information to the specific needs of each student. A model which indicates the abilities of the learner with respect to the domain being taught and a corresponding diagnostic mechanism should be provided. System diagnosis depends on the learner's ability to express himself interacting with the tutor and offering useful information to the student modelling component. Self promotes the idea to allow the learner to participate explicitly in the construction and maintenance of the student model [15]. Some advantages of such an approach have been shown - building a more accurate student model, promoting the learner's reflection, and making the student model a learning resource [6]. However, a number of problems remain, such as how to represent the learner model to the learner and how to communicate with the learner in a collaborative dialogue. We have chosen to address these issues through an exploration of collaboration based on conceptual graphs. The role of CGs is twofold: they are a tool for external knowledge representation which "open" the learner model as well as a medium for communication.

In an educational interaction both the system and the learner use their domain models to build their own beliefs about the learner's knowledge. In the previous sections we discussed the system's domain model based on CGs. We consider the system's representation of the learner's misconceptions to be based again on CGs - as an overlay model for the incomplete

knowledge and as a library of erroneous CGs representing incorrect knowledge.

Based on [7] we assume that both the learner's domain model and the learner's beliefs about his/her domain knowledge are based on semantic networks, conceptual graphs in particular. Having their own separate representations based on the same representational formalism, the learner and the system can negotiate on the learner's domain knowledge externalised with CGs (see figure 4). They would try to fill some gaps and to solve some misunderstandings in their representations of the student's domain expertise. Hence, the system and the learner collaborate in seeking to diagnose the learner's understanding by trying to achieve an agreement about the learner model.

We carried out a simple study investigating whether the students can read, build, manipulate, and communicate with CGs. The subjects were Bulgarian students chosen as users who will learn Computer Science terminology in English. The results of the study show that the students can understand information presented with CGs and, to a certain degree, express their knowledge by using CGs. The students extracted the relationships between concepts and understood the questions created with CGs. They changed conceptual graphs comparatively easily by adding a new concept to a CG. The study showed some negative details in communicating with conceptual graphs. The students faced difficulties with selecting the appropriate conceptual relation. Sometimes the students mixed the arrows between nodes, which could be regarded as a result of their lack of experience with CGs. Students failed in building a new conceptual graph. Most of them described this as a difficulty to distinguish the main concepts and the relations that held between them in an English sentence. Some students made useful comments, for example, "CGs helped me to distinguish clearly the concepts and the relationships between them" or "CGs keep language ambiguity and I am still confused with the meaning." We could roughly conclude from the study that the students would use CGs to communicate in a terminological area but with a restricted number of options.

As an external representation for the learner model, CGs provide a representation which is: (1) *natural* - being a kind of semantic networks CGs represent the conceptual knowledge in a way similar to the learner's own representation; (2) *facilitative* - being a graphical representation CGs inherit the role of diagrams to help in thinking and understanding [4]; (3) *effective* - being an external representation based on the same formalism as the system's internal representation CGs allow easy transformation between external and internal knowledge; (4) *expressive* - being a terminological language CGs allow the representation of the basic characteristics of the domain not only internally but also externally. Considered as a language for communication CGs: (1) have natural language expressiveness; (2) avoid natural language ambiguity (to a certain degree); (3) avoid language communication problems and help the learners in conceptualisation and understanding.

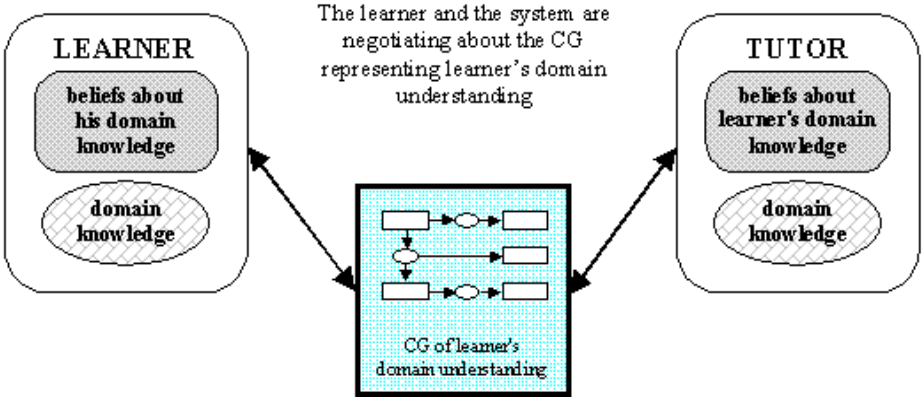

Figure 4. Collaborative diagnosing based on conceptual graphs.

5. Future work

Our future plans are to further elaborate the CGs terminological model for building modifiable learner models. This includes a deeper investigation about how learners acquire conceptual knowledge presented by CGs which concerns the granularity of the external representation as well as the way the learner has to be facilitated to express his/her thoughts in a CGs language. Another interesting question is how to organise the dialogue which we suppose to be based on the symmetrical (to a certain extent) role of both the system and the learner. We plan to apply dialogue games [13] techniques to specify the dialogue moves the system and the learner can use to engage in collaborative diagnosing. This also requires the learner to be provided with appropriate tools to allow him to participate in the dialogue.

References

[1] G. Angelova and K. Bontcheva, DB-MAT: Knowledge Acquisition, Processing and NL Generation Using Conceptual Graphs. In: P. Eklund, G. Ellis, G. Mann (eds.), Conceptual Structures: Knowledge Representation as Interlingua, Proceedings of the Fourth International Conference on Conceptual Structures. ISBN: 3-540-61534-2. Springer-Verlag, Berlin, 1996, pp. 115-129.

[2] B. Biebow and G. Chaty, A Comparison Between Conceptual Graphs and KL-ONE. In: G. Mineau, B. Moulin, J. Sowa (eds.), Conceptual Graphs for Knowledge Representation, Proceedings of the First International Conference on Conceptual Structures. ISBN: 3-540-56979-0. Springer-Verlag, Berlin, 1993, pp. 75-89.

[3] R. Brachman and J. Schmolze, An Overview of the KL-ONE Knowledge Representation System, *Cognitive Science*, **9** (1985) 171-216.

[4] P. Brna, R. Cox and J. Good, Learning to Think and Communicate with Diagrams. In: A. Blackwell (ed.), Thinking with Diagrams Discussion Papers, Portsmouth, England, 1997.

[5] M. Bucheit, F. Donini and A. Scaerf, Decidable Reasoning in Terminological Knowledge Representation Systems, *Journal of Artificial Intelligence Research*, **1** (1993) 109-138.

[6] S. Bull, Collaborative Student Modelling in Foreign Language Learning, PhD Thesis, University of Edinburgh, 1997.

[7] A. Collins and M. Quillian, Semantic Hierarchies and Cognitive Economy, *Journal of Verbal Learning and Verbal Behaviour*, **8** (1969) 7-240.

[8] D. Dicheva and V. Dimitrova, An Approach to Representation and Extraction of Terminological Knowledge in ICALL, *Journal of Computing and Information Techniologies*, 1998 (to appear).

[9] H. Felber, Terminology Manual. UNESCO, Paris, 1984.

[10] N. Fridman and C. Hafner, The State of the Art in Ontology Design: A Survey and Comparative Review, *AI Magazine*, **18** (1997) 53- 74.

[11] C. Galinski and G. Budin, Terminology. In: R. Cole *et al.*, (eds.), Survey of the State of the Art in Human Language Technology, 1995, http://www.cse.ogi.edu/CSLU/HLTsurvey/HLTsurvey.html.

[12] T. Gruber, Toward Principles for the Design of Ontologies Used for Knowledge Sharing, *International Journal of Human-Computer Studies*, **43** (1995) 907-928.

[13] J. Levin and J. Moore, Dialogue Games: Meta-Communication Structures for Natural Language Interaction, *Cognitive Science*, **1** (1980) 395-420.

[14] J. Sager, A practical course in terminology processing. John Benjamins, Amsterdam/Philadelphia, 1990.

[15] J. A. Self, Bypassing the Intractable Problem of Student Modelling. In: Proceedings of ITS'88, Montreal, 1988, pp. 18-24.

[16] J. Sowa, Conceptual Structures: Information Processing in Mind and Machine, ISBN: 0-201-14472-7, Addison-Wesley, MA, 1984.

[17] J. Sowa, Conceptual Graphs Summary. In: T. Nagle, G. Nagle, L. Gerholz and P. Eklund, (eds.), Conceptual Structures: Current Researches and Practice, Ellis Horwood, 1992, pp. 3-52.

[18] J. Wagner, R. Baud and J. Scherrer, Using the Conceptual Graphs Operations for Natural Language Generation in Medicine. In: G. Ellis, R. Levinson, W. Rich and J. Sowa, (eds.), Conceptual Structures: Applications, Implementation and Theory, Proceedings of the Third International Conference on Conceptual Structures, ISBN: 3-540-60161-9, Springer, Berlin, 1995, pp. 115-128.

The first author is partly supported by the British Overseas Research Students Awards programme.