Pre-publication draft of a paper which appeared in The Proceedings of the Joint Australian Conference on Artificial Intelligence (AI'87) pp 488-502.

Domain and tutoring knowledge in computer-aided learning

Geoffrey I. Webb

Computing and Information Studies Unit, Griffith University, Queensland, Australia

Abstract

Previous approaches to the utilisation of expertise in computer-aided learning have emphasised expertise in the subject domain. By contrast, this paper details an approach that emphasises tutoring expertise and only relies on minimal domain expertise. This has several advantages. The intelligent use of restricted domain expertise enables the detailed evaluation of the students' understanding of the domain. This permits the provision of very flexible tuition that uniquely adjusts to each student's understanding of the domain. Further, due to the restricted nature of the domain knowledge that is required, the developmental overheads associated with a lesson are minimal. Finally, the type of domain knowledge required has a well defined semantics further enabling its intelligent manipulation. The approach described is domain independent. This paper describes the general system architecture, the knowledge representation formalism used and the tutoring strategies that are employed.

1 INTRODUCTION

The traditional model of an intelligent tutoring system includes three components -

1. the domain expertise;
2. the student model; and
3. the tutoring expertise.

These interact as depicted in Figure 1. A good summary of recent approaches to such systems can be found in Sleeman and Brown (1982).

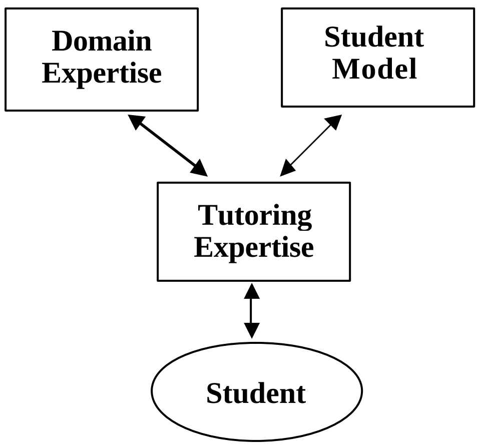

Figure 1: Components of an intelligent tutoring system

Unfortunately, in practice most research has focused on the first two of these components. Tutoring expertise has not been developed to any level of sophistication in most intelligent tutoring systems. The Domain-Analysis Based Instruction System (DABIS) attempts to remedy this situation. DABIS uses a simple form of knowledge representation for driving sophisticated tutoring strategies. This knowledge base is used for student evaluation and for global lesson management rather than for generating the low level interactions with the students.

Because the utilisation of the knowledge base is restricted, a far simpler form of knowledge representation can be used. This has two major advantages. First, the overheads of producing the knowledge base are far lower. Second, and more importantly, because there are low overheads in the utilisation of the knowledge base and because the knowledge base has a very well defined semantics, it is easier to utilise than alternative approaches and thus more sophisticated strategies for the knowledge-based management of the instructional process may be employed.

2 GENERAL ARCHITECTURE

Unlike most knowledge-based CAL systems, DABIS does not utilise its knowledge base to drive the immediate low level interactions with the students. Its knowledge base is used solely for the management of flow of control within the lesson and for student evaluation.

The immediate low level interactions with the students are generated by small instructional segments called *tutorial specifications*. These may be any form of instructional material- AI-based or set in a more traditional CAL mould.

One of the pedagogical premises on which DABIS is founded is that learning is facilitated by the examination of general principles in the context of the examination of specific examples. This is in accord with modern educational psychology (Piaget, 1970) and the pedagogical principle of *learning by doing* (Papert, 1980).

To this end, DABIS utilises *courseware abstraction* - the use of general courseware that is applied to many specific concrete examples (Webb, 1986). This has numerous advantages. Among these advantages are -

1. students have the opportunity to examine multiple concrete problems in the context of a consistent general theoretical framework;
2. the overheads of developing and maintaining the lesson are reduced as only the one general treatment need be developed. Once the general treatment is developed, the cost of specifying the specific examples is negligible; and
3. It provides a convenient framework in which to develop sophisticated analyses of the students' performance.

The knowledge that a lesson covers can be regarded as consisting of three components -

1. a set of concepts that the students must master,
2. a set of operations that the students must be able to perform; and
3. a set of discriminations that the students must be able to apply. These will be illustrated in terms of an example from elementary arithmetic.

Consider the addition of the numbers 15 and 27. In order to be able to solve this problem students must have a grasp of many different *concepts*, two primary ones being a number and an arithmetic operation. If a student does not have either of these (or numerous other) concepts s/he will not be able to tackle the problem. Given the appropriate concepts, the first step is the application of a *discrimination*- - selection of which arithmetic operation to apply. In this case the answer is addition. The application of this operation requires a further *discrimination* - selection of which low level arithmetic operation to apply. In this case the correct selection is single digit addition. Next the digits to which it is to be applied must be selected (which requires a further discrimination). The correct digits are 5 and 7. The result of correctly applying the single digit addition *operation* to these digits is 2. The next *discrimination* is to which digit the carry *operation* should be applied and which digit should be retained as part of the solution. And so the process continues.

Of the three types of knowledge - concepts, operations and discriminations, DABIS only covers operations and discriminations. It does not teach or evaluate the students' understanding of concepts. It is assumed that the students have already acquired all concepts relevant to the lesson. This should not be regarded as down-playing the need for the treatment of concepts in intelligent tutoring systems. Rather, it is simply an issue which has not yet been dealt with in this project.

3 FEATURE NETWORKS

Domain knowledge is represented in the system by feature networks. Feature networks are a variant of system networks, a formalism created by M.A.K. Halliday (1973) for the representation of linguistic systems. Despite these origins, feature networks serve as a highly versatile form of knowledge representation. Beside several lessons on linguistics, subjects already treated by the system range from the philosophy of the mind to motor mower maintenance.

The following is a highly informal description of feature networks. A formal description of feature networks and their semantics may be found in Webb (1986).

A feature network consists of a set of nodes. Each node can be one of six types - network entry points, feature choices, features, disjunctive entry conditions, conjunctive entry conditions and simultaneous branches. The differences between these types of node are detailed below.

The core elements of a feature network are its feature choices. Descending from each feature choice is a set of features. In terms of a CAL lesson, a feature choice represents the ability to perform a particular operation or discrimination. For example, for the domain of elementary arithmetic one feature choice may be the set of arithmetic operators that the student is expected to master. Each arithmetic operator, for example addition, would be a feature from that feature choice. Such a feature choice would represent a discrimination, the student's ability to determine which operator applied to a specific problem.

If a feature choice represents an operation, each feature that descends from it is the possible outcome of that operation. If the feature choice represents a discrimination, each feature represents a property that can be identified. The range of possible outcomes may be infinite. For example, a feature choice for the addition operation would have all numbers as features as any number can be the result of an addition operation.

A feature network describes the inter-relationships between the properties that can be exhibited by the *instances* of a domain. These instances are the objects that students are to examine or the exercises that they are to perform. In the domain of elementary arithmetic, the instances will be arithmetic problems that the student is to solve.

Nodes of a feature network *apply* to instances from a domain, if a feature choice applies to an instance then exactly one of its features applies to that instance. Thus, with respect to a feature choice for the addition operation and the instance *15 + 17*, the feature 32 would apply, that being the correct outcome of the operation for that instance. The feature 32 can then be regarded as the *addition outcome* property of the instance *15 + 17*.

Other than feature choices and features, feature networks have four types of nodes -

1. *the network entry point*. This applies to all instances. There may only be one per feature network and all nodes that that descend from it apply to all instances from the domain;
2. *simultaneous branches*. Many nodes may descend from a simultaneous branch. All such nodes apply to all instances to which the simultaneous branch applies;
3. *disjunctive entry conditions*. These are nodes that may descend from any number of other nodes but from which only one node may descend. If any node from which one descends applies to any instance then the node that descends from the disjunctive entry condition also applies to that instance; and

4. *conjunctive entry conditions*. These may also descend from any number of nodes and be descended from by a single node, if and only if all nodes from which a conjunctive entry condition descends apply to an instance does the node that descends from the conjunctive entry condition also apply to it.

These four types of nodes describe the relationships between the properties represented by the features.

There are two ways in whic h these relationships may be viewed. The first is as a definition of the logical relationship between the properties represented by the features. For example, if the operation that applies to an instance from the domain of two number arithmetic is addition then the first low level operation will be two digit addition and will not be two digit multiplication.

The second manner in which to view the relationships that these nodes represent is as providing an order or precedence to the epistemological elements of the domain. For example, that it is necessary to determine whether the global operation that applies to an instance is addition or multiplication (or subtraction, division, etc), before determining what low level operations apply to it (such as single digit addition, single digit multip lication, etc).

It should be noted that feature choices are the only non-deterministic nodes in a feature network. It is possible to determine for any other type of node exactly which other nodes will apply to an instance if that node applies. By contrast, exactly one of the features that descends from a feature choice will apply to any instance. The exact one may only be determined with reference to factors outside the feature network.

Figure 2 contains a feature network for the domain of the syntactic analysis of English pronouns. The network entry point (the top most edge of the network) has one child - the Pronoun Type feature choice. This indicates that this feature choice applies to all instances from the domain.

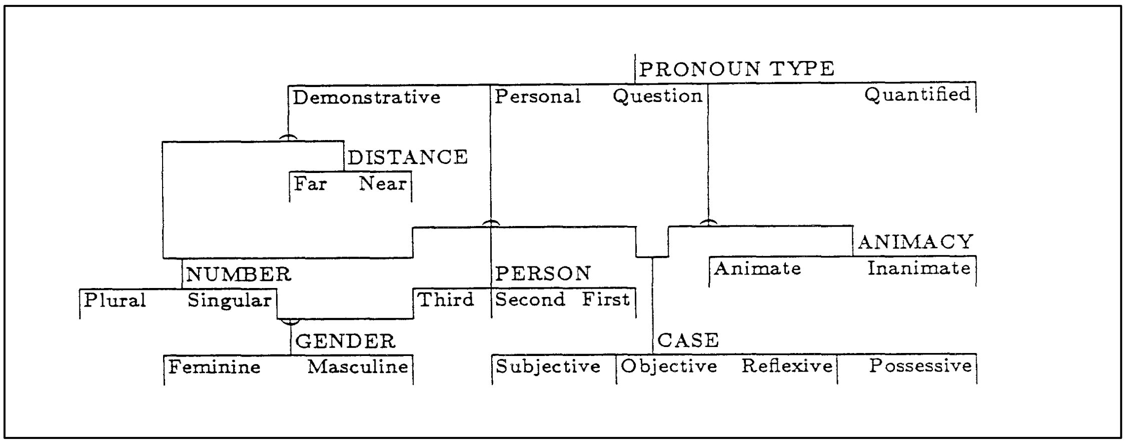

Figure 2: Feature network for the English Pronouns Domain.

The Pronoun Type feature choice has four children—the Demonstrative, Personal, Question and Quantified features. This indicates that all instances to which the feature choice applies (which is all instances in the domain) must have exactly one of those features.

A simultaneous branch descends from the Personal feature. Three further nodes descend in turn from this simultaneous branch. This means that all three of those nodes apply to all instances with the Personal feature. Two of these nodes are disjunctive entry conditions and the third is the Person feature choice. Thus, the Person feature choice applies to all and only Personal pronouns.

The leftmost disjunctive entry condition that descends from the simultaneous branch also descends from the simultaneous branch descending from the Demonstrative feature. The Number feature choice descends in turn from this disjunctive entry condition. This indicates that the Number feature choice applies to all and only Demonstrative and Personal pronouns.

The Gender feature choice descends from a conjunctive entry condition that descends from the Singular feature in the Number feature choice and the Third feature in the Person feature choice. This indicates that the Gender feature choice applies to all and only instances that exhibit both the Singular and Third Person features.

Some instances from the Pronouns Domain are displayed in Figure 3.

identifier	features
'he'	{Singular, Third, Masculine}
'she'	{Singular, Third, Feminine}
'I'	{Singular, First}
'you'	{Singular, Second}
'us'	{Plural, First}
'you' $_2$	{Plural, Second}
'them'	{Plural, Third}

Figure 3: Some instances from the English Pronouns Domain.

4 FEATURE NETWORKS IN CAL

Clearly, feature networks are not a form of knowledge representation that enable all knowledge that relates to a domain to be represented. For example, it is not possible to determine the correct outcome for any particular equation solely from the feature networks for arithmetic operations, only what the range of possible outcomes is.

Domain knowledge at this level- the level of how to perform the relevant operations - is encoded in the tutorial specifications. These are small instructional segments. Usually, each tutorial specification relates to a specific node of the network and is responsible for teaching and evaluating the students understanding of the aspect of the domain represented by that node. Tutorial specifications may be AI-based or created by any other means that a lesson author feels is appropriate.

The global knowledge base - the feature network - is only used for global flow-of-control management and student evaluation. This has the advantage that the tremendous overheads associated with the use of the detailed knowledge bases that are required for the generation of the low level interactions with the students are not carried over to the global level. With minimal computational overheads it is possible to gain all the advantages of knowledge- based student evaluation and flow of control management.

A DABIS lesson consists of three components - the feature network that provides a general description of the domain; a set of tutorial specifications that provide the actual interactions with the students; and a set of instances to which the lesson can be applied.

The instances are specified in two ways-with a general description for display to the students (usually textual, but graphics or any other means could be used) and a list of the features that apply to it. The latter is used by the system to identify the correct analysis for each instance. The reader is again referred to Figure 3 for a set of example instance specifications.

4.1 Flow of Control Management

Given the use of courseware abstraction, flow of control must be determined at two levels:

1. *extra-instance flow of control* - the selection and sequencing of instances to examine; and
2. *intra-instance flow of control* - the selection and sequencing of tutorial specifications while examining an instance.

Feature networks efficiently encode exactly the information that is required for these two purposes.

The major component of extra-instance flow of control is the selection of which of the available instances students should examine. The most appropriate types of instances for a student to examine are those with combinations of features for which the student experiences difficulty. By reference to the feature network and to the student's performance, the system is able to produce a detailed model of the combinations of features with which the student experiences difficulty. The instance descriptions can then be used to select instances with those combinations of features.

To give a simple example, if the student has a high error rate when attempting subtraction problems which require carry, but an extremely low error rate when attempting other subtraction problems, then it is going to be appropriate to concentrate on subtraction problems which involve the use of carry, rather than subtraction problems that do not. That is, if the student experiences difficulty with instances exhibiting the features *Subtraction* and *Carry* but not with instances exhibiting the features *Subtraction* and not the feature *Carry* then it is going to be more beneficial for the student to receive continued tuition on instances exhibiting the former rather than the latter combination of features.

Naturally, such a strategy requires the initial presentation of instances for which the student's performance can be examined so as to construct the initial student profile. The lesson author specifies a set of instances to be used for this purpose.

A simple strategy is used to determine the order in which instances are presented to the student. As remedial instances are selected they are simply added to the pool of those available for selection. Each time that an instance is selected for presentation, a random selection is made from the available pool. The lesson author may specify an ordering of various groups of instances (for example, ranked by difficulty) in which case later groups are slowly added to the pool of available instances as the lesson progresses. Instances for which a student does not produce a correct analysis are returned to a point slightly up the queue for the student to re-examine at a later date.

Unlike the simplistic strategy utilised for extra-instance flow of control, many different strategies may by used to manage intra-instance flow of control. As the global system has no knowledge of the interactions that occur between the student and the tutorial specifications, it is left to the latter to determine the specific flow of control strategy that should be used (within the constraints provided by the system).

The general strategy followed when examining an instance revolves around the traversal of the feature network for the lesson. During this traversal, the tutorial specification for each node that applies to the instance is invoked as that node is traversed. Thus, at feature choices, only the feature which applies to the current instance is traversed, that only those tutorial specifications that apply to an instance are invoked.

As the traversal starts with the network entry point and descends from there through the feature network, the more general aspects of an instance are examined before the more specific.

When it is completed, a tutorial specification is able to return an evaluation of the student's understanding of the application of one (or possibly more) feature choice to the current instance. Usually, this evaluation will be of the student's understanding of the current node, but this is not necessary as will be discussed below.

It also returns an evaluation as to whether the traversal of the network should continue to those nodes that descend from the current node. This allows a great deal of variation in the manner in which a network can be traversed.

Four major strategies have been developed to date—the *interrogative*, *test*, *declarative*, and *immediate* presentation strategies. Each is used for different pedagogical purposes.

4.2 The Interrogative Presentation Strategy

One powerful pedagogical method involves the utilisation of judiciously selected questions to establish a profile of a student's understanding of a domain and then the provision of tuition in those areas that the student experiences difficulty. This pedagogy is realised in the DABIS system by the interrogative presentation strategy.

Under this strategy the initial set of instances that the lesson author creates are selected so as to enable the system to produce a detailed profile of the student. To this end, they must cover the full range of significant combinations of features that are possible. As particular problems are identified, instances with the same sets of features are added to the pool of instances to be examined, thus providing tuition that directly relates to the student's individual understanding of the domain.

The major nodes to have tutorial specifications under this strategy are the feature choices.

When an instance is presented to the student, the system starts traversing the network from the network entry point. Each node that applies to the instance is traversed and as it is traversed its tutorial interaction is presented to the student.

The tutorial interactions at feature choices play two roles under this strategy. The first role is to teach the student how the feature choice should be applied to the instance under examination. The second role is to evaluate the student's understanding of this same knowledge.

These two aims are generally achieved by asking the student to apply the feature choice (or rather the operation or discrimination that the feature choice represents) to the instance. The student's performance is monitored and remediation is provided as appropriate. The tutorial specification's evaluation of the student's understanding of the application of the feature choice to the instance is formed on the basis of its analysis of the student's performance.

This evaluation is returned to the global system for incorporation in the student model.

Extremely detailed remediation can be easily provided. For any instance being examined the tutorial specification has available both a description of the correct features and those that the student has identified. Thus, DABIS is able to identify the exact nature of any error that may occur.

If a student is not able to apply a feature choice then it is likely that s/he will not be able to apply (nor even make sense of) the aspects of the domain that descend from that feature choice. For example, with reference to the English Pronouns Domain, depicted in Figure 2, if the student is unable to determine that a particular pronoun is a personal pronoun, s/he is unlikely to be able to make sense of attempting to apply the Person feature choice to that pronoun. Without a basic understanding of the instance a more complex analysis is not possible. Remediation of the more basic aspect of the domain should precede attempts to analyse more specific aspects.

As a result, when using the interrogative presentation strategy, if the student is unable to apply the feature choice to an instance then the tutorial specification returns an evaluation to the global system that further aspects of the domain that descend from the current node should not be investigated.

This does not necessarily mean that the examination of the current instance will cease immediately. The traversal of simultaneous branches prior to the traversal of the current node may mean that there are other aspects of the domain which have not yet been examined that do not depend upon an understanding of the current feature choice. In this case the system will proceed with a treatment of those nodes.

A tutorial specification may treat the knowledge that it covers in two quite different ways. An *explicit* treatment involves the discussion of the feature choice in explicit terms. Figure 4 depicts a simple explicit interrogative treatment of the Pronoun Type feature choice from a DABIS lesson on the English Pronouns Domain. This form of interaction is only appropriate if the student is expected to have a conscious mastery of both the terminology and expertise of the domain. For the English Pronouns Domain this style of interaction may be appropriate for a linguistics student

```
Is the pronoun ‘he’:

a) Demonstrative;

b) Personal;

c) Question; or

d) Quantified?

? c

No, a question pronoun can appear as the question element in
a clause. The pronoun ‘he’ cannot. An example of a question
pronoun is ‘which.’

Have another try.
```

Figure 4: A simple explicit interrogative treatment of the Pronoun Type feature choice from the DABIS Classes lesson.

However, students will frequently not be expected to have either the appropriate terminology or even a conscious mastery of the expertise covered by a domain. For example, all fluent speakers of English will have mastered the discriminations represented in the English Pronouns Feature Network but few will even realise that such a complex analysis exists. It is frequently desirable to train students without providing an explicit terminology for them to master and even without explicit reference to the knowledge or skills which are to be cultivated.

This form of teaching can be accommodated in the interrogative presentation strategy through the use of implicit treatments. An implicit treatment does not use explicit terminology or references to the knowledge with which it deals. Figure 5 demonstrates a possible simple implicit interaction for the Pronoun Type feature choice. This form of interaction may be appropriate for language students. It tests the students’ general competence in a domain rather than their formal mastery of it.

```
Select which sentence slot the word ‘he’ best fills.

A)    ‘..... is the right direction.’

B)    ‘..... is going now.’

C)    ‘..... of these is the correct answer.

13)   ‘Here are ..... of the answers.’

? c

No, a pronoun like ‘which’ that is used for introducing
questions fits much better into the slot in sentence C than
does the pronoun ‘he.’

Have another try.
```

Figure 5: A projected example of a simple implicit treatment of the Pronoun Type feature choice from the DABIS Classes lesson.

An interrogative tutorial specification may use any form of interaction with the students so long as it tests their ability to apply the appropriate domain knowledge and provides remedial instruction should they fail. The simplistic forms of multiple choice interaction that are presented in Figures 4 and 5 are intended to provide a clear demonstration of the two styles of interaction - not to in any way imply the scope or limitations of the forms of interaction that may take place.

The individual assessments of the students' application of each feature choice to each instance are analysed to produce detailed models of their understanding of the domain. This analysis is discussed further below.

4.3 The Test Presentation Strategy

It is frequently necessary to evaluate a student's understanding of a domain, either so as to determine what tuition is appropriate or for accreditation purposes. The interrogative presentation strategy allows very detailed evaluation to take place. However, it is not suitable for general student assessment as it also attempts to remediate the student's misunderstandings as it proceeds. Thus, the student's errors will alter as the lesson progresses preventing a clear profile being formed of the students understanding before the start of the lesson.

The test presentation strategy enables DABIS to be used for this purpose.

The objective of this strategy is to obtain a detailed evaluation of the students understanding of the domain. This is achieved by selecting instances and traversing the network in the same manner as for interrogative mode. The difference is that only testing of the student occurs at the tutorial specifications. Feedback on the student's performance is not provided. Thus, a detailed profile can be constructed in exactly the same manner as for the interrogative presentation strategy but without the result being biased by feedback from the system.

Given an existing interrogative lesson, a test lesson may be created simply by disabling all the feedback. In this manner the desired sophisticated evaluation can be achieved at very little cost.

It is sometimes desirable to produce large batteries of test items on a particular subject. Cooper and Lockwood (1981), Anew (1982) and Derevensky and Cartwright (1981) all provide examples of the need for this and of the large developmental overheads associated with it. The test presentation strategy allows such tests to be developed and delivered with a minimum of overheads while providing the extremely detailed student assessment possible through the intelligent use of a description of the knowledge being assessed.

4.4 The Declarative Presentation Strategy

It is not always appropriate to require students to perform analyses while examining a domain. Either the students may not yet have a sufficient understanding of the domain to enable them to perform such analyses or didactic considerations may rule against it.

In this case it is desirable to operate in a form of 'lecturing' mode in which the system performs all of the necessary analyses for the students' benefit, explaining the principles involved as they are applied. This mode of operation is accommodated by DABIS in the declarative presentation strategy.

Under this strategy the same general approach is employed for traversing the feature network as under the previous two strategies. However, the content of the tutorial specifications is quite different. Instead of requiring a student to perform the appropriate analyses, the tutorial specifications demonstrate those analyses to the student.

This results in students obtaining detailed tuition in the application of the general principles of the domain in the context of a series of concrete examples.

As no analysis is performed of the students' understanding of the principles involved, all relevant aspects of a domain are examined for each instance. A further consequence is that no profile of the students' understanding of the domain can be constructed and thus, while employing this strategy, the system is unable to identify and concentrate on the students' weaknesses. As a result, the use of this strategy is considered to be more appropriate for a general introduction to a subject area than for extended tuition.

It may also be employed for specific remedial purposes once particular student difficulties have been identified by the other presentation strategies.

4.5 The Immediate Presentation Strategy

The three preceding presentation strategies are all *domain decomposition strategies*. That is, they all analyse the instance being examined separately in terms of each feature choice that applies. By contrast, the immediate presentation strategy involves a holistic approach to the analysis of instances.

Rather than working through the feature network applying each of the relevant analyses as each feature choice is traversed, this strategy requires the student to perform a single analysis which specifies all of the features that apply to an instance.

This is often appropriate when an implicit analysis of an instance is to be used. In such a case it is frequently not desirable to have the student separately identify each feature. Rather, the one analysis can uniquely specify the entire set of features for the instance.

Figure 6 provides an example of a possible immediate presentation strategy exercise for the domain of English Tenses. This domain consists of two feature choices, one between the features Past, Present and Future and the other between the features Simple and Continuous. The student is presented with a sentence template that contains a blank from which a word is missing. A set of possible words is presented and the student must select the most appropriate word for the given template. The word form selected uniquely specifies a feature from both of the feature choices.

```
I ..... the ball then threw it to John.

A) bounce

B) was bouncing

C) will bounce

D) will be bouncing

E) bounced

F) bouncing

Select the correct form to fill the gap in the sentence.

?
```

Figure 6: A projection of an immediate presentation mode exercise on English Tenses for the planned DABIS Tenses lesson.

No traversal of the feature network is required for this presentation strategy. Rather, a single tutorial specification is associated with the network entry point. To prevent the system from traversing the entire network, this tutorial specification indicates that no subsequent nodes should be traversed.

The system is able to construct a detailed profile of the student's comprehension of the domain in exactly the same manner as with the interrogative and test presentation strategies. Exactly the same extra-instance flow of control strategy can be utilised so that the most appropriate instances for the student's comprehension of the domain will be examined.

4.6 Student Analysis

The individual assessments of a student's performance at each feature choice for each instance is subjected to detailed analysis to construct a comprehensive profile of the student's understanding of the domain.

Previous approaches to student modelling have only been able to produce student models that include either the failure to adopt particular aspects of the knowledge base or the adoption of erroneous principles that have explicitly been described to the system. By contrast, DABIS is able to produce an analysis of the student's understanding of a domain that describes forms of error for which no explicit prior allowance has been made by the lesson author.

This is not to imply, of course, that the system can model the student in terms of principles for which it has been given no description. Rather, the system is able to determine associations between the different aspects of the domain that the student has erroneously adopted.

A detailed description of these analyses and the manner in which they are performed is contained in Webb (1987). The following is a brief summary.

The simplest form of error is where the student simply fails to understand the principles that a feature choice represents. This will be detected by the system if the student's performance at that feature choice is random.

Alternatively, the student may have a basic grasp of the feature choice but not fully understand some aspects of it. In particular, the student may over or under-generalise the principles behind a particular feature. Over-generalisation is detected if a student always correctly identifies a feature when it is present but has a tendency to also identify it when it is not present. Under-generalisation is detected if the student only identifies a feature if it is present but has a tendency not to identify it on occasions when it is present.

Another form of error relating to a single feature is a failure to understand the principles that it represents. This is detected for a feature *f* when a student correctly identifies other features from the feature choice except that instances exhibiting other features are occasionally identified as exhibiting *f* and the identifications of instances exhibiting *f* are random.

More sophisticated forms of error relate to the student's adoption of associations between aspects of the domain that do not actually exist. For example, the student may always use the carry operation after the single digit addition operation even when it is not appropriate. This is detectable by the system if the feature *A* is always identified when the feature (or set of features) *B* is present. Note, however, that reference to the feature network is necessary here to ensure that the feature *B* is not *necessarily* present for all instances with the feature *A*.

As important as the ability to determine which aspects of a domain the student does not understand is the ability to determine which are understood. The system can detect that a student understands a feature choice if it is correctly applied to all instances. Even if a feature choice is not understood, the principles represented by certain features from it may be. This can be detected if those features are always correctly identified when present and are not incorrectly identified when not present.

Finally, some errors may result from the inability to analyse specific instances, rather than from a lack of understanding of the underlying principles of the domain. Either the student may have an incorrect understanding of the instance, in which case the same set of features will always be identified for it, or s/he may simply not understand how to analyse it, in which case the features identified will vary randomly from examination to examination.

5 OVERHEADS

The knowledge-based component of DABIS has extremely low overheads. This contrasts sharply with other knowledge-based CAL systems that typically require tremendous personnel resources to develop and tremendous computational resources to operate. See for example, Clancey (1984) and Anderson and Skwarecki (1986).

The reason for the low overheads for DABIS is that the direct interactions with the student in a DABIS lesson need not be knowledge-based. Rather, the knowledge base is used for selecting which aspects of the lesson should be presented to the student and which instances should be chosen for presentation.

All that is required for these purposes are a feature network describing the domain, a description of the available instances that lists their features, and a means of relating lesson segments to the feature network.

Developing the feature network should not be too onerous for an expert in an area. All that it requires is the identification of the significant features of instances from the domain and how they interact. Once the lesson author has developed the network, a simple network editor enables the specification of a network in a matter of minutes.

The next stage is to specify the lesson. The lesson editor allows lesson segments to be explicitly linked to portions of the feature network for a lesson. Linking lesson segments to the network is trivial - it simply requires the identification of which node of the network a segment is associated with. The main overhead in lesson development occurs at this stage. It is the specification of the actual interactions to take place with the student. There is some evidence that the use of a knowledge-base actually reduces this aspect of lesson development due to the manner in which the development of a feature network provides an explicit epistemological structure for the domain on which the author can base her/his lesson (Webb, 1986).

Once the network and lesson have been developed, the instance editor enables the lesson author to specify instances, their features and any other information that a lesson may require in a straight forward manner. Again, the authoring time is negligible assuming that the lesson author has a sound grasp of the subject matter.

The largest lesson to be developed to date, the Classes lesson, took six hours and eighteen minutes to develop. This was taken up by six minutes of network editing, three hours and forty-four minutes of lesson editing and two hours and twenty-eight minutes of instance editing. The average recorded student use time for this lesson was thirty minutes and seventeen seconds. This gives a lesson preparation time to effective lesson developed time of under 13:1. This rates extremely favourably compared with alternative methods. For example, Anderson and Skwarecki (1986) claim to be producing material for their knowledge-based CAL system at the rate of forty hours of coding to one hour of remedial lesson. They claim that this is an extremely low rate compared with other systems. However, as it is remedial material that they are producing it is unlikely to be presented to many students and thus the ratio of coding to average student lesson time is presumably far greater than the figure of 40:1 that their statements initially suggest.

Finally, some errors may result from the inability to analyse specific instances, rather than from a lack of understanding of the underlying principles of the domain. Either the student may have an incorrect understanding of the instance, in which case the same set of features will always be identified for it, or s/he may simply not understand how to analyse it, in which case the features identified will vary randomly from examination to examination.

6 CONCLUSION

By the intelligent use of a simple form of knowledge representation with a well defined semantics DABIS is able to provide sophisticated tutoring with minimal overheads. This demonstrates the value of emphasising tutoring expertise rather than domain expertise in intelligent tutoring systems. Limited well defined domain expertise used effectively is far superior to extensive domain expertise employed ineffectually.

DABIS is a domain independent tutoring system. The same general system can be applied to any number of domains.

By using its description of the knowledge that the student is to be taught the system is able to automatically construct detailed profiles of the student's understanding of the domain. These profiles are then used to manage the flow of control within the lesson so that the tuition is closely tailored to the individual student's needs.

By the provision of a knowledge-based system that is not weighed down by the tremendous overheads associated with extensive domain expertise, DABIS has demonstrated that tutoring expertise can be developed in economically viable systems.

References

[1] Anderson, J.R, and Skwarecki, E. (1986). **The automated tutoring of introductory computer programming**, *Communications of the ACM,* 29:842-849.

[2] Anew, R. (1982). **A management system for foreign language tests**, *Computers and Education* 6:117-120.

[3] Clancey, W.J. (1984). **Use of MYCIN's rules for tutoring**, in Buchanan, B, B. & Shortliffe, E.H. (eds), *Rule Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*, Addison-Wesley, Reading, Mass., pp 468-489.

[4] Cooper, A. and Lockwood, F. (1981). **The need for, provision and use of a computer-assisted interactive tutorial system**, in N. Rushby (ed), *Selected Readings in Computer-Based Learning*, Kogan Page, New York, pp. 218-221.

[5] Derevensky, J. and Cartwright, G. (1981). **The use of computer-assisted testing in an introductory course of educational psychology**, in Rushby, N. (ed), *Selected Readings in Computer-Based Learning*, Kogan Page, New York, pp. 127-131.

[6] Halliday, M.A.K. (1973). *Explorations in the Functions of Language*, Edward Arnold, London.

[7] Papert, S. (1980). *Mindstorms: Children, Computers and Powerful Ideas*, Basic Books, New York.

[8] Piaget, J. (1970). *Science of Education and the Psychology of the Child*, Orion Press, New York.

[9] Sleeman, D. and Brown, J. S. (1982). (eds*). Intelligent Tutoring Systems*, Academic Press, London.

[10] Webb, G.I. (1986). **Knowledge Representation in Computer-Aided Learning: The Theory and Practice of Knowledge-Based Student Evaluation and Flow of Control**, *PhD Thesis*, La Trobe University, Melbourne.

[11] Webb, G.I. (1987). **The Differential Model of Student Understanding**, to be published.