DecSys: An intelligent tutoring system for decimal numeration

T. Boneh, **A. Nicholson**
School of Computer Science and Soft Eng.,
Monash University, 3800, Australia,
{boneh,annn}@csse.monash.edu.au

E. Sonenberg
Department of Information Systems,
The University of Melbourne, 3010, Australia,
l.sonenberg@unimelb.edu.au

K. Stacey, and **V. Steinle**
Department of Science and Mathematics Education,
The University of Melbourne, 3010, Australia,
{k.stacey,v.steinle}@unimelb.edu.au

Abstract

This technical report describes the design and construction of a teaching model in an adaptive tutoring system designed to supplement normal instruction and aimed at improving students' conceptions of decimal numbers. Distinctive features of the teaching task are: (i) the domain abounds with misconceptions; and (ii) the aim is conceptual change rather than the acquisition of new facts or arithmetic procedures. The teaching model exploits cognitive conflict and incorporates a model of student misconceptions and task performance, represented by a Bayesian network. The system allows students to work through a diagnostic test and four computer games, which are designed to develop conceptual understanding of decimals. Preliminary evaluation of the implemented system shows that the misconception diagnosis and performance prediction performed by the BN reasoning engine supports the item sequencing and help presentation strategies required for teaching based on cognitive conflict. Field trials indicate the system provokes good long term learning in students who would otherwise be likely to retain misconceptions.

Appendices to this technical report include detailed descriptions of: (A) the known ways of decimal thinking upon which the tutoring system is based; (B) detailed descriptions of the computer games with example screens; (C) a full description of the types of items presented to students by the system, which correspond to the observation nodes in the Bayesian network; and (D) a list of the hypothesis nodes used in the network.

1 Introduction

This technical report describes the design and construction of the teaching model in an adaptive tutoring system designed to diagnose and modify students' conceptions of decimal numbers. The approach is of interest because it is designed primarily to bring about conceptual change, rather than teach new facts or procedural skills. In general terms, the teaching strategy is designed to provide an individual student with a sequence of tasks that (a) engage their interest (hence the use of computer games), (b) expose them to a wide range of topic specific task types (so fully exploring their domain understanding), and (c) vary in difficulty – with "difficulty" individualised to the student (by exploiting the rich model of student misconceptions). Difficulty is varied to raise a student's awareness that they have something to learn (by including tasks they are likely to get wrong) while keeping the student involved (by ensuring a student has enough tasks that they are likely to get right).

When students come to realise that there is something wrong with their existing interpretation of a situation, *cognitive conflict* occurs, and creating such conflict in the learner is one strategy that has been recommended in the education literature for situations where students need to move from one way of thinking to another [2, 12]. An individual needs to be conscious of having conflicting views at a particular point of time to experience cognitive conflict. In our environment that conflict arises during interaction with the computer system. Teaching by creating cognitive conflict has its roots in Piagetian studies where students were assisted to appreciate conservation of quantities, for example, by working with a teacher or a more advanced peer [12]. More recently, Bell [2] incorporated cognitive conflict into a classroom teaching strategy called Diagnostic Teaching, which uses group work to stimulate cognitive conflict, followed by teacher-led class discussion to resolve the conflict and reinforce correct interpretations. Bell and colleagues [2, 27] conducted a series of studies examining teaching where conceptual change is required, one of which also involved the decimals domain, and showed that classroom teaching generating cognitive conflict was much more effective than "positive-only" teaching (which avoids students presenting misconceptions to each other and focuses on presentation of correct information only), especially when improvement was measured over the long term. They also showed that both elements of the teaching strategy were important: the group work is important to raise and discuss alternative ideas and the teacher-led discussion is important to secure the resolution.

Despite its promise, this teaching method has not been adopted as normal teaching practice, where positive-only teaching strategies dominate. In normal teaching situations, both students and teachers take little notice of questions that are not answered correctly, and teachers very frequently do not realise that all answers from a student, both correct and incorrect, may have been generated by the same erroneous thinking. In addition, during "normal" schoolwork the range of item types that students have to deal with is often quite limited, contributing to the persistence of misconceptions as students' understandings are not tested thoroughly. Our machine-based system can ensure that student errors are sufficiently frequent to make them realise there is something to be learned, by posing questions that, based on the system's current student model, the student is likely to get wrong. Further, the immediate feedback and the putative authority of the computer make it likely that students will take notice of wrong answers in a computer game. These factors should ensure the system avoids a common problem with educational games [11], whereby they often do not generate the constructive reasoning required for learning.

Our teaching model incorporates a model of student misconception and task performance, represented by a Bayesian network (BN) [16]. BNs have previous success in intelligent tutoring applications [5, 4, 13, 28]. They offer an intuitive graphical representation with efficient probabilistic algorithms for updating beliefs in the light of new evidence. In our case, the beliefs are the estimates of the probabilities that a given student has a particular misconception or will demonstrate certain behaviour when playing the games. Others have shown that user modelling with this technology can be very effective in domains where domain knowledge can be decomposed into small observable components with known interdependencies, e.g. [10]. We use a BN to model the interactions between a student's misconceptions, their game playing abilities and their performance on a range of test items. Building the student model from misconceptions (c.f. [19, 8]) rather than in terms of gaps in correct pieces of domain knowledge, (c.f. [5]), is unusual, and is only viable because of the nature of the domain and the extensive research on student understanding in the domain. The information provided by the BN reasoning engine is used for diagnosis, assessment and control of the teaching.

In Section 2 we explain some important features of the domain and describe the extensive research on student thinking in this domain that provides the basis of a well-structured student model. The creation of the adaptive system has required developing a teaching model, described in Section 3, that draws upon diagnosis of student (mis)conceptions, presents students with tasks that make them aware of their lack of knowledge and provides information to assist them in constructing correct and integrated understanding. Selecting the frequency of items to get right and to get wrong is part of the teaching model. Results from field trials of the system on individual students in clinical interview settings are given in Section 4, which indicate good success with students who have developed misconceptions through normal school instruction.

Appendices to this technical report include detailed descriptions of: (A) the known ways of thinking about decimals upon which the tutoring system is based; (B) detailed descriptions of the computer games with example screens; (C) a full description of the types of items presented to students by the system, which correspond to the observation nodes in the Bayesian network; and (D) a list of the hypothesis nodes used in the network.

2 The teaching context

2.1 The decimal numeration domain

To understand the meaning and size of numbers written using a decimal point requires good knowledge of ordering whole numbers, which has been extended with new ideas about place value and fractional quantities. Examples of decimal numeration knowledge include: (i) being able to interpret a number such as 2.845 in terms of place value in several linked ways (e.g. the "8" means 8 tenths, which is the same as 80 hundredths or 800 thousandths); and (ii) knowing the size of decimal numbers (e.g. that 2.845 is between 2 and 3, much closer to 3 and close to 2.9 etc). Many complex ideas are involved and various misconceptions are commonly observed in school age students. Many of these misconceptions arise from overgeneralising whole number knowledge and others are from drawing false analogies e.g. with fractions, often embellished with knowledge of isolated facts.

Teaching students to understand decimal numeration has the following important characteristics, which influenced the nature of the adaptive system developed. First, current human teaching does not do this well, at least in Western countries, yet it is an important topic, underlying basic understanding of numbers. Reports on the inadequacies of student understanding of decimal numbers in many Western countries have been frequent for 20 years (e.g., [7]). Our testing [22] of over three thousand students from Grades 4 to 10 (with 5383 tests) has indicated that less 70% of Year 10 students (age about 15 years) understand the numeration well enough to reliably judge the relative size of decimals. On the other hand, more than 30% of Grade 5 students (age about 10 years) exhibit strong understanding of this important concept. Expertise grows only very slowly throughout the intervening years under normal instruction in our schools, and so an intelligent tutoring approach to this important topic is of interest.

Second, as students are reaching expertise throughout the five or six middle years of schooling (and probably beyond), a machine-based system that supplements normal instruction and is available for individual students to use at different times may therefore be useful.

Third, this is a conceptual, rather than a procedural, teaching problem. One can, for example, teach students procedures for ordering decimal numbers by one of two routine procedures: by left to right comparison of the digits making up the numbers or by adding zeros to equalise the lengths and then choosing the bigger whole number. However, knowing these routine procedures will not assist students when they move to other tasks which rely on understanding the size of a decimal number such as placing numbers on a number line, rounding, identifying significant figures or interpolating when reading a scale. They may be able to say that 3.2 is less than 3.21, without knowing that the distance between them is the same as the difference between 6.34 and 6.35 and much more than the distance between 1.02 and 1.021. The teaching task is therefore described as conceptual rather than procedural because it is not a question of acquiring more skills, for comparing decimals, rounding them or placing them on number lines, etc. Instead, the task is to integrate knowledge of different aspects, and to encourage students to make decisions by reasoning from the principles of place value.

Fourth, although this is a topic plagued by misconceptions, very often students don't know they harbour them.

A prime task of any teaching strategy is therefore to make sure that students know whether or not they do really understand. Our data on students' thinking has been gathered by researchers in the classroom from carefully planned and conducted interviews; within the classroom, a teacher is rarely able to gather such information. The system has the potential to diagnose and target an individual's wrong way of thinking about decimals, using results from detailed research, providing a invaluable tool for the classroom teacher.

2.2 How students interpret decimal numeration

Students' understanding of decimal numeration has been mapped using a short test, the Decimal Comparison Test (DCT), where the student is asked to choose the larger number from each of 24 pairs of decimals [22, 23]. The pairs of decimals are carefully chosen so that from the patterns of responses, a student's (mis)understanding can be diagnosed as belonging to one of a number of classifications. These classifications have been identified manually, based on extensive research [22, 17, 18, 21, 23]. For most students, there is consistency in their responses to similar test items and some students display the same misconception over long periods of time. The crucial aspects are that misconceptions are prevalent, that students' behaviour is very often highly consistent, and that misconceptions can be identified from patterns amongst simples clues.

We describe the task of comparing two decimal numbers as a conceptual task, rather than a procedural task even though there are two apparently simple algorithms for it. One can compare digits from left to right, or add zeros to equalize length and then compare as whole numbers. For students who use the first procedure, systematic errors can arise as explained by VanLehn's repair theory [3]. However, students with misconceptions most common in the middle school years do not try to apply either procedure. Instead they conceptualise the decimal part of a number in a quite different way and so they apply procedures for other mathematical entities (e.g. fractions). For this reason, we say that the decimal comparison task tests conceptual rather than procedural knowledge.

About a dozen mutually exclusive misconceptions have been identified [22, 23]. Table 1 shows the rules the domain experts originally used to classify students based on their response to 6 types of DCT test items: High = High number correct (e.g. 4 or 5 out of 5), Low = Low number correct (e.g. 0 or 1 out of 5), with '.' indicating that any performance level is observable for that item type by that class of students other than the combinations seen above. An example of a misconception is 'whole number thinking'. Many younger students think 4.8 is smaller than 4.63 because there are 8 parts (of unspecified size, for these students) in the first number and 63 parts (also of unspecified size) in the second. However, these 'whole number thinkers' (LWH, Table 1) get many questions right, e.g. 5.736 compared with 5.62, with the same erroneous thinking. So-called 'reciprocal thinking' students (SRN, Table 1) correctly choose 4.8 as greater than 4.63 but for the wrong reason, as they draw an analogy between fractions and decimals and use knowledge that 1/8 is greater than 1/63 to decide that $4.8 > 4.63$, thus getting a high score on Type 1 items. Using the same thinking they obtain high scores on Item Type 3 and low scores on all of the others.

A brief description of each way of thinking from [26] is given in Appendix A, which also includes several ways of thinking found during the development of the present ITS. Several of the major misconceptions (represented in the fine classification) occur with variations, two of which

Table 1: Response patterns expected from students with different misconceptions. (Item types illustrated with sample comparison items)

Coarse Class	Fine Class	Item type (with sample item)					
		1 4.8 4.63	2 5.736 5.62	3 4.7 4.08	4 0.4502 0.45	5 0.4 0.3	6 0.42 0.35
A	ATE	High	High	High	High	High	High
	AMO	High	High	High	Low	High	High
	AU	High	High	.	.	.	.
L	LWH	Low	High	Low	High	High	High
	LZE	Low	High	High	High	High	High
	LRV	Low	High	Low	High	High	Low
	LU	Low	High	.	.	.	.
S	SDF	High	Low	High	Low	High	High
	SRN	High	Low	High	Low	Low	Low
	SU	High	Low	.	.	.	.
U	MIS	Low	Low	Low	Low	Low	Low
	UN	.	.	.	.	.	.

are included in the current system [20]. First, some students in several of the misconception groups think decimal numbers are less than zero (though not necessarily negative); this is called '*negative thinking*' and is tracked using the DCT and the Flying Photographer game (see below). Second, some students in various misconception groups think that whole numbers and decimals are not on the same number line and therefore do not recognise, for example, that 3.0 and 3 are equal numbers; this is called '*disjoint thinking*' and is monitored alongside the fine classification. Beyond these misconceptions and variations, other aspects of decimal knowledge are involved in playing the games and hence can be tracked (see Section 3.3 below).

The key to designing the DCT was the identification of "item types". An item type is a set of items which a student with any misconception should answer consistently (either all right or all wrong). In Table 1, an example of item type 2 is the comparison of 5.736 and 5.62 where the longer number is larger. Other items of this type are the comparisons of 8.91 and 8.7 and 1.645 and 1.11. Students in the S classifications (that is, allocated to the coarse class S) will get all of these items wrong, students in the L and A classifications will get them right. The definition of item types depends on both the mathematical properties of the item and the psychology of the learners. In practice, the definition is also pragmatic – the number of theoretically different item types can be very large, but the extent to which diagnostic information should be squeezed from them is a matter of judgement. The decisions we have made for DECSYS are documented in Appendix C.

We note that the fine misconception classifications (e.g. LWH, LZE, LRV, LU, etc.) have been "grouped" by the domain experts into a coarse classification – `L` (generally think longer decimals are larger numbers), `S` (shorter is larger), `A` (correct on straightforward items (Types 1 & 2)) and `U` (other). The `LU`, `SU` and `AU` are "catch-all" classifications for students who on their answers on Type 1 and 2 items behave like others in their coarse classification, but differ on other item types. These and the `UNs` (unclassified) may be students behaving consistently according to an unknown misconception, or students who are not following any consistent interpretation or students who are unable to reliably carry out the comparisons according to their own misconception (e.g. unable to compare the complex fraction comparisons which they may erroneously think are required).

2.3 The computer games

The computer game genre was chosen to provide students with an experience different from, but complementary to, normal classroom instruction and to appeal across the target age range (Grades 5 to 10). Each game focuses on one aspect of decimal numeration, thinly disguised by a story line. The games are described in detail in Appendix B together with example screen shots; we give here a brief description.

- In the "Hidden Numbers" game students are confronted with two decimal numbers with digits hidden behind closed doors; the task is to find which number is the larger by opening as few doors as possible. Requiring similar knowledge to that required for success on the DCT, the game also highlights the place value property that the most significant digits are those to the left. The order in which doors are opened is monitored by the system. Some students successful on the DCT do not realise at first how they can play without opening all the doors.
- The game "Flying Photographer" requires students to "photograph" an animal by clicking when an "aeroplane" passes a specified number on a numberline. This task requires an understanding of the relative size of decimals, as well as their order as required in the DCT and Hidden Numbers. This task also requires understanding of decimal numeration and can be used to contribute to diagnosis of misconceptions. For example, clinical interviews such as those reported in [14] confirm that whole number thinkers (LWH in Table 1) usually expect a number like 0.23456 to be very large and are surprised to see it close to zero.
- The "Number Between" game is also played on a number line, but particularly focuses on the density of the decimal numbers; students have to type in a number between a given pair. The main situation which produces errors is that many students (including both LWH and SRN in Table 1) are unable to insert a number between 3.46 and 3.47, as they think these are consecutive numbers.
- "DecimAliens" is a classic shooting game, designed to link various representations of the value of digits in a decimal number. For example, the 4 in the number 3.46 is to be identified as representing 4 tenths, 0.4, 4/10 as well as in more difficult representations requiring reunitising [1] as 40 hundredths, 400 thousandths etc.

These games, together with the DCT, address several of the different tasks required of an integrated knowledge of decimal numeration based on the principles of place value. Therefore, it is possible for a student to do well in one game or the diagnostic test, but do poorly in another; emerging knowledge is often compartmentalised.

3 The Teaching Model

3.1 The architecture

The distinctive features of the teaching context outlined above lead us to design an adaptive learning system based around a Bayesian network. The high-level architecture of our system is shown in Figure 1. The BN is initialised with a generic model of student understanding of decimal numeration constructed using the DCT results from a large sample of students [24]. The network is tailored to an individual student using results from an online DCT and could also be tailored to use their age (since the prevalence of misconceptions varies significantly with age). During a student's use of the system, the BN is given information about the correctness of the student's answers to different item types encountered in the computer games. The student's responses are used as evidence to perform ongoing diagnosis of student misconceptions, to predict the student's

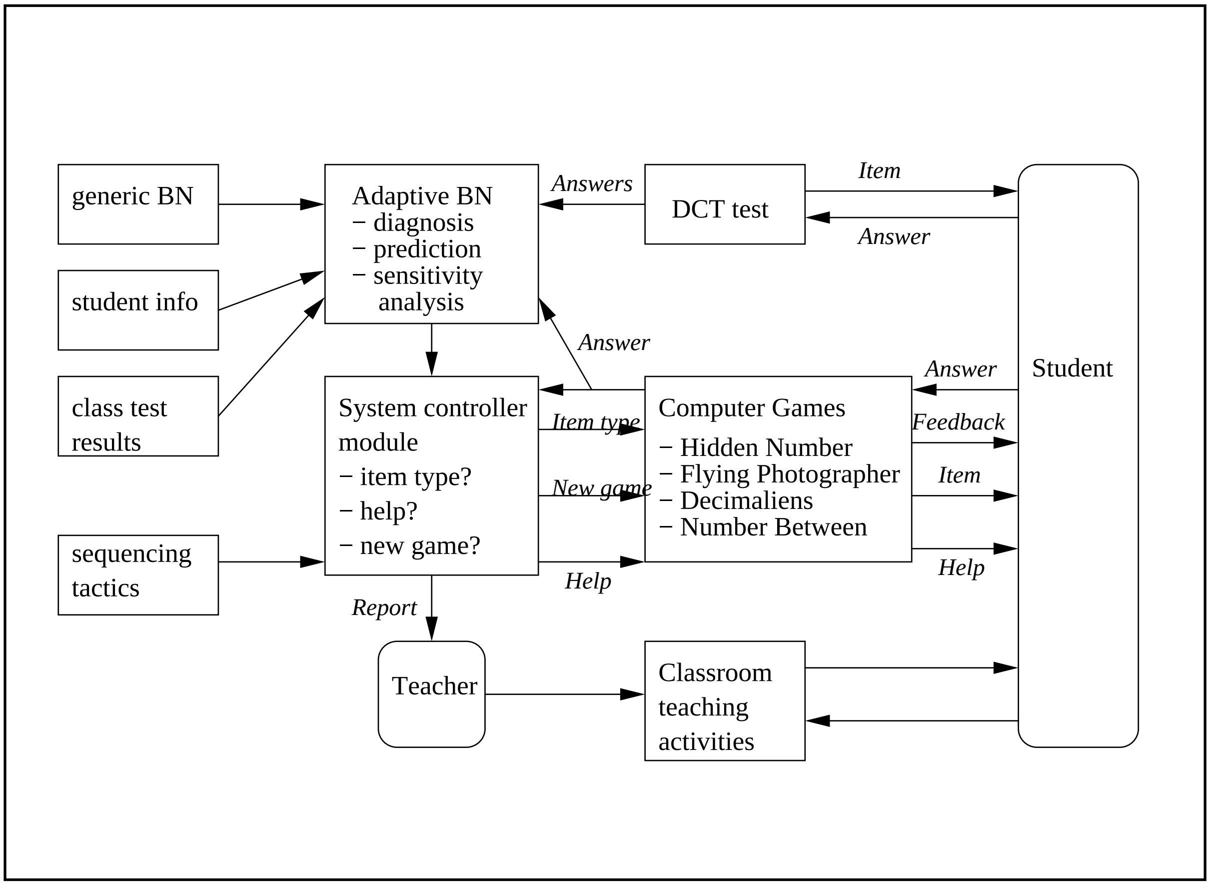


Figure 1: Intelligent Tutoring System Architecture

performance on other item types, and to assess the most useful item types to ask in order to improve on the current diagnosis. This information is, in turn, used by a controller module, together with the specified sequencing tactics (see below), to select items to present to the student, to decide whether additional help presentation is required, or to decide when the user has reached expertise and should move to another game. The controller module also makes a current assessment of the student available to the teacher, and reports on the overall effectiveness of the adaptive system.

3.2 The BN structure

The BN representing the student model was constructed through a combination of elicitation from the domain experts and automated methods. [15] gives a detailed description of the construction process, the alternative network structures and parameters considered, and the qualitative and quantitative network validation and evaluation undertaken. Here we shall describe briefly the BN being used in the deployed ITS.

The network structure is shown in Figure 2. It is comprised of four fragments that are currently disconnected. Interactions between the fragment may not be fully understood by the domain experts, or there may not be reliable connections, or the relationships may be too complex to warrant connecting. Each fragment contains *observation nodes* and *hypothesis nodes*.

The observation nodes correspond to item types (see above), and represent the student's performance with regards to this item type. A student's incorrect answers are sometimes categorised as being 'too big', 'too small' etc. Observation nodes may also correspond to student's performance in a game and not to a specific item type (i.e. `HN_gos`, `HN_nod`).

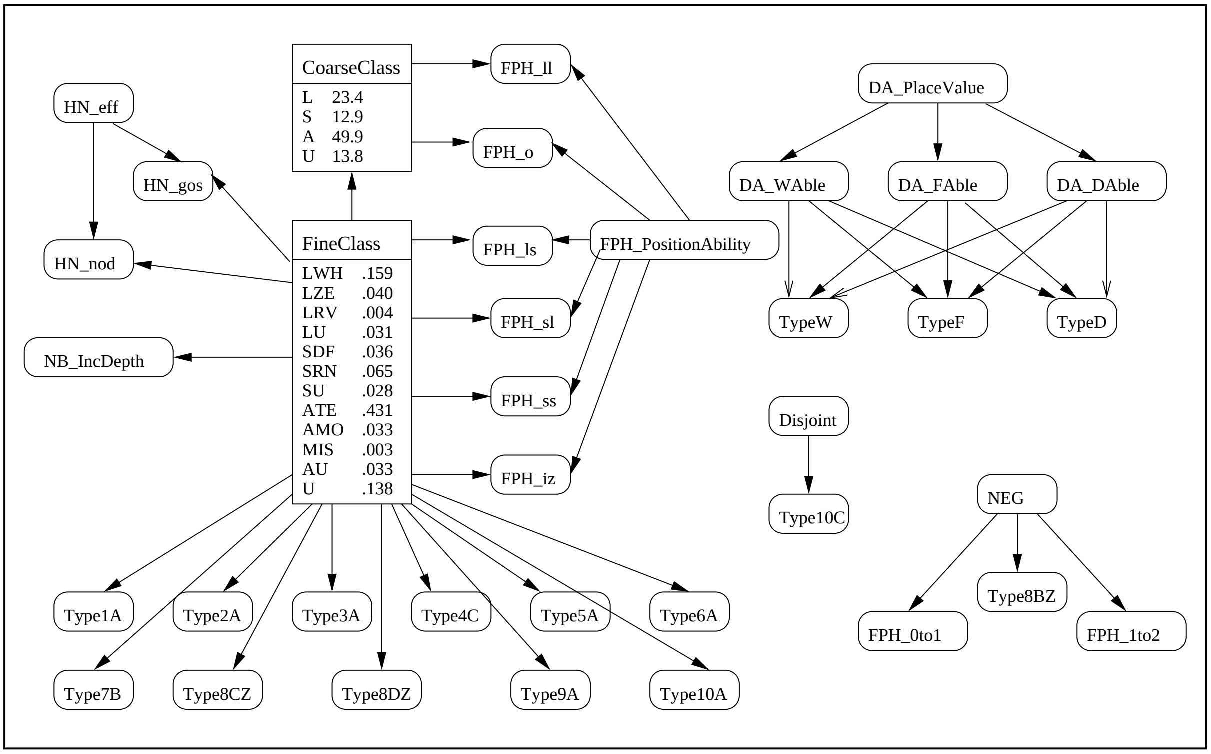


Figure 2: BN representation of the student model.

The hypothesis nodes represent the student's understanding. Hypotheses that are considered mutually exclusive by the domain experts are represented by states of a hypothesis node (e.g. `coarseClass` and `fineClass` nodes).

Arcs were added from the hypothesis nodes to the relevant item type nodes, showing qualitatively that a student's answer depends on their understanding. A conditional probability table associated with each node specifies this influence quantitatively. Entering evidence for the observation nodes should update the posterior probabilities of a student having (or not having) misconceptions. No connections were added between any of the item type nodes, reflecting the domain experts' intuition that a student's answers for different item types are independent, given their understanding. (i.e. if the hypothesis nodes are known).

3.3 Hypothesis nodes

As students play the games, they reveal information about their misconceptions and whether the misconceptions are changing. These misconceptions are represented in the BN by the so-called *hypothesis nodes*. The `coarseClass` and `fineClass` nodes are hypothesis nodes that can take the misconception types identified by the domain expert (see Table 1, columns 1 and 2) as values. The main reason for including the `coarseClass` node is for cases where students who belong to a different 'fine classification' but to the same 'coarse classification' exhibit the same behaviour on a certain item type (e.g. `FPH_ll`). The other hypothesis nodes are `NEG` and `Disjoint`, which represent the negative and disjoint thinking described in Section 2.2.

The games also reveal certain wider aspects of students' decimal understanding which is measured/tracked with other hypothesis nodes. For example, declarative knowledge of place value in decimal, fraction and verbal forms (e.g. recognising the 5 in 3.456 has place value 5 hundredths or 5/100 or 0.05) is tracked from DecimAliens with the `DA_PlaceValue` node . As another example, the order in which doors are opened in hidden numbers reveals understanding of the relative contribution of place value columns to the size of decimal numbers and so is tracked using the HN_eff node. A list of these hypothesis nodes is given in Appendix D.

3.4 Observation nodes

Most observation nodes represent the different item types, but the game nodes (i.e. FPH_X, HN_X and DA_X) represent actions during the games (except as noted above). The "HN" nodes relate to the Hidden Numbers game, with evidence entered for the number of doors opened before an answer was given (HN_nod), and a measure of the "goodness of order" in opening doors (HN_gos). Together these two nodes measure the efficiency of playing the Hidden Numbers game. The nodes (DA_X) represent observations on knowledge of place value in fractions (Type F), words (Type W) and decimal form (Type D).

3.5 Parameters

A well recognised obstacle to using numerical measures of uncertainty is the identification of appropriate prior and conditional probabilities to be used in the model. Our BN priors are available from extensive longitudinal data [22]. Estimates for the required conditional probabilities were based on the classification rules in Table 1.

All the conditional probability table entries for the item type nodes take the form of

$$P(Type = Value|Classification = C).$$

As we have seen from the domain description, the domain experts expect particular classes of students to get certain item types correct, and others wrong. A student with a particular misconception may not always behave precisely in the predicted manner – they may not be able to work out the consequences of their (wrong) interpretation, they may be trying another idea, they may accidentally hit the computer keys and so on. We model this uncertainty by allowing a small probability of a careless mistake on any one item. The probability of 0.1 is used in the current implementation.

In [15] we did experimental evaluation of different probabilities for a single careless mistake (*pcm*=0.03, 0.11 and 0.22), with the CPTs calculated in this manner, to investigate the effect of this parameter on the behaviour of the system. These number were chosen to give a combined probability for HIGH (for 5 items) of 0.99, 0.9 and 0.7 respectively, numbers that our experts thought were reasonable. Those results showed that this probability did not make much difference to the performance of the system. The current implementation uses *pcm*=0.1, a somewhat conservative choice given that empirical data from [21] showed that usually 95-98% of students answer an item consistently with their misconception. This careless mistake probability can be readily changed, for example to reflect a different emphasis on assessment compared to teaching.

More difficult than handling the careless errors in the well understood behaviour of the specific known misconceptions, is to model situations where either (1) the domain experts do not know how a student will behave (i.e. where the domain experts specified '.' for the classifications LU, SU, AU and UN in Table 1) or (2) the student's misconception does not lead to any answer. We modelled this by using 0.5 in the conditional probability table, representing a probability of 0.5 that a student in the classification will get that item correct.

Given the model and evidence about answers to one or more item types, the Bayesian belief updating algorithm then performs diagnosis; it calculates the reverse probabilities, that is, that a student with these behaviours has a particular misconception. Changes in the beliefs in the various misconceptions are in turn propagated within the network to perform prediction; the updating algorithm calculates the new probabilities of a student getting other item types right or wrong. After each set of evidence is added and belief updating performed, the student model stored within the network is updated, by changing the priors of the understanding nodes. These are the root nodes (`fineClass`, `HN_eff`, `FPH_PositionAbility`, `DA_PlaceValue`, `Disjoint`, `NEG`) that reflect a student's understanding of various aspects of decimals. Updating the network in this way, as a student plays the game (or games), allows changes in the student's thinking and skills to be tracked. The identification of the misconception with the highest probability provides the best estimate of the students' current understanding. In order to capitalise on the brief opportunity to address a misconception during a game, whenever there is doubt, it is assumed that a student is not an expert. Thus, the system does not assume a student is an expert during the games, even if this has the highest probability. However, the classification with the highest probability (expert or not) is reported at the end of the session.

3.6 Algorithm for item type selection

The major proactive task of the system controller module is to decide which item types the user should be presented with next in the game and whether the student should try the item with or without help. The algorithm for item type selection incorporates several features of the teaching model described earlier. For example, users are presented with examples of all item types at some stage during each session.

After the network has been initialised with the results of the short diagnostic test, students meeting a new game can be presented with items which they will be very likely to get correct by using the predictive capacity of the Bayesian updating algorithm described above. This is to ensure they have understood the rules and purposes of the game: if they have, they are likely to get the first questions right and if they make mistakes, it is likely that they have not understood the game. This avoids assuming that the errors were the result of domain specific misconceptions.

If the probabilities of competing hypotheses about the student's misconception classification are close, the system gives priority to diagnosis and proposes a further item to be given to the user. By employing the value of information measures provided by the Netica BN software,[2] the user can be presented with an item that is most likely to distinguish between competing hypotheses.

Perhaps the most interesting proactive use of the network is to select subsequent items in an appropriate sequence for the learner. There are several possibilities for the sequencing tactic. Human teachers generally select questions for students beginning with "easy" items and progressing to harder items, while a teaching strategy that aimed to maximise cognitive conflict might use items that the student is predicted to get wrong. Alternatively, perhaps to keep motivation higher by ensuring some success, the sequence may alternate easy and hard items. The current implementation of the system allows the comparison of three different sequencing tactics: *extreme* (presenting the hard items first), *encouraging* (presenting the easy items first), and *mixed* (presenting easy and hard items alternatively).

In the discussion above, it is important to remember that in this topic, the terms easy and hard are relative to the individual student, indicating the probability that the student will get the item correct. To a task expert, both easy and hard items may appear to be the same, involving exactly parallel processes.

A more detailed specification of the item selection algorithm is given in Figure 3. The inputs to this algorithm are:

$TaskSequencing$**:** this can take the values {Encouragement, Extreme Cognitive Conflict, Mixed}

$UnusedItemTypes(UIT)$**:** This is the set of item types that as yet have not been not introduced. In general item types are chosen more then once, however, the system will choose all item types before repeating an item type.

[2] `www.norsys.com.au`, Netica's "sensitivity to findings" function.

$ItemType_{i-1}$: this is the previously presented item type.

The algorithm uses two types of information to choose the next item type (given in algorithm as "measures used"). First, it looks at the hypotheses about the student's misconception (using the `fineClass` categories) with the highest probabilities. These beliefs are the most recently computed posterior probabilities given the evidence entered into the network to date. Second, it looks at the sensitivity to findings measures produced by the BN software. If the most probable misconceptions are less than some threshold apart (0.1 in the current implementation), then the item type is the unused item type that will have the most influence on the `fineClass` node. Otherwise, the choice is determined by a combination of the task sequencing being used, and the previous item type chosen.

Algorithm: Choose Next Item Type
Input: $TaskSequencing$, $UnusedItemTypes(UIT)$, $ItemType_{i-1}$
Output: $ItemType_i$.
Measures used:

- Current belief Bel(fineClass=$misconception$) = P($fineClass = m$|evidence to date)
- Influence of an item type $Type$ on fineClass node is given by computed "sensitivity to findings" measure, represented by $StF(Type, fineClass)$

1. Identify the states of the `fineClass` node m_1 and m_2 with the highest posterior probabilities, i.e.,

   $Bel(fineClass = m_1) \geq Bel(fineClass = m_2) \geq Bel(fineClass = m_i)$, for all other m_i

2. **If** $|Bel(fineClass = m_1) - Bel(fineClass = m_2)| < 0.1$

   then

   $$ItemType \longleftarrow \max_{type \in UIT} StF(type, fineClass)$$

   else if ($TaskSequencing$ = Encouragement) **or**
   ($TaskSequencing$ = Mixed **and** $ItemType_{i-1}$ = Extreme Cognitive Conflict)

   $$ItemType \longleftarrow \max_{type \in UIT} P(type = Correct)$$

   else

   $$ItemType \longleftarrow \max_{type \in UIT} P(type = Incorrect)$$

   end if

3. $UIT \longleftarrow UIT \setminus ItemType$

4. Return $ItemType_i$.

Figure 3: Item type selection algorithm

3.7 Designing the help-presentation tactic

In addition to raising the student's awareness that they may have something to learn, we also incorporated two forms of teaching into the games: clear feedback and visual scaffolding. It was imperative that this teaching be suitable to the game genre, not excessively interrupting the flow of the game nor requiring high reading skills. In each of the games there is immediate feedback on any answers given (correct or incorrect), with the nature of the feedback varying from game to game. In addition, when a student has made some errors, the system provides some additional visual scaffolding ('help screens'), particular to the game, to assist the student in the next item. The decision to make this type of help automatically available after an error was taken after a series of trials with students. In the trial, help was available on request. Several students were observed to deliberately avoid asking for help so that they could "do it themselves" but in the judgement of the observers this impeded their learning [14].

The details of the immediate feedback and visual scaffolding for each game are described in Appendix B, together with example screen shots.

4 Evaluation

All components of the complete system have now been field-tested. The games have been trialed with individual students holding known misconceptions and their responses and learning have been tracked (e.g. the study of the game Flying Photographer is reported in [14]). This has refined the design of the games and of the visual scaffolding and led to the decision to provide the visual scaffolding automatically. Initial trials indicate that this is a satisfactory approach, although students do not always look at the help provided or understand what it means without teacher intervention. Feedback to the students' own answers seems generally to be examined closely and so the games provide this in all cases. There is clearly some conflict between wishing to diagnose a students' thinking and wishing to teach, but in most circumstances, the latter takes priority so help and feedback should be offered.

The BN structure for misconception diagnosis and the conditional probabilities assigned to each event have been tested by comparing the results of by-hand diagnosis according to Table 1 with the diagnosis by the system on the results of 2432 students who had completed the DCT. The results showed from 80-90% agreement between the network and the by-hand classification, depending on the particular parameter setting (see [15] for details). It is important to note here that the by-hand classification is only a best-guess of what a student is thinking – it is not possible to be certain of the "truth" in a short timeframe.

The complete system has also been field tested with 25 students in Grades 5 and 6, who had persistent misconceptions after normal school instruction [6, 9]. Population data [22] indicates that students of this age can be expected to have misconceptions. Students played in pairs so that the observer could, without intervention, monitor their thinking as revealed by their conversations as they played the games. More specifically, students worked with a partner (almost always with the same misconception) for up to 30 minutes, without adult intervention. The observer recorded their conversations, which were linked to computer results and analysed to see where students learned or missed learning opportunities and how cognitive conflict was involved. Long term conceptual change was measured by re-administering the DCT about three weeks later.

Ten students tested as experts on the delayed post-test, indicating significant progress. Seven other students demonstrated improvement while eight retained their original misconception. There were some instances where students learned from the visual scaffolding of the help screens, but active teacher intervention seems required for most students to benefit fully from these. In a classroom situation, this is easy to achieve. Teachers could demonstrate the games to the whole class before individual use and could also draw students attention to the visual scaffolding. In the trials, no explanations were given to the students: the games stood alone.

Very frequently, students learned by observing and discussing with their partners, but they did not always learn the same things at the same time. This means that the computer diagnosis was not necessarily meaningful for both students so that the item type selection may not perform as designed for either student. This disadvantage needs to be weighed against the benefits of working with a partner.

Feedback provided by the games provoked learning in two ways. In some instances students added new information to their conceptual field, without addressing misconceptions (e.g. learned that 0 in the tenths column makes a number small, without really changing basic whole number thinking). In other instances the feedback provoked cognitive conflict and sometimes this was resolved within the session, resulting in a significant change from a misconception to expertise, maintained at the delayed post-test. The item type selection was set to alternate between "easy" and "hard" items for these field trials but this experiment indicated that it gave too many easy items. Following up an error with another hard item (either of the same and another type) may be more effective. The real-time updated diagnosis by the system of the student's thinking patterns was (generally) consistent with the observer's opinion. Discrepancies between classifications and the delayed post-test were tracked to known limitations of the DCT, which could not diagnose an unusual misconception prevalent in that class. The system has since been modified to address this.

On balance, the system seems to be a useful supplement to class instruction. By providing a wide range of item types, students' understanding is probed in a way which teachers cannot do without extremely detailed planning. Making mistakes in the games does provoke cognitive conflict in students and this can be partially resolved within the games. The system, however, does not provide the thorough instruction in place value that is necessary to produce real understanding in young students; this has to come from a teacher.

5 Conclusions

We have described the considerations behind the design of the teaching model for a system to supplement classroom instruction in the domain of decimal numeration – an important topic, not well grasped by significant numbers of students. Distinctive features of the teaching model reflect particular aspects of the domain – student misconceptions abound, there is new body of research expertise surrounding the types of misconceptions and their identification, and teaching is aimed at conceptual change.

The teaching model, based on the stimulation and resolution of cognitive conflict, includes four purpose built computer games and a diagnostic test based on the research literature. It uses a Bayesian network to identify when to provide feedback and what activities to select next. The detailed design of the teaching model draws heavily on research into student misconceptions in this domain. The keys to exploiting this research are that: item types can be identified fairly simply; one can introduce item sequencing tactics; and the ability to thoroughly explore a student's understanding by use of the full range of item types. Further, relationships between student behaviour on particular item types and their domain understanding can be embedded in the teaching model. So, as the student interacts with it, the system can adapt its presentation to the perceived needs of the individual student. Exploiting this research in a teaching approach accessible in the classroom, is arguably only feasible in a machine-delivered format because of the fine-grained pedagogical content knowledge required, and yet is likely to be best suited as a supplement to conventional instruction because of the limitations of current machine-based teaching methods.

From a modelling perspective, we found that using a Bayesian network as the underlying reasoning engine has a number of advantages. The availability of priors probabilities from large cross-sectional studies ensures the system starts with a usable generic student model. The incremental incorporation of a student's answers is achieved by adding evidence and performing belief updating.

This updating provides misconception diagnosis information, as well as predictions of the student's performance on other items required for sequencing strategies. The sequential updating with each new piece of evidence means that the BN works with whatever information has been provided by the student and is always ready with an opinion about the student's understanding, regardless of when the student stops interacting with the system. The BN structure was based around fragments for the different games, which facilitated the incremental construction and testing of the model, and confirms that the technology can be used for user modelling in bigger domains, as long as the domain knowledge can be decomposed into tractable fragments with known inter-dependencies [10].

The suite of games currently implemented is not comprehensive and could be extended if desired. Integrating new games, or other activities designed to support learning, would involve modification of the underlying BN: adding nodes to incorporate key knowledge items associated with the activity, adding links to record associations and dependencies with existing nodes, and providing appropriate prior and conditional probabilities.

Design of the system has highlighted for us in new ways several aspects of teaching. Known features of the teaching task lead us to adopt the computer game genre and the desire to provoke change through cognitive conflict. The observation that item type 'difficulty' is not absolute, but dependent on the type of understanding held by a student, means that individualisation is needed for many purposes: to get students firmly started playing the games according to the rules, and to stimulate cognitive conflict while at the same time keeping motivation high.

Whereas individualisation plays an important role in what item is presented to the student, perhaps surprisingly, it plays a very small role in the teaching presented to the student. Feedback of whether an item was right or wrong and showing the right answer is the same for all students. In addition, the visual scaffolding is generally the same for all students, presenting correct information related to basic principles of decimal numeration. Whereas tasks presented need to be individualised to provoke cognitive conflict, the instructive feedback is the same for all, although given in the individualised context. For example, in Flying Photographer, it will be important that a reciprocal thinker (SRN) will need to place a number like 0.99 on the interval [0,1] and that whole number thinker (LWH) should place a number like 0.165432. However, both will benefit from seeing the same help screen (a grid of 0.1, 0.2, 0.3 etc over the interval), which shows 0.99 between 0.9 and 1 and 0.165432 between 0.1 and 0.2.

Further empirical evaluation of individual design decisions and modifying the sequencing tactics is planned. We hope that further work with the system will also shed light on generic questions concerning provision of support and feedback, and other broad issues in the design of intelligent systems for education.

Acknowledgements

Thanks to Elise Dettmann for early work on the architecture, Jenny Flynn and Sally Helm for conducting field trials, and to Brent Boerlage for help with Netica. Thanks also to Shona Archer, Caroline Condon, Janine McIntosh, Nick Nethercote, Alwyn Ngai, Nick Scott, Calvin Tromp, students in Education and Computer Science at the University of Melbourne who contributed to the development, trialling and programming of the computer games. The project has been supported by grants from the Australian Research Council.

References

[1] A. R. Baturo and T. J. Cooper. Reunitising hundredths: prototypic and non-prototypic representations. In E. Pehkonen, editor, *Proceedings of the 21st Conference of the International*

Group for the Psychology of Mathematics Education, volume 2, pages 57–64. PME, 1997.

[2] A. Bell. Principles for the design of teaching. *Educational Studies in Mathematics*, 24(1):5–34, 1993.

[3] J.S. Brown and K. VanLehn. Repair theory: A generative theory of bugs in procedural skills. *Cognitive Science*, 4:379–426, 1980.

[4] C. Conati, A. Gertner, and K. VanLehn. Using Bayesian Networks to manage uncertainty in student modeling. *User Modeling and User-Adapted Interaction*, 12(4):371–417, 2002.

[5] C. Conati, A.S. Gertner, K. VanLehn, and M. Druzdzel. On-line student modeling for coached problem solving using Bayesian Networks. In *UM97 – Proceedings of the 6th International Conference on User Modeling*, pages 231–242, 1997.

[6] J. Flynn. The effectiveness of a computer games program in teaching decimal notation. Honours thesis. Faculty of Education. University of Melbourne. 2002.

[7] A. Grossman. Decimal notation: An important research finding. *Arithmetic Teacher*, 30:32–33, 1983.

[8] R. Hagen and E.A. Sonenberg. Automated classification of student misconceptions in physics. In C. Rowles, H. Liu, and N. Foo, editors, *Proceedings of the 1993 Australian Joint Conference on Artificial Intelligence, AI'93*, pages 153–159. World Scientific, 1993.

[9] S. Helm. The use of a computer games to improve understanding about decimal notation. Honours thesis. Faculty of Education. University of Melbourne. 2002.

[10] E. Horvitz, J. Breese, D. Heckerman, D. Hovel, and K. Rommelse. The Lumiere project: Bayesian user modeling for inferring the goals and needs of software users. In *Proceedings of the 14th Conference on Uncertainty in AI*, pages 256–265, 1998.

[11] M. Klawe. When does the use of computer games and other interactive multimedia software help students learn mathematics? In *NCTM Standards 2000 Technology Conference*, 1998.

[12] P. Light and M. Glachan. Facilitation of individual problem solving through peer interaction. *Educational Psychology*, 5(3 and 4):217–225, 1985.

[13] M. Mayo and A. Mitrovic. Optimising ITS behaviour with Bayesian networks and decision theory. *International Journal of AI in Education*, 12:124–153, 2001.

[14] J. McIntosh, K. Stacey, C. Tromp, and D. Lightfoot. Designing constructivist computer games for teaching about decimal numbers. In J. Bana and A. Chapman, editors, *Mathematics Education Beyond 2000. Proceedings of the 23rd Annual Conference of the Mathematics Education Research Group of Australasia*, pages 409–416. Freemantle, 2000.

[15] A. Nicholson, T. Boneh, T. Wilkin, K. Stacey, L.Sonenberg, and V. Steinle. A case study in knowledge discovery and elicitation in an intelligent tutoring application. In *Proceedings of the 17th Conference on Uncertainty in AI*, pages 386–394, Seattle, 2001.

[16] J. Pearl. *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufmann, San Mateo, Ca., 1988.

[17] L. B. Resnick, P. Nesher, F. Leonard, M. Magone, S. Omanson, and I. Peled. Conceptual bases of arithmetic errors: The case of decimal fractions. *Journal for Research in Mathematics Education*, 20(1):8–27, 1989.

[18] C. Sackur-Grisvard and F. Leonard. Intermediate cognitive organization in the process of learning a mathematical concept: The order of positive decimal numbers. *Cognition and Instruction*, 2:157–174, 1985.

[19] D. Sleeman. Mis-generalisation: an explanation of observed mal-rules. In *Proceedings of the 6th Annual Conference of the Cognitive Science Society*, pages 51–56, 1984.

[20] K. Stacey, S. Helme, and V. Steinle. Confusions between decimals, fractions and negative numbers: A consequence of the mirror as a conceptual metaphor in three different ways. In M. van den Heuvel-Panhuizen, editor, *Proceedings of the 25th Conference of the International Group for the Psychology of Mathematics Education*, volume 4, pages 217–224. PME, 2001.

[21] K. Stacey and V. Steinle. Refining the classification of students' interpretations of decimal notation. *Hiroshima Journal of Mathematics Education*, 6:49–69, 1998.

[22] K. Stacey and V. Steinle. A longitudinal study of childen's thinking about decimals: a preliminary analysis. In O. Zaslavsky, editor, *Proceedings of the 23rd Conference of the International Group for the Psychology of Mathematics Education*, volume 4, pages 233–241, Haifa, 1999. PME.

[23] V. Steinle. *Changes with age in students' misconceptions of decimal numbers.* PhD thesis, University of Melbourne, 2004.

[24] V. Steinle and K. Stacey. The incidence of misconceptions of decimal notation amongst students in grades 5 to 10. In C. Kanes, M. Goos, and E. Warren, editors, *Teaching Mathematics in New Times, MERGA 21.*, pages 548–555. MERGA, 1998.

[25] V. Steinle and K. Stacey. Exploring the right, probing questions to uncover decimal misconceptions. In L. Bragg, C. Campbell, G. Herbert, and J. Mousley, editors, *Proceedings of the 26th Annual Conference of the Mathematics Education Research Group of Australasia*, volume 2, pages 634–641. MERGA, 2003.

[26] V. Steinle and K. Stacey. Grade-related trends in the prevalence and persistence of decimal misconceptions. In *Proceedings of the 27th Conference of the International Group for the Psychology of Mathematics Education*, volume 4, pages 259–266. PME, 2003.

[27] M. Swan. *Teaching Decimal Place Value: A Comparative Study of 'Conflict' and 'Positive Only' Approaches.* Shell Centre for Mathematical Education, Nottingham Univ., 1983.

[28] K. VanLehn and Z. Niu. Bayesian student modelling, user interfaces and feedback: a sensitivity analysis. *International Journal of AI in Education*, 12:154–184, 2001.

A Known ways of decimal thinking

This appendix describes known ways of thinking associated with each classification (adapted from [25]). A table (Table 2) linking DECSYS to code in the Steinle thesis [23] follows.

Task expert (ATE): Correctly completes the task of comparing decimals. Various correct and incomplete strategies might be used singly or in combination throughout the test. Students may "fully understand" or rely on rote rules.

Money thinking (AMO): Treats first 2 decimal places like the (whole) number of cents (or cm) so unsure when these are equal. Sees decimals as discrete. Difficulties with Type 4 (eg 4.45/4.4502) as both numbers are like $4.45, and then may truncate or round or guess. (Will be coded as AMO if consistently chooses incorrectly on these items, else falsely classified as ATE or AU.)

First digits only thinking and Failed left to right thinking (classified as AMO or ATE): First digits only makes comparison only with the first digits (one or two places) after the decimal point but strategy fails when these are equal. Failed left to right thinking refers to an incomplete version of a correct procedure. When comparing 3.26 with 3.2618 digits from left to right, the '1' needs to be compared with the "invisible zeros" at the end of the 3.26 to successfully complete the algorithm. Like money thinking, these students are generally correct but need to guess when their procedures fail.

Whole number thinking (LWH): Treats decimal portion as another whole number, so $4.8 < 4.75$ as $8 < 75$. Two variations: Numerator focussed thinking chooses $0.53>0.006$ as $53>6$, while String length thinking chooses $0.53 < 0.006$ as 006 has 3 digits and 53 has two.

Column overflow thinking (LZE): Correctly chooses $4.03 < 4.2$ as 3 hundredths $<$ 2 tenths, but incorrectly chooses $4.8 < 4.75$ as 8 tenths $<$ 75 tenths. The presence of a zero indicates the need to use a new "name". Generally correct on equal length decimals.

Zero-makes-small thinking (LZE): Uses whole number thinking (LWH) with an additional (isolated) fact that a zero after the decimal point 'makes the number smaller'. Correctly chooses $4.03 < 4.2$ as the zero in 4.03 makes it small, but incorrectly chooses $4.8 < 4.75$.

Reverse thinking (LREV): Believes right-most columns have largest place value, so compares from the right-most column first, either due to mishearing column names (hundredths as hundreds etc) OR an overgeneralisation of symmetry (larger value columns on outside). So, $4.8 < 4.75$ as 5 hundred 7 tens $>$ 8 tens, and $0.42 < 0.35$ as $2 < 5$.

Denominator focussed thinking (SDF): Reads a one digit decimal as a number of tenths, a two digit decimal as a number of hundredths etc and then incorrectly generalises the fact that 1 tenth is greater than 1 hundredth to 'any number in the tenths is greater than any number in the hundredths'.

Place value number line thinking (classifies as any S): Works from false analogy between place value columns and number lines. Moving from far left to far right, numbers are indicated in this sequence, numbers in the hundreds (3 digits) then tens (2 digits) then single digit numbers (including 0 which is a 'whole number') then single digit decimals (tenths), two digit decimals (hundredths), three digit decimals (thousandths) etc. Thinks 0.6 less than zero, because zero is in the ones column and 0.6 is in the tenths.

Reciprocal thinking or Negative thinking (SRN): Treats decimal portion as another whole number but then as something analogous to the denominator of a fraction (reciprocal) OR as a number 'on the other side of zero' or less than zero (not necessarily negative!). So, 4.82 < 4.3 as 1/82 < 1/3 or as -82 < -3. 'The larger it looks the smaller it is'. Generally makes incorrect judgements on equal length decimals.

Misread/misrule (MIS): Students who get nearly all questions wrong. Either a task expert (ATE) who misreads the instructions, circling the smaller number throughout the test, OR a student following a correct comparison rule (like ATE) but then believing that there is a reversal in size (by loose analogy with fractions and negative numbers). Support for misrule being widespread is that two thirds of these students select 1.3 > 0.86, whilst being incorrect on almost every item with the same integer part.

Table 2: Matching of DecSys codes to Steinle thesis codes.[23]

Way of thinking	Steinle thesis Code	DecSys Code
task expert	A1	ATE
money thinking	A2	AMO
unclassified A	A3	AU
whole number thinking* & decimal point ignored thinking	L1	LWH
zero makes small thinking & column overflow thinking	L2	LZE
reverse thinking	L3	LRV
unclassified L	L4	LU
denominator focussed thinking & place value number line thinking	S1	SDF
reciprocal thinking & negative thinking	S3	SRN
unclassified S	S4	SU
unclassified	U1	UN
misread, misrule, mischievous	U2	MIS

* whole number thinking = numerator focussed thinking + string length thinking

B Game Descriptions

The introductory screen for the ITS is shown in Figure 4. The screen for the DCT test is shown in Figure 5.

Figure 4: Introductory Screen for ITS

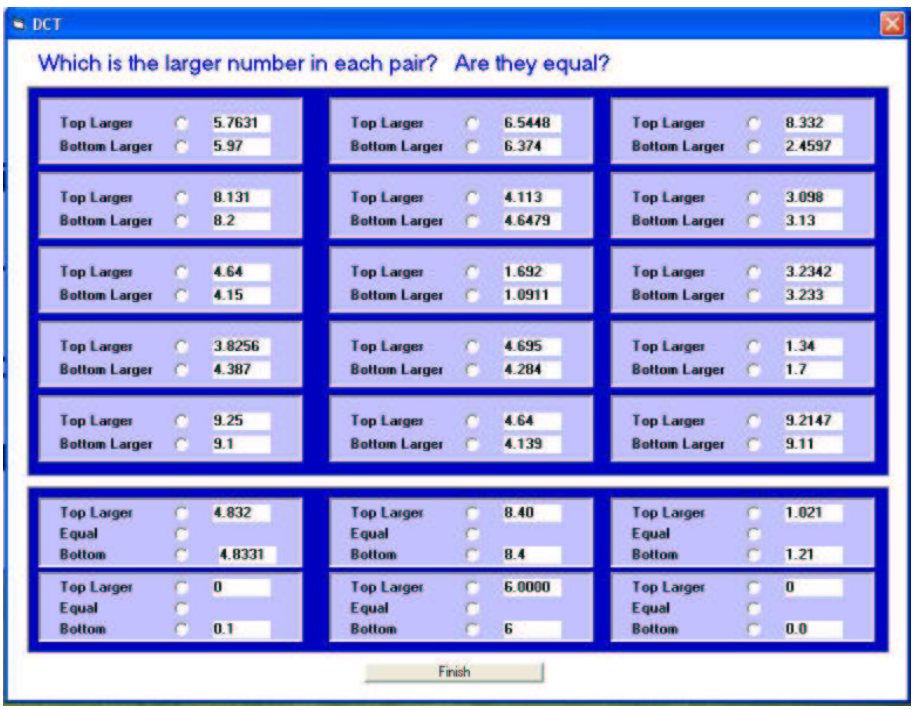


Figure 5: Decimal Comparison Test Screen

B.1 Hidden Numbers

The Game In the "Hidden Numbers" game students are confronted with two decimal numbers with digits hidden behind closed doors.

The Task To find which number is the larger by opening as few doors as possible.

Story Line The game starts with $2000 credit to the player. It costs $50 to open a door; a correct answer rewards $500 and a wrong answer causes the loss of $500. The player aims to win as much money as possible.

Game Ends The Game ends either because the player runs out of money (the player loses) or the player answers all question-types correctly (the player wins). In both cases the game alerts the player that the game is finished and what the reason is.

Updating the BN An incorrect answer generates the same question type (different numbers) and invokes the scaffolding mechanism. In case of a correct answer or two successively incorrect answers the network is updated and the next question-type is chosen (see "Choosing question types").

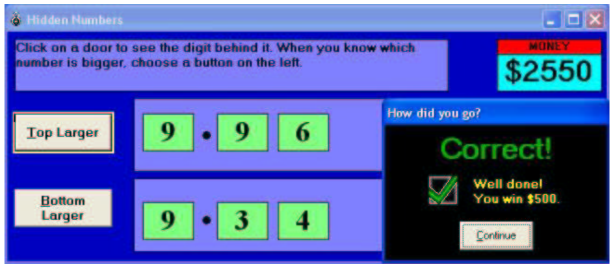
Feedback Is given after the student choses the larger number.

If correct, the message "`Well done! You win $XXX`" is displayed. Otherwise, the message "`Bad luck. You lose $XXX`" is displayed. When the player runs out of money the message is "`Oh dear. You ran out of money`". Appropriate feedback is also displayed in cases where the player opens too many doors: "`Save Money, Open Less Doors`" or too few doors: "`Be Careful, Open More Doors`".

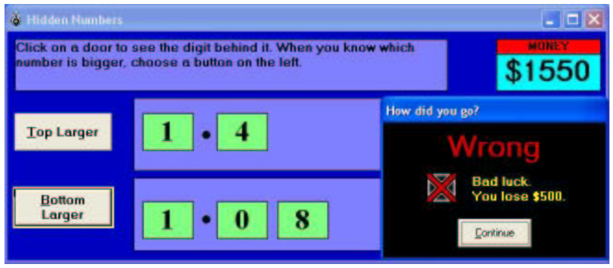
Visual Scaffolding When a wrong answer is chosen a visual scaffolding appears for the next item: the non-fractional-units are automatically set to "zero". By opening any door, the player revels a digit. A horizontal line is displayed, which illustrates graphically the number of fractional-units the digit represents. The length of the line represents the place value of the digit. For example: if the number is 0.3X, then a line three tenths units long is displayed. Opening any additional door leads to the display of additional and differently coloured units to the displayed line. The length of the units is relative to the position of the reveled digit in the number.

Knowledge Requiring similar knowledge to that required for success in the DCT, the game also highlights the place value property that the most significant digits are those to the left.

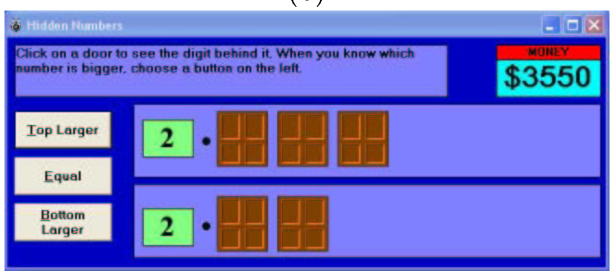
Item Types Updates item types: `Type1A`, `Type2A`, `Type3A`, `Type4C`, `Type5A`, `Type6A`, `Type7B`, `Type8CZ`, `Type8DZ`, `Type9A`, `Type10A`. In addition: `HN_nod` and `HN_gos`.



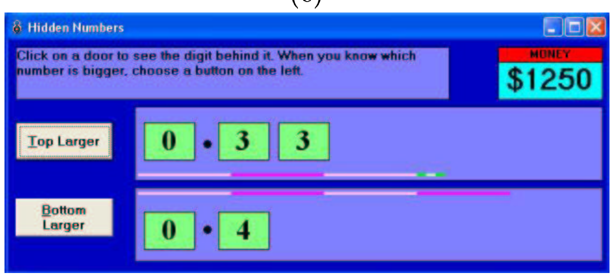
(a)



(b)



(c)



(d)

Figure 6: Hidden Number Screens: (a) Feedback to correct answer (b) Feedback to wrong answer (c) Additional equal option provided when appropriate (d) Visual Scaffolding (Help Screen)

B.2 Flying Photographer

The Game In the Flying Photographer game students are confronted with a decimal number and a number-line.

The Task To find the correct position of a given number on the number-line and click when an "aeroplane" passes over this position.

Story Line The player is a photographer who wants to photograph animals from a plane. Each animal's position is given as a decimal number between a pair of endpoints. When the player thinks that they are above the animal they click the camera button to take a photo.

The players have three fly-overs per animal. The numbers of good and bad photos are counted to measure the player's skills.

The game has a practice mode in which the players can practice before they actually start the game.

Game Ends The game ends when the player asks to quit or when they reach a certain number of correct answers. In the latter situation, the player gets two more numbers to position on the number-line. The number-line is then divided into four sections: the one on the left is not marked; the next point, to its right is marked 0, the next is 1 and the right unit is marked 2. The first number the player gets to position is in the form "`0.x`" and the second is "`1.x`" (where `x` is a digit between 1 and 9). This additional 0,1,2 numberline is presented to identify the so-called negative thinkers.

Once the game ends the player is notified how well they did, that is, how many good photos they took, and their skill is represented as the percentage of good photos.

Updating the net If the players click the right position (correct answer) or finish three fly-overs with the wrong answer, then the network is updated accordingly.

Feedback Is given every time the player clicks the camera. If the clicked position is incorrect a sad red face appears at the marked point. A correct position is marked by a happy green face. In case of a correct answer the player receives the message "`Well done.`" and in case of three wrong answers the message is "`Wrong. Look at the green triangle.`"

Visual Scaffolding A wrong answer (three wrong passes) invokes the Visual Scaffolding mechanism for the next number. Another number-line is displayed, marked with intermediate numbers.

Knowledge This task requires understanding the relative size of decimal numbers and can be used to contribute to the diagnosis of misconceptions.

Item Types Updates item types: `FPH_ll`, `FPH_ls`, `FPH_sl`, `FPH_ss`, `FPH_iz`, `FPH_o`. In addition: `FPH_1To2` and `FPH_0To1` (when the number is shown at a different scale of 0, 1, 2, to identify negative thinkers).

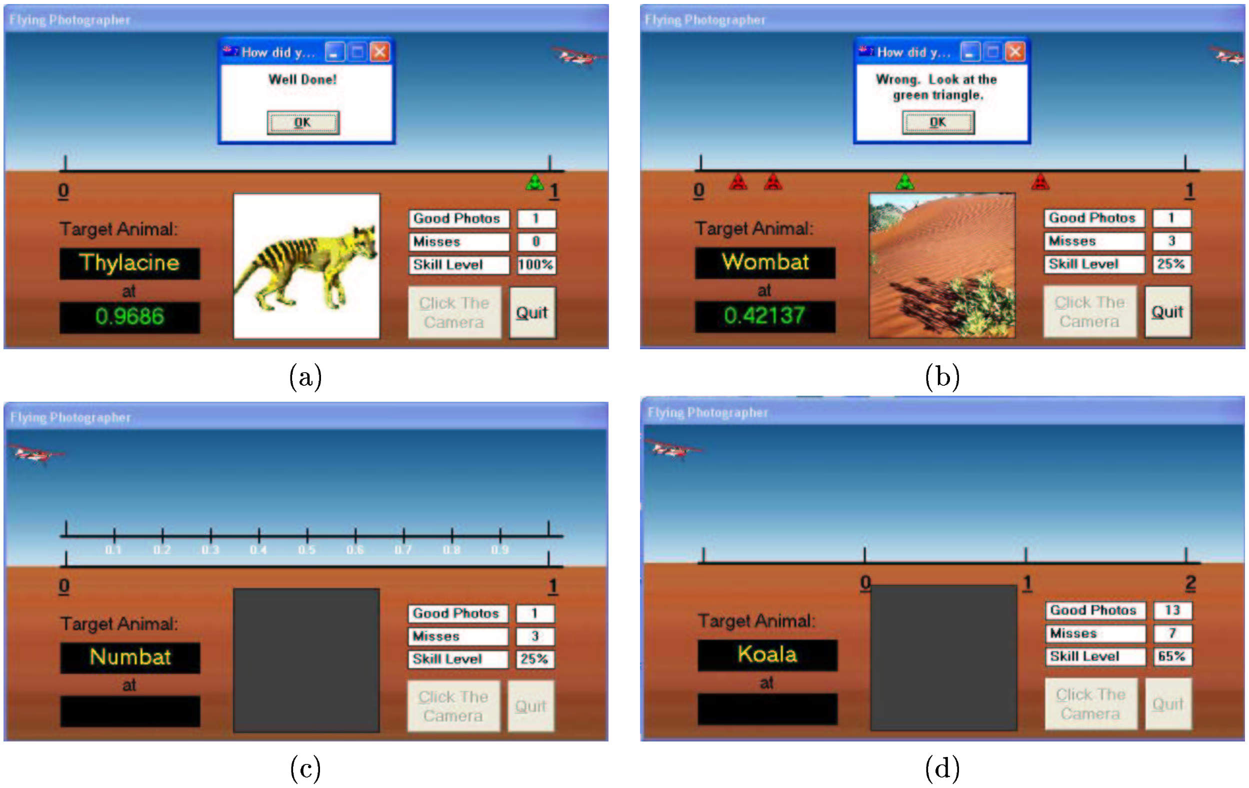


Figure 7: Flying Photographer Screens: (a) Feedback to correct answer (b) Feedback to wrong answer (c) Visual Scaffolding (Help) (d) Different scale for number line, used to identify negative thinkers.

B.3 Number Between

The Game In the Number Between game students are presented with a number-line and two endpoints.

The Task To find a number between two endpoints.

Story Line A number-line with endpoints is displayed. The player is asked to type in a decimal number between the two given endpoints. The player gets three chances to give the correct answer. When the player makes a correct choice this number randomly becomes one of the new endpoints and the player needs to choose a new number that is situated between the new endpoints. With each correct number a coloured triangle appears. This process can be repeated up to six times, and the aim of the player is to get a colourful hexagon before the game ends.

Game Ends The game ends if the player cannot find a correct answer (three wrong answers were given) or if six correct answers were found.

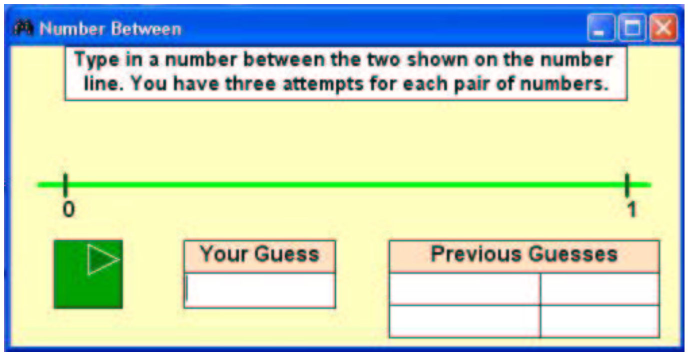
Updating the net This game tests whether students can choose a correct number when it requires adding another digit representing a new 'decimal level' to the number. For example, the student may be required to find a number between 0.3 and 0.4. If the player does not find a correct number, the game ends and the network is updated with a wrong answer. The network is updated with a correct answer only if the player has had to answer correctly by adding a digit that represents a new 'decimal level', *and* also if the visual scaffolding was not displayed, because in this game the visual scaffolding is giving away the answer.

Feedback Is given after any number entered. If the number is wrong, the number is displayed with a corresponding remark "`Too Big`" or "`Too Small`". If the number is correct it becomes one of the endpoints (randomly) and a coloured triangle is displayed. When the game ends the players are notified how successful they were, what level they reached and how many wrong answers they gave.

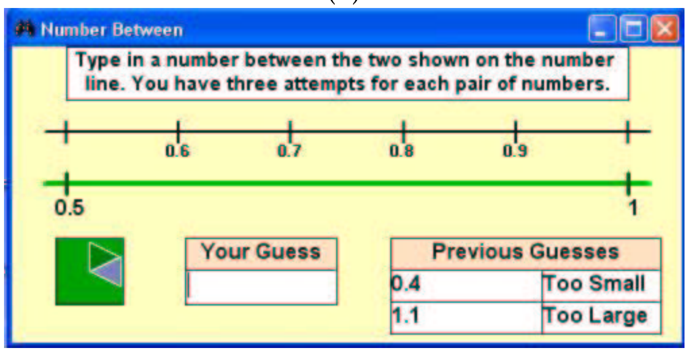
Visual Scaffolding A second incorrect answer invokes the Visual Scaffolding mechanism, which is similar to the one in the Flying Photographer game (see above). Another number-line is displayed, marked with intermediate numbers.

Knowledge This game tests knowledge of the ability to reunitise decimal numbers, e.g. to see 0.3 as 0.30 (three tenths as 30 hundredths) and 0.4 as 0.40, so that the number 0.35 is understood to be between them.

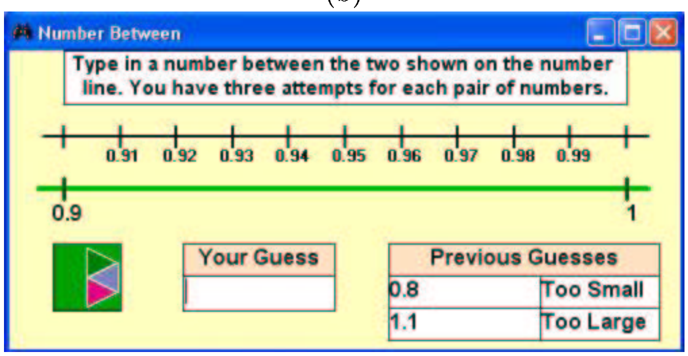
Item Types Updates item types: `NB_IncDepth`



(a)



(b)



(c)

Figure 8: Number Between Screens: (a) Initial Screen (b) Visual Scaffolding (Help) – another number-line marked with intermediate numbers. In this case, the answer is being given to the student, so a correct answer does not result in an updating of the network. (c) More Visual Scaffolding (Help)

B.4 DecimAliens

The Game In the DecimAliens game students are confronted with a decimal number that is comprised of marked and unmarked digits. The unmarked digits form a decimal number. There are four options to choose from, of which only one matches the number which the unmarked digits form. The four options can be given in three different representation styles: Decimal (0.4), Fraction (4/10) or Word fraction (4 tenth).

The Task To choose (by clicking) the correct number among four options.

Story Line Several fleets of the dreaded DecimAlien ships are headed towards Earth, intent on its destruction. The player is the last line of defence and this is his/her mission:

The DecimAlien fleet forms a decimal number and the unshielded (unmarked) alien will form part of that number. In order to destroy this unshielded part, the player must match it with the correct number, one of four options. By choosing the correct answer the unshielded DecimAlien ship is hit. When choosing a wrong answer, the player's ship is hit.

There is a practice mode and three different levels of skills. The levels differ by the length of the numbers (level 1: xx.xx, level 2: xx.xxx, level 3: xxx.xxx), the complexity of the partitions (level 1: only one alien is uncovered at a time, level 2: only one alien or two adjacent (possibly straddling the decimal point) are uncovered at a time, level 3: only one, two or three aliens, adjacent or not (possibly straddling the decimal point) are uncovered at a time).

Game Ends The Game ends because the player's ship is hit four times (the player loses), or the player successfully destroys a certain number of alien fleets (e.g. 7) (the player wins).

Updating the net The mechanism for choosing the question type to be updated is different in this game in that the question types are not chosen by the network but randomly by the program. All choices are displayed in a DECIMAL style for the first number, in a FRACTION style for the second number, in WORD style for the third, and in MIXED style (each option can be displayed in a different style that is chosen randomly) for the fourth. The style is chosen randomly from {DECIMAL, FRACTION, WORD, MIXED}, for any additional fleet number. The style of the correct answer determines the question type to be updated.

There are three question type nodes: `TypeD`, `TypeF` and `TypeW` representing the different representation styles. Each of these nodes has four states: one Correct state and three Incorrect states (i.e. for `TypeD` the four states are: Correct, Incorrect and the chosen wrong answer is a DECIMAL, Incorrect and the chosen wrong answer is a FRACTION, and Incorrect and the chosen wrong answer is a WORD).

Any choice the player makes updates the network accordingly.

Feedback The chosen number is displayed in a decimal style in green if correct and in red if incorrect. Missiles are shot towards the digits that are represented by the chosen number. If the number is correct then the missiles hit the unshielded digits and destroy them. If the number is incorrect the missiles hit shielded digits, which are not destroyed, and missiles are shot from the unshielded digits and hit the player's ship.

Visual Scaffolding The visual scaffolding provides an explanation of the digit as a number and the name of the column the digit is in (e.g. 6 tens, 6 hundreds or 6 tenths). This is displayed in rotated text vertically under each digit. In level 1 the visual scaffolding is static. In levels 2 and 3 the player can change the visual scaffolding, by clicking a button (the red help ball), to display the number in different forms e.g. (6 tens can be 60 units or 6000 hundredths.)

Knowledge Conversion between decimal and fractional forms with reunitising required at higher levels.

Item Types Updates item types: `TypeW`, `TypeF`, `TypeD`.

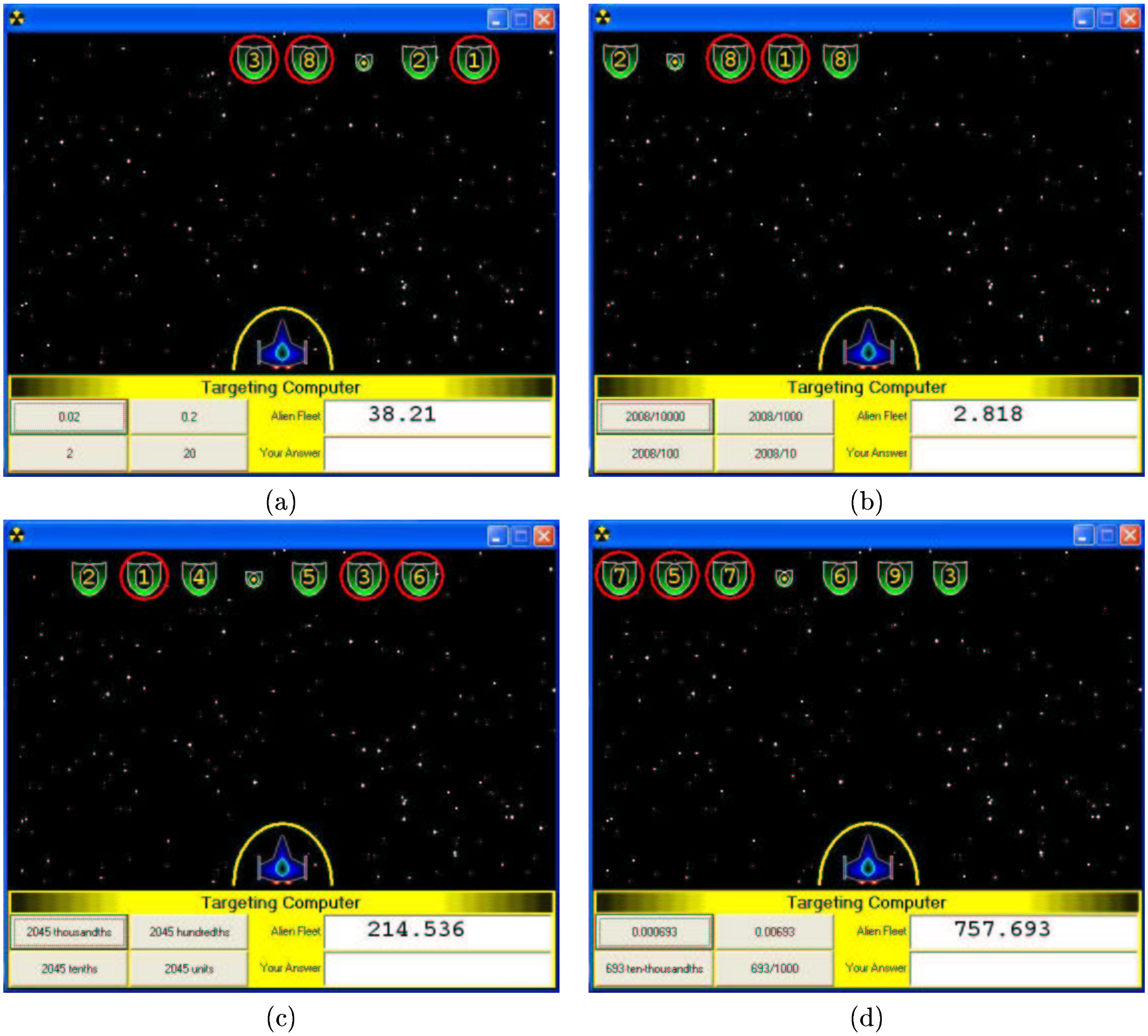
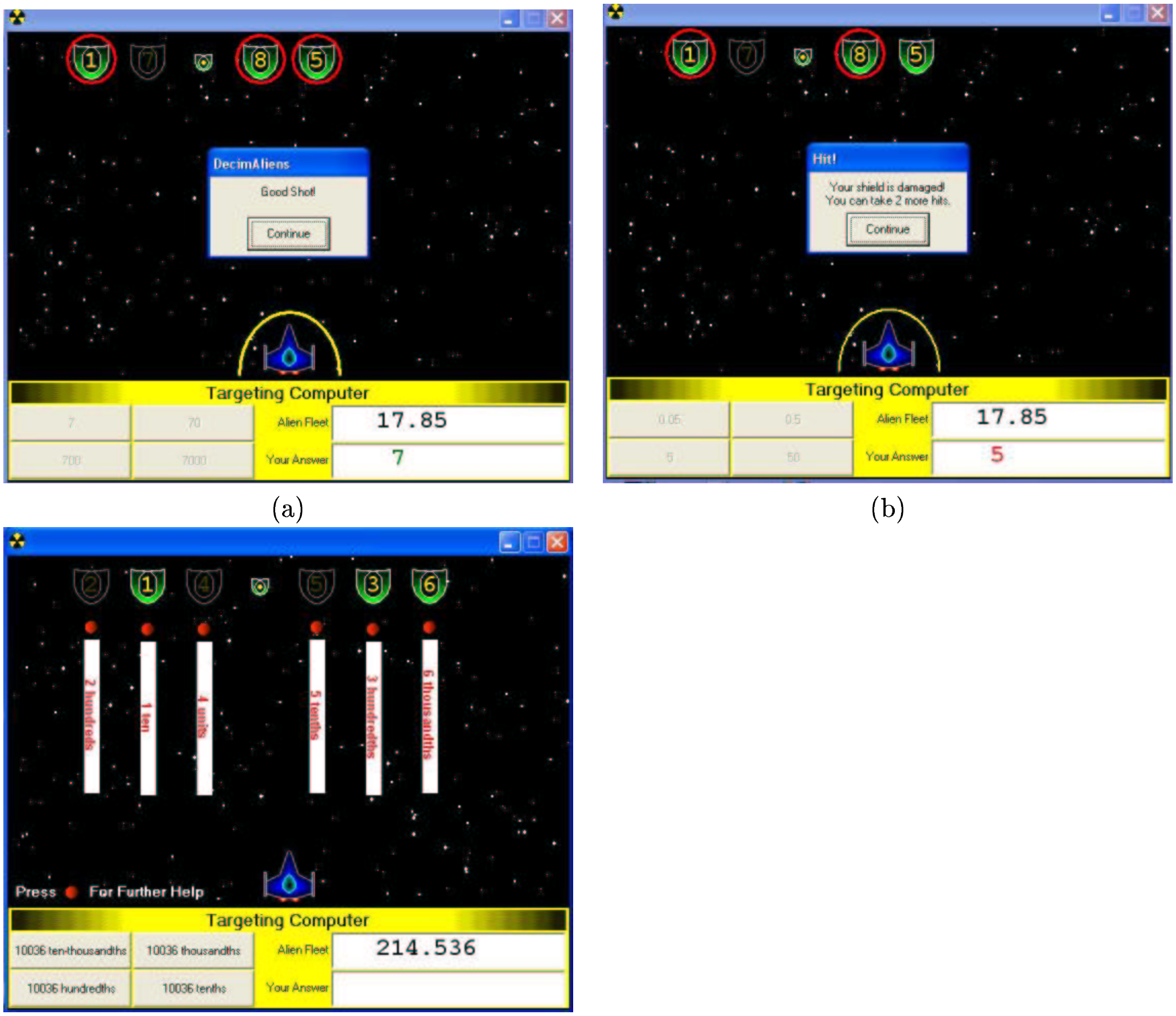


Figure 9: DecimAliens Screens: (a) decimal representation (b) fraction representation (c) word representation (d) mixed representation.



(a) (b) (c)

Figure 10: DecimAliens Screens: (a) Feedback Screen to correct action (b) Feedback Screen to being "hit" (c) Visual scaffolding (Help). Note that clicking the red ball will cycle through the different representations (e.g. 5 tenths as 50 hundredths, etc.).

C Item Types

This appendix describes the different item types that are part of the system, theses are the observation nodes. Other item types will be included in the next versions of the system.

The following notations are used for every pair of decimals :

- The larger number is of the form: $A_0.A_1A_2A_3\ldots A_m$.
- The smaller number is of the form: $B_0.B_1B_2B_3\ldots B_n$.
- If only some of the digits are of importance then this is denoted by $A_0.A_1X$. This means that X represents one or more digits, however all digits in X have the same rule (e.g. they are between 1 and 9, between 1 and 4, equal 0 etc.). It was decided not to use zeros in some cases because it can affect the item type.

In addition, the following rules apply:

- The number of digits in all numbers (m and n) is ≤ 5. If help (i.e. visual scaffolding) is required, $m \leq 3$ and $n \leq 3$.
- The units (A_0, B_0) can be any number between 1 and 9. In most cases $A_0 = B_0$. If help is required they are 0.
- The digits that are in the tenths in the numbers (A_1, B_1) are between 1 and 9 and if help is required they are between 1 and 4.

Item Types for DCT							
Item Type	**Example**	**Games**	**Rule-** left is:	**Distinguish**	**Who get wrong**	**Node & states**	**Affect nodes**
`Type1A`	4.8X/4.73Y	DCT, HN	Shorter	L–S	L	`Type1` T/F[3]	`FineClass`
Desc.	$A_1 > B_1 + 1$, B_2 free or $A_1 = B_1 + 1$ & $B_2 < 5$, $X, Y \in [1, 9]$, keep m<n						
Type2A	5.73X/5.6Y	DCT, HN	Longer	L–S	S	`Type2` T/F	`FineClass`
Desc.	$A_1 > B_1 + 1$, or $A_1 = B_1 + 1$ & $B_2 < 5$, $X \in [1, 9]$, $Y \in [1, 4]$, keep m>n						
Type3A	3.72/3.074	DCT, HN	Shorter	LWH–LZE	LWH, LRV	`Type3` T/F	`FineClass`
Desc.	B_1=0, $A_1 < B_2$, $X, Y \in [1, 9]$, keep m<n						
Type4C	6.512X/6.51	DCT, HN	Longer	AMO–ATE	S, AMO (Equal)	`Type4` T/F	`FineClass`
Desc.	$A_1 = B_1 < 9$, $A_2 = B_2 < 9$, $A_3 < 5$, $B_3 < A_3$, keep m>n						
Type5A	1.4X/1.2Y	DCT, HN	Equal	SRN-SDF	SRN	`Type5` T/F	`FineClass`
Desc.	$A_1 > B_1$, $X, Y \in [1, 9]$, keep $m = n$						
Type6A	1.42X/1.27Y	DCT, HN	Equal	LRV-LWH	LRV, SRN	`Type6` T/F	`FineClass`
Desc.	$A_1 > B_1 + 1$, $A_2 < B_2$, $A_3 < B_3$, keep $A_i < B_i$, $m = n$						
Type7B	4.3X/2.84Y	DCT, HN	Shorter	LWH	LDPI[4]	`Type7` T/F	`FineClass`
Desc.	$A_1 < B_1$, $X, Y \in [1, 9]$, keep $m < n$						
Type8BZ	0.6/0	DCT	Longer	`NEG`	`NEG`	`Type8BZ` T/F	`NEG`
Desc.	$A_0 = B_0 = 0$, $A_1 \in [1, 9]$						
Type8CZ	0.6/0.00X	HN	Shorter		L	`Type8CZ` T/F	`FineClass`
Desc.	$A_0 = B_0 = B_1 = B_2 = 0$, $A_1 \in [1, 9]$, $X = 0$						
Type8DZ	0.6/0.0	HN	Equal		SRN	`Type8DZ` T/F	`FineClass`
Desc.	$A_0 = B_0 = B_1 = B_2 = 0$, $A_1 \in [1, 9]$						
Type9A	1.21/1.021	DCT, HN	Shorter	LWH	LWH, SRN (Equal??)	`Type9` T/F	`FineClass`
Desc.	$B_1 = 0$, $B_i = A_i - 1$						
Type10A	3.40X/3.4Y	DCT, HN	Longer		L, S	`Type10A` T/F	`FineClass`
Desc.	$A_0 = B_0$, $A_1 = B_1$, $A_2 = 0$, $X = Y = 0$, keep $m > n$						
Type10C	3.0X/3	DCT	Longer	`Disjoint`		`Type10C` T/F	`Disjoint`
Desc.	$A_0 = B_0 \in [1, 9]$, $A_1 = 0$, $X = 0$, keep $m > n$						
Type10CZ	0.0X/0	DCT	Longer	`Disjoint`		`Type10CZ` T/F	`FineClass`
Desc.	$A_0 = B0 = 0$, $A_1 = 0$, $X = 0$, keep $m > n$						

[3] True/False
[4] Not yet part of the system.

Ability Observations for the Hidden Numbers Game							
Item Type	**Example**	**Games**	**Rule-left is:**	**Distinguish**	**Who get wrong**	**Node & states**	**Affect nodes**
Number of doors						`HN_nod` F/R/M[5]	`FineClass` `HN_eff`
Updating Method	If opened the optimal number of doors +/- 1 door, it is **JustRight** If opened less, it is **TooFew** and if opened more it is **TooMany**						
Good Order Score						`HN_gos` T/F	`FineClass` `HN_eff`
Updating Method	If the first and the second door to open are the 10ths, it is a **good** order score If the first door to open is one of the last ones, it is a **bad** order score Else it is a **Mix**						

[5] TooFew/JustRight/TooMany

Items for the Flying Photographer game (probabilities of generation)						
Item Type	**Example**	**Games**	**Distinguish**	**Who get wrong**	**Node & states**	**Affect nodes**
LargeLong	.7X, .8X, .9X	FPH	S	S, Too Small	FPH_ll B/R/S[6]	CoarseClass FPH_PositionAbility
Desc.	$A_0 = 0$, A_1=7(25%), 8,9(75%), $X \in [1, 9]$, m=3, 4, 5					
LargeShort	.7, .8, .9	FPH	L	L, Too Small	FPH_ls B/R/S	FineClass FPH_PositionAbility
Desc.	Only 0.7(25%) or 0.8, 0.9(75%)					
SmallLong	0.1X, 0.2X	FPH	L	L, Too Big	FPH_sl B/R/S	FineClass FPH_PositionAbility
Desc.	$A_0 = 0$, A_1=1or2, $X \in [1, 9]$, m=3, 4, 5					
SmallShort	.1, .2, .3	FPH	S	S, Too Big	FPH_ss B/R/S	FineClass FPH_PositionAbility
Desc.	Only 0.1, 0.2 (75%) or 0.3(25%)					
InitZero	0.XY	FPH	L	L, Too Big	FPH_iz B/R/S	FineClass FPH_PositionAbility
Desc.	$A_0 = 0$, $X = 0$, 00, 000, $Y \in [1, 9]$, m=3, 4, 5					
Others	To cover all numbers				FPH_o B/R/S	CoarseClass FPH_PositionAbility
Desc.	0.00≤X≤0.29 or 0.70 ≤ X ≤ 0.99 (33%) 0.3 ≤ X ≤ 0.6999 (66%)					
0To1	0.A_1		NEG	NEG	FPH_0To1 L/F/G[7]	NEG
Desc.	$A_0 = 0$, $A_1 \in [1, 9]$, $m = 1$					
1To2	1.A_1		NEG	NEG	FPH_1To2 L/F/G[8]	NEG
Desc.	$A_0 = 1$, $A_1 \in [1, 9]$, $m = 1$					

[6] TooBig/JustRight/TooSmall
[7] LessThan0/From0To1/GreaterThan1
[8] LessThan1/From1To2/GreaterThan2

Items for the DecimAliens game				
Item Type	**Example**	**Games**	**Node & states**	**Affect nodes**

Decimal Repres.	0.705
	7.05
	70.5
	705
		DA	`TypeD`
			W/F/DC/DI[9]
				`DA_DAble`
				`DA_FAble`
				`DA_WAble`
				Indir: `DA_PlaceValue`

Updating Method	If the correct answer is a Decimal and the chosen correct answer is a Decimal - DansC If the correct answer is a Decimal and the chosen wrong answer is a Decimal - DansI If the correct answer is a Decimal and the chosen wrong answer is a Fraction - FansI If the correct answer is a Decimal and the chosen wrong answer is a Word - WansI			

Fraction Repres.	2/1000
	2/100
	2/10
	2
		DA	`TypeF`
			W/FC/FI/D[10]
				`DA_DAble`
				`DA_FAble`
				`DA_WAble`
				Indir: `DA_PlaceValue`

Updating Method	If the correct answer is a Fraction and the chosen correct answer is a Fraction - FansC If the correct answer is a Fraction and the chosen wrong answer is a Fraction - FansI If the correct answer is a Fraction and the chosen wrong answer is a Decimal - DansI If the correct answer is a Fraction and the chosen wrong answer is a Word - WansI			

Word Repres.	5hundredths
	5tenths
	5units
	5tens
		DA	`TypeW`
			WC/WI/F/D[11]
				`DA_DAble`
				`DA_FAble`
				`DA_WAble`
				Indir: `DA_PlaceValue`

Updating Method	If the correct answer is a Word and the chosen correct answer is a Word - WansC If the correct answer is a Word and the chosen wrong answer is a Word - WansI If the correct answer is a Word and the chosen wrong answer is a Decimal - DansI If the correct answer is a Word and the chosen wrong answer is Fraction - FansI			

[9] Wans/Fans/DansC/DansI
[10] Wans/FansC/FansI/Dans
[11] WansC/WansI/Fans/Dans

D List of Hypothesis Nodes

`FineClass`: Fine classification of misconceptions.

`CoarseClass`: Coarse classification of misconceptions.

`NEG`: True/False

`DA_PlaceValue`: True/False

`DA_Wable`: True/False

`DA_Fable`: True/False

`DA_Dable`: True/False

`Disjoint`: True/False

`HN_eff`: True/False

`FPH_PositionAbility`: True/False