Stereotypes, student models and scrutability

Judy Kay
Basser Dept of Computer Science
Madsen F09
University of Sydney
AUSTRALIA 2006
email: judy@cs.usyd.edu.au

Abstract. Stereotypes are widely used in both Intelligent Teaching Systems and in a range of other teaching and advisory software. Yet the notion of stereotype is very loose. This paper gives a working definition of stereotypes for student modelling. The paper shows the role of stereotypes in classic approaches to student modelling via overlay, differential and buggy models.

A scrubtable student model enables learners to scrutinise their models to determine what the system believes about them and how it determined those beliefs. The paper explores the ways that scrubtable stereotypes can provide a foundation for learners to tune their student models and explore the impact of the student model. Linking this to existing work, the paper notes how scrubtable stereotypes might support reflection and metacognition as well as efficient, learner-controlled student modelling.

1 Introduction

Stereotype-based reasoning takes an initial impression of the student and uses this to build a detailed student model based on default assumptions. This paper explores stereotypes because they constitute a powerful mechanism for building student models and because this form of inference seems to be particularly important for student and user modelling.

We see some rudimentary forms of stereotypic reasoning within a large range of customisable software. For example, many systems offer help which can be customised at one of two levels: beginner or advanced. This usually operates very simply as follows. Users are assumed to be at the beginner level unless they alter the profile settings for help. This means that the default is to assume the user is a beginner.

The form of help offered to a beginner is based on a raft of assumptions about the knowledge and needs of the typical beginner. Similarly, the advanced help is based upon assumptions about typical advanced users. Most systems do not explicitly represent these assumptions. Typically, they reside in the head of the author of the help.

This paper explores the role of stereotypic student models that are explicit and available to the student.
2 Stereotypes

The use of stereotypes in user modelling began with GRUNDY [26] [27] [28]. Rich defined stereotypes thus:

A stereotype represents a collection of attributes that often co-occur in people. ... they enable the system to make a large number of plausible inferences on the basis of a substantially smaller number of observations. These inferences must, however, be treated as defaults, which can be overridden by specific observations. [28]:35.

In GRUNDY, the user would give several words of self-description. For example, a user might say they are *athletic*. GRUNDY used this as a trigger for a large number of stereotypic inferences about the user. In the case of the athletic person, GRUNDY might infer they were likely to be motivated by excitement, have personal attributes like strength and perseverance, and are interested in sports. Each of these inferences had a rating indicating its strength. From this collection of inferences about the user, GRUNDY recommended books that matched these motivations and attributes. After making recommendations and allowing the user to respond to them, GRUNDY refined the student model by adjusting the rating on each component of the model.

Stereotypes have been explicitly employed in several teaching systems, for example [1] [2] [11] [23]. And the double stereotype was critical to KNOME's construction of user models for the Unix consultant [7] [8]. KNOME reasoned from the user's actions to a classification of their expertise. So, for example, if the user appeared to make competent uses of sophisticated aspects, they were assumed to be expert. In addition, once a user was classified as an expert, KNOME inferred they knew aspects of Unix an expert is likely to know.

Suppose a stereotype $M$ is part of the student modelling in a system which represents a set of components $\{c_j\}$, each of which represents some aspect of the user. For example, one component might represent whether the student knows about *loops* in the programming language, Python.

The stereotype has a set of trigger conditions, $\{tM_i\}$, where each $tM_i$ is a boolean expression based upon components of the student model. Any $tM_i$ may be a single component $c_j$ of the user model or a function of several components, $f(\{e_k\})$. For example, consider a stereotype intended to capture inferences about an expert C++ programmer’s knowledge of concepts in Python. One trigger condition might be based on a component which models whether the student is an expert C++ programmer.

The primary action of the stereotype is:

$$\text{if } \exists i, \ tM_i = \text{true} \rightarrow \text{active}(M)$$

meaning that when any trigger $tM_i$ becomes true, the stereotype $M$ becomes active.

There is a set of retraction conditions, $\{rM_i\}$. Consider an example of a retraction condition for the C++ programmer stereotype. Suppose, for example,
we determine that the student knows neither Python’s *while-loop* nor the *if*. Since these constructs are essentially the same in both Python and C++, this condition (that the student does not know Python-while and does not know Python-if) is a retraction condition for the stereotype $M$.

A stereotype is deactivated when any of the retraction conditions, $rM_i$, becomes true:

$$
\exists j, \ rM_j = true \rightarrow not\ active(M)
$$

(2)

and finally, the effect of a stereotype activation is that a collection of stereotype inferences $\{sM_k\}$ can be made:

$$
active(M), \rightarrow \{sM_k\}
$$

(3)

Some triggers may be essential:

$$
\exists e, \ (tM_e \in \{tM\}) \text{ and } \not\ (tM_e \in \{rM\})
$$

(4)

meaning that like any trigger, $tM_e$ can activate a stereotype. In addition, if $tM_e$ is known to be *false*, the stereotype is deactivated.

A natural way to think about the stereotype can be based on an agent model. Initially, each stereotype is inactive but waiting for one of its activation conditions to become true. Once it is active, it waits for one of its deactivation conditions to become true.

An important characteristic of stereotypes is that the size of the set of components involved in each trigger function is usually far smaller than that of the inference set.

Rich suggested that another characteristic of stereotypes is that they serve only as default assumptions. These apply only until other, presumably more reliable evidence becomes available. We prefer to generalise this, to allow the possibility of even less reliable sources of evidence. For example, when we ran coaching experiments [18], the student model kept track of cases where the coach had sent advice to a student. We considered this to be a very weak form of evidence for the student knowing aspects coached. It would have been quite reasonable to consider it as weaker evidence than a stereotypic inference.

A student modelling system might operate as follows when it needs to know the value of a particular component $c_j$:

- ask all active stereotypes for information about $c_j$;
- seek other sources of information about $c_j$;
- if there is more than one piece of information about $c_j$, resolve any conflicts about the value of $c_j$ by making assessments of the relative reliability of the information available.

An important characteristic of stereotypic inference is that it is intended to be *statistically* valid. For a population of users who belong to a stereotype $M$,

$$
\forall i, \ sM_i \in \{sM_j\}, \ p(sM_i) > p_M
$$

(5)
where \( p_M \) is some probability value that is accepted as the threshold for including an inference in the stereotype. This value \( p_M \) is an important defining characteristic of a stereotype. It establishes the standards applied by the designer of the stereotype in deciding which inferences to allow.

Of course, the statistical character of the stereotype means that \( p_M \) can give no guarantees for an individual. This means that for an individual, if the stereotype \( M \) is active, some of the inferences in \( \{s_{M_I}\} \) may well be incorrect. In fact, we would expect that, for a typically large stereotype with many inferences, some of those inferences in \( \{s_{M_I}\} \) would probably be incorrect. The whole point of stereotypic inference is that it gives a set of useful default assumptions which are generally useful for a population of users. A good set of stereotypes should enable a system to be more effective for most students, even if it may be quite ineffective for a small proportion of students.

This statistical character of stereotypes should be distinguished from many other sources of uncertainty in knowledge-based reasoning. For example, we might have an inference:

\[
knows(A) \rightarrow knows(B)
\]

meaning that a system can infer from the fact that the student knows \( A \) to conclude that they know \( B \). An instance of such an inference might be:

\[
knows(loops) \rightarrow knows(variables)
\]

meaning that if a student knows the concept of \( loops \) in C++, we infer that they know the concept \( variables \) since it is a prerequisite. Suppose that we are uncertain whether the student knows \( loops \), perhaps assigning a probability \( p_{loops} \) to the truth of the assertion that the student knows \( loops \). In that case, the inference about \( variables \) would also have an associated probability related to \( p_{loops} \).

We can contrast this form of uncertainty from that due to stereotypic inferences (which may also have associated probabilities with each inference). For example, one inference might be

\[
active(M) \rightarrow knows(local\ scope)
\]

which may be the inference that average C++ programmers will understand the notion of \( local\ scope \). We may have written this stereotype after studying the knowledge of many C++ programmers; we may have found that 87% of average C++ programmers understood \( local\ scope \). We might then associated a probability \( 87 \) with this stereotypic inference. This means that we would expect to find 13% of people who are average C++ programmers and for whom this inference does not hold. The complete stereotype \( M \) will have many such inferences.

3 Stereotyped Student Models

The stereotypes described above may seem quite unlike the student modelling in most systems. Indeed, aside from the small number of systems mentioned earlier,
most systems ostensibly seem to operate quite differently. This section shows the use of stereotypes in most student modelling. This will serve as a foundation for the next section’s description of the important role of scorable stereotypes.

An appealing property of the stereotype is that it should enable a system to get started quickly on its customised interaction with the student. That quick start is often based upon a brief initial interaction with the user or, less commonly, a short period observing the user. For example, a system might ask the user just a few questions. Equally, it might set the student an initial task which is used to assess their level. From this small base of information, the system infers the values of a large number of components of the student model.

Consider the case of a system which teaches Python. If it knows nothing about the student, it would logically have a default initial student model for the *typical person* and this might reasonably set all components of the student model to indicate the student knows no Python concepts. This is the implicit stereotype of the typical beginner’s programming book. Equally, it is the implicit stereotype for a classic CAI system.

By contrast, an ITS adapts its teaching to the individual student. So it may begin the interaction with some attempt to construct an initial student model. For example, it might begin by asking the student to indicate their level of knowledge of various programming languages. Suppose the student assesses themself as an expert in C++ but having no knowledge of Python. This can activate a stereotype which assigns the value known for the components which model the student’s knowledge of the many concepts which are essentially the same in C++ and Python. This represents the intuitive reasoning that a person who is expert in C++ can be expected to know its core concepts and, where these are common to Python, that person should have a conceptual level of knowledge for those concepts in Python. There may be a hundred or more such concepts. For example, these include understanding such notions as *loops*, *while loops*, *booleans to control loops* and *nested loops*. So the single question about C++ expertise can have a fanout inference of more than a hundred student model components. If a single question about C++ expertise can be used to infer so much information, a system might quickly begin its customised, highly effective teaching of Python.

A second stereotype can be triggered by the the users claim of no knowledge of Python. This could assign the value unknown for components representing the student’s knowledge of the detailed syntax and idiom of Python.

Yet another stereotype inference could assign the value unknown to those Python concepts which are quite different from anything in C++. It could also set as unknown, those Python concepts which clash with knowledge of C++, because there are similar elements but important differences. An example of this is the *for loop* which is a looping construct in both languages but it operates differently in each. The trigger for this stereotype is the user’s claimed expertise in C++ combined with their claimed ignorance of Python.
3.1 Novices, Intermediates, Experts and others

We now review some major approaches to representing student models: the overlay, differential and buggy models. We identify the stereotypic inference that occurs in all of these.

The commonest form of student model is the overlay which represents the learner’s knowledge as a subset of the total domain knowledge modelled. This may be the expert’s knowledge. Of course, the notion of an expert domain model is stereotyped; in practice, different experts disagree on some aspects of their domain of expertise.

The differential model is a form of overlay model which represents a subset of domain knowledge. This student model deals only with the aspects that the system intends the student to learn. We might call this plausibly ideal student; a stereotype of the sort of student knowledge and skills we might reasonably expect to be achieved after learning with the system. This differs from the overlay on an expert model because it distinguishes those aspects of the expert model the student is expected learn from others. In a sense, it represents aspects of the domain that are within the scope of the teaching system. It captures the system designer’s view of knowledge that will have been acquired by the student who learns all the aspects taught by the system.

In contrast to overlay models, buggy student models represent incorrect beliefs that learners may hold. The classic systems in this group were BUGGY [4] and PROUST [20], both of which developed a body of very interesting work on learner’s misconceptions and errors. This work can be seen as involving construction of a stereotype model of student errors: it represented a number of the mostly commonly observed errors. Essentially, the buggy student model captures the statistically most common misconceptions. It is not expected that any one learner would have all of them. Indeed, each may be quite uncommon: a relatively common misconception might only be held by 30% of beginners. However, the system represents them because there is an underlying assumption that the system may be better able to understand some of the learner’s actions by interpreting them in light of the buggy model. Where a misconception is held by 30% of all beginners, it may be much more common among beginners who are observed to make certain classes of errors.

There is a large literature on differences between novices versus experts, such as [6]. This provides a foundation for constructing stereotypes of beginners and experts in particular domains.

3.2 Building Stereotypic Student Models

Building stereotypes involves defining: the triggers \( t_M \); the retraction conditions \( r_M \); the stereotype inferences \( s_M \); and the threshold probability, \( p_M \), for inferences in the \( M \) population.

**Hand-crafted Stereotypes.** This is a very obvious approach. Nonetheless, it deserves mention because it seems to be so widespread in teaching systems.
Essentially, the designer of the system makes assumptions about the stereotype groups. For example, there may be stereotypes for the beginner and the advanced student. Although this approach may often be ad-hoc, its value and importance should not be underrated. For example, an expert teacher may have built up invaluable stereotypes of typical student knowledge at various stages of their learning. Capturing and encoding this experience in stereotypes could be an important contribution to the body of knowledge of about how to teach effectively.

Another important potential role for handcrafted stereotypes arises in local customisation of systems, for example, an experienced teacher can observe their own students. In addition, that teacher knows the context of the learning activities. So, that teacher is ideally placed to define stereotypes of the individual knowledge, learning goals and common problems for their own students. This is likely to be an important role for stereotypes as ITSs are deployed.

**Empirically-based Stereotypes.** These approaches do not rely on elicitation of an expert teacher’s knowledge of students. Instead, we collect data about students and use this to construct stereotypes. This has considerable appeal where a student works with an online tool such as a spreadsheet. In such cases, it is straightforward to monitor their actions.

For example, we might run empirical studies where users are asked to attempt a task. We then monitor user actions as they attempt the task. If we repeat this experiment over many tasks, we can construct a stereotype which maps from sequences of user actions to the likely task the user was attempting to do. This constitutes a set of stereotypes whose triggers are user actions and each inference set infers both the tasks the user was attempting and the lack of knowledge associated with flawed approaches to tasks. This approach has been applied in Lumiere [16] which can be viewed as a teaching system which gives just-in-time advice, at the time the user needs to learn in order to achieve a task.

More broadly, there is an important role of machine learning in acquiring stereotypes [29] [33] as well as careful study of empirical data to identify stereotypes [32]. There are important potential links between this task and the construction of similar stereotypes for information retrieval and filtering. This goes under various names including community, collaborative, clique-based approaches [24].

**Stereotypes Inference.** Collection of information for triggering stereotypes comes from three main sources:

- directly elicit information from the student;
- observe the user interacting with the system;
- diagnostic tasks.

The first is very simple and we have already given examples of the student being asked to assess their expertise in a programming language.
The other two are closely linked to each other. For example, in the context of a system which teaches about an operating system, it might be feasible to monitor the student's use of that system. Then, as in the Unix Consultant, use of sophisticated commands might be used to infer expertise. The third method is more common in ITSs. It might ask the student to do set tasks. If the student can do difficult tasks, making effective use of sophisticated commands, this can be used to infer expertise.

4 Stereotypes and Scrutability

Scrutability of stereotypes should mean that a student can scrutinise the system to find answers to questions like the following.

– Am I a beginner?
– What are the implications of being a beginner?
– What would be different if I were an expert?
– How can I let the system model me as a beginner, but have it recognise some of the more advanced things I know?

There seems to be the potential for considerable benefit if learners can explore such issues. Some relate to the possibility of encouraging reflection. This has been described by Goodman, Soller and Linton [13]:

Reflective activities encourage students to analyse their performance, contrast their actions to those of others, abstract the actions they used in similar situations and compare their actions to those of novices and experts.

Others have discussed and explored this notion of the variously described open, accessible or transparent student models and systems. See, for example, [3] [9] [10] [11] [12] [15] [22] [25] [30]. They identify benefits of such approaches in terms of:

– potential learning benefits if access to the model can nurture reflection and metacognition;
– the enhanced learner control over the personal information typically held in a student model;
– the possibility of improving the quality of the student model as learners are able to correct errors in it.

We can expect that the particular case of stereotype-based student modelling would be likely to share these potential advantages.

5 Discussion

We now consider the special relevance of scrutability in association with stereotypes for student modelling.
5.1 Corrections to Stereotype Models

The nature of stereotypes makes them especially important as targets for user access and correction. This is because stereotypes are constructed in terms of their accuracy and utility for a population of users. Equally, there is a corresponding expectation that some inferences \( sM_k \) will be incorrect for some users. There are two levels of control associated with stereotypes.

- The whole stereotype: The student can decide that an active stereotype should be deactivated, or vice-versa. So, for example, the student can decide to deactivate the beginner stereotype and possibly choose to activate some other.
- Individual inference level: The student can alter the value any single inference \( sM_k \). For example, the student may be content to have the beginner stereotype active. They might check several of beginner inferences and be happy with these, However, they may see that it makes some incorrect inferences. The student should be able to correct these in their own model.

The first of these could be achieved if we extend the notion of stereotypes as follows: every stereotype has a built-in retraction condition which can be set by the student.

The second can be achieved by regarding the student input as a more reliable source of student modelling information. Then, the set of information about a component \( e_j \) could potentially include the inference from the stereotype and the information volunteered by the student. So long as the student modelling system treats the latter as more reliable, we have a simple mechanism for retaining the active stereotype but allowing the student to fine-tune the details.

5.2 Stereotypes, Teaching and Learning Agendas

Typically, a student model represents just those aspects the system needs. Some parts of the student model drive the adaptation of the teaching. Some may assist the system in its interpretation of the student’s actions. Yet others represents the core learning goals for the system. We now focus on these.

The student model will typically track the learner’s progress; hopefully, the student model will reflect the student’s ongoing progress as they learn each of these. Stereotypes can be useful for initialising these aspects of the student model. For example, a few carefully chosen questions or diagnostic tasks might be used to classify the student as intermediate-level and then to infer the initial model, with some of the teaching goals set as known. This initialises the system’s teaching agenda.

Another important potential role for stereotypes relates to the student’s own learning agenda. In theory this could be modelled separately from the teaching goals. This would mean representing both the student’s knowledge and whether they want to learn each aspect. The default stereotype assumption might set all unlearnt teaching goals as learning goals. Scrutability of and control over this stereotype would enable the student to tune the learning goals.
One important source of problems for learners can occur when there is a mismatch between the teacher’s goals and the learner’s appreciation of the overall and, particularly, the current goals [10]. Scrutability of the student model offers the potential to reduce the effect of such problems. As Self notes, [31] student models capture a precise definition of essential state in a teaching system. This is a foundation for individualisation and for shared understanding between the learner and the system, with the learner being able to better understand what drives the system.

5.3 **Buggy Stereotype as Learning Objects**

If a student modelling system makes use of buggy stereotypes, these encode a potentially useful set of information for learners and teachers. Consider the following scenario. A student is classified as a beginner in the domain of Python programming. Suppose they are trying to write a first Python program and they have problems. A clever ITS might diagnose the difficulty. Equally, if there is a good presentation of stereotypic errors by beginners in this task, the student might read this and work out what their problem is. Yet another possibility is that a human teacher might be better able to help the student, aided by this list of stereotypic errors. Just this use was intended for the IDEBUGGY extension of work on BUGGY.

5.4 **Individual or Stereotype – Is There a Conflict?**

At first glance, one might think that individual and stereotypic student modelling are at odds. In practice, stereotypes can support highly individual student models in two ways. First, a rich collection of stereotypes can ensure that each student will have many active stereotypes at once. The possibility of many combinations of stereotypes leads to a correspondingly large collection of different models, all based purely on stereotypes. Beyond this, if the stereotypes are used as initial default inferences which are refined over time, we can expect each student’s model to become more individualised as more data becomes available to refine it.

6 **Conclusion**

We have defined a stereotype \( M \) as:

- triggers, \( \{ tM_i \} \), which activate a stereotype
- retraction conditions, \( \{ rM_i \} \), some of which may correspond to the negation of essential triggers, and learner control requires a built-in retraction condition which can be set by the student
- stereotypic inferences, \( \{ sM_i \} \)
- threshold probability for inferences, \( p_M \), which captures the minimum probability of each inference for a population of users matching this stereotype.
The action of a stereotype is to make large numbers of inferences when a trigger becomes true. Many student models can be regarded as using stereotypic inferences, although they are often implicitly coded.

Scrubtivity of student models seems to offer potential benefits in terms of improvements in learning and in the accuracy of the student model. Where student models are based on stereotypic inference, there are even stronger arguments for scrubtivity since the inferences are only valid in a statistical sense. The elements listed above indicate the aspects which the student might scrutinise to understand the stereotypic reasoning applied in their own student model.

References