CHAPTER 3

CooTutor System

The CooTutor (Coordinate Tutor) system proposed in this study is an adaptive Web-based learning environment with interactive 3D media for SGT learning. How to design architecture and user interface to consider both the media and method concerns will be the main issue that we concern in this section.

3.1 System architecture

Since CooTutor is developed to fit the Web, the inherent server-client architecture affects the design of the system. The contemporary E-Learning standard, SCORM (Sharable Content Object Reference Model) offers a good reference of generic Web-based Learning Environment (WBLE) architecture [1]. CooTutor's run-time environment is shaped to meet the integration need of interactive 3D media. The proposed architecture is illustrated in Figure 3.1.

As Figure 3.1 shows, at the client side, three main elements form the user interface of the learning area. The *main document* area presents main contents, including textual presentation and mathematical symbols. By embedding scripting codes in the document, we can visualize 3D presentations (e.g. 3D models, animation scripts) in the *3D blackboard* module. In CooTutor, these functionalities are implemented by using Java 3D technology and FastScript3D toolkits [47]. The interactivity between textual documents and 3D contents are achieved by the communication between these two modules.

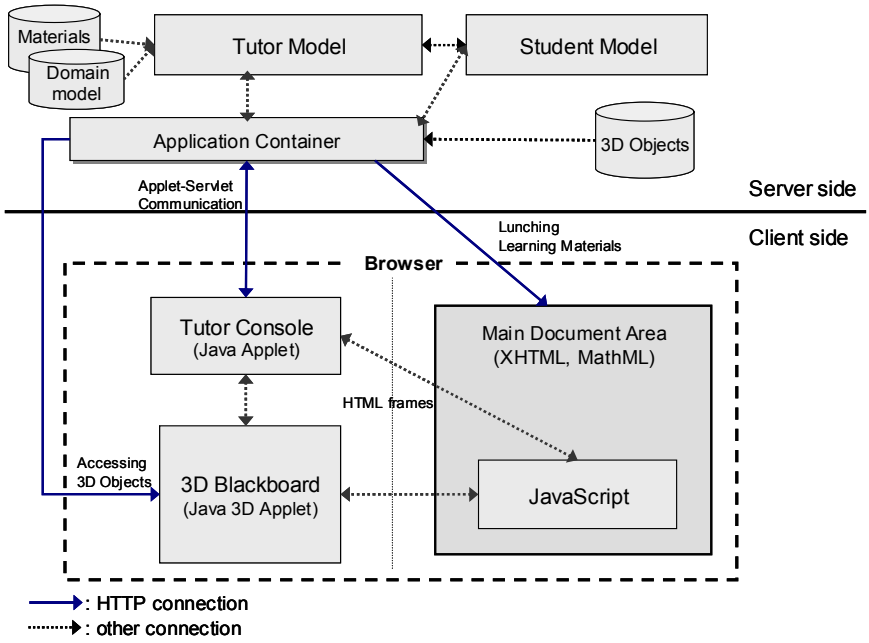


Figure 3. 1: CooTutor system architecture.

All learning resources at the server side are delivered through the HTTP protocol to the client side. In addition, the tutor model and the student model are used to achieve adaptivity, established by the collaborative communication between server and client.

3.2 User interface with Interactive 3D Media

The graphical user interface of the system, as illustrated in Figure 3.2, consists of three parts: the *tutor console*, *main document* area, and the *3D blackboard*.

The tutor console takes the responsibility of client-side management including lunching learning materials, recording learners' behavior (e.g., staying time, button pressed etc.), and further communicating with the server. The tutor console acts like a control center residing at the client side to assure that this media-rich environment works compactly.

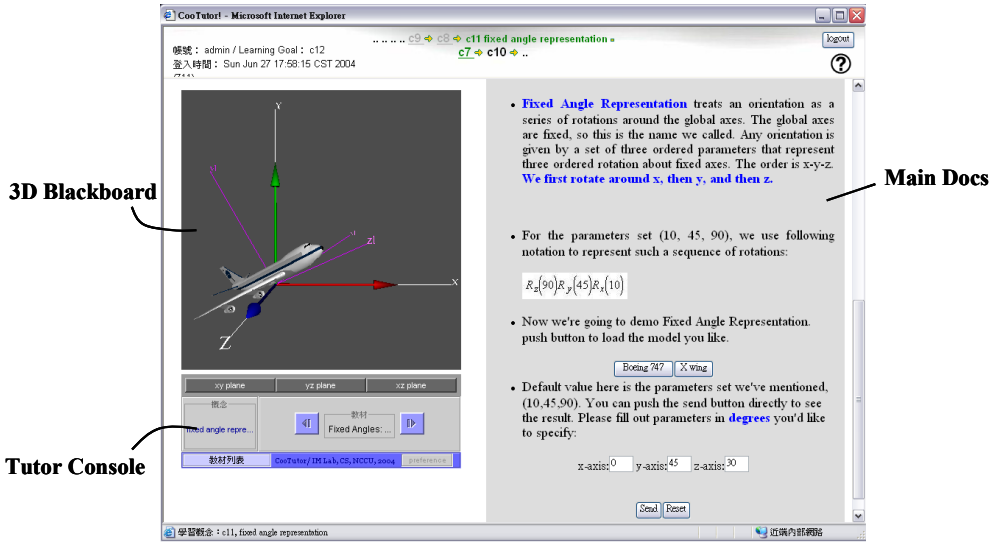


Figure 3. 2: User interface with 3D blackboard.

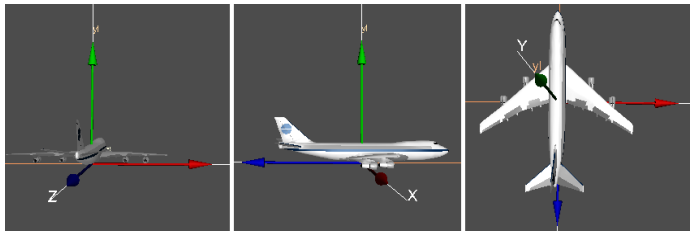


Figure 3. 3: Describing spatial relations with 3D navigation.

The 3D blackboard is the module that provides 3D visualization and interactivity for learners to attain better reasoning. The term, *interactive 3D media*, is used to emphasize its characteristics of interactivity and 3D visualization.

It is worth noting that pure 3D representation might be not enough for learning from the viewpoint of multiple media representations [67]. Though most students today prefer pictorial material with visual effects rather than plain textual representation, we cannot ignore the potential advantages of using textual and mathematical symbols in learning science contexts. In our system, by incorporating suitable interactivity between different media in this system, we can make the integration of multiple representations more meaningful and effective on learning.

Two kinds of interactions are considered. One is the interaction between 3D blackboard and learners. Learners are allowed to navigate the scene with different modes, e.g. zoom, pan and rotate, to obtain a good view for better understanding of the content. The other kind of interaction is between textual documents and the 3D blackboard. Learners read the textual content in the main document area, and directly manipulate objects in 3D scenes by adopting interaction objects (e.g. buttons, input fields) embedded in the Web pages. The links between textual descriptions and 3D instantiations are made according to the needs of understanding particular topics. By the theory of anchored instruction [67], the main document area is used to provide the contexts of the topic. The interaction between the main document and the 3D blackboard allows learners to explore the scene and do experiments freely. This type of user interfaces provides an environment for learner-centered construction.

Figure 3.3 shows how CooTutor presents spatial relations by using the 3D blackboard. Learners are allowed to change the viewpoint by dragging the mouse in the scene or by pressing default buttons (e.g., “change viewpoint to the X-Y plane”) to a better location for better understanding of the spatial configuration.

3.3 Adaptivity in CooTutor

Adaptivity is the common feature of ITSs and AEHs. In the proposed concept of intelligent media, the design of computerized adaptivity directly refers to the “method” concern mentioned in Chapter 1 that attempts to capture instructional methods and strategies originally possessed by human tutors. CooTutor adopts the course sequencing approach to attain adaptivity [12][13]. In brief, the task is to select a set of learning materials for learners according to the student model and learning materials’ features. Figure 3.4 shows the adaptive mechanism of CooTutor. Three main levels exist: *concept sequencing*, *material selection* and *client-side tuning*.

Server-side decision making is divided into two levels. Both processes are performed by the tutor model as Figure 3.1 shows. First, we sequence the concepts in accordance with learners’ knowledge status. The objects to be sequenced at this level are not learning materials directly, but concepts. Therefore, this phase is also called *concept sequencing*. After determining what concept should be delivered, *material selection* proceeds. Appropriate learning

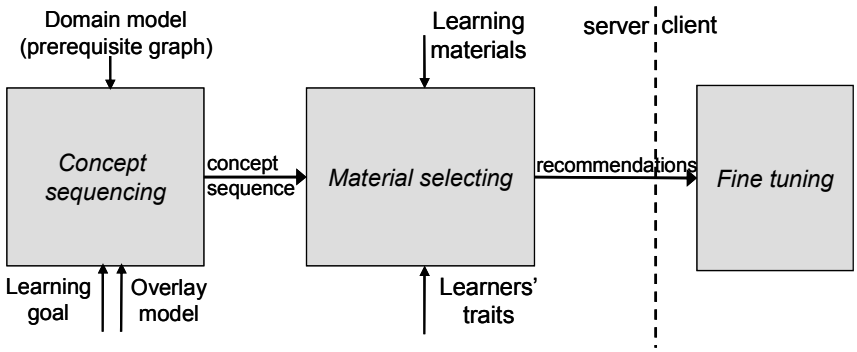


Figure 3. 4: The flow of adaptivity in CooTutor.

materials are subsequently selected to illustrate the concept. This is similar to the two levels of curriculum sequencing (i.e. knowledge sequencing and task sequencing) mentioned by Brusilovsky in [11].

Our approach could achieve adaptivity on both levels. Different aspects of adaptivity are considered in each level. For the level of concept sequencing, we account for learners' prior knowledge. That is, we consider what concepts should be delivered next. For the level of material selecting, we will take learners' traits and preferences into account and select the kind of learning materials and presentation that would benefit particular learners most.

We have described the overall view of the adaptive mechanism. In the following sub-sections, details behind the mechanism will be revealed.

3.3.1 Domain modeling

In typical procedure-based systems e.g. CAI (computer-assisted instruction), the system firmly specify the sequence of learning materials in general. The approach greatly lacks flexibility. In contrast, model-based systems (i.e. ITS. AEH) could attain personalized learning with generalization. That is, different kinds of factors involved in computerized tutoring are grouped and modeled explicitly. A rather concise definition of "model" in intelligent systems was: "A model can be defined as an abstraction of reality, faithful to the reality in ways deemed important. [37]" For intelligent tutoring, the modeling task is largely to consider *how to design a sensible, inspectable and computable (the most important one in the case of intelligent tutoring) knowledge representation* for factors involved, such as the learner, the learn-

ing domain and the pedagogical strategies. The knowledge representation could be of the form of data structure, or could be embedded as latent intentions underlying the process of decision making. Just as the example of procedure-based systems, each time the system pushes their learners to follow a specific learning flow locally based on a single test, it is similarly that user modeling is performed implicitly. However, for intelligent tutoring and adaptive systems, researchers preferred to employ explicit modeling if possible. Implicit modeling are also used partially for reducing the complexity of user modeling, such as the method of stereotyped user modeling. Next section will describe this point further.

Here the first factor to be modeled is the learning domain. We employ the graph structure as the method of knowledge representation in this case. Then we could use algorithms or agent techniques to attain adaptive effects.

The domain to be learned (i.e., the learning domain) in CooTutor is organized as a *prerequisite graph*, a directed acyclic graph (DAG) similar to that proposed in [38]. Vertices in the prerequisite graph represent concepts in the domain. Edges in the graph represent the *conjunctive* relations. For example, assume that we are given a prerequisite graph $G=(V, E)$, where V is a set of vertices and E is a set of edges. Suppose there exists three vertices, $u, v, w \in V$ and we have $(u, w) \in E$, and $(v, w) \in E$. This indicates that both concepts u and v should be known (i.e. learned) before learning concept w. Note that these vertices are concepts, *not* learning materials, as usually used in other researches. When the nodes are learning material, disjunctive relations are usually required. However, such a model with disjunction relations is usually too complex and inflexible because the computational complexity of adaptivity increases rapidly [38]. When the domain knowledge is modeled at a more abstract level, it is sufficient to adopt conjunctive prerequisite graph, as used in CooTutor, to find the feasible concept sequence for learning targeting a specific objective.

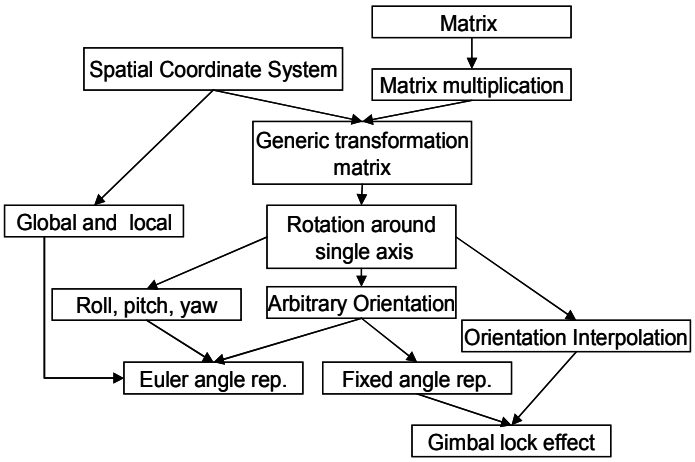


Figure 3. 5: Organizing concepts as a prerequisite graph (partial).

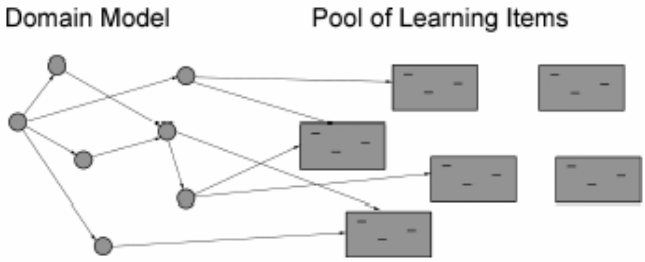


Figure 3. 6: Linkage between the domain model and learning items, adopted from [13].

Figure 3.5 demonstrates part of the domain model in CooTutor. The arcs between nodes specify the conjunctive prerequisite relationship. For instance, for learning the concept "generic transformation matrix", "spatial coordinate system" and "matrix multiplication" are prerequisites required to be learned as the background.

We have emphasized that nodes of the domain model represent concepts, which is the abstraction of underlying learning materials (or items). Hence we have to consider an appropriate linkage between learning materials and concept nodes. Brusilovsky et al. described such a relation in [13], as Figure 3.6 illustrated. Moreover, for the needs of making adaptation to

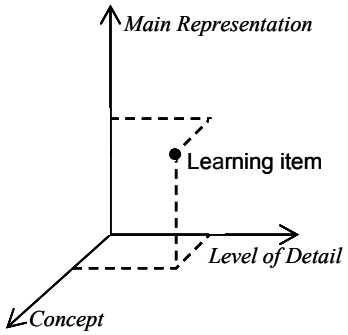


Figure 3. 7: Learning items as points in the attribute space (partial).

learners' traits, each learning material should be indexed by its features (or attributes). Conceptually, we could treat each learning item as a specific point in the *feature space*, as Figure 3.7 shows. By identifying these features of learning materials, we can organize them in a systematic manner. We will describe features of learning materials in section 3.3.4.

3.3.2 Student modeling

Student modeling, as a branch of user modeling, could be a complicated task actually. In some application context, user modeling could even be an intractable problem as indicated by [59]. One may think that student modeling is merely to record learners' data into the database. This is just partially correct. The salient point of student modeling is not about data itself, but how to make use of such data in terms of the view of adaptivity. Therefore, suitable representation, summarization, generalization and inference are needed in this task.

In CooTutor, there are three distinct types of information related to a learner should be noticed. They are:

(1) *learners' prior knowledge*, i.e., what concepts an individual has already known. An individual's prior knowledge changes often during the learning process in general. This information could be determined via tests or quizzes. Specifically, computerized adaptive test is recognized potential on this issue to minimize the time required for testing and maximize the accuracy of testing. It is also viable to detect learners' prior knowledge by observation. This is

the approach that stereotype user modeling usually employ. For example, in an on-line bookshop, for a user who bought a book about advanced engineering mathematics, it could be inferred as that the user has learned and known about calculus in mathematics. However, such approach is obviously not accurate, especially when the grain-size of the learning domain is not that large at the level of curriculum. Besides, in the on-line bookshop case we illustrated, learners' *knowledge* and *interest* is compound. To employ which kind of method to derive the information of knowledge would depend on the purpose, needs and concerns of the system.

(2) *learners' traits* are relatively stable characteristics of an individual and unlikely to change during a learning session. For example, personal interests, learners' capabilities of acquisition, learning styles and spatial ability[2] belong to this type of information. The main solution to derive this type of information is by using psychometric instruments [62], such as a survey of learning styles [61] and the spatial ability test [9]. At present, there is no suitable theory to support the idea of deriving this type of information by observing small amount of user behaviors. However, the subject to be modeled is a population of users not an individual, it seems possible to use methods of data mining to summarize and derive useful messages in terms of learners' traits.

(3) *learners' low-level behavior* is information about how the user interacts with the user interface, such as navigating in 3D scene, pressing buttons, dragging the mouse, and the staying time in each page. This implicit profiling approach gradually receives researchers' notices due to the prevention of unnecessary intervention to users for asking questions explicitly [34]. The main problem would be that such implicit information is not as reliable as the explicit answers/scores directly derived from users. Also, how to properly interpret users' behavior into their intention is another problem because many operations performed by the users might mean nothing actually. Just as what we have mentioned, no suitable theory could interpret such low-level behaviors in a sensible manner. Hence CooTutor does not rely on this type of information at present. But we also recognize that users' interaction patterns with the 3D scene would probably encode amount of meaningful information implicitly, what is worth to be explored in future. We think it is possible to employ such type of information to refine the

[2] Though many studies (including this study) have proposed that spatial ability could be properly enhanced (i.e. could be changed), but the range of enhancement seems quite small in comparison with other typical domains. For the purpose of adaptivity, we treat it as a stable trait.

Table 3. 1: The comparison of three types of learner information.

Types of learner Information	Knowledge	Traits	Low-level Behavior Patterns
Stability	Instable	Stable	N/A[3]
Method of acquisition	Observation, Conventional test, Adaptive test	Psychometric instruments, Spatial ability test, Data Mining	Observation
Example	Known(learned) concepts	Cognitive/Learning styles, Interests, Spatial reasoning skills	Mouse dragging, Time spent on pages
Corresponding adaptive level in CooTutor	Concept sequencing	Material selection	Material selection
Modeling techniques in CooTutor	Overlay model	Used as triggers in stereotyped model	Not yet employed (Theory refinement)

student model. This point will be described in section 3.3.5.

The comparison of these three types of learner information is shown in Table 3.1. Different modeling techniques are used according to the characteristic of the type of information. Here we discuss these techniques employed in CooTutor, including overlay student modeling and stereotyped student modeling.

[3] We do not address the stability of behavior patterns in this study. The meaning of low-level behavior is ambiguous. This is an issue worth to be explored further, but beyond our scope.

- Overlay Student Model

For prior knowledge that is assessed via tests, we represent each learner's *known concepts* as a subset of domain model, the prerequisite graph. This is also known as the *overlay modeling* technique [15][37]. Let $G_{dm}=(V, E)$ as a prerequisite graph to model the domain, where $V=(c_1, c_2, \ldots c_{n-1}, c_n)$ is the ordered set of concepts in this model. We can describe an individual's knowledge about the domain at a particular timestamp t as an ordered set $SM_t = (s_1, s_2, \ldots s_{n-1}, s_n)$, where each element s_i represents the learning state regarded to its corresponding concept in V. That is,

$$s_k \in \{known, intermediate, unknown\}, k=1 \ldots n$$

Note that $|SM_t| = |V|$, thus the learner's knowledge status of the domain is exactly *overlaid* on the domain model. Though there are only three possible learning states of each concept here, describing the state as a numeric value is also allowed by this model.

- Stereotyped Student Model

For learners' stable traits, we simply record it. While the system is going to make decisions, these data are employed as *triggers* for stereotypic inference. This technique is known as stereotyped student modeling.

A *stereotype* is a collection of characteristics that occur together conceptually. For example, if someone works in HsinChu Science Park, this person is probably majored in electrical engineering, 25 to 35 years old, and is a male. In this case, we only know about *where the person works at*, that is so called *trigger* in stereotypic inference. A collection of information called *stereotype* is further inferred. In this example, the stereotype named as *People_in_HSP*: <*major=EE*, *age=25-35*, *sex=male*> is inferred. An on-line shop such as amazon.com could use this stereotype to make personalized recommendations.

It is obvious that stereotypes might make incorrect assertion with respect to the truth. Even

so, people use stereotypes in their daily life to make simplification and categorization of complex scenarios in order to make decisions efficiently [57]. Note that stereotypes are usually used to initialize a default model efficiently. Researchers also employ theory refinement or machine learning techniques to enhance the accuracy of the model. The main advantage of stereotypic inference is its efficiency. Especially when the system could only observe limited facts, this approach is powerful to construct the user model. To address stereotyped student model more formally, as Judy Kay described in [46]:

> "Stereotype-based reasoning takes an *initial impression* of the student and uses this to build a detailed student model based on default assumptions."

The main components of a stereotype M are identified as [46][68]:

(1) a set of triggers, $\{tM_i\}$, which activate a stereotype,
(2) a set of retraction conditions, $\{rM_i\}$, employed to deactivate an active stereotype, and
(3) a set of stereotypic inferences, $\{sM_i\}$.

A formalized mathematical description given by Kay is briefly summarized as follow [46]:

Stereotypic inference $\{sM_i\}$, the main action of the stereotype, is:

$$if \ \ \exists i, tM_i = true \rightarrow active(M). \tag{3.1}$$

What means that if any trigger becomes true, the system would activate the stereotype M. Similarly, if any condition in $\{rM_i\}$ is observed, stereotype M should be deactivated. Once M is activated, consequent stereotypic inference $\{sM_k\}$ can be made, and so on.

The concept of stereotyped modeling is applied to generate adaptation based on learners' traits. As Table 3.1 shows, spatial ability and learning styles are used as triggers in stereotype modeling. Stereotypes are used to select appropriate materials fitted to particular learners. That is,

```
Concept_Seq ( GoalConcept g,
              PrerequisiteGraph G_dm,
              OverlayStudentModel sm )
1   List topo_order ← Topological_Sort (G_dm);
2   List seq ← topo_order – {known concepts in sm};
3     for every node c in seq do
4       if (exist a path from c to g in G_dm) then
5           retain c in seq;
6       else
7           remove c from seq;
8     return seq;
```

Figure 3. 8: The algorithm for concept sequencing, *Concept_Seq*

the mechanism of *material selection*. We will describe it further in section 3.3.4.

3.3.3 Concept sequencing

From the view of educational psychology, the prime principle of learning should be to build the connections between prior knowledge and the topic to be learned (or taught). No matter the type of education used is teacher-centered instruction or learner-centered construction, most educators will agree on this point. Just as Ausubel et al. mentioned, "the most important single factor influencing learning is what the learners already knows."[5]

For realizing adaptive educational systems on the Web, the role of prior knowledge is specifically essential. The process of learning in this type of environment is expected to be self-paced by learners. Learners own the right to determine which hyperlink to be launched, how much time to be spent on viewing content, and when to finish the learning session. Therefore, the system should offer adequate recommendations to prevent unnecessary detour and disorientation in the information space. Some preliminary studies have shown the effectiveness of considering prior knowledge to generate adaptive presentation and adaptive navigation support [24].

As we have mentioned in section 3.3.1, CooTutor defines the prerequisite relations of concepts in the domain model. Learners' prior knowledge is modeled as an overlay model on top of the domain. Thus we can derive the proper learning sequence accordingly. Targeting a spe-

cific goal concept, learners with enough prior knowledge could skip fundamental concepts over, whereas a beginner to this domain can receive fundamental lessons first as necessary complements.

Figure 3.8 presents the algorithm for concept sequencing in CooTutor. Given a goal concept g, the task is to find out a best sequence of concepts according to the domain model G_{dm} and the overlay student model sm, where $G_{dm}=(V, E)$, $g \in$V. The goal g could be specified by learners or inferred by the system. At line 1 of the algorithm, *topological sort* algorithm for graphs is performed. A topological sort of a DAG $G=(V, E)$ generates a linear sequence $S=\{v_1, v_2, \ldots, v_{|v|}\}$ of all vertices in G. For every pair of vertices v_i and v_j in the sequence S, if $(v_i, v_j) \in$ E, then $i<j$ is established [22][55]. Therefore, we could assure that the resulting sequence *seq* generated at line 1 conforms to the constraint of prerequisite relations defined by G_{dm}. At line 2, it removes concepts that the learner has known from the sequence. From line 3 to 7, the algorithm checks that if each concept in *seq* is relevant to learn the given goal. This step could be done by performing depth-first or breadth-first searches for the graph. At the end, we derive the final result, concept sequence *seq*. Once the concept sequence is derived, re-sequencing is needed only when the goal is changed.

3.3.4 Material selection

Learning materials could be authored as typical Web pages, 3D-integrated pages, PDF documents, etc. With the advances of Web hypermedia, there have been amounts of multimedia technologies appearing that could be used in authoring and enriching learning materials. Available technologies nowadays, including Macromedia's Flash, W3C's SVG (Scalable Vector Graphics), and video streaming technologies etc., have been already used in educational purposes widely. It is observed that by using these media technologies, learning materials on the Web could be of various "flavors", or more formally, pedagogical styles. Though it is still rather limiting to realize every instructional method and pedagogical style at the level of learning materials, it is recognized beneficial to have a systematic method to address this concern. However, less study has been undertaken for the needs [62]. The proposed method in CooTutor is the mechanism of *material selection*.

We can treat a learning material as a self-contained content object associated with *features*

Table 3. 2: Features of learning materials.

Features	*Values (numeric between 0 and 1)*
Main representation	2D-based / 3D-based
Abstractness	Concrete / Abstract
Activity type	Lecture / Experiment
Level of details	(unidirectional)

or *metadata*. Learners' traits, including spatial ability and learning styles are transformed into corresponding needs of particular types of materials, which could be formulated as a *query* to retrieve appropriate learning materials. Therefore *material selection* could be abstracted as a task of information retrieval (IR).

However, this task still differs a lot to typical IR, such as Web retrieval or IR for digital library. Note that in most cases of text retrieval, documents are indexed by a feature vector of terms (i.e., keywords) appearing in documents. IR in these cases is thus the task of computing the proximity between the query given by users and the feature vector associated to documents, in the vector space of keywords [6]. The underlying logic is to find out documents containing part/all keywords of the query. Nevertheless, in the case of material selection, it is not intended to retrieve documents at the level of keyword-matching. Material selection aims to select materials matching learners' traits, such as spatial ability and learning styles. That is, this type of IR is to retrieve documents at the level of *pedagogical styles*. Here we describe what features are used to index learning materials, and how to formulate learners' traits as queries properly.

- Features of learning materials

Each learning material (or content objects) is indexed by features shown in Table 3.2. The feature of *main representation* indicates how the object presents contents. For example, 3D content will be presented in the 3D blackboard. The feature of *abstractness* refers to the degrees of abstraction each material adopts to describe the concept. The feature of *activity type* marks up the underlying pedagogical method of each content object. The feature of *level-of-details* means how detailed the object describes a particular concept.

Note that the feature of *main representation*, *abstractness* and *activity type* are bidirectional. For example, a research paper with plenty mathematical descriptions is thought to be more abstract. Then the feature of *abstractness* would be assigned a high value, such as 0.8. Similarly for concrete and practical learning materials, a lower value would be assigned on the same feature. That is, we use only one field to record such bidirectional characteristic of the feature. This is viable since these bidirectional features are deemed to be continuous between the two ends of that dimension.

These features are selected by considering *whether it is a proper discriminant for classifying learning materials*. For example, *level-of-difficulty* is not recognized as a proper discriminant in CooTutor, though this feature has been proposed frequently by other studies. Our reason is that for fundamental concepts in our domain, it is natural to consider the learning materials as "simple" ones, i.e., low level-of-difficulty. Similarly, for advanced topics all learning materials are necessarily to be "difficult". Instead of level-of-difficulty, abstractness is probably a better one to be used with respect to the characteristics of our domain, SGT. We imports previous observation made by [63] as a reference to select these features.

Besides the features described above, "concept" is another feature that has to be specified obviously, as we have shown in Figure 3.6 and Figure 3.7. Note that the use of this feature is somewhat different from others. Recall that the first phase of decision making in CooTutor is the task of *concept sequencing*. The task we are discussing—material selection, is performed after a specific concept is chosen to be delivered. Therefore, the feature of concept is used by the system to *project* materials in the content repository into a sub feature space in terms of the specific concept. So materials irrelevant to learning the specific concept will not be considered in the process of material selection.

- Query formulation from learners' traits

In CooTutor, *learners' traits* including *spatial ability* and *learning styles* are considered as the sources for material selection. Psychometric studies have proposed formal tools to assess human spatial ability [9][53]. We employ the spatial ability test, *Purdue Visualization of Rotation Test (PVRT test)* in CooTutor to assess such competence [9].

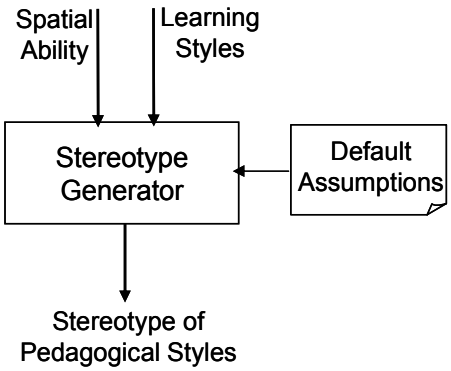


Figure 3. 9: The process of query formulation.
The output stereotype is used as the query.

Besides PVRT for spatial ability, the *Index of Learning Styles Questionnaire* (ILS questionnaire) proposed by Soloman et al. in [61] is used to assess learners' learning styles. This instrument consists of 40 question items. Four dimensions of learning styles can be assessed by the questionnaire. Each dimension of learning style is measured by 10 items evenly. The four dimensions are *visual/verbal learning style*, *sequential/global learning style*, *sensing/intuitive learning style*, and *active/reflective learning style*. Since we intend to use spatial ability as features that also address the visual/verbal concern, items of the questionnaire targeting to measure the visual/verbal dimension of learning style are not used. Similarly, since it seems not viable to address the sequential/global learning style at the level of learning materials, items for this dimension of learning style are not employed either. After all, we assess learners' *spatial ability*, *sensing/intuitive learning style*, and *active/reflective learning style* for material selection.

Learners' traits are used to trigger *stereotypes of pedagogical styles*. These stereotypes are then employed as queries to retrieve learning materials that best fit learners' needs in terms of traits. The process of query formulation is shown schematically in Figure 3.9. By using learners' quantitative scores of spatial ability and learning style as external inputs, with default assumptions, then the stereotype of pedagogical styles is derived as the result. Recalling that in Chapter 2 we have mentioned that there is still no consensus on what kind of instructional methods would be beneficial to tackle learning styles. Therefore, the process of query formu-

lation is conceptually equivalent to using default assumptions to generate *stereotypical decision making*.

The default assumptions used are enumerated here:

A. Spatial ability: according to our analysis in Chapter 2, the higher a learner's spatial ability is, the less degree of visualization she/he will need.

B. Sensing/intuitive learning style: sensing learners would prefer concrete or practical learning materials, while intuitive learners prefer theoretical and abstract materials, such as mathematical descriptions.

C. Active/reflective learning style: active learners would prefer doing experiments, while reflective learners would prefer learning materials in the form of typical lecture.

D. Level-of-details is set as medium (numeric value 0.5) initially.

In other words, the assumptions given above are pedagogical heuristics. We import such heuristics offered by [61] in order to make our decision more sensible and pedagogically valid.

The form of stereotypes is illustrated below, which is represented as a vector consisted of seven elements:

$Q = <is\_2D, is\_3D, is\_concrete, is\_abstract, is\_lecture, is\_experiment, level\_of\_details>$

Each element has a numerical value varies from 0 to 1. Following rules of complement establish as well:

$is\_2D + is\_3D = 1$,
$is\_concrete + is\_abstract = 1$, and
$is\_lecture + is\_experiment = 1$.

The job of the *stereotype generator* shown in Figure 3.9 is to transform learners' traits into the query *Q*. The process of inference is a simple matching. The default assumption *A* shown above is applied to determine the value of elements *is_2D* and *is_3D*; default assumption *B* is applied to *is_concret* and *is_abstract*; default assumption *C* is applied to *is_lecture* and

is_experiment; while finally default assumption *D* is applied to *level_of_details*. Note that the representation of stereotype is as the same form with what we will describe next—the feature vector of learning materials.

- Computing similarity for material selection

We have described the features used to mark-up learning materials and how we formulate the queries. Subsequently, the *(dis)similarity measures* (i.e., distance between objects) used in clustering algorithms in machine learning is employed to select learning materials. A similar approach is used in [68]. But the objective is somewhat different.

In typical IR, it is quite popular to use the cosine measure between vectors as the similarity measure. A detailed comparison of different methods of measuring similarity has been presented in [65]. By considering the characteristics of our task, here we intend to use *extended Jaccard coefficient* to measure the similarity between content objects [33][65]. The advantage of using extended Jaccard coefficient is evident that it can tackle binary values and numerical values both. To compute the measure, each object should be represented as a feature vector. In our case, features of learning materials are transformed to the feature vector which consists of seven elements:

M = <*is_2D*, *is_3D*, *is_concrete*, is*_abstract*, *is_lecture*, *is_experiment*, *level_of_details*>

As mentioned, this feature vector is of the same form as the query. Each element of the vector has a numerical value varies from 0 to 1. It is clear that the first two elements *is_2D* and *is_3D* stemming from the feature of *Main_representation* shown in Table 3.2. For example, for the feature-value pair, *Main_representation* = 0.8, it could be transformed as *is_2D* = 0.2 and *is_3D* = 0.8. Similarly, elements *is_concrete* and *is_abstract* are transformed from the feature of *Abstractness*. While the elements, *is_lecture* and *is_experiment* are from the feature of *Activity_type*. Finally, we directly adopt the numeric values of the feature, *Level-of-details* as the value of the last element. No expansion is needed because this feature is itself an unidirectional one.

Given two vectors of distinct objects *x* and *y*, the extended Jaccard coefficient is computed as [65]:

$$S_{Jaccard}(x,y) = \frac{x^T y}{\|x\|_2^2 + \|y\|_2^2 - x^T y} = \frac{x^T y}{x^T x + y^T y - x^T y} \tag{3.2}$$

In this case, to compute the similarity of the query Q and each learning material's feature vector M_i:

$$S_{Q,M_i} = S_{Jaccard}(Q, M_i) = \frac{Q^T M_i}{Q^T Q + {M_i}^T M_i - Q^T M_i} \tag{3.3}$$

the higher the measure derived by equation (3.3), the more similar it is between this learning material and the query. In other words, this content object will be ordered with higher priority. By repeatedly measuring the similarity measure of the query and all candidate learning materials. A threshold could be set to divide learning materials into two categories, "recommended" and "not recommended". That is:

$$S_{threshold} = \{h \in \mathbb{R} \mid 0 \le h \le 1\} \tag{3.4}$$

Note that such a threshold is set *arbitrarily*, and should be determined by considering characteristics of the set of learning materials, the learning domain and pedagogical strategies/instructional designs intended to be applied. Then, the category C_M of each learning material M_i can be determined by:

$$C_M = \begin{cases} recommended, & if\ S_{Q,M_i} \geq S_{threshold} \\ not\ recommended, & otherwise \end{cases} \quad (3.5)$$

It is believed that besides spatial ability and learning styles, there are still other types of features could be imported into this framework for selecting materials in terms of pedagogical styles and strategies. So the task of material selection, or more precisely, features identification, can be used to reflect educational concerns at the system level. This mechanism also demonstrates a positive support on the benefits of using model-based approach in computer-based educational systems [75].

3.3.5 Client-side tuning

Client-side computing could be quite beneficial in CooTutor. The use of client-side computing could be two-folded. They are (1) to collect browsing behaviors for refining the student model (i.e., theory refinement, as shown in Table 3.1) (2) to realize client-side adaptivity by the use of *local adaptation rules*. Now we describe these two aspects of use respectively.

First, learners' browsing patterns recorded by the tutor console module could help to determine *how learners interact with 3D objects*, *if learners are satisfied with the recommendations*, etc. However, as mentioned previously in section 3.3.2, interpreting low-level behavior patterns and using it sensibly are still difficult. Therefore, CooTutor's system architecture retains the flexibility for such an extension but does not rely on this type of information heavily. When learners finish learning a concept and proceed to the next one, such information is encoded in HTTP requests to refine the original student model. It is suspecting that techniques employed by pattern recognition (PR) could be employed to address this issue. For example [45] has proposed a user behavior model based on the Hidden Markov Model (HMM) approach.

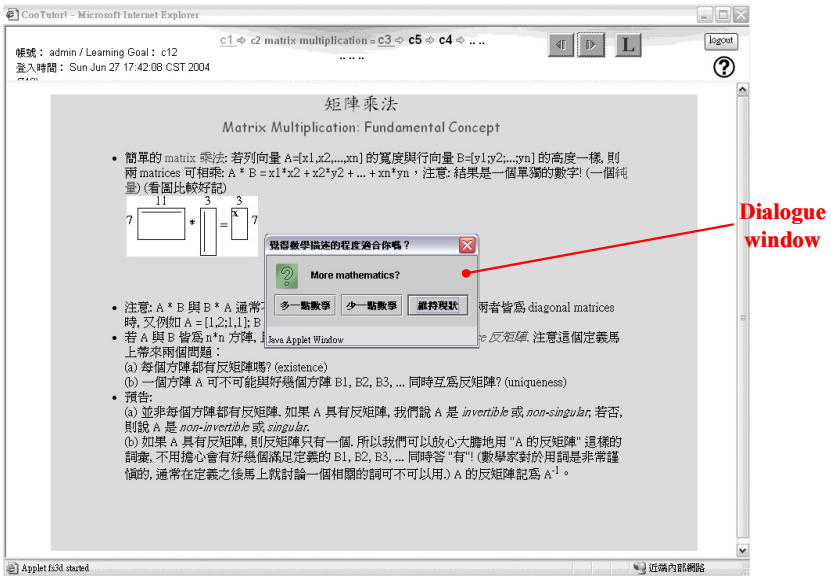


Figure 3. 10: A snapshot demonstrating theory refinement from users' response. The dialogue window is asking the learner "More Mathematics?"

Though in current version of CooTutor, the use of low-level behaviors is not the salient point, a design of simple theory refinement by directly asking the learner has been implemented. It is recognized necessary to incorporate proper mechanism of theory refinement due to the nature that stereotypical decision making is based on default assumptions mentioned in section 3.3.4, and is likely to be inaccurate. As shown in Figure 3.10, after the learner finishing a concept, she/he is enquired if she/he wants to tune the pedagogical styles of the presentation, such as by asking them *more or less graphics (math)?* The underlying action of theory refinement refers to adjusting the stereotype of pedagogical styles (i.e., the query vector, Q) associated to the learner. Some heuristics are used to tune elements of Q accordingly, such as:

- `If` *learner_response=more_math* `then add value to the element` *is_abstract*, and
- `If` *learner_response=more_graphics* `then add value to the element` *is_3D*

Note that such refinement is incremental, and thus may not best reflect learners' needs immediately. Besides, if we intend to use learners' browsing pattern as the basis for theory refine-

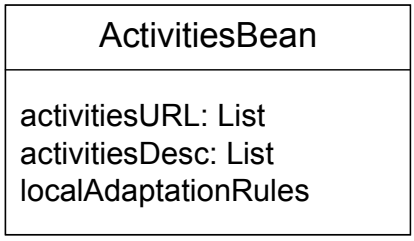
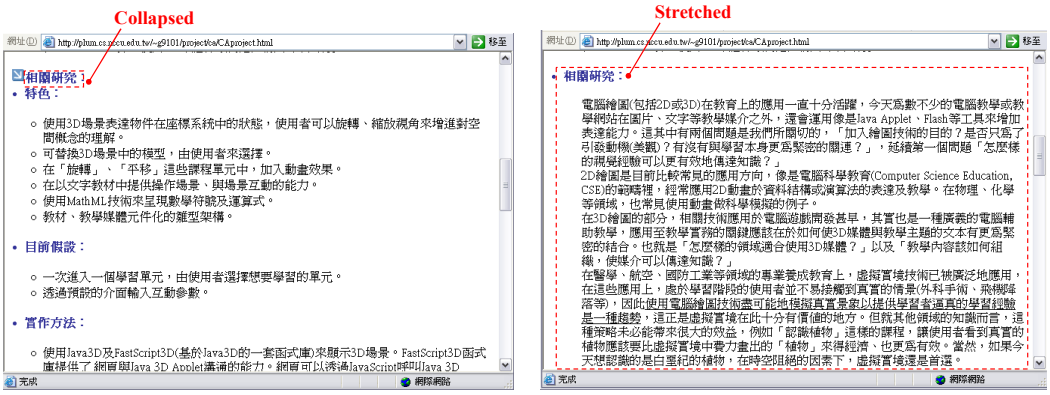


Figure 3. 11: The static class diagram of the class, ActivitiesBean (partial)



(a) (b)

Figure 3. 12: The screenshots illustrating the effect of the adaptive technique—stretch-text, (a) collapsed presentation (b) stretched presentation.

ment, it is obvious that formulating corresponding heuristics, though possible, is not easy.

Second, it is tenable to incorporate *local adaptation rules* to perform sensitive adaptivity at the client side. Note that learning materials selected by CooTutor to illustrate a specific concept are delivered to the client side in the form of *a set of URLs*, which is packaged and serialized as an *activities bean object* at run time for delivery. When the tutor console module located at the client-side receives this object, it will extract and arrange these URLs from the object for the needs of presentation. Figure 3.11 depicts the class, ActivitiesBean and its attributes. In this class, the attribute, activitiesURL is of the type as a List object, which consists of ordered URLs of learning materials that have been arranged by server-side decision making. Evidently, we could attach some local adaptation rules in this class, which specifies the condi-

tions and actions of client-side adaptivity. This idea has not yet been implemented in current version of CooTutor, and will be realized in future.

For example, the author may apply the well-known technique of adaptive presentation, *stretch-text* [10] in their learning materials. Figure 3.12 illustrates how the adaptation effect looks like. The technique aims to make some parts of learning materials initially collapsed (i.e., hidden) in order to prevent learners for being perplexed by overloaded information, especially those not fit to their knowledge or traits. The author may want to realize a pedagogical strategy that once the learner chooses to expand these hidden parts, then the system should always expand hidden parts subsequently as the default setting. This type of concerns could be collected and authored properly in the form of local adaptation rules. Though not yet been implemented in current version of CooTutor, it is recognized realizable and beneficial as an extension of the system. We also note that our activities bean objects could be conceptually interpreted as Learning Objects (LO) appearing in the SCORM specification [1]. Especially, it is possible to make CooTutor as a SCORM-compatible platform in the future. Then we can employ and transform the Simple Sequencing specification used by SCORM as the local adaptation rules. At the long run, it is expectable to realize the integration of LO and AH into an adaptive and reusable learning platform under the framework we proposed [75].