Chapter 6

Adaptive Courseware Generation

Science is to see what everyone else has seen and think what no one else has thought.

Albert Szent-Gyögyi.

This chapter presents the customized courseware and testware generation as incorporated in the proposed ITS. A brief review of related work in the area of dynamic courseware generation is presented. The proposed scheme for generating course material based on ANFIS is described next. Bloom’s Taxonomy is reviewed briefly and the test formulation and generation is described in the light of this taxonomy. Finally, the chapter concludes with summary and discussion.

6.1 Introduction

An important characteristic of a Web-based tutoring system in order to succeed to its educational potential is the opportunities it offers for personalized learning. It is well known that the appropriate match of the learners to the learning experience significantly affects their achievement [Castells & Szekely, 1999], as the order and the manner in which topics are treated can produce very different learning experiences [Weber & Specht, 1999]. However, present-day intelligent tutoring systems have been criticized for the lack of adaptivity [Capuano et al., 2003]. These criticisms are:

i. The common e-learning systems usually do not allow the customization of the courses on the student profile but simply propose standard courses.

ii. They usually limit themselves to deliver learning material to the student, without trying to interact with him through a model of his mind.
iii. They usually exploit possible test results only for checking the students' acquired knowledge level but do not use such a knowledge also for changing the quality and quantity of the delivered learning material itself.

An Intelligent Tutoring System should be typically strongly adaptive, working in a well-structured information space; gathering data about the user's movements and using this information to dynamically modify the presentation and functionality of the system in clearly defined ways. It is again to emphasize that adaptivity is not a technology, but a goal. Adaptivity is a common functional goal of intelligent systems. In summary, for a system to be called an adaptive system it must have the following characteristics [Eklund & Brusilovsky, 1999]:

a. It should be based on hypertext (or hypermedia).

b. It should have an explicit user-model, which records some features of the individual user.

c. It should have a domain model, which is a set of relationships between knowledge elements in the information space.

d. It should be capable of modifying some visible or functional part of the system on the basis of information contained in the user-model.

Although these characteristics are outlined in the context of an adaptive hypermedia system, they are equally applicable to ITSs that attempt to advocate the adaptivity. As explained in previous chapters we have incorporated:

• An explicit user-model that captures some features of the individual learner.

• A three-layer domain model to establish relationship among knowledge elements in the information space, and
• An XML binding for knowledge representation satisfying the hypertext condition.

Adaptivity may be at the content level or at the link level. Content-level adaptivity means dynamic generation of content based on a user model. Link level adaptivity on the other hand assumes a static content, and the appearance or prominence of the links connecting elements of this hyperspace is altered.

Towards this direction, we investigate the use of methods from computational intelligence to support adaptation in a tutoring system and focus on the domain model and the lesson generation process that influence the educational effectiveness of personalized learning. An adaptive network based fuzzy inference system for facilitating a lesson-oriented way of 'teaching' is proposed. The interactions between the fuzzy inference system and the learner assessment procedure in the lesson generation process is described. Experiments and performance results are presented to evaluate the proposed approach. It is shown that the proposed approach fully supports the customization of the course content based on the student profile. The learning material is not only delivered to the student, but the system also interacts with him through a model of his mind. Further, test results are used for checking the students' acquired knowledge level as well as for changing the quality and quantity of the delivered learning material itself.

6.2 Related Work

Much of the work on adaptivity of course content has been reported for adaptive hypermedia systems in comparison to intelligent tutoring systems. These strategies for creating adaptive software have yielded considerable advancements in the adaptive hypermedia context over the last few years. Representative systems

Chapter 6: Adaptive Courseware Generation
include DCG [Vassileva, 1995, 1997], ELM-ART [Weber & Specht, 1999], InterBook [Brusilovsky et al., 1998], TANGOW [Carro et al., 1999] and PERSEUS [PERSEUS, 2003].

ELM-ART and DCG use an explicit representation of domain concepts, interrelated in a prerequisite graph. DCG includes a planner that guides the student along a path to reach goal concepts starting from already-known concepts. ELM-ART uses a sophisticated system to estimate the knowledge acquired by the user in relation to a concept map of the course, according to which the system dynamically proposes the student a path to follow at each moment. Both in ELM-ART and DCG the structure of courses is fixed.

TANGOW generates the course structure at runtime. It models student activity in the form of a hierarchy of tasks that represent didactic units that the student can perform.

InterBook is a tool for delivering adaptive textbooks on the World Wide Web. It uses adaptive annotation technology, a form of adaptive navigation support, to augment hyperlinks with a comment which informs users about the current state of the nodes behind the annotated links, and does this on an individual basis by adapting links through a user-model, an individual record of a student's progression through the courseware.

In PERSEUS, course construction is achieved by defining a content model, by means of an intuitive user interface that allows the course designer to build general objects and class ontologies as basic tools for the design of the educational system.

Some systems, like KA2 [Murray, 1996], and EonITS [Staab, 2000], are focused on ontology engineering.
KA2 has been conceived for semantic knowledge retrieval from the web, building on knowledge created in the knowledge-acquisition community. To structure knowledge, ontology has been built in an international collaboration of researchers. The ontology constitutes the basis to annotate WWW documents of the knowledge acquisition community in order to enable intelligent access to these documents and to infer implicit knowledge from explicitly stated facts and rules from the ontology.

The EonITS uses ontology to define the types of topics and topic links allowed in a semantic net representation of the tutor’s knowledge called the “Topic Network”. Here, the ontology is not specific to the domain, but appropriate for a class of domains. The ontology specifies a number of other things, such as topic properties (e.g. difficulty, importance), and allowed values in the student model.

6.3 Adaptive Courseware Generation

A main issue in the development of an educational system, capable to support pedagogical decisions, is the domain knowledge to include multiple curricular viewpoints on the same knowledge [Wenger, 1987]. To this end, a three-layer curriculum representation has been suggested with each layer providing a different type of pedagogical information [Lesgold et al.,1987]. This distributed approach to subject matter representation emphasizes the notion of lesson rather than that of model as a reservoir of domain knowledge and forms the basis of three-layer architecture [Wenger, 1987]. The proposed model is designed in such a manner that adaptivity of lesson to the learners’ knowledge level is simplified and it is based on domain independent metrics.

Information about the subject, unit and topic comes from the tutor module. The learners’ performance in a particular lesson is inferred from the test results. A
major design decision is to decide the frequency of tests. The principle of total quality management applied to tutoring suggests that the quality test should be at the lesson level as it is the smallest learning object presented to the tutee. It should be however noted that the lesson object is aggregated from the assets and as such lesson is an abstract entity. If the performance is found to be ‘excellent’ in a lesson, the tutor module increments its lesson pointer or the topic pointer to the next lesson or topic and provides information to the course generator to aggregate another lesson. Thus the tests are kept at the lesson level with adaptivity at the content level. The physical layer of the subject domain is already organized according to the ontology of the subject.

Internally the course generator is composed of the following modules:

i. Interface module: interacts with the tutor module and the student model to get the input.

ii. Search module: that searches the required assets for aggregation.

iii. Aggregation module: that retrieves the physical assets and aggregates them.

iv. Transformation module: that applies XSL transformation to the lesson.

v. Presentation module: that transfers the lesson object to the browser at the learners’ site.

6.4 The Lesson Aggregation Module

Since adaptivity should be provided at the granularity level of the lesson, the most intelligent behavior is expected from the aggregation module. While the other four modules can be implemented algorithmically, the aggregation module needs to emulate the behavior of a human tutor. We demonstrate an adaptive network based fuzzy inference system (ANFIS) to select the asset objects that should be
aggregated. The aggregation module maintains a log book for every learner to keep record of the assets that were used to aggregate the previous lessons. The information about the current performance of the learner is fed to the fuzzy inference system to decide the next set of assets to aggregate a lesson. It is to be noted that for a particular lesson of a particular topic there can be several asset objects, which demands an intelligent behavior. Each asset is associated with certain attributes that can be utilised as aggregation guidelines. These attributes are resource-id, concept, example, explanation and the most important of all- the resource ‘class’. The class attribute is assigned to each resource object by the human expert taking the guidelines from instructional design practice [Ertmer & Newby, 1993] as described in the chapter on domain modeling.

One implementation could be simply to use students’ performance as input to the aggregation module and select the same class of concept, example and explanation for lesson generation. However, we adapted a different technique. It could be a point for criticism that the performance of the learner is aggregated from his test results and a cumulative performance index is used for subsequent decisions not considering the understanding level, memorizing ability and misconception individually. This is a valid point that could be justified at the content aggregation level. To give an appropriate weightage to the level of understanding, memory retention and misconception, the inputs to the lesson aggregation module are kept same as those to student modeling component. The results from tests are input to the ANFIS that is trained to select particular class of objects to be aggregated into the required lesson. Twenty three classes of learning objects are decided. The rationale for these classes is the possible weaknesses that could be identified during the learning experience of a learner. These weaknesses
could be low memorization, low understanding level, high misconception and so on. As described in tutor module design, each of the lessons is associated with a particular state of the fuzzy finite state machine. Further, the FFSM transits from state to state due to the change in student’s performance level finally terminating on ‘excellent’ performance. Thus only first three states are considered for deciding the class of the content. Twenty three classes are associated as follows to each state: seven classes to state 1; nine classes to state 2 and seven classes to state 3. Table 6.1 shows the rules that are used to train the ANFIS to select a class. These classes encompass all the possible combination of the content that need to be presented to the learner according to his performance level.

Table 6.1. ANFIS Rules for Lesson Generation

<table>
<thead>
<tr>
<th>Memory</th>
<th>Understanding</th>
<th>Conception</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>01</td>
</tr>
<tr>
<td>Low</td>
<td>Low</td>
<td>Medium</td>
<td>02</td>
</tr>
<tr>
<td>Low</td>
<td>Low</td>
<td>High</td>
<td>03</td>
</tr>
<tr>
<td>Low</td>
<td>Medium</td>
<td>Low</td>
<td>04</td>
</tr>
<tr>
<td>Low</td>
<td>High</td>
<td>Low</td>
<td>05</td>
</tr>
<tr>
<td>Medium</td>
<td>Low</td>
<td>Low</td>
<td>06</td>
</tr>
<tr>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>07</td>
</tr>
<tr>
<td>Low</td>
<td>Medium</td>
<td>Medium</td>
<td>08</td>
</tr>
<tr>
<td>Low</td>
<td>Medium</td>
<td>High</td>
<td>09</td>
</tr>
<tr>
<td>Low</td>
<td>High</td>
<td>Medium</td>
<td>10</td>
</tr>
<tr>
<td>Medium</td>
<td>Low</td>
<td>Medium</td>
<td>11</td>
</tr>
<tr>
<td>Medium</td>
<td>Low</td>
<td>High</td>
<td>12</td>
</tr>
<tr>
<td>Medium</td>
<td>Medium</td>
<td>Low</td>
<td>13</td>
</tr>
<tr>
<td>Medium</td>
<td>High</td>
<td>Low</td>
<td>14</td>
</tr>
<tr>
<td>High</td>
<td>Low</td>
<td>Medium</td>
<td>15</td>
</tr>
<tr>
<td>High</td>
<td>Medium</td>
<td>Low</td>
<td>16</td>
</tr>
<tr>
<td>Low</td>
<td>High</td>
<td>High</td>
<td>17</td>
</tr>
<tr>
<td>Medium</td>
<td>Medium</td>
<td>Medium</td>
<td>19</td>
</tr>
<tr>
<td>Medium</td>
<td>Medium</td>
<td>High</td>
<td>20</td>
</tr>
<tr>
<td>Medium</td>
<td>High</td>
<td>Medium</td>
<td>21</td>
</tr>
<tr>
<td>High</td>
<td>Low</td>
<td>High</td>
<td>22</td>
</tr>
<tr>
<td>High</td>
<td>Medium</td>
<td>Medium</td>
<td>23</td>
</tr>
<tr>
<td>High</td>
<td>High</td>
<td>Low</td>
<td>25</td>
</tr>
</tbody>
</table>
It is to be noted that the ANFIS is going to indicate the class of the content to be used in aggregation. The learning material is designed and organized through XML document type definition in such a manner that the material can be searched using its class with the unit, topic and the lesson number. Further, the aggregation module always aggregates the concept, example, explanation, numericals, simulations and case study in that order. Thus, using the class type taken from the ANFIS, the aggregation module simply retrieves the content pointers and submits them to the transformation module for subsequent processing.

6.4.1 ANFIS-based Lesson Aggregation Model

As already described, adaptive network based fuzzy inference system utilizes a backpropagation algorithm or a hybrid algorithm to tune its parameters from the input-output data pairs. In case of lesson generation process, the rules are taken from the human experts. The criterion for rule framing is based on the learners’ knowledge level. Three parameters used for student modeling are again utilized here. The rules are given in the Table 6.1. Each rule identifies a particular class such that a customized lesson can be generated to emphasize that portion of the lesson where the learners’ performance appears to be unsatisfactory. For example, class ‘1’ is selected on the basis of the test results where learner shows ‘low’ memory, ‘low’ level of understanding and ‘low’ level of conception (i.e. high level of misconception). The material corresponding to class ‘1’ is written in such a way that it further elaborates the same concept- may be with little bit review of prerequisite concept, a more elaborative example and simpler explanation. This class concept provides domain independent but learner-centred approach for lesson adaptation. It is, however, to be noted that the proposed
approach requires much more efforts on the part of instructional designer, as the contents need to be arranged according to their classes.

The proposed ANFIS model is implemented using Matlab Fuzzy Logic Toolbox. 150 input-output data pairs—mapping the test results with the classes—from expert teachers were collected. The data file is given in Appendix A. The input-output data pairs were fed to the ANFIS to tune fuzzy set membership functions. The parameters for fuzzy inference system are shown below.

<table>
<thead>
<tr>
<th>DCG using ANFIS</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANFIS Parameters and Methods:</td>
</tr>
<tr>
<td>Model Type:</td>
</tr>
<tr>
<td>Number of input fuzzy sets:</td>
</tr>
<tr>
<td>Number of output fuzzy sets:</td>
</tr>
<tr>
<td>AND method:</td>
</tr>
<tr>
<td>OR method:</td>
</tr>
<tr>
<td>Defuzzification method:</td>
</tr>
</tbody>
</table>

| Membership function type: | Pi. |
| Parameter ‘Memory’: | a | b | c | d |
| For ‘low’ | -0.3365 | -0.1385 | 0.2086 | 0.3086 |
| For ‘medium’ | 0.3072 | 0.4157 | -0.5498 | 0.7551 |
| For ‘high’ | 0.7030 | 0.7821 | 1.1491 | 1.3470 |

| Membership function type: | Pi. |
| Parameter ‘Understanding’: | a | b | c | d |
| For ‘low’ | -0.3356 | -0.1396 | 0.3053 | 0.3038 |
| For ‘medium’ | -0.7228 | 1.8570 | 0.7380 | 0.8394 |
| For ‘high’ | 0.6688 | 0.7100 | 1.1370 | 1.3330 |

| Membership function type: | Pi. |
| Parameter ‘Conception’: | a | b | c | d |
| For ‘low’ | -0.3365 | -0.1385 | 0.3001 | 0.3096 |
| For ‘medium’ | 0.1491 | 0.7614 | 0.6474 | 0.7490 |
| For ‘high’ | 0.7166 | 0.8466 | 1.1491 | 1.3470 |

The details are indicated in the Screenshots 6.1 to 6.7.
A Fuzzy-Neural Intelligent Tutoring System For The World Wide Web

Screenshot 6.1. Fuzzy Inference system for Lesson-classification.

Screenshot 6.2. ANFIS structure for Lesson-classification

Chapter 6: Adaptive Courseware Generation
A Fuzzy-Neural Intelligent Tutoring System For The World Wide Web

Screenshot 6.3. ANFIS training data pattern and Testing

Screenshot 6.4. PI Membership function and tuned parameters for 'Memory'

Chapter 6: Adaptive Courseware Generation
Screenshot 6.5. PI Membership function and tuned parameters for ‘Understanding’

Screenshot 6.6. PI Membership function and tuned parameters for ‘Conception’
Screenshot 6.7. ANFIS Rulebase showing the rule:

\[
\text{if Memory is Medium \& Understanding is Medium \& Conception is Medium}
\]

\[
\text{then lesson Class is 14.}
\]

As pointed out by Jang [Jang et al., 1997] that the designer of an ANFIS should first analyse his actual data rather than jumping for a fancy model, the data-pairs were carefully analysed first. Same input data used in student modeling are taken and the corresponding lesson class as the output is engineered in consultation with the human experts. Several inference systems -following a trial and error approach- were generated and tested with different membership functions available in the fuzzy logic toolbox; parameter tuning being done through training with backpropagation algorithm, hybrid algorithm or the combination thereof. The 'PI' membership function trained through backpropagation with 5000 epochs is found to give an average testing error of 0.083702 as shown in the Screenshot 6.3.
Thus this structure is finalised. This structure is tested with several individual input data using the ruleviewer and the classification was found to be very satisfactory. The Screenshot 6.7. shows the verification of one of the rules with an individual input.

6.5 Adaptive Testware Generation

As it is well known, the purpose of any teaching-learning system should be to achieve certain objectives. Hence, an evaluation method must be devised that can be used to judge to what extent the desired objectives are met. This evaluation method may be applied at the course level, subject level, unit level, topic level or at the lesson level. In our work we decided to assess the learner at the lesson level. The rationale for this decision, as already mentioned, is the principle of total quality management which suggests that rather than applying the quality control to the finished product at the last stage of production, the quality checks should be applied at the component levels thereby rectifying the defects right at that level. This principle as applied to education and training suggests an assessment method at the lowest possible level of knowledge acquisition. While this may be practically difficult in classroom-like teaching setup, computer-based tutoring appears to be the most appropriate framework for lesson-level assessment. Before describing the proposed adaptive testware generation scheme a brief review of Blooms' taxonomy of learning domains is presented. This taxonomy is used as the basis for course structure design, course content design as well as for formulating the tests.

6.5.1 Review of Bloom's Taxonomy

Bloom's Taxonomy [Bloom et al., 1964] underpins the classical 'Knowledge, Attitude, Skills' structure of learning method and evaluation. It provides an excellent structure for planning, designing, assessing and evaluating
training and learning effectiveness. The model also serves as a sort of checklist, by which it can be ensured that training is planned to deliver all the necessary development for students, trainees or learners, and a template by which the validity and coverage of any existing training (be it a course, a curriculum, or an entire training and development programme for a large organisation) can be assessed.

Bloom's Taxonomy model is in three parts, or 'overlapping domains':

i. Cognitive domain (intellectual capability, i.e., knowledge, or 'think')

ii. Affective domain (feelings, emotions and behaviour, i.e., attitude, or 'feel')

iii. Psychomotor domain (manual and physical skills, i.e., skills, or 'do')

In each of the three domains, Bloom's Taxonomy is based on the premise that the categories are ordered in degree of difficulty. An important premise of Bloom's Taxonomy is that each category must be mastered before progressing to the next. As such the categories within each domain are levels of learning development, and these levels increase in difficulty.

The simple matrix structure enables a checklist or template to be constructed for the design of learning programs, training courses, lesson plans, etc. Effective learning should arguably cover all the levels of each of the domains, where relevant to the situation and the learner. Since the cognitive domain is relevant to our design, salient features of only this domain are briefly described in the Table 6.2.
Table 6.2. Major Levels and Categories in Cognitive Domain.

<table>
<thead>
<tr>
<th>Level</th>
<th>Category</th>
<th>Behavior descriptions</th>
<th>Examples of activity to be trained, or demonstration and evidence to be measured</th>
<th>'key words' (verbs which describe the activity to be trained or measured at each level)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Knowledge</td>
<td>Recall or recognize information.</td>
<td>Multiple-choice test, recount facts or statistics, recall a process, rules, definitions; quote law or procedure.</td>
<td>Arrange, define, describe, label, list, memorize, recognize, relate, and reproduce select, state.</td>
</tr>
<tr>
<td>2</td>
<td>Comprehension</td>
<td>Understand meaning, restate data in one's own words, interpret, extrapolate, and translate.</td>
<td>Explain or interpret meaning from a given scenario or statement, suggest treatment, reaction or solution to given problem, create examples or metaphors.</td>
<td>Explain, reiterate, reward, critique, classify, summarize, illustrate, translate, review, report, discuss, re-write, estimate, interpret, theorize, paraphrase, reference, example.</td>
</tr>
<tr>
<td>3</td>
<td>Application</td>
<td>Use or apply knowledge, put theory into practice, use knowledge in response to real circumstances.</td>
<td>Put a theory into practical effect, demonstrate, solve a problem, and manage an activity.</td>
<td>Use, apply, discover, manage, execute, solve, produce, implement, construct, change, prepare, conduct, perform, react, respond.</td>
</tr>
</tbody>
</table>
Table 6.2. Major Levels and Categories in Cognitive Domain (Continued.).

<table>
<thead>
<tr>
<th>4</th>
<th><strong>Analysis</strong></th>
<th>Interpret elements, organizational principles, structure, construction, internal relationships; quality, reliability of individual components.</th>
<th>Identify constituent parts and functions of a process, concept, deconstruct a methodology or process, making qualitative assessment of elements, relationships, values and effects; measure requirements or needs.</th>
<th>Analyze, break down, catalogue, compare, quantify, measure, test, examine, experiment, relate, graph, diagram, plot, extrapolate, value, and divide.</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td><strong>Synthesis</strong> (create/build)</td>
<td>Develop new unique structures, systems, models, approaches, ideas; creative thinking, operations.</td>
<td>Develop plans or procedures, design solutions, integrate methods, resources, ideas, parts; create teams or new approaches, write protocols or contingencies.</td>
<td>Develop, plan, build, create, design, organize, revise, formulate, propose, establish, assemble, integrate, re-arrange, modify.</td>
</tr>
<tr>
<td>6</td>
<td><strong>Evaluation</strong></td>
<td>Assess effectiveness of whole concepts, in relation to values, outputs, efficacy, viability; critical thinking, strategic comparison and review; judgment relating to external criteria.</td>
<td>Review strategic options or plans in terms of efficacy, practicability; assess sustainability; analyze in relation to alternatives; produce a financial justification for a proposition or venture, calculate the effects of a plan or strategy; perform a detailed and cost risk analysis with recommendations and justifications.</td>
<td>Review, justify, assess, present a case for, defend, report on, investigate, direct, appraise, argue, project manage.</td>
</tr>
</tbody>
</table>

Chapter 6: Adaptive Courseware Generation
Bloom's Taxonomy is a wonderful reference model for all involved in teaching, training, learning, coaching - in the design, delivery and evaluation of these development methods. At its basic level, the Taxonomy provides a simple quick and easy checklist to start to plan any type of training and evaluation system. It helps to open up possibilities for all aspects of the subject or need concerned and suggests a variety of the methods available for delivery of teaching and learning. As with any checklist, it also helps to reduce the risks of overlooking some vital aspects of the development required. The more detailed elements within each domain provide additional reference points for learning design and evaluation, whether for a single lesson, session or activity, or training need, or for an entire course, program or syllabus, across a large group of trainees or students, or a whole organization.

6.5.2 Adaptive Test-ware Generation

As pointed out by one of the experts [Chapman, 1995], training or learning design and evaluation need not cover all aspects of the taxonomy; the designer is free to choose those aspects that are appropriate. We have based our courseware as well as testware design on cognitive domain. Memorizing capability, concept understanding level and misconception are the three characteristics fixed for testing the knowledge level of the learner. The nature of tests is mainly multiple choice questions (MCQs) supported by fill-in-the-blanks, true-or-false and match-the-pairs type questions. MCQs have been identified as powerful tool to evaluate the knowledge level of a learner based on Blooms' taxonomy [MCQ, 2004]. While fill in the blanks and multiple choice type questions may be used for deciding the memorizing capability, true or false type questions for evaluating the understanding level, match the pair type questions may be employed for testing...
A Fuzzy-Neural Intelligent Tutoring System For The World Wide Web

misconception. We pass on this responsibility to the content developer as to which type of question should be used for testing a particular characteristic. However, to facilitate ease of evaluation we provide the ‘class’ attribute associated with each question. This attribute helps indicate the kind of student characteristics likely to be evaluated through this question.

The testware generation is an integral part of the adaptive courseware generator. The learning material at the physical layer is arranged in such a manner that the questions along with the correct answers accompany each of the learning objects. This provides not only simplicity of design but also ease of authoring. The instructional designer can develop questions right at the time of content development. As described in the previous section, the contents are arranged into classes; the classes provide the emphasis to strengthen the memory, the understanding, the true conception or the combination thereof. Thus, within the same content can be kept the relevant questions with the answers.

The test-ware generator is composed of three specific modules:

i. Accumulation module: It accumulates the links for the questions related to the currently aggregated lesson and keeps them in temporary storage.

ii. Test module: It retrieves the actual questions and answers using the links and submits them to the transformation module. Transformation module subsequently passes it to the presentation module.

iii. Evaluation module: It accumulates the learners’ responses, compares them with the correct answers, allocates marks to correct responses and transfers the scores to the student model for subsequent processing.
6.5.3 MCQs Based on Bloom's Taxonomy

Indicated below is a sample of MCQs from the subject domain of "Operating Systems". These MCQs are organised according to Blooms’ Taxonomy in cognitive domain. As is evident, framing an MCQ for synthesis category at level five is quite difficult and has not been attempted here.

Level 1: Knowledge

At this level, one simply requires the recall of acquired knowledge.

Example:

Which one of the following is the most critical component to be managed by any operating system?

a. Main Memory
b. Hard Disk
c. Processor
d. Display device

Note that the responses are internally consistent - they are all the names of essential components managed by any operating system. The answer is ‘c’.

Level 2. Comprehension

At this level, knowledge of facts, theories, procedures etc. is assumed, and the understanding of this knowledge needs to be tested.

Example: Which one of the following describes the characteristic of a good solution to 'critical section' problem?

a. Critical section of a code is executed by more than one processes concurrently.
b. Only one process should execute in the critical section at any given instant of time.
c. Non-sharable resources requested by multiple processes may cause critical section problem.

d. Deadlock must be avoided to eliminate the critical section problem.

In this question, the knowledge of the three characteristics of good solution to critical section problem must be recalled (KNOWLEDGE), and the learner is tested for an understanding (COMPREHENSION) of the meaning of each term, in this case, "mutual exclusion" in ‘b’.

Level 3: Application

In order to classify a question into this group, the author should consider if prior knowledge of the background to the question is assumed to be both known and understood, and whether one is merely expected to apply this knowledge and understanding. Calculations based on known formulae are good examples.

Example: In a multiprogramming time-sharing environment, each process is allocated 1 unit quantum of time slice. There are in all five processes in the system currently, all having same priority, same time of arrival and same execution time as 5 quantum. The average turnaround time for any process will be:

a. 25 units.
b. 23 units.
c. 20 units.
d. 18 units.

Here, the formula Turnaround-Time = Waiting-Time + Execution-Time must be known (recall of knowledge) and the meaning of the various terms in the
formula understood (comprehension) in order to answer this question. The correct answer is 'b'.

Level 4: Analysis

Analysis refers to the ability to break down material into its component parts so that its organizational structure may be understood. Examples of learning objectives at this level are: recognize unstated assumptions, recognize logical fallacies in reasoning, distinguish between facts and inferences, evaluate the relevancy of data, and analyze the organizational structure of a work.

Example: In an operating system, the designer implemented an algorithm to check if the system remains in the safe state or it enters an unsafe state after the allocation of the requested resources. If the system is found to remain in safe state then only the actual allocation is made otherwise the request for the resources is deferred.

This strategy of deadlock handling is known as:

b. Deadlock Avoidance.
c. Deadlock Detection.
d. Deadlock Recovery.

In the above example, the student is expected to know and understand the four methods of handling deadlock situation in an operating system and to apply this knowledge to an actual example of implementation. The ability to analyse the situation in terms of each of the four methods is what is being tested. The correct answer, by the way, is 'b'.
Level 0: Evaluation

At this level, learner is asked to pass judgment on, for example, the logical consistency of written material, the validity of experimental procedures or interpretation of data.

Example: A student was asked the following question: "Briefly list and explain the various characteristics of a good solution to the critical section problem".

As an answer, this student wrote the following:

"The critical section of the code is that portion of a program that may be executed by several processes, in a multiprogramming timesharing environment. However, due to concurrent nature of execution, race around condition may occur. To remedy this situation, a solution to critical section problem is required. This solution should ensure MUTUAL EXCLUSION, BOUNDED WAITING and PROGRESS. Further, the solution should make no assumption regarding the speed of the processes. The solution should satisfy mutual exclusion condition means it should not allow more than one process to enter critical section at any time. Bounded waiting means a request to enter in critical section must be granted in finite time. Progress means only those processes that are not executing in their remainder section of the code will take part in decision making of as to which process will enter next into the critical section. This requirement ensures that each process gets a fair chance to enter critical section."

How would you judge this student’s answer?

a. EXCELLENT (all correct in the right order with clear and correct explanations)
b. GOOD (all stages correct in the right order, but the explanations are not clear).

c. MEDIOCRE (one or two stages are missing OR the stages are in the wrong order, OR the explanations are not clear OR the explanations are irrelevant)

d. UNACCEPTABLE (more than two stages are missing AND the order is incorrect AND the explanations are not clear AND/OR they are irrelevant).

In the above question, one is expected to make value judgment on the content of the given text (KNOWLEDGE of the subject is required), the meaning of the terminology used (COMPREHENSION of the subject matter), and its structure (ANALYSIS) of the answer for the right order of events. The correct answer here is 'a'.

6.6 Summary & Discussion

This chapter presented the adaptive courseware and testware generation process. The typical criticism regarding contemporary intelligent tutoring system is that they lack intelligence and adaptivity. We suggested a computational intelligence approach to adaptive course generation and shown that ANFIS can mimic the behavior of a human tutor in content selection according to the knowledge level of the learner- a desirable characteristic of one-to-one tutoring. This decision has been arrived at after the review of related literature to dynamic course generation. It has been identified from the survey of the related work that content 'granularity' plays a vital role in both supporting and providing the adaptivity. By applying the principle of separation of concerns, we separated the adaptivity of the lesson content from the physical content. The suggested design procedure wherein each of the learning object belongs to a particular class provides
this adaptivity. As described earlier, the adaptivity should be based on the learners’ knowledge level. A lesson level adaptivity guided by the learners’ knowledge level at each stage of lesson generation has been proposed and its implementation through computational intelligence techniques has been shown. The ANFIS directed lesson generation process along with class-oriented content structure provides real adaptivity.

Following are the salient features and advantages of the proposed approach:

i. An adaptive rule based approach provides the advantage of rule flexibility. While we have shown a rule base designed on the basis of expertise gathered from several teachers, an individual teacher’s expertise can also be formalised to reflect a personalized judgment.

ii. The proposed model is domain independent. It does not depend upon a particular subject domain as it follows the principle of separation of concerns.

iii. The instructional designer has full control on the content organization independent of the rulebase. The ‘content class’ concept gives flexibility to the content designer to emphasize the significance of the material.

iv. The testware design is motivated from Bloom’s taxonomy—a time-tested theoretical foundation. This provides a true means of evaluation to gauge the learner’s cognitive level.

We believe that simply making the student model or tutor model or course generation model ‘intelligent’ is not sufficient to provide real adaptivity. It is also necessary to structure the content ‘intelligently’. This belief can be paralleled with the principles of algorithm design and data structures drawn from the field of computer science. It is widely accepted that the main objective of algorithm design and structuring of data is to combat the ‘time and space’ problem, here the term
'time' refers to the processor time and 'space' refers to the main memory in a computer system. It is an established principle in computer science that the combination of an efficient algorithm with proper data structure constitutes an efficient computer program. Similarly, an efficient courseware generator (algorithm) with properly structured domain content (data structure) should constitute an intelligent courseware generator.

The technical feasibility of the model is obvious as already shown in previous chapters on student modeling and tutor modeling based on ANFIS. The model supports subject-domain independance because of the 'class' concept. Following are certain limitations envisaged:

i. It needs expertise on the part of domain experts to identify correct rules for content classification.

ii. The number of classes suggested may not be adequate for all the subject domains.