Individualized Exercises for Self-Assessment of Programming Knowledge: An Evaluation of QuizPACK

Peter Brusilovsky and Sergey Sosnovsky

University of Pittsburgh

Individualized exercises are a promising feature in promoting modern e-learning. The focus of this article is on the QuizPACK system, which is able to generate parameterized exercises for the C language and automatically evaluate the correctness of student answers. We introduce QuizPACK and present the results of its comprehensive classroom evaluation during four consecutive semesters. Our studies demonstrate that when QuizPACK is used for out-of-class self-assessment, it is an exceptional learning tool. The students' work with QuizPACK significantly improved their knowledge of semantics and positively affected higher-level knowledge and skills. The students themselves praised the system highly as a learning tool. We also demonstrated that the use of the system in self-assessment mode can be significantly increased by basing later classroom paper-and-pencil quizzes on QuizPACK questions, motivating students to practice them more.

Categories and Subject Descriptors: K.3.1 [**Computers and Education**]: Computer Uses in Education - *Distance learning;* K.3.2 [**Computers and Education**]: Computer and Information Science Education - *Self-assessment*
General Terms: Human Factors, Performance, Languages
Additional Key Words and Phrases: E-learning, individualized exercises, parameterized questions, assessment, code execution, introductory programming, classroom study

1. INTRODUCTION

Web-based quizzes became the primary tool for assessment and self-assessment of student knowledge in the context of web-based education [Brusilovsky and Miller 1999; 2001]. All major web learning management systems (LMS) support authoring and delivery of online quizzes made from static questions. LMS vendors made a significant effort in extending the range of question types by making question-authoring easier for teachers, providing support for managing quizzes, and supplying pools of questions. Yet even with these modern and powerful tools, authoring web-based questions and quizzes remains a difficult and time-consuming activity. As a result, the number of questions in a typical web-based course remains relatively small. Within an assessment, the lack of questions results in cycling through the same or nearly the same set of questions for each student in a class (and often for different classes and/or different semesters). This arrangement invites cheating, even in a controlled classroom context; and for take-home and web-based assessment, cheating becomes a major problem, often preventing teachers

This research was supported by the National Science Foundation.
P. Brusilovsky, Department of Information Science and Telecommunications, School of Information Sciences, The University of Pittsburgh, PA; 15260; email: peterb@pitt.edu. S. Sosnovsky, Department of Information Science and Tele-communications, School of Information Sciences, The University of Pittsburgh, PA 15260; email: sas15@pitt.edu

from relying on the results. In self-assessment where cheating is not a problem, the number of available questions is typically insufficient for students to assess the level of their knowledge or guide them in further study.

A known remedy for this problem is the use of *parameterized questions and exercises.* A parameterized question is essentially a template for the question created by an author. At presentation time, the template is instantiated with randomly generated parameters. A single question's template is able to produce a large or even unlimited number of different questions. Thus, during assessment, a reasonably small number of question templates can be used to produce individualized assessments for large classes. Moreover, the same templates can be used in different versions of the same course, different semesters, and even different courses. In self-assessment, the same question can be used again and again with different parameters, allowing every student to achieve understanding and mastery. Cheat-proof exercise templates become reusable, nondevaluating assets, which can be accumulated over courses and years in reusable assessment libraries/pools.

Individualized exercises were explored previously, but recently came back into focus as one of the promising research areas in web-enhanced education. A number of pioneering systems such as CAPA [Kashy et al. 1997]; WebAssign [Titus et al. 1998]; EEAP282 [Merat and Chung 1997]; or Mallard [Graham et al. 1997] explored the use of individualized exercises in different contexts. CAPA, the most influential of these systems, was evaluated in a number of careful studies [Kashy et al. 2001; 1997], providing clear evidence that individualized exercises can significantly reduce cheating and improve student understanding and exam performance.

The main challenge to the use of parameterized questions and exercises as an educational technology is the evaluation of student answers. A traditional static question can be evaluated by simply comparing the student answer to a set of answers provided by the author. However, a parameterized question requires a runtime "domain expert" that can find the correct answer for each instantiation of the question. As a result, almost all *known* applications of parameterized questions and exercises were developed for physics and other math-related subjects, where a correct answer to a parameterized question can be calculated by a formula that includes one or more question parameters.

This article reports our attempts to implement parameterized questions for assessment and self-assessment of student *programming knowledge*. Assessing programming knowledge is quite different from knowledge assessment in math-related subjects, so the use of individualized questions has not yet been systematically explored there. We describe a specific approach for the automatic generation and evaluation of parameterized questions for programming courses and the system QuizPACK, which implements this approach for the programming languages C and C++. We also report on our experience with the system and present the results of formal classroom evaluations of QuizPACK during four consecutive semesters with more than 150 students. In brief, our data shows that parameterized questions used in a self-assessment mode are an exceptionally powerful learning tool that can significantly increase the students' knowledge of programming, and is highly appreciated by them. Simultaneously, we investigated how the value of this technology can be increased by creating a context that motivates students to use it.

2. QUIZPACK: THE APPROACH AND THE SYSTEM

While analyzing several large pools of questions created for classroom courses and based on languages like Lisp, C, and Java, we found that a large portion of these questions are

code-execution questions, where the user is provided with some fragment of a program and is asked to predict the value of a particular variable or a string if it were printed at the end of execution of this fragment. These kinds of questions are very important in that they enable the teacher to check the student's understanding of the semantics of programming language structures. Learning the semantics of a programming language is an important part of programming knowledge in itself, and is also a prerequisite to acquiring higher-level programming skills.

In the pools that we analyzed, code-execution questions are authored in the standard, multiple-choice style by specifying the text of the question and a set of possible answers. The student answers are evaluated by comparing them to the correct answers provided by the author. However, it is known that code-execution questions allow a different approach to evaluating the correctness of student answers. Using a language interpreter or compiler, the system can run the given fragment of code and calculate the correct answer without the need for a teacher or author to provide it. Then the calculated answer can be compared with the student answer. In order to save some sizeable amount of question-development time and to avoid authoring errors, we used this approach in several earlier systems such as ITEM/IP [Brusilovsky 1992]; ILEARN [Brusilovsky 1994]; and ELM-ART [Weber and Brusilovsky 2001]. When developing our system, QuizPACK, (Quizzes for Parameterized Assessment of C Knowledge), we attempted to combine our earlier work on automatic evaluation of code-execution questions with the idea of parameterizing the questions. Our goal was to create a system that would allow teachers to create parameterized code-execution questions and quizzes in a streamlined way, rather than creating with the current authoring tools the same questions and quizzes over and over again in a static form. QuizPACK functions in a very straightforward way. A teacher provides the core content of a question: a parameterized fragment of code to be executed and an expression (usually a variable) within that code. The student answers with the ending value of that expression. The system does the rest: it randomly generates the question parameter, creates a presentation of the parameterized question in a web-based quiz, receives the student's input, runs the parameterized code in order to generate the answer, compares this result to the student’s answer, gives a score to the student, and records the results.

The key element here is the ability of the system to execute the parameterized code fragment in realtime while the student is taking the quiz. There are at least two ways to do this. One way is to use a specially designed interpreter or code evaluator that can execute the code of any exercise at runtime. Another way is to use a standard language compiler. Using a simple code transformation, the exercise code can be automatically converted into a function that takes the parameter value and returns the value to be checked. This function can be compiled into a CGI program that will compare the student’s answer to this value at runtime. In this way, each exercise receives a dedicated checker, automatically constructed and compiled when authored.

The first method can provide additional educational value by showing the visual execution of code or by enhancing the exercise with explanations. It was in ITEM/IP [Brusilovsky 1992]; ELM-ART [Weber and Brusilovsky 2001], and a number of systems developed by Kumar [Dancik and Kumar 2003; Fernandes and Kumar 2002; Shah and Kumar 2002] to explain to students how the correct answer to the exercise is produced. The negative side here is the need to develop such an interpreter. While it is relatively easy for systems dealing with small languages (such as ITEM/IP for Turingal) or with language subsets, it becomes a challenge for a system that intends to support an unrestricted professional language such as C or C++. For this reason as well as for better

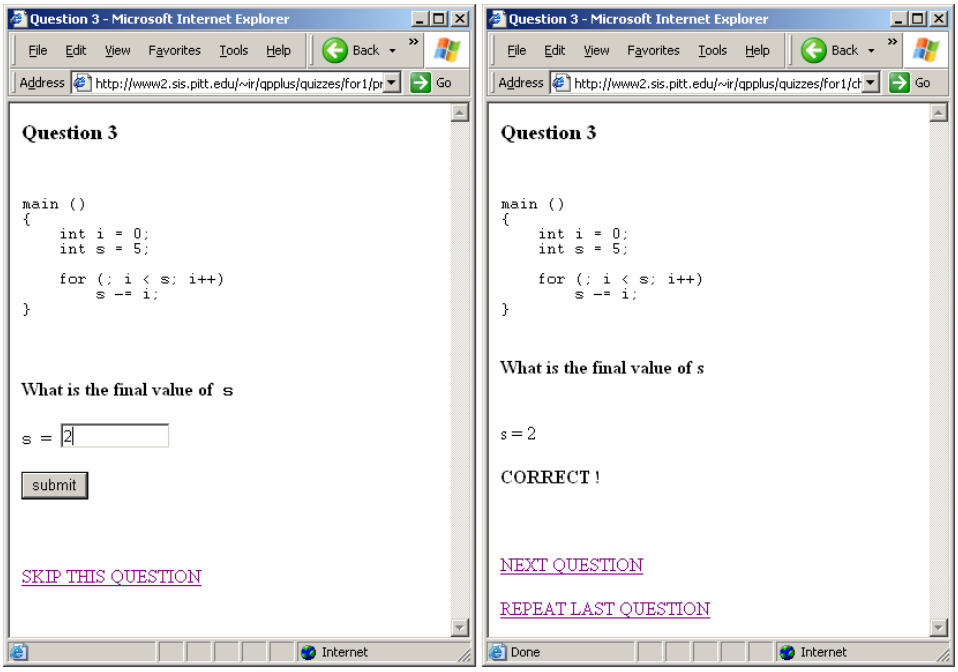

Fig. 1.Student interface for QuizPACK.

efficiency (which is critical for a web-based system that is simultaneously accessed by multiple users), QuizPACK uses the second method.

The current version of QuizPACK can support a complete programming-related course based on C or C++. It supports simple questions based on a few lines of code, as well as very advanced questions that include multiple functions and header files. Figure 1 demonstrates QuizPACK's student interface A parameterized question is presented to the student in a browser as though it were a normal static fill-in question (Figure 1, left). One or more constants that the student can see in the body of the question are actually instantiated parameters, which are randomly generated for students taking the quiz as well as for a single student attempting the question several times. The student fills in the answer and hits the submit button. In response, the system generates an evaluation screen for the student (Figure 1, right), which lets the student rethink the question and the answer. In particular, the student may want to attempt the same question again by using the "repeat last question" link. The same question can be attempted several times with different presentations, until it is fully understood (and, as our study shows, many students actively use this opportunity).

In the context of this journal, which focuses on resources to enhance information and computer science education, it is important to stress that QuizPACK is not only an innovative tool, but also a reliable practical system. It is available for both online use and free download (at http://www2.sis.pitt.edu/~taler/QuizPACK.html). We welcome the readers to try it. Both online and downloadable versions are provided with a large number of questions, covering a complete introductory programming with C course. The downloadable version also includes a toolset for authoring new questions, which makes it possible to turn a code fragment into a QuizPACK question in just a few minutes. Further

details about QuizPACK authoring support and implementation are outside the scope of this article; however, details can be found in Pathak and Brusiolvsky [2002].

3. EVALUATION OF QUIZPACK

3.1 QuizPACK Use at the University of Pittsburgh

QuizPACK has been used to support programming courses to undergraduate information science students at the University of Pittsburgh since 2001. During these years we tried this system in several contexts and ran a number of classroom studies. The first version was developed in 2001 and used for two semesters in a data structures course based on the C language. The system and its first limited evaluation were reported in Pathak and Brusilovsky [2002]. Also in 2002 the School of Information Sciences introduced a new undergraduate introduction to programming class that became the main framework for QuizPACK use and evaluation.

QuizPACK was originally developed for both assessment and self-assessment of programming knowledge. However, we soon discovered that the nature of the system prohibits its application as a reliable out-of-class automatic assessment. In an uncontrolled environment, resourceful students could simply compile the code of the question or run it in a debugger to find the answer. This way of solving a problem has a certain educational value, but it prevents the system from being used as a reliable take-home or web-based assessment. Obviously, QuizPACK could be used for an assessment in a controlled environment such as a classroom quiz or an exam, but this requires a large computer laboratory (40-50 computers for our typical classes), which our school does not have.

Since QuizPACK was designed to work on hand-held computers equipped with web browsers, it can also be used for assessment in a wireless classroom where all students have wireless laptops or hand-held computers. We actually tried this, distributing wireless HP Jornada™ hand-helds to students before each class, but gave it up after two attempts because of technical problems. In a follow-up questionnaire, almost all the students said they preferred traditional paper-based classroom quizzes to Jornada-based ones. It is interesting that only about half the students cited technical problems, while the others gave different arguments in favor of paper-based quizzes. Among the nontechnical reasons, students stated that they preferred seeing all of the questions at once, preferred to work on them in random order, and liked the ability to return to previously answered questions.

Due to the problems above, we decided to systematically explore the use of QuizPACK in a self-assessment context. Since spring 2002, a range of QuizPACK individualized, self-assessment quizzes has been available to all students of an introductory programming class. The quizzes are provided through the KnowledgeTree portal [Brusilovsky 2004] along with other learning content such as lecture notes, program examples, and educational simulations. For each lecture in the course, the students are given access to one or two quizzes (made up of five parameterized questions each) that are focused on the programming constructs in that lecture. All work with QuizPACK is logged: each question attempted by a student leaves a record in a log file, complete with timestamp, student login name, and evaluation result. In addition, we administer pre- and post-questionnaires each semester, to collect student demographic data (including their previous programming experience) and feedback about the system. To motivate students to fill in a relatively long post-questionnaire, we offer three extra credit points for filling it in. However, since we are interested in an informed feedback, only students who attempted a specified number of questions are eligible to fill the

questionnaire.[1] The log and questionnaire data used regularly to evaluate the effectiveness of the system and the students' attitude towards it. This article summarizes the results of an evaluation of QuizPACK as a self-assessment tool in an introductory programming context that was used for four consecutive semesters in 2002 and 2003. Since the context in which QuizPACK was used differed slightly between 2002 and 2003, we present the data separately for the two semesters of each year. In 2002, QuizPACK was used solely for delivering nonmandatory self-assessment quizzes. Though our evaluation results (see Section 3.2) demonstrated that QuizPACK strongly benefited students, we discovered that it was greatly underused. Less than a third of students used QuizPACK relatively actively, by attempting more than 30 questions. Given the positive value of working with QuizPACK, we decided to provide some additional motivation for students to do so by changing the format of classroom quizzes.

The role of classroom quizzes in motivating student learning is well known to teachers. In each of the four semesters in 2002 and 2003 we administered 8 to 10 classroom paper-and-pencil quizzes. During each quiz the students were allowed 10 minutes to answer 5 questions related to the topics of the two most recent lectures. In 2002 we used rather standard multiple-choice quizzes unrelated to QuizPACK. In 2003, we switched to fill-in-the-blank paper quizzes produced with QuizPACK, using the following rule: For each of the quizzes, we selected 5 randomly instantiated QuizPACK questions out of the 15-20 questions for self-assessment on the material of the two most recent lectures. The students were informed about this rule. Thus, the students who worked with QuizPACK after each lecture had a chance to practice each of the forthcoming classroom questions, though with different parameters. We hoped that this arrangement would motivate the students to use QuizPACK more broadly. Indeed, the percentage of students actively using QuizPACK increased more then 2.5 times: from 27% in 2002 to 70% during 2003. In addition, as the data presented below shows, this arrangement significantly changed the profile of student use of QuizPACK and increased the objective and subjective values of QuizPACK as a learning tool.

3.2 Objective Evaluation of QuizPACK

The goal of the objective summative evaluation of QuizPACK is to measure the *objective value* of the system as a learning tool. To determine the objective value we tried to find the relationships between the students' work with the system recorded in the log file and their course performance. The student course performance had three major measures: (1) the total score on weekly in-class quizzes; (2) the final exam grade; and (3) the final course grade. Note that these parameters are quite different. In-class quizzes assessed students' knowledge of the syntax and semantics of the C language. Due to QuizPACK, this knowledge was directly influenced by students' work with the system. In contrast, the final exam in our course mostly measured student programming skills; that is, their ability to understand, modify, and write programs. However, work with QuizPACK can influence these skills indirectly, since they are based on an understanding of semantics. The final course grade (a combination of classroom quizzes, homework programming assignments, and two exams) measures the full range of programming knowledge and skills.

[1] This arrangement did provide motivation to fill in the questionnaire - the majority of students who worked with the system enough to qualify did so. However, as the data below shows, it didn't provide them with enough additional motivation to use the system itself.

Work with QuizPACK could be characterized by a number of measures. In our studies we used two relatively independent ones: activity and success. *Activity* was computed as the total number of QuizPACK questions attempted by the student. *Success*, which is the percentage of questions answered correctly, was calculated as the total number of correctly answered questions divided by the number of attempted questions. Note that each parameterized question could be attempted several times. *Each* of these attempts (correct or incorrect) is counted in the *activity* and *success* measures. Most typically, students worked with the same question until the very first correct attempt; however, some of them kept working with the question even after the first success, which usually resulted in several registered successful attempts.

3.2.1 *Using Parameterized Self-Assessment Quizzes Without Additional Motivation.* During the two semesters of 2002, QuizPACK was available for 81 (39+42) students on the undergraduate level. Only 49 students tried QuizPACK, attempting an average of 38 questions. The number of students using QuizPACK "actively" (that is, attempting at least 30 questions) was much smaller, only 22 (see Table II for more data).

The profile of system use in 2002 was quite irregular (Figure 2). About 40% of the students merely tried the system on one or more occasions, attempting fewer than 35 questions (bar labeled "0" in the graph). Another 40% used it more regularly, attempting between 35 and 71 questions (bar labeled "35" in the graph). The remaining 20% used the system quite seriously, yet there were large individual differences in the amount of work they did with the system; the largest number of questions attempted by a single student over a semester was 319.

Given the irregular profile of system use in 2002, it was hard to expect statistically significant connections between student work with QuizPACK and their class performance. However, we were able to find some relationships which gave insights into the pedagogical value of QuizPACK. To find the relationship between working with QuizPACK and class performance, we conducted regression analysis, which is a standard way to show cause and effect. As "explained" or dependent variables, we used the stu-

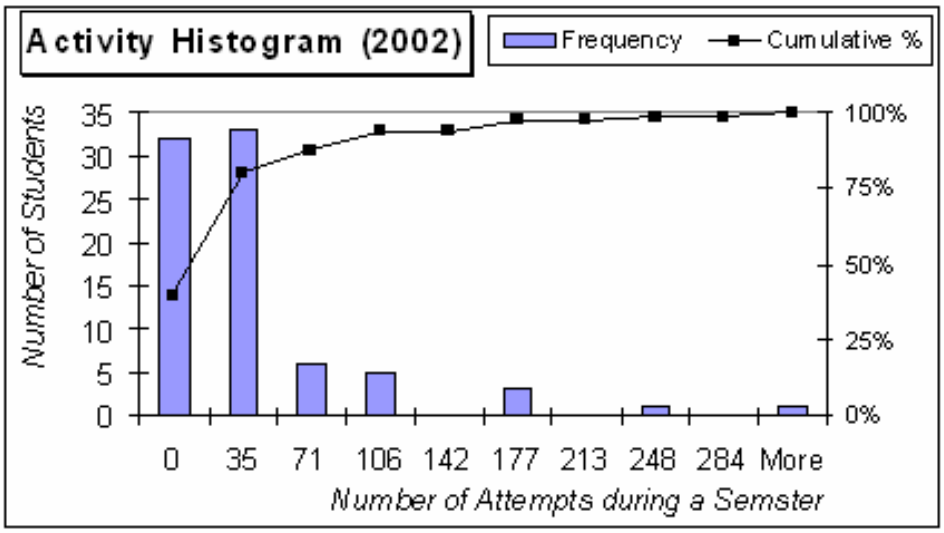

Fig. 2. Profile of QuizPACK 2002. Each bar shows the number of students trying a specified range of questions.

Table I. Regression Analysis Results (class performance versus QuizPACK measures, fall 2002)

Model	Number of observations	Sum of squares		Mean Squares		r^2	F-statistics (df)	p-value
		SSE(df)	SSR(df)	MSE	MSR			
In-Class Quizzes = (*activity*)	22	0.3103 (1)	0.4643 (20)	0.3103	0.0232	0.401	13.37 (1, 20)	0.0016
Final Exam Grade = (*success*)	22	0.2219 (1)	1.1056 (20)	0.2219	0.0553	0.167	4.01 (1, 20)	0.0589
Final Grade = (*activity*, *success*)	22	0.1546 (2)	0.4541 (19)	0.0773	0.0239	0.254	3.23 (2, 19)	0.0618

dent's *grade on the final exam*, that is, *in-class quiz performance* and the *final course grade* (all measured in percents). Note that in-class quizzes used in fall 2002 were in regular multiple-choice format, not paper versions of QuizPACK. Explanatory variables were *activity* and *success*. Table I presents the results for the 2002 fall semester.

As we can see, there is a strong, statistically significant relationship between the grade for in-class quizzes and student activity with QuizPACK: $F(1,20)=13.37$, $p=0.0016$. The coefficient of determination for this model is also very high: $r^2=0.401$, which means that 40% of the *grade for in-class quizzes* can be explained by the *activity* variable. We also discovered the dependence between *success* and the *final exam grade* and between both *activity* and *success* and the *final course grade*. These two models do not show the same level of confidence, however; p-values for both are very close to the significance threshold value of 0.05; the values of the coefficient of determination for both models are also fairly high.

Since in-class quizzes measure knowledge of semantics, we concluded that working with QuizPACK significantly increases the student's knowledge of C language semantics. We also discovered some evidence that working with the system contributes to the growth of higher-level knowledge and skills, which are measured by the final exam and course grade. As we note above, this data allowed us to regard QuizPACK as a strong learning tool and to experiment with increasing student motivation to use QuizPACK by switching to QuizPACK-based classroom quizzes in 2003.

3.2.2 *Using Parameterized Self-Assessment Quizzes with Additional Motivation.* During the 2003 spring and fall semesters, 73 students had access to QuizPACK, 60 tried it at least once, and 51 (70%) students worked with the system regularly; the average number of attempted questions was about 110.

Table II. QuizPACK Usage in 2002 and 2003

Year	Total students	Students who used the tool at least once	Active students	Average activity	Average # of sessions	Average course coverage	Average success
2002	81	49	22 (27%)	37.93	3.44	12.91%	49.34%
2003	73	60	51 (70%)	110.18	9.18	37.56%	43.55%

As we can see from Table II, the use of paper-based QuizPACK quizzes for regular classroom assessment changed the amount of system use dramatically. We measured the quantity of student work with QuizPACK using several measures. The average student *activity* (number of questions attempted) measures the plain volume of student work. The average *number of sessions* (number of independent sessions with the system) and the average *course coverage* (the ratio of questions attempted at least once, compared to the total number of questions) measure the distribution of this work over the duration of the course and course topics. The *percentage of active students* (students who attempted to answer more than 30 questions) measures the distribution of this work over the students of the class. All these performance measures *tripled or nearly tripled* in 2003 in comparison with 2002 (it is surprising that the rates of growth for these different measures parallel each other).

The profile of student activity also changed considerably. Figure 3 shows both the very active use of QuizPACK by students and its even distribution among them. The maximum number of questions attempted by a single student rose to 510. At the same time, in 2003, the average success score measuring the quality of student work decreased slightly. We can suggest at least one reason for this: average course coverage tripled. The students started to use more complicated quizzes regularly (which caused lower success rates) from the advanced sections of the course (which were ignored by most students in 2002).

To check the relationship between student work with QuizPACK and class performance in the new context, we performed a regression analysis similar to the one done with the 2002 data. In addition to the three class performance measures, we introduced a fourth: *knowledge gain*. We used this measure in hopes of finding a more reliable dependency between student work with QuizPACK and growth of their knowledge of semantics. The problem is that in an introductory programming course, the level of student knowledge of C and programming in general can vary significantly –

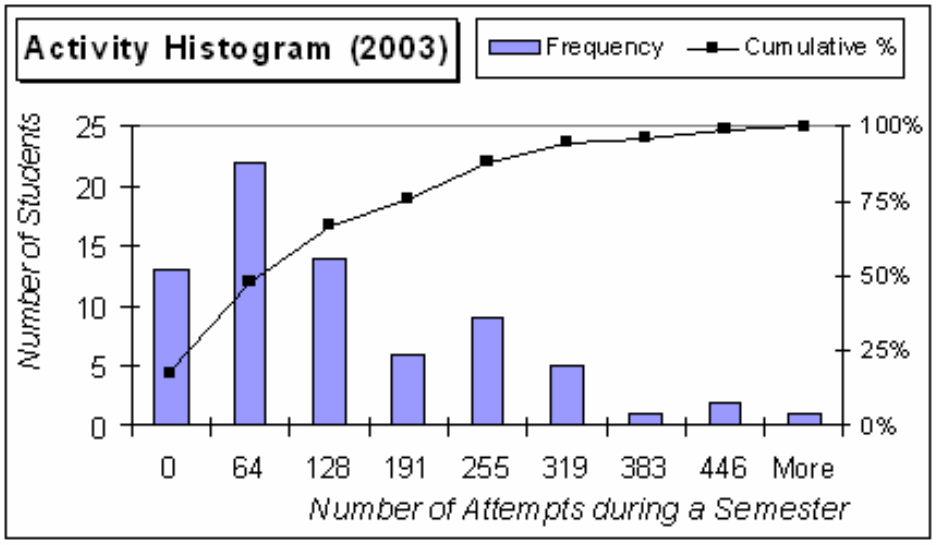

Fig.3. Profile of QuizPACK use in 2003; bars show the number of students who attempted the questions specified by the range.

Table III. Regression Analysis Results (class performance characteristics versus QuizPACK use, spring and fall 2003)

Model	Number of observations	Sum of square		Mean Squares		r^2	F-statistics (df)	p-value
		SSE (df)	SSR (df)	MSE	MSR			
In-Class Quizzes = (*activity*)	73	0.5877 (1)	4.0355 (71)	0.5877	0.0568	0.127	10.34 (1, 71)	0.0020
Knowledge Gain = (*activity*)	64	28.1175 (1)	285.6325 (62)	28.1175	4.607	0.09	6.10 (1, 62)	0.0163
Final Exam = (*success*)	60	1603.3632 (1)	7241.3491 (58)	1603.3632	124.851	0.181	12.84 (1, 58)	0.0007
Final Grade = (*activity*, *success*)	60	0.3936 (2)	1.3337 (57)	0.1968	0.0234	0.228	8.41 (2, 57)	0.0006

from complete novices to students who write reasonably-sized programs (the latter arguing that they need this course as a refresher). It was natural to expect that student performance measured at the end of the course depended not only on their work over the duration of the course (including QuizPACK work), but also on their past knowledge. To isolate the past experience factor, we administered a pretest (before the first lecture on C) and a posttest (during the final exam). The paper-and-pencil pretest and posttest featured the same ten QuizPACK fill-in-the-blank questions with different parameters. We calculated the *knowledge gain*, which measured the growth of student knowledge of semantics, as the difference between the posttest and pretest scores.

Table III shows the results of the relationship analysis between QuizPACK work and course performance for the two semesters of 2003. The *grade for in-class quizzes* and the *final course grade* were measured in percents; the *final exam grade* and the *knowledge gain*, in points. The new analysis confirmed the significant connection between QuizPACK work and the growth of students' knowledge of semantics; both measures -- the *knowledge gain* and the *grade for in-class quizzes* -- correlate significantly with the amount of work with QuizPACK. In addition, the larger number of students working actively in 2003 allowed us to discover significant relationships between QuizPACK and the remaining two course performance measures. We found that the *final exam grad*e correlates with the *success* variable. Students who strive to answer QuizPACK questions correctly have generally better grades on the final exam. The *final course grade* depends on both QuizPACK use measures. The p-values for all the models are well below the 0.05 threshold. Note that for models based on activity, we considered all 73 students in the class (only 64 completed both the pretest and posttest that we need to calculate knowledge gain). For models based on success, we considered only the 60 students who had used QuizPACK at least once, since success can't be calculated for students who have not used the system.

3.2.3 *QuizPACK*: *A Learning Tool in Programming Classes.* While our studies uncovered the relationship between usage of QuizPACK and class performance, this

relationship does not imply the direction of dependency. Indeed, the most natural way to explain the relationship is to claim that work with QuizPACK allows everyone to achieve better knowledge. It could be, however, that students in our course who already had a strong programming potential (due to a programming talent or a solid beginning level of knowledge) would have achieved better grades anyway, but simply preferred to use QuizPACK more. Fortunately, the specific organization of our course helped us to discover evidence in favor of the former scenario.

The introductory programming course that is the focus of our research consists of two components. The goal of the first component (6 lectures) is to introduce basic programming concepts (such as conditions and loops) with the help of Karel the Robot [Pattis et al. 1995], a simple, visual minilanguage [Brusilovsky et al. 1997]. The goal of the second part (16 to18 lectures) is to gain programming knowledge and skills while learning a reasonable subset of the C language. Both QuizPACK and paper-and-pencil quizzes were used for the second part of the course. While Karel and C share a solid number of programming concepts and constructs, they require different levels of effort to master. Karel, due to its simplicity and visual programming environment, can usually be mastered relatively easily. Students with strong programming potential typically achieve very good results on the Karel part with little effort. In contrast, the C language has an abundance of second-level details, which require a significant effort to master, even from students with strong programming potential. In the process of mastering the C language, QuizPACK may provide support that could result in changed scores.

To determine whether QuizPACK has any pedagogical effect we compared student performance on the C parts of the course with their overall performance. We calculated *class performance change* by dividing the grade for the C part (in percent) by the total course grade (in percent) that including both Karel and C performance. The grades for the Karel and C parts were compiled from grades for homework assignments and exam questions that focused on Karel and C, respectively. The *class performance change* measures the relative change in performance during the C part of the course, as compared to the whole course (Karel plus C). The *class performance change* is less than 1 for students who achieved better results in the Karel part of the course and greater than 1 for students who achieved better results in the C part. The larger this proportional change, the better the relative grade increase for the C part. If the work with QuizPACK can help students improve their “native” programming performance, as measured by the Karel grade, we may expect a positive correlation between change in class performance and the amount of work with the system. If the work with QuizPACK harms student progress, we may expect a negative correlation. If the student performance is mainly determined by their programming potential and other aspects not related to QuizPACK, we may expect no correlation between these parameters (and we can expect no visible difference between the C part and whole-class performances). While any of these conclusions still cannot be considered an ultimate proof, the *class performance change* analysis provides more reliable evidence about the role of QuizPACK than simple correlation analysis.

Table IV demonstrates the results of regression analysis on *class performance change* versus *activity*. There is strong statistical evidence that change in course performance is connected to use of QuizPACK: $F(1,43)=7.79$, $p=0.0078$. The value of the coefficient of determination tells us that 15.3% of the change in the students' grade for the C part of the course can be explained by the *activity* of their work with self-assessment quizzes. This data gives as better evidence to conclude that working with QuizPACK caused the observed growth of the students’ knowledge. If we hypothesize that student performance on the Karel part reflects their programming potential, then work with QuizPACK allows

Table IV. Regression Analysis Results (change in class performance versus activity with QuizPACK, spring 2003)

Model	Number of observations	Sum of squares		Mean Squares		r^2	F-statistics (df)	p-value
		SSE(df)	SSR(df)	MSE	MSR			
Class Performance Change = (*activity*)	45	0.2063 (1)	1.1385 (43)	0.2063	0.0265	0.153	7.79 (1, 43)	0.0078

the students to *leverage* their programming potential and even reach beyond it. Students who have not invested in the effort to learn the details of C with help from QuizPACK may expect their course performance to drop below their potential. In contrast, the students who worked with QuizPACK extensively may expect to perform above their starting potential.

In Fall 2003, we used another approach to help determine the role of QuizPACK as a learning tool. During that semester, the School offered two sections of the Introduction to Programming course, naturally forming two groups: experimental (28 students) and control (22 students). The syllabus and class settings for both classes were comparable. The students in the experimental group had access to QuizPACK during the course (and were motivated by their classroom quizzes to use it), whereas students in the control group had no access to the system. At the beginning and end of the course, each group took the same pre- and post-tests. The results of these tests were used to calculate knowledge gain. The average knowledge gain for the control group was 1.94 (Sigma = 1.55); for the experimental group it was 5.37 (Sigma = 2.17). In other words, motivated use of QuizPACK more than doubled the knowledge, providing a difference that is larger than 1.5 standard deviations. Such a result is typically considered to be very strong evidence in favor of an educational technology [Bloom 1984].

In addition to determining the value of QuizPACK to the class as a whole, we attempted to determine in our research whether QuizPACK provides different impacts on different categories of students. We were interested in finding the categories of students who benefit more or less from the system than others. As categorical variables we attempted to divide the group by such students' characteristics as gender, domain knowledge (measured by final letter grade), initial programming experience, and even learning style. Unfortunately, none of these variables were able to divide students into categories that were differently influenced by QuizPACK usage. We tried both simple classification and clustering approaches. To reveal possibly hidden groups of students formed by combinations of categorical variables, we applied the expectation-maximization algorithm for Naïve Bayes with a hidden class variable and missing values. This algorithm has many desirable features: a strong statistical basis, theoretical guarantees about optimality, easily explainable results, and robustness to noise and to highly skewed data. [Dempster et al. 1977]. The cluster picture we received did not have intuitive interpretation and did not match the data from semesters not used for model training.

3.3 Subjective Evaluation of QuizPACK Value

To evaluate the students' subjective attitudes toward the system, we administered a questionnaire at the end of each semester to each class. The questionnaire included a

Table V. Six Questions About QuizPACK and Summary of Student Answers

Question	Answer		Number of answers		
			2002	2003	Total
1. I think that self-assessment quizzes:	3	significantly helped me during the course	20	37	57
	2	helped me during the course	25	8	33
	1	sometimes were of help	8	3	11
	0	were useless for me during the course	0	0	0
2. The system's ability to generate the same question with different data provides a chance to work with the same question again and again. I think, this feature:	3	is very useful	18	36	54
	2	is useful	24	10	34
	1	could be useful, but in very few cases	11	2	13
	0	is useless	0	0	0
3. The type and the content of self-assessment quizzes was:	3	Exactly right to be most helpful	5	11	16
	2	Good and helpful overall	43	33	76
	1	Sometimes helpful, could be much better	5	4	9
	0	Not helpful at all	0	0	0
4. Considering this interface of QuizPACK system I think that:	3	It is very good	12	23	35
	2	It is good	32	18	50
	1	Have some problems or lacks some features	8	7	15
	0	Have some major problems	1	0	1
5. I think that in the context of IS0012 course self assessment questions:	3	should become one of the key courseware elements and provided with each lecture	12	29	41
	2	are very helpful and should be provided in several critical points	25	13	38
	1	could be useful for some students and probably should be provided as a part of exam preparation	16	6	22
	0	hardly worth time spent for their preparation	0	0	0
6. I will recommend my friends taking this course next semester to use self-assessment quizzes:	3	strongly agree	7	28	35
	2	agree	12	18	30
	1	no strong opinion	4	2	6
	0	disagree	0	0	0

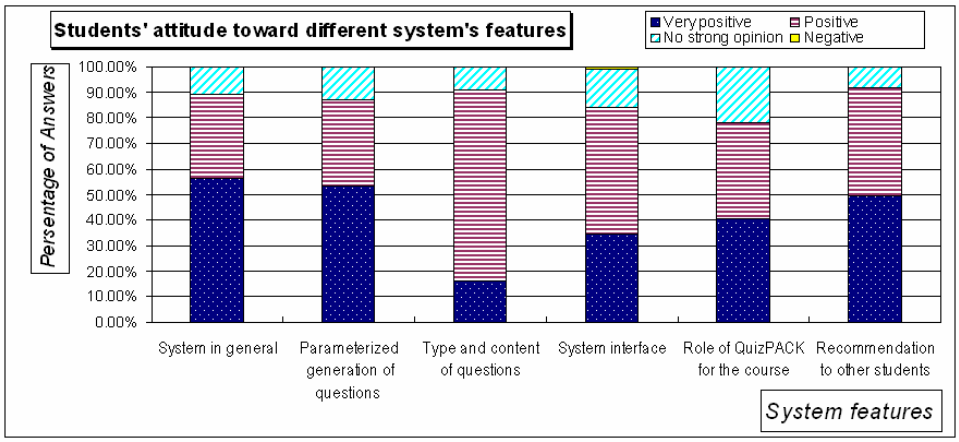

Fig.4. Students' attitude toward the system (summary statistics over four semesters).

range of questions focused on the system's value, its features, and aspects of its usage. The number of questions grew from 9 in spring 2002 to 14 in fall 2003. Participation was voluntary, but to motivate students we offered three extra credit points to every qualified participant. A student was qualified to take the questionnaire if he or she worked on at least 10 or more 5-question quizzes, associated with 6 or more different lectures, and used this feature during several sessions for at least 20 days before being offered the chance to take the questionnaire. In total, 101 filled-in questionnaires were collected for the four semesters of 2002 and 2003.

In the following analysis we focus on the six multiple-choice questions that are most relevant to the topic of this article. The questions, the possible answers, and the number of students that selected each answer are listed in Table V. Each of these questions (as well as the majority of questions in the questionnaire) provided a student with four optional answers: *strongly positive*, *positive*, *neutral*, or *negative*. These choices were coded by values from 3 (strongly positive) to 0 (negative). The profile of student answers is summarized in Figure 4. This chart represents the feedback of all 101 students over the 4 semesters (with the exception of question 6 which was not used in the spring semester of 2002).

As this analysis shows, the students were very positive about the system; on average, ~~o~~80% to 90% students provided at least somewhat positive answers to the questions. More than 50% said that the system *significantly* helped them during the course. More than 40% considered the system a critical part of the course and were ready to recommend it strongly to other students. We have been evaluating different systems in the classroom and this is the strongest result we have ever achieved. In our class, QuizPACK was the *clear champion*. When students compared all support tools they used in class, they rated QuizPACK as the second most valuable, right after lecture slides (in spring 2002 they actually preferred QuizPACK). Even the relatively plain system interface – arguably the least-appreciated feature of the system - received more than 80% light-positive feedback.

We also noticed that the profile of student answers to question 2, on the system's ability to generate the same question with different data, is close semester-by-semester to the profile of their answer to question 1 on the helpfulness of QuizPACK. We hypothesized that the students' highly positive attitude to QuizPACK is strongly

connected with its ability to deliver and evaluate parameterized quizzes; statistical analysis confirms this finding. The correlation coefficient between the general attitude to the system and to question parameterization is fairly high – 0.48; it is larger than corresponding correlation coefficients for any other system feature.

The regression analysis of student attitudes to QuizPACK in general against attitudes to different features of the system supports these results (see Table VI). Student opinion about three main features (parameterized question generation, type and content of question material, and system interface) have a statistically significant relationship to the evaluation of the system in general. Question generation has both the strongest relationship ($F(1,99)=29.52$, $p=3.98E-7$) and the largest coefficient of confidence ($r^2=0.23$) among all the features. However, the most interesting model seems to be the last one, which regresses the general attitude to the system against all three partial attitudes. This model has even larger values of confidence ($p=1.48E-8$) and coefficient of determination ($r^2=0.332$). It means that all three features contributed significantly to the overall student attitude to the system.

To complement the general analysis, we compared the attitudes of different groups of students to QuizPACK. We determined three characteristics for dividing students into groups: gender, final course grade, and initial programming experience (novice, some experience, considerable experience). We calculated an average attitude for each of the groups for every question and every semester by averaging the answer codes (Table V). Then we compared the profiles of their answers for each feature. We were not able to find any consistent difference between groups of students with different initial experiences. But students of different genders as well as different final course grades did demonstrate distinct patterns that were repeated semester by semester.

Figure 5 shows the difference in profiles for the answers given by female and male students. The average answers in this figure are shown in percentage of maximum positive answers (i.e., 100% means that all students in this category give the strong-positive answer). It is remarkable that the graph corresponding to female students dominates the one for males for almost all questions (except the one about parameterized generation). The difference is quite visible; it varies from 3% to almost 10% for several

Table VI. Regression Analysis Results (on student attitudes to QuizPACK in general versus attitudes to various system features, spring and fall 2003)

Model	Number of observations	Sum of squares		Mean squares		r^2	F-statistics (df)	p-value
		SSE (df)	SSR (df)	MSE	MSR			
General Attitude = (*question generation*)	101	10.8057 (1)	36.2438 (99)	10.8057	0.3661	0.230	29.52 (1, 99)	3.98E-7
General Attitude = (*question material*)	101	7.7817 (1)	39.2678 (99)	7.7817	0.3966	0.165	19.62 (1, 99)	2.44E-5
General Attitude = (*system interface*)	101	4.3136 (1)	42.7359 (99)	4.3136	0.4318	0.092	9.99 (1, 99)	0.002086
General Attitude = (*parameterization, question material, system interface*)	101	15.6185 (3)	31.4310 (97)	5.2062	0.3240	0.332	16.07 (3, 97)	1.48E-8

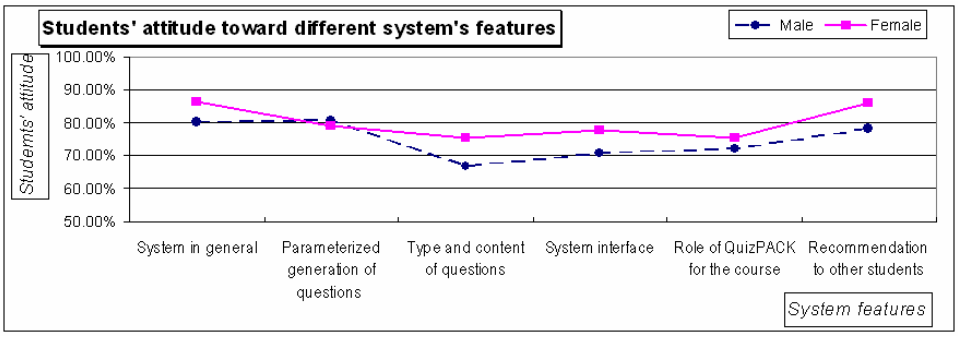

Fig. 5. Student attitudes to the system (gender distribution).

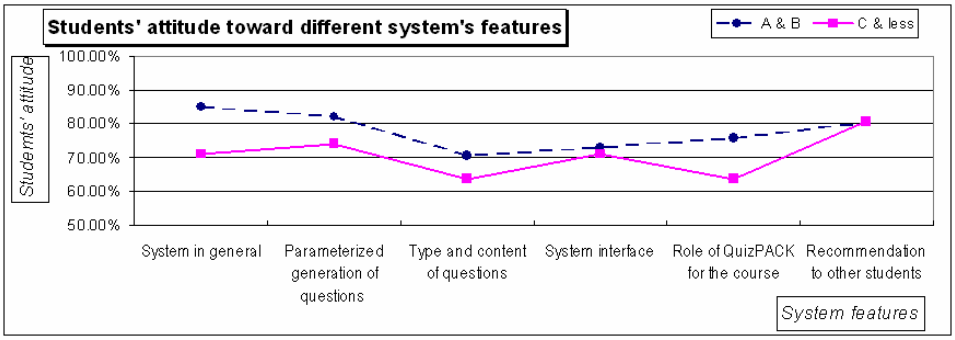

Fig.6. Student attitudes toward the system (grade distribution).

questions. While Figure 5 summarizes the comparative data over four semesters, we observed the same dominance every semester. The more positive female attitude correlates with the fact, that, on average, female students were more active with QuizPACK, (For students who participated in the questionnaire, the average values of *activity* for female and male students were about 130 and 98, respectively.) At the same time, the difference between average values of *success* for female and male students was almost zero (51.78% of correct answers for female students and 50.66% for males). We hypothesize that the very positive attitude of female students towards QuizPACK and their higher level of activity with the system express the fact that QuizPACK was a more critical tool for female students over the duration of the course. It is remarkable that in our course female students, who usually lag behind male students in programming subjects, achieved the same average grade as males.

Figure 6 demonstrates a similar influence on students' knowledge of the subject as measured by the final grade. One of the lines on the graph shows the average for answers by students who got positive final course grades (from B- to A+); another line shows the average for the answers of the rest of the class. As we see from the figure, the *AB* line is above the *C&less* line for all questions but one. For two questions, the difference is about 15%, for three others it is close to 5%. The difference in QuizPACK activity for these student groups is very similar to the difference in attitude. The average value of *activity* for AB-students is about 120, while for CDF-students it is only 47. The average values for success for these two groups are nearly the same (51.07% and 50.52%).

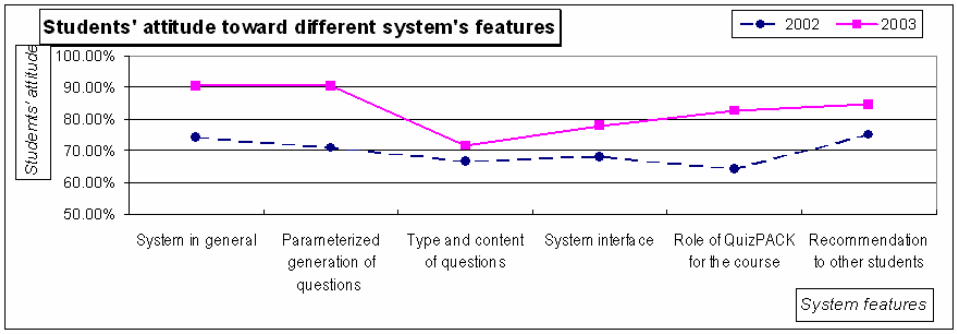

Fig. 7. Change in student attitude to the system after introduction of QuizPACK-based classroom quizzes.

Figure 7 shows the change in attitude toward the system after the introduction of QuizPACK-based paper quizzes. As we can see, in 2003 students were much more positive about QuizPACK than they were the year before. Not surprisingly, their attitudes toward the type and content of questions that were mostly the same in 2002 and 2003 changed very moderately. However, their appreciation of the parameterized question generation, which became more important in the 2003 system, went up dramatically, from 70% to 90%. Similarly, their overall attitude toward the system, for example the evaluation of its helpfulness or its role in the course, rose about 20%.

The differences presented above may have two explanations. First, we can argue that the system is simply more helpful for the groups of students who demonstrate better attitudes toward the system (i.e., females, students with better knowledge of the subject, and students motivated by QuizPACK-based paper quizzes), which would also explain the greater use of the system by these groups. However, the opposite may also be true: that greater use of the system is the cause of a better attitude. Further studies are required to compare the value of the system for male and female students and those with varying levels of knowledge.

3.4 Assessment or Self-Assessment?

As we pointed out above, our early attempts to use QuizPACK for classroom assessment caused generally negative student reactions. To elicit their opinions about the most appropriate context in which to use the system, we added to every questionnaire administered in 2002 and 2003 a multiple-answer question asking students to check all contexts in which they would like to use QuizPACK (Figure 8). As we can see in the figure, the option "*right in class in assessment mode to replace the paper quiz*" was the least popular. Only 23 students (about 10% of the total number) selected it, while the remaining 90% selected self-assessment contexts exclusively. Among the scenarios for using QuizPACK for self-assessment, out-of-class self-assessment was the champion.

To elicit student feedback about the new arrangement for in-class quizzes, in 2003 we added two questions to the questionnaire. An analysis of the answers for the spring and fall semesters of 2003 show that students mostly approved. The question on the use of QuizPACK-based paper-and-pencil quizzes for classroom assessment elicited the following opinions: 66.67% of students wrote that "*it was a very good arrangement*"; 25.00% believed, that "*it was quite good*"; 8.33% answered "*it made some sense, though it was far from perfect*"; and no one answered that "*it was a completely wrong arrangement.*" The question of assessing the value of QuizPACK as a tool to prepare for classroom quizzes drew the following answers: 64.58% of students answered that "*it*

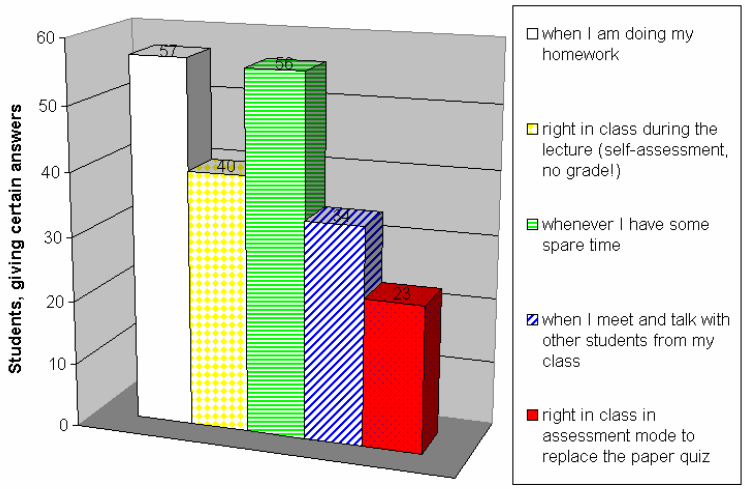

Fig. 8. Student context preferences using QuizPACK (summary statistics: 2002 and 2003).

helped them a lot"; 27.08% that "*it was quite helpful*"; 8.33% that "*it helped a little*"; but no one answered that "*it did not help at all.*" Thus about two-thirds of students gave strong positive feedback on the new arrangement, with more than 90% of the total being positive or strongly positive.

We think that the answers given by the students to the three questions analyzed above provide good evidence that our most recent arrangement is one of the most attractive and useful ways for students to use the parameterized code evaluation quizzes. This arrangement provides an interesting combination of assessment and self-assessment: using QuizPACK to deliver automatically-graded self-assessment quizzes in an out-of-class context while assessing students in the classroom via paper-and-pencil quizzes generated with QuizPACK.

4. QUIZPACK AND THE AUTOMATED EVALUATION OF PROGRAMMING KNOWLEDGE

This article is not complete without an attempt to place QuizPACK in the context of other work on automated evaluation of programming knowledge. There are several ways to classify the range of existing systems. From the authors' perspective, the most important classification is the kind of knowledge that a specific system is trying to assess. In the past, it was customary to distinguish between three kinds of knowledge: knowledge of syntax (how to compose valid programs); semantics (how programming constructs work); and pragmatics (how to solve commonplace problems). All three kinds of knowledge are evaluated in various tests and assignments. During the past 15 years, researchers focused mainly on the knowledge of semantics and pragmatics (knowledge of syntax is considered much less important nowadays than it was in the early days of computer science education). Correspondingly, two kinds of automated assessment systems emerged: program evaluation and program generation. Systems based on program evaluation attempted to test the students' knowledge of semantics by asking them to trace (evaluate) an existing program and then to assess the results of the trace. Systems based on writing programs attempted to test the students' knowledge of pragmatics by asking them to write (generate) code and then assess the code.

Since the ability to solve programming problems and produce working code are considered most important for computer science graduates, the majority of work on automated evaluation of programming knowledge is devoted to the latter kinds of systems. A number of program assessment systems have been produced over the last 15 years; they differ in two ways: the amount of code they require the student to write and the level of code analysis. Question-oriented systems typically require the student to fill a relatively small gap in an existing program [Arnow and Barshay 1999; Garner 2002]. Assignment-oriented systems assess complete programming assignments, often written from scratch or from a small program skeleton [Benford et al. 1994; Daly 1999; Higgins et al. 2006; 2003; Jackson and Usher 1997; Joy et al. 2006]. While the technology behind these systems is often similar, the question-oriented systems mostly assess the student's ability to use programming constructs in context, while the assignment-oriented systems mostly assess the ability to solve programming problems. These skills belong to different levels of Bloom's Taxonomy of Educational Objectives [Bloom 1956].

The QuizPACK system described in this article belongs to the first class of systems based on program evaluation (tracing); currently they are less popular among developers. In addition to QuizPACK, we can only cite a range of tutors developed by Amruth Kumar [Dancik and Kumar 2003; Fernandes and Kumar 2002; Kostadinov and Kumar 2003; Kumar 2006; 2002; 2005]. The evaluation-based approach, however, is a popular way of assessing student knowledge in the areas of algorithms and data structures. Systems like PILOT [Bridgeman et al. 2000]; TRAKLA [Korhonen and Malmi 2000]; TRAKLA2 [Malmi et al. 2006]; and MA&DA [Krebs et al. 2005] are similar to QuizPACK and Kumar's tutors: they ask a student to mentally execute an algorithm from a specific starting state and submit a trace of actions or the final results of algorithm execution. We can hope that all the cited work will draw more attention to the importance of assessing students' knowledge of semantics and their program-tracing skills. We think that this knowledge is an important component of the student's ability to understand and write programs. This opinion is shared by a large team of researchers [Lister et al. 2004], which has recently stressed the importance of assessing student program-tracing skills.

5. SUMMARY AND FUTURE WORK

This article explores a specific kind of educational activity in programming: parameterized code-execution exercises. In a code-execution question, the student is given a fragment of a program and is asked to predict the value of a particular variable or a string to be printed after execution of this fragment. We suggest an approach to the automatic generation and evaluation of code-execution exercises based on the runtime execution of parameterized code fragments. We also developed the QuizPACK system, which implements the suggested approach for the specific case of teaching the C language. The focus of this article is on the classroom evaluation of the QuizPACK system over four consecutive semesters in an introductory programming class.

Our studies demonstrate that QuizPACK, used in an out-of-class self-assessment mode, is an exceptional learning tool. As demonstrated by regression analysis and a comparison with a control group, working with QuizPACK significantly improved the students' knowledge of semantics and positively affected higher-level knowledge and skills. The students themselves praised the system highly as a helpful learning tool. About 90% of the students stated that the system helped or significantly helped them during the course and that they would recommend it to their friends taking the same course. The ability to generate parameterized exercises was rated especially high among

system features and, as shown by correlation analysis, contributed to the overall satisfaction with the system.

We also demonstrate that the use of the system in self-assessment mode can be significantly increased (with a corresponding increase in positive student attitudes and knowledge) by motivating students through QuizPACK-based classroom paper-and-pencil quizzes. While we originally planned to use QuizPACK for automating knowledge evaluation in the classroom, our data indicates that a combination of QuizPACK-based classroom paper-and-pencil quizzes and out-of-class web-based self-assessment with QuizPACK is among the most student-preferred ways of using the system.

Our current work is focused on providing guidance for the users of QuizPACK to help them select the most useful exercises in order to advance their learning goals and level of knowledge. Our first experiments with adaptive guidance demonstrated that in addition to guiding the students to the most relevant exercises, it also increased student motivation to continue to work with the system and so further improve their knowledge [Brusilovsky and Sosnovsky 2005]. We plan a more comprehensive exploration of this phenomenon. We also hope that the QuizPACK system will begin to be used more broadly, enabling us to collect more data and perform some deeper analyses of QuizPACK's influence on different categories of students. In particular, we intend to explore QuizPACK with new categories of students, such as high school and community college students.

REFERENCES

ARNOW, D. AND BARSHAY, O. 1999.WebToTeach: A web-based automated program checker. In *Proceedings of the Frontiers in Education Conference* (FIE99, San Juan, Puerto Rico).

BENFORD, S.D., BURKE, E.K., FOXLEY, E., AND HIGGINS, C.A. 1994. Ceilidh: A courseware system for the assessment and administration of computer programming courses in higher education. In *Proceedings of the Interdisciplinary Workshop on Complex Learning in Computer Environments* (CLCE94, Joensuu, Finland).

BLOOM, B.S. 1956. *Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain*. David McKay Co. Inc., New York.

BLOOM, B.S. 1984. The 2 sigma problem: The search for methods of group instruction as effective one-to-one tutoring. *Edu. Res. 13*, 3.

BRIDGEMAN, S., GODDRICH, M.T., KOBOUROV, S.G., AND TAMASSIA, R. 2000. PILOT: An interactive tool for learning and grading. In *Proceedings of the 31st ACM SIGCSE Technical Symposium on Computer Science Education* (Austin, TX). 139-143.

BRUSILOVSKY, P. 1992. Intelligent tutor, environment and manual for introductory programming. *Edu. Training Tech. Int. 29*, 1, 26-34.

BRUSILOVSKY, P. 1994. ILEARN: An intelligent system for teaching and learning about UNIX. In *Proceedings of the SUUG International Open Systems Conference* (Moscow), 35-41.

BRUSILOVSKY, P. KnowledgeTree: A distributed architecture for adaptive e-learning. In *Proceedings of the 13th International World Wide Web Conference* (WWW 2004, New York). Alternate track papers and posters, 104-113.

BRUSILOVSKY, P., CALABRESE, E., HVORECKY, J., KOUCHNIRENKO, A., AND MILLER, P. 1997. Mini-languages: A way to learn programming principles, *Edu. Inf. Technol. 2*, 1, 65-83. Available at http://www.chapmanhall.com/ei/sample/ei020106.pdf.

BRUSILOVSKY, P. AND MILLER, P. 1999. Web-based testing for distance education. In *Proceedings of the WebNet'99 World Conference of the WWW and Internet* (Honolulu, HI), 149-154.

BRUSILOVSKY, P. AND MILLER, P. 2001. Course delivery systems for the virtual university. In: *Access to Knowledge: New Information Technologies and the Emergence of the Virtual University*. T. Tschang and T. Della Senta, eds. Elsevier Science, Amsterdam, 167-206.

BRUSILOVSKY, P. AND SOSNOVSKY, S. 2005. Engaging students to work with self-assessment questions: A study of two approaches. In *Proceedings of the 10th Annual Conference on Innovation and Technology in Computer Science Education* (ITiCSE'2005, Monte de Caparica, Portugal), 251-255.

DALY, C. 1999. RoboProf in an introductory computer programming course. *SIGCSE Bull. - Inroads 31*, 3, 155-158.

DANCIK, G. AND KUMAR, A.N. 2003. A tutor for counter-controlled loop concepts and its evaluation. In *Proceedings of the 2003 Frontiers in Education Conference* (FIE 2003,Boulder, CO), Session T3C.

DEMPSTER, A.P., LAIRD, N.M., AND RUBIN, D. 1977. Maximum likelihood estimation from incomplete data via the EM algorithm. *J. Royal Stat. Soc. 39*, 1, 1–38.

FERNANDES, E. AND KUMAR, A.N. 2004. A tutor on scope for the programming languages course. In *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education* (Norfolk, VA).

GARNER, S. 2002. Reducing the cognitive load on novice programmers. In *Proceedings of the ED-MEDIA'2002 - World Conference on Educational Multimedia, Hypermedia and Telecommunications* (Denver, CO), 578-583.

GRAHAM C.R., SWAFFORD, M.L., AND BROWN, D.J. 1997. Mallard: A Java enhanced learning environment. In *Proceedings of the WebNet'97 World Conference of the WWW, Internet and Intranet* (Toronto, Ont.), 634-636.

HIGGINS, C., GRAY, G., SYMEONIDIS, P., AND TSINTSIFAS, A. 2006. Experiences on automatically assessed algorithmic exercises. *ACM J. Edu. Resources Comput.* In this issue.

HIGGINS, C., HEGAZY, T., SYMEONIDIS, P., AND TSINTSIFAS, A. 2003 The CourseMarker CBA system: Improvements over Ceilidh. *Edu. Inf. Technol. 8*, 3, 287-304.

JACKSON, D. AND USHER, M. 1997. Grading student programs using ASSYST. *SIGCSE Bull. 29*, 1, 335-339.

JOY, M., GRIFFITHS, N., AND BOYATT, R. 2006. The BOSS online submission and assessment system. *ACM J. Edu. Resources Comput.* In this issue.

KASHY, D.A., ALBERTELLI, G., ASHKENAZY, G., KASHY E., NG, H.-K., AND THEONNESSEN, M. 2001. Individualized interactive exercises: A promising role for network technology. In *Proceedings of the 31st ASEE/IEEE Frontiers in Education Conference* (Reno, NV).

KASHY, E., THEONNESSEN, M.., TSAI, Y., DAVIS, N.E., AND WOLFE, S.L. 1997. Using networked tools to enhance student success rates in large classes. In *Proceedings of the 27th ASEE/IEEE Frontiers in Education Conference* (Pittsburgh, PA), 233-237.

KORHONEN, A. AND MALMI, L. 2000. Algorithm simulation with automatic assessment. In *Proceedings of the Fifth Annual SIGCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education* (ITiCSE 2000, Helsinki), 160-163.

KOSTADINOV, R. AND KUMAR, A.N. 2003. A tutor for learning encapsulation in C++ classes. In *Proceedings of the ED-MEDIA 2003 World Conference on Educational Multimedia, Hypermedia and Telecommunications* (Honolulu, HI), 1311-1314.

KREBS, M., LAUER, T., OTTMANN, T., AND TRAHASCH, S. 2005. Student-built algorithm visualizations for assessment: Flexible generation, feedback, and grading. In *Proceedings of the Tenth Annual Conference on Innovation and Technology in Computer Science Education* (ITiCSE'2005, Monte de Caparica, Portugal), 281-285.

KUMAR, A.N.2006. Generation of problems, answers, grade and feedback - Case study of a fully automated tutor. *ACM J. Edu. Resources Comput*. In this issue..

KUMAR, A.N. 2002. A tutor for using dynamic memory in C++. In *Proceedings of the Frontiers in Education Conference* (FIE 2002, Boston, MA), Session T4G.

KUMAR, A.N. 2005. Results from the evaluation of the effectiveness of an online tutor on expression evaluation. In *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education* (St. Louis, MO), 216-220.

LILSTER, R. ET AL. 2004. A multi-national study of reading and tracing skills in novice programmer. *SIGCSE Bull. 36*, 4, 119-150.

MALMI L., KARAVIRT, V., KOHONEN A., AND NIKANDER, J. 2006. Experiences on automatically assessed algorithmic exercises. *ACM J. Edu.Resources Comput.* In this issue.

MERAT, F.L. AND CHUNG, D. 1997. World Wide Web approach to teaching microprocessors. In *Proceedings of the FIE'97 Frontiers in Education Conference* (Pittsburgh, PA), 838-841.

PATHAK, S. AND BRUSILOVSKY, P. 2002. Assessing student programming knowledge with web-based dynamic parameterized quizzes. In *Proceedings of the ED-MEDIA'2002 - World Conference on Educational Multimedia, Hypermedia and Telecommunications* (Denver, CO),1548-1553.

PATTIS, R.E., ROBERTS, J., AND STEHLIK, M. 1995. *Karel the Robot, A Gentle Introduction to the Art of Programming*, 2nd ed. Wiley, New York.

SHAH, H.AND KUMAR, A.N. 2002. A tutoring system for parameter passing in programming languages, *SIGCSE Bull. 34*, 3, 170-174.

TITUS, A.P., MARTIN, L.W., AND BEICHNER, R.J. 1998. Web-based testing in physics education: Methods and opportunities. *Comput. Phys. 12* (Mar/Apr), 117-123. Available at http://www.webassign.net.

WEBER, G. AND BRUSILOVSKY, P. 2001. ELM-ART: An adaptive versatile system for web-based instruction, *Int. J. AI Edu.. 12*, 4, 351-384. Available at http://cbl.leeds.ac.uk/ijaied/abstracts/Vol_12/weber.html.

Received August 2004; accepted December 2004