GÁBOR TÖRLEY

VISUALIZATION IN PROGRAMMING EDUCATION

THESES OF THE DOCTORAL DISSERTATION

PhD School of Informatics
Eötvös Loránd University Faculty of Informatics

The title of the PhD Program:
Basics of informatics and its methodology

Leader of the PhD School: Dr. András Benczúr
Leader of the PhD Programme: Dr. János Demetrovics
Advisor: Dr. Péter Szlávi

Budapest, 2013.
Introduction

The feature of this research is educational-methodological and it focuses on algorithmic thinking and the development of problem-solving skills.

The starting points of this research are the following:

1. According to PISA-report\(^1\) and Hungarian experiments\(^2\), the Hungarian teenagers are in the middle-rank of the international comparisons in point of problem-solving skills and creativity. That is why, it is a relevant goal to develop the problem-solving thinking of the students.

2. The current syllabuses in secondary education let only 20% of the informatics lessons to use for teaching programming and in general the number of informatics lessons are very few. In some secondary schools, there is opportunity to learn programming, but usually a pupil can learn programming in technical colleges.

3. In general, the used programming languages in education are for example Pascal, C, C++, BASIC, which runs in DOS environment.

4. Programming is the hardest chapter of informatics, even if the pupil is not motivated to learn that.

5. Teaching programming looks a suitable tool to solve the problem, which I mentioned in 1.

6. Among the aims of the Hungarian secondary education, the development of students’ cognitive skills and the improvement of their thinking gets more emphasis. Everybody stepping in the “Labyrinth of Life” needs conscious thinking to find a quick and efficient answer for everyday problems. Acquiring the ability of algorithmic thinking provides also help to reach this goal.

During the phase of algorithm planning, the student puts concrete ideas in order (thus elaborates an algorithm), then he/she takes time for reflection on these ideas, for classification of these ideas, and considers the strategy developed. It is important that he/she does not jump to conclusions too early. Finally, he/she evaluates the solution that he/she has found, and is able to see the initial problem in its integrity, as he/she has got to a solution through several efforts, often misdirected. He/she can afterwards recall his/her experiences,

---

1 Executive Summary PISA 2006: Science Competencies for Tomorrow’s World © OECD 2007
and use them during the resolution of future tasks. Like this, algorithms contribute to the
development of the cognitive skills of students.

Programming education can take a decisive role in improving the students’ cognitive
skills by teaching programming theorems (basic algorithms), however, according to the
experiences so far – abroad included \(^3\) \(^4\) –, it is rather complicated to learn and teach
algorithms. Today’s computer science education has an important task to develop the
teaching methods in this field. \(^5\) \(^6\)

Motivation of choosing the theme

In 2005, I did a questionnaire survey with 70 pupils (in the part of my thesis of the
master’s degree). One of the results was that only the one third of the pupils thought that the
textbook is useful for them to learn programming. This raises the question that why is it so
and is it able to make the programming education more efficient and to extend the textbook’s
textual and the teacher’s oral information with introducing a new tool.

According to my experiences as a teacher, the hardest part of learning programming
theorems is when we are searching for the answer to the question why that given algorithm
will be suitable and why that given theorem will solve the problem. That is why, the
“verification” of the given algorithm (of course, not with tools of math, at this level) is difficult,
because the pupil should understand the steps of the algorithm and the connections among
them. Further problem-duo is that if the pupil understood the algorithm of the theorem, then
how will “arose” the concrete algorithm for the concrete task and afterwards the functioning
code. It can be summarized in one problem: how can the pupil apply, specify the abstract
theorem on the given task.

As practice, I taught programming theorems in a secondary school in Budapest. I
demonstrated the mechanism of the given algorithm to the pupils by the help of an animation
nested in a presentation, so the pupils saw the animation and heard the explanation at the

\(^3\) Matti Lattu, Veijo Meisalo, Jorma Tarhio, A visualisation tool as a demonstration aid, Computers &
\(^4\) Urquiza-Fuentes, J., and Velázquez-Iturbide, J. (2013): Toward the effective use of educational program
animations: The roles of student’s engagement and topic complexity, Computers & Education, vol. 67, pp. 178
- 192
\(^6\) Szlávi Péter: Programkészítés és gondolkodás. Informatika a felsőoktatásban 2008 Konferencia, Debrecen
same time. I could teach them much more than if we have studied only the text of the algorithm. Beyond the visual and audible channel I could use the time as well, because the pupils could observe the temporal change of the algorithm. This effect is called “Multimedia effect” by Mayer\(^7\).

I concluded from my experiences that if I used tools during teaching that help to imagine what is happening inside the algorithm, I could promote the process of understanding. Algorithm visualization (AV) tools support this purpose.

Object oriented programming (OOP), which is one of the segment of teaching programming, became the most effective programming paradigm in the last few decades. OOP is widely used in secondary education, industry and higher education. According to programming communities, it is useful to teach OOP because this paradigm supports the learning of structured programming principles and the teaching of modularization and concept of design programming.

OOP’s approach can map the real world in a natural way, because the “things” around us can also describe with properties, state and methods so the OOP paradigm develop the ability of abstraction.

However, the teaching and learning of OOP is a challenge for the teacher and the student as well. On the one hand because of the requirement of abstract thinking (concept of classes), on the other hand the object, which is created by programming, will be “materialized” only at the end of the programming process.

According to the experience above, the understanding of the object’s functioning and the coding can take less time if we have tools which can help us to imagine the given object.

Kölling found\(^8\), that it is difficult to understand for some students the terminology of OOP because what they can see on the teacher’s presentation is the code itself. If the student is expected to talk and reason about classes, about their relationships, then it is useful to visualize them. These statement is true when we observe the objects in runtime. It is easier to understand the concept of OOP with visual representation.

---


Antecedents and objectives

In the autumn semester of academic year 2007/2008, I have been to Helsinki with the Erasmus scholarship programme. I did a research on the Finnish informatics and programming teaching methods. I visited lessons in secondary school and university, I made interviews with teachers and lecturers. I participated in teaching programming as a student. I got to know a new tool, Jeliot, and a concept of algorithm visualization (AV). In Israel and USA, this tool is used for supporting teaching and learning programming in secondary schools. I wrote a report about my experiences.  

The goal of this research to introduce an AV tool and a related investigational and teaching method which support especially the students with average skills to close up.

Applied methods

In the autumn semester of academic year 2012/2013, I performed a combined, two-group pedagogical experiment at ELTE Faculty of Informatics, where I investigate the work of 1st grader students of informatics in the Basics of programming course. 5 seminar groups took part in the experimental group and 5 seminar groups in the control group, altogether 153 students. The following table shows the number of the members involved in the experiment:

<table>
<thead>
<tr>
<th>Experimental groups</th>
<th>Control groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>18</td>
</tr>
<tr>
<td>15</td>
<td>16</td>
</tr>
<tr>
<td>16</td>
<td>14</td>
</tr>
<tr>
<td>16</td>
<td>15</td>
</tr>
<tr>
<td>15</td>
<td>12</td>
</tr>
<tr>
<td><strong>Altogether:</strong></td>
<td><strong>78</strong></td>
</tr>
</tbody>
</table>

The first part of the experiment was filling a questionnaire on previous studies by the experimental and control group (this questionnaire can be found in the appendix of the dissertation). The goal of the questionnaire was to quantify that how many lessons had the students on informatics and programming in secondary schools, and to measure the state of algorithm thinking of the students before the course would impact that.

---

During the experiment, the students got a score and a grade on the questions no. 3-5 about algorithm thinking; on the basis on this the students’ progress can be measured. Question no. 5 measured the abstraction ability. At the evaluation of the questions, I investigate the correctness and the elaboration of the answers. Elaboration is important beyond correctness, because with this the “multiplicity” of thinking can be measured: how much takes the student attention on the details of the task. The score was higher if the student gave more important precondition. Maximum score for the questions no. 3-5 was 14 points.

In order to define the progress, the evaluation should be converted to five grade, because the latter tests are in five grade system. The scoring system was the following:

<table>
<thead>
<tr>
<th>bottom point</th>
<th>grade</th>
<th>ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0%</td>
</tr>
<tr>
<td>5,5</td>
<td>2</td>
<td>39%</td>
</tr>
<tr>
<td>7,5</td>
<td>3</td>
<td>54%</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>71%</td>
</tr>
<tr>
<td>12</td>
<td>5</td>
<td>86%</td>
</tr>
<tr>
<td>14</td>
<td>(max)</td>
<td></td>
</tr>
</tbody>
</table>

After defining the grades of the input tasks, we have a starting point in order to determine the measure of progress. During the semester, the students had four tests:

- Seminar group test no. 1: coding two task which can be solved by simple programing theorem. The students get the specification and the algorithms for the tasks (goal: mostly understanding the algorithm + coding knowledge),
- Seminar group test no. 2: writing algorithms by structogram for three tasks which can be solved by simple programing theorem. The students get the specification only (goal: understanding the specification + creating algorithm, this test needs more abstraction ability than the previous test),
- submitting exercise at home , the students should solve 4 tasks, one of them can be solved by complex programming theorem, the other 3 by simple one (goal: demanding solution of a larger task, “product-construction”),
- Semester closing test, the students should solve 4 tasks, 2 of them can be solved by complex programming theorem, the other 2 by simple one (full solution of a bigger problem).
I did not take account of “submitting exercise at home” at defining the measure of progress because there is a little guarantee of independent work of the students so the authenticity could be queried. I weighted the other 3 tests according to the time, because the more further we are (in time) from the first lesson, the deeper should be the understanding. I defined the measurement of progress \((F)\) with the following formula:

\[
F = \left[ (CsZH_1 - E) + 1,5 \ast (CsZH_2 - E) + 2 \ast (EvfZH - E) \right] / 18,
\]

where \(CsZH_1\) is the grade of group seminar test no. 1, \(CsZH_2\) is the grade of group seminar test no. 2, \(EvfZH\) is the grade of semester closing test, and \(E\) is the grade of the score of the test of the prior studies (questions no. 3-5). The result of the formula above is a real number between -1 and 1. The sign shows the direction of the progress, the value shows the measure.

A results of the test of questions no. 3-5 were similar, I did not experience significant difference. That is way, it can be stated, that the two groups started from similar level so the differences of the progress values are not caused be the input knowledge.

Theses of the dissertation

Thesis no. 1: AV can be included to informatics in secondary education

The mandatory informatics framework curricula contains the “Problem solving by informatics methods and tools” thematic unit. Its goal is to develop the algorithm abilities, because the activities of real life for example shopping list, travelling build up from series of decisions.

I show application examples in my dissertation, for which topic and how can AV be used. I present a recommendation for the numbers of the lessons base on the recommendation of the Ministry of National Development.

The goal is to extend the problems at the lessons with animation in order to the pupils can use them for understanding, creating and later coding algorithms; and the teacher can the use the opportunities given by the animation, especially the continuance of the animation.
Thesis no. 2: Teaching with AV tool supports the catching up of students with nearly average knowledge

The great majority of the students (84%) got their score in interval [3,9) for the prior study questionnaire. According to the results, 78% of the experimental group and 89% of the control group are in this interval. This two diagrams show the measure of progress, the second one shows the interval [3,9):

![Figure 1. Measure of progress.](image1)

![Figure 2. Measure of progress in interval [3,9].](image2)
According to the diagrams above, a conclusion can be made: 84% of the groups which participated in the experiment, students who used AV, reached greater progress than the students in the control group. Teaching with AV system supported most the nearly average students. Based on the experiment, average student is whose score is in interval [5,6), because the average score of the participants of the experiment was 5,74. The greatest progress (0,79) was reached by the students of the experimental group with score in interval [4,5).

Thesis no. 3: Students who used AV improve more and reach better results than the students who do not use AV.

This diagram shows the comparison of the results at the end of the semester.

![Average results of the tests](image)

**Figure 3. Average results of the tests.**

I took notice of the average of the average of the tests’ grade, because this shows the progress of the students better than only term mark itself.

In interval [3,9), there was a significant difference for the experimental group. This result is relate to progress as well, so teaching with AV tool has a positive impact on students with average prior knowledge.
Figure 4. Average results of the tests in interval [3,9).

The table shows that the students, who used AV, reached better average, almost one grade better, than the students of control group.

<table>
<thead>
<tr>
<th></th>
<th>Experimental group</th>
<th>Control group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average of tests</td>
<td>3,58</td>
<td>2,65</td>
</tr>
</tbody>
</table>

Learning with AV program has an impact on the failure ratio, because significant less student failed the course in the experimental group than in the control group. The two third of the experimental group reached the 3,5 average tests grade (in the control group inly 37%).

Thesis no. 4: Use of AV improves the abstraction ability better than if this tool were not used.

The impact of AV on abstraction ability shows the results of question no. 5 and seminar group test no. 2. The score of the question no. 5 showed where was this ability, and the grade of seminar group test no. 2 showed if this ability progressed or not.

<table>
<thead>
<tr>
<th></th>
<th>Experimental group</th>
<th>Control group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average result of question no. 5</td>
<td>1,60</td>
<td>1,67</td>
</tr>
<tr>
<td>Average result of seminar group test no. 2</td>
<td>4,13</td>
<td>3,18</td>
</tr>
</tbody>
</table>

According to the table above, the experimental group starts somewhat more behind then the control group, but the former reached much better result than the control group,
which means, the use of AV tool improve more the abstraction ability than if the students have not been used it.

Publications of the author according to the topic

- Törley Gábor: *Problem based teaching method in teaching programming* (Info Savaria Conference 2006., Szombathely)
- Törley Gábor: *Algorithm visualization in programming education* (Info Savaria conference 2008., Szombathely)
- Törley Gábor: *Algorithm visualization in programming education* (Multimedia in Education Conference 2009., Debrecen)
- Törley Gábor: *Object oriented programming with Jeliot* (Info Éra Conference 2010., Füzesgyarmat)
• Törley Gábor: Teaching object oriented programming with visualization tools (InfoDidact Conference 2012, Zamárdi)