A Visual Approach for Concretizing Sorting Algorithms

Ilkka Jormanainen
Department of Computer Science
University of Joensuu
P. O. Box 111
FIN-80101 Joensuu
FINLAND
ilkka.jormanainen@cs.joensuu.fi

Abstract

Algorithm visualization is an efficient way to teach programming. The Concretization Environment Framework (CEF) combines algorithm visualization with concrete objects (e.g. Lego Mindstorms robots). CELM, Concretization Environment for Lego Mindstorms, is an application of this framework. By using the framework, the user can turn the mental model the user has, into a concrete one. User feedback on the framework and its application has confirmed the functionality of the concept.

1 Introduction

One of the main difficulties that students of computer science face is understanding algorithms. Traditionally, algorithms have been taught by verbal explanation with the use of blackboard or slides. With these it is only possible to visualize algorithms in a static way. In past decades, researchers have developed different kinds of systems for algorithm visualization. Most of these systems allows the user to interact with the visualization and the algorithms are often visualized through animation (Stasko (1990); Ben-Ari et al. (2002)).

Robot technology has become cheaper and has been adopted widely to teach programming and computer science especially to novices. Robotics has been used to motivate students to learn programming. A student can create concrete new knowledge and learn in constructionist way by interacting with real world objects (Ben-Ari, 1998). This can also lead more to hands-on learning with algorithms. In algorithm concretization, the algorithm’s execution is emulated by robotics or other real world objects. In this way, robots engage the student with the algorithm thereby fostering learning.

This paper is based on my Master’s Thesis (Jormanainen, 2004). In this paper, I will present the framework for concretizing algorithms and an implementation of the application which is based on the framework. With this application, the user is able to make concretizations for sorting algorithms more easily than it has been done in (Gonzalez, 2004), which is basis of this work. Furthermore, I will present a novel idea about role-based concretization, which can be used with the application.

2 Background: Concretizing Bubble Sort algorithm with Lego Mindstorms

In his research, Javier Gonzales developed some concretizations of sorting algorithms with Lego Mindstorms (Gonzalez, 2004). The main idea in his work was to use a master robot which controls other robots (slaves). In this scheme, every robot has an individual id and a weight. These informations are used to sort the robots with a sorting algorithm. Algorithms for the robots were developed in NQC (Not Quite C), which is a C-like programming language for Lego Mindstorms robots (Baum et al., 2000), and in Java with LeJOS. For more information about implementation, see Gonzalez (2004).

As it can be seen in Gonzalez (2004), the implementation is complicated. This makes it difficult to use concretization in an efficient way, for example, when teaching programming or algorithms to novices. It needs a long code to implement even a simple sorting algorithm, like
Bubble Sort (see Gonzalez (2004)). However, as it has been stated in Gonzalez et al. (2004), this method of teaching is promising and worth further study.

3 The Aim of the Project

The main goal of this project was to develop an application which may help the user (teacher, instructor) construct concretizations for sorting algorithms. The user defines the interesting events of the algorithm and concretizations for these events. The user can use, for example, role-based concretization.

Role-based concretization is a novel way to define concretizations with robots. The concept is based on idea that the data of a program or an algorithm has a certain role. It has been found that the following list of roles of variables covers 99% of all variables in novice-level programs: Constants, stepper, follower, most-recent holder, most-wanted holder, gatherer, one-way flag, temporary and organizer (Sajaniemi, 2002). With these roles, it is possible to define a representation for each variable in the program.

However, in this application we see the concept of role in a different way. Gonzalez (2004) has defined two roles for concretizing sorting algorithms: Left and right. In this case, the role is defined based on physical position of the robot or other object. In Figure 1, Item 2 has the role left and Item 5 has the role right. During the execution of the algorithm, the robots make pre-defined behaviours based on these roles.

Concretizations are defined by dragging robots in the application. After that, the user allows the application to upload the code to the robots. Then, the robots will execute the concretization. In this way, the mental models which users have in theirs minds, become concretized in the material world (Figure 2).

4 Design and Implementation of the Framework and the Application

The architecture of the framework contains three separate layers. For each layer, there is some output which serves as input in the next layer. Communication between layers is bi-
directional. This means that physical objects can communicate and send information about their states to the application. In this way, it is possible to track the movement of the robot. Figure 3 presents this structure and communication between layers.

![Figure 3: The relation between the framework and the application.](image)

At the moment, the whole system is implemented in Java. However, it is also possible to produce any layer of the framework with some other programming language. The most important issue is to ensure that the layers give output in the right format. Also, each layer has to have the capability to use the output of the previous level as its input. These layers can be replaced with another one. When replacing the object layer, it might be necessary also to replace the transfer layer, or part of it. However, one can replace the whole transfer layer or part of it without changing the object layer or the environment layer at all.

To achieve full support for these features, definitions for interfaces between layers must be developed further. Especially, interfaces from the object layer to the environment layer (via the transfer layer) have to be developed carefully in the future. For this project, I decided on an interface from the environment layer to the transfer layer. This interface contains codes which the application at the environment layer has to produce. The transfer layer has to have the capability to transfer this code to the native code for robots (or other objects) at the object layer.

The environment layer contains an application which is dedicated to control one or more robots. With the application, the user can construct movements and other behaviours for robots, and send them to the robot (Figure 4). The application has been implemented purely with Java, so it can be used in diverse platforms, such as Windows, Linux or Macintosh. The only requirement is the need for a LeJOS environment, which is used in the transfer layer to compile the code produced by the environment layer to the native code of the object layer.

5 Conclusion and future work

In this paper I have presented a concept of algorithm concretization, which is based on research done at the Department of Computer Science, University of Joensuu (Gonzalez et al., 2004). However, there is a need for a framework and for an application which a user (for example a teacher or other instructor) can use to produce concretizations. To answer this need, I developed a framework, which makes it possible to use diverse platforms in an easy way. In this paper, I have presented the framework and one possible application for it. Furthermore, I have described a novel concept, role-based concretization, which can be used when designing concretizations for algorithms.

To assess the potential usefulness of this approach, I conducted a questionnaire in which I asked for opinions and suggestions about the application and the framework. Six researchers in computer science education answered to the questionnaire (Jormanainen, 2004). According the answers, the application and robotics can be used to illustrate abstract concepts, such as
sorting and searching algorithms especially when teaching children or novices. Answers also indicated that the concept role can be used with this approach. However, some doubts about the approach were presented. Answers presented some very concrete suggestions for future development concern the framework and the application.

References


