Neo-Piagetian Theory as a Guide to Curriculum Analysis

Claudia Szabo and Katrina Falkner (2014)

Summary

This paper presents a framework for conducting a curriculum analysis process to map how instructional activities and assessment strategies map to the learning abstraction levels arising from neo-Piagetian theory. As part of this presentation, the authors present refinements to how computer science topics can be identified within specific levels and identifies the shift from thinking about a concept within a single programming language to its application within multiple programming languages as a key separation between the concrete operational and formal operational levels.

Familiarity

This paper draws upon neo-Piagetian theory, especially the Developmental Epistemology of Programming proposed by Lister et al. It also draws upon traditions of curriculum development and curriculum mapping. These are all familiar bodies of work for me but will likely be new to many faculty and students in computer science.

Strengths & Weaknesses

This paper presents a process for curriculum mapping – something that can fill a small textbook – and needs to also introduce the theoretical foundations and squeeze in a case study. That is a lot to accomplish in six pages of a conference paper, and for the most part it does a solid job of introducing the approach. It makes good use of diagrams and relatable examples to quickly convey core concepts. It also presents new subcategories of several neo-Piagetian learning levels specific to concerns from computer science.

For people not familiar with curriculum development or neo-Piagetian theory, following the references will be needed to fill in background information. Likewise, there is not a lot of clarity on exactly how the developmental level of learning activities and assessments was determined – the criteria were presented, but specific guidelines would be helpful for those attempting to implement this process. Likewise, the examples presented were all grounded in C++ (the language used at the school) – examples drawn from other languages would make this paper more accessible for a broader audience.

Impact

This paper largely presents a new approach to thinking about our curriculum grounded in neo-Piagetian theory and identifies two common issues in curriculum that can cause problems for student learning – prerequisite knowledge that hasn't been covered and assessment leaps where students are assessed at a different level than they have been taught. This approach has promise for improving computer science instruction, which would benefit students and help broaden participation and success in the field.

Presentation/Grammar

The paper does a good job introducing definitions alongside new terms, is well-written, and uses good diagrams. It is tailored to a computer science educator audience, so common ideas in computer science (memory management, types, pointers) are simply used without introduction, which could trip up other audiences.

Audience

This paper is really intended for folks responsible for computer science curriculum development – i.e. computer science faculty and potentially curriculum developers working with them.

Overall

This is a solid paper and example of introducing a new approach for conducting a technical process (in this case, evaluating a curriculum). It could easily be expanded into a small textbook, for which this paper would serve as an overview of that work.