An Epistemic Hypermedia to Learn Python as a Resource for an Introductory Course for Algorithmics in France

Christophe Reffay[1], Mahdi Miled[1], Pascal Ortiz[1] and Loïc Février[2]

[1] STEF (Science Education Education training), Ecole Normale Supérieure of Cachan, France
{christophe.reffay, mahdi.miled, pascal.ortiz}@ens-cachan.fr
[2] MOCAH team, LIP6, University Pierre et Marie Curie, France, loic.fevrier@france-ioi.org

Abstract. We launched an original large-scale experiment concerning informatics learning in French high schools. We are using the France-IOI platform to federate resources and share observation for research. The first step is the implementation of an adaptive hypermedia based on very fine grain epistemic modules for Python programming learning. We define the necessary traces to be built in order to study the trajectories of navigation the pupils will draw across this hypermedia. It may be browsed by pupils either as a course support, or an extra help to solve the list of exercises (mainly for algorithmics discovery). By leaving the locus of control to the learner, we want to observe the different trajectories they finally draw through our system. These trajectories may be abstracted and interpreted as strategies and then compared for their relative efficiency. Our hypothesis is that learners have different profiles and may use the appropriate strategy accordingly. This paper presents the research questions, the method and the expected results.

Keywords: Adaptive hypermedia, Navigation, Programming learning, Python, Trajectories.

1 Introduction

Despite the major role of technology and especially computers in the society, teaching informatics still remains barely present in primary and secondary French schools [1]. Nevertheless, an optional course called "ISN" (Informatics and Digital Sciences) is opening this year at the secondary school. This novelty may contribute to promote a renewal for teaching informatics in France. At the organization level, it presents a problem to have enough well-prepared teachers to take charge of such a technical topic. It seems to us that the France-IOI platform could be useful for the training of teachers as individuals, and a special sequence of exercises (currently 108) has been designed for their pupils too. During the last three years, a novel approach to build an adaptive hypermedia has been developed by the third author. It has been applied to teaching programming with Python [2]. We decided to combine France-IOI exercises and the adaptive hypermedia to support the practical part of the ISN course.

The rest of this paper is organised as follows. The next two sections present each of the main resources of this environment: France-IOI exercises and the adaptive hypermedia for Python. We then present how they are combined in the proposed experiment and what are the research questions, methods and expected results. Finally, we give a short conclusion and many perspectives of this work in progress.

2 The France-IOI Infrastructure and Sequence of Exercises

The main objective of the France-IOI non-profit association is to prepare French teenagers for International Olympiads in Informatics. For this particular public, the goal is to produce very fast thinkers and programmers mastering optimization techniques. The practice-oriented platform supports algorithmics learning and multi-language programming [3]. Its main feature consists in an automatic validation of learners' source-codes. Learners are invited to solve lots of exercises from programming basics to advanced algorithmic concepts.

Fig. 1. Exercise from the France-IOI platform: an example.

As France-IOI does for Olympiads challengers, the ISN course focuses on algorithmics more than programming. But the target level of expertise is dramatically lower for ISN pupils. For this reason, a sequence of many exercises (currently 108) including some concept presentation has been adapted for ISN pupils. This sequence of exercises is already available on the France-IOI platform [4]. One of those is illustrated in Fig. 1. Its task consists in writing an ordered sequence of instructions.

These exercises are organized in 12 topics, namely: display text and sequence, repeating instructions, variables and calculation, reading entries, test and conditions, advanced structures, advanced conditions and Boolean operators, "repeat" instruction, floats, array, strings and functions. New concepts are presented in some exercises in a

just-in-time and just-what-is-needed spirit. A global scenario tells a story where each exercise is a new challenge. These last two aspects lead designers to present these exercises in an explicitly suggested order. But pupils can access and solve exercises in any order. Each exercise is presented in a web page (as illustrated in Fig. 1) providing 5 parts (presented as modular tabs): Task, Resolution, Hints, (extra) Activity and (commented) Solution. For each learner, only the "Solution" remains unreachable until a valid code has been submitted to solve the corresponding problem. If a learner achieves only part of the exercises, leaving holes in the sequence, s/he will hardly figure out which concepts or important notion s/he missed. This is precisely where the epistemic graph proposed by P. Ortiz should help.

3 An Emerging Adaptive Hypermedia for Python Learning

This part focuses on the graph of *epistemes* and their pre-requisite dependencies. We first present a sub-graph extraction for illustration in Fig. 2. The heart of this section emphasizes four principles that guided the construction of the domain knowledge and the content of each *episteme*. We then show how these principles greatly simplified the construction of the particular subset, dedicated to support the resolution of France-IOI/ISN set of exercises. The last section situates this work in progress in the literature related to Adaptive Hypermedia applied to programming.

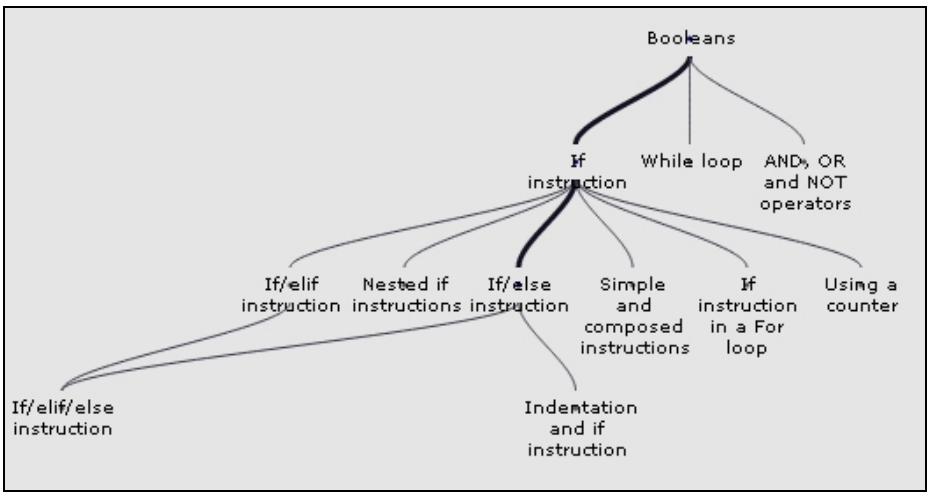

Fig. 2. Sub-graph headed from Boolean

Learning programming languages is usually assisted by documentation or textbooks which are rather linear. As presented in [5] for object-oriented concepts, initiation into programming involves many new concepts that are seen by novices as complex in the sense that it is hard to see where we have to start to be able to achieve a particular goal. For us, this is true even for basic concepts for a novice programmer.

Python appears to be a good candidate for a first programming language. We propose a hypermedia model based on a typical representation of the knowledge domain for Python by epistemic units also called "*epistemes*" [2]. These *epistemes* are linked by prerequisite relationships and form a Direct Acyclic Graph (DAG).

3.1 The Graph: Fine Grain Concepts and Prerequisite Dependencies

This part can be seen as the "Document Space" defined in [6]. The graph proposed in Fig. 2 is the result of a systematic process guided by four principles presented in the next section.

Each node (*episteme*) focuses only on a very limited concept or language construction and presents it in a *typical* program code. As an example, this is the reason why the various aspects of the "If" construction are split in: "If instruction" → "Indent an If" → "if/else instruction" → "if/elif/else instruction" (see Fig. 2).

A→B generally means: "concept A is a prerequisite for concept B". Sometimes this rule is relaxed and could be interpreted as "A is useful to build a more *typical* and comprehensive example presenting B".

For a large coverage of the language, in order to serve as an integrated continuum from introductory course to reference manual for Python, this DAG should count several hundreds of *epistemes*. For this reason we need various tools around the DAG in order to build an adaptive hypermedia. These tools may support: graph and episteme edition, graph browsing, recommendation, information retrieval (for a novice who may use inappropriate labels for concepts), etc. A first study aiming at selecting the optimal graph visualisation tool for a given task is presented in [7].

3.2 Four Principles Guiding the Graph and Episteme Edition

As mentioned above, the graph construction and episteme description are guided by the application of the following four strong principles (detailed in [2]):

1. **Salience**: Only the focused concept may be visible in an episteme;
2. **Separation**: Each concept should be presented independently to any other;
3. **Occam razor**: present only what is necessary, delete everything else;
4. "**In code we trust**": the more a program code is frequent in the code ocean (python source code available on the web), the more **typical** and **relevant** it is for us.

How little is an episteme?

– Very little!

It should present a typical code using the concept and comment it. When appropriate, it will point out typical errors. For complex concepts, more interactive media may be used (demo, slideshow etc.) in order to reduce the textual explanation.

3.3 Producing a Sub-graph on Demand

As he was building the Python programming knowledge domain (G_1), the third author extracted the sub-graph (G_2) of 76 *epistemes*, sufficient for a learner to write in Python a solution for all 108 exercises (ISN) proposed by the France-IOI platform.

The main differences between G_1 and G_2 are due to the execution context that is simplified (in G_2) by the fact that pupils write their program code directly into the web-based platform. In opposition, G_1 provides *epistemes* that specifically support learners in their local Python programming environment installation phase and others describing the different ways to produce, interpret and execute Python code. Marginal adaptation has been made also because of a shift of Python version: 2.7 to 3.1.

Except these two main differences, G_1 has been extensively reused to build G_2. Let us give some details on this systematic process:

For each exercise, provide the Python code of (at least) one acceptable solution. From each code, define the list of concepts/*epistemes* this code requires. If the concept is already presented by an episteme in the graph, just bind the exercise to it. If not, define and integrate the missing *episteme(s)* in the graph according to the 4 principles mentioned in the previous section. To build G_2.reusing G_1, only 3 new *epistemes* had to be defined and 30 were modified (repeatedly on the same facets).

3.4 Related Works in the Adaptive Hypermedia Research Field

According to the Adaptive Hypermedia (AH) review [8], methods, framework and techniques in this domain have been already studied extensively in lab conditions on theoretical concepts that shaped this research domain. Our system can be considered as a practical educational web-based AH where adaptation is mainly based on a map for navigation support like RATH [9]. However, we did not find large-scale experiments in secondary schools as proposed in this project in the literature.

Concerning programming, many ideas and adaptation techniques have been successfully implemented for example around QuizPACK (Quizzes for Parameterized Assessment of C Knowledge) [10] and QuizJET (Java Evaluation Tool) [11]. Other systems like BlueJ or Scratch [12-13] may be considered as IDE preferred by teachers for novices in order to consider interesting programming concepts as fast as possible without having to write heavy source code or deal with the syntax troubles. Our preference goes to better support the first steps (code correction) including the language syntax and typical constructions.

4 Experiment

We have presented in both of the previous parts the France-IOI platform and its proposed list of exercises for ISN on the one hand, and the graph of epistemes for Python on the other hand. We also emphasised the positive context in the introduction. Being currently in the process of finalizing software, we cannot measure the platform audience yet.

This part describes the objectives of the experiment, defines *trajectories* and presents the method through the 3 phases: data collection, analysis and expected results.

4.1 Objectives and visual adaptation of the navigation map

In this exploratory experiment, we want the navigation map to visually differentiate for an authenticated pupil: visited epistemes, validated exercises and recommended ones as a next step. We will focus only on these basic properties and adapt the navigation tool provided to a particular user according to his/her profile. For each exercise, a sub-graph representing the transitive closure of all needed epistemes will be provided in *graph* tab. The user will be able to browse this sub-graph where unvisited and non- validated epistemes are emphasized. Bigger navigation maps (i.e.: for a chapter or the entire course) may be tested in this experiment.

Our purpose is to leave the locus of control to the learners for their progression. We want to observe afterwards the different trajectories they finally draw through the system. These trajectories will be analyzed as patterns and interpreted as strategies.

4.2 Collecting Data

On the platform, each user is authenticated and may be related to an ISN group. We collect for all users the timestamps of page visits (exercise task, hints, episteme) and exercise resolution attempts (failures and success). These observed events are considered as primary traces (time-situated) [14] and will constitute the essential part of the corpus for this first experiment. If needed for disambiguation purpose, or better description of profiles, we may employ interviews or questionnaires for pupils.

4.3 Defining Trajectories

Considering primary traces, we could compose them (selection, filtering or composition techniques) to elaborate more significant traces which can also be called modelled traces [14]. In such a framework, we can define trajectories at different levels:

- The micro-level trajectory is the ordered sequence of events (visited objects, submission etc.) for a specific user between the (first) visit of the exercise description (i.e.: task definition) and the first submission of a correct solution.
- The macro-level trajectory is the ordered sequence of successfully solved exercises for a specific user in a given period and for a given set of exercises.

4.4 What is Measured? Analyzed? How are Trajectories Compared?

Taking classmates into account, individual trajectories will be compared in a class or for all classes to detect and enlighten recurrent patterns. For example, for a given exercise some students will visit epistemes (as hints) and others will not. We are interested to see if there is an impact on the resolution time or on the number of attempts.

Which are the criteria to evaluate trajectories? What is an efficient trajectory? Did the user visit the episteme recommended by the navigation tool? Did this visit positively impact his/her trajectory? Are there strong and weak learners' specific trajectories?

"Resolution duration" for a given trajectory is hard to define. We may have to distinguish a "global resolution duration" (i.e.: simple timestamps' difference between the first and the last event of the micro-trajectory) and a "connected resolution duration" considering only the (short) connected time periods within this time interval.

We plan to provide to the teachers and pupils of ISN groups an automatic representation with a global graph where all epistemes would be represented as pies reflecting the proportion of pupils (in the class) who validated it.

4.5 What is Expected?

The first expected element is about the use of different strategies to solve an exercise. Automatic analysis of primary and modelled traces will give hints about recurrent patterns in trajectories at micro and macro levels. We could expect stronger students solving exercises without using hints or visiting epistemes contents. We will identify the most visited epistemes and try to evaluate their impact on the resolution process. Resolution duration statistics should inform on the practical difficulty pupils are facing while solving exercises in an introductory programming course.

5 Conclusion

Observing practical processes of algorithmics and programming learning at a national level, we are testing a first implementation of an original combination between a set of exercises and an adaptive graph of epistemes based on strong principles. Being operational in time for the first cohorts of the ISN course, we hope this may attract many teachers and pupils, offering a large number of participants for this experiment. The many traces we hope to collect should (1) inform designers about acceptability and usability of the scenario and tools (including the DAG of epistemes) and (2) shed light on our research questions. Potential reuse of these data with data mining and smart visualisation techniques may contribute to educational data mining or learning analytics fields. But other fields of investigation may concern these issues:

- How to generalize for other programming languages? Other subjects? Other levels? Other countries?
- How to support navigation and editing processes for such an epistemic interactive hypermedia? How can we share, reuse and collectively update it?
- How to introduce clones or forks of such an epistemic interactive hypermedia or adapted parts in order to better fit particular needs or constraints?
- And more generally: How to support teachers and decision-makers to refine programming curricula from such usage experiences?

As a next step, we can investigate if opening the navigation history of pupils to their peers as shown in [15] could positively influence other pupils' trajectories.

References

1. Baron, G.-L., Bruillard, É. : L'informatique et son enseignement dans l'enseignement scolaire général français : enjeux de pouvoirs et de savoirs. In Lebeaume, J., Hasni, A., Harlé, I. (eds.), Recherches et expertises pour l'enseignement scientifique. Bruxelles : De Boeck, pp. 79–90 (2011) http://www.stef.ens-cachan.fr/annur/bruillard/2010_B&B_DeBoeck.pdf (last checked 1/31/2013)
2. Ortiz, P. : Hypermédia adaptatif à épistèmes pour l'apprentissage des langages de programmation. Poster aux Rencontres Jeunes Chercheurs en Environnements Informatiques pour l'Apprentissage Humain, Amiens, 22-23/05 (2012)
3. Hiron, M., Février, L.: A Self-paced Learning Platform to Teach Programming and Algorithms. Olympiads in Informatics, Vol. 6, pp. 69–85 (2012) http://www.mii.lt/olympiads_in_informatics/htm/INFOL105.htm (last checked 1/31/2013)
4. http://france-ioi.org, The platform of the France-IOI non-profit Association
5. Pedroni, M., Meyer, B.: Object-Oriented Modeling of Object-Oriented Concepts: A Case Study in Structuring an Educational Domain. In Hromkovic, J, Kralovic, R. & Vahrenhold, J. (eds). ISSEP 2010, January 2010, Zurich, CH. LNCS 5941, pp. 155–169 (2010)
6. Henze, N., Nejdl, W.: Logically Characterizing Adaptive Educational Hypermedia Systems, Workshop Session at WWW 2003, Adaptive Hypermedia (2003)
7. Miled, M. : Vers une mise en relation des activités d'édition et de navigation dans les ressources d'apprentissage : cas de l'apprentissage d'un langage de programmation. Rencontres Jeunes Chercheurs en EIAH, Amiens, May 22–23 (2012) http://hal.archives-ouvertes.fr/hal-00705889
8. Brusilovsky, P.: Adaptive Hypermedia. User Modeling and User Adapted Interaction 11 (1/2), 87-110. (2001) http://www.sis.pitt.edu/~peterb/papers/brusilovsky-umuai-2001.pdf (last checked 1/31/2013)
9. Hockemeyer, C., Held, T., Albert, Rath, D.: A relational adaptive tutoring hypertext WWW-environment based on knowledge space theory. Proceedings of CALISCE'98, 4th International conference on Computer Aided Learning and Instruction in Science and Engineering, Goeteborg, Sweden, pp. 417–423 (1998)
10. Brusilovsky, P. Sosnovsky, S.: Individualized Exercises for Self-Assessment of Programming Knowledge: An Evaluation of QuizPACK, ACM Journal on Educational Resources in Computing, 5 (3) (2005)
11. Hsiao, I., Brusilovsky, P., Sosnovsky, S.: Web-based Parameterized Questions for Object-Oriented Programming. In: J. Nall and R. Robson (eds.) Proceedings of World Conference on E-Learning, E-Learn 2008, Las Vegas, USA, Nov. 17–21 (2008)
12. Kölling, M., Rosenberg, J.: An object-oriented program development environment for the first programming course. SIGSE Bulletin, 28 (1), pp. 83–87 (1996)
13. Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., Resnick, M.: Scratch: A Sneak Preview. Second International Conference on Creating, Connecting, and Collaborating through Computing. Kyoto, Japan, pp. 104–109 (2004)
14. Djouad, T., Mille, A., Reffay, C., Benmohammed, M.: A new approach based on modelled traces to compute collaborative and individual indicators' human interaction. In Proceedings of the 10th IEEE International Conference on Advanced Learning Technologies, Sousse, Tunisie (2010) http://liris.cnrs.fr/Documents/Liris-4768.pdf (last checked 1/31/2013)
15. Hsiao, I-H., Brusilovsky, P.: Motivational Social Visualizations for Personalized E-learning, In: Proceedings of 7th European Conference on Technology Enhanced Education (ECTEL), ECTEL 2012, Saarbrücken, Germany, September 18–21, 2012, Springer-Verlag, Volume 7563/2012, pp. 153–165 (2012)