On a Methodology for Creating School Curricula in Computing

Krassimir MANEV, Nely Maneva

XI IOI Conference, Tehran 2017

Content

- Introduction
- Teaching Computing in secondary schools
- ACM-IEEE Computing Curricula Guidance
- Implementing the ACM-IEEE methodology
- A sample for implementing the methodology
- Conclusion

Introduction

- Two semantics of the word curriculum:

- ”externel“ - lessons and academic content taught in a school or in a specific course or program
- internal - the ***learning standards*** or ***learning objectives,*** the ***units*** and ***lessons***, the ***assign-ments*** and ***projects,*** the ***learning resources,*** the ***assessment resources***
- and even ***curriculum philosophy, curriculum packages*** (of external organization), ***ameliorating strategy*** etc.

Introduction

- ***Computing*** – the very large domain of the science and the technologies, which is dedicated to computers, their programming and usage
- ***Goal*** - to propose a methodology for development of curricula in Computing for the schools
- The idea raised during the ***IOI Workshop***, held in Bitola, Macedonia, in April 2015

Teaching Computing in secondary schools

- Participating 12 countries presented current state in teaching Computing
- Main observations:
  - most of the countries has national curriculum for teaching Information Technologies
  - countries has officially approved national curriculum for teaching Informatics
  - teaching Informatics as available only in some math schools and out of class forms (preparing for IOI or regional contests)

Teaching Computing in secondary schools

- Other observation is that the (12) countries:
  - started in different moments and so - has different history of teaching Computing;
  - has different educational strategies and different traditions;
  - has different level of preparation of the teachers to teach courses in Computing
  - has different orientations of the related industry (software vs. hardware)

Teaching Computing in secondary schools

- Initial intention of the participants was to start work on an international curriculum in Com putting for schools!
- Having in mind mentioned above differences we changed it to more promising: to try **to develop methodologies for creating Com-puting curricula** that are adequate to some educational concept
- As a base of our attempt we selected the Guidance of ACM-IEEE (Computer Society)

ACM-IEEE Computing Curricula Guidance

Principles

- identify the **fundamental knowledge** and **skills**
- provide a **methodology** for creating courses
- to be **international** in scope, **broadly based** and must include **professional practices**
- permit **ongoing review** of the corresponding curricula through the years
- to be **sensitive to progress** in technology, pedagogy and lifelong learning

ACM-IEEE Computing Curricula Guidance

Structure

- ***Body of knowledge*** of the domain describes it‘s fundamental concepts, theories and notions.
- ***Core of knowledge*** specifies the body of knowledge elements that essential for the education
- Body of knowledge is hierarchically structured in ***fields***, ***areas***, ***units*** and ***topics***.
- ***Curriculum models*** present different approaches for organizing the educational process
- ***Course descriptions*** contains detailed description of the *courses*
- *Learning objectives* (not considered here)

ACM-IEEE Computing Curricula Guidance

Hierarchy of Body of knowledge

- Body of knowledge is composed of 5 ***fields***

- *Computer Engineering*,
- *Computer Science* (CS),
- *Information Systems*,
- *Software Engineering*
- *Information Technologies* (IT).

ACM-IEEE Computing Curricula Guidance

Hierarchy of Body of knowledge

- ***Areas.*** For Example CS field is composed of the following areas:
  - Discrete Structures
  - Programming Fundamentals
  - Algorithms and Complexity
  - Architecture and Organization
  - Operating Systems
  - **Software Engineering** etc.
- ***Rem.*** It is possible to have same area in few fields – for example **Software Engineering**

ACM-IEEE Computing Curricula Guidance

Hierarchy of Body of knowledge

- ***Units***. For Example, Discrete Structures area is composed of the units:
  - Sets, relations, and functions (6);
  - Basic logic (10); Proof techniques (12);
  - Basics of counting (5); Graphs and trees (4);
  - Discrete probability (6).

Rem. All units are in the Core of knowledge; numbers in brackets – suggested class hours in corresponding unit

ACM-IEEE Computing Curricula Guidance

Hierarchy of Body of knowledge

- ***Topics***. For Example, Sets, relations, and functions unit is composed of the topics:
  - Sets (Venn diagrams; Union, intersection, complement, Cartesian product, Power set; Cardinality of finite sets).
  - Relations (Reflexivity, symmetry, transitivity; Relations of equivalence and Partial orders).
  - Functions (Surjections, injections, bijection; Inverses; Composition).

Rem. If appropriate an additional level of the hierarchy could be appended – subtopics (in brakets)

ACM-IEEE Computing Curricula Guidance

Teaching models

- This is the part of Curricula Guidance that is **closely connected to the university** education and not appropriate for our goal
- Teaching models for secondary schools have to be developed in another form
- We will give below some proposal how to develop a model

ACM-IEEE Computing Curricula Guidance

Courses creating

Figure 6-3. Coverage of core units
Objects-first introduction
Compressed approach

	CS111 O. OO Programming	CS112 O. OO Design	CS115. Discrete Structures	CS210 C. Algorithm Analysis	CS220 C. Computer Architecture	CS226 C. OS and Networking	CS262 C. Info+Knowledge Mgmt	CS292 C. Software Dev and Practice	Total	Extra hours
DS1. Functions, relations, and sets			6						6	
DS2. Basic logic			10						10	
DS3. Proof techniques			9	3					12	
DS4. Basics of counting			5						5	
DS5. Graphs and trees				4					4	
DS6. Discrete probability			6						6	
PF1. Fundamental programming constructs	7	2							9	
PF2. Algorithms and problem-solving	2	2		3					7	+1
PF3. Fundamental data structures	3	8		3					14	
PF4. Recursion	2	3							5	
PF5. Event-driven programming		2				2		2	6	+2
AL1. Basic algorithmic analysis		2		2					4	
AL2. Algorithmic strategies		2		6					8	+2

Implementing the ACM-IEEE methodology

- **Body of knowledge (for regular schools)**

- ~~*Computer Engineering*~~ (technical school)
- *Computer Science*
- ~~*Information Systems*~~
- ~~*Software Engineering*~~
- *Information Technologies*

Rem. Some topics from the eliminated fields could be appended

Implementing the ACM-IEEE methodology

- **Body of knowledge – the same elimination proces to be applied for the other levels of the hierarchy:**

- *Areas*
- *Units*
- *Topics*
- *Subtopics (if appropriate)*

Implementing the ACM-IEEE methodology

Models (for regular schools) parameters

- Levels and sub-levels: ***Low*** (1–4 grade), ***Middle*** (5-7 grade), ***First high*** (8-10 grade), ***Second high*** (11-12 degree)
- Kinds of courses: ***compulsory***, ***compulso-ry-elective*** (m out of n), ***free-elective***
- Number of classes per year, number of courses per years (or semester) and number of classes per week ***schema***

Course(s) creation – decision table

A sample for implementing the methodology

- **The model for course Informatics**
  - First and Second high level
  - One compulsory course per year
  - 36 weeks per year
  - 2 classes per hour in the first high level and 3 classes per hour in the second high level

A sample for implementing the methodology

- The decision table

Table 1. Decision table for the Curriculum

	8	9	10	11	12
DS. Discrete Structures (46 hours)					
DS1. Functions, relations, and sets (10)	10				
DS3. Proof techniques (10)		10			
DS4. Basics of counting (12)				12	
DS5. Graphs and trees (14)		4	10		
PF. Programming Fundamentals (88 hours)					
PF1. Fundamental programming constructs (14)	10	4			
PF2. Algorithms and problem-solving (14)	6	8			
PF3. Fundamental data structures (30)	10	10	10		
PF4. Recursion (12)				6	6
PF5. Event-driven programming (18)	6	6	6		

Conclusion

- Teaching Computing is fundamentally important
- It has not the place it deservs in schools
- This could not continue infinitely
- Computing community (including IOI community) could and have to search answers the questions:
  - which parts of the knowledge/skills of the domain to be taught/practiced in the schools,
  - when to teach them,
  - and how?

Conclusion

- Single journal paper could not cover all aspects of such complex activity as curricula development. For implementing the proposed here ideas:
  - broad discussion is necessary for selecting the **items of the body of knowledge**,
  - the last level of the body of knowledge hierarchy which was not considered in the sample – **the topics** – has to be included in consideration,
  - the early school years which were not considered in the sample has to be included in consideration too.

Thank you for the attention!

Any questions?