Linguistic and Philosophical Investigations
ISSN: 1841-2394, e-ISSN: 2471-0881
Vol 23 (1), 2024
pp. 1141 - 1150

Enduring Difficulties in Computer Science Education: What the Causes?

[1][2]Paolo Rocchi

[1]IBM Italia, Roma, Italy

[2]University LUISS Guido Carli, Roma, Italy

Abstract: *Objectives.* Since the 1960s, the major professional societies in computing and other entities have prepared curricula for various kinds of schools. Since the early beginnings, curricular authors and commentators blame enduring cultural difficulties about the core of computing, the areas and subareas of CS, the competencies and other broad arguments. We are convinced of the complexity of the raised issues and this paper means to search for the root-causes of the lasting drawbacks. In fact, when the causes of a problem have been found, the scientific community begins to discover how the problem is or will be eradicated.

Study Method. After a concise survey of the literature, the first stage of this inquiry disproves a false opinion about the reason of present educational difficulties. The second stage makes a comparative textual analysis of recent curricula and examines methodically the didactical strategies. The third step goes through theoretical research in CS.

Findings. The sad state of theoretical computer science emerges as the root-cause of enduring educational issues.

Conclusions. This report relates the issues met by curricular authors to the partial and rather confusing theories of computing and concludes that those authors and working teachers of CS pay the consequences of enduring cultural shortcomings from which they suffer through no fault of their own.

Keywords: Curricula of computer science, computing as a discipline, didactical triangle, theoretical computer science.

Received: 07 May 2024 **Revised:** 13 June 2024 **Accepted:** 24 June 2024

1. Introduction

In 1946 Columbia University offered one of the first academic-credit course in computing (*computing, informatics* and *computer science* are used as synonyms throughout this report), but difficulties in education arose very early and grew over time so that various cultural institutions proposed curricular guidelines in order to aid teachers around the world. The published reports are so numerous as it is difficult to give the complete account.

1.1 A Concise Survey

For the sake of simplicity, we group the reports in two including respectively comprehensive curricula (usually for undergraduate students), and curricula for special purposes or special categories of students.

(1) In 1968 the ACM (Association for Computing Machinery) following on a series of preliminary studies, delivered the first education recommendations and arranged this domain of knowledge into three areas: computer science (CS), computer engineering, and information systems. The ACM planned to update this initial report at about 10-year intervals and created Curriculum 78 that, by other things, proposed a

standard syllabus. The IEEE-CS (Computer Society of Institute of Electrical and Electronics Engineers) cooperated with the ACM in 1991 to write the third report which was influenced by the famous article of Denning and colleagues [1]. The authors of Curriculum 1991 [2] recognized that computing is simultaneously a mathematical, scientific, and engineering discipline, and set out three methodologies for the nine subject-areas that have been censured. The topics to teach were presented as a collection of knowledge-units that could be combined in various ways so that different schools could utilize the units in conformity with the educational targets. In 2001 the number of new subject-areas increased to twelve while experts from AIS (Association for Information Systems) joined to ACM and IEEE. A questionnaire which was compiled in advance of the publication of Curriculum 2001 [3], led the authors to go beyond the previous organization of knowledge and offer guidance in terms of course design. The questionnaire also urged identification of the core tenets and skills that all the students should possess. In the time scale Computing Curricula 2020 [4] is the last comprehensive work which presents various original sides. The joint task force includes fifty experts on education from twenty countries in order to go beyond a 'North American-centered' vision. It deals with undergraduate programs in computer engineering, computer science, cybersecurity, information systems, software engineering and data science. It means to drive *an evolution from traditional knowledge-based education to competency-based education*. Finally, Curricula 2020 commits to the educators' community to a more ongoing development of the guidelines.
UNESCO's International Bureau of Education has the mission of examining the curricula of various matters and their implementation worldwide. In 1994 this bureau requested IFIP (International Federation for Information Processing) to develop a study which finished with the publication of 'A Modular Curriculum in Computer Science' which was defined as a resource to build different plans of study, including courses for further or in-job training. Later on, IFIP delivered 'The Informatics Curriculum Framework' [5] which was designed to cope with a large diversity in demands for informatics education; in fact, it offered different guidelines for eight professional role categories. The IFIP TC3 task force was anew established in 2014 in response to calls for curricular updates.
Also, consortia and private firms have searched sharable educational curricula in CS such as [6]. Career Space is an association of eleven major companies dealing with computers in Europe which has outlined guides for experts preparing courses. Carrier Space [7] started with a survey about the skills profiles in Europe and arranged a curriculum with hierarchical organization, where every core module includes a set of area-specific modules.

(2) Several teams have developed guidelines for precise scopes, for example, to serve a country or an institution, to detail the lessons of a single area, to support a didactical project and so forth [8] Regional authorities, educator leaders, information technology (IT) specialists and also scholars who do not belong to IT [9], are examples of authors of this kind of curricula. In the 2000s, ACM and other major professional organizations planned to create 'specific reports' focusing, for example, on information systems [10], software engineering [11], cybersecurity, and so forth, while the overview report, belonging to group **(1),** serves as an umbrella that coordinates all the specific reports.

1.2 Everlasting Arguments

Such huge investments were decided to give adequate and effective guidance to working teachers but de facto significant difficulties, noticed in the early beginnings, remain. Meetings, working groups, questionnaires and other inquiries have pointed out some weak points. For example [12] notices both pluses and minuses f the curricular recommendations about the algorithm design strategies; [13] looks into the obstacles met by teachers in shaping curricular practices even when input regulation has been tightly prescribed; Northrup and colleagues [14] anticipates the challenges to implementing a CS curriculum, and then goes on to compare these conclusions with the challenges identified by teachers; Ni [15] analyzes the factors influencing the decisions of teachers on whether to adopt a new CS curriculum.
Besides a variety of specific and rather narrow topics, commentators raise broad educational issues, for example, the survey of Sentance and Csizmadia [16] underscores that docents of informatics are concerned about the depth and breadth of their own technical knowledge; Dörge [17] raise doubts about the base

educational targets such as 'skill', 'competency' and 'qualification' and what should be taught as 'basic information technology'; Webs and colleagues [18] delves into the conceptual problems that preliminarily opposes the design of a curriculum. The very joint task forces authoring the ACM-IEEE-AIS curricula share this kind of concern, more precisely curricula [2][3][4][19] devote one or more pages arguing on the core of informatics, the key topics to be taught, the perimeter of each area, the nature of computing, whether it is a branch of mathematics or technology, the relationship with other sciences, what competencies should be acquired, and so on.
The authors have felt the need to identify acceptable working definitions for computer science, information and other tenets of IT, but the questions remain substantially unanswered, and the resulting cultural situation can be summarized in the following way:

"Despite the deep and pervasive impact of computing and the creative efforts of individuals in academic institutions, […] *computing education today often looks much as it did several decades ago*". [20] [Italics is mine]

Normally a research paper aims directly to solve the problem under consideration; however, the educational issues mentioned above give body to an intricate web of questions:
Are computer fundamentals inherently difficult to explain?
Is informatics a branch or a part of mathematics?
What principles should be taught?
What are the areas pertaining to computer science?
Do teaching difficulties come from the rapid IT progress?

We are facing with a complex educational situation, and we believe that before seeking a solution, its precise causes should be identified. In fact, the scientific community becomes aware that a problem can be eradicated when the causes of that problem have been discovered. Hence, this paper aims to shed light on the factors resulting in the mentioned problems.

2. A Commonplace

Let us begin with a very common diagnosis. *The very writers of curricula* (e.g. [2][3][19] and [4]) *relate the enduring issues to computer science which is a new and fast advancing discipline.*
We put forward the following annotations further developed in [21]:

(a) *Nanotechnology* and *environmental science* are examples of new disciplines that rapidly grow. The former is discovering the properties of atoms and molecules for industrial purposes, the latter is resolving broad issues under the pressure of climate change. Despite this turmoil progress, small teams and single educators set up education programs, and no international organization – e. g. NWA (Nanotechnology World Association) and AESS (Association for Environmental Studies and Sciences) – has published or plans to publish comprehensive curricula for school purposes. Nanotechnology and environmental science are expanding in terms of innovative solutions and also in terms of differentiated areas of application, just like informatics, yet they do not encounter the didactical difficulties complained of in the computer sector.

(b) Significant changes characterize the history of IT, yet the major professional societies do not meet insurmountable obstacles in describing the latest technological innovations. Rarely a curriculum was criticized for lacking cutting-edge topics. Innovative technical contents did not stop the mission of didactical programs or has raised crucial pedagogical questions. Educational programs have accepted the challenging technical needs of an increasingly complex society.

Points **(a)** and **(b)** lead us to conclude that the rapid progress of technology is by no means the main cause of the everlasting arguments cited in Section 1.2. Clear evidence belies the idea that IT innovations hinder curriculum writers and in turn teaching processes.

3. Support of Science Education

CS curricula argue on the uncertain profile of this discipline, its questionable core and undecided structure. These enduring arguments suggest the idea *that teachers do not encounter didactical difficulties rather than conceptual difficulties, so we begin to draw attention to the general relations existing amongst science, education and the related theoretical base.*

Subject-centered curricula of humanities are authored by specialists and even by public administrators, sociologists, politicians, and others. The authors are ordinarily influenced by didactical doctrines, philosophical positions, political viewpoints, and past decisions too. Developing a plan of study of *English literature* or *Greek philosophy* for example, is a demanding task since the authors do not have univocal criteria to follow [22].

By contrast, science provides precise contents to educators who are little influenced by social and historical factors. Solid scientific theories directly suggest the school programs [23], for instance, the lessons of *chemistry* or *mechanics* are guided by the very logic of each matter. The plans of study do not prove so challenging because the theoretical references trace the path of educators. Even, the most rapidly evolving field does not raise great problems because of its theoretical background [24][25]. For example, electronics – a fast progressing and expanding discipline – introduces the notions of electrical current, voltage, resistance, capacitance, inductance and so forth. Next the learner meets the principal equations such as Ohm's law, Maxwell equations, Kirchhoff's laws etc. which improve student's knowledge and provides complete mastery of the most recent technologies [21].

Science education develops parallelly on the abstract and practical planes (Figure 1). For example, the courses of electronics go from elementary concepts (e. g. current, voltage etc.) to complex formal problems. At the same time, the lessons go from the simplest items (e. g. circuit, lamps, battery, etc.) toward the most complicated devices. The first path improves intellectual cognition, the second leads to professional abilities. The theoretical foundations of electronics are essential for both the pathways and eventually a student specializes in optoelectronics or microelectronics, power electronics, analog technologies or another area.

General Notions → Specific Notions
Elementary Competence → Complex Competence

Figure 1 – Parallel progress of knowledge and competencies

Speaking in general, the theoretical base of each scientific discipline drives teachers' activity. The concepts underpin the lessons' contents and for this reason leading professional entities are not occupied with curricula. Using a metaphor, science education expands like the branches of a tree growing from the roots that are the theoretical base (Figure 2).

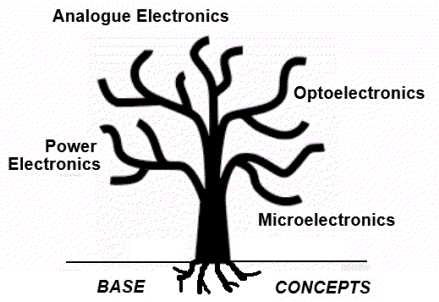


Figure 2 – Metaphorical structure of electronics

4. Evidence

As the years went by, the joint task forces tend increasingly to focus on practical skills. Curriculum 2020 openly sanctions this passage and advocates the transition from knowledge-based to competency-based education. Let us examine the viewpoints of curricular authors and especially this significant breakthrough.

4.1 Textual Analysis

We have conducted a *textual analysis* [26] of curricula 2001, 2013 and 2020 to examine carefully how the writers have changed orientation by time passing. We have selected the following groups of marker-words:

- 'Theory', 'theoretical', 'mathematics', 'mathematical' and 'principle' have abstract-theoretical meanings (printed with normal fonts in Table 1).

- 'Competency', 'skill', 'engineer', 'engineering' refer to concrete-applied topics (printed in Italics in Table 1).

Words	**Occurrences in Curriculum**		
	2001	**2013**	**2020**
theory(-ies)	76	138	25
theoretical	16	18	7
mathematics	4	46	6
mathematical	5	62	7
principle(-s)	101	134	27
competency(-ies)	0	1	734
skill(-s)	137	115	302
engineer(-s)	4	5	25
engineering	77	114	208

Table 1 – Occurrences registered in ACM-IEEE-AIS curricula.

The reader can observe how the occurrences of the first group of words dramatically drop down in the rightmost column, while the occurrences of the second group increases. These contrasting tendencies demonstrate that Curriculum 2020 is leaving out abstract discourses and focusing mainly on professional themes; the third task force relinquishes tentative philosophical considerations and suggests practical directions to follow in teaching. It recommends what students should do rather than what students should know; and this means that the theoretical foundations of computing which, in principle, should support the knowledge delivery (Section 3), are insufficient. The evolution of curricula seems to indicate a failure.

4.2 Didactical Processes

The most explicit version of the didactical triangle exhibits all the operators [27]. Aside the Learner and the Teacher, we find the Researcher who plays the role of content provider (Figure 3). In fact, teachers do not invent the scientific topics, but impart cognition and skill to the Learner (= a) on the basis of the knowledge delivered by the research (= b). The direction of arrow (= c) highlights how the students require the contents that are useful to them nowadays and in future professional activities. The didactical triangle means to say that the Teacher and the Researcher are called for serving the needs of the Learner who is placed on top, and visually shows a precise hierarchical order.

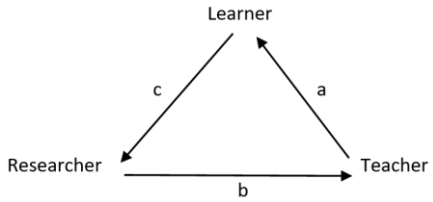


Figure 3 – The didactical triangle

Speaking in general, researchers devote themselves to practical and theoretical issues; therefore, the processes a and b split in two (Figure 4) [28][29]. The didactical triangle exhibits engineers who make educators aware of the state of the software and hardware including advanced solutions. Parallelly, theorists should provide the explanations necessary to the docents and in turn to the learners. Curriculum 2020 recommends competency-based education, namely it confines attention to the arrow who functions and, in essence, enshrines the theoretical default.

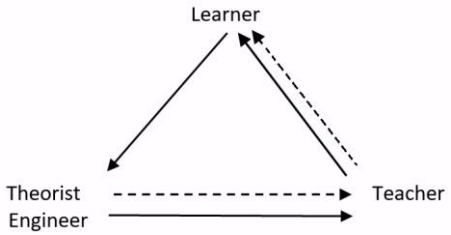


Figure 4 - The didactical triangle in CS

5. The State of the Art

The remarks and evidence of the previous sections illustrates the consistent behavior of curricular writers and lead us to go through *theoretical computer science* (TCS) which emerges as a research area whose experts have interests in many computer areas and spawn a wealth of initiatives. The numerous published articles, books, essays and the attended annual meetings demonstrate how much they are active.

5.1 Theoretical Research

TCS turns out to be as a large filed of research which we divide in two parts.

A – The scientists of this circle make *analytical and formal inquiries*, namely they are mathematicians and logicians who provide rigorous results using postulates, theorems, corollaries, logical inferences etc. For example, we quote, the ACM Special Interest Group on Algorithms and Computation Theory (SIGCAT) [30] which was founded in 1968 and promotes the formal analysis of efficient computation and computational processes. The ACM Special Interest Group on Logic and Computation (SIGLOG) [31] is an organization dedicated to the advancement of logic and computation, and formal methods in CS. The ACM Special Interest Group on Programming Languages (SIGPLAN) [32] has the intent to explore programming language concepts and tools, focusing on theory, design, implementation and practice.

What leaps to the eye is the large number of topics investigated by the group **A**. For instance, SIGCAT declares to make inquiries on algorithms, data structures, computational complexity, parallel and distributed computation, probabilistic computation, quantum computation, automata theory, information theory, cryptography, program semantics and verification, machine learning, computational biology, computational economics, computational geometry, computational number theory and algebra. SIGLOG explores a spectrum of arguments including categorical models, automated deduction, domain theory,

finite model theory, program analysis, higher-order logic, hybrid systems, lambda and combinatory calculi, linear logic, logic and automata theory, logic and computational complexity, logical aspects of database theory, logics in AI, logics of programs, logic programming, modal and temporal logics, model checking, process calculi, programming language semantics, reasoning about security, rewriting, semantics of concurrency and distributed computation, type systems and type theory, and formal verification. The IFIP Technical Committee on The Foundations of Computer Science [33] makes research about complexity, foundations of systems, cellular automata, concurrency theory, string algorithmics and other topics.

Speaking in general, theorists called upon to lay the formal foundations of a science should seek the core of that science, that is, they should deal with precise elementary concepts. The numerous themes of research demonstrate that researchers are far away the target; and tend to formalize any element of the computer system, no matter whether that element plays essential role or has trivial significance. As a consequence, theorists address problems somewhat distant from professional practice and from schooling programs too. For example, they consider 14 unsolved problems about computational complexity (e. g. NL and P problems), 11 polynomial problems (e. g. clustered planar graph problem), 17 algorithmic problems (e. g. 3 SUM solved in strongly sub-quadratic time), and 4 problems about natural language processing (e. g. Winograd schema challenge) [34].

B – The second part includes philosophers, thinkers and other scholars who mean to address broad arguments about computers and systems, their works could be defined as synthetic and informal because the authors tend not to use mathematical language. For example, the following circles of researchers belong to this group.

- In 1948 Norbert Wiener published 'Cybernetics' and inaugurated an innovative scientific movement [35]. The Cybernetic Society (UK), the American Society for Cybernetics (USA) and other associations promote the study of biological and artificial mechanisms from Wiener's perspective and also provide insights into management and sociology [36]. Various topics have been reviewed including digital computing, artificial intelligence and neurophysiology.

- In the 1950's the biologist Ludwig von Bertalanffy outlined the general system theory later furthered by writers coming from different areas who, amongst other things, created a large association called Society for General Systems Research. Eminent authors endeavored to draw a comprehensive view of systems [37] which aim to cast light on computing.

A conspicuous ensemble of studies – better known as Philosophy of Computing, Philosophy of Information, Philosophy of Artificial Intelligence and Philosophy of Programming – deals with arguments arising from within academic computer science as well as from the practice of software development [38][39]. Thinkers discuss broad themes, they are involved in the epistemology of computing, the ontology of software programming, the ethical issues arising from the use of systems and other themes. They aim to deepen understanding of computers and have holistic intent, in conformity with the arguments raised by CS curricula, yet they tend to overlook formal conceptualization.

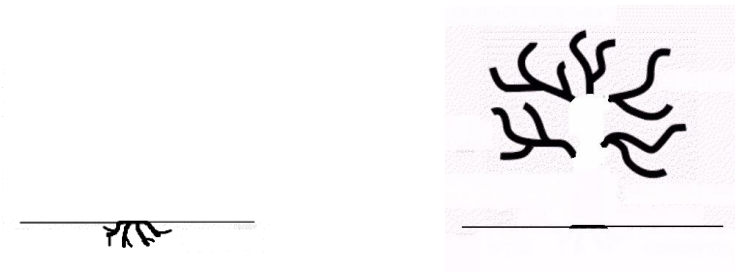


Figure 5 – The metaphor of TCS.

In conclusion, the circle **A** provides logical-mathematical results that are rigorous but narrow and abstract; **B** confronts with broad topics but argues on qualitative terms and ignores the analytical-deductive

approach typical of science. Using metaphorical images, we can say that the former prepares logical roots which do not give life to any tree (that is CS); the latter arranges branches rich in leaves which do not stem from any trunk (Figure 5).

5.2 Which Discipline?

The metaphorical conclusions help intuition; let us examine more carefully TCS.

Basically, theorems about computing refer to the Turing machine (TM) which provides details for the central unit of the computer system but overlooks the peripherals, mass storage and networking, namely TM ignores three out of four fundamental system components. Moreover, TM has unlimited tape on both sides and contradicts real systems that can only have a finite amount of memory. Finally, TM does not provide a fine account of competing processes, interactive programs and objects-programming [40]. TM perhaps was a good reference ninety years ago, but presently exhibits evident limits and we conclude with Kandel [41]:

"It is quite obvious that there is no effective theory of computer science as such. In fact, there are no effective models of computers."

Henry Schaeffer concisely describes this unscientific state: "[Computer science is] a conglomeration of many small areas that we pretend are a unified whole" and Hayes [42] adds:

"The resulting Balkanization of computing seems unwise and unhealthy, a recipe for reinventing wheels and making the same mistake three times over. Calls for unification go back at least 45 years, but the estrangement continues. (…) I find the standoff deeply frustrating".

Matti Tedre [43][44] has conducted an extensive inquiry into the scientific status of informatics debated for more than seventy years. He notices how computer experts have failed to establish the contents and purposes of this field and concludes:

"There is no consensus on what computing as a discipline is actually comprised of. Asking ten computing researchers what computing as discipline is will yield ten different answers."

If theorists had clarified the principles of computing, it would be no need for persistent discussions on didactics, and also the very investments to prepare curricula would be unnecessary; in fact, scientific disciplines (even the fastest progressing) ignore those initiatives. TCS provide constructs so distant from the expected goals that many authors doubt whether computer science can be recognized as a true discipline.

At this point the root-cause of the everlasting educational difficulties (Section 1.2) should appear evident to the reader. The writers are called upon to plan the study of a science that as such does not exist; it is merely "a conglomeration of many small areas that we pretend are a unified whole".

6. Conclusion

Curricula for teaching computer science have been proposed for decades around the world but commentators and curriculum authors themselves admit the difficulties they systematically encounter. The same stagnant intellectual situation and educational weaknesses have been registered year after year. Criticism and complaints have gone essentially unheeded and in the end Curriculum 2020 rejects lessons based on questionable theories and recommends competency-based education.

This paper conducts an articulated investigation to look for the responsibilities of this cultural defeat that has been dragging on for years; and concludes that major problems are not didactic, psychological or communicative but come from theoretical studies which should provide the precise and solid base for teaching computing. Essentially, educators pay the consequences of cultural enduring drawbacks which they suffer through no fault of their own.

We guess this inquiry will propel research toward the right direction. As for the future, we have not lost

hope and have faith in a breakthrough since there is no lack of outsider proposals [45].

References

[1] Denning P.J., Comer D.E., Gries D., Mulder M.C., Tucker A., Turner A.J., Young P.R. (1989) - Computing as a discipline - *Computer*, 22(2): 63-70.

[2] Computing Curricula 1991- Tucker A.B. (ed) *Comm. of the ACM*, Available at: https://dl.acm.org/doi/pdf/10.1145/103701.103710

[3] *Computing Curricula 2001: Final Report* - ACM Press and IEEE Computer Society Press. Available at: http://www.computer.org/education/cc2001/

[4] *Computing Curricula 2020: Paradigms for Global Computing* - ACM Press and IEEE Computer Society Press. https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2020.pdf

[5] Mulder F., van Weert T. (2000) - ICF/2000 Informatics Curriculum Framework 2000 for Higher Education - *ACM SIGCSE Bulletin*, 33(4): 75-83.

[6] Goode J., Chapman G. (2011) - *Exploring Computer Science version 4.0* - Computer Science Equity Alliance, Los Angeles. https://www.exploringcs.org

[7] Career Space (2001) - *Curriculum Development Guidelines/New ICT Curricula for the 21st Century: Designing Tomorrow's Education* - Office for Official Publications of Computing/ICT Available at: https://www.cedefop.europa.eu/files/2204_en.pdf

[8] Impagliazzo J., Campbell-Kelly M., Davies G., Lee J.A.N. (1999) - History in the computing curriculum - *IEEE Annals of the History of Computing*, 21(1): 4-16.

[9] Bruce K.B., Cupper R.D., Scot Drysdale R.L. (2010) - A history of the liberal arts computer science consortium and its model curricula - *ACM Transactions on Computing Education*, 10(1), 3:1-3:12.

[10] IS 2002: *Curriculum Guidelines for Undergraduate Degree Programs in Information Systems* - ACM Press.

[11] SE 2004: *Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering*- ACM Press.

[12] Levitin A. (2016) - Algorithm design strategies in CS curricula 2013: Hits and misses - *Journal of Computing Sciences in Colleges*, 31(3): 78-84.

[13] Drew V., Priestley M., Michael M.K. (2016) - Curriculum development through critical collaborative professional enquiry - *Journal of Professional Capital and Community*, 1(1): 92-106, Available at: https://doi.org/10.1108/JPCC-09-2015-0006

[14] Northrup A.K., Burrows A.C., Slater T.F. (2022) - Identifying implementation challenges for a new computer science curriculum in rural western regions of the United States - *Problems of Education in the 21st Century*, 80(2): 353-370.

[15] Ni L. (2009) - What makes CS teachers change? Factors influencing CS teachers' adoption of curriculum innovations - *ACM SIGCSE Bulletin*, 41(1): 544-548.

[16] Sentance S., Csizmadia A. (2017) - Computing in the curriculum: Challenges and strategies from a teacher's perspective - *Education and Information Technologies*, 22(2): 469-495.

[17] Dörge C. (2010) - Competencies and skills: Filling old skins with new wine - In Reynolds N. and Tursanyi-Szabo M. (eds.) *Key Competencies in the Knowledge Society*, Springer: 78-89.

[18] Webb M., Bell T., Davis N., Katz Y.J., Reynolds N., Chambers D.P., Sysło M.M., Fluck A., Cox M., Angeli C., Malyn-Smith J., Voogt J., Zagami J., Micheuz P., Chtouki Y., Mori N. (2017) - Computer science in the school curriculum: Issues and challenges - In: Tatnall A., Webb M. (eds) *Tomorrow's Learning: Involving Everyone. Learning with and about Technologies and Computing*, vol 515: 421-431, Springer.

[19] *Computing Curricula 2005: An Overview Report* - ACM Press and IEEE Computer Society Press. Available: http://www.acm.org/education/curric_vols/CC2005-March06Final.pdf

[20] NSF (2008) *CISE Pathways to Revitalized Undergraduate Computing Education* (CPATH).

[21] Rocchi P. (2016) - Guest Editorial: Informatics and electronics: Some educational remarks - *IEEE Transactions on Education*, 59(3): 233-239.

[22] Eisner E.W., Vallance E. (1974) - *Conflicting Conceptions of Curriculum* - McCutchan Publ.

[23] Tyler R.W. (2013) - *Basic Principles of Curriculum and Instruction* - University of Chicago Press.

[24] Corrigan D., Dillon J., Gunstone R. (2011) - *The Professional Knowledge Base of Science Teaching* - Springer Dordrecht.

[25] Kamina P., Iyer N.N. (2009) - From Concrete to Abstract: Teaching for Transfer of Learning when Using Manipulatives - *Northeastern Educational Research Association Conference Proc*.: 1-10.

[26] Burrows, J. (2004) - Textual analysis. – in *A Companion to Digital Humanities*, S. Schriebman, R. Siemens, J. Unsworth (eds), 323-347. Available at:
https://companions.digitalhumanities.org/DH/?chapter=content/9781405103213_chapter_23.html

[27] Houssaye J. (2000) - *Le Triangle Pédagogique. Théorie et Pratiques de l'Education Scolaire* - Peter Lang Publisher.

[28] Loughran J. (2000) - Teaching about Science Teaching and Learning: research should inform practice - *Asia-Pacific Forum on Science Learning and Teaching*, Volume 1, Issue 1, Article 1: 1-15.

[29] Osborne J., Dillon J. (2010) - *Good Practice in Science Teaching: What Research Has to Say* - Open University Press.

[30] SIGCAT (2023) - *Special Interest Group on Algorithms and Computation Theory* - Available at: https://www.sigact.org/

[31] SIGLOG (2023) - *Special Interest Group on Logic and Computation* - Available at: https://dl.acm.org/sig/siglog

[32] SIGPLAN (2023) - *Special Interest Group on Programming Languages* - Available at: https://www.sigplan.org/

[33] IFIP 2023 - *IFIP Technical Committee 1: Foundations of Computer Science* - Available at: https://ifip.org/tc/?tc=tc1

[34] Wikipedia (2023) - *List of Unsolved Problems in Computer Science*, Wikipedia, The Free Encyclopedia https://en.wikipedia.org/wiki/List_of_unsolved_problems_in_computer_science

[35] Rav Y. (2002) - Perspectives on the History of the Cybernetics Movement: The Path to Current Research Through the Contributions of Norbert Wiener, Warren Mcculloch, and John Von Neumann - *Cybernetics and Systems*, 33(8): 779-804.

[36] Tondl L. (2008) - Sixty years of cybernetics, the philosophical foundations of cybernetics – *Kibernetika*, 44(3): 299-306.

[37] Hammond D. (2003) - *The Science of Synthesis. Exploring the Social Implications of General Systems Theory* - University Press of Colorado.

[38] Turner R. (2018) - *Computational Artifacts: Towards a Philosophy of Computer Science* - Springer.

[39] Floridi L. (ed) (2003) - *The Blackwell Guide to the Philosophy of Computing and Information* - Blackwell.

[40] Wegner P., Goldin D. (2003) - Computation Beyond Turing Machines - *Communications of the ACM*, 46(4): 100–102.

[41] Kandel A. (1972) - Computer Science: A Vicious Circle - *Comm. of the ACM*, 15(6): 470-471.

[42] Hayes B. (2015) - Cultures of Code - *American Scientist*, 103(1): 10-13. Available at: https://www.nsf.gov/pubs/2008/nsf08516/nsf08516.pdf

[43] Tedre M. (2006) - *The Development of Computer Science: A Sociocultural Perspective* - Thesis, University of Joensuu, Finland, Available at: ftp://cs.joensuu.fi/pub/Dissertations/tedre.pdf

[44] Tedre M. (2015) - The *Science of Computing: Shaping a Discipline* - CRC Press.

[45] Rocchi P. (2013) - *Logic of Analog and Digital Machines* - Nova Science Publishers (second edition).