Research on Parallel Computing Teaching: state of the art and future directions

Thiago de Jesus Oliveira Duraes
Dept. of Computer Systems / ICMC
University of Sao Paulo
Sao Carlos, Brazil
duraes.tjo@usp.br

Paulo Sergio Lopes de Souza
Dept. of Computer Systems / ICMC
University of Sao Paulo
Sao Carlos, Brazil
pssouza@icmc.usp.br

Guilherme Martins
Dept. of Computer Systems / ICMC
University of Sao Paulo
Sao Carlos, Brazil
guilhermemartins@usp.br

Davi Jose Conte
Dept. of Computer Systems / ICMC
University of Sao Paulo
Sao Carlos, Brazil
daviconte@usp.br

Naylor Garcia Bachiega
Dept. of Computer Systems / ICMC
University of Sao Paulo
Sao Carlos, Brazil
naylor@usp.br

Sarita Mazzini Bruschi
Dept. of Computer Systems / ICMC
University of Sao Paulo
Sao Carlos, Brazil
sarita@icmc.usp.br

***Abstract*—This research full paper identifies how the teaching of parallel computing has been developing over the years. The learning of parallel and distributed computing is fundamental for computing professionals, due to the popularization of parallel architectures. Teaching parallel computing involves theoretical concepts and the development of practical skills. Its content is dense and comprises different disciplines in computer courses. Although there is growing concerned about this type of teaching, the organization and depth of parallel computing teaching at universities change widely. The available literature on the teaching of parallel computing shows some experiences about how to teach parallel computing; however, it is not easy to determine the state of the art with challenges and gaps. Our objective is to identify essential aspects related to the teaching of parallel computing as methodologies, supporting resources, subjects taught, the satisfaction of students with learning and curricula. We carried out a systematic mapping to extract information from the literature, which is composed of three phases: planning, conduction, and reporting. We initially selected 819 papers from the Scopus, IEEE, ACM, and Google Scholar databases. After a previous analysis, we performed a full read of 94 papers. The use of different teaching methodologies appears in the publications, however, the traditional teaching methodology still is the most used. There is a small number of students in parallel computing courses, a concern of different authors. Educational software or hardware resources are reported, with software proposals corresponding to most of them. The teaching of parallel computing at the beginning of undergraduate courses appear in different papers. This paper contributes to research in teaching parallel computing, pointing out the state of the art of this area, highlighting challenges that should be the focus of investigations.**

***Index Terms*—Education, Parallel Computing, Systematic Review, Educational Methods, Educational Resources**

I. Introduction

Continuous advances in parallel architectures and parallel programming models make it possible to use parallel computing in different areas of knowledge increasingly. Multi and many cores are available in affordable equipment at lower and lower costs [1], [2]. In this context, the market requires computer professionals with Parallel Computing skills and competencies to develop computational solutions that explore the hardware and parallel software available [3], [4].

Despite this need, it is clear from the available literature that the number of students interested in learning parallel computing is below expectations [1], not being compatible with market demand [4], [5]. Contributing to this scenario is the amount of content taught in parallel computing, the organization of curricula in computing courses, and the lack of adequate educational resources for the complexity of the subject, which involves theoretical knowledge and practical skills of developing parallel software for different platforms.

IEEE and ACM for a decade have pointed to the importance of adapting the curricula of different computing courses to this reality and identify parallel and distributed computing as one of the main Knowledge Areas for computing courses [6], [7].

Specialists in the teaching of parallel computing are developing research to improve this scenario on different aspects, such as teaching methodologies and techniques, computational resources, and changes in course curricula. Despite these initiatives in the area, it is not easy to determine its state of the art with future directions and open challenges when searching for papers.

Thus, this paper identifies how the teaching of parallel computing has been developing over the last years. Our objective is to identify essential aspects related to the methods and techniques to teach parallel computing, educational resources, subjects taught, curriculum, and the satisfaction of students to learn parallel computing.

We carried out a systematic mapping based on Kitchenham [8] to extract information from the literature. We gathered 819 papers from Scopus, IEEE, ACM, and Google Scholar databases. After a previous analysis, we performed a full read of 94 papers, and present conclusions regarding the state

This work has the financial support of CAPES (PROEX-11548012/M), FAPESP (2013/07375-0), CNPq, and USP.

of the art of the parallel computing teaching, showing gaps, challenges, and directions.

The remainder of this paper contains the organization below. Section II presents related work. Section III describes the research method used in our Systematic Mapping. Section IV presents our results, describing the scenery and gaps for the parallel Computing teaching. A discussion about our findings is presented in Section V. Section VI describes the threats to validity. Finally, Section VII concludes the paper.

II. Related Work

Fernández et al. [9] present a literature review related to the training of supercomputing focusing on positive effects of High-Performance Computers (HPC) on educators and researchers. The authors used the 34 best-ranked papers of 136, to answer research questions mainly related to the organization of subjects on supercomputing, adaptations in curricula, use of problem-solving training, and qualification of teachers.

Soares et al. [10] developed a systematic mapping (in Portuguese) about teaching parallel programming in undergraduate courses. The authors analyzed 83 papers to answer four research questions about teaching methodologies, learning difficulties, languages, and year or period that students study parallel programming.

Unfortunately, we could not find other reviews or surveys related to the teaching of Parallel Computing beside these two above papers. The focuses of them are different from our research since we look for broader answers about the teaching of parallel computing.

We do not restrict our search to parallel programming teaching but open the scope to parallel computing. We also do not analyze the effects of parallel computing on educators and researchers. We aim in this paper to characterize the teaching of parallel computing by describing and classifying primary studies. In this sense, we present papers approaching mainly methods and techniques to teach parallel computing, educational resources, topics taught, and changes in curricula.

III. Research Method

The research methodology used followed the systematic mapping proposed by Kitchenham [8]. Initially, the research questions were defined according to the PICO model, an acronym for **P**opulation, **I**ntervention, **C**omparison, and **O**utcomes [11]. This model, widely applied to the health area, contributes to the research question to keep the focus on the problem under investigation and facilitates its evaluation. In the PICO model the Problem component delimits the individuals or groups of interest for the study, Intervention represents the actions of interest for the research, Comparison establishes a standard to compare with the results obtained, and Outcomes defines the expected results.

For the investigation presented in this paper, the components *PICO* were defined as follows. Problem refers to the descriptions found in the literature on the teaching of parallel computing in undergraduate courses in computing. Intervention is the actions taken for teaching parallel computing that we want to analyze. The Comparison was not applied in this investigation, and Outcome is the state of the art of parallel computing teaching, considering characteristics such as teaching methods and techniques, contents, curricular structures, educational resources, evaluations employed, and the impact of the research developed.

The described PICO model guided the definitions of the search strategies and the research bases used [12]. The keywords and their synonyms were chosen because they are broader for the parallel computing teach context. The term concurrent is sometimes used analogously to the parallel one, and because of this, we also included it.

The term "Parallel Architecture" was not included in this list because it is associated with computer architecture, which does not belong to the scope of this study. Despite this, the search for teaching initiatives in parallel or concurrent computing did not exclude the teaching of parallel hardware, when it appeared in the papers found.

The search string was created based on this process:

("parallel computing" OR "concurrent computing" OR "parallel programming" OR "concurrent programming") AND ("teach" OR "educational resource" OR "ER")

The databases searched with this string were Scopus, IEEE Xplore, ACM Digital Library, and Google Scholar, [12]. Searches in the four databases surveyed were carried out in the second half of 2019 and returned 1,157 papers, 338 of which were replicated (see Table I).

TABLE I
Papers obtained by executing the search string in the databases.

Database	Results
Scopus	413
IEEE Xplore	307
ACM Digital Library	294
Google Scholar	143
Total with replications	1,157
Total without replications	819

The selection of relevant works, obtained from the databases, was based on inclusion criteria (which indicate that a paper can be included in the list of publications to evaluate) and exclusion criteria (which justify the elimination of a paper from the final list). The inclusion and exclusion criteria are presented below.

A. Inclusion Criteria

- **IC 1**: Publications related to the teaching of Parallel Computing

B. Exclusion Criteria

- **EC 1**: Publications not related to the teaching of parallel computing;
- **EC 2**: Unfinished research papers (ex.: abstracts, call for papers and short papers);

TABLE II
DATA COLLECTED FROM PUBLICATIONS FOR FURTHER ANALYSIS.

Attribute	Description
Index	Numerical index for reference
Date	Data extraction date
Author	Authors' names
Title	Title of publication
Year	Year of publication
Abstract	Summary extracted from the study itself
Keywords	Keywords extracted from the study itself
Source	Where the study was published
Database	Database where the publication was obtained
Type	Publication type (journal paper, book, conference paper, workshop paper)
Approach	What methodology and resources are used
Technology	What technologies (frameworks, APIs, languages) are used
Conclusions	What are the conclusions and impact of the publication

- **EC 3**: Grey literature, such as reports (technical, research, project and others), working papers, government documents, and white papers;
- **EC 4**: Publications with a language different from English;
- **EC 5**: Publications not available for reading or data collection. Material not reached by search engine or accessible only through payment.
- **EC 6**: Replicated publications.

The application of the inclusion and exclusion criteria considered three phases, which allowed the incremental reading of the works, selecting them recursively according to these criteria. In Phase I we selected 177 papers filtering only the titles and abstracts of them (without considering repetitions). In Phase II, we collected 134 papers reviewing the results and conclusions, and in Phase III, we selected 94 papers to perform a full reading. Three researchers analyzed the papers in these three phases, in order to validate the process and avoid bias in the paper selections. A total of 725 papers were excluded in the three phases.

The extraction of data for further analysis was supported by a form, available in Table II [8]. All data are available at http://tiny.cc/map_results.

C. Research Questions

Research Questions (RQ) guided the search that we made on the selected papers. Our primary (and general) research question is: How has the teaching of parallel computing been developing over last the years in undergraduate courses in Computer Science? We defined the following sub-questions to determine the state of the art of specific aspects as methodologies, supporting resources, curricula, subjects taught, types of evaluations, and the satisfaction of students with the learning.

- **RQ1**: What educational methodologies and techniques apply to the teaching of parallel computing? The RQ1 aims to identify the different methodologies that stand out in the literature for teaching parallel computing, such as traditional, PBL (Project or Problem Based Learning), Gamification, Collaborative Learning (by Peers or Team-Based Learning), among others.
- **RQ2**: What educational resources have been applied to the teaching of parallel computing? Educational resources for teaching parallel computing are the focus of RQ2, whose objective is to find out which resources were developed and how to use them in parallel computing courses. We do not consider as a teaching resource for this search those resources of general use, such as presentations, infrastructure, compilers, and basic development environments. Despite their importance, these primary resources are already widely used and are no longer the focus of this research.
- **RQ3**: How is Parallel Computing content addressed in the curricular structures of undergraduate computer courses? We want to identify in RQ3 at what point in undergraduate courses is parallel computing taught (Are the students without prior knowledge in computing learning parallel computing?), and if there are proposals to change the curricular structures to fit the teaching of parallel computing.
- **RQ4**: What content is taught in Parallel Computing teaching? RQ4 aims to identify, if possible, which topics are addressed in the classroom, such as parallelism theory (complexity, performance evaluation, among others), encouragement to "parallel thinking" of students, parallel algorithm projects and programming, use of different parallel architectures, or infrastructure.
- **RQ5**: Do the proposals presented in the literature consider the academic performance of students under quantitative or only qualitative aspects? RQ5 aims to show whether proposals are evaluated quantitatively, considering the level of knowledge from students or qualitatively, where students express their satisfaction and motivation about the proposal.
- **RQ6**: What is the impact of the research developed on the teaching of parallel computing? RQ6 uses as a metric the number of citations of the selected papers, to identify whether the research results contribute to the evolution of knowledge and, thus, indirectly to the training of human resources in parallel computing.

IV. PARALLEL COMPUTING TEACHING: SCENERY AND TRENDS

Figure 1 presents a time view of the publications selected between 1989 and 2019, where it is possible to see that the number of publications since 2010 has increased significantly and has remained at a higher level since then. Despite fluctuations in some years such as 2014, 2015, and 2018 (2019 was still in progress when this research was carried out), it is clear that there is an interest in the community in disseminating its research results in parallel computing education.

The publications selected in Phase III were analyzed and grouped into five categories (see Table III), all related to the research sub-questions already presented. The categories are:

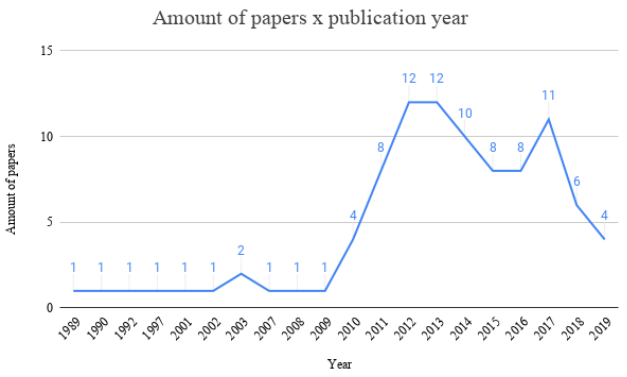

Fig. 1. Amount of works published per year

- **Teaching Methodologies and Techniques**: This category brings publications that address methodologies (such as traditional and active) and teaching techniques (such as collaborative and based on problem solving), associated courses or disciplines that address parallel computing.
- **Resources**: This category groups publications that propose or describe the use of teaching resources in parallel computing. We do not include in this category publications that address the use of consolidated resources, such as slides or the gcc compiler for C programs. The resources found were classified according to their nature or objective, such as: hardware/software, use as a graphic tool, use of programming standards, online availability or intended for automatic code correction in challenges (auto-grading).
- **Curricular Structures**: We grouped papers in this category that discuss the different aspects of parallel computing teaching in undergraduate computer courses. This grouping allows us to verify issues such as the anticipation of teaching parallel computing for students entering computer courses.
- **Teaching Focus/Contents Studied**: This category groups publications by the content covered in the teaching of parallel computing. To avoid an explosion of items here, we chose to analyze aggregating topics such as parallel programming, computer architecture, and the development of parallel thinking.
- **Evaluation of proposals**: Publications usually present some type of evaluation of their proposals. Some publications carry out quantitative assessments, considering metrics such as student learning scores. Other works, on the other hand, carry out qualitative assessments based on metrics such as the degree of satisfaction and motivation, of teachers and students.

Performing a direct analysis of the literature, it is observed that most of the publications analyzed (84%, without replications) address teaching methods and techniques in disciplines or courses in Parallel Computing [4], [13]–[15].

The use of the traditional teaching method is predominant

TABLE III
TABLE DESCRIBING CATEGORIES, MAIN FEATURES, AMOUNT OF PUBLICATIONS HAVING SUCH FEATURE, AND PERCENTAGE OF THEM IN RELATION TO TOTAL OF PAPERS SELECTED IN PHASE III.

Categories	Main Features	Amount	%
Teaching methodologies and techniques	Traditional Learning	70	74.5%
	Project/Problem Based	4	4.3%
	Gamification	4	4.3%
	Collaborative Learning	3	3.2
Educational Resources	Software	39	41.5%
	Hardware	3	3.2%
Curricular Structures	Changes/Proposal of New Grades	77	81.9%
	Beginners/Without Prior Knowledge	17	18.1%
Teaching Focus	Programming	92	97.9%
	Architecture	5	5.3%
	Parallel Thinking	5	5.3%
Proposal Evaluation	Previous Theoretical Knowledge	5	5.32%
	Objective	18	19.2%
	Subjective	76	80.9%

(74.5% of the publications found), although there is a consensus in the community about the need to improve the teaching practices of this subject in computing courses, due to the importance and growth of the use of parallel computing in the labor market [16], [17]. We found a total of 70 papers (74.5% of selected publications) that address either improvements to the traditional teaching method or alternatives to this method.

Four works fully implement the PBL (Problem or Project Based Learning) methodology in a parallel computing discipline. Bernabé et al. [18] presented its course based on matrix factoring, where students develop different versions of this problem using different libraries and parallel programming models. The course presented in Cuenca and Giménez [19] is also another example that follows the problem-based methodology, offering the concepts of parallelism and applying them immediately, with a more challenging problem being presented after the correction of the first problems.

Lupo et al. [5] use Project Based Learning and Collaborative Education, where two classes are joined: the first of them of computer graphics, focused on advanced rendering, and the other of fundamentals of parallel computing. The projects are assigned to students who work in groups to consolidate the knowledge from both disciplines. The students of the parallel computing discipline showed an improvement in learning and were also able to learn another area of computing with the formed groups. In Manogaran [15] another discipline based on PBL and Collaborative Education is presented, making a comparison with the traditional teaching methodology.

In addition to these works with PBL, another 13 publications use problem solving as a modification of the traditional teaching method. De Freitas [20] and Zakharova and Zakharov [21], for example, showed learning results based on the development of research projects by students during the course.

We found three papers with collaborative teaching, repre-

senting only 3.7% of publications in teaching methodology. Pfundt et al. [22], for example, introduced a three-module course with a collaborative teaching approach based on a project, whose final product is an intelligent camera using heterogeneous architectures. The concepts of each module contribute to the construction of the camera. In the end, the authors suggest the use of similar teaching methodologies for heterogeneous parallel architectures.

The use of gamification in the teaching of parallel computing is described in four papers. Kitchen et al. [23] showed the use of games in the classroom for teaching parallel programming. Fresno et al. [24] and Popović et al. [25] applied a similar methodology, using software tools to assist teaching. Valls-Vargas et al [26] present a tool to offer automation of the generation of the proposed challenges.

Educational software resources were presented and used in 39 analyzed papers. Only 03 publications address educational resources focused on hardware. The proposed resources have different objectives, but the proposals that support the visualization of the execution of parallel programs, such as the ThreadMentor and ConcurrentMentor tools, stand out [27], [28]. These tools assist in learning content such as thread execution and communication and synchronization primitives. Another proposal with this objective is demonstrated in Ferner et al. [29], where compiler directives are used to indicate how a sequential program could be parallelized.

The tool titled Let's HPC allows the comparison of execution times of several parallel solutions to classic computing problems, such as linear algebra and graph algorithms [30]. It is possible to configure the machines used and the problems solved, providing the user with results of metrics such as speedup and efficiency.

OnRAMP is another teaching tool that aims to lower the entry barrier for parallel and distributed computing, caused by the use of different architectures [31]. It simplifies the use of servers and clusters, made available by the teacher to the students, through an interface that encapsulates the details of the execution platform, reducing the need to configure it.

Two initiatives presented the use of software tools for autograding the codes developed by students during the teaching of parallel programming. Almeida et al. [32] show the experiences acquired with the Spanish parallel programming marathon, using the tools and problems developed for the marathon in a parallel programming course [33].

Habanero-Java library (HJlib) is an implementation of the Habanero-Java pedagogical language for teaching parallel programming, with a focus on usability and security of programming [34]. Aziz et al. [35] showed a software to classify code submitted by students automatically, through threads, and synchronization mechanisms in Java with HJlib. Such analysis aims to provide feedback to students quickly, facilitating the learning of concurrent/parallel programming in Java [36], [37].

We identified 15 papers that use code patterns for teaching parallel programming. In these cases, students apply code standards as a basis for developing assigned tasks.

Carro et al. [38] introduced communication and synchronization models between processes to present the contents independently to the programming languages. According to the authors, the results are positive regarding the occurrence of errors in the implemented codes.

Adams et al. [39] showed examples of strategies for parallel implementations and applies them to real problems, to improve students' understanding of the development of their codes. Adams [40] demonstrated a collection of simple programs for teaching parallel computing concepts, resulting in an improvement in student learning.

Following the idea of deepening the teaching of parallel architectures, two works showed the use of Raspberry Pi and Parallella boards as teaching tools at low cost and with the possibility of showing different aspects of the infrastructure necessary for this type of computing. [41], [42]. Bui et al. [43] presented three undergraduate research projects for animation rendering, photo processing, and image transcoding with support of distributed computing clusters. The main result of this paper is the development of skills in high performance and high throughput computing in students.

The analysis of the papers provided an overview of the implementation of parallel computing courses in undergraduate computing courses. Altogether, we found 77 publications that present a course proposal or their first experiences. In some of these papers, there is a low demand on the part of students and a concern on the part of educators to improve motivation for student participation [1], [18]. To this goal, some initiatives propose alternative models of curricula or changes to more traditional ones. In general, the authors state that students have improved learning with such proposals, although it is not common to disclose students' performances in papers.

The most recent studies are based on the curriculum recommendations developed by the IEEE/ACM 2013 [7], [31], [44], [45]. Previous versions of this document distributed the topics on parallelism among the other Knowledge Areas (KA) [6]. However, the 2013 version of this document describes Parallel and Distributed Computing (PD) as a new KA that requires greater teaching coverage on its topics.

Experiences with courses already consolidated can help the proposal and implementation of new strategies. Sakellariou [17] presented well the experiences of planning and teaching a parallel computing course for a decade; it is a useful reference for a new course. Brown et al. [16] show the results of months of discussions between professors and professionals in the field on the adaptation of a computing course for the insertion of parallel computing, together with strategies and justifications to achieve this objective.

There is a community concern to anticipate the teaching of parallel computing, with 17 papers recommending or applying teaching to students with little or no prior knowledge in sequential programming or other fundamentals of computing. Rague [46] and Ghafoor et al. [47] are just two of these papers that discuss such an approach for students entering computing.

The authors reported the teaching of parallel programming in 92 of the analyzed papers (97.9% of the selected) and only

05 papers described the teaching of some aspect of hardware (some publications report the teaching of both).

The four programming models with the most usage reports are OpenMP, MPI, CUDA, and JAVA, respectively with 33, 27, 19, and 9 publications. Other programming models, such as those related to PThreads, OpenCL, C++, C#, Map Reduce, and Haskell, were also described in the papers, although in a smaller amount [48]–[51].

The use of different parallel architectures, such as multi-core processors, computer clusters, many-core processors, and vector processors, affect the development of parallel solutions [52]. Such architectures impact the communication and synchronization, granularity of processes or threads, load balancing, among many other factors. Even with this reality, the architecture of parallel computers is poorly studied in parallel computing disciplines or in specific disciplines, such as Organization and Computer Architectures [52], [53].

Ernst [53] and Bunde et al. [52] highlight the little importance given to architectures in parallel computing courses and present their experiences with the study of GPU architecture to prepare students for the parallel programming learning.

Among the initiatives for teaching parallel computing, the works that develop students' parallel thinking also stand out. Teaching focused on parallel thinking is usually employed in the early stages of computer education, as students are now setting their standards for program development. The teaching of parallelism at this stage of learning allows students to naturally identify the parallel aspects of the problems they will solve, allowing for the development of optimized codes focused on high-performance computing [46].

Some proposals distribute the concepts of parallel programming through the computer science course, with the most basic concepts of parallelism being presented right at the beginning of the course, as an extension of sequential programming, thus allowing students to develop parallel thinking more smoothly in their algorithms [54].

Some works seek to understand students' prior theoretical knowledge about parallel computing. Libert and Vanhoof [55] and Feldhausen et al. [1] analyzed previous knowledge about parallelism in secondary/high school students, generating useful knowledge for the implementation of parallel computing disciplines for undergraduate students. Similar research was carried out by Rague [46] and Lammers and Brown [56], redirecting the focus to undergraduate students.

Raj and Jha [57] looked to understand the influence of student performance in introductory, theoretical, and computer system-related disciplines with performance in parallel programming disciplines. This research has shown that these disciplines have an impact on future performance in parallel programming, indicating the possible areas where the concepts of parallel computing are most affected.

As a way of verifying the effectiveness of the presented proposals, 18 papers considering the students' performance as a metric to evaluate the objective of their experiences. Burtscher et al. [44] and Shamsi et al. [58] are examples of works that use this method to evaluate the disciplines described. However, the most common way to verify the proposals existent in the selected papers was through subjective evaluations, with 76 publications considering this type of evaluation. Subjective evaluations are usually carried out by applying questionnaires to students, obtaining answers on motivation, perception of learning, assessment of the use of tools, and on the methods used [15], [18], [19].

V. Discussion

We believe that the increased interest of the scientific community in the teaching of parallel computing (see Figure 1), is due to the need for professionals with skills and competencies to develop solutions for the current multi and manycore heterogeneous processors. This demand for professionals is growing since the development of parallel solutions is no longer restricted to research centers, being requested in different commercial, industrial and financial areas, given the large volume of data handled in these sectors of the economy.

The literature in the area shows that the scientific community is active about teaching practices in Parallel Computing. This proactive stance of the community is in line with the recommendations of the IEEE/ACM [7].

The literature reports classes inspired by the use of problems or projects for teaching parallel programming. However, few works implement PBL.

Like PBL, a few works also implement Collaborative Learning (CL) or Gamification in teaching. Lupo et al. [5] show that CL is efficient by combining classes with different objectives, in addition to also applying PBL. Such methodologies (PBL, TBL, CL, and Gamification) have been presented in a small number of studies, but with positive experiences, which suggests that there is room for growth for their use with the possibility of success.

RQ1: What educational methodologies and techniques apply to the teaching of parallel computing?

The analyzed papers present different methodologies and techniques for parallel computing classes. However, the possible teaching methodologies and techniques such as flipped classroom, PBL or TBL, are not yet appropriately applied (with depth and breadth). Today, it is still common to use the traditional teaching method with some adaptations. There is a gap here between the theory and the practice for teaching methodologies and techniques in parallel computing.

RQ2: What educational resources have been applied to the teaching of parallel computing?

The analyzed literature shows that there are different educational resources proposed and being used. Among the software tools presented, we highlighted those that indicate the regions of the sequential code with the most significant potential for parallelization, helping students in the development of the parallel solution.

We also highlighted the tools that allow visualizing the functioning of parallel programs and performance metrics on

the parallel execution of these programs, improving students' understanding of the concepts of parallelism.

There is a tendency to make resources available online, both for students in the classroom and remote courses. Following this trend, tools for automatic correction of parallel programs are presented, as well as software for programming marathons, with two successful experiences.

The analyzed papers present a reasonable variety of educational resources such as ThreadMentor, Habanero, and On-Ramp, showing an interest of the community in reducing the complexity of teaching parallel computing through automation or computational support to the teaching of the area.

Despite the diversity of resources shown in the selected literature, their use is generally limited to the groups that developed them. It is also common to offer these resources, followed by their application, still in the same paper or in subsequent works by the same authors. There is a lack of impact on the resources produced in the area.

RQ3: How is Parallel Computing content addressed in the curricular structures of computer courses?

The publications present several computing courses that include the teaching of parallel computing, both in new disciplines and in reformulations of already established disciplines.

Publications in the area show that there is a concern with students' motivation to study parallel computing. Some initiatives aim to increase interest in improving the knowledge of future professionals.

Following this line, the recommendation to distribute the concepts of parallel computing throughout the course is made and implemented in some papers, with works recommending the teaching of these concepts in the first years of undergraduate courses. This seems to be a strong trend in the area.

RQ4: What content is taught in Parallel Computing?

The selected papers showed several initiatives related to teaching programming. Despite the emphasis on programming, even due to the search, initiatives were observed to use a simpler and didactic hardware infrastructure for students.

Regarding the programming models adopted, there is a predominance of the use of OpenMP, MPI, and CUDA in the courses presented, with the appearance of alternatives based on Java among others.

In order to facilitate students' future learning in the disciplines that deal with parallel programming, some initiatives introduce concepts of parallelism in courses in architecture and computer organization.

Another concern with the teaching of parallel computing is the development of parallel thinking in students, intending to improve understanding and design parallel solutions. This practice shows that it is a tendency in the area and, by focusing on conceptual aspects, has the potential to produce substantial positive impacts on the training of future professionals who will develop solutions with parallel computing.

The use of examples and programming patterns proved to be a good alternative for teaching basic concepts of parallel programming. This approach offers a good abstraction to acquire the concepts easily. The papers that demonstrated the use or new resources of this approach are recent, opening space for original contributions.

RQ5: Do the proposals presented in the literature consider the academic performance of students under quantitative or only qualitative aspects?

We noticed that practically all the works carry out some type of evaluation of its proposal, as expected. Most of these works, however, make this assessment subjectively, considering motivation and degree of satisfaction of students and teachers about learning and teaching. Other works, in smaller quantities, evaluate their results more quantitatively and objectively, analyzing the students' performance and comparing these results with other editions of the course, for example. We understand that objective and subjective assessments are complementary and should coexist. The lack of these evaluations makes it difficult for third parties to use the proposals in the future.

RQ6: What is the impact of the research developed on the teaching of parallel computing?

It is possible to estimate the impact and relevance of the 94 papers by the citations to the selected publications. The select papers have a total of 548 citations until the development of this research, in the second half of 2019, with an average of 5.8 citations per paper. Some of these works stand out for receiving a higher number of citations, such as [34], which presents a parallel programming framework in Java and currently has 63 citations.

The citations also show a community interest in better organizing the teaching of parallel computing, observing the citations to the works of Brown et al. [16] and Torbert et al. [59]. These works provide data and proposals for the implementation of courses and there are, respectively, 23 and 30 citations for these papers, values well above the general average of citations (Figure 2).

VI. Threats to Validity

We researched by papers following the protocol of a systematic mapping [8]. However, some threats to the validity of this process still can be present.

We can highlight three possible threats to this work. In the first one, researchers can insert a bias when they analyze the publications. For the second, access to some scientific databases can be restricted, and such fact can prevent us from getting some papers. The third threat is a possible incorrect classification of the selected papers. We tried to minimize these threats by taking the following actions.

We reduced the risk of the researcher´s bias (first threat), by defining inclusion and exclusion criteria in the protocol. Four researchers involved in the study set these criteria to verify the application of them. We mitigate the impact of limited accesses

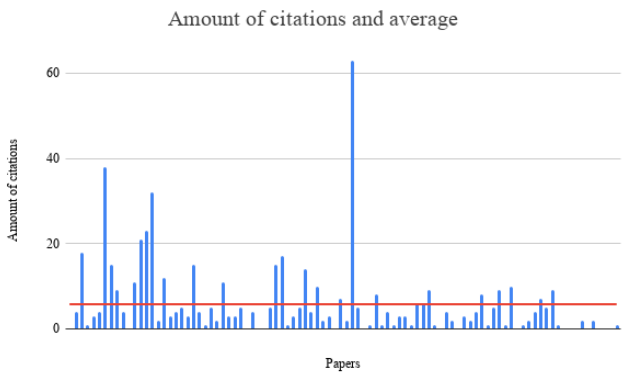

Fig. 2. Amount of citations per paper and average of citations (red line).

to databases (second threat), by asking for the full publication to the university physical library and, if necessary, to the paper´s authors. The possibility of incorrect classification of the selected papers (third threat), was reduced in two ways. First, we used a form to organize the data extracted from publications. Second, the classification of the papers was made by four different researchers.

VII. Concluding Remarks

We analyzed the publications related to the teaching of parallel computing in undergraduate courses of computing from 1989 to 2019. Our search string gathered 819 distinct papers and we selected 94 of them, considering a systematic mapping with inclusion and exclusion criteria.

There is a growing number of publications in the last ten years in this area (from 2010), which contribute positively to the learning of future professionals in computing. The analyzed papers have a still low number of citations, with some standing out concerning teaching strategies and frameworks that facilitate the dynamics of teaching and learning.

Teaching methodologies appear in 79 papers of 94, but the traditional methodology still is the most used: 70 papers cited its use. Options to insert parallel computing teaching in curricular structures appear in 82% of the papers, which adhere to the IEEE/ACM 2013 Curricula [7]. There is a small number of students in parallel computing courses, a concern of different authors. The parallel computing contents focus mostly on parallel software development (98%), with some proposals of stimulating parallel thinking (5%). Around 80% of the papers presenting proposals with subjective validations, taking into account aspects as perception and motivation.

Educational resources are also reported, being the software-resource responsible for 94% of them. The main examples of software-resources related in the selected papers are tools for visualization/functioning of parallelism, online tools, and programming marathons.

Despite the several studies regarding the teaching of parallel computing, our results show that is necessary more research in this area. Among the methods and techniques presented, for example, are imperative new studies about theoretical and practical classes with problem/project based learning, collaborative teaching (TBL and by pairs), gamification, flipped classrooms, and others.

Although there are different proposals for computational resources, they are cited basically by the groups that developed them. We did not also find initiatives as intelligent tutors in parallel computing. Many proposals emphasize online courses, which should receive more attention in the future.

Computer Science and Computer Engineering Curricula are tending to spread and anticipate the parallel computing syllabus, allowing them to expose students to parallel thinking already at the beginning of their studies. We did not find the description of the teaching of topics as performance evaluation, testing of parallel programs, and security in the context of parallel computing.

The assessments described in the papers, contrary to expectations, do not present evaluations made by students about technical content learned with the course. The development of tools to automate continuous assessments with quick feedback could guide the student´s learning more effectively.

References

[1] R. Feldhausen, S. Bell, and D. Andresen, "Minimum time, maximum effect: Introducing parallel computing in cs0 and stem outreach activities using scratch," in *ACM Int. Conf. Proc. Ser.* ACM, 2014.

[2] R. Brown and E. Shoop, "Csinparallel and synergy for rapid incremental addition of pdc into cs curricula," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. Workshops, IPDPSW*, 2012, pp. 1329–1334.

[3] C. Ferner, B. Wilkinson, and B. Heath, "Using patterns to teach parallel computing," in *Proceedings of the Int. Parallel and Distributed Processing Symp., IPDPS*. IEEE Comp. Society, 2014, pp. 1106–1113.

[4] L. Ivanov, "The right balance: Restructuring the parallel and scientific computing course," *JCSC*, vol. 27, no. 3, p. 115–121, Jan. 2012.

[5] C. Lupo, Z. Wood, and C. Victorino, "Cross teaching parallelism and ray tracing: A project-based approach to teaching applied parallel computing," in *SIGCSE 2012*, 2012, pp. 523–528.

[6] L. Cassel, A. Clements, G. Davies, M. Guzdial, R. McCauley, A. McGettrick, B. Sloan, L. Snyder, P. Tymann, and B. W. Weide, "Computer science curriculum 2008: An interim revision of cs 2001," New York, NY, USA, Tech. Rep., 2008.

[7] A. f. C. M. A. Joint Task Force on Computing Curricula and I. C. Society, *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. New York, NY, USA: Association for Computing Machinery, 2013.

[8] B. Kitchenham, "Procedures for performing systematic reviews," *Keele, UK, Keele University*, vol. 33, no. 2004, pp. 1–26, 2004.

[9] Álvaro Fernández, C. Fernández, J. Ángel Miguel-Dávila, M. Ángel Conde, and V. Matellán, "Supercomputers to improve the performance in higher education: A review of the literature," *Computers & Education*, vol. 128, pp. 353 – 364, 2019.

[10] F. A. Lara Soares, C. Neri Nobre, and H. Cota de Freitas, "Parallel programming in computing undergraduate courses: a systematic mapping of the literature," vol. 17, no. 08, pp. 1371–1381, 2019.

[11] C. M. d. C. Santos, C. A. d. M. Pimenta, and M. R. C. Nobre, "The pico strategy for the research question construction and evidence search," *Rev. latino-americana de enfermagem*, vol. 15, no. 3, pp. 508–511, 2007.

[12] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the systematic literature review process within the software engineering domain," *Journal of systems and software*, vol. 80, no. 4, pp. 571–583, 2007.

[13] J. Chen, L. Shen, J. Yin, and C. Zhang, "Parallel programming course development based on parallel computational thinking," in *ACM Int. Conf. Proc. Ser.* ACM, 2018, pp. 103–109.

[14] P. Strazdins, "Experiences in teaching a specialty multicore computing course," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. Workshops, IPDPSW*, 2012, pp. 1283–1288.

[15] E. Manogaran, "Act-pbl: An adaptive approach to teach multi-core computing in university education," in *Proceedings - 2013 IEEE 5th Int. Conf. on Tech. for Education, T4E 2013*. IEEE Computer Society, 2013, pp. 19–23.

[16] R. Brown, E. Shoop, J. Adams, C. Clifton, M. Gardner, M. Haupt, and P. Hinsbeeck, "Strategies for preparing computer science students for the multicore world," in *Proceedings of the Conf. on Integrating Technology into Computer Science Education, ITiCSE*, 2010, pp. 97–115.

[17] R. Sakellariou, "Experiences with teaching a second year distributed computing course," *LNCS*, vol. 10104, pp. 28–37, 2017.

[18] G. Bernabé, J. Cuenca, L.-P. García, D. Giménez, and S. Rivas-Gomez, "A high performance computing course guided by the lu factorization," in *Procedia Computer Science*, vol. 29. Elsevier, 2014, pp. 1446–1457.

[19] J. Cuenca and D. Giménez, "A parallel programming course based on an execution time-energy consumption optimization problem," in *Proceedings - 2016 IEEE 30th International Parallel and Distributed Processing Symposium, IPDPS 2016*. IEEE, 2016, pp. 996–1003.

[20] H. De Freitas, "Method for teaching parallelism on heterogeneous many-core processors using research projects," in *Proceedings - Frontiers in Education Conference, FIE*, 2013, pp. 108–113.

[21] I. Zakharova and A. Zakharov, "Key issues of low-level parallel programming in the individual projects for graduate students," *International Journal of Engineering Education*, vol. 34, no. 4, pp. 1250–1260, 2018.

[22] B. Pfundt, M. Reichenbach, and D. Fey, "Comprehensive curriculum for reconfigurable heterogeneous computer architecture education," *IET Circuits, Devices and Systems*, vol. 11, no. 4, pp. 292–298, 2017.

[23] A. Kitchen, N. Schaller, and P. Tymann, "Game playing as a technique for teaching parallel computing concepts," in *ACM SIGCSE Bulletin*, vol. 24, no. 3, 1992, pp. 35–38.

[24] J. Fresno, A. Ortega-Arranz, H. Ortega-Arranz, A. Gonzalez-Escribano, and D. Llanos, *Applying gamification in a parallel programming course*. IGI Global, 2016.

[25] M. Popović, K. Vladimir, and M. Šilić, "Application of social game context to teaching mutual exclusion," *Automatika*, vol. 59, no. 2, pp. 208–219, 2018.

[26] J. Valls-Vargas, J. Zhu, and S. Ontanon, "Graph grammar-based controllable generation of puzzles for a learning game about parallel programming," in *ACM Int. Conf. Proc. Ser.*, H. C. Z. J. D. S. Canossa A., Sicart M., Ed., vol. Part F130151. ACM, 2017.

[27] S. Carr, J. Mayo, and C.-K. Shene, "Threadmentor: A pedagogical tool for multithreaded programming," vol. 3, no. 1, p. 1, 2003.

[28] S. Carr, C. Fang, T. Jozwowski, J. Mayo, and C.-K. Shene, "Concurrentmentor: A visualization system for distributed programming education," in *Proc. Int. Conf. Parallel Distrib. Proces. Tech. Applic.*, A. H. M. Y. Arabnia H.R., Mun Y., Ed., vol. 4, 2003, pp. 1676–1682.

[29] C. Ferner, B. Wilkinson, and B. Heath, "Toward using higher-level abstractions to teach parallel computing," in *Proc. - IEEE Int. Parallel Distrib. Process. Symp. Workshops PhD Forum, IPDPSW 2013*. IEEE Computer Society, 2013, pp. 1291–1296.

[30] B. Chaudhury, A. Varma, Y. Keswani, Y. Bhatnagar, and S. Parikh, "Let's hpc: A web-based platform to aid parallel, distributed and high performance computing education," *JPDC*, vol. 118, pp. 213–232, 2018.

[31] S. Foley, D. Koepke, J. Ragatz, C. Brehm, J. Regina, and J. Hursey, "Onramp: A web-portal for teaching parallel and distributed computing," *J. of Parallel and Distributed Computing*, vol. 105, pp. 138–149, 2017.

[32] F. Almeida, J. Cuenca, R. Fernández-Pascual, D. Giménez, and J. Benito, "The spanish parallel programming contests and its use as an educational resource," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. Workshops, IPDPSW*, 2012, pp. 1303–1306.

[33] D. Giménez, "A practical parallel programming course based on problems of the spanish parallel programming contest," in *Procedia Computer Science*, C. M., Ed., vol. 80. Elsevier B.V., 2016, pp. 1978–1988.

[34] S. Imam and V. Sarkar, "Habanero-java library: A java 8 framework for multicore programming," in *ACM Int. Conf. Proc. Ser.*, vol. 13-December-2014. ACM, 2014, pp. 75–86.

[35] M. Aziz, H. Chi, A. Tibrewal, M. Grossman, and V. Sarkar, "Autograding for parallel programs," in *Proc. EduHPC: Workshop Educ. High-Perform. Comput. - Held conjunction SC: Int. Conf. High Perform. Comput., Network., Storage Anal.* ACM, Inc, 2015.

[36] M. Grossman, M. Aziz, H. Chi, A. Tibrewal, S. Imam, and V. Sarkar, "Pedagogy and tools for teaching parallel computing at the sophomore undergraduate level," *JPDC*, vol. 105, pp. 18–30, 2017.

[37] V. Sarkar, M. Grossman, Z. Budimlic, and S. Imam, "Preparing an online java parallel computing course," in *Proc. - IEEE Int. Parallel Distributed Process. Symp. Workshops, IPDPSW*. IEEE Inc., 2017, pp. 360–366.

[38] M. Carro, A. Herranz, and J. Mariño, "A model-driven approach to teaching concurrency," *ACM Transactions on Computing Education*, vol. 13, no. 1, 2013.

[39] J. Adams, R. Brown, and E. Shoop, "Patterns and exemplars: Compelling strategies for teaching parallel and distributed computing to cs undergraduates," in *Proc. - IEEE Int. Parallel Distrib. Process. Symp. Workshops PhD Forum*. IEEE Computer Society, 2013, pp. 1244–1251.

[40] J. Adams, "Patternlets — a teaching tool for introducing students to parallel design patterns," *JPDC*, vol. 105, pp. 31–41, 2017.

[41] B. Levandowski, D. Perouli, and D. Brylow, "Using embedded xinu and the raspberry pi 3 to teach parallel computing in assembly programming," in *Proc. - IEEE Int. Parallel Distrib. Process. Symp. Workshops, IPDPSW*. IEEE Inc., 2019, pp. 334–341.

[42] S. J. Matthews, "Teaching with parallella: A first look in an undergraduate parallel computing course," *J. Comput. Sci. Coll.*, vol. 31, no. 3, p. 18–27, Jan. 2016.

[43] P. Bui, T. Boettcher, N. Jaeger, and J. Westphal, "Using clusters in undergraduate research: Distributed animation rendering, photo processing, and image transcoding," in *Proc. IEEE Int. Conf. Cluster Comput.*, 2013.

[44] M. Burtscher, W. Peng, A. Qasem, H. Shi, D. Tamir, and H. Thiry, "A module-based approach to adopting the 2013 acm curricular recommendations on parallel computing," in *SIGCSE*, E. K. T. J. Alphonce C., Decker A., Ed. ACM, Inc, 2015, pp. 36–41.

[45] T. Newhall, A. Danner, and K. Webb, "Pervasive parallel and distributed computing in a liberal arts college curriculum," *JPDC*, vol. 105, pp. 53–62, 2017.

[46] B. Rague, "Measuring cs1 perceptions of parallelism," in *Proceedings - Frontiers in Education Conference, FIE*, 2011.

[47] S. Ghafoor, D. Brown, and M. Rogers, "Integrating parallel computing in introductory programming classes: An experience and lessons learned," *LNCS*, vol. 10659, pp. 216–226, 2018.

[48] K. Barton, S. Dodson, D. Fenske, B. Whitehead, and J. Mache, "Haskell as an introduction to parallel computing for undergraduates," in *Proceedings of the International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)*, 2013, pp. 1–4.

[49] A. Radenski, "Integrating data-intensive cloud computing with multicores and clusters in an hpc course," in *Annu. Conf. Innov. Technol. Comput. Sci. Educ. ITiCSE*, 2012, pp. 69–74.

[50] C. Sadowski, T. Ball, J. Bishop, S. Burckhardt, G. Gopalakrishnan, and J. Mayo, "Practical parallel and concurrent programming," in *SIGCSE*, 2011, pp. 189–194.

[51] Y. Ko, B. Burgstaller, and B. Scholz, "Parallel from the beginning: The case for multicore programming in the computer science undergraduate curriculum," in *SIGCSE 2013*, 2013, pp. 415–420.

[52] D. Bunde, K. Karavanic, J. MacHe, and C. Mitchell, "Adding gpu computing to computer organization courses," in *Proc. - IEEE Int. Parallel Distrib. Process. Symp. Workshops PhD Forum, IPDPSW*. IEEE Computer Society, 2013, pp. 1275–1282.

[53] D. Ernst, "Preparing students for future architectures with an exploration of multi- and many-core performance," in *ITiCSE - Proc. Annu. Conf. Innov. Technol. Comput. Sci.*, 2011, pp. 57–61.

[54] M. Arroyo, "Teaching parallel and distributed computing to undergraduate computer science students," in *Proceedings - IEEE 27th International Parallel and Distributed Processing Symposium Workshops and PhD Forum, IPDPSW 2013*. IEEE Computer Society, 2013, pp. 1297–1303.

[55] C. Libert and W. Vanhoof, "Analysis of students' preconceptions of concurrency," in *Proc. of the 1st ACM SIGSOFT Int. Workshop on Education through Advanced Software Engineering and Artificial Intelligence*, ser. EASEAI 2019. New York, NY, USA: ACM, 2019, p. 9–12.

[56] G. Lammers and C. Brown, "Exploring student understanding of parallelism using concept maps," in *Proceedings - Frontiers in Education Conference, FIE*, 2012.

[57] S. Raj and S. Kumar Jha, "Predicting success in undergraduate parallel programming via probabilistic causality analysis," in *Proc.-IEEE Int. Parallel Distrib. Process. Symp. Workshops*. IEEE, 2018, pp. 347–352.

[58] J. Shamsi, N. Durrani, and N. Kafi, "Novelties in teaching high performance computing," in *Proc. - IEEE Int. Parallel Distributed Process. Symp. Workshops, IPDPSW*. IEEE Inc., 2015, pp. 772–778.

[59] S. Torbert, U. Vishkin, R. Tzur, and D. Ellison, "Is teaching parallel algorithmic thinking to high school students possible? one teacher's experience," in *SIGCSE*, 2010, pp. 290–294.