The Journal of Supercomputing (2021) 77:3899–3913
https://doi.org/10.1007/s11227-020-03416-6

Facilitating the learning process in parallel computing by using instant messaging

Ángel Manuel Guerrero-Higueras[1] · Lídia Sánchez-González[1] · Miguel Ángel Conde-González[1] · Manuel Castejón-Limas[1]

Published online: 9 September 2020

Abstract

Parallel Programming skills may require a long time to acquire. "Think in parallel" is a skill that requires time, effort, and experience. In this work, we propose to facilitate the students' learning process in parallel programming by using instant messaging. Our aim was to find out whether students' interaction through instant messaging tools is beneficial for the learning process. In order to do so, we asked several students of an HPC course of the Master's degree in Computer Science of the University of León to develop a specific parallel application, each of them using a different application program interface: OpenMP, MPI, CUDA, or OpenCL. Even though the used APIs are different, there are common points in the design process. We encouraged students to interact with each other by using Gitter, an instant messaging tool for GitHub users. Our analysis of the communications and results demonstrate that the direct interaction of students through the Gitter tool has a positive impact on the learning process.

Keywords High-performance computing · Instant messaging · Parallel programming

✉ Ángel Manuel Guerrero-Higueras
am.guerrero@unileon.es

Lídia Sánchez-González
lidia.sanchez@unileon.es

Miguel Ángel Conde-González
mcong@unileon.es

Manuel Castejón-Limas
manuel.castejon@unileon.es

[1] Department of Mechanical, Computer Science, and Aerospace Engineering, University of León, Campus de Vegazana S/N, 24071 León, Spain

1 Introduction

In today's digitally driven world, advanced technology has become an essential tool for sciences creating every day even more demands to be satisfied. In order to deal with many problems of areas such as Meteorology, Climatology, Astrophysics, Healthcare, Biology, or Artificial Intelligence, High-Performance Computing (HPC) is required due to the data volume they manage or the execution time they need. For this reason, the use of a supercomputer that provides such extreme computational power is mandatory.

Governments and organizations are investing in HPC to leverage and strengthen infrastructures that provide HPC services. Besides the hardware, the use of all the capabilities available is a key task for professionals in technology; this is commonly named as the "think in parallel" skill. Therefore, this competence must be acquired by future professionals and it is an essential issue to be focused on.

There is an increasing demand for professionals to be aware of the benefits of HPC so as to handle these problems. Therefore, teaching future professionals implies acquiring skills related to HPC such as parallel programming or managing HPC environments. One of the key skills to acquire for the effective use of HPC facilities is "Think in parallel".

However, the acquisition's evaluation of the "think in parallel" skill is not easy since it requires to obtain enough evidence to determine performance metrics. The simplest way to be considered might be the evaluation of the final result or the hand-ins. Nevertheless, a complete evaluation should include all the aspects involved during the learning process. For example, Team Work Competence (TWC), a skill also necessary to succeed during the professional career. Traditionally, this competence has been performed by giving a task to a group of students who usually split it into as many pieces as group members and finally they collect the partial solutions to build a final result. This presents several drawbacks since it is common that students only learn their tasks and lastly they end working alone. On the contrary, when they start their careers, they have to collaborate with their colleagues, cooperating with them to solve problems and achieving the established goals.

This paper evaluates whether teamwork has a positive influence on the learning process when students using parallel computing skills. The considered degree to be analyzed is the Master in Computer Science of the University of León, Spain. The students of the High-Performance Computing subject have been encouraged to work cooperatively during the development of their practical assignments, interacting with an Instant Messaging (IM) tool.

IM applications are currently available in all kinds of devices, in particular in smartphones. Such apps provide a way to communicate synchronously among users by exchanging text, chatting in a group, sharing media, and talking.

The most popular applications for IM used in October 2019 were: WhatsApp (around 1600 millions of active users), Facebook messenger (around 1300 million), WeChat (around 1133 million), QQ mobile (around 808 million), Snapchat (around 314 million), and Telegram (around 200 million). These stats are published in stadista.com based on We are Social and other sources [1].

Therefore, students use actively IM apps choosing them above other communication media like email or phone [2, 3]. Surprisingly, students even do not identify IM as a technology but as a common communication channel [4]. On the other hand, it is important to point out that IM usage may have a negative impact on learning [5].

There are many uses of IM apps in learning processes with different strategies; among others, a significant approach is the use of IM tools to improve communication among peers and faculty [6–12]. In those contexts, results show that IM apps make communication easier, yielding an interactivity increase between peers and facilitating the knowledge share. In addition to this, the sense of presence that users perceive is higher, fostering collaboration and enabling ubiquity [13, 14]. Nevertheless, the main drawback of IM applications is inappropriate to use since if it is not employed with educational purposes, it becomes a distraction during the learning process [12, 15].

As a result, if certain educational purposes want to be achieved, it is required certain monitoring of the use of IM applications during the educational activities. Considering these assumptions, IM helps the TWC acquisition by students since it provides an attractive way of communication. With this approach, there are researchers that employ IM apps to carry out the communication among students in a group [12] and they also study their interactions [15]. The main drawback of these studies is the fact that pieces of evidence of the learning process are not considered to estimate TWC acquisition. On the contrary, in this work, the proposed methodology uses the learning pieces of evidence to assess the acquisition of such skill.

For all the above, IM can be adopted as a useful way to establish communication among students working as a group. In this paper, the use of Gitter is described because students are encouraged to employ it as it is linked with the platforms where students must store their source code, those are Github and Github Classroom. In this sense, students use Github as a Version Control System for their practical assignments and even though Gitter is not one of the most known IM apps, its functionalities are the same as others, so students get easily used to it.

This work outperforms previous research on this field since it proposes to use Gitter to simplify student cooperation when they are working on their practical assignments, and later students' interaction is analyzed in order to assess whether they have acquired the "think in parallel" skill. To achieve that goal, several experiments with first-year students that attend to the HPC subject of the Computer Science Master's degree at the University of León along different years have been carried out. In the 2015–2016, and 2019–2020 courses, students had to develop a parallel application of a real sequential problem. Each student had to choose one Application Program Interface (API) among OpenMP, MPI, CUDA, and OpenCL. Despite the API diversity, a requirement was that students had to justify their strategies with their mates and lecturers by using Gitter. The obtained results in those courses are compared with the ones obtained by students of the same subject but in different courses, specifically 2014–2015, 2016–2017, 2017–2018, and 2018–2019, where students did not use IM.

This paper is structured as follows: Sect. 2 describes the materials used in the experiments and the methodology used to evaluate the results; Sect. 3 gathers results that are discussed in Sect. 4. Finally, Sect. 5 poses some conclusions and works in progress.

2 Materials and methods

A set of experiments was carried out to evaluate the impact of using IM in the learning process of parallel computing. During the experiments, students of the HPC subject of the master's degree in Computer Science of the University of León were assigned a practical task consisting of developing a parallel application. Next, each of the elements is described, as well as the methodology and metrics used in the experiment are presented.

2.1 High-performance computing

High-performance computing is a subject in the first year of the Master's Degree in Computer Science at the University of León (Spain) where students learn the fundamentals of supercomputing and parallel programming. The main difficulty students deal with is the development of parallel algorithms to solve typical problems in computer science. Students find it hard to think in parallel so a set of basic examples is introduced to know how the typical parallel programming APIs, such as OpenMP, MPI, and CUDA can be used.

Later, a more real problem to be solved in parallel is presented to students. For that purpose, a sequential program is provided, as well as its explanation. Next, students apply the knowledge they have acquired during the practical lessons to design a parallel solution using the studied libraries. The students' grades in the subject come exclusively from the evaluation of such practical assignment.

The organization of the subject was designed based on the knowledge acquired during two Erasmus teaching stays to collaborate with the subject Parallel Computer in the Computer Science degree at the University of Groningen. Not only did the students work with how APIs allow programmers to execute code in parallel, but also they had to develop complete parallel programs for different fields such as image processing or physics.

To carry out the experiments, we have gathered data from 2014–2015, 2015–2016, 2016–2017, 2017–2018, 2018–2019, and 2019–2020 courses. Table 1 shows the number of students enrolled and the rate of students who failed a course and enrolled again

Table 1 The number of students enrolled and the rate of students who failed a course and enrolled again the next year in the HPC subject of the Master's Degree in Computer Science at the University of León (Spain) from 2014–2015 to 2019–2020

	14–15	15–16	16–17	17–18	18–19	19–20
Number of students	13	8	5	14	11	5
Old students rate	$\frac{0}{12}$	$\frac{0}{8}$	$\frac{0}{5}$	$\frac{0}{14}$	$\frac{0}{11}$	$\frac{0}{5}$
Old students %	0	0	0	0	18	0

the next year in such courses. In such courses, the lecturer was always the same, specifically Dr. Sánchez-González, who signs this work. Regarding the practical assignment, described below, was also the same in such courses. So, the only difference between them is that in 2015–2016, and 2019–2020 courses, students were encouraged to use IM to collaborate with their mates and lecturers. In addition, in the 2015–2016 course, the Comprehensive Training Model of the Teamwork Competence (CTMTC) methodology was used to promote the TWC [16]. The usage of this methodology, plus to encourage teamwork, allows for gathering data that will be later used to measure the effect of using IM in students' results.

It is important to point out that the students enrolled in the above courses have a similar background. They all come from the degree in Computer Science at the University of León. It is a common circumstance at the University of León, most students of the Master's degree in Computer Sciences, are enrolled as soon as they finish the degree in Computer Sciences. This is the case of the ones who attended the HPC subject in courses from 2014–2015 to 2019–2020. However, even if it was not the case, all students must have a computer science background to be enrolled at the Master's degree in Computer Sciences at the University of León. The requisites for accessing the Master are regulated according to Appendix 1, section 4.2 of [17]. According to this regulation, students must come from a degree in Computer Science; and if they do not, they must attend and pass some formative complements prior to enrolling the Master.

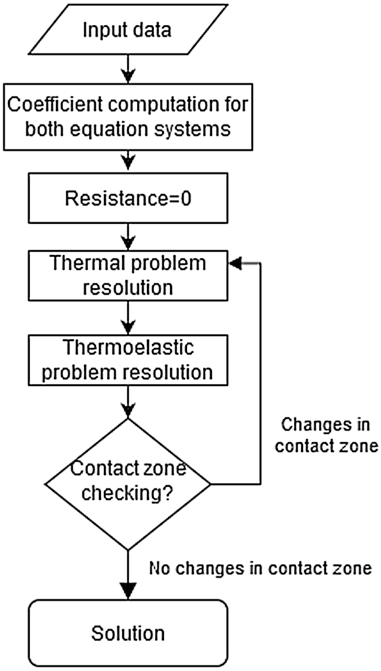
Finally, the rate of students who failed a course and enrolled again the next year is quite low, as shown in Table 1, so there is not a sign of such students to influence the performance of their mates.

2.2 Practical assignment

A real problem was presented to the students using a sequential version written in C adapted from a previously existing Fortran version. The program solves contact problems between 3D solids by applying the Boundary Element Method. This method is presented and explained with detail in [18].

In short, two solids mesh in triangular elements and then certain coefficients are computed for each pair of elements. To do this, a numerical or an analytic integral is solved depending on whether the pair is formed by the same element or not. Integrals are computed from one element to the node of another element, which is located at its barycentre. As these coefficients are independent among elements, its computation is susceptible to be run on parallel. An algorithm of this problem is shown in Algorithm 1 [19]. Figure 1 shows the workflow of the program.

Fig. 1 Scheme of the sequential program


Input data
Coefficient computation for both equation systems
Resistance=0
Thermal problem resolution
Thermoelastic problem resolution
Contact zone checking?
Changes in contact zone
No changes in contact zone
Solution

Algorithm 1 Sequential Algorithm to compute coefficients.

```
for i=1 to N do
   for j=1 to N do
      if i=j then
         AnaliticIntegral(ei)
      else
         NumericalIntegral(ei,ej)
      end if
   end for
end for
```

With these values and some other parameters of the problem, a system of equations $Ax = B$ is computed. Being N the number of elements that comprise both solids, A is a matrix N by N for the thermal contact problem and a matrix N*3 by N*3 for the elastic or thermoelastic contact problem. B is the independent vector and x are unknowns. Typical examples of contact problems involve at least 1200 elements so as to be able to design a proper geometry of the problem. The system of equations is also potential candidates to run on parallel.

Therefore, giving these details to the students and highlighting the two potential problems to run in parallel, basically, the students' task was to speed up the execution time by using several parallel techniques. Although they had to use several parallel techniques, they worked as a team so they had to collaborate to achieve the final result. Consequently, the supervision of all the development process and the decision making was required. For that purpose, in 2015–2016 course, the CTMTC methodology was applied.

Parallel Computing APIs Students were encouraged to use the parallel computing API they prefer. During the course they learned OpenMP, MPI, and CUDA; but any other API was allowed, such as OpenCL which some students decided to use.

- *Open Multi-Processing (OpenMP)* It supports multi-platform shared memory multiprocessing programming in C, C++, and Fortran on most platforms such as Linux, macOS, and Windows. OpenMP includes a set of compiler directives, library routines and environment variables that influence run-time behavior.
- *Message Passing Interface (MPI)* It is a portable message-passing standard that defines the syntax of a core of library routines useful to write portable message-passing programs in C, C++, and Fortran. There are several implementations of MPI, many of which are open-source, such as Open MPI. MPI has encouraged the development of portable and scalable large-scale parallel applications.
- *Compute Unified Device Architecture (CUDA)* It is a parallel computing platform and API model created by Nvidia [20]. It allows using a CUDA-enabled graphics processing unit (GPU) for general-purpose processing. The CUDA platform is a software layer that gives direct access to the GPU's virtual instruction set and parallel computational elements. The CUDA platform is designed to work C, C++, and Fortran. This accessibility makes it easier for specialists in parallel programming to use GPU resources. Also, CUDA supports programming frameworks such as OpenACC and OpenCL.
- *Open Computing Language (OpenCL)* It is a framework for writing programs that execute across heterogeneous platforms consisting of central processing units (CPUs), graphics processing units (GPUs), and some other processors or hardware accelerators [21]. OpenCL specifies its own programming language and APIs to control the platform and execute programs. It provides a standard interface for parallel computing using task- and data-based parallelism.

Supecomputación Castilla y León (SCAyLE) To develop the application students were granted access to Caléndula, the cluster of SCAyLE. It has several calculation clusters with different computer technology architectures. Its theoretical peak performance is currently 131.8 Teraflops [22].

Specifically, the cluster they accessed for developing their practical assignment is composed by 114 servers with the following technical specifications: 2 processors Intel Xeon E5-2630 v3 (Haswell architecture) with 8 cores @ 2.40 GHz, 32 GB (2 GB/core ratio) of main memory, Two 1-GbE interfaces, and one Infiniband FDR 56Gb/s interface.

CTMTC As mentioned above, in the 2015–2016 course the CTMTC methodology was applied to fulfill the practical assignment. It was used as part of a bigger study to measure teamwork competence acquisition [16]. This study included several subjects in addition to HPC. However, the usage of such methodology provided the data we have used to evaluate the students' interaction with their mates and lecturers through IM.

CTMTC explores the group results and how each individual has acquired competence. The methodology relies on the analysis of learning evidence from data generated by the use of IT-based learning tools by student teams during project development [23].

2.3 Gitter

Gitter is an open-source IM platform for developers and GitHub users. It provides a chat room system linked to users' repositories which allows discussing topics related to the source code. It provides most common IM features such as notifications, inline media files sharing, viewing, and subscribing chat rooms, but also a set of specific features for developers such as subscribing changes on files or issues.

Several Gitter chat rooms were created specifically for discussing issues related to the practical assignment. They were used by students to collaborate among them and with the lecturers in the design and development of their solutions. Data from students' conversations in these rooms were gathered for later analysis.

2.4 Evaluation

A three-step methodology is used to demonstrate our hypothesis. First, a statistical analysis of the grades is done (Step 1). We start with a normality test of the grades of each course. Since the number of grades in each course is quite low ($5 \leq n \leq 14$, see Table 1), we think that the non-parametric tests suit better for this problem. Thus, we use the Kolmogorov–Smirnov test with Lilliefors correction to compute the normality. Depending on the result of the normality test, a parametric or non-parametric test for comparison of means or medians is performed for each pair of courses in order to know among which variables there are statistically meaningful differences. Since not normal samples are expected and the number of students per course is different, we complete the Post Hoc comparisons by performing the Mann-Whitney rank tests.

After analyzing the statistical differences among the courses a qualitative analysis of the data will be done (Step 2). We will compare the final grades obtained by the students of 2015–2016, and 2019–2020 courses, where IM was used to interact with their mates and lecturers, with the results of the remaining courses, specifically with 2014–2015, 2016–2017, 2017–2018, and 2018–2019 courses, where IMA was not used. This analysis may indicate whether the usage of IM applications may be helpful for students in order to improve their "Think in parallel" skill. To evaluate the impact of IM we will consider the following Key Performance Indicators (KPIs): number of students, number of students who passed the course, number of students enrolled in the course who do not turn in the practical assignment, number of students who fail the course, number of students who get a C grade, number of students who get a B grade, number of students who get an A grade, number of students who get an A+ grade.

Finally, an in-depth analysis of the students' interaction during the 2015–2016 course will be carried out (Step 3). In this course, the students' interaction data

Table 2 *p* values obtained in normality tests ($\alpha = 0.05$)

Course	*p* value
14–15	0.000
15–16	0.000
16–17	0.000
18–19	0.000
19–20	0.000

Table 3 *p* values obtained by Mann–Whitney *U* test

	15–16	16–17	17–18	18–19	19–20
14–15	0.318	0.421	0.490	0.271	**0.030**
15–16		0.210	0.329	0.204	0.136
16–17			0.335	0.500	0.087
17–18				0.183	0.082
18–19					0.230

Values in bold denote *p* values $\alpha = 0.05$

through chat rooms were gathered for later analysis of TWC acquisition [16]. We aim to use this data to demonstrate that this interaction also was helpful to acquire the "think in parallel" skill. In order to do so, the conversations between students will be reviewed to extract some KPIs which will result helpful in a better comprehension of the effects of using IM in the learning process of parallel computing. The KPIs that will be considered are the number of messages and words that each student wrote in the chat rooms. According to [24], there is a strong positive correlation between students' grades and messages. So, in order to evaluate the effect of using IM on the students' results, the Pearson Correlation Coefficient (PCC) between the above KPIs and the grades is computed.

3 Results

Number of students enrolled and the rate of students who failed a course and enrolled again the next year in the HPC subject of the Master's Degree in Computer Science at the University of León (Spain) from 2014–2015 to 2019–2020. The number of students is quite low ranging between 5 and 14.

Table 2 shows the results of the normality tests for every course grades. As expected due to the samples' size, any of them do not follow a normal distribution since their *p* values are lower than the significant level ($\alpha = 0.05$).

Table 3 shows Post Hoc comparisons for each pair of courses. According to the table, we cannot assert that there are statistically significant differences among the samples since their *p* values are lower than the significant level ($\alpha = 0.05$) except between 2014–2015 and 2019-2010 courses.

Table 4 shows the number of students who passed each course, who failed, and enrolled in the course but did not turn in the assignments. Regarding the students who passed the course, the table also shows the letter grades' distribution of each course.

Table 5 shows some descriptive statistics for the grades of each year, specifically: minimum, first quartile, mean, median, third quartile, and maximum.

Table 6 shows the Students' numerical and letter grades in 2015–2016 course. The table also includes the number of messages and words that each student wrote in the chat rooms. Figure 2 shows scatter plots of messages versus grades, and words versus grades. Table 7 shows the PCC of grade versus messages (0.84) and grades versus words (0.91).

Table 4 Number of students that being enrolled in the course but do not turn in the assignments; that fail the course; that pass the course; and that get a C, B, A, and A+ grade

	14–15	15–16	16–17	17–18	18–19	19–20
Not turned in	1	0	0	0	0	0
Fail	0	3	0	4	1	0
Pass	12	5	5	10	10	5
C	2	2	2	2	3	0
B	8	0	2	3	3	2
A	2	2	1	6	4	2
A+	0	1	0	0	0	1

Table 5 Descriptive statistics for students' grades

	14–15	15–16	16–17	17–18	18–19	19–20
Minimum	5.0	3.0	6.7	0.0	2.0	7.4
1st quartile	7.0	3.3	6.7	2.6	5.9	7.9
Mean	7.4	6.3	7.8	6.2	7.4	8.8
Median	7.5	5.5	7.2	7.9	7.9	9.0
3rd quartile	8.0	9.9	9.1	9.2	9.9	9.7
Maximum	10.0	10.0	9.6	9.6	9.9	9.8

Table 6 Students' grades in 2015–2016 course

Student id.	Numerical grade	Letter grade	Messages	Words
1	4	F	0	0
2	3.1	F	9	199
3	10	A	70	4093
4	9.5	A	29	2008
5	10	A+	56	3732
6	5.5	C	10	421
7	3	F	16	449
8	5.4	C	2	258

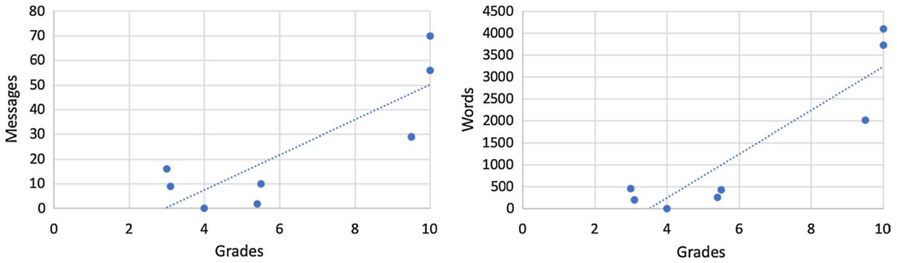


Fig. 2 Scatter plots of messages versus grades (left) and words versus grades (right)

Table 7 PCC between grade and messages and words

	Messages	Words
Grade	0.84	0.91

4 Discussion

In order to assert that IM helps students to acquire the “think in parallel” skill, we need to analyze the above results. Specifically, we have to compare the students’ results in 2015–2016 and 2019–2020 courses, where IM was used to interact with mates and lecturers, with the results in 2014–2015, 2016–2017, 2017–2018, and 2018–2019 courses, where IM was not used.

Table 1 shows the number of students of each course from 2014–2015 to 2019–2020 (6 courses). The average number of students per course is 9, the maximum is 14, and the minimum is 5. With such numbers, a non-parametric test to compute the normality is the best option. The Kolmogorov–Smirnov tests demonstrate that there are not normal samples as shown in Table 2. Table 3 shows Post Hoc comparisons between each pair of samples to check whether there are statistically meaningful differences among them. According to the table, there are not statistically meaningful differences between most samples, except between 2014–2015 and 2019–2020 courses.

Table 4 shows the number of students who pass the HPC subject each course. 2015–2016 course got the second-highest failure rate. This could be a hint of IM is not helpful to acquire the “think in parallel” skill. The in-depth analysis of the 2015–2016 course performed below will allow us to test it out. Table 4 also shows the letter grades obtained each course. It shows that the highest grades correspond to 2015–2016 and 2019–2020 courses. To intuitively illustrate this, Fig. 3 shows the grades’ percentage distribution. This figure shows that not only there is a high percentage of A grades in 2015–2016 and 2019–2020 courses, but also they are the only courses with A+ grades. Moreover, regarding the students who reached the highest grades (A and A+), the average ratio is considerably higher in 2015–2016, and 2019–2020 courses, 40% (As), and 20% (A+s); than the average ratio in the remaining courses, 34.17% (As), and 0% (A+s). This fact may be an indicator that IM is helpful to acquire the “think in parallel” skill.

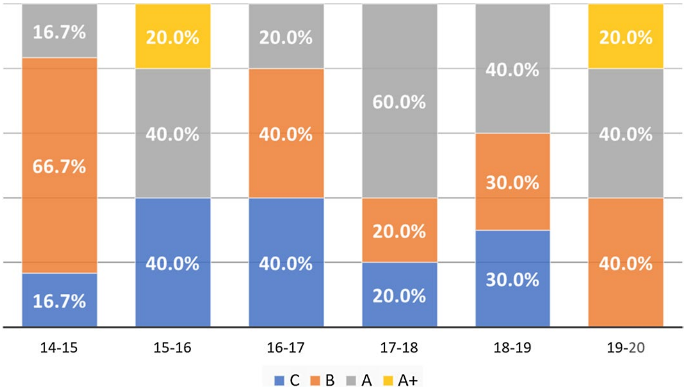


Fig. 3 Grades' percentage distribution

Regarding the descriptive statistics in Table 5, the mean and median in 2019–2020 course (8.8, and 9.0), where IM was used, are the highest compared to the remaining courses. These values may be an indicator that IM helps students to acquire the "think in parallel" skill. However, if we look out the mean and median in the 2015–2016 course (6.3, and 5.5) are the lowest compared to the remaining courses. Again, an in-depth analysis of 2015–2016 course is required to explain such variations.

Further analysis of the 2015–2016 course was made by applying the CTMTC methodology. The methodology considers five main aspects: objectives, responsibilities, planning, norms, and execution. A chat room was created for each one.

First, students had to design the goal and objectives of their assignment. As a result, they had to identify the technologies that were going to use and the code fragments to be run on parallel.

Then, students chose which API they preferred to work with, although they were aware that they also had to help others. After a debate process in the planning chat room, they reached an agreement of who was responsible for each approach. Observing how they made decisions, it was clear who had understood perfectly well what had been studied during the previous lessons since they were more active giving arguments and deciding how to face the problem. They also found more challenging the use of libraries that had not been studied during the previous lessons such as OpenCL.

Planning was the hardest part since they found it really difficult to estimate the time of each task according to their messages. Lecturers had to be involved giving some advice about how much time they should expend designing, developing, and testing.

In order to properly work as a group, they established a set of rules after debating them in the norms chat room.

Finally, they commented on all the aspects related to the execution in the execution chat room. Following their discussions, it was clear to identify who was able to solve most of the problems, who provided useful solutions, who needed help, etc.

Lecturers as observers of the chat were able to give proper orientations to students right away, supporting some good contributions of some students or refusing those that were impossible to develop on time.

Table 6 shows detailed data of the 2015–2016 course. In addition to the numerical and letter grades, we can see the number of messages and words that each student wrote in the chat rooms. Figure 2 shows the scatter plot of the number of messages versus grades, and the number of words versus grades. The figure shows a strong positive correlation. To compute this correlation, Table 7 shows the PCC of messages versus grades (0.84) and words versus grades (0.91).

According to Table 6 students 1, 2, and 7 did not pass the course. As mentioned above, it is a high failure percentage comparing to other courses. Attending to the number of messages, students with a high grade (A or A+), wrote between 29 and 70 messages. Considering the number of words, they wrote between 2008 and 4093 words. Numbers are quite different for students with a low grade (C), who wrote between 2 and 10 messages, and between 258 and 421 words. If we look out to the students who failed the course, the number of messages ranges between 0 and 16, and the number of words ranges between 0 and 449. These numbers allow for explaining the high failure ration in the 2015–2016 course. This rate has to do with the fact that students who failed or got a low grade did not interact with their mates and that this fact was a drawback to them.

5 Conclusions

The "think in parallel" skill is essential for educational institutions and in order to improve the employability of students. This study describes and evaluates the use of IM applications to facilitate the learning process of parallel computing in Higher Education.

The case study suggests that using Gitter has a positive impact on students' interactions and final grades. Furthermore, students have a positive perception of the experience, and they feel that it helps them to improve their skills in parallel computing.

Results in 2015–2016, and 2019–2020 courses, compared to 2014–2015, 2016–2017, 2017–2018, and 2018–2019 courses, demonstrate that using IM to interact with mates and lecturers has a positive effect on students' results. Moreover, results in the 2015–2016 course show that the usage of using IM has a strong positive correlation with the grades. Students who interacted most using the chat rooms to collaborate with their mates got the highest results. On the contrary, students who were not so actives in the chat rooms achieved a low grade or even failed the course. The reason for that failure, since all students were working collaboratively, has to do with the fact that students who failed were not able to keep the work pace of their mates and they quit. However, students that continued working helped and were helped by their peers, so finally they passed the subject. Gitter conversations make clear the students that helped most to their mates and solve the most important problems they had to deal with got the best results. Tracking those instant messages also makes it possible to realize how students that in other circumstances would not be able to pass the subject, were encouraged by their mates and oriented so as they kept working achieving the objectives.

Further works have to do with the fact that there are not too much IM interaction data to work with. Obtaining more data will allow for analyzing such interactions to understand their effects on academic success. On the other hand, since the sample size is small, students' individual motivation may have a high impact on the results, so its analysis would be desirable.

It would be also interesting to compare interaction using IM, with other interaction methods such as email, forums, etc. This comparison would allow for asserting that IM encourages interaction more than any other kind of interaction between students.

Besides, if IM usage is going to be encouraged, its scalability should be studied, since staying offline for some hours can make it very difficult following or even understanding some conversations.

Acknowledgements This work was in part supported by the European Commission Erasmus+ (2018-1-ES01-KA201-050939) Grant, and by the Spanish Ministry of Science, Innovation, and Universities RTI (RTI2018-100683-B-I00) Grant.

References

1. Stadista. Most popular global mobile messenger apps as of October 2019, based on number of monthly active users. https://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-apps/. Accessed 19 Apr 2020
2. Carnevale D (2006) Email is for old people. Chron High Educ 53(7):A27
3. Wang J (2013) What higher educational professionals need to know about today's students: online social networks. Turk Online J Educ Technol 12(3):180
4. Lewis C, Fabos B (2005) Instant messaging, literacies, and social identities. Read Res Q 40(4):470
5. Jeong W (2007) Instant messaging in on-site and online classes in higher education. Educause Q 30(1):30
6. Hrastinski S, Edman A, Andersson F, Kawnine T, Soames CA (2014) Informal math coaching by instant messaging: two case studies of how university students coach k-12 students. Interact Learn Environ 22(1):84
7. Lents NH, Cifuentes OE (2010) Increasing student–teacher interactions at an urban commuter campus through instant messaging and online office hours. Electron J Sci Educ 14:2–13
8. Smit I, Goede R (2012) Whatsapp with blackberry; can messengers be mxit, a philosophical approach to evaluate social networking sites. Cape Peninsula University of Technology https://repository.nwu.ac.za/handle/10394/13628. Accessed 7 Sept 2020
9. Sweeny SM (2010) Writing for the instant messaging and text messaging generation: using new literacies to support writing instruction. J Adolesc Adult Lit 54(2):121
10. Lauricella S, Kay R (2013) Exploring the use of text and instant messaging in higher education classrooms. Res Learn Technol 21:19061

11. Klein AZ, Junior JCdSF, Barbosa JLV, Baldasso L et al (2018) The educational affordances of mobile instant messaging (mim): results of whatsapp® used in higher education. Int J Distance Educ Technol 16(2):51
12. Bouhnik D, Deshen M, Gan R (2014) Whatsapp goes to school: mobile instant messaging between teachers and students. J Inf Technol Educ Res 13(1):217
13. Fox AB, Rosen J, Crawford M (2009) Distractions, distractions: Does instant messaging affect college students' performance on a concurrent reading comprehension task? Cyber Psychol Behav 12(1):51
14. Junco R, Cotten SR (2011) Perceived academic effects of instant messaging use. Comput Educ 56(2):370
15. Gronseth S, Hebert W (2019) Groupme: investigating use of mobile instant messaging in higher education courses. TechTrends 63(1):15
16. Conde MÁ, Rodríguez-Sedano FJ, Sánchez-González L, Fernández-Llamas C, Rodríguez-Lera FJ, Matellán-Olivera V (2016) In: Proceedings of the Fourth International Conference on Technological Ecosystems for Enhancing Multiculturality, pp 787–794
17. BOE-A-2009-12977. Resolución de 8 de junio de 2009, de la Secretaría General de Universidades, por la que se da publicidad al Acuerdo del Consejo de Universidades, por el que se establecen recomendaciones para la propuesta por las universidades de memorias de solicitud de títulos oficiales en los ámbitos de la Ingeniería Informática, Ingeniería Técnica Informática e Ingeniería Química. https://www.boe.es/boe/dias/2009/08/04/pdfs/BOE-A-2009-12977.pdf. Accessed 21 Apr 2020
18. Vallepuga-Espinosa J, Sánchez-González L, Ubero-Martínez I, Riego-Del Castillo V (2018) In: The 13th International Conference on Soft Computing Models in Industrial and Environmental Applications. Springer, pp 544–552
19. González R, Vallepuga J, Sánchez L (2011) In: XXII Jornadas de Paralelismo, La Laguna, Spain
20. NVIDIA. CUDA Zone. https://developer.nvidia.com/cuda-zone. Accessed 31 Jan 2020
21. K. group. OpenCL overview. https://www.khronos.org/opencl/. Accessed 31 Jan 2020
22. Supercomputación Castilla y León (SCAyLE). Sistemas y recursos. https://www.scayle.es/solicitar_una_cuenta_de_usuario/sistemas-y-recursos/. Accessed 31 Jan 2020
23. Lerís D, Fidalgo Á, Sein-Echaluce ML (2014) A comprehensive training model of the teamwork competence. Int J Learn Intellect Cap 14 11(1):1
24. Fidalgo-Blanco Á, Sein-Echaluce ML, García-Peñalvo FJ, Conde MÁ (2015) Using learning analytics to improve teamwork assessment. Comput Hum Behav 47:149

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.