Visual Analogy Videos for Understanding Fundamental Parallel Scheduling Policies

Nasser Giacaman[a], Oliver Sinnen[a], Joel Adams[b]

[a] *Department of Electrical, Computer, and Software Engineering*
University of Auckland, Auckland, New Zealand.
[b] *Department of Computer Science*
Calvin University, Grand Rapids, USA.

Abstract

Parallel and distributed computing (PDC) education is increasingly gaining greater recognition as a core topic in undergraduate computing degrees. While the *application* of PDC concepts to software development involves the use of highly-technical tools and libraries typically reserved for advanced courses, PDC educators are seeking pedagogical approaches that can be used to introduce PDC concepts in earlier, introductory courses. This study presents such an approach, and aims to introduce undergraduate students to fundamental PDC *concepts* without the expectation that they can apply those concepts. The proposed approach is inspired by the success seen in the wider computing education literature, where analogies and visualization have helped students understand other abstract computing topics. The proposed learning resources comes in the form of a series of short videos, carefully aligned to a learning activity that guides towards achieving the intended learning outcomes. In addition to being a simple activity to complete with students, evaluations illustrate its value even with minimal guidance from the instructor. The proposed approach is studied as both a synchronous in-class activity guided by the instructor, as well as an asynchronous online self-directed activity. These two studies produced different outcomes with respect to student learning, revealing an important implication for designers of instructional material to consider.

Preprint submitted to Elsevier *22nd April 2021*

Keywords: Analogies, online learning, parallel computing, videos, visualization.

1. Introduction

The pervasiveness of parallel and distributed computing (PDC) technologies has been recognized by various computing curricula, making it important to cover them at the undergraduate level [1, 2]. Despite the difficulty of PDC concepts, they are deemed important for early computer science (CS) courses [3]. It is also important to begin exposing CS students to these PDC concepts early on, rather than delaying their introduction to later years [4, 5]. In the broader computing education field, the use of teaching tools that emphasize programming concepts instead of technical details, often using visualization, has frequently proven to be successful [6, 7].

One successful approach is to use carefully constructed analogies that focus on the underlying concepts and simplify technical details [8]. Such analogies can help students by making abstract topics more concrete [9]. On their own, analogies may unintentionally embed many weaknesses; as a textual or verbal anecdote, an analogy may remain too abstract. However, if students are left to interpret and visualize such analogies using their own imagination, this can potentially lead to misconceptions [10], which can be worsened if an analogy is ill-thought-out and "spur-of-the-moment".

By contrast, well-thought-out visualizations have the potential to help make abstract concepts more concrete, and therefore improve a student's understanding of those concepts [11].

The work in this paper incorporates the combined potential of analogies and visualization; *visual analogies* are viewed as a way to make the analogy less vague, and thus help students visualize the analogy as pedagogically intended.

Without a visual aid, students will only be frustrated if they cannot relate to the analogy [12].

Videos have been shown to enrich student learning [13, 14]. Incorporating videos into the learning process has been shown to improve motivation and cater for diversity in learning styles [15, 16]. To aid the delivery of visual analogies, the approach in this paper uses short video clips to convey fundamental parallel programming concepts. The use of short videos is a factor in increasing student engagement [17]. This paper explores the impact of the videos in two teaching contexts: face-to-face synchronous and online asynchronous learning environments [18].

The contributions of this paper include:

- A quantitative and qualitative analysis on the value of visual analogy videos for teaching fundamental parallel programming concepts.
- A comparative study of an activity's impact when implemented as a guided in-class (synchronous) activity versus an individual online (asynchronous) activity, revealing an important implication for designers of instructional material.
- The resources descrited in the paper are all available for instructors to use in their courses, including the videos and activity forms, for both synchronous and asynchronous formats.

The primary research question to be investigated is:

> ***RQ:*** *To what extent do visual analogy videos help students learn fundamental parallel programming concepts?*

The rest of this paper is structured as follows. Section 2 presents related work on analogies and visualization for PDC education. Section 3 presents the methodology, including the course context, videos and activity. The results are presented

in Section 4, followed by a discussion in Section 5. Conclusions and future work are presented in Section 6.

2. Related Work

An analogy helps students to understand an abstract concept (the *target*) by relating it to experiences with which they are familiar (the *source*) [19, 8]. A well-defined analogy benefits a student's understanding of an abstract concept by making that concept concrete [9]. Analogies are made effective when the source concepts are already familiar to and understood by students. In addition, clear semantic and structural correspondences between source and target analogs also contribute to an analogy's effectiveness [8, 19]. However, the use of analogies also presents a risk when there is a clear disconnect between the correspondence of the source and target analogs. This risk can lead to students' misunderstanding of the concept, resulting in misconceptions [8]. Examples of how misconceptions are induced include misleading or missing properties, and focusing on surface-level descriptive aspects [20].

Analogies and metaphors are frequently used across the discipline of computing [21]. Despite analogies being commonly used, there is little research on their impact in computing education [22]. While metaphors and visualizations are assumed to improve student learning, more empirical research is welcomed [23]. The benefits are most easily noticed in short-term learning; understanding the long-term benefits of analogies is more difficult [24]. Multiple metaphors can also be interleaved together to form an *allegory*; however, measuring the differences of metaphors versus allegories is again difficult [25].

Visualizations are often used by educators to represent and describe abstract concepts to students through the use of illustrations. These visual illustrations are found to be a useful method to assist students' understanding of abstract

concepts [26]. The use of 3D visualizations, rather than 2D visualizations, can also help improve the metaphor's clarity [27, 28]. However, just like analogies, visualizations can backfire if the concept and its corresponding visual components are poorly mapped [29]. This misrepresentation leads to erroneous understanding of the concepts being taught, therefore giving rise to misconceptions. Even if the visualization is well constructed, learners must actively engage with the visualization for it to be an effective learning activity as opposed to only passively viewing it [11].

The PDC education community is gradually building up the number of resources dedicated to teaching PDC concepts [30]. These include a repository of unplugged PDC activities [31] that includes analogies, role-playing activities, and games. A classification system is also emerging in response to the expansion of PDC activity repositories [32]. Given the general difficulties of learning PDC concepts, different teaching approaches have been explored in an effort to move away from traditional teaching approaches. For example, a flipped classroom approach has been shown to improve students' understanding and subsequent application of PDC concepts [33]. A practical in-class programming approach has also improved students' understanding [34].

The Thread Safe Graphics Library (TSGL) supports the use of visualization in learning PDC concepts, allowing students and instructors to observe the behavior of a parallel application at runtime [35]. ParaVis has similar motivations, with students engaging more in the parallel lab activities [36]. Inspired by popular block-based languages (such as Scratch [37]), extensions are emerging that utilize PDC concepts [38].

Finally, this paper extends the initial work introduced with ParallelAR [39], which provided an augmented reality mobile app for demonstrating parallel scheduling policies. The contributions made beyond this earlier work includes

an in-depth evaluation of the learning impact of using short visual analogy videos in a PDC course. This paper's two-year study was carried out in both a traditional, on-campus format, as well as in an online, virtual format. The results from this evaluation provide insightful lessons to help future efforts in developing visual analogies.

3. Methodology

This section presents details of the study's evaluation design. This includes the context of the course in which the analogy was used, the activities carried out by participants, and details of the analogy videos.

3.1. Course Context

The study was carried out in the context of a 4^{th} year undergraduate course, which is also offered to masters-level graduate students. The students are from the University of Auckland, New Zealand, specializing either in Computer Systems Engineering or Software Engineering. The 12-week teaching semester is split into two sections: The first half involves instructor-led teaching, then the second half involves student-led teaching in the form of presentations. A previous publication dedicated to this course provides additional insights [40], but this is unnecessary for the purposes of this paper.

This study was carried out near the beginning of the semester, when students are first presented with fundamental parallel programming concepts and the technical details needed to make use of them, such as concurrency, threads, thread-safety, synchronization, task parallelism, threadpools, and related topics. All these are taught through live coding, which has been shown to help students understand the technical programming process [41, 42]. While important for students to be able to *apply* these concepts [43] (hence the live coding),

understanding the underlying parallelization concepts being used (such as balancing workload and minimizing overhead) is also important.

3.2. Learning Activity Design

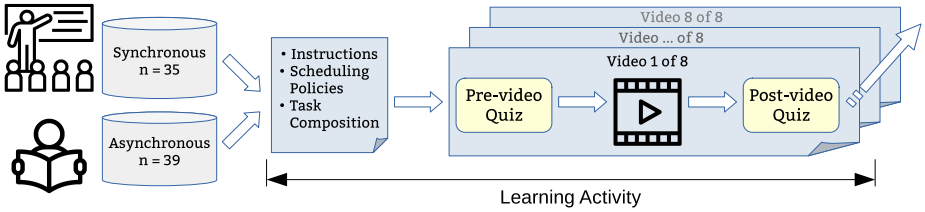


Figure 1: The process of incorporating the video learning activities was identical in both the face-to-face synchronous and online asynchronous course offerings. Students were provided with instructions (always visible in the synchronous version, and downloadable in the asynchronous version), repeating pre- and post-video quizzes for each of the eight videos.

Figure 1 illustrates the overall methodology of this study. The core elements of the study are focused on the *Learning Activity*, which is comprised of instructions followed by a series of eight videos (each with its own pre-video and post-video quizzes). The specific content of each video will be detailed in Section 3.4, but first a higher-level overview is presented. The Learning Activity requires approximately 30-45 minutes to complete in its entirety, and is ungraded. This Learning Activity was carried out twice, in the following course-formats:

1. **Synchronous (guided, face-to-face):** In the first iteration in this study, 69 students were enrolled in the course. As is the case in most years, the course was taught face-to-face with the students and instructor physically co-located in the same room on campus for all the course lessons, including the Learning Activity being described here. Since it took place in a classroom that has no computers, it required students to Bring Your Own Device (BYOD) to input their answers to the video-quiz questions. This iteration of the activity was *guided by the instructor*, who used a projector

to play and display the videos for the students, and included time for the students to complete the respective pre-video and post-video quizzes. Each video was played exactly twice, except for the sequential-execution videos which were only played once. When the videos were played to students, the instructor did not provide any explanations; students were left to interpret the videos on their own to ensure their post-video responses were based purely on the videos. For the same reason, the instructor did not field questions from students during the activity. Due to low attendance on the day of the activity (and possibly not all students having a BYOD), 35 students (51% of the class) participated.

2. **Asynchronous (individual, online):** In the second iteration of this study, 44 students were enrolled in the course. This course offering coincided with the COVID-19 pandemic, during which face-to-face teaching was not possible due to a national lockdown. The course was subsequently delivered online, with all teaching (including the "live coding") delivered to students via videos of screen-recordings. As the original learning activity was already in an online-ready format (i.e., it required a BYOD to answer questions), the same activity was largely repeated. The primary change was that links to the videos were embedded directly within the learning activity, at the precise point where the instructor had played it for the students in the first iteration of the study. With this format and no instructor guiding the activity, students were able to work at their own pace. Between the pre-video and post-video quizzes, students could watch the videos as many times as they wished as they progressed through the activity. Because the videos were hosted online, usage data was collected for the students' video-watching behavior and how fast they progressed. At the end of the activity, students were asked open-ended questions regarding what aspects of the activity helped their learning, and what aspects

did not work well. As the asynchronous format provided greater flexibility (and due to the pandemic), students were given two days in which to complete the activity. A total of 39 students (89% of the class) participated.

The Learning Activity consisted of the following steps:

- Students were given a one-page PDF handout containing simple instructions. This handout was mostly used as a reference to describe what the students would encounter in the videos, including the key parameters that varied from one video to another:

  - The *scheduling policy options*: *Sequential* (i.e., single-threaded), *Fully Parallel* (one thread per task), *Static Taskpool* (one thread per core, tasks pre-assigned), or *Dynamic Taskpool* (one thread per core, tasks assigned dynamically.
  - The *task composition options*: *Coarse-Grained* (eight 4-second tasks), *Fine-Grained* (forty 1-second tasks), or *Mixed* (six 4-second tasks, six 3-second tasks, and twelve 1-second tasks).

  By specifying how many tasks were in each composition, how much time each task required to complete, how many threads were present, and how many physical cores are present, this handout provided key details the students needed to answer the pre- and post-video questions.

  In the case of the synchronous offering, this handout was displayed using a secondary classroom projector. In the case of the asynchronous offering, students were provided with a downloadable version.

- Using the handout, students completed a sequence of eight exercises, each representing a computation with a unique combination of scheduling policy and task composition. Each exercise involved three steps:

1. For a given combination of scheduling policy and task-type, students completed a short *pre-video quiz*. After consulting the handout, students were asked to estimate the computation's completion time and (optionally) to describe the key "learning points" (i.e. pros and cons) for this combination of scheduling-policy and task-type.
2. Students then watched the short *video* (details in Section 3.4) showing the behavior of the computation for the given combination.
3. After watching each video, students completed a short *post-video quiz*. Here, students again estimated the completion time and (optionally) described the combination's key learning points. Using a 5-point Likert scale, students were also asked to rate the video for its helpfulness.

The three-step nature of each exercise (*pre-video quiz* $\twoheadrightarrow$ *video* $\twoheadrightarrow$ *post-video quiz*) is inspired by Peer Instruction (PI), which aims to engage students by having them apply core concepts in the classroom [44]. PI involves posing a question for all students (here, the *pre-video quiz*), followed by a "peer discussion" (here, the peer discussion is replaced with watching the *video*), and finally concluded by the repeat questioning (here, the *post-video quiz*). This design helps measure the immediate learning impact of the videos.

Students were not given the results of their performance until the conclusion of the study, when all students completed the activities. This was particularly important in the asynchronous version, as otherwise students that might have already attempted (at least parts of) the activities may influence the responses of other students. It was therefore important to withhold the results until all students finished, as is typical in any assessment activity.

3.3. Analogy Videos

The short videos presented to students were based on an analogy that has been used in an existing augmented reality (AR) app [39]. The core elements of the analogy are illustrated in Figure 2, showing the mapping of technical content (threads and cores) to the analogy (workers and desks):

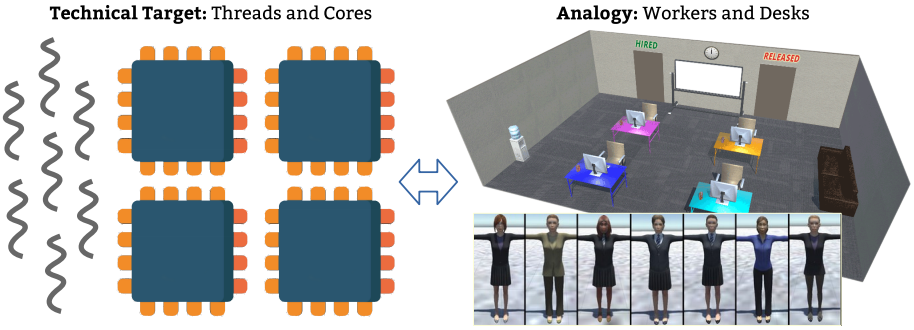


Figure 2: The analogy [39] is based on the concept of an office space (representing the computer system), with workers (representing software threads) assigned to desks (representing physical cores) to execute tasks.

The number of workers and the tasks assigned to workers differ, depending on the scheduling policy. There are always four desks in the analogy, representing a quad-core system. The videos were reproduced directly from the original AR app using a screen recorder; each video was less than a minute in length. The videos were stored in the mp4 format, allowing playback on most digital devices. Figure 3 shows a screenshot of one of the analogy videos.

3.4. Video Content

Table 1 details the content of the eight videos. Each video depicts a behavior that is determined by a unique combination of a scheduling policy and task composition; this behavior illustrates the relevant learning points. Based on the specific combination, students were asked to estimate the overall completion

Table 1: Video Content Details

Task Composition	**Video**	**Scheduling Policy**	**Intended Learning Points**	**Target Answer**	
				Time	**Speedup**
Identical Coarse-Grained:	V1	Sequential	Under-utilization of multi-core system, only one core is used.	32 seconds	–
› 8× 4-second tasks	V2	Fully Parallel	Appreciate the value of parallelization.	~ 9 seconds	×3.6
Identical Fine-Grained:	V3	Sequential	Under-utilization of multi-core system, only one core is used.	40 seconds	–
› 40× 1-second tasks	V4	Fully Parallel	Large overhead when creating and scheduling a large number of small tasks. High cost of context switching due to high number of threads compared to the low number of available cores.	~ 28 seconds	×1.4
	V5	Static Taskpool	Appreciate the value of reusing threads. Statically assigning tasks to threads upfront improves core utilization and reduces runtime overhead.	~ 11 seconds	×3.6
Mixed (Coarse- and Fine-Grained):	V6	Sequential	Under-utilization of multi-core system, only one core is used.	54 seconds	–
› 6× 4-second tasks › 6× 3-second tasks › 12× 1-second tasks	V7	Static Taskpool	Static scheduling does not work well when the workload distribution is not balanced due to tasks having mixed levels of computation.	~ 23 seconds	×2.3
	V8	Dynamic Taskpool	Appreciate the value of dynamically allocating tasks at runtime for unbalanced tasks, resulting in a balanced workload among the threads.	~ 14 seconds	×3.8

Figure 3: Screenshot of a visual analogy video (Video #2 from Table 1). It demonstrates the behavior of eight threads (workers) as they context switch on a system with four physical cores (desks), i.e., a quad-core processor. Each thread is assigned one of the coarse-grained tasks, shown at the top of the screen.

time (called "*target time*"). Estimating this time for the *Sequential* policy is trivial (one thread performing eight 4-second tasks requires 32 seconds). Estimating the times for the parallel policies (i.e. *Fully Parallel*, *Static Taskpool*, and *Dynamic Taskpool*) requires more effort, as the threads performing the tasks are now time-shared across the four processing cores. As such, target times are approximated for parallel executions, taking into account (exaggerated) elements of the analogy (e.g., overhead from worker context switches). The order of the videos is such that students are gradually exposed to parallel concepts; the introduction of each new scheduling policy is motivated by the recognition of how an already-seen scheduling policy is inefficient when used with a particular task composition. For example, Video #6 has one thread performing tasks of mixed granularities, leading to inefficient core utilization. Video #7 improves that situation by using one thread per core with pre-assigned tasks, but that

leads to poor load-balancing of the tasks across the threads, which motivates Video #8.

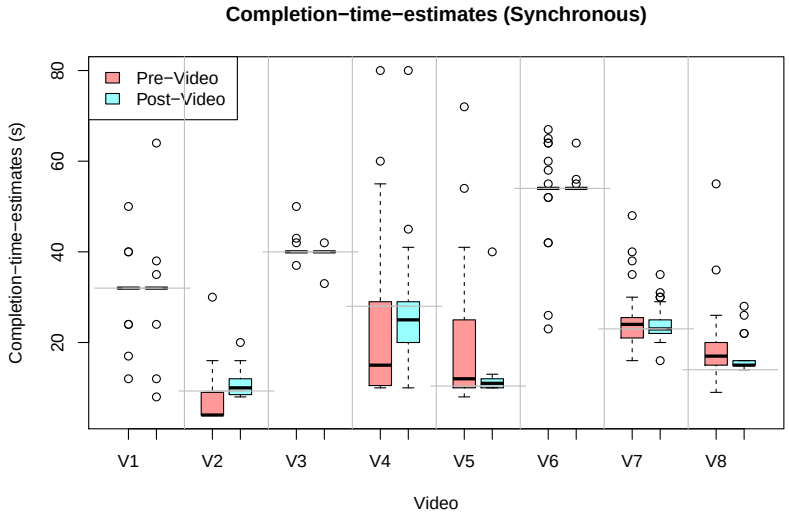
4. Results

This section presents the key results of the learning activity for the synchronous and asynchronous courses. Although the course has both undergraduate and graduate-level students, the analysis does not separate them into smaller groups as there are no notable differences.

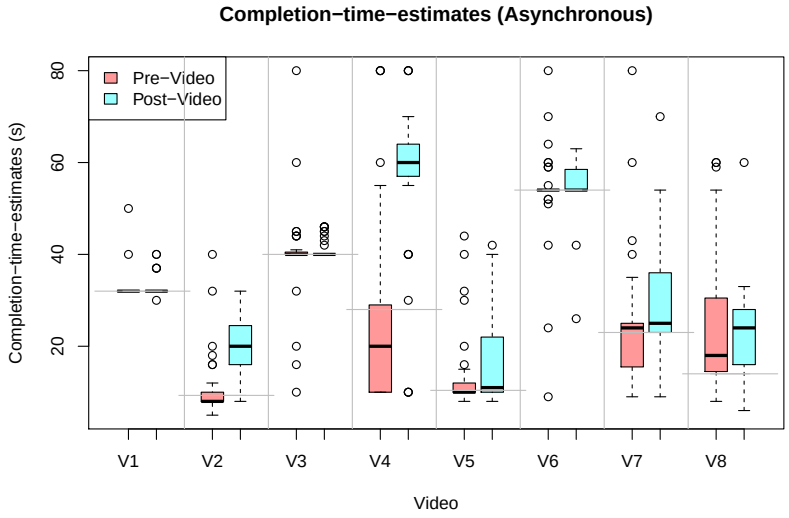
4.1. Learning Impact (Pre-video vs Post-video Quiz Performance)

Figure 4 reports the distribution of students' time estimates for each of the eight videos for the two course offerings. Each plot includes both the *pre-video* time-estimates and the corresponding *post-video* estimates. The short horizontal bar for each video represents the "target time" (as defined in Table 1). In both courses, the students found it much easier to estimate completion times for sequential programs than for parallel programs. Figure 4 thus provides a quantitative indication of the relative difficulty of analyzing the behavior of a parallel program compared to that of a traditional sequential program.

For each of the videos, the plots also display a short horizontal line representing the respective "target time" for that video (as in Table 1). Across both versions of the activity, the range of student time-estimates were much closer to the intended target for the *Sequential* policy (V1, V3, and V6), in which a single thread performs all the tasks. All the other scenarios involve multithreading and a parallel scheduling policy, in which case the students made more diverse estimates that were further from the respective target time. This holds true in both the *pre-video* and *post-video* estimates, suggesting that students are inherently having more difficulty inferring the performance of a parallel program compared to a sequential program.



(a) Synchronous



(b) Asynchronous

Figure 4: Distribution of students' completion-time-estimates for each video in both courses.

To understand the significance of the results presented in Figure 4, a statistical analysis is presented in Table 2. For each of the course-offerings, the

pre-video averages ($\bar{x}_{pre}$) and *post-video* averages ($\bar{x}_{post}$) are computed, along with the respective Coefficient of Variation ($CV\%$). A lower $CV\%$ denotes a smaller spread of estimates, suggesting a tighter convergence of understanding across the set of students. One-tailed t-tests were carried out for each video, to determine if there was any statistically significant difference between $\bar{x}_{pre}$ and $\bar{x}_{post}$. Using a significance threshold of 0.05, the bolded results in Table 2 denote statistically significant differences. For those results that differ significantly, *Impact* indicates whether the students' post-quiz performance changed for the better (positively, ↗) or for the worse (negatively, ↘) in comparison to their pre-quiz performance.

In both offerings, none of the *Sequential* videos (V1, V3, and V6) resulted in any statistically significance difference—regardless of whether the Learning Activity was conducted synchronously or asynchronously. As can be seen in Figure 4, students were largely calculating the correct target answer in their *pre-video* quizzes, and nothing in the Learning Activity led them to calculate different answers in their *post-video* quizzes.

In the Synchronous course-offering, four of the five parallel videos (V2, V4, V5, and V8) exhibited statistically significant improvements (↗) in the students' time-estimates (another parallel video, V7, was close at p=0.068). Quite surprisingly in the Asynchronous course-offering, four of the five parallel videos (V2, V4, V5, and V7) exhibited statistically significant *worse* differences (↘) in the students' time-estimates! This can also be seen in Figure 4, where the students' *post-video* quiz time-estimates tended to diverge from the intended target time.

In addition to estimating the completion times, students were also asked to write optional learning points for both the *pre-video* and *post-video* quizzes. The goal of this was to give students an opportunity to reflect on the core concepts

Table 2: Statistical significance of pre-video and post-video time-estimate differences

Video	Target	Synchronous						
		$\bar{x}_{pre}$	$\bar{x}_{post}$	$CV_{pre}\%$	$CV_{post}\%$	t-value	p-value	Impact
V1	32 s	32.8	31.7	30.0	24.8	0.536	0.702	
V2	~ 9 s	7.6	10.5	74.4	26.4	**-3.595**	$<$ **0.001**	↗
V3	40 s	40.3	39.9	4.6	3.1	1.183	0.877	
V4	~ 28 s	21.9	26.3	74.5	47.4	**-1.956**	**0.029**	↗
V5	~ 11 s	20.0	11.7	76.7	42.9	**3.343**	**0.001**	↗
V6	54 s	53.1	54.5	16.3	3.2	-0.787	0.782	
V7	~ 23 s	25.0	24.0	26.6	15.6	1.528	0.068	
V8	~ 14 s	18.7	16.2	41.8	19.9	**1.918**	**0.032**	↗

Video	Target	Asynchronous						
		$\bar{x}_{pre}$	$\bar{x}_{post}$	$CV_{pre}\%$	$CV_{post}\%$	t-value	p-value	Impact
V1	32 s	32.7	32.7	9.6	6.7	-0.161	0.564	
V2	~ 9 s	10.6	18.7	63.6	34.9	**-5.789**	$<$ **0.001**	↘
V3	40 s	40.1	41.1	25.7	5.2	-0.688	0.684	
V4	~ 28 s	26.6	56.5	78.1	29.7	**-9.397**	$<$ **0.001**	↘
V5	~ 11 s	13.5	16.5	62.4	52.8	**-2.984**	**0.002**	↘
V6	54 s	53.5	54.9	19.7	11.0	-0.929	0.821	
V7	~ 23 s	24.5	29.8	54.7	35.9	**-2.341**	**0.012**	↘
V8	~ 14 s	25.0	23.0	61.1	39.4	1.108	0.137	

at play for the given combination of task composition and scheduling policy. Across the 35 students in the synchronous offering, a total of 165 comments were written (an average of 4.7 comments per student). In the case of the asynchronous offering, the 39 students collated 383 written comments (an average of 9.8 comments per student). This suggests that students may be more inclined to provide additional comments (or be reflective) when they are comfortably able to work at their own pace. Students in the synchronous offering may have felt pressured to work faster through the quiz and thereby spend less time on the optional parts of the activity.

4.2. Video Helpfulness Ratings

After watching each video, students would rate its *helpfulness* using a 5-point Likert scale as part of their *post-video quiz*. Table 3 summarizes these rating-results across both the synchronous and asynchronous versions of the activity. In addition to each video's average rating in each of the two versions ($\bar{x}_G$ and $\bar{x}_I$), the table also includes two-tailed paired Wilcoxon signed-ranks tests, in which for each of the task compositions (V1 for *Identical Coarse*, V3 for *Identical Fine*, and V6 for *Mixed*), the videos for non-*Sequential* policies are compared against those of the corresponding *Sequential* policy. The three *Sequential* videos are thus effectively used as baselines to determine whether the students find value in the parallel visualizations for the given task composition.

When looking at the helpfulness rating for each video-version, some patterns can be seen. In both the synchronous and asynchronous offerings, the *Sequential* videos (V1, V3, and V6) consistently receive the lowest ratings. When V2 introduces the first form of parallelism, the average rating is slightly higher than V1 but not enough to be statistically significant (W=43 p=0.549 for $\bar{x}_G$, and W=43 p=0.549 for $\bar{x}_I$). V2 is considered the simplest kind of parallel programming (i.e., using a *Fully Parallel* scheduling policy), with straightforward intended

Table 3: Video helpfulness rating

Video	**Synchronous**								**Asynchronous**								$\bar{x}_G$ **vs** $\bar{x}_I$	
	Rated Helpfulness						**vs seq**		**Rated Helpfulness**						**vs seq**			
	1	**2**	**3**	**4**	**5**	$\bar{x}_G$	**W**	**p-value**	**1**	**2**	**3**	**4**	**5**	$\bar{x}_I$	**W**	**p-value**	**W**	**p-value**
V1 (seq)	2	2	6	10	15	3.97	—	—	2	8	10	9	10	3.44	—	—	**859**	**0.048**
V2	0	2	8	11	14	4.06	43	0.549	3	4	7	15	10	3.64	114	0.299	806.5	0.162
V3 (seq)	1	3	11	8	12	3.77	—	—	3	9	10	8	9	3.28	—	—	833	0.094
V4	0	2	7	11	15	4.11	**0**	**0.007**	0	3	7	15	14	4.03	**28.5**	**< 0.001**	721	0.663
V5	1	2	2	10	20	4.31	**7**	**< 0.001**	1	5	5	14	14	3.90	**0**	**< 0.001**	844.5	0.060
V6 (seq)	2	3	5	10	15	3.94	—	—	4	8	8	8	11	3.36	—	—	850.5	0.060
V7	0	2	1	11	21	4.46	**7**	**0.007**	0	5	8	13	13	3.87	**21**	**0.008**	**911**	**0.008**
V8	0	3	2	11	19	4.31	**29.5**	**0.042**	0	5	6	10	18	4.05	**5.5**	**< 0.001**	769.5	0.310
Sum						32.94								29.56			**873.5**	**0.039**

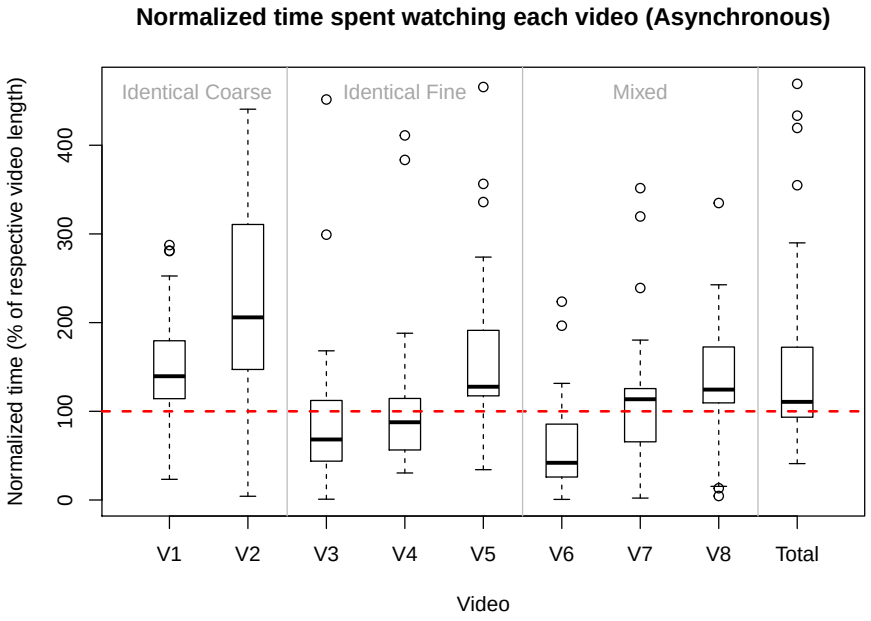
learning points. However, when students viewed video V4 (the same scheduling policy but with a different task composition), students valued the video much higher than its *Sequential* counterpart V3. The same can be said for all the other parallel videos when compared to their *Sequential* counterparts.

Finally, Table 3 also presents results for the two-tailed unpaired Wilcoxon rank-sum test comparing the averages across the different versions (i.e. $\bar{x}_G$ versus $\bar{x}_I$). The overall rating sum across all eight videos was higher for the synchronous offering (32.94/40) compared to the asynchronous offering (29.56/40), with a statistically significant difference (W=873.5, p=0.039). When each video is considered individually, the $\bar{x}_G$ version is consistently higher than the $\bar{x}_I$ version. However, this is only statistically significant for V1 (W=859, p=0.048) and V7 (W=911, p=0.008). It seems that students, in general, appreciated the value of the analogy videos more when they were guided by the instructor in the classroom. In the next section, the data collected from the asynchronous students' watch-time behaviors sheds light on what happened in that offering.

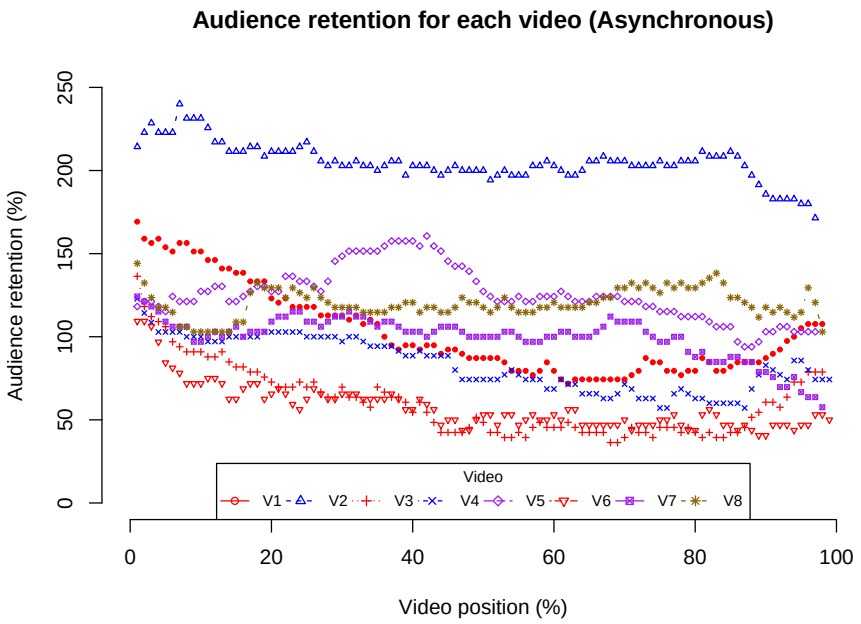
4.3. Watch Times for Asynchronous Students

In order to get a closer understanding of students' appreciation of the videos, the data depicting their engagement while watching the videos were analyzed. This analysis was only useful for the asynchronous version, where students were working at their own paces, and thus had the freedom to watch the videos as much (or as little) as they wanted between the *pre-video* and *post-video* quizzes.

Figure 5(a) presents the distribution of the times the students spent on each video. As the videos each had slightly different lengths, these times are normalized to the length of the respective video. The red dashed line represents the normalized full-video threshold (100%), which corresponds to remaining on the page long enough to watch the entire video. A value below this threshold corresponds to the student leaving the page sooner than the video's length—



(a) Distribution of time students spent on video page



(b) Audience retention at respective points of video

Figure 5: Watch-time patterns for the videos during the asynchronous course activity, including (a) spread of time spent on each video page normalized to the video's length, and (b) actual audience retention at specific positions of each video (retrieved from built-in YouTube stats).

therefore not watching it in its entirety; a value above this threshold corresponds to the student remaining on the page longer than the video's length—possibly to watch it more than once. Figure 5(b) shows finer-grained watch-time data for each video, as measured by YouTube's audience-retention data. For each video, this data shows the point at which typical students would be losing or gaining interest within the video.

Figure 5(a) reveals that more than 25% of the students did not remain on the combined video pages long enough to watch all the videos (denoted by the 25^{th} quartile of the *Total* being under 100%). Looking at a subset of the videos at a time, grouped based on their *Task Composition*, there appears to be a common "stepping" pattern. In all cases, the initial *Sequential* videos (V1, V3, and V6) are least popular in their respective grouping. V1 is the only *Sequential* video above the 100% threshold, which can most likely be attributed to the students' curiosity regarding the first video. V2, the variation on V1 using the *Fully Parallel* policy, received over 200% watch time—indicating that on average, most students watched the video twice. However, these same two scheduling policies (*Sequential* and *Fully Parallel*) did not attract as much attention when they were repeated again in V3 and V4 with different task compositions. It was only when a new scheduling policy was introduced (*Static*, in V5), that students' attention was again captured enough to watch the full video. A similar pattern again appears within the third group of videos, where the *Sequential* V6 is again the lowest (with most students watching less than half of it), and the newly-introduced parallel scheduling policy (*Dynamic*, in V8) receiving the students' highest attention.

4.4. Overall Themes of Student Feedback

The asynchronous offering of these activities concluded with two open-ended questions, to provide students with an opportunity to give feedback on (i) how

Figure 6: A word cloud generated from the open-ended student feedback, using MonkeyLearn's AI-powered WordCloud Generator [45].

the activities helped their learning, and (ii) what aspects of the activities did not work well. A total of 63 comments were made by the 39 students, presented as a word cloud in Figure 6. To synthesise the themes of these comments, a thematic analysis [46] was carried out. With the help of a qualitative data analysis software application, 30 codes were generated during the initial coding. These codes were used to tag 56 of the comments, yielding a total of 114 comment-tags, as some comments attracted multiple tags. The remaining seven comments were irrelevant, so were ignored from further analysis. Subsequent phases of the thematic analysis resulted in the eventual aggregation of the codes into four main themes. In Figure 7, these themes are depicted as the four central blue lines:

In the rest of this section, we briefly explore these four themes that emerged from the students' open-ended comments.

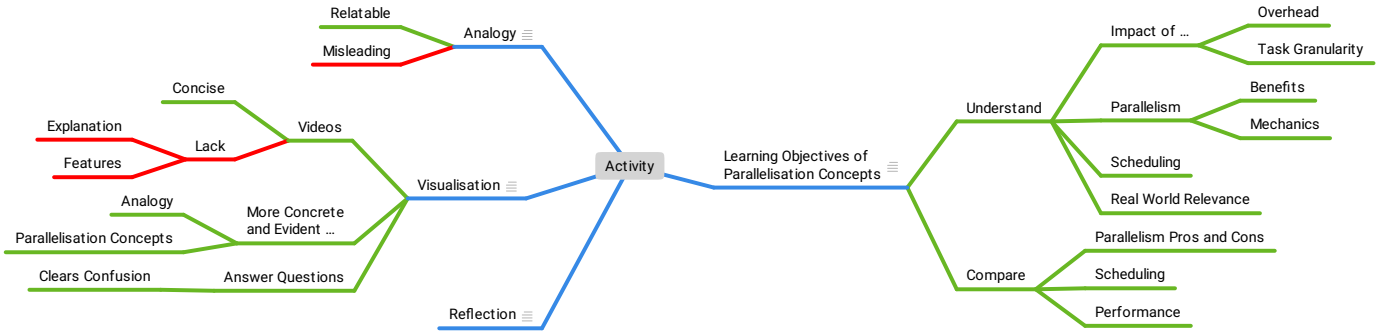


Figure 7: Mind map representing four main themes (*Analogy*, *Visualization*, *Reflection*, and *Learning Objectives of Parallelization Concepts*), represented by the blue edges. The synthesized themes were inspired by the thematic analysis process [46], applied to activity feedback from students. The themes are further decomposed into their respective facets: green edges represent positive aspects, while red edges represent negative aspects.

- **Learning Objectives of Parallelization Concepts:** In the most dominant theme, students shared how the activity helped them learn specific parallel programming concepts. These were broadly synthesized into two sub-themes, motivated by Bloom's taxonomy [47]: *Understand* and *Compare* (a form of *Analyze*). An example comment for this theme was:

  > "*Helped to understand how coarse/fine grained affected utilization which I wouldn't have guessed intuitively. The dynamic vs. static and mixed tasks also taught me something because I made the assumption that the static allocation would be the best possible, but in the real world we don't usually know how long a task will take.*"

- **Visualization:** Appreciation of the visualizations provided the next dominant theme. The overall essence of this theme related to the how the visualizations increased clarity by: (i) making abstract concepts (both the analogy and parallelization concepts) more concrete, and (ii) enabling

students to have their questions "answered" by the visualization:

> "*The visualization aspect helped me especially with the "fully parallel" workers as I did not grasp the concept that changing tasks internally within a processor in some situations would reduce the efficiency. Additionally the reinforcement in all videos was useful to make the concepts concrete.*"

While the videos were complemented for their conciseness, some students expressed their desires for richer in-video explanations and additional features (e.g., an in-video timer):

> "*There could be a bit more explaining in the videos, a lot of it is based on deductive reasoning instead of given information."*

- **Analogy:** The analogy were largely appreciated for being relatable:

  > "*I think the metaphor of workers as threads and desks as cores works well. It makes an especially good point about how one thread per task usually isn't optimal, as the threads are constantly swapping adding massive overhead."*

  But some students also pointed out they could be misleading:

  > "*The computer can calculate millions of instructions per second. Does change of a task require a few seconds?"*

  It is possible that the exaggeration of overhead in the analogy videos might have misled students to over-estimate the overhead, resulting in overly-high student time-estimates in the *post-video* quizzes.

- **Reflection:** Finally, some students valued the pre-video and post-video quizzes as providing an opportunity to be reflective:

> "*By asking the same questions twice, one has to reflect on how the concepts actually work.*"

5. Discussion

Recall that the the goal of this study was to answer the following research question:

> ***RQ:*** *To what extent do visual analogy videos help students learn fundamental parallel programming concepts?*

The results from the preceding section reveal a number of insightful lessons regarding the value these analogy videos delivered to the students.

First and foremost, the results have highlighted the inherent complexity of comprehending the runtime behavior of a parallel program in comparison to that of a sequential program. Most students could accurately predict the behavior in all three sequential scenarios, but they noticeably struggled to do so in the parallel scenarios. Even "trivial" parallel programming scheduling policies (such as *Fully Parallel* and *Static Pool*) elicited wide variations in the students' predictions. This indicates that students find sequential behavior much easier to understand than parallel behavior, and highlights the need for additional research in developing effective parallel computing learning resources for students.

When led by an instructor in a guided synchronous environment, most students performed significantly better than students who watched the videos asynchronously. Compared to their asynchronous peers, the synchronous students' post-video predictions converged more closely, and their predictions demonstrated statistically significant improvements in how closely they could estimate the target time. Some asynchronous students neglected to watch an entire video even once, but under the instructor's guidance, synchronous students were exposed to each parallel video *twice in its entirety*. The instructor being present

to cause these repeated viewings may well have contributed to synchronous students better absorption of the key learning points.

When students carried out the same activities in the asynchronous setting, the benefits of analogies were no longer observed. Student predictions still seemed to converge, but were actually further away from the correct target times. While the video-viewing data revealed that asynchronous students watched the videos less than the instructor-guided synchronous students, it is still the case that most asynchronous students watched most of the videos once (although barely). This implies that a synchronous instructor plays a valuable role in guiding students in their use of time and directing their attention to engage with the key learning points. More precisely, an instructor can help by ensuring that students do not skip over important learning opportunities, especially when they might be underestimating the complexity of the concepts (and overestimating their own understanding).

This difference in the two groups' video-viewing behaviors may also explain the synchronous students' generally-higher ratings of the videos helpfulness: watching the parallel videos twice helped students recognize and remember the key learning points. In the asynchronous version, most videos were watched only once. Many students even skipped watching some of the videos (even the parallel ones), which seems likely to have contributed to their poorer post-video quiz performance (and hence lower appreciation of the videos). This highlights that further research is needed to explore the impact of visualization tools, particularly when students are left to use them at their own pace.

In the qualitative feedback, some asynchronous students stated that they found the videos misleading and lacking the information needed for them to be used on their own. It is possible that the videos were misleading, and were unintentionally creating misconceptions. Alternatively, the watch-time data suggests

that students' misconceptions may have arisen because these asynchronous students were rushing through videos without giving them much thought or close attention. More research is needed to distinguish between these possibilities.

These observations reveal the quality of the analogies (as seen in their positive contribution to learning in the synchronous setting), but also their possible misuse that could harm learning (as seen in their negative contribution in the asynchronous setting). This is an important consideration for designers of instructional material to consider, especially as they design materials for online, asynchronous courses.

While the qualitative data was not analyzed for correctness in terms of students discussing each video's learning outcomes, it is clear that the asynchronous group wrote many more comments at each stage. Even though the asynchronous format produced poorer learning outcomes for the students (in terms of the program-performance predictions), it apparently provided a very comfortable learning environment, in the sense that students felt inclined to spend more time reflecting on the intended learning points. This suggests that future research opportunities exist to better understand how to obtain the best of both worlds: how can one blend synchronous instructor guidance with asynchronous reflective self-paced learning to obtain the benefits of both approaches?

In terms of helpfulness, students did not seem to ascribe much value to the videos demonstrating basic (trivial) forms of parallel concepts. However, they appreciated the analogy videos much more when they encountered complex scenarios in which the learning outcomes were non-trivial. This theme was also present in the qualitative feedback, where a common theme was the students' appreciation of how the videos helped them *understand* overhead and task granularity, and *compare* them across different scheduling policies.

Even though the asynchronous students watched V2 twice, they did not seem

very impressed with it compared to V1, according to the helpfulness results. This could explain why V3 and V4 were watched less and rated lower, as students felt these videos were "more of the same" (despite these videos covering the different task compositions). V2 introduced the first form of parallelism, but it was the simplest form of parallel computing: eight threads performing eight identical coarse tasks on four cores. There was little overhead and no surprising effect, so that video was appreciated less. Its score was higher than that of the sequential video, but not enough to be statistically significant. Other videos explored more complex scenarios, and were thus seen as more helpful by the students.

Threats to validity

There are a number of possible threats to the validity of this study, including:

- In the asynchronous offering, a large representation of the entire class completed the Learning Activity online, at their own pace. In the synchronous offering, students had to physically come to the classroom to complete the activity, and a smaller percentage of the class participated. These latter students might have been the more-engaged students in the class, which could skew their post-video quiz results positively, compared to their asynchronous peers. This could be a factor, but to the authors, the difference in time spent on task (i.e., two viewings of each video for every student in the synchronous group vs. not even one full viewing of some videos by some asynchronous students) seems more likely to be the primary factor contributing to the difference in the two groups' performances.

- In the asynchronous offering, the video's watch-time measurements could be inaccurate— students could have been doing other things (i.e., "multitasking") while "watching" the videos and answering the quiz questions,

rather than being fully engaged with the exercise. However, YouTube's retention statistics indicate that in the asynchronous offering, many students answered questions without watching all the videos adequately. In the feedback, a few students even mentioned that they accidentally pressed "next" in the quiz before clicking to open the video link. The quiz did not allow students to "go back", as the asynchronous version of the activity was designed to replicate the synchronous version as closely as possible.

- Students could have based their post-video quiz answers on the (wall-clock) time taken by the computation in the videos. This is a possibility, but it seems unlikely, given how the accuracy of the synchronous students' post-video quiz time-estimates increased, but the asynchronous students' accuracy decreased. If it were a factor, it should affect both groups equally.

6. Conclusions

Parallel and distributed computing (PDC) is notoriously more difficult to understand than traditional sequential computing, which is why it has traditionally been a specialized elective for higher-level computing courses. The changing landscape of computing technology has seen increasing curricular efforts to incorporate PDC concepts into the earlier core computing courses. This study describes an easily-accessible activity, merging the pedagogical benefits of analogies and visualizations. Both synchronous and asynchronous evaluations are presented, to better understand the implications for other instructors seeking to incorporate such an activity. These evaluations first and foremost illustrate the struggle students have in conceptualizing parallel computing in comparison to sequential computing. In addition, this study has presented an extensive quantitative and qualitative analysis of the learning impact, engagement, and student-perceived helpfulness of the activity.

The materials related to this study are all publicly available, including the videos[1], the asynchronous self-guided form[2], the synchronous self-guided form[3] related to this study, plus the general app and its related material[4]. It is hoped that not only will these materials be of immediate value to the PDC education community, but that the general approach taken will encourage others to create and share similar PDC teaching resources. We envision this to be in the form of a publicly-available website where instructors can find and share videos and related materials that are suitable for teaching particular concepts in their courses. Such future work will be especially valuable if the instructional materials incorporate best-practice guidelines for the creation of effective learning videos [17], including the use of signaling to highlight important information and aligning analogy elements to the key concepts being taught. Incorporating interactivity will generally improve engagement by giving students more control [17].

There is additional work to be done in identifying the optimal pedagogical design for carrying out such learning activities. This work includes the analysis of an activity that incorporates an active learning component and compares its impact to the activity described in this paper. For example, rather than using videos, conducting a controlled study that instead uses a different pedagogical approach (e.g., peer instruction) to see if such an approach improves the learning of these same PDC concepts. Such a study could be further modified to an activity that combines *both* videos and the other pedagogical approach, to see if such an approach could eliminate any potential misconceptions the analogy-videos might produce for the students.

[1] `https://www.youtube.com/playlist?list=PLTniUCm8Xpapy0IlV-tRrBD0IWD2vlyZ4`
[2] `https://auckland.au1.qualtrics.com/jfe/form/SV_eQGC7iTHWbIORhP`
[3] `https://auckland.au1.qualtrics.com/jfe/form/SV_8OzlUKm7Il9SLit`
[4] `https://parallel.auckland.ac.nz/education/parallelar`

References

[1] M. Sahami, A. Danyluk, S. Fincher, K. Fisher, D. Grossman, E. Hawthorne, R. Katz, R. LeBlanc, D. Reed, S. Roach, E. Cuadros-Vargas, R. Dodge, R. France, A. Kumar, B. Robinson, R. Seker, A. Thompson, Computer science curricula 2013: Curriculum guidelines for undergraduate degree programs in computer science, Association for Computing Machinery (ACM)-IEEE Computer Society (2013).

[2] S. K. Prasad, A. Chtchelkanova, M. G. F. Dehne, A. Gupta, J. Jaja, K. Kant, A. L. Salle, R. LeBlanc, A. Lumsdaine, D. Padua, M. Parashar, V. Prasanna, Y. Robert, A. Rosenberg, S. Sahni, B. Shirazi, A. Sussman, C. Weems, J. Wu, NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributed Computing - Core Topics for Undergraduates, Version 1, `http://www.cs.gsu.edu/~tcpp/curriculum` (2012).

[3] S. Ghafoor, D. W. Brown, M. Rogers, Integrating parallel computing in introductory programming classes: an experience and lessons learned, in: European Conference on Parallel Processing, Springer, 2017, pp. 216–226.

[4] Y. Ko, B. Burgstaller, B. Scholz, Parallel from the beginning: The case for multicore programming in thecomputer science undergraduate curriculum, in: Proceeding of the 44th ACM technical symposium on Computer science education, 2013, pp. 415–420.

[5] T. R. Gross, Breadth in depth: a 1st year introduction to parallel programming, in: Proceedings of the 42nd ACM technical symposium on Computer science education, ACM, 2011, pp. 435–440.

[6] B. Moskal, D. Lurie, S. Cooper, Evaluating the effectiveness of a new instructional approach, in: Proceedings of the 35th SIGCSE technical symposium on Computer science education, 2004, pp. 75–79.

[7] A. Luxton-Reilly, I. Albluwi, B. A. Becker, M. Giannakos, A. N. Kumar, L. Ott, J. Paterson, M. J. Scott, J. Sheard, C. Szabo, Introductory programming: a systematic literature review, in: Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education, 2018, pp. 55–106.

[8] P. Thagard, Analogy, explanation, and education, Journal of research in science teaching 29 (6) (1992) 537–544.

[9] P. Simons, Instructing with analogies, Journal of Educational Psychology 76 (3) (1984) 513.

[10] A. G. Harrison, D. F. Treagust, Teaching and learning with analogies, in: Metaphor and analogy in science education, Springer, 2006, pp. 11–24.

[11] T. L. Naps, G. Rößling, V. Almstrum, W. Dann, R. Fleischer, C. Hundhausen, A. Korhonen, L. Malmi, M. McNally, S. Rodger, J. A. Velazquez-Iturbide, Exploring the role of visualization and engagement in computer science education, in: ACM SIGCSE Bulletin, Vol. 35, 2002, pp. 131–152.

[12] A. G. Harrison, The affective dimension of analogy, in: Metaphor and analogy in science education, Springer, 2006, pp. 51–63.

[13] J. Lim, H. H. Pellett, T. Pellett, Integrating digital video technology in the classroom, Journal of Physical Education, Recreation & Dance 80 (6) (2009) 40–55.

[14] M. L. Khan, K. Richards, M. L. Wu, Understanding the effectiveness of video-based instruction versus text-based instruction, in: EdMedia+ Innovate Learning, Association for the Advancement of Computing in Education (AACE), 2010, pp. 3638–3646.

[15] C. Gaudin, S. Chaliès, Video viewing in teacher education and professional development: A literature review, Educational Research Review 16 (2015) 41–67.

[16] M. Holtzblatt, N. Tschakert, Expanding your accounting classroom with digital video technology, Journal of Accounting Education 29 (2-3) (2011) 100–121.

[17] C. J. Brame, Effective educational videos: Principles and guidelines for maximizing student learning from video content, CBE Life Sciences Education 15 (4) (2016).

[18] P. J. Martínez, F. J. Aguilar, M. Ortiz, Transitioning from face-to-face to blended and full online learning engineering master's program, IEEE Transactions on Education 63 (1) (2019) 2–9.

[19] I. Blanchette, K. Dunbar, How analogies are generated: The roles of structural and superficial similarity, Memory & cognition 28 (1) (2000) 108–124.

[20] R. J. Spiro, Multiple analogies for complex concepts: Antidotes for analogy-induced misconception in advanced knowledge acquisition, Center for the Study of Reading Technical Report (1988).

[21] T. R. Colburn, G. M. Shute, Metaphor in computer science, Journal of Applied Logic 6 (4) (2008) 526–533.

[22] J. P. Sanford, A. Tietz, S. Farooq, S. Guyer, R. B. Shapiro, Metaphors we teach by, in: Proceedings of the 45th ACM technical symposium on Computer science education, 2014, pp. 585–590.

[23] J. Sajaniemi, P. Byckling, P. Gerdt, Animation metaphors for object-oriented concepts, Electronic Notes in Theoretical Computer Science 178 (2007) 15–22.

[24] Y. Cao, L. Porter, D. Zingaro, Examining the value of analogies in introductory computing, in: Proceedings of the 2016 ACM Conference on International Computing Education Research, 2016, pp. 231–239.

[25] J. Hidalgo-Céspedes, G. Marín-Raventós, V. Lara-Villagrán, L. Villalobos-Fernández, Effects of oral metaphors and allegories on programming problem solving, Computer Applications in Engineering Education 26 (4) (2018) 852–871.

[26] T. Clear, The nature of cognition and action, ACM SIGCSE Bulletin 29 (4) (1997) 25–29.

[27] S. Schez-Sobrino, M. Á. García, C. Gómez, D. Vallejo, A. I. Molina, C. Lacave, C. Glez-Morcillo, J. A. Albusac, M. Á. Redondo, Angela: a novel approach of graphic notation based on the metaphor of road signs to facilitate the learning of programming, in: Proceedings of the Seventh International Conference on Technological Ecosystems for Enhancing Multiculturality, 2019, pp. 822–829.

[28] T. Tanielu, R. 'Akau'ola, E. Varoy, N. Giacaman, Combining analogies and virtual reality for active and visual object-oriented programming, in: Proceedings of the ACM Conference on Global Computing Education, 2019, pp. 92–98.

[29] G. Domik, Do we need formal education in visualization?, IEEE Computer Graphics and Applications 20 (4) (2000) 16–19.

[30] D. W. Brown, V. Ford, S. K. Ghafoor, A framework for the evaluation of parallel and distributed computing education resources, in: EduPar 2020: 10th Workshop on Parallel and Distributed Computing Education, IEEE, 2020.

[31] S. Matthews, PDCunplugged: A free repository of unplugged parallel & distributed computing activities, in: EduPar 2020: 10th Workshop on Parallel and Distributed Computing Education, IEEE, 2020.

[32] A. Goncharow, A. Boekelheide, M. Mcquaigue, D. Burlinson, E. Saule, K. Subramanian, J. Payton, Classifying pedagogical material to improve adoption of parallel and distributed computing topics, in: 2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), IEEE, 2019, pp. 312–319.

[33] S. V. Moore, S. R. Dunlop, A flipped classroom approach to teaching concurrency and parallelism, in: 2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), IEEE, 2016, pp. 987–995.

[34] N. Giacaman, S. Kalra, O. Sinnen, The active classroom: students and instructors parallel programming... in parallel, in: Proc. of 5th NSF/TCPP Workshop on Parallel and Distributed Computing Education (EduPar-15), in conjunction with IPDPS, 2015.

[35] J. C. Adams, P. A. Crain, C. P. Dilley, C. D. Hazlett, E. R. Koning, S. M. Nelesen, J. B. Unger, M. B. V. Stel, TSGL: A tool for visualizing multithreaded behavior, Journal of Parallel and Distributed Computing 118 (2018) 233–246.

[36] A. Danner, T. Newhall, K. C. Webb, Paravis: A library for visualizing and debugging parallel applications, in: 2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), IEEE, 2019, pp. 326–333.

[37] J. Maloney, M. Resnick, N. Rusk, B. Silverman, E. Eastmond, The Scratch

programming language and environment, ACM Transactions on Computing Education (TOCE) 10 (4) (2010) 1–15.

[38] A. Feng, M. Gardner, W. C. Feng, Parallel programming with pictures is a Snap!, Journal of Parallel and Distributed Computing 105 (2017) 150–162.

[39] M. Abernethy, O. Sinnen, J. Adams, G. De Ruvo, N. Giacaman, ParallelAR: An augmented reality app and instructional approach for learning parallel programming scheduling concepts, in: 2018 IEEE International Parallel and Distributed Processing Symposium Workshops, IEEE, 2018, pp. 324–331.

[40] N. Giacaman, O. Sinnen, Ea: Research-infused teaching of parallel programming concepts for undergraduate software engineering students, in: 2014 IEEE International Parallel & Distributed Processing Symposium Workshops, IEEE, 2014, pp. 1099–1105.

[41] M. J. Rubin, The effectiveness of live-coding to teach introductory programming, in: Proceeding of the 44th ACM technical symposium on Computer science education, 2013, pp. 651–656.

[42] A. G. S. Raj, J. M. Patel, R. Halverson, E. R. Halverson, Role of live-coding in learning introductory programming, in: Proceedings of the 18th Koli Calling International Conference on Computing Education Research, 2018, pp. 1–8.

[43] N. Giacaman, O. Sinnen, Preparing the software engineer for a modern multi-core world, Journal of Parallel and Distributed Computing 118 (2018) 247–263.

[44] C. H. Crouch, E. Mazur, Peer instruction: Ten years of experience and results, American journal of physics 69 (9) (2001) 970–977.

[45] MonkeyLearn, WordCloud Generator, https://monkeylearn.com/word-cloud (2021).

[46] V. Braun, V. Clarke, Using thematic analysis in psychology, Qualitative research in psychology 3 (2) (2006) 77–101.

[47] D. R. Krathwohl, L. W. Anderson, A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives, Longman, 2009.