Building a Motivating and Autonomy Environment to Support Adaptive Learning

Qiong Cheng
University of North Carolina at Charlotte
qcheng1 @uncc.edu

David Benton
University of North Carolina at Charlotte
dbento10@uncc.edu

Andrew Quinn
University of North Carolina at Charlotte
aquinn16@uncc.edu

***Abstract*— This Innovative Practice Full paper presents an effort on fostering student motivation and autonomy in support of adaptive learning. Higher education has been transformed by digital technology to be more interactive, self-paced, and adaptive to students. But this technology presupposes that students possess autonomy and are innately well motivated. Unfortunately, many students entering college lack motivation and self-control. These dispositions retard self-paced adaptive learning and limit the effectiveness of imparting improved adaptability in this technology. Taking inspiration from the self-determination theory, we investigated, in the context of adaptive learning, the importance of nurturing student autonomy and enhancing student situated motivation. In this paper, we present our experiential study in a gateway core course of computer science, in which adaptive preparation and learning pedagogy has been adopted for four semesters, one semester without motivating support and others with this support. By comparing and analyzing student engagement and performance over these four semesters, we observe that nurturing student autonomy and enhancing motivation are critical factors in maximizing the effectiveness of adaptive learning.**

Keywords*— *data structures and algorithms, growth mindset, self-determination theory, adaptive learning, active learning

I. Introduction

With the Computer Science for All initiative prospering, more and more states started bringing computing into primary and secondary schools [36]. It was observed that students who have been exposed to computing prior to college are more likely to perform well in both introductory and advanced computer science courses [35]. However not every college-entering student had this experience. Some had never participated in any computing-related activities [6-8].

Even when students have taken computing classes before entering college, programming languages and/or requirements widely vary among schools and institutions. Transfer students often possess different background knowledge or prior programming experience. Within this diverse range of entrants is a wide range of individual self-efficacy and self-determination levels along with programming and problem-solving skills. There is evidence that many students possess lower degrees of autonomy and skills [26].

In a traditional one-size-fits-all context, increasing disparities among students lead to the alienation of the struggling students while boring those who are more experienced. It presents challenges to those programming core courses and their instructors. At issue is the appropriateness of the one-size-fits-all pedagogical model.

An alternative is the adaptive learning model. This learning model offers customized learning experiences related to need. By meeting individual student needs, the adaptive learning model aims at helping all students to attain mastery of the content of a course.

Existing computer-assisted adaptive software systems offer an array of adaptability. They utilize artificial intelligence and machine learning to mine student needs and employ decision-making technology to execute adaptability. However, these systems presuppose that students are well motivated and ready to take responsibility for their own learning [22] which implies that students possess autonomy, and are sufficiently self-disciplined to manage their time and behaviors.

Unfortunately, many students entering computer science classes possess low self-efficacy and lack internal control and purposeful motivation, easy to be distracted. As motivation theory and learning psychology [21-23] inform, active engagement and self-directed learning cannot be coerced. Students must choose to learn. When students possess autonomy, they are well motivated to learn and succeed.

The growing student disparities along with the implicit underlying assumptions of student capacities made by adaptive learning technology create challenges for both instructors and students. This brings forth barriers to the collaboration between these two parties working towards developing a positive and effective learning community environment through a semester in a course, where concept understanding is shared.

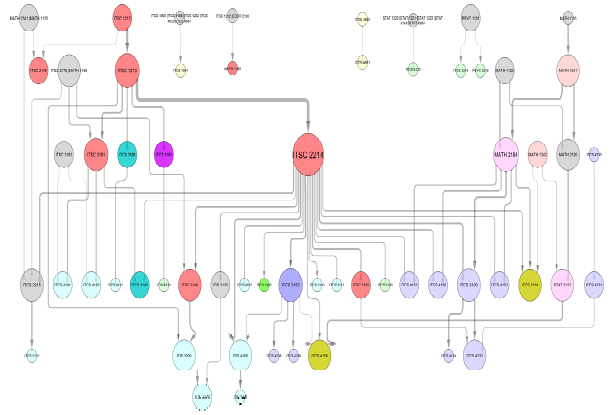
We started adopting the self-paced adaptive learning preparation and active learning pedagogy in 2019. The percentage of students who finished all preparation work of a module varied, the lowest being 39%, the highest 79%; more than 20% of students in the last two modules shown no preparation work at all. The unsatisfactory student engagement in fall 2019 increased the variation of knowledge levels and negatively impacted instructors' decision-making process. Many students felt left behind and were frustrated, due to poor pacing. These combined difficulties called for instructors to prepare students not only technically but also mentally. It is necessary to train students to use the technology and guide them to be self-directed [23].

In responding to these emerging requirements, we drew inspiration from self-determination theory and proceeded to explore the nurturing of student autonomy and enhancement of motivation. We busied in enriching the class with activities and instructions, providing hands-on training, motivating, and guiding students to learn. Our hypothesis was that increasing student autonomy and motivation could help promote student engagement in the adaptive learning preparation work and further improve student performance.

In this paper, we present our experimental study in a core gateway course CS2/DS. In investigating the effectiveness of the innovative practice, we measured the students' deployment of time and effort in adaptive preparation work. We also administrated for the same group of students before and after each module text-based reflections along with two self-efficacy and belongingness surveys respectively in the beginning and end of a semester. The statistical significance

of the survey results indicated that 1) the practice of increasing student autonomy and motivation improved student engagement in self-paced preparation work, and 2) it promoted student performance, increased student interest, sustained self-confidence, and strengthened a sense of belongingness.

The rest of the paper describes our pedagogical methodology and autonomy learning practice (section 3) after a primer on our academic context (Section 2). We then evaluate our exploration in section 4. Section 5 discusses related comments and issues that require further investigation.



departmental course map in our first day of class. In the course map, each circle corresponds to a course and each arrow connecting two courses corresponds to course dependency. ITSC 2214 is the CS2/DS course ID, which is prerequisite to at least 18 advanced computer science courses.

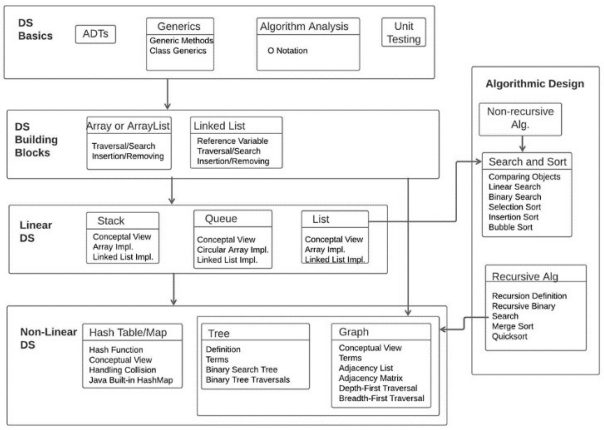


Fig. 2 Learning concept map in our CS2/DS. This course contains ten modules: generics and ADTs, algorithmic analysis, queues, stacks, lists, search and sorting, recursion, trees, hash tables/maps, and graphs.

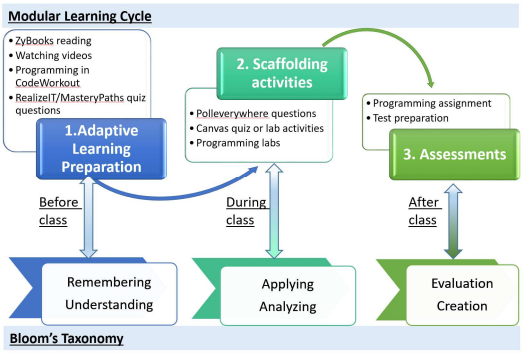


Fig. 3 Modular learning cycle fits with the revised Bloom's Taxonomy.

II. Academic Context

Data Structures and Algorithms (CS2/DS) is a core gateway course that plays a critical and intermediary role in computer science [31]. This course is required for all majors and most minors in the College of Computing and Informatics (CCI) and is a prerequisite to almost all of our senior-level courses (see Fig.1). It is the next step for students who wish to continue after finishing two-semester fundamental programming courses (CS1). All students who complete the course with an A, B, or C grade can move forward to their senior-level courses.

This course is broad in topic coverage (see Fig. 2) and sufficiently rigorous in computational logic and reasoning. It introduces the key conceptual views of diverse data structures, their design and implementations, and fundamental algorithm analysis and design strategy. It helps students to boost problem-solving and programming skills and prepares them to delve deeper into computing.

In this course, we use JAVA as the programming language. In a regular session, more than half of our students (58.2%) were transfer students; approximately half of the students had jobs; around 83.3% of the students were male.

We adopted the adaptive learning pedagogical strategy for four semesters. In fall 2019 and spring 2020, it was scheduled as three 50-minute face-to-face lectures per week. In spring 2020, we were transitioned to synchronous distance learning in the ninth week due to COVID-19. Our second exam (midterm exam) was accordingly moved to online synchronous mode. The university informed students later that they would be able to replace any letter grade that was a D or higher with a Pass, which would allow for progression but would not affect GPA. In fall 2020 and spring 2021, it has two 75-minute online synchronous classes per week. Pass/no grade policy was still valid but students should have a C or higher to pass the class.

Our course has been organized with three tests and ten modules. The preparation work for each module has been launched in an adaptive learning platform, named Realize^{IT} [9], which integrates with CodeWorkout and ZyBooks. ZyBooks [10-12, 33] functions as an interactive textbook, CodeWorkout [14] is an online open system for students to practice small programming problems, and students read lecture notes, watch short videos, and answer quiz question(s) on Realize^{IT} (see Fig. 5).

Each module follows a common learning cycle, which fits with Bloom's taxonomy (see Fig. 3):

1. Before class, students work on preparation work for a course module. This adaptive platform customizes a learning path and differentiates preparation assignments among students, based on the student's knowledge determination. Accordingly, the workload in preparation assignments may vary from one student to another.

2. In class, we briefly discussed student's performance with the preparation work (see Fig. 5) and had a short quiz launched on the PollEverywhere [34], followed with a conceptual view discussion, and then we held in-class lab activities. Depending on the modules, we may have two to three in-class sessions.

3. After class, students were required to independently work on programming assignment(s) or take a test to assess their

understanding, according to Bloom's taxonomy from low-order to high-order.

III. PEDAGOGICAL METHODOLOGY

A. Self-determination Theory: Motivation & Growth Mindset

Motivation, defined as a psychological or emotional term, explains or influences one's behaviors. There are many personal or social factors impacting motivation, such as attitudes, beliefs, intentions, and efforts. All these factors change motivation with a varying rational level over time in cycles: motivation affects behaviors, behaviors drive performance, performance accordingly tunes motivation [22].

An early theory of motivation was Abraham Maslow's hierarchy of needs. Maslow claimed that our behaviors are driven by a five-tier model of human needs within a pyramid [22], where needs in a lower tier of the hierarchy must be satisfied before one can pursue needs in a higher tier. This schema suggests we need to understand individual student needs if we are to construct a motivational environment.

Two other needs theories are Frederick Herzberg's two-factor theory and Alderfer's ERG theory [29]. Frederick Herzberg claimed that "respect for me as a person" is one of the top motivating factors. Alderfer categorized people's needs in a broad approach, claiming that three groups of core needs: existence, relatedness, and growth, which continue to grow when we progress.

Vroom's "expectancy theory" explains the role of self-discipline involved in the pursuit of a personal goal. The drive theory states that a desire or interest is the main propellant of a goal. Leon Festinger's cognitive dissonance theory explained the discomfort and efforts involved in change when our worldviews differ from our personal feelings and actions.

In the self-determination theory [22, 32], motivations are divided into intrinsic (internal) motivation and extrinsic (external) motivation.

Extrinsic motivation involves action propelled by an external drive. The extrinsic motivation could be controlled by an external/internal award or punishment, value or importance that a student has identified, or goal meeting-based regulation. The first control needs more external regulations but they are instantaneous while the second and third regulations occur among more autonomous students, and they may shift in an individual student when the goal or value changes.

Intrinsic motivators can be key to a student's achievement, in that it is an intrinsic desire that drives a student to engage, learn, and grow [24, 25]. Intrinsic motivation is inherent, referring to innate needs for interest, enjoyment, or satisfaction. Intrinsic motivation arises from full satisfaction of three psychological needs: competencies, autonomy, and relatedness.

Inner competence is the ability to learn and succeed. The relatedness is a social factor that drives students to learn. Student autonomy has been defined as a combination of skills, knowledge, and beliefs that motivates and regulates a student to take the responsibility of controlling his/her learning, such as establishing a learning goal, pacing time, and executing autonomous actions needed to reach the goal. The most important elements in the development of a student's autonomy are time management, self-efficacy, locus of internal control, self-perceived competence, and motivation [29 -32].

According to self-determination theory [22, 32], students achieve self-determination when their needs for competence, relatedness, and autonomy are fulfilled. In contrast to a fixed self-determination mindset that impairs the attainment of a student's potential, a growth mindset is a concept developed by Carol Dweck. It is argued that in a growth mindset all skills, including but not limited to computing skills and self-determination skills, can be learned through guided effort and practice. Accordingly, intrinsic motivation can be learned and transformed.

According to self-determination theory, instructor attitudes influence student attitudes and behaviors. A motivating and autonomy support environment is essential in promoting or explaining the motivation and emotions of students. In order to maximize the effectiveness of adaptive and active learning, we focused on an understanding of the average student's needs in all phases of their course learning and in building a motivating context for students that would reinforce their perseverance in their learning.

B. Building a Motivating Context of Supporting Adaptive Learning

We used the same adaptive learning pedagogy and modular learning cycles (see Fig. 3) for the four semesters, Fall 2019, Spring/Fall 2020, and Spring 2021. Over the last three semesters, we built and maintained a motivating context of nurturing student autonomy and enhancing motivation in adaptive learning.

To encourage student autonomy, we upgraded our canvas course with timely updated schedules, signals of growth mindset, and links to sources that built satisfactory study habits and effectively managing time (see Fig. 4). With the course progressing, we emphasize increasing students' situated motivation and growth mindset through our modular learning cycle (see Fig. 3). This canvas version has been adopted by four-course sessions involving around 310 students taught by four instructors in spring 2021.

Our motivating interventions started at the very beginning of our first day of class, when we built relatedness by introducing the IT job market with a highlight on top ten IT jobs, their relationship with the course, and the role that CS2/DS plays in our departmental course map. The entry-level salary report was an extrinsic motivation for most of our students. Viewing from the course map in Figure 1, a student can tell that ITSC 2214 (CS2/DS) is a core gateway course and prerequisite to at least 18 senior-level computer science courses and that this class is critical to our computer science curriculum. We expanded instruction and lab activities, designed a mini-project to mimic the card game War, and for a more convenient cell phone-based daily messaging we adopted the tool, Remind App. We also used a community-based discussion platform, Piazza [28] (in 2019 and 2020) and campuswire.com (in 2021).

These efforts spanned across a semester. We then built a motivating environment involving the signaling of the growth mindset, autonomy, and relatedness at different points of our modular learning cycle. All were intended to support adaptive and active learning in our pedagogy. These positive-thinking signals happened in all phases of the learning cycle:

Students were motivated by an introduction of a new module, where we demonstrated the relationship of the upcoming new module with other learned modules by giving real-life examples. This helped students to recognize and build upon the relevance. These demonstrations included: card games, queues, and stacks, which were demonstrated by a

Hanoi Tower toy, and circuit toys to display the references to linked nodes, trees, and graphs. Such motivating interventions took around 10 minutes in a face-to-face class while in remote learning we conveyed via canvas announcements or emails. Such interventions stimulated the interest of students and fortified their determination to complete the self-paced pre-class preparation. It is our intuitive understanding that the more that students are interested in and relate to the material, they become more assiduous.

During class, students engaged in higher cognitive levels of learning through scaffolding activities and instructions. Under the guidance of instructors, students closed their gaps in knowledge and got hands-on experience in designing and implementing different data structures through mini-labs. In short, our intuitive understanding is that: 1) the more students understand, the more they feel related to learning and easily get excited about learning when facing appropriate challenges, and 2) the more they are excited, the harder they work.

After class, students either worked on independent programming assignments or prepared for tests. They were allowed to review or remediate their work on Realize^{IT}. TA help sessions were open to students who had questions. Our interventions stress the belief that the harder students work, the more they achieve. Students were asked to complete a reflection before or after each test to discuss how to improve their next test.

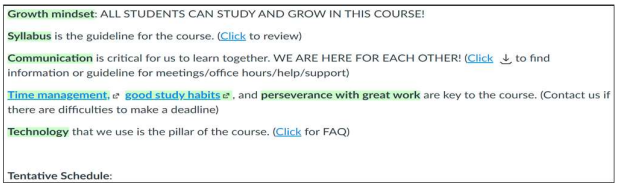


Fig. 4 A partial snapshot of course homepage on canvas.

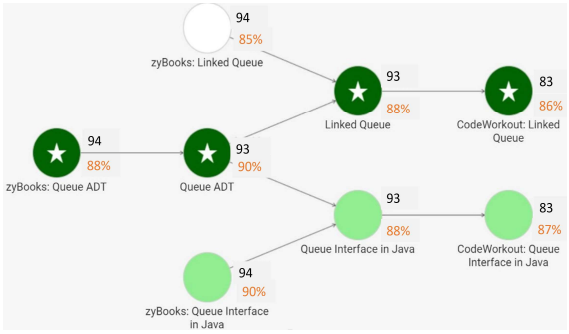


Fig. 5 Student performance in a preparation work example (Module 5 in Spring 2021, including 94 enrolled students). The circles correspond tasks and arrows correspond to task dependency. There are two numbers associated with each circle. The numbers in black represent number of students who finished the task and the percentages in orange represent average composite score among students who finished the task.

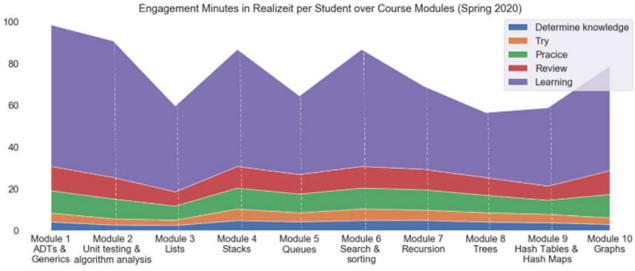


Fig. 6 Landscape of engaging minutes on Realize^{IT} over these ten modules per student in Spring 2020 (F-test p-value =0.0001). Engagement in external tools is not included. Student spent time in determining knowledge, practicing, reviewing, and learning.

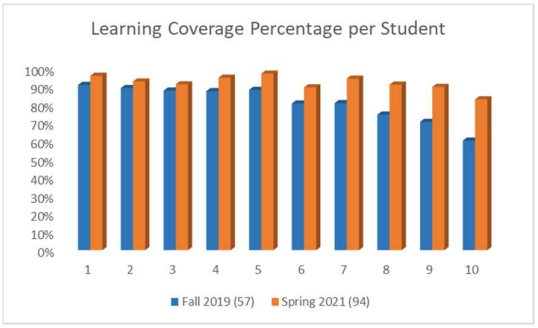


Fig. 7 Learning coverage percentage per student on Realize^{IT} in Fall 2019 versus Spring 2021.

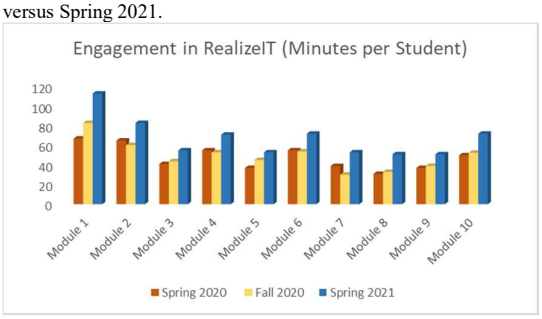


Fig. 8 Engaging minutes per student for each module on Realize^{IT}. The average engaging minutes among all ten modules per student is 48 minutes in Spring 2020, 49 in Fall 2020, and 69 in Spring 2021. In Spring 2021, students should get at least C to pass this course but other semesters allowed a D to pass, which might change student goals and explain why more minutes were spent on average in Spring 2021.

	Learning Covered Percentage Per Student				Percentage of Students who Finished all			
	Fall 2019 (57)	Spring 2020 (76)	Fall 2020 (88)	Spring 2021 (94)	Fall 2019 (57)	Spring 2020 (76)	Fall 2020 (88)	Spring 2021 (94)
Module 1	91%	97%	87%	96%	79%	83%	77%	87%
Module 2	89%	99%	92%	93%	72%	95%	90%	94%
Module 3	88%	92%	89%	91%	68%	76%	73%	89%
Module 4	87%	95%	79%	95%	61%	88%	73%	84%
Module 5	88%	94%	81%	97%	68%	87%	74%	87%
Module 6	80%	89%	77%	89%	54%	70%	61%	76%
Module 7	81%	85%	85%	94%	61%	74%	74%	88%
Module 8	74%	80%	81%	91%	39%	63%	74%	72%
Module 9	70%	80%	70%	90%	51%	67%	61%	85%
Module 10	60%	69%	66%	83%	39%	45%	55%	60%
Average	81%	88%	81%	92%	59%	75%	71%	82%

Table 1 Student engagement over course modules on Realize^{IT}. In a motivating context that supports student autonomy in 2020 and Spring 2021, higher percentage of students finished all task nodes, and each student finished more percentage of task nodes on average. Student enrollment has consistently increased: 57 students in Fall 2019, 76 in Spring 2020, 88 in Fall 2020, and 94 in Spring 2021.

While quantifying student engagement effort, we measured the amount of learning that was covered, and how many students completed the task nodes on average in a module concept map. We then examined the percentage of students who finished all task nodes in a module as well as the percentage of learning coverage per student among all these four semesters (see Table 1 and Fig. 7). We observed that on average, within a motivating context, there were 23% more students who finished all task nodes and prepared all learning concepts before class in Spring 2021 than Fall 2019, during which the motivating environment was absent; and there were about 11% more learning covered by each student in the class. The analysis indicates that 1) a higher percentage of students finished all task nodes in an autonomous and motivating support environment as against those studying without this motivating environment, and 2) each student on average in the

motivating environment completed a higher percentage of the task nodes.

Furthermore, we extracted the engagement time in the Realize^{IT} platform (see Figure 6 and 8) and observed that due to the different workloads and levels of difficulty of each module the engagement time varies over modules and that students spent more time in Spring 2021 on preparation than in Spring and Fall 2020. One of the reasons why this happened is that students' goals changed since it is required to get at least a C to pass this course in Spring 2021.

C. *Student Behaviors & Performance*

Our early analysis on student exam performance was conducted for Fall 2019 and Spring 2020. We compared multi-choice exam questions from these two semesters, grouped them by assessed topics on each module, and evaluated student performance on each common assessed topic. We observed that students from Spring 2020 performed significantly better than Fall 2019 and had a better mastery of course content. On average, 80% of first attempts correctly answered questions in Spring 2020 whereas only 50% in Fall 2019. These questions were not exactly identical, but the questions asked on the 2020 exams were generally more rigorous which made the success of 2020 students even more impressive.

Students' performance in such tests indicated that adaptive learning facilitated higher test scores. We observed that in general, 1) the higher the Realize^{IT} score, the higher the test score, and 2) The higher the test score, the less time spent in Realize^{IT} (see Figure 9).

Adaptive learning requires adaptive teaching and interaction with students. Based on the Realize^{IT} preparation performance and test scores (see Figure 10), we clustered students according to their behavior patterns and communicated to those students in clusters who exhibited similar behavior patterns. The motivating message varied from cluster to cluster, highlighting each group's strengths and providing encouragement to work on weaker areas.

We conducted experiments for each test and evaluated the statistically significant difference individually on test scores, non-programming question scores, and programming question scores between diverse student cohorts by a t-test. In the experiment on Realize^{IT} data, we were interested in two student groups: 1) motivated and diligent students who spent above-average time and scored above average, and 2) unmotivated students who spent below-average time and scored below average.

We hypothesized that the first group would score much better on multiple-choice questions than those unmotivated students who rushed through Realize^{IT} and prepared poorly for class (see Table 2). When we compared the multiple-choice questions of these two groups in the final exam via a t-test, we observed that the difference is significant. The same significant performance difference was also observed on the programming questions.

D. *Measuring Student Perception*

We measured student perceptions through 1) confidence and belongingness surveys respectively at the beginning and end of the semester, which are in the format of 5-point Likert scale questions and written comments, and 2) the reflections on learning and lessons learned.

As inspired by the learning theory of practice community [1-2], we believe that students are more highly motivated when they develop a sense of belonging to the institution or course, construct learner identities, make determined efforts to learning, and subsequently, they gain confidence [31]. Belongingness has been conceived as a student's sense of being accepted, valued, included, and encouraged by teachers and peers, along with a sense of self and that they are an important part of the life and activity of the class [31].

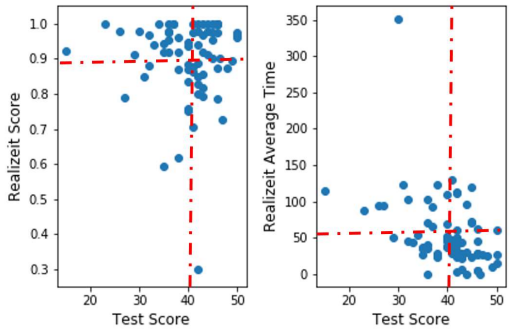


Fig. 9 Scatter plot of average work minutes versus average knowledge score in Realize^{IT} (Red dash corresponds to the average results). Each data point corresponds to a student record in test 1 (Spring 2020).

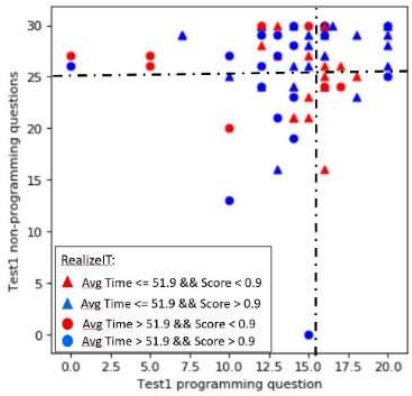


Fig. 10 Programming performance versus non-programming performance in test 1. Each data point corresponds to one student record. Triangles (/circles) corresponds to those students who spent less (/greater) than 51.9 average minutes in a module preparation work in Realize^{IT}. Red (/blue) corresponds to a Realize^{IT} knowledge score of less (/greater) than 0.9 (Spring 2020).

	Average score of unmotivated students	Average score of motivated & diligent students	P-value of t test
Non-programming questions	41.6	46.3	0.012
Programming questions	34.0	42.8	0.019
Total score	76.6	90.1	0.034

Table 2: Statistically significant difference in test performance between two groups of students (final exam). Motivated and diligent students performed better than unmotivated students (Spring 2020).

Survey results from consenting students demonstrated that the adaptive learning pedagogy assisted in increasing the interest and strengthening the student's sense of belonging: 84% felt more interested in working on programming questions as a result of the course; 81% felt more interested in working on a challenging program; 72% believed that the selection of their field of study was wise and compatible with their attributes; 94% felt the field of Computer Science was welcoming of them; 75% felt this was where he or she belonged. Additionally, through ANOVA analysis we compared confidence results before and after the course was

taken and discovered that our students' levels of confidence were significantly sustained.

IV. CONCLUSION

In our study of computer science learning, we nurtured student autonomy and enhanced motivation as a means of supporting self-paced adaptive learning and active learning. We executed the confidence and belongingness surveys at the beginning and end of the semester, and the students also filled other two reflection surveys. Our analysis revealed that 1) student autonomy assists in raising active engagement in self-paced preparation work, 2) it improves performance as well as increasing interest in learning, sustains confidence, and strengthens belongingness in the discipline of computer science. These observations were encouraging. Our interpretation is that adaptive coursework in the context of situated motivation served to stimulate students to learn at their own pace and achieve mastery of course content, while at the same time preventing slower students from feeling that they had fallen behind and stimulating the faster learners to further explore the rewarding contents of the course.

Our takeaway is that building a motivating context in support of adaptive learning is essential to the construction of a successful computer science course. This is one that promotes engagement and raises average levels of achievement. With this understanding of effective personalized learning and teaching, we intend to direct our immediate research in the task of quantifying the impact of different motivating interventions from a neuroscience perspective.

ACKNOWLEDGMENT

We would like to acknowledge Celine Latulipe and Manuel Perez Quinones for establishing course content in RealizeIT, John Healy, and Prabhu Balashanmugam for providing us professional support with RealizeIT platform services, and my colleagues Angela Berardinelli, Bruce Long, Mohsen Dorodchi, Bruce Richards, and Mary Lou Maher for discussion.

REFERENCES

[1] Wenger, E. (2010). Communities of practice and social learning systems: the career of a concept. In Social learning systems and communities of practice (pp. 179-198). Springer, London.

[2] Wenger, E. (1999). Communities of practice: Learning, meaning, and identity. Cambridge university press.

[3] Lister, R., Box, I., Morrison, B., Tenenberg, J., & Westbrook, D. S. (2004). The dimensions of variation in the teaching of data structures. ACM SIGCSE Bulletin, 36(3), 92-96.

[4] Hakulinen, L. (2011, November). Card games for teaching data structures and algorithms. In Proceedings of the 11th Koli Calling International Conference on Computing Education Research (pp. 120-121).

[5] Gelfand, N., Goodrich, M. T., & Tamassia, R. (1998, March). Teaching data structure design patterns. In Proceedings of the twenty-ninth SIGCSE technical symposium on Computer science education (pp. 331-335).

[6] Blum, L., & Cortina, T. J. (2007). CS4HS: an outreach program for high school CS teachers. ACM SIGCSE Bulletin, 39(1), 19-23.

[7] Hazzan, O., Gal-Ezer, J., & Blum, L. (2008, March). A model for high school computer science education: The four key elements that make it!. In Proceedings of the 39th SIGCSE technical symposium on Computer science education (pp. 281-285).

[8] Bockmon, R., Cooper, S., Gratch, J., & Dorodchi, M. (2020, February). Validating a CS attitudes instrument. In Proceedings of the 51st ACM Technical Symposium on Computer Science Education (pp. 899-904).

[9] RealizeIt. http://realizeitlearning.com/wp-content/uploads/2018/04/System-Overview-Feb-2018.pdf

http://realizeitlearning.com/community/

[10] Edgcomb, F. Vahid. How Many Points Should Be Awarded for Interactive Textbook Reading Assignments? Proc. of Frontiers in Education conference (FIE), El Paso, Oct. 2015.

[11] A. Edgcomb, F. Vahid, R. Lysecky. Students Learn More with Less Text that Covers the Same Core Topics, Proc. of Frontiers in Education conference (FIE), El Paso, Oct. 2015.

[12] A. Edgcomb, F. Vahid, R. Lysecky, A. Knoesen, R. Amirtharajah, and M.L. Dorf. Student Performance Improvement using Interactive Textbooks: A Three-University Cross-Semester Analysis, Proc. of ASEE Annual Conference, Seattle, June 2015.

[13] A. Edgcomb and F. Vahid. Simplifying a Course to Reduce Student Stress so Students Can Focus Again on Learning, Proc. of ASEE Annual Conference, New Orleans, June 2016. https://www.zybooks.com/research

[14] Buffardi, K., & Edwards, S. H. (2014, March). Introducing CodeWorkout: an adaptive and social learning environment. In Proceedings of the 45th ACM Technical Symposium on Computer Science Education (pp. 724-724). ACM.

[15] David A. Kolb.1984. Experiential Learning: experience as the source of learning and development. Prentice-Hall,London.

[16] Idit Ed Harel and Seymour EdPapert. 1991. Constructionism. AblexPublishing.

[17] Masika, R., & Jones, J. (2016). Building student belonging and engagement: insights into higher education students' experiences of participating and learning together. Teaching in Higher Education, 21(2), 138-150.

[18] Freeman, S., Eddy, S. L., McDonough, M., Smith, M. K., Okoroafor, N., Jordt, H., & Wenderoth, M. P. (2014). Active learning increases student performance in science, engineering, and mathematics. Proceedings of the National Academy of Sciences, 111(23), 8410-8415

[19] Forsyth, B., Kimble, C., Birch, J., Deel, G., & Brauer, T. (2016). Maximizing the adaptive learning technology experience. Journal of Higher Education Theory and Practice, 16(4).

[20] Jennifer A. Fredricks, Phyllis C. Blumenfeld, and Alison H. Paris, "School Engagement: Potential of the Concept, State of the Evidence," Review of Educational Research 74, no. 1 (2004): 59; online at http://www.isbe.net/learningsupports/pdfs/engagement-concept.pdf.

[21] Thompson, J. (2013). Types of adaptive learning. 2015-05-20]. https://www. cogbooks. com/white-papers-adaptive. html.

[22] Ryan, R. M., & Deci, E. L. (2000). Intrinsic and extrinsic motivations: Classic definitions and new directions. Contemporary educational psychology, 25(1), 54-67.

[23] Forsyth, B., Kimble, C., Birch, J., Deel, G., & Brauer, T. (2016). Maximizing the adaptive learning technology experience. Journal of Higher Education Theory and Practice, 16(4).

[24] Ng, B. (2018). The neuroscience of growth mindset and intrinsic motivation. Brain sciences, 8(2), 20.

[25] Froiland, J.M.; Worrell, F.C. Intrinsic motivation, learning goals, engagement, and achievement in a diverse high school. Psychol. Sch. 2016, 53, 321–336. [CrossRef]

[26] Vervaeke, J.; Ferraro, L. Relevance, meaning and the cognitive science of wisdom. In The Scientific Study of Personal Wisdom; Springer Netherlands: Berlin/Heidelberg, Germany, 2013; pp. 21–51.

[27] Lishinski, A., Yadav, A., Good, J., & Enbody, R. (2016, August). Learning to program: Gender differences and interactive effects of students' motivation, goals, and self-efficacy on performance. In Proceedings of the 2016 ACM Conference on International Computing Education Research (pp. 211-220).

[28] Hutto, C.J. & Gilbert, E.E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI, June 2014. Conference Name:ACM Woodstock conference

[29] Computer Science Curricula 2013. https://www.acm.org/binaries/content/assets/education/cs2013_web_final.pd

[30] Fazey, D. M., & Fazey, J. A. (2001). The potential for autonomy in learning: Perceptions of competence, motivation and locus of control in first-year undergraduate students. Studies in Higher Education, 26(3), 345-361.

[31] Murray, G. (2014). The social dimensions of learner autonomy and self-regulated learning. Studies in Self-Access Learning Journal, 5(4), 320-341.

[32] Núñez, J. L., & León, J. (2015). Autonomy support in the classroom: A review from self-determination theory. European Psychologist, 20(4), 275.

[33] Vahid, F. (2016, March), zyBooks Paper presented at 2016 EDI, San Francisco, CA. https://peer.asee.org/27376

[34] Shon, H., & Smith, L. (2011). A review of poll everywhere audience response system. Journal of Technology in Human Services, 29(3), 236-245.

[35] Alvarado, C., Umbelino, G., & Minnes, M. (2018, February). The persistent effect of pre-college computing experience on college CS course grades. In Proceedings of the 49th ACM Technical Symposium on Computer Science Education (pp. 876-881).

[36] Decker, A., & McGill, M. M. (2019). A systematic review exploring the differences in reported data for pre-college educational activities for computer science, engineering, and other STEM disciplines. Education Sciences, 9(2), 69.