Improving Student Success Through Parachuting*

Chris Bourke[1] and Nirnimesh Ghose[2]
School of Computing
University of Nebraska–Lincoln
Lincoln, NE 68588-0115
[1]chris.bourke@unl.edu
[2]nghose@unl.edu

Abstract

This paper reports on our experiences implementing a *parachute program* that identifies struggling students in a Computer Science I course and invites them to "parachute" out and start over mid-semester in a lower-level computing course that still confers credit toward their degree program. This program gives students a second chance at success, aims to improve student outcomes in early computing curricula, and to improve retention overall in computing majors.

1 Introduction

Many students struggle in introductory STEM courses. Within computing disciplines the number of DFW students (those receiving a non-passing grade of D or F or withdrawing from the course) has remained consistent across time, delivery mode, programming language, etc. [15]. Much research and effort has been done in an attempt to address this issue and to improve student outcomes. Introductory computing courses are especially challenging because of the non-uniform and non-standard coverage of computing in high schools. Students come into college with highly varying backgrounds in computing.

We present a novel approach to this problem through a *Parachute Program* that has been used at the University of Nebraska–Lincoln's School of Computing for the last two years. Through a combination of self- and instructor-based assessments, struggling students are identified at pre-defined milestones in a Computer Science I (CS1) course. Students who are at risk of failing are invited to "parachute" out of the CS1 course and safely land in a lower-level Computer Science 0.5 (CS0.5). In this new course they are given the opportunity to start over.

The goal of this program is to improve student outcomes overall. For struggling students in particular, it aims to improve retention by strengthening their foundation and giving them a chance to acclimatize themselves to college life.

2 Related Work

Methods have been tried across many STEM disciplines to address struggling students. Placement exams are commonly used to place students into an appropriate introductory course that matches their background knowledge and readiness. However, these exams are far from perfect. Underplacement (placing well-prepared students in lower level courses) is far more common than overplacement (placing students in a course they are not prepared for) [13]. Placing students into remedial courses can have wide-reaching and long-term negative impacts [7]. Taking a remedial course also has an an impact on time-to-graduation and disproportionately affects under-represented minority students. Not only do remedial courses have extensive costs to institutions and students, the overall research on their benefit is mixed at best [2].

Other efforts attempt to identify struggling students while they are taking a course. Math departments have been administering "gateway" exams since at least the 1980s which have evolved over time. The key difference with gateway exams is that they are given within a course and intended to measure skills which every student in a course should develop rather than whether or not they are prepared to take the course [10]. Reporting indicates passing rates are typically high (though initially may be as low as 50% the first time a student takes it). However, this is in contrast to the typical failure rates of college math courses [4] which are as high as 27% in Calculus I [5].

In addition, though a gateway exam may be part of a student's grade, they only serve as a guide to the student and instructor as to whether or not they have learned the material that is expected of them or whether they are on the right trajectory to complete the course. Whether or not they pass a gateway exam does not affect a student's enrollment in the course.

Within computing disciplines several placement exams have been developed [14, 12]. These efforts have faced challenges because the typical CS-related K–

12 curriculum is not nearly as standard as (say) math and may not exist at all. As a consequence, students enter college with a much wider variety of prior computing experience.

However, these tests have been used mostly to determine whether or not a student should be placed in CS1 (low or no prior experience) or is prepared enough to start off in CS2. This is because historically most computing curricula only have these two options for introductory courses. With the explosion in computing enrollment and the expansion of computing sub-disciplines (CS+X, data science, informatics, etc.), this is less true. Many institutions have started to offer a wider variety of introductory courses catering to different student interests, backgrounds and goals.

A recent effort has been made to adapt and extend existing placement exams as an instrument to assess a student's preparedness to take CS1 or to take an alternative introductory course [3]. Even if no validated instrument is used, some departments have started offering multiple sections of CS1 targeted toward students with differing prior computing experience [9]. Though they may cover the same topics, these courses may differ in the delivery or assessment or offer more or alternative opportunities for practice and collaboration.

No evidence could be found that this parachute program or any similar program is used by other institutions. Nevertheless we mention that the parachute program was inspired by a similar program that was supposedly used in some chemistry departments (folklore).

3 Parachute Program

The parachute program was first piloted in spring 2023 and involved several significant changes to curriculum and to the structures, schedule, and topics of courses.

3.1 Courses

The School of Computing at the University of Nebraska-Lincoln offers a wide variety of introductory computing courses. The two most relevant courses to the parachute program are CSCE 101 and CSCE 155A.

CSCE 101 is a CS0.5 course in that it is more than a rudimentary computing skills course but not fully a CS1 course. It emphasizes problem solving and introduces basic python programming. As a service course, it is intended mostly for non-computing majors as it satisfies general education requirements. Traditionally 101 has been a terminal computing course. As a consequence, enrollment is extremely diverse, attracting students from all levels and dozens of different majors.

CSCE 155A is a full CS1 course [1] intended for computing majors. It covers problem solving and software development principles. Enrollment consists mainly of incoming freshman computing majors serving as their first college-level computing course. There is no computing prerequisite and no prior computing or programming experience is necessary or expected. This course introduces computational thinking by solving puzzles. Students are introduced to programming by first writing formulas in Google Sheets and then developing rudimentary programs to solve real-world examples using Coral [6]. Once the students are comfortable using computational thinking to write programs, Python is introduced using a flipped classroom model. The students learn the basics before the class using an interactive, auto-graded textbook. Class time is spent on live coding demonstrating the problem solving process on problems and puzzles from the well-known Algorithmic Puzzles book [11]. For example, when recursion is covered, class time is used to develop code to solve the three jugs puzzle (given an 8-pint jug full of water and two empty jugs of 5- and 3-pint capacity, get precisely 4 pints of water in one of the jugs by filling up and emptying jugs into others) using recursion. Students are then assessed using a more complex problem for which they develop a full program. For recursion, students are challenged to develop a program to draw the decision tree for the four knights puzzle [8].

Both courses are offered as 15-week semester courses in fall and spring semesters and are synchronized with the same weekly lecture and lab schedules.

3.2 Process

Student progress in CSCE 155A is closely tracked in the first five weeks. Assessments are evenly distributed throughout the semester so that approximately one third of the assessment has been completed by the end of the 5th week. In the first week students are given a Computing Skills Inventory "exam" [3] which is graded and gives both the student and the instructor a baseline on whether or not the student is likely to succeed (pass) in 155A.

In week 5 students whose grade is less than 73% (C) are invited to join the parachute program and switch their enrollment to CSCE 101. The invitation is entirely voluntary. Those that opt-in to the parachute program are re-enrolled into 101 and "restart" their introductory course in week 6. The re-enrollment and administrative processes are handled entirely by faculty and staff. Those that choose not to opt-in continue their enrollment in CSCE 155A. In all cases, tracking of student progress continues through the end of the semester.

3.3 Curricular Changes

Prior to adopting the parachute program, several curricular changes were made. First, CSCE 101 was reorganized so that the first 5 weeks consisted of general computing topics and computational thinking but no formal programming. Introduction to coding (Python) was moved to begin in week 6. Second, curriculum rules were changed to allow CSCE 101 to count toward degree requirements of computing majors *provided* that it was completed before any other computing courses. Prior to this, it did not count toward any computing degree (other than general credit hours).

These two changes were necessary to make the parachute program possible. Students being parachuted into 101 from 155A would have been completely lost if simply dropped into a course 5 weeks into the semester. By front-loading the course with general computing topics, parachuted students get a fresh (re)start in 101 at the point that Python is being introduced in both courses.

It was also necessary to ensure that 101 counted toward a computing degree program so that credit hours were not lost by the student. This change also mitigates the impact that parachuting has on the time-to-graduation for computing students.

4 Results

The parachute program began in spring 2023 and has been offered for four semesters total. During this period, enrollment and DFW rates (see Table 1a) have been fairly static with overall rates being 15.25% and 37.37% for CSCE 101 and CSCE 155A respectively.

The number of invitees and those who accepted the invite to parachute from CSCE 155A to CSCE 101 can be found in Table 1b. Spring 2023 was the initial pilot for this program and so was more targeted with fewer invitees. The number of invitees has grown as adjustments and refinements to the process have been made. The number of students accepting the parachute opportunity is somewhat volatile but consistent with respect to on/off semesters (the bulk of incoming freshman students are in fall while spring has a more diverse enrollment of transfers, retakes, non-majors, etc.).

For students that accepted our invitation the outcomes were very positive. They successfully integrated into CSCE 101 and were able to perform fairly well. The majority (20/25) passed mostly with A (11) or B (6) grades. Nevertheless some were not able to make the transition and either failed (1) or ultimately withdrew (4).

In contrast, those that declined our invitation (37) to parachute did extremely poorly. While one student was able to achieve a B and 6 were earned a C, the vast majority earned grades of D (6) or F (13) or ultimately withdrew

Table 1: Course Data during the parachute program from Spring 23 (S23) through Fall 2024 (F24)

	CSCE 101		CSCE 155A	
Sem	Enr.	DFW	Enr.	DFW
S23	131	18.32%	69	34.78%
F23	99	18.18%	132	35.61%
S24	157	12.74%	86	36.05%
F24	118	12.71%	85	43.53%
Total	505	15.25%	372	37.37%

(a) Semester-by-semester enrollment (Enr.) and DFW rates

Sem	Invited (%)	Accepted (%)
S23	4 (5.80%)	4 (100%)
F23	14 (10.61%)	7 (50%)
S24	24 (27.91%)	3 (12.5%)
F24	20 (23.85%)	11 (55%)
Total	62 (16.67%)	25 (40.32%)

(b) Parachute program invitees and accepts(numbers and percentage of the class) by semester.

(11) giving an overall DFW rate of 81.08%, far exceeding typical failure rates. A sankey flow diagram of these outcomes can be found in Figure 1.

Beyond the success rates of these introductory courses, one of the aims of the parachute program is to retain more students in computing majors. Unfortunately, among students prior to fall 2024 that received an invitation to participate (42) nearly half (20) had dropped or failed out of college entirely (though it is possible some transferred to another institution). Nevertheless, among those still enrolled, nearly one third (7/22) persisted in their computing major and were still matriculating a year later. Though the remaining students found different majors (15/22), some continued with computing as a minor. Though it is still early, data from fall 2024 provides a bit more optimism. More students in this cohort (7/11) are continuing with computing courses and all are still enrolled.

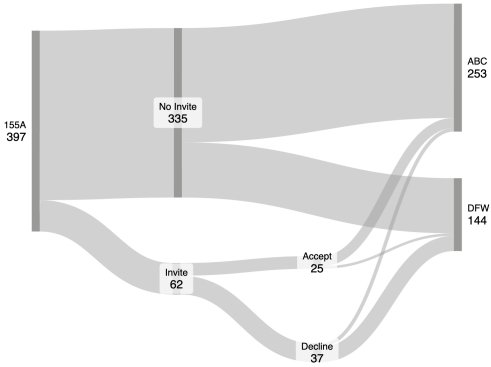

Figure 1: Flow of students through CSCE 155A and the Parachute Program.

5 Discussion

The numbers presented in this paper are admittedly small. However, we are making no claims of generalization or formal methodological analysis. The

parachute program attempts to engage with students who are already on the margin and so the numbers will necessarily be small. Nevertheless, our experiences and data over the last two years give several interesting points of discussion. Our experiences have also led to refinements and we've identified some best practices.

5.1 Common Outcomes

Given that students who accepted our invite did well and those that declined tended to fail, an obvious first question is: why did students decline our invitation? Unfortunately, we don't have much in the way of direct data to answer this question. A small number of students were *unable* to accept the invitation because they had already taken and passed the parachute course or they were non-computing majors whose program required 155A (or would not count 101). We do have some anecdotal evidence that students wanted to persist with CSCE 155A and intended to improve, a sort of sunken cost fallacy. However, more commonly we observed that students who were already failing at that point were completely disengaged with the course to the point that they ignored even our outreach efforts.

This theme continued when we looked at the near-term data. About half of all students who declined parachuting had failed or dropped out of our institution entirely within the next year. This suggests that these students were experiencing problems that went well beyond these courses and the computing curriculum. Most had failed more than one course or even the majority of their courses.

Nevertheless, the parachute program did result in significant positive outcomes. It was able to successfully retain a good number of students who may not have persisted in computing or in college at all. Given the minimal investment in administrative time and effort, these are very positive outcomes.

Though retention in computing majors is certainly a goal, it is not the only positive outcome. Many students switched to other disciplines (which is common) but still earned credit for the parachute course toward their degree program. This program serves as an easy "off-ramp" to students who were not entirely set on a computing major. They were able to get a taste of computing and decide that it was not for them without losing out on the credit hours or negatively impacting their GPA. These students also have the potential to return to computing later on or in alternative ways such as receiving a minor.

Though it was not observed directly, the invitation process itself may serve as a wake-up call to students who are under-performing early in the course and can serve as an early intervention mechanism.

5.2 Refinements

Though the program is only two years old, several refinements and adjustments have been made. Only about a third of the invitees are represented in the final DFW students. That is, a good majority of DFW students were not initially identified and invited to parachute in the first 5 weeks. This suggests that students do not necessarily "fail fast" in this course. To address this, we have adjusted the cutoff for invitees downward over time in an effort to capture more students who would otherwise fail. In the most recent offering of the course, students meeting the parachute criteria were contacted directly with an invitation, however, the entire class was made aware of the program. Students who were doing well or reasonably well were also given the opportunity to opt-into the program. Several students took this opportunity citing stress, workload, or perceived difficulty of the course as reasons.

In Fall 2024, the program was expanded to include another CS1 section designed for students with some computing background. However, among 8 students invited, all declined and all eventually failed. Nevertheless, we intend to continue and expand the program going forward.

5.3 Best Practices

We see the curricular changes discussed in Section 3.3 as necessary to a program like this. If students are to be re-enrolled in a different course during a semester, it is absolutely essential that the course count toward their degree program in a significant way. Without this change, parachuting would only exacerbate time-to-graduation challenges. Any program considering a change like this needs to reexamine and potentially realign their early curriculum as a first step.

Recall that it was necessary to ensure that the content of the parachute course, CSCE 101, be realigned to accommodate students entering the course at week 6. However, our CS1 course content also had to be aligned so that assessments were more evenly distributed using smaller assessments and front-loading the course so that early success or risks could be established. If the course had a more traditional structure, say heavily exam-based, trends in assessment could not have been observed.

Another necessary component is clear and frequent communication. Students need to know if they are doing well in the course even before the 5 week invite period. The invitation also needs to be crafted so that it is not an indictment of the student's performance but instead framed as a way to provide them help and an opportunity to succeed.

Now that we have a good amount of data, we intend to update our communication strategy to provide more transparency with respect to that data. In particular, we intend to share general success rates of those who choose to

parachute versus those that do not.

Though not a best practice, another necessary component to a program like this is administrative buy-in. When this program was originally proposed there was initial concern that students could not be administratively re-enrolled mid-semester. There were questions as to the impact on financial aid, scholarships and other legal issues. As a result, the program, though approved, is required to be voluntary for students. Administrative support was also necessary to handle the logistics of re-enrollment and to prevent late enrollment fees.

6 Conclusion

As with any intervention, change and improvement are incremental; there is no silver bullet. The results of our parachute program are somewhat mixed, but the positive results are compelling especially given the minimum amount of work required to implement and administer it. In doing so, it has also led to other positive curricular changes. This paper has outlined the general process, its results, and best practices so that other institutions might think about adopting or adapting a similar program.

References

[1] B. A. Becker and K. Quille. 50 years of cs1 at sigcse: A review of the evolution of introductory programming education research. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, SIGCSE '19, page 338–344, New York, NY, USA, 2019. Association for Computing Machinery.

[2] A. Boatman and B. T. Long. Does remediation work for all students? how the effects of postsecondary remedial and developmental courses vary by level of academic preparation. *Educational Evaluation and Policy Analysis*, 40(1):29–58, 2018.

[3] R. Bockmon and C. Bourke. Validation of the placement skill inventory: A cs0/cs1 placement exam. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, SIGCSE 2023, page 39–45, New York, NY, USA, 2023. Association for Computing Machinery.

[4] B. Bonham and H. Boylan. Developmental mathematics: Challenges, promising practices, and recent initiatives. *Journal of Developmental Education*, 34(3):1–10, 2011.

[5] D. M. Bressoud, V. Mesa, and C. L. Rasmussen. Insights and recommendations from the maa national study of college calculus. *Mathematics Teacher*, 109(3), 2015.

[6] A. D. Edgcomb, F. Vahid, and R. Lysecky. Coral: An ultra-simple language for learning to program. In *2019 ASEE Annual Conference & Exposition*, number 10.18260/1-2–32550, Tampa, Florida, June 2019. ASEE Conferences. https://peer.asee.org/32550.

[7] N. C. for Education Statistics. National postsecondary student aid study 2020 (npsas:20), 2020.

[8] M. Iranpoor. Knights exchange puzzle—teaching the efficiency of modeling. *INFORMS Transactions on Education*, 21(2):108–114, 2021.

[9] M. S. Kirkpatrick and C. Mayfield. Evaluating an alternative cs1 for students with prior programming experience. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '17, page 333–338, New York, NY, USA, 2017. Association for Computing Machinery.

[10] P. G. LaRose and R. Megginson. Implementation and assessment of on-line gateway testing. *PRIMUS*, 13(4):289–307, 2003.

[11] A. Levitin and M. Levitin. *Algorithmic puzzles*. Oxford University Press, 2011.

[12] M. S. Peteranetz and A. D. Albano. Development and evaluation of the nebraska assessment of computing knowledge. *Frontiers in Computer Science*, 2:11, 2020.

[13] J. Scott-Clayton, P. M. Crosta, and C. R. Belfield. Improving the targeting of treatment: Evidence from college remediation. *Educational Evaluation and Policy Analysis*, 36(3):371–393, 2014.

[14] A. E. Tew and M. Guzdial. The fcs1: A language independent assessment of cs1 knowledge. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*, SIGCSE '11, page 111–116, New York, NY, USA, 2011. Association for Computing Machinery.

[15] C. Watson and F. W. Li. Failure rates in introductory programming revisited. In *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education*, ITiCSE '14, pages 39–44, New York, NY, USA, 2014. Association for Computing Machinery.