Summer 2024

The Graduation Walk: Pareto Optimization of Degree Paths

Arianna Gail Sy Chaves
Missouri University of Science and Technology

Department:

THE GRADUATION WALK: PARETO OPTIMIZATION OF DEGREE PATHS

by

ARIANNA GAIL SY CHAVES

A THESIS

Presented to the Graduate Faculty of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

in

COMPUTER ENGINEERING

2024

Approved by:

Sahra Sedigh Sarvestani, Advisor
Ali R. Hurson, Co-Advisor
Ahmad Alsharoa

ABSTRACT

A student's academic history, course availability at their institution, and the overall degree of difficulty of the schedule for each semester are all critical factors in their academic success and experience. This thesis proposes an advanced recommendation algorithm that considers these real and conflicting factors that are involved in identifying a prudent degree path - a semester-by-semester course schedule to graduation - for each student. The original contribution of this work is the use of weighted-sum multi-objective constraint programming to minimize an increased number of optimization criteria and identify Pareto-optimal degree paths to be recommended to the student. The conflicting objectives optimized in this problem are minimum time-to-degree, maximum projected course grades, maximum alignment with the student's interest, and maximum degree path robustness. The proposed approach is validated through simulation and its efficacy is compared to previous results on the Pervasive Cyberinfrastructure for Personalized Learning and Instructional Support (PERCEPOLIS) platform.

ACKNOWLEDGMENTS

I would like to thank everyone who has supported me throughout my studies at Missouri S&T, including my family, significant other, close friends, and on-campus music community, for being a positive and supportive presence over the years at this university. I would also like to thank my academic colleagues - my advisors, team members, fellow organization members, and classmates - for imparting their wisdom and expertise upon me and helping me grow as a scholar and engineer.

I especially wish to thank my advisors, Dr. Sedigh Sarvestani and Dr. Ali Hurson, for guiding my academic journey since my second year of undergraduate studies and enabling my success every step of the way. I would also like to thank Dr. Ahmad Alsharoa for serving on my committee and providing helpful technical insight on my research. Next, I would like to thank the other members of the PERCEPOLIS team - Nic Dobbins, Mitchell Skaggs, Colton Walker, Austin Beckerdite, and Seth Sievers - for their collaboration on the project and assistance throughout my research journey.

Finally, I would like to thank the National Science Foundation (NSF), as this work is supported, in part, by their generous contribution under Award DUE-1742523.

TABLE OF CONTENTS

Page

LIST OF ILLUSTRATIONS

LIST OF TABLES

Nomenclature

a_c	Set of semesters in which course c is unavailable
b_c	Set of semesters in which course c will cause delay
C	Set of courses taken in previous semesters
c	Course
CHL_n	Student-set credit hour limit for semester n
CHR	Total minimum credit hour requirement
D	Set of recommended semesters r_n
DP	Set of recommended degree paths
f(D, W)	Weighted-sum optimization function
GPA_c	Average GPA of past students for course c
gpa_c	Predicted grade for a course c
h_c	Nominal credit hour load for a course c
h_n	Total nominal credit hour load for semester n
H_p	Number of credit hours taken in previous semesters
i_c	Course importance
j	Number of similar students sufficient for drawing a comparison
k_i	Set of grades for student i
L	Number of recommended semesters r_n
N	Maximum number of semesters, acts as an arbitrary upper bound
n	Semester
N_min	Minimum number of semesters
P_n	Set of prerequisites for courses in semester n

p_n	Perceived load for semester n
Q_n	Set of corequisites for courses in semester n
R	Set of all recommended courses
R_int	Set of recommended courses that match student's interests
r_n	Set of recommended courses for semester n
S_n	Set of available courses for semester n
SEL_n	Selected (locked) courses in semester n
t_c	Earliest semester that requires course c as a prerequisite
U	Set of students in data set
U_S	Set of similar students
v	Number of courses that match student's interests
W	Set of objective weights

1. INTRODUCTION

Technology-Enhanced Learning (TEL) aims to significantly improve the quality of education at academic institutions and enrich students' educational experiences. TEL is especially relevant in the field of personalization in higher education, which has been identified by the National Academy of Engineers as one of fourteen "Grand Challenges for Engineering" in the 21st century [1]. To increase the customizability of students' higher education, researchers have dedicated significant effort to developing recommender systems for university course enrollment [2].

The most widely implemented approach to curriculum personalization is individual advising and manual course selection. This approach is tedious, time-consuming, and cost-ineffective, with students and advisors having to obtain information from broad course descriptions, assumptions, and other students' anecdotes to determine a personalized course schedule. These inefficiencies can lead to missed course opportunities, poor class performance, overspending on a degree, delayed graduation, and decreased student retention and success rates [3]. TEL platforms have previously sought to improve on curricula personalization, but few have seen any large-scale implementation due to shortcomings, such as a limited scope of personalization or a lack of confidence in recommendations.

The Pervasive Cyberinfrastructure for Personalized Learning and Instructional Support (PERCEPOLIS) system is an intelligent recommender system designed to modernize postsecondary education. Previous work has developed PERCEPOLIS to accurately recommend complete degree paths that ensure graduation and minimize time-to-degree, considering factors such as student interests and performance. More recently, efforts on PERCEPOLIS focused on degree path recommendation based on similar students' decisions in the past [4] and further improvements to the model's performance. Thus far, however, PERCEPOLIS is limited to recommending one degree path, and the scope of optimization focuses largely on minimizing time-to-degree.

The original contribution of this research is the formulation of an advanced Pareto optimization problem for generating and recommending a set of prudent and unique degree paths. To this end, I have carried out the following research tasks:

- Identified novel objectives pertaining to course feasibility and student success.
- Identified appropriate constraints to ensure validity of the generated schedule.
- Formulated a multi-objective optimization problem for schedule generation using a weighted sum approach.

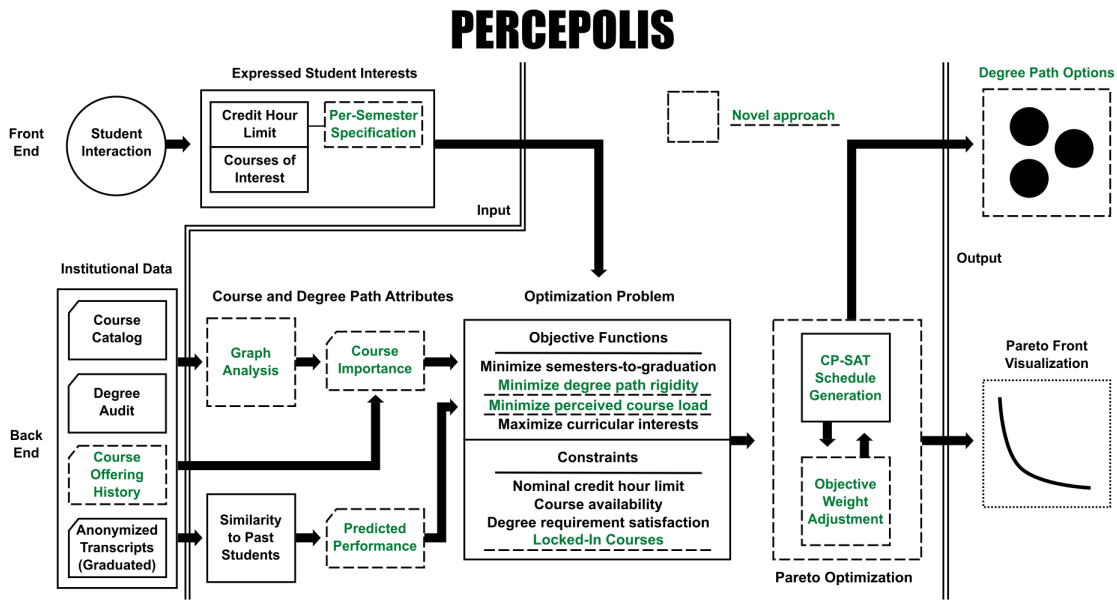

Figure 1.1. High-Level diagram of PERCEPOLIS

- Generated a set of unique degree paths that represent the Pareto front of optimal solutions.
- Compared the generated Pareto-optimal degree paths to those generated without multi-objective optimization.
- Validated the feasibility of generated degree paths using the Pervasive Cyberinfrastructure for Personalized Learning and Instructional Support (PERCEPOLIS) platform [5].

Figure 1.1 depicts the additions made by the proposed approach to the PERCEPOLIS system. The uniqueness of this work lies in the increased scope of objectives such as course importance in a curriculum, relative academic performance, topical fitness, and perceived semester workload. These aspects go beyond considerations made in current literature [2], better encapsulate the enrollment decision-making process, and add significant complexity to implementing recommender systems such as PERCEPOLIS. The introduction of *course importance* - a measure of a course's criticality and fragility in a student's degree path - is one such objective that has previously not been considered by other methods. Prioritizing one consideration over another may lead to a trade-off, resulting in multiple degree paths that are equally yet differently effective. The system generates and curates these paths so that students can choose from them based on their individual preferences. This approach preserves the multi-objective nature of the original problem by not relying on a single set

of arbitrary objective weights; instead, parts of the Pareto front of optimal schedules are explored. The flexibility provided by a set of suggestions allows recommendations to adapt easily to a variety of factors, such as changes in enrollment status and changes in university enrollment cost structures. By giving students greater and more informed control over their academic careers, PERCEPOLIS aims to reduce time-to-degree, improve academic performance, and increase retention of students.

The remainder of this thesis begins with Section 2, which presents the current approaches to single- and multi-objective course recommendation and the gaps in research that this work fills. Section 3 details the proposed multi-optimization technique, including criteria formulation, degree path generation, and solution ranking. Section 4 describes the implementation of this approach in PERCEPOLIS and seeks to validate the results of this work using this platform. Finally, Section 5 covers the known limitations of this algorithm and expands on future work in this field.

2. LITERATURE REVIEW

2.1. RELATED WORKS

Figure 2.1 shows the landscape of works relating to course recommendation. The university course enrollment process has two significant components: course selection and course scheduling. In course selection, algorithms focus on specific aspects of a student's higher education experience to determine which course or courses would fit best in a student's degree plan. These attributes can measure a student's progress towards a degree or potential success during their educational career. In course scheduling, algorithms determine the sequence of courses that a student should take to reach their graduation goal. While many of these works focus only on either recommendation or scheduling, they provide a necessary background for identifying what aspects of course selection have been formulated and analyzed for this purpose. The wide range of novel approaches shows how diverse the field is, and strategies such as collaborative filtering, graph theory, machine learning, genetic programming, and more have all been used to solve the same problem.

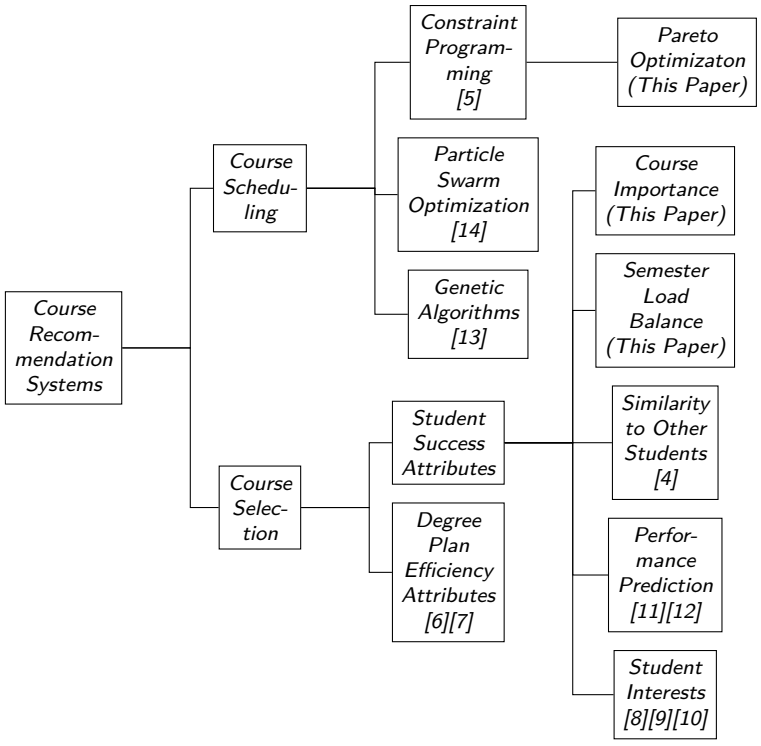

Figure 2.1. Taxonomy of works related to course recommendation systems

Table 2.1. Comparison of course recommender systems

	Type of Metric				Degree Planning					Software Features	
	Attributes			Multi-Objective Optimization	Holistic Validity Check	Proof of Optimality	Adaptability Considerations	Manual Adjustment	Multiple Recommendations	Integration with Other Enrollment Software	Open License Development
	Course Availability	Past Performance	Student Life Considerations								
PERCEPOLIS	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Stellic	✓				✓			✓			
Degree Compass	✓	✓									
Curricular Analysis	✓				✓	✓	✓			✓	

2.1.1. Course Selection. An important characteristic of the course selection process is its ability to advance a student toward graduation. Considerations for recommendations in this process include degree requirements and credit hour counts. The most straightforward approach to recommending courses in this manner is through a degree audit, typically performed and analyzed manually by an expert such as an advisor. Many web-based platforms have implemented ways of recommending courses due to their ability to satisfy degree requirements, including Degree Compass [6], Curricular Analytics [7], Stellic, uAchieve 5.0, and the Pervasive Cyberinfrastructure for Personalized Learning and Instructional Support (PERCEPOLIS) [15] [16] [17] [5]. Table 2.1 compares these products by the types of personalization possible and additional algorithm characteristics. Compared to PERCEPOLIS, many of these products focus on either recommendation or scheduling, but not both.

In addition to degree advancement and degree requirements, a student's academic history should be considered to determine which courses are best for a given curriculum. Aspects to academic history include expressed student interest, predicted performance, and similarity to other students who have successfully graduated in the same program in the past. Content-based and collaborative filtering is a popular approach to addressing expressed student interest [8]. Newer methods of tailoring course recommendations to students' interests include knowledge graph embedding [9] and deep learning [10]. Another metric for course recommendation is predicting a student's performance in particular courses. [11] discusses several approaches that utilize genetic algorithms to predict academic performance. The PERCEPOLIS implementation of Q-learning on a Markov Decision Process suggests courses from semesters with high expected GPAs [4]. These methods often incorporate student similarity considerations to better tailor schedules towards student interests and higher predicted performance. Aside from the aforementioned collaborative filtering techniques, the Pearson Correlation Coefficient

has been used to determine students' likeness with each other to prune data used in performance analysis [4]. A measure related to predicted performance is course difficulty, defined by Premalataha in their work on course recommendation [12].

2.1.2. Course Scheduling. Many works propose optimization-based processes to determine the sequence of courses a student should take to reach graduation. For example, implementations of genetic algorithms are proposed to arrange courses into a cohesive degree plan that a student can follow through graduation in works such as Wicaksono's and Putra's [13]. Another method of course scheduling is particle swarm optimization, which improves upon many genetic algorithms by decreasing the time-to-convergence on a feasible solution [14]. To handle cases of infeasibility not otherwise covered by previously mentioned methods, researchers have turned to constraint programming to perform course scheduling, which involves formulating and solving several objective functions at once [5]. Constraint programming defines each objective or constraint as one of the decision factors a student considers when enrolling in a semester of courses and combines them into a single optimization problem via the weighted sum method. The PERCEPOLIS platform uses this process to solve for minimal time-to-degree, minimal credit hours per semester, and maximal student interest [5].

2.2. BACKGROUND

The proposed approach uses principles from multiprocessing, optimization, and networking to generate degree paths. These concepts apply well to the course recommendation because this model handles sequential "tasks" - in this case, courses within semesters - that can be rearranged in their chronological execution.

2.2.1. Multiprocessing and Job Scheduling. The problem of course scheduling shares many similarities with job scheduling in a multiprocessing platform. The work of Kasapidis et al. shows an evolutionary computation-based approach to generating a schedule for a set of tasks characterized by precedent links [18]. These tasks are akin to courses in a conventional university system, with lower-level courses acting as precedents (i.e., prerequisites) to higher-level ones. Much like how processors have limited resources to dedicate to a set of tasks at time t, students can only take a certain number of credit hours at semester s. Thus, enrollment in a specific set of courses at semester s dramatically affects the time in a student may complete a degree.

In multiprocessing, the execution time of a task is bounded below by the length of its *critical path*: the longest sequence of sub-tasks that must be completed in an application [19]. Curricular Analytics demonstrates how such a critical path can be identified in a given degree program [7]. The critical path is representative of the minimum time-to-degree for a student, and other courses outside of this path can be taken in "parallel" as long as there is room in the semester. Tasks in non-critical paths must be distributed across the degree schedule using stalls, or "no-ops," to accommodate for resource limitations.

2.2.2. Similarity Identification. Course performance can be predicted for a given student by analyzing the past performance of similar students, as addressed in [4]. This approach uses the Pearson Correlation Coefficient (PCC) to partition a large set of anonymized student data and identify a set of similar students. The system can then, with this refined dataset, predict and optimize academic course outcomes representative of a student.

2.2.3. Constraint Programming - Propositional Satisfiability (CP-SAT) Pareto Optimization. Previous work on PERCEPOLIS uses the weighted-sum approach to optimize schedules with boolean constraints [5]. This approach uses a single set of pre-defined coefficients, which requires an *a priori* understanding of balances and tradeoffs to generate a single, optimal solution. To better personalize recommendations, the proposed enhancement generates multiple schedules that prioritize time-to-degree, expected performance, and schedule flexibility differently. In their work, Guerreiro et al. propose an approach that uses a satisfiability (SAT) solver to generate and prove a Pareto front using minimal correction subsets [20]. Similarly, the research described in this thesis generates a Pareto front of schedules by slightly adjusting the coefficients of a weighted sum optimization problem to generate different points.

2.3. SUMMARY

Significant emphasis has been placed on either the efficiency *or* the quality - i.e. chance of student success - of a student's recommended degree plan, and methods to combine them are largely unexplored. The course recommendation space has consistently seen new ways of recommending courses or scheduling courses, but few works have combined the two in ways that strengthen the overall recommendation of the system. PERCEPOLIS is unique in this regard, as both feasible and high-quality degree plans can be generated through the recommendation process. Even within the PERCEPOLIS system, though, many factors of schedule quality, such as course importance and effective course load, are not taken into account when optimizing degree plans. Previous works also

focus on making a single recommendation, despite multi-objective optimization having more than one "correct" answer on the Pareto boundary of the optimization problem. As such, the work described in the following sections aims to add significant growth to the field of course recommendation systems.

3. METHODOLOGY

3.1. OVERVIEW

The enhanced course recommender system involved numerous enhancements to the problem statement and the optimization process implemented in PERCEPOLIS, as reported in [5]. Much of this work focused on formulating and optimizing practical objectives that consider the multitude of academic factors that permeate the enrollment process alongside the constraints that assure degree path validity. To better understand the enhanced course recommender system, in this section first discusses previous effort on PERCEPOLIS, and then the proposed enhancement is detailed.

3.1.1. Foundation of Work. The proposed approach builds off the model designed by Dobbins, Hurson, and Sedigh Sarvestani [5]. This work defines the decision variables of the optimization problem as the courses selected to be taken in each semester (Equation 1).

$$r_n = \{c : c \in S_n\} \; and \; r_n \subseteq S_n \tag{1}$$

The previous model defines the primary objective as minimizing time-to-degree, or the number of semesters to graduation. This equation (Equation 2) [5] remains a high priority and is incorporated in this model as one of the objective functions in the weighted sum.

$$\begin{aligned} \underset{r_n}{\text{minimize}} \quad & L \\ \text{subject to} \quad & L = \{max(n) : r_n \neq \emptyset\} \end{aligned} \tag{2}$$

The secondary objective is also carried over to this work. This objective maximizes the number of preferred courses in the recommended degree path (Equation 5)[5]. The trade-off analysis performed by this approach ensures that only a feasible number of courses are selected from R_int in cases where R_int is exceedingly large.

$$\begin{aligned} \underset{r_n}{\text{maximize}} \quad & v \\ \text{subject to} \quad & v = |R_{int} : R_{int} \subseteq R| \end{aligned} \tag{5}$$

While the previous work considered the minimization of credit hours taken per semester, the current model omits this. By doing so, this work accommodates enrollment systems in which the number of credit hours does not strictly imply a higher cost of tuition. One such model is flat-rate tuition [21], which charges the same amount within a range of taken credit hours in a given semester. Finally, the previous work defines the constraints for this model. Equation 7a [5] gives the degree audit constraint, stating that a degree path must contain all the courses necessary to achieve a specific degree. Equation 7b [5] constrains the model to include the minimum number of credit hours defined by a given institution. Equation 7c and 7d [5] define prerequisite and corequisite constraints, ensuring that courses that must be taken before or alongside other courses are scheduled in the correct chronology. Equation 7e constrains the model to ensure that non-repeatable courses are not taken more than once in the recommended degree path. The credit hour limit constraint from the previous model is redefined in the next section to increase the amount of control the student has on their degree path.

$$degreeAudit(C \bigcup R) = True \tag{7a}$$

$$H_p + \sum_{n=1}^{N} h_n \geq CHR \tag{7b}$$

$$P_n \subseteq \{\bigcup_{i=1}^{n-1} r_i\} \cup C \tag{7c}$$

$$Q_n \subseteq \{\bigcup_{i=1}^{n} r_i\} \cup C \tag{7d}$$

$$R = \bigcup_{n=1}^{N} r_n \; and \; |\{c \in R\}| = 1 \tag{7e}$$

3.1.2. Proposed Approach. A new set of objective functions and a few additional constraints are combined with the previous problem to increase the amount of personalization possible within the model. These elements include degree path flexibility, perceived course load, varying preferences for credit hour limits in different semesters, and course locking.

3.1.2.1. Degree path flexibility. To make a schedule more adaptable or robust, this work considered the flexibility of the overall degree path using the concept of importance. This work defines *course importance* as the inability of a course to shift earlier or later in the degree path. According to [22], importance consists of a node's *fragility* and *criticality* in a graph-based system.

In the context of degree path generation, this work defines a course's fragility as its scarcity - that is, the likelihood that a course will not be available to be rescheduled in another semester. This information is used to formulate the scarcity of a course (Equation 3a), i.e., the semesters that are *not* expected to contain an offering for that course. Common patterns of semester-based course offerings include semesterly (every fall and spring), annually (either every fall or every spring), and biennially (every fall or spring on either even-numbered or odd-numbered years). Some institutions use quarterly or eight-week periods, which is outside the scope of this work. When a student is planning semesters without explicit information regarding availability, they may use an availability prediction instead. By minimizing scarcity, a degree path will be less susceptible to missing enrollments due to academic delays or unexpected changes in a course's offering pattern, such as skipped semesters during cooperative education or work program stints and degree restructuring performed by the university. The set of semesters in which a course will not be available is defined as:

$$a_c = n : c \notin S_n \tag{3a}$$

A course's criticality is the severity of the effect that delaying the course's enrollment by a semester will have on other courses in the degree path. This metric uses the latest possible semester in which a course can be scheduled; for example, a course that acts as a prerequisite to one other course cannot be scheduled any later than the second-to-last semester in the recommended degree path. Courses that are the earliest prerequisites in a path - especially in the critical path-to-degree - are more likely to delay graduation if the course unexpectedly becomes unavailable or is delayed for reasons such as employment-based breaks in enrollment. Equation 3b defines the set of all recommended courses that rely on course c as a prerequisite. Equation 3c defines the set of semesters in which placement would delay a dependent course.

$$t_c = \begin{cases} min(n) : c \in P_n, \quad c \in \{\bigcup_{i=n}^{N} P_i\} \\ \\ min(n) : r_n \neq \emptyset, \quad c \notin \{\bigcup_{i=n}^{N} P_i\} \end{cases} \tag{3b}$$

$$b_c = \{n : n \geq t_c\} \tag{3c}$$

Combining criticality and scarcity yields the course's importance (Equation 3d), Minimizing this metric reduces the number of courses that are at higher risk of causing enrollment problems.

$$\begin{aligned} \underset{c}{\text{minimize}} \quad & i_c \\ \text{subject to} \quad & i_c = |a_c \cup b_c \end{aligned} \tag{3d}$$

3.1.2.2. Perceived course load. The perceived course load of a given student's semester can be derived from two pieces of information: the nominal course load as defined by an institution and the predicted academic performance as inferred from a priori information of other students' outcomes in the course. To predict a student's performance in a given course, this approach first identifies a cluster of previously graduated students with similarities in past performance and degree trajectory:

$$U_S = \{k_i\} \; and \; U_S \subset U \tag{4a}$$

From this cluster, this approach determines the predicted performance (Equation 4b) and perceived load (Equation 4c) for courses with enough data - that is, at least j similar students have taken a specific course.

$$gpa_c = \begin{cases} \dfrac{1}{|\{k_i \; : \; c \; \in \; k_i\}|} \sum GPA_c, & k \geq j, \; j = 30 \\ GPA_C, & k < j, \; j = 30 \end{cases} \tag{4b}$$

where $GPA_c \in U_S$.

$$\begin{aligned} \underset{r_n}{\text{minimize}} \quad & p_n \\ \text{subject to} \quad & p_n = \sum h_c(1 - \frac{gpa_c - GPA_{AVG}}{GPA_A}) \end{aligned} \tag{4c}$$

where $GPA_A = 4.0$ and $GPA_{AVG} = 2.0$, which follows the standard 4.0 grading system at most four-year universities.

3.1.2.3. Combined objective function. The objective functions are intended to quantify vastly different aspects of a semester schedule. Thus, one key design element of the algorithm is the normalization of each term to appropriately combine them into a single weighted sum for optimization. Combining Equations 2, 3d, 4c, and 5 with weights and normalization factors yields Equation 6, with weights that can be adjusted to obtain a unique and differently optimal solution.

$$
\begin{gathered}
minimize\ f(D,W) \\
where \\
f(D,W) = w_0\left(\frac{1}{N}\right)(L) + w_1\left(\frac{1}{N}\right)(i_c) \\
+w_2\sum\left(\frac{1}{CHL_n}\right)(p_n) + w_3\left(\frac{1}{|R|}\right)(|R| - v) \\
and \\
W = \{w_0, w_1, w_2, w_3\}
\end{gathered}
\tag{6}
$$

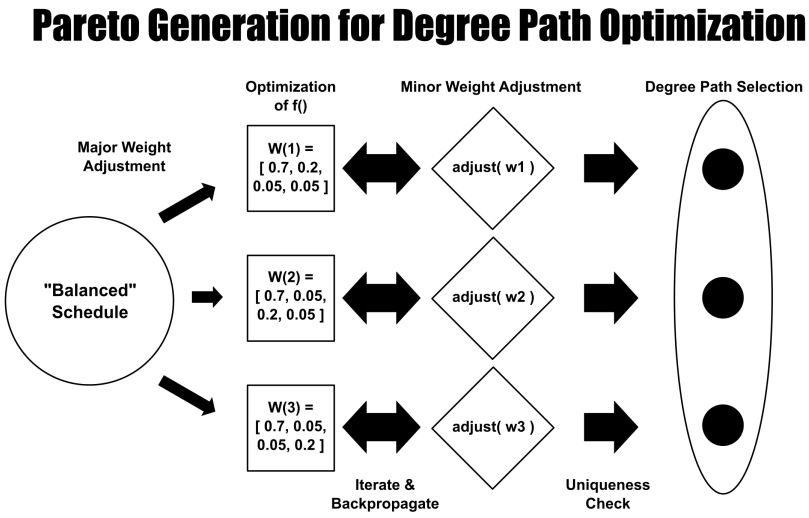

Figure 3.1. Pareto Optimization Diagram

3.1.2.4. Per-semester credit hour limit. Instead of having students define a single credit hour limit that applies to each semester, the new approach allows the student to set each semester's limit individually. This added control is especially useful for those students who have significant extracurricular commitments - clubs, design teams, leadership positions, part-time jobs, etc. - in some semesters but not others. The per-semester credit hour limit constraint is given in Equation 8.

$$h_n \leq CHL_n \ \forall\ n \leq N \tag{8}$$

3.1.2.5. Locked courses. In scenarios where students wish to see whether a certain selection of courses will keep them on track to graduate within a given timeframe, the recommender enforces additional "locked course" constraints to simulate enrollment in a future semester and make

recommendations accordingly. This constraint is defined in Equation 9.

$$SEL_n \subseteq r_n \; \forall \; n \leq N \tag{9}$$

3.1.3. Optimization Process. Each run of the optimization algorithm yields a singular degree path recommendation. The optimization algorithm processes multiple times with varying weights to generate the Pareto front of solutions for degree paths. The initial weights were set to 0.7, 0.1, 0.1, and 0.1, as these were empirically found to allow for adequate variance in degree paths without diluting the time-to-degree objective by an excessive amount. The algorithm adjusts weights two different ways. Major weight adjustment ensures that diverse schedules are achieved by selecting weights within distinct regions of the Pareto front and rerunning the optimization. Minor weight adjustment tweaks the weights within this region in smaller increments to determine the best score within that region. The normalized objective functions make the comparison of schedules with different weights possible. Once the system finds a unique solution from minor weight adjustment, the algorithm can prove the degree path's optimality before adding the degree path to the list of recommendations for the student. Degree paths are unique from one another if the preferred course selections, recommended credit hour count, and semester distributions are considerably different between each path. This process is depicted in Figure 3.1, and its pseudocode is shown in Algorithm 1.

3.2. IMPLEMENTATION

The above approach runs on the PERCEPOLIS system: a web-based platform with client-server topology. The front end of this system runs on Angular and allows students to input their explicit preferences, such as credit hour limit, preferred courses, and additional semesters to be considered, such as summer semesters. The back end houses the optimization algorithm and runs on Spring Boot and Kotlin. The chosen optimizer for this implementation is the CP-SAT solver from the Google OR-Tools package. The CP-SAT solver can handle large numbers of linear variables and constraints and provide proof of optimality, which is paramount to generating trustworthy schedules for students. Implementation of this approach was performed on an 8-core personal computer running at 3.5 GHz and 32 GB of RAM.

Algorithm 1 Algorithm for Pareto Optimization Using Iterative Weighted-Sum

Input: D, f(D, W)

Output: DP

Initialization :

$w_0 \leftarrow 0.7$

$w_1 \leftarrow 0.1$

$w_2 \leftarrow 0.1$

$w_3 \leftarrow 0.1$

$maxTime \leftarrow 240$

$timeout \leftarrow 30$

$numRuns \leftarrow 5$

$uniquenessThreshold \leftarrow 0.75$

$score \leftarrow 0$

$balancedSchedule \leftarrow optimize(f(D, W), maxTime)$

LOOP Process

for $i = numRuns$ to 0 **do**

  while $score \geq uniquenessThreshold$ **do**

    $W \leftarrow minorAdjust(W)$

    $newSchedule \leftarrow optimize(f(D, W), timeout)$

    $score \leftarrow diff(newSchedule, balancedSchedule)$

  end while

  $DP \leftarrow optimize(f(D, W), maxTime)$

  $adjust(W)$

end for

return *DP*

3.3. SUMMARY

The Methodology section detailed the objective functions and algorithmic approaches for generating a set of Pareto-optimal degree paths. The additional objectives significantly increased the complexity of the optimization problem, necessitating additional algorithms to improve system performance and retain the ability to prove optimality. The next section implements this approach within the PERCEPOLIS platform and validates the generated degree paths.

4. VALIDATION AND ANALYSIS

4.1. OVERVIEW

This approach is validated by first performing a series of optimizations that demonstrate the effect that degree path flexibility, predicted performance, and preferred course considerations have on degree path generation. Next, this work runs a second student through the optimization algorithm to generate four schedules: one that equally weights degree path flexibility, predicted performance, and courses of interest; one that emphasizes flexibility; one that emphasizes predicted performance; and one that emphasizes courses of interest. These courses are characterized as "balanced," "flexible," "grade-focused," and "topical." Each generated degree path was then confirmed to be feasible with all of the characteristics described previously, and their uniqueness was compared to validate the claim of distinct schedules. Lastly, the overall effect of the approach is evaluated in regards to performance, measured in time to feasible solution and time to optimality.

The following assumptions were made during the validation of this work:

- Past student data was simulated such that the set of similar students satisfied $k > 30$. The average GPAs of similar students for course c - and therefore the predicted performance for a given student - was predicted as follows:
  - Courses with a 5XXX course number were given a predicted grade of 2.5 on the standard 4.0 grading scale
  - Courses with a 4XXX or 3XXX course number were given a predicted grade of 3.0
  - Courses with a 2XXX course number were given a predicted grade of 3.5
  - Courses with a 1XXX course number were given a predicted grade of 4.0
  - Predicted grade point averages were deducted an amount equal to the hundreds place of the course number divided by 100
  - For example, any course with a 25XX number has a predicted grade of $3.5 - 0.05 = 3.45$
- To evaluate the availability objective, 5XXX courses with an odd second digit were assumed to become unavailable during the fall semester; 5XXX courses with an even, non-zero second digit were assumed to become unavailable during the spring semester. All other courses were assumed to be available every semester.

Table 4.1. Test cases

Test Case #	Degree Program	Required Hours		# "Hard" Required Courses	# Electives With x Options			
		Total	From Requirement Groups		$x < 5$	$5 \leq x < 20$	$20 \leq x < 50$	$x \geq 50$
1	Computer Engineering B.S.	128	125	30	5	3	8	1
2	Ceramic Engineering B.S.	128	128	31	2	1	7	3

- The minimization of credit hours per semester was not included in time-to-degree optimization during validation. This simulates the desire for greater semester utilization and does not necessarily imply increased cost - flat-rate tuition universities, for example, charge the same amount for a 12-credit hour semester as they would a 17-credit hour semester. As described below, the perceived course load objective tends to decrease the overall number of credit hours taken, which takes the place of the previous hours-per-semester objective. The explicit optimization of this parameter was implemented previously in [5] and can be reintroduced to the problem if desired by the student.
- Special permission courses, such as 4099 undergraduate research credits, are only available to the student if specified as a preferred course.
- Depicted degree paths show some - but not all - pre- and co-requisite relationships between courses. Any notable relationships that are not depicted are described in detail below.

4.2. TEST RESULTS

Table 4.1 summarizes the academic degrees and relevant requirement information for the first two test cases. The third test case is a comparison in performance between the single-objective optimization and multi-objective optimization approaches implemented in PERCEPOLIS.

4.2.1. Test Case 1: Validation of Objective Functions. Figure 4.1 shows the recommended degree path for a B.S. Computer Engineering student who provided no preferred or previously taken courses. This would be typical of a newly enrolled freshman with no prior college credits obtained. This degree path, while satisfying the requirements for earning a Computer Engineering degree, makes many arbitrary decisions about certain courses. For example, Art 3221 is an elective with many other options that can be taken in its place. Another example is Comp Eng (Computer Engineering) 5220 and Comp Eng 5510, which are relatively high-level courses that are being taken

very early on in the degree path. While not inherently a bad thing, this may not be preferred by a student in their early years, or they may want to swap them out for easier courses that still fill the degree requirements.

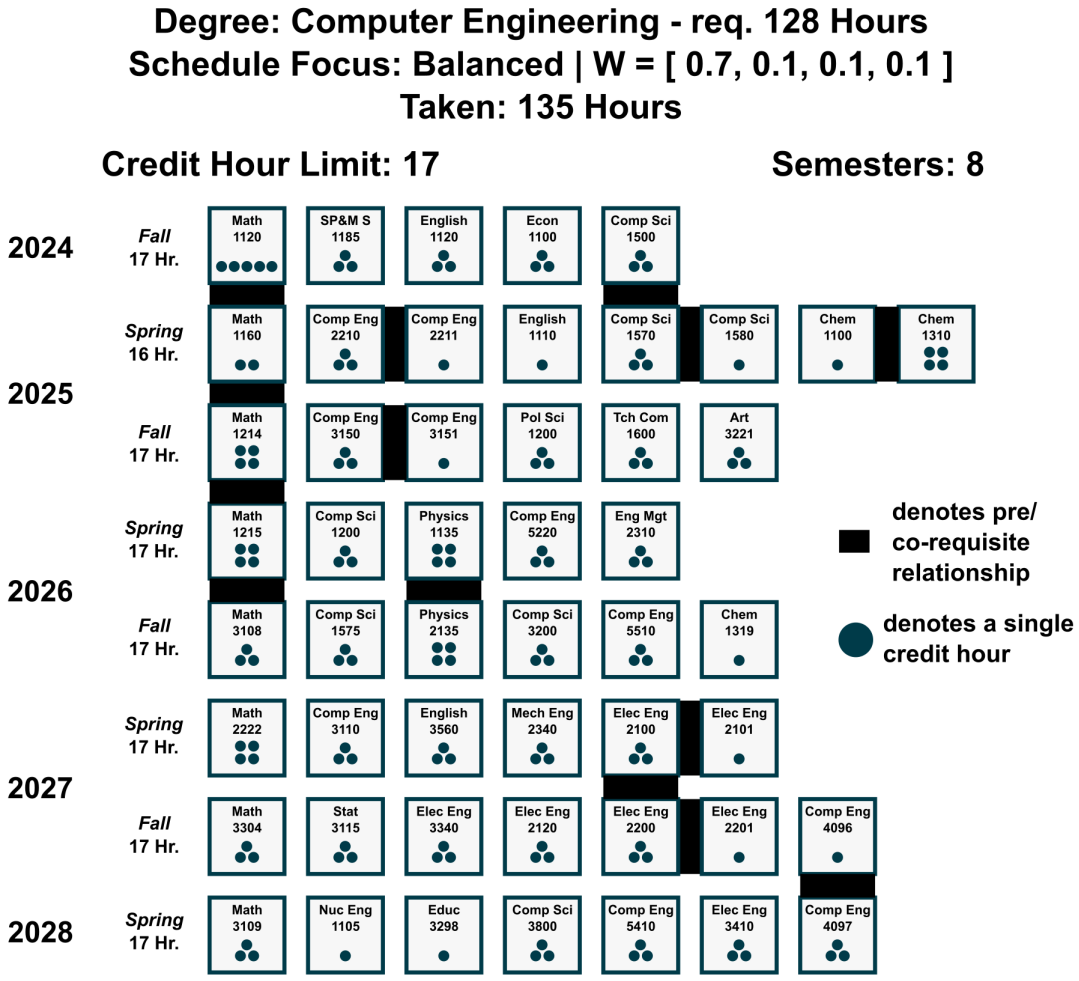

Figure 4.1. Recommended path for test case 1*

Test Case 1A (Figure 4.2) applies additional objectives to the degree path. This degree path demonstrates the effect of measuring course flexibility. Two required courses, Comp Sci (Computer Science) 1500 and Math 1120, are each the first prerequisite in paths of three or more required courses: Math 1120, 1160, 1214, 1215, and 2222 and Comp Sci 1500, 1570, 1575, and 2500, respectively. The rigidity of the degree path is minimized when such critical courses are taken earlier in the degree path; thus, they have been placed in the first semester to reduce the chances of delayed graduation

if a student fails to complete a subsequent course on time. Additionally, courses that consist of a single credit hour are inserted throughout the degree path, such as Music 1130 in semester four and Music 1136 and 1140 in semester eight. These courses are selected for two reasons. Firstly, they do not have any prerequisites and are available in every semester; thus, they yield a lower rigidity score than other options that rely on earlier courses and are therefore susceptible to delays. Secondly, these courses yield very high predicted performance as defined in this simulation, resulting in a lower perceived course load.

$$N_{min} = min(n) \mid n \geq \frac{CHR - H_p}{\frac{1}{n}\sum_{i=1}^{n} CHL_i} \tag{10}$$

Equation 10 was used to verify the minimum number of semesters for each degree path. In this case, the right side expression equaled 7.53 for all n. Thus, $n = 8$ is the lowest possible semester count that satisfies the inequality and signifies the minimum number of semesters to complete the degree. Note that, despite the nominal credit hour requirement being 128 hours, an additional seven hours that did not count towards any degree requirements or electives were taken to satisfy prerequisites for Math 1214.

Test Case 1B (Figure 4.3) shows the recommended degree path given that Music 1130 is specified as a preferred course. Music 1130 is especially pervasive, as it can be retaken multiple times and only takes up one credit hour. Therefore, taking this course at every possible opportunity maximizes courses of interest while leaving degree path robustness mostly unaffected, as the course is able to replace any arts/humanities and free electives. Since Music 1130 can only be taken once per semester, some courses were shifted to different semesters; however, the courses most important to degree path flexibility - Math 1120, Comp Sci 1500, and Comp Eng 2210 - were not shifted. The perceived course load is also not compromised, as all 1XXX have very high predicted performance and therefore a very low perceived course load.

Test Case 1C (Figure 4.4) demonstrates the ability of the algorithm to handle fluctuating credit hour limits set by the student. This aspect of personalization is especially helpful for students with varying work loads outside of academics, such as part-time jobs or other extracurricular commitments that only take place during some parts of the academic school year. In this example, the simulated student has an interest in Computer Architecture courses and sets a 15-credit hour limit on fall semesters and an 18-credit hour limit on spring semesters. This decision brings the right side of Equation 10 up to 7.75 for $n = 8$ and the nominal credit hour requirement of 128. However, as noted in Case 1A, this student has to account for an additional seven credit hours in order to

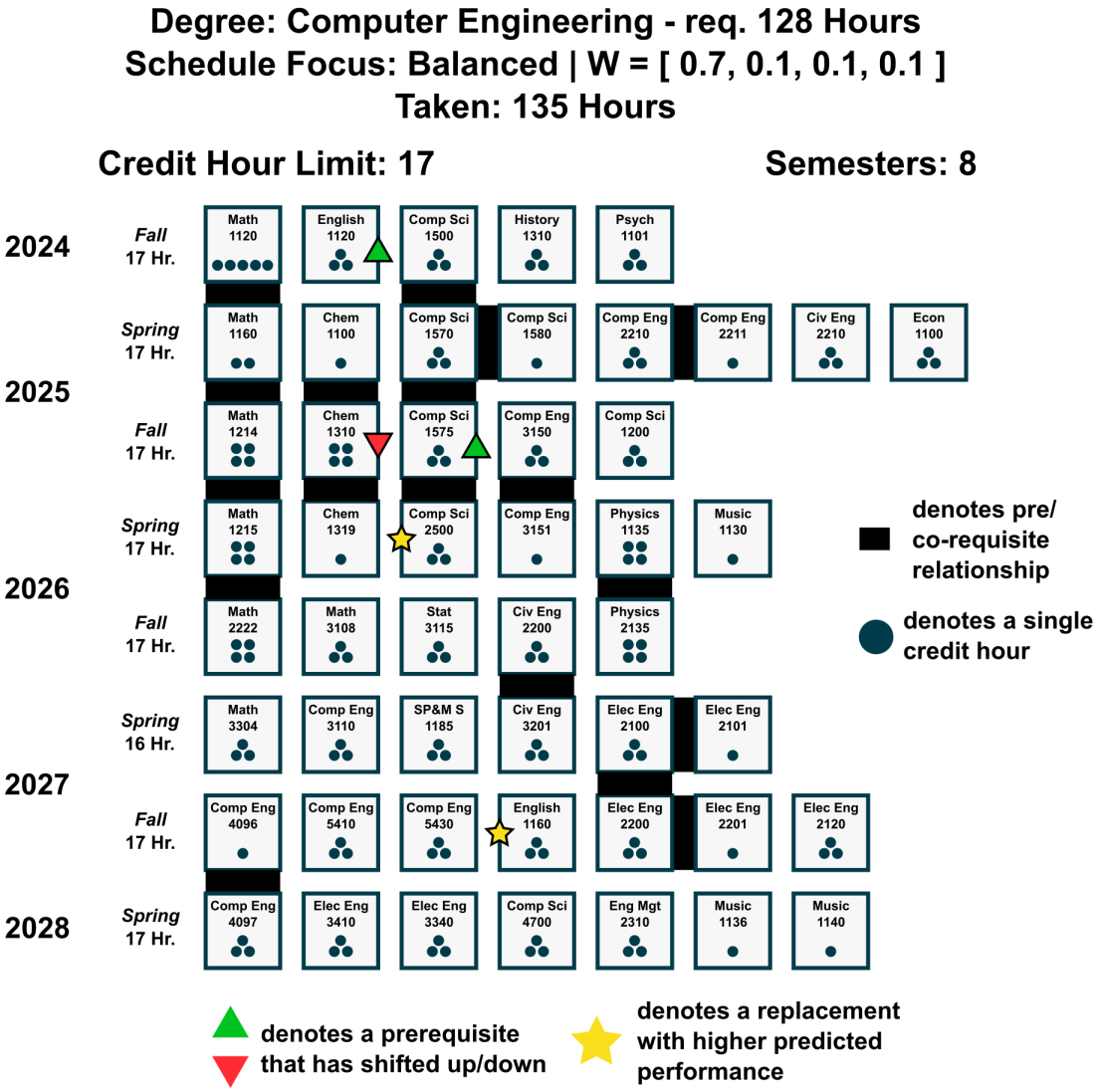

Figure 4.2. Recommended path for test case 1A

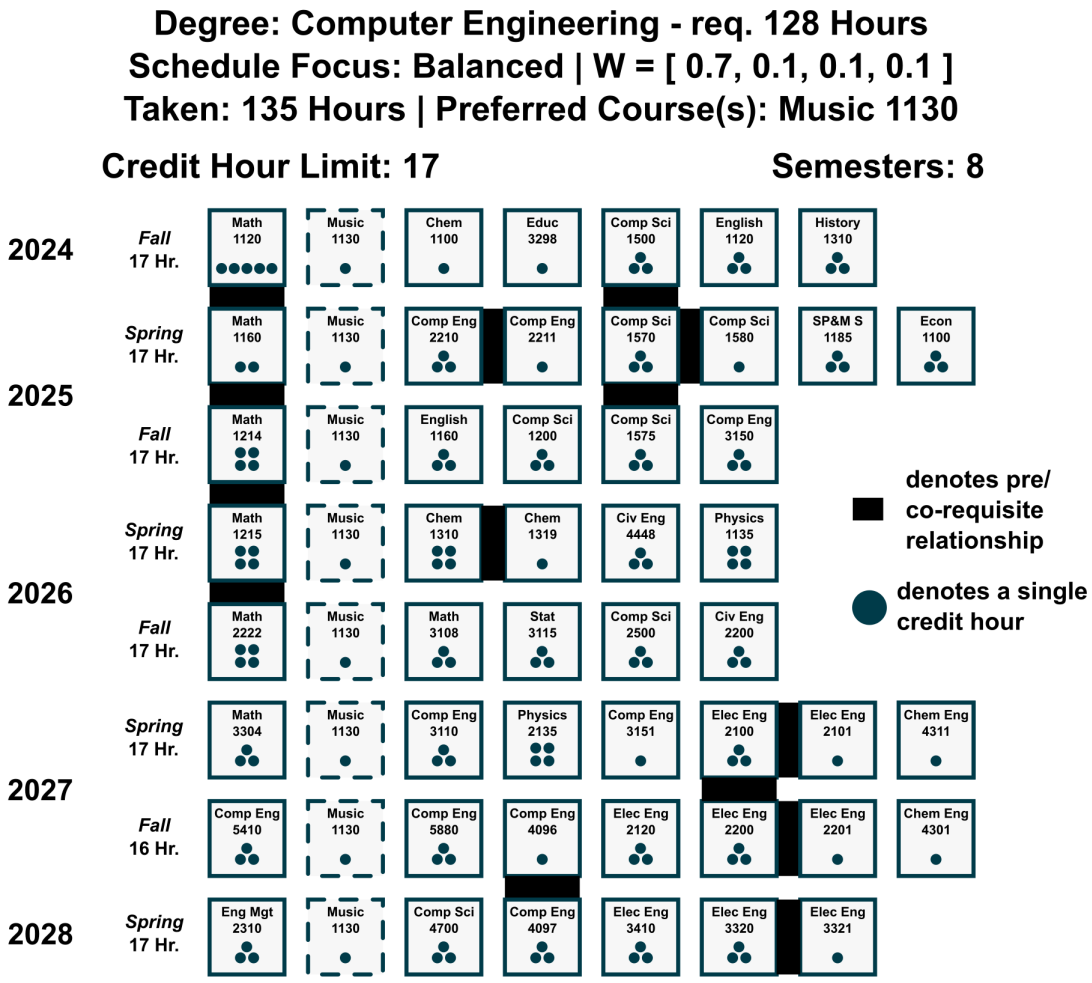

Figure 4.3. Recommended path for test case 1B

fulfill prerequisites, bringing the inequality up to 8.12 for $n = 8$ and necessitating a ninth semester. Note that the final semester, Fall 2028, is intentionally left incomplete with the assumption that a student would have to take additional courses to meet any minimum enrollment requirements. Adding these courses would result in an increase perceived course load and is therefore not considered in the optimization problem. Despite the fluctuating semester credit hour limits, the algorithm was able to schedule four courses related to Computer Architecture: Comp Eng 5160, Comp Eng 5151, Comp Eng 5110, and Comp Eng 5170.

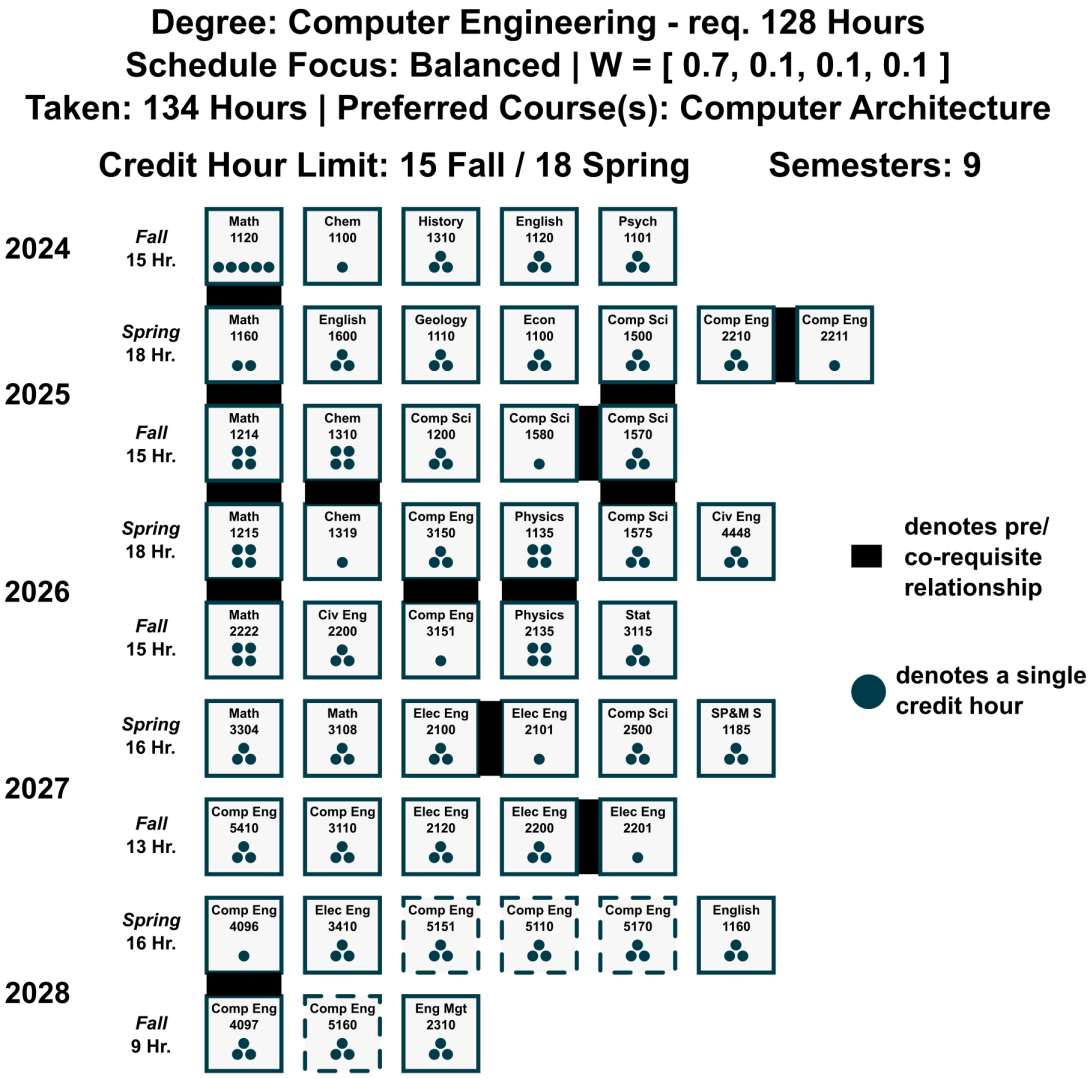

Figure 4.4. Recommended path for test case 1C

Test Case 1D (Figure 4.5) is a continuation of 1C in which the student decides to lock certain courses pertaining to embedded systems and regenerate the schedule with a new set of interests. Additionally, this student has received approval to take Math 1120 and Math 1160 before their first semester and undergraduate research - Comp Eng 4099 - throughout their degree path. By removing Math 1214's prerequisites from the degree path and allowing Comp Eng 4099 to be selected, the student is now able to complete their degree path in 8 semesters as with cases 1A and 1B. Similar to Music 1130 in Case 1A, Comp Eng 4099 has no prerequisites and is available in every semester. Comp Eng 4099 can be taken for more than one credit hour each semester; however, doing so would increase the perceived course load without making any improvements to the degree path's flexibility as defined in this model. Thus, Comp Eng 4099 is scheduled for one credit hour across multiple semesters instead of multiple credit hours to pad out shorter semesters. Case 1D also demonstrates the effect of both preferred courses and course-to-semester locked courses. Even though this student has expressed a new interest in computer networks, the locked positions of Comp Eng 5160 and Comp Eng 5151 already fill some of the required computer engineering-specific electives. Thus, only Comp Eng 5420 was selected as a course tailored to an interest in Computer Networks. Since this is the balanced model where flexibility, perceived course load, and student preference are equally prioritized, the optimizer opted not to schedule the additional course Comp Eng 5430, as doing so would have replaced certain 1XXX electives that score lower on both perceived course load and rigidity.

4.2.2. Test Case 2: Validation of Weight Variation for Pareto Optimization. Test Case 2 simulates the degree path generation for a ceramic engineering student in their second year, having taken the following courses prior to their third semester: Chem 1100, Chem 1310, Chem 1319, Chem 1320, Math 1214, Math 1215, English 1120, History 1200, Physics 1135, Mech Eng 1720, Econ 1100, and Music 1132. This student also showed interest in certain psychology classes: specifically, Psych 1101, 3501, and 4400. Figures 4.6 through 4.9 show the four schedules generated by the Pareto optimization algorithm with different weights. Test case 2A, shown in Figure 4.6, shows the balanced schedule generated for this student.

In test case 2B (Figure 4.3), the degree path's flexibility is given additional weight over the other two objectives. The most notable difference in this schedule is the significant increase in recommended credit hours. Since the perceived course load has much less effect on this schedule and credit hours per semester are not explicitly optimized in this model, this degree path maximizes the utilization of each semester. This, combined with the low number of credit hours per semester

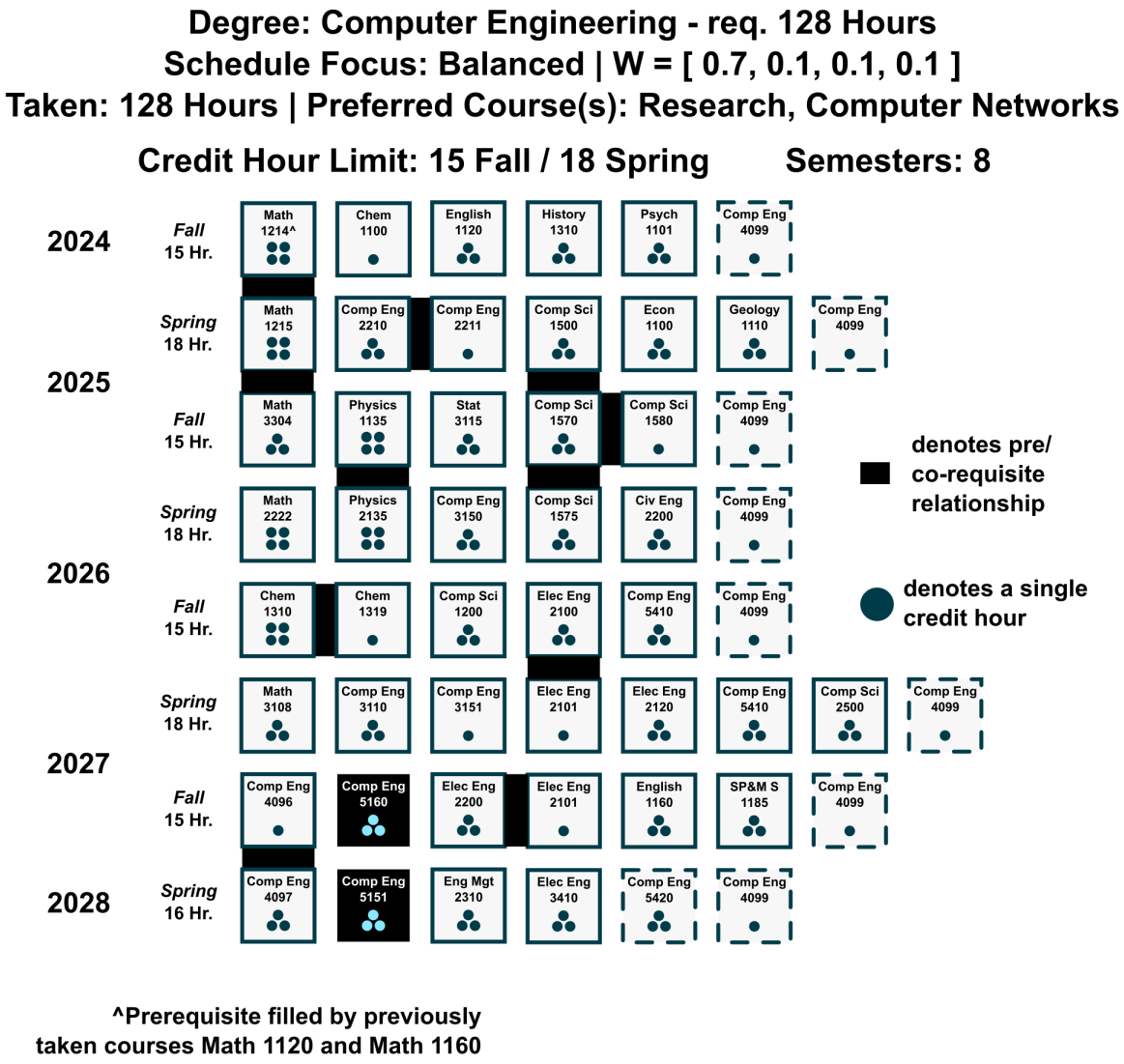

Figure 4.5. Recommended path for test case 1D

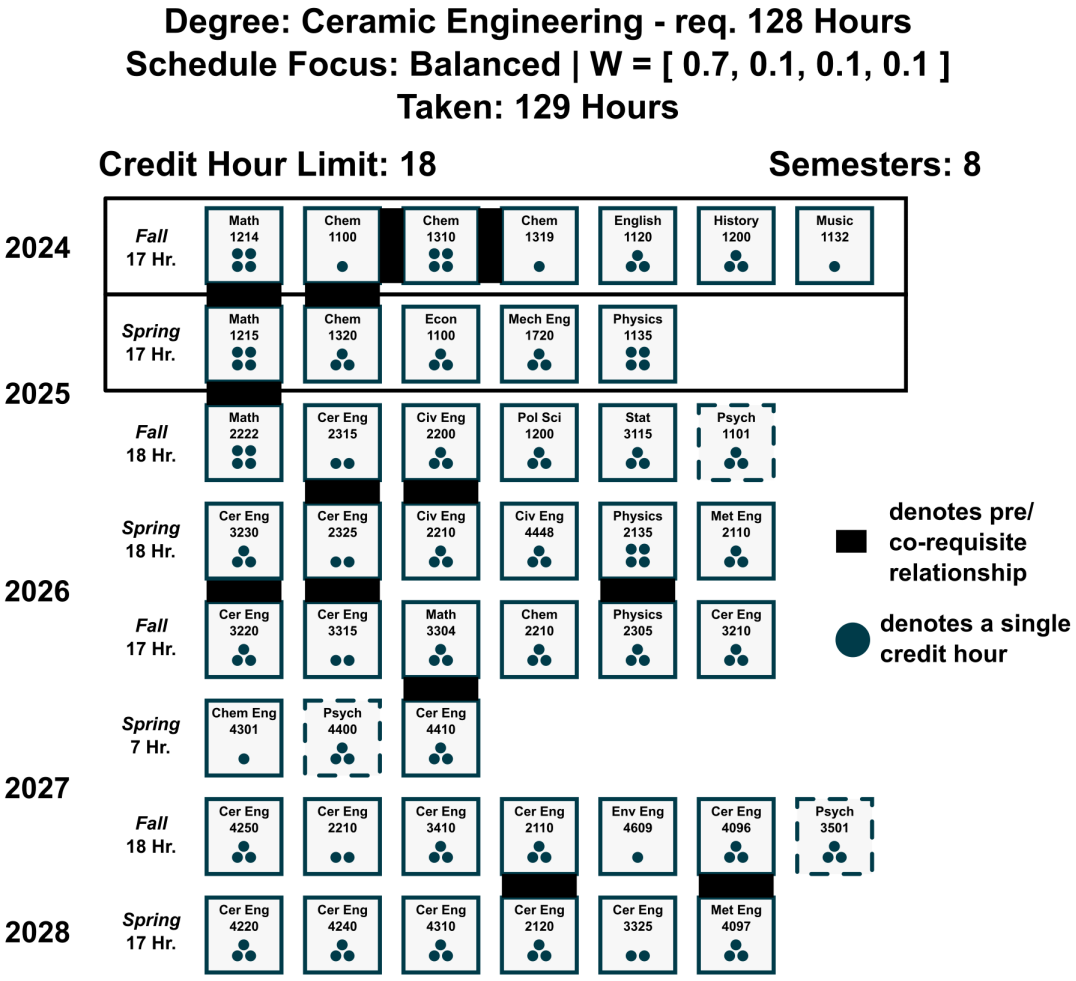

Figure 4.6. Recommended path for test case 2A

needed to reach the minimum 128 credit hours for the degree, leads to an additional 13 credit hours in the degree path. This minimizes degree path rigidity since more courses are scheduled that fill enrollment or degree requirements; thus, if one course becomes unavailable or is delayed for another reason, another course may fill that requirement instead and prevent any delays in graduation. This is shown with the set of Bio Sci courses: Bio Sci 1113, 22213, and 2219. These courses fill a science elective that is also covered by courses like Geology 1110 and Comp Sci 1972. In the event that one of these is not completed and counted towards degree completion, the others will take their place instead.

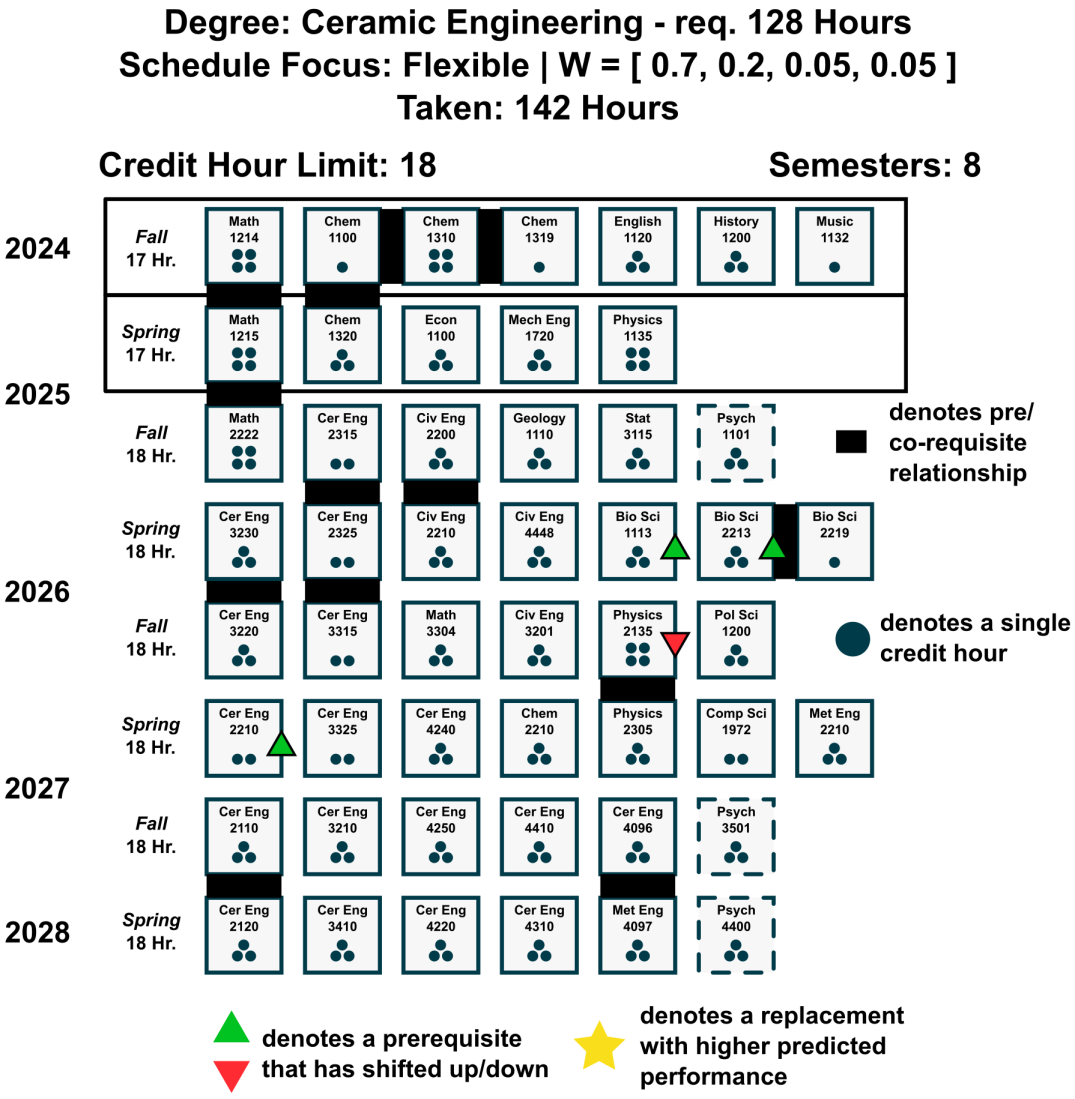

Figure 4.7. Recommended path for test case 2B

Figure 4.8 shows a schedule with academic performance prioritized. As with test case 2B, many of the courses remain the same since Ceramic Engineering requires a large amount of "fixed" course numbers. However, this set of weights makes a few key adjustments that distinguishes this degree path from the others. In this example, the student indicated a preference for two higher-level psychology classes: Psych 3501 and Psych 4400. In this simulation, Psych 3501 has a predicted grade of 2.95, and Psych 4400 has a predicted grade of 2.96. Since the "grade-based" schedule prioritizes selecting courses with high predicted academic performance over selections that conform to the student's preferred courses, these courses are replaced with History 2791 and Theatre 2141. These two courses have predicted grades of 3.43 and 3.49, respectively, which leads to a lower perceived course load. The "grade-based" degree path gives greater weight to perceived course load and makes this trade-off to minimize the overall score of the degree path, leading to another unique schedule.

The results of test case 2D, represented in Figure 4.9, demonstrates the possibility of objective prioritization having little effect on the generated schedule. In test case 2A, the optimizer was able to schedule all preferred courses without sacrificing the "balance" of the other two objectives: degree path flexibility and perceived course load. This was partly due to the low number of preferred courses provided by the student. Since these courses can only be taken once, the optimizer cannot improve on the solution in case 2A in terms of the preferred course objective - any differences in course distribution between cases 2A and 2D are due to symmetries in the other two objectives and a possible loss in precision due to the lower weights. Thus, the resulting schedule is just as effective as the one in case 2A, rendering this recommendation unhelpful for the student. The low uniqueness of this schedule would result in the optimizer not recommending this schedule to the student as a distinct option.

4.2.3. Performance Analysis. The added complexity of this model has a performance trade-off with the CP-SAT solver. The increase in time-to-optimality is mostly affected by the restrictiveness of the degree requirements themselves. Test case 1, which has more free electives and less linearity in degree path requirements, has its performance affected heavily by the added complexity of this approach's additional objectives. Between test cases 1* and 1A, the time to reach optimality increases by almost five times, as shown in Figures 4.10 and 4.11. Similarly, the largest score drop happens approximately seven minutes later in test case 1A when compared to test case 1*. Test cases 1B-1D also suffer similarly large performance drops.

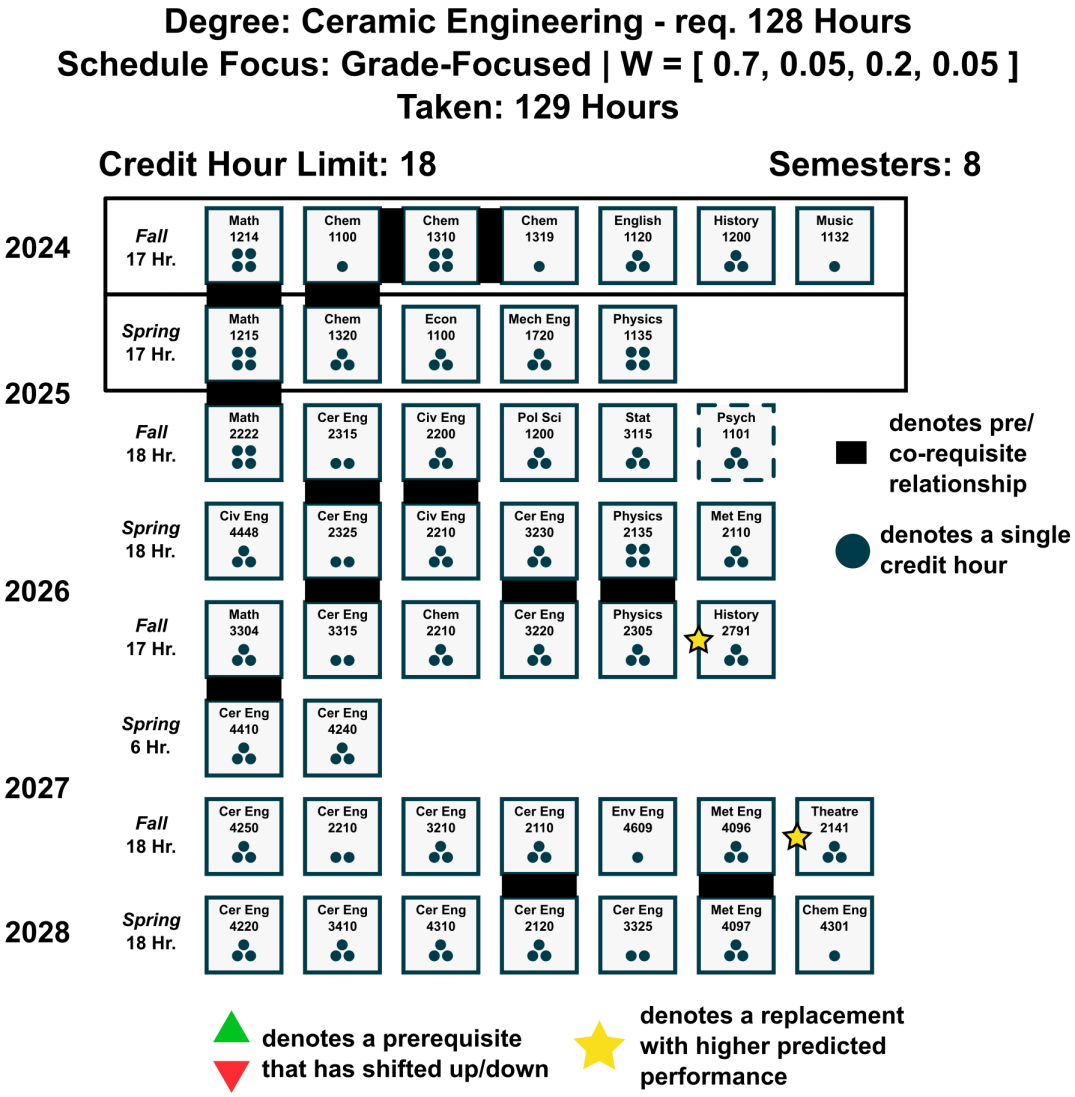

Figure 4.8. Recommended path for test case 2C

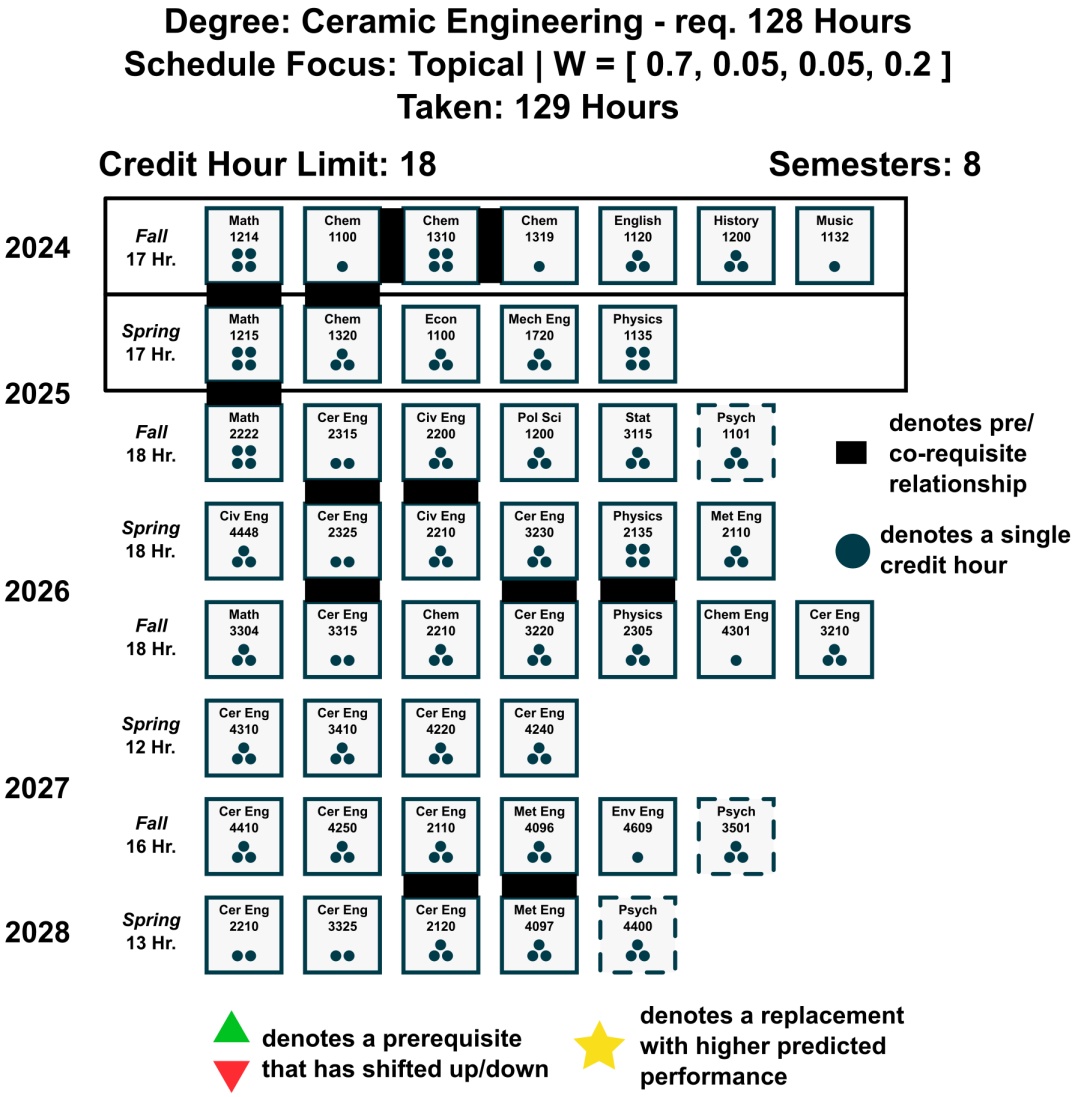

Figure 4.9. Recommended path for test case 2D

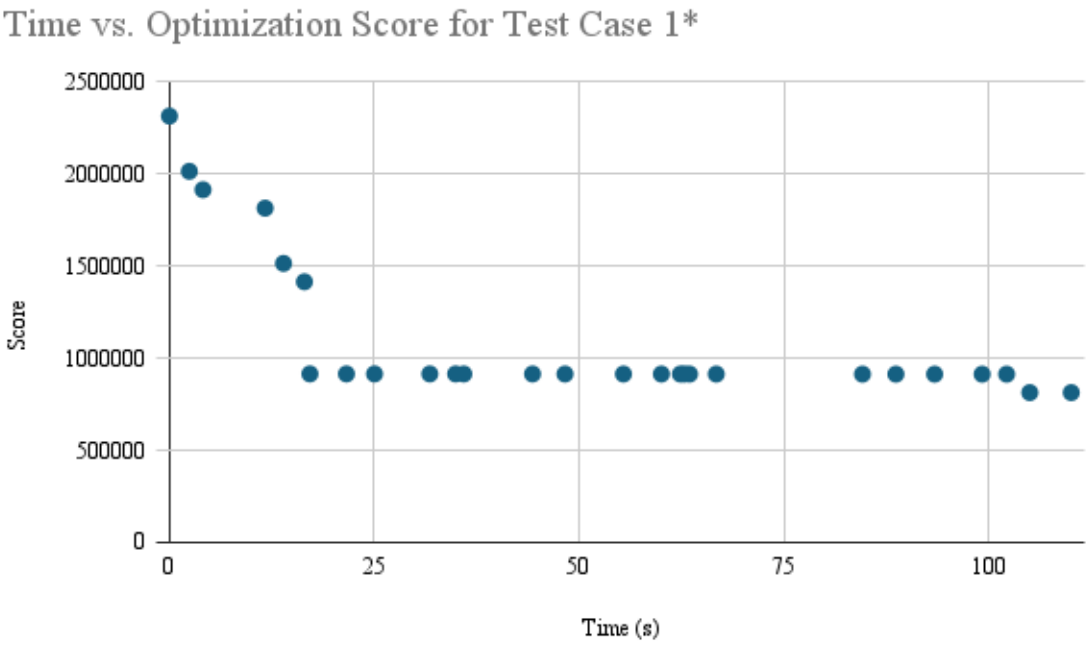

Figure 4.10. Plot of test case 1* score convergence

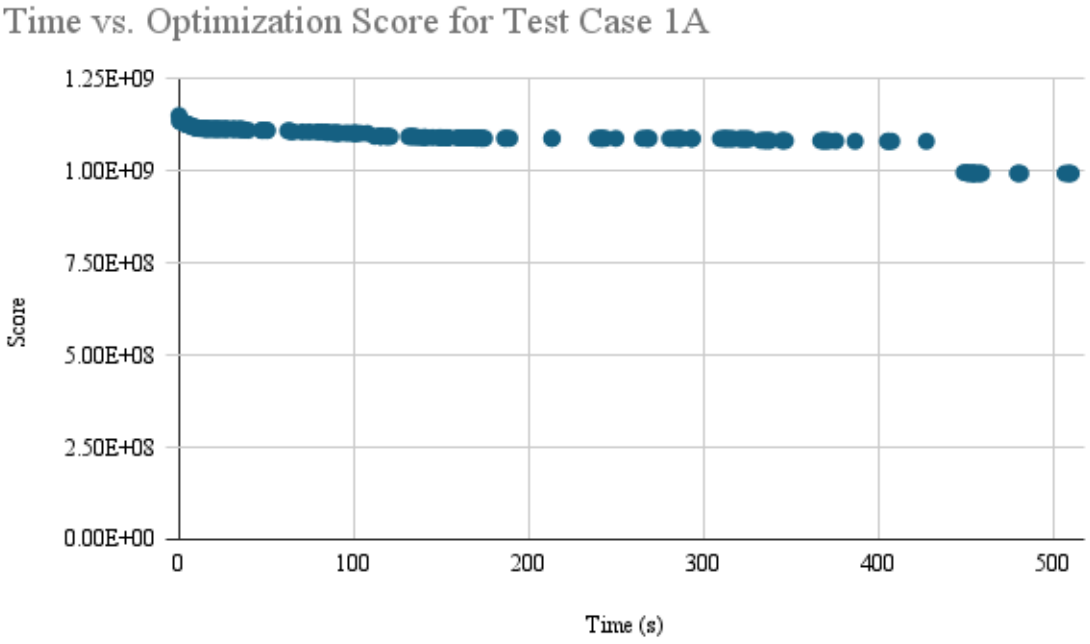

Figure 4.11. Plot of test case 1A score convergence

For this analysis, test case 2 was run with the original approach from [5] in test case 2*. Test case 2, in contrast with its predecessor, is significantly more restrictive with its choices in both elective selection and scheduling. As a result, the difference in performance between the two cases is significantly less dramatic. Figures 4.12 and 4.13 show the similarity in convergence time between the original approach and the new approach with balanced weights. Test case 2A only took 34.78 seconds to drop very near the optimal score, which is only 9.33 seconds slower than test case 2*. Similarly, the time to prove optimality increased by a minute as opposed to the seven minutes from test case 1. This demonstrates the affect of a priori information on the system. Providing additional information, such as a specific semester for a locked course or a larger number of preferred courses, can help the solver make decisions and converge on a solution much more quickly. Both cases demonstrate the much smoother descent to the minimum score that the proposed method provides compared to the original approach.

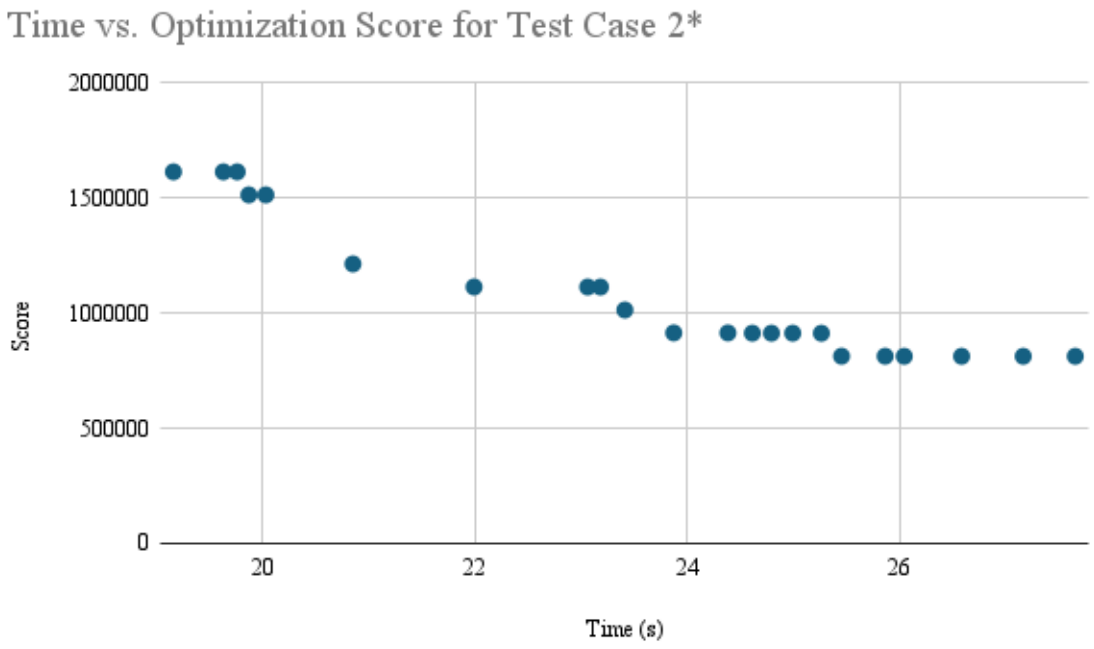

Figure 4.12. Plot of test case 2* score convergence

4.3. SUMMARY

The recommendation system was shown to successfully recommend multiple unique schedules, each with distinct priority that reflects a student's enrollment needs and preferences. The effects of each objective on the overall degree path were also demonstrated by adjusting the weights within

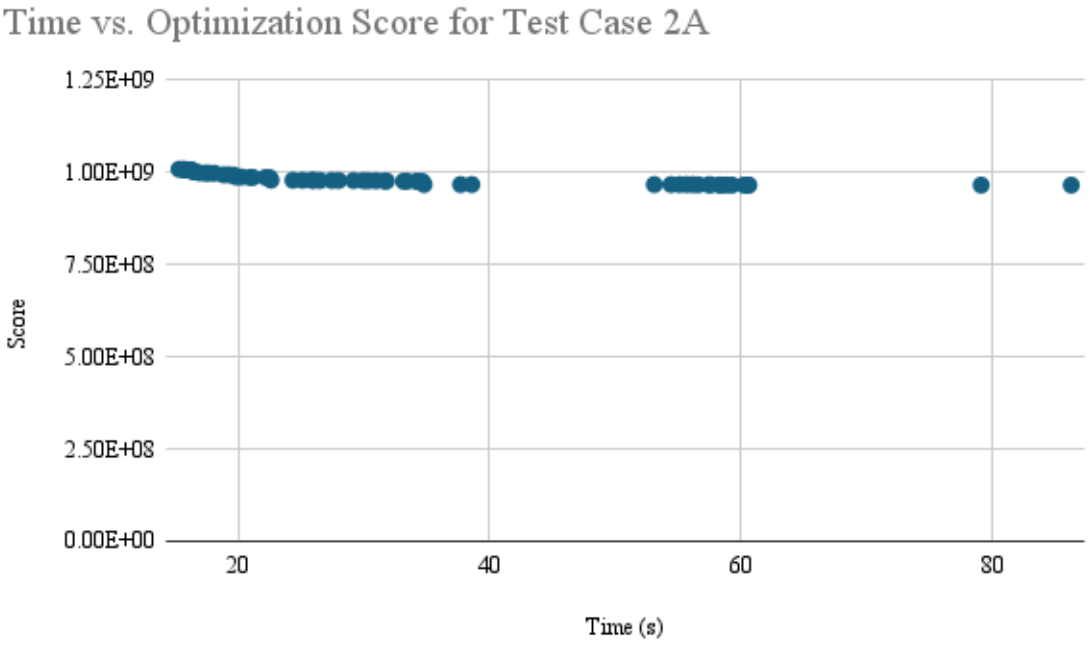

Figure 4.13. Plot of test case 2A score convergence

each region of the Pareto front. The performance trade-off of this algorithm was shown to be less severe when running a stricter degree path, showing that providing additional hints and constraints to the system can improve overall execution time.

5. CONCLUSION AND FUTURE DIRECTIONS

The work presented in this paper showed the efficacy of a Pareto optimization algorithm in recommending a variety of degree paths. The increased personalization was shown to appropriately place courses in ways that prioritized important academic factors such as predicted performance, student interests, and enrollment flexibility. Though the additional complexity introduced by such factors had a significant impact on algorithm performance, further testing showed that the generated solutions were still feasible and comparable to a true optimal solution without sacrificing the ability to prove optimality altogether.

An approach to find multiple solutions to a job scheduling problem is applicable to many other fields, including supply chain management and resilience analysis of interdependent systems. As with a degree path, the time to job completion may not be strictly more important than the ability of the job to be robust in the case of unexpected delays or failures. Thus, generating solutions that are optimized with both time-to-completion and per-component importance in consideration may be necessary to capture the trade-off between these objectives.

A central avenue of future work in developing Pareto optimization of degree paths is the implementation of more accurate objective functions, particularly in the field of performance prediction for calculating perceived course load. Such improvements include analyzing the effect that breaks between prerequisite courses and future courses has on academic performance and balancing predicted performance across semesters. Additionally, the performance of the adjusted weighted-sum optimization has significant room to improve. More constraints and hints to the problem space can also be implemented to speed up both convergence time and time to prove optimality.

REFERENCES

[1] National Academy of Engineering, 'Grand Challenges for Engineering,' Technical report, 2008.

[2] Lynn, N. D. and Emanuel, A. W. R., 'A review on Recommender Systems for course selection in higher education,' IOP Conference Series: Materials Science and Engineering, March 2021, **1098**(3), p. 032039, ISSN 1757-8981, 1757-899X, doi:10.1088/1757-899X/1098/3/032039.

[3] Zhang, X., Gossett, C., Simpson, J., and Davis, R., 'Advising students for success in higher education: An all-out effort,' Journal of College Student Retention: Research, Theory & Practice, January 2017, **21**(1), pp. 53–77, doi:10.1177/1521025116689097.

[4] Walker, C., 2024.

[5] Dobbins, N., Sedigh Sarvestani, S., and Hurson, A. R., 'Personalizing student graduation paths using expressed student interests,' in 'Proceedings of the 47th IEEE International Computers, Software, and Applications Conference (COMPSAC 2023),' Torino, Italy, June 2023 pp. 142–143, doi:10.1109/COMPSAC57700.2023.00027.

[6] Denley, T., 'How predictive analytics and choice architecture can improve student success,' Research & Practice in Assessment, 2014, **9**(2), pp. 61+.

[7] Heileman, G. L., Abdallah, C. T., Slim, A., and Hickman, M., 'Curricular analytics: A framework for quantifying the impact of curricular reforms and pedagogical innovations,' ArXiv, 2018, **abs/1811.09676**.

[8] Javed, U., Shaukat, K., A. Hameed, I., Iqbal, F., Mahboob Alam, T., and Luo, S., 'A Review of Content-Based and Context-Based Recommendation Systems,' International Journal of Emerging Technologies in Learning (iJET), February 2021, **16**(03), p. 274, ISSN 1863-0383, doi:10.3991/ijet.v16i03.18851.

[9] Yang, Y., Zhang, C., Song, X., Dong, Z., Zhu, H., and Li, W., 'Contextualized knowledge graph embedding for explainable talent training course recommendation,' ACM Trans. Inf. Syst., sep 2023, **42**(2), ISSN 1046-8188, doi:10.1145/3597022.

[10] Dien, T. T., Hoai-Sang, L., Thanh-Hai, N., and Thai-Nghe, N., 'Course recommendation with deep learning approach,' in T. K. Dang, J. Küng, M. Takizawa, and T. M. Chung, editors, 'Future Data and Security Engineering. Big Data, Security and Privacy, Smart City and Industry 4.0 Applications,' Springer Singapore, Singapore, ISBN 978-981-33-4370-2, 2020 pp. 63–77.

[11] Hellas, A., Ihantola, P., Petersen, A., Ajanovski, V. V., Gutica, M., Hynninen, T., Knutas, A., Leinonen, J., Messom, C., and Liao, S. N., 'Predicting Academic Performance: A Systematic Literature Review,' in 'Proceedings ACM Conf on Innovation and Technology in Computer Science Education,' 2018 pp. 175–199, doi:10.1145/3293881.3295783.

[12] Premalatha, M. and Viswanathan, V., 'Course sequence recommendation with course difficulty index using subset sum approximation algorithms,' Cybernetics and Information Technologies, 2019, **19**(3), pp. 25–44, doi:doi:10.2478/cait-2019-0024.

[13] Wicaksono, F. and Putra, B. P., 'Course Scheduling Information System Using Genetic Algorithms,' ITEJ (Information Technology Engineering Journals), August 2021, **6**(1), pp. 35–45, ISSN 2548-2157, 2548-2130, doi:10.24235/itej.v6i1.55.

[14] Imran Hossain, S., Akhand, M., Shuvo, M., Siddique, N., and Adeli, H., 'Optimization of university course scheduling problem using particle swarm optimization with selective search,' Expert Systems with Applications, 2019, **127**, pp. 9–24, ISSN 0957-4174, doi:https://doi.org/10.1016/j.eswa.2019.02.026.

[15] Morrow, T., Hurson, A. R., and Sedigh Sarvestani, S., 'Algorithmic decision support for personalized education,' in 'Proceedings of the 17th IEEE International Conference on Information Reuse and Integration (IRI 2016),' Pittsburgh, USA, July 2016 pp. 188–197, doi:10.1109/IRI.2016.32.

[16] Morrow, T., Hurson, A. R., and Sedigh Sarvestani, S., 'A multi-stage approach to personalized course selection and scheduling,' in 'Proceedings of the 18th IEEE International Conference on Information Reuse and Integration (IRI 2017),' San Diego, USA, Aug. 2017, invited pp. 253–262, doi:10.1109/IRI.2017.58.

[17] Morrow, T., Hurson, A. R., and Sedigh Sarvestani, S., 'Algorithmic support for personalized course selection and scheduling,' in 'Proceedings IEEE Intl Computers, Software, and Applications Conf (COMPSAC),' 2020 pp. 143–152, doi:10.1109/COMPSAC48688.2020.00027.

[18] Kasapidis, G. A., Paraskevopoulos, D. C., Repoussis, P. P., and Tarantilis, C. D., 'Flexible Job Shop Scheduling Problems with Arbitrary Precedence Graphs,' Production and Operations Management, November 2021, **30**(11), pp. 4044–4068, ISSN 1059-1478, 1937-5956, doi:10.1111/poms.13501.

[19] Shen, J. and Lipasti, M., *Modern Processor Design: Fundamentals of Superscalar Processors*, McGraw-Hill, 2003, ISBN 978-0-07-123007-0.

[20] Guerreiro, A. P., Cortes, J., Vanderpooten, D., Bazgan, C., Lynce, I., Manquinho, V., and Figueira, J. R., 'Exact and approximate determination of the Pareto front using Minimal Correction Subsets,' Computers & Operations Research, 2023, **153**, p. 106153, ISSN 0305-0548, doi:https://doi.org/10.1016/j.cor.2023.106153.

[21] Keenan, C., Chhatwal, G., and Wang, J., 'A study of the effects and policy implications of flat rate tuition,' INTERNATIONAL JOURNAL OF RESEARCH IN EDUCATION METHODOLOGY, 08 2015, **6**, pp. 837–849, doi:10.24297/ijrem.v6i2.3879.

[22] Woodard, M., Marashi, K., Sedigh Sarvestani, S., and Hurson, A. R., 'Survivability evaluation and importance analysis for cyber–physical smart grids,' Reliability Engineering & System Safety, 2021, **210**, p. 107479, ISSN 0951-8320, doi:10.1016/j.ress.2021.107479.

VITA

Arianna Gail Sy Chaves earned her Bachelors of Science in Computer Engineering from Missouri S&T in May 2023. As an undergraduate student and S-STEM Scholar, she worked with Dr. Sahra Sedigh Sarvestani, Dr. Alireza Hurson, and other student researchers on the PERCEPOLIS project. This work carried into her master's program as she continued her education at Missouri S&T, earning her Masters of Science in Computer Engineering in May 2024. She also worked on Aviation UI systems starting in 2022 as an embedded software engineering intern at Garmin International, where she was later hired as a full-time software engineer in 2024.