12-2022

A Two-Phase Multicommodity Flow Approach for Classroom Assignment

Hannah Smith
University of Tennessee, Knoxville, hsmit118@vols.utk.edu

To the Graduate Council:

I am submitting herewith a thesis written by Hannah Smith entitled "A Two-Phase Multicommodity Flow Approach for Classroom Assignment." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Industrial Engineering.

James Ostrowski, Major Professor

We have read this thesis and recommend its acceptance:

Mingzhou Jin, William Dunne

Accepted for the Council:
Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

A TWO-PHASE MULTICOMMODITY FLOW APPROACH FOR CLASSROOM ASSIGNMENT

A Thesis Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

Hannah O. Smith
December 2022

ACKNOWLEDGEMENTS

I want to express my deepest gratitude to my advisor, Dr. James Ostrowski, for his guidance throughout the preparation and presentation of my work. I would also like to thank my committee members, Dr. Mingzhou Jin and Dr. William Dunne for their counsel throughout this process. Additionally, I would like to thank Dr. Ostrowski and the Classroom Upgrade Committee for funding this project and my education at The University of Tennessee. The success of this project would not have been possible without the professors who provided me with the resources to complete my thesis. I would also like to thank my colleagues for their support and encouragement throughout my time here at The University of Tennessee.

ABSTRACT

A common problem faced by universities is the assignment of courses to campus-hosted spaces. An optimal solution is difficult to reach, given the number of constraints present. Increasing student enrollment across campus leads to a need for a model that optimally assigns courses to spaces that maximizes the total number of courses taught in person. This paper poses a solution to this problem by generating a two-phase model to allocate courses to campus-hosted spaces. The two-phase approach to classroom assignment consists of a minimum-weighted bipartite matching for priority assignment and a multicommodity flow model to assign remaining courses. The optimal solution minimizes the walking distance between classrooms for professors teaching back-to-back courses and the walking distance between classrooms and professors' offices. The proposed model also minimizes the excess capacity of used campus-hosted classrooms and considers priority assignment. The model was tested on Spring 2022 course data from The University of Tennessee. The results of this model show a decrease in professor walking distance between courses and offices, as well as a decrease in percent excess capacity across used campus-hosted classrooms. With an optimal assignment of courses to campus-hosted classrooms, university resources can be used in a more optimal way that can increase student and professor retention rates.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION AND BACKGROUND

A common problem faced by every university each semester is determining which courses to assign to a set of classrooms across campus. The assignment process is time-consuming and requires many capacity and capability constraints to be met that are crucial to the success and learning experience of students. Capacity considerations must be considered for every course to ensure students have a seat in a given classroom, in addition to ensuring that not too much space is wasted across used classrooms. Little research has been done on classroom assignment and innovative assignment processes for large universities are almost nonexistent. This paper explores a two-phase multicommodity flow model that changes the way courses are assigned to campus-hosted spaces in a way that best utilizes the space available to students and enhances the college experience for both students and professors.

This paper will focus on an integer programming approach to the classroom assignment problem. One of the common integer programming models includes the multicommodity flow problem. An integer program was developed that includes an adaptation of the model commonly used for multicommodity flow to find an optimal assignment of courses to a set of campus-hosted classrooms that minimizes wasted capacity of these rooms across campus and minimizes the walking distance between rooms for professors who teach back-to-back courses.

This project intended to find a method to optimally assign courses to a campus-hosted classroom based on several requests from the university's registrar's office. The

current process of assigning rooms is not robust and does not allow individual professor requests. The Classroom Upgrade Committee developed this project in hopes of finding a new way to assign courses to campus-hosted classrooms in a way that would minimize the wasted classroom space on campus and would be robust to any last-minute classroom changes. Common complaints from students, professors, and the Registrar's Office regarding classroom assignment include professors having a significant walking distance between courses or from their office to the assigned classrooms, upper-level courses located too far from their home department, the assigned classroom does not have the desired technology or layout requested by the professor, and the inefficiency of the current assignment process. In addition, due to a growing student population and classroom renovations recently, many courses have had to be forced to meet virtually instead of in person. This online meeting environment could negatively affect a student's overall experience and decrease retention rates. Thus, a method of classroom assignment needs to be developed that effectively uses classrooms to prevent courses from being moved to a virtual setting within weeks or days before the first day courses meet. The resulting model is an application of both a minimum-weighted bipartite matching and the well-known multicommodity flow problem to create a two-phase approach to the classroom assignment problem, which will be explored throughout this paper.

Given this context, this thesis begins with an introduction, including a literature review of similar problems solved using various optimization methods and an in-depth overview of bipartite graph matching and the multicommodity flow problem. The methodologies section explains the steps taken to generate the graphs for the bipartite

matching and multicommodity flow models, as well as a description of the integer programs to be solved. The results section discusses the model's output and the improvements made with the new assignment process. The conclusion discusses the impact of this project and a proposal for future work that can be done to increase the efficiency of the assignment process even greater.

The classroom assignment problem has been tackled previously, but not with a multicommodity flow approach. Previous research includes integer programming methods and heuristics to solve the classroom assignment problem. Phillips et al. approach the problem with integer programming methods, as well as heuristics [1]. They propose a lexicographic optimization approach that includes maximizing the number of courses assigned to a particular room, maximizing the total number of hours a student spends in a given room, maximizing seat utilization, maximizing the number of room preferences from professors that are fulfilled, and maximizing the robustness of classroom assignment. This approach is a multi-tiered optimization approach, where each objective function value is used as a constraint in the next tier of optimization. This approach yields the solution's upper bounds being preserved and solved quicker than a similar linear program. It has a high modeling power, can handle multiple competing quality measures, and is scalable to larger problems. This paper, however, does not address distance from an academic department for a given course.

Another approach to the classroom assignment problem includes a zero-one integer programming model. Waterer proposes that the classroom assignment problem can be solved by minimizing a total cost function that minimizes the distance between courses and

academic departments, minimizes seats wasted, and gives priority to the seniority of the course [2]. The proposed model only considers standard time blocks during a given day, making it a simple approach to the classroom assignment problem. The results of this model led to decreased time in the classroom assignment process.

A parallel problem to the classroom assignment problem includes assigning classrooms for exam schedules. Similar to the classroom assignment problems discussed above, the goal is to assign all courses to a room during their final exam time without having two exams scheduled in the same classroom simultaneously. Dammak et al. propose a zero-one integer programming model and a transportation model to solve the classroom assignment problem for exam timetabling [3]. The formulation for this problem is relatively simple and has no objective function. The only constraints are to ensure that each exam is scheduled in a room with a capacity greater than the number of students taking the exam and ensure only one exam is assigned to each room at any given time. To solve this problem, a heuristic was implemented that first involves assigning exams in descending order of capacity. If a point is reached where the capacity of the exam is greater than the capacity of the remaining rooms, the exam is split into two different rooms. This process is then repeated until a feasible solution is found. If there is no feasible solution, an exam is assigned to the room with the greatest positive residual between capacities. If there is no room for an exam to fit in any room with excess capacity, the exam is split into different groups and assigned to rooms with enough capacity. The result will either be one exam per classroom, at most two exams per classroom or no feasible solution. The results of this study include an algorithm that should solve the classroom assignment problem for exam-

timetabling in a relatively small amount of time and aims to find a feasible solution that only assigns up to two exams per classroom.

Elloumi et al. also propose a solution to the classroom assignment problem for exam timetabling [4]. They show that the problem of classroom assignment without standard time blocks belongs to the class of NP-hard problems and propose two reduction methods to reduce the size of the problem. The model's performance was evaluated by calculating the sum of total capacity across all assigned rooms and the total number of exams still left to be assigned post-reduction and comparing this value to a calculated result from a Variable Neighborhood Search (VNS) algorithm. The first reduction is a reduction by ordering technique, which is similar to the method proposed by Dammak et al. above. The second reduction technique is a reduction by dominance criterion technique. Exams are divided into two sets and similar exams are assigned to the same room. After these two size reduction techniques are performed, a VNS algorithm is implemented to solve the classroom assignment problem for exam timetabling. The results from this model show that the heuristic obtains a relatively low objective value compared to the lower bound. It should be noted that in both of these cases, an exam occurring is a one-time event. This problem is a simpler version of the classroom assignment problem, where each event, which is a course in this situation, can take place over multiple days. Thus, room availability needs to be considered for up to 5 days per course instead of one day.

In addition, a handful of other methods have been proposed for the classroom assignment problem. Carter proposes a Lagrangian relaxation approach to solve the multi-period assignment problem [5]. Martinez-Alfaro et al. propose a simulated annealing

technique to solve the classroom assignment problem [6]. However, the research in the field of classroom assignment is sparse compared to other assignment problems. The problem presented in this paper is unique to the classroom assignment field, as it considers the walking distance between classes and individual offices for professors as well as capacity constraints. The multicommodity flow approach has not been proposed in the literature for classroom assignments. This project aims to present an effective method for classroom assignment that considers more than the capacity constraints of the classrooms.

In addition to considering the previous work done in classroom assignment, it is important to evaluate the research conducted in bipartite graph matching and multicommodity flow. The following subsection gives an overview of bipartite graphs and how the matching process works, as well as a background of multicommodity flow and applications of both methods.

A bipartite graph is a type of graph that consists of two sets of nodes and arcs connecting these two sets of nodes. The two sets of nodes are disjoint, meaning no nodes are adjacent in the same set. For example, if sets U and V are sets of disjoint nodes, then every node in U is connected to a node in V. Figure 1.1 displays an example of a bipartite graph. Once a graph is developed, a set of matchings can be found. A matching is a set of arcs that do not share common nodes. A maximum matching is the maximum number of arcs that do not share a common node. Figure 1.2 displays an example of a maximum matching for a bipartite graph, where all red arcs are elements of the set of the maximum matching. There could be multiple maximum matchings for a given bipartite graph.

To find a maximum matching of a bipartite graph, the following integer program

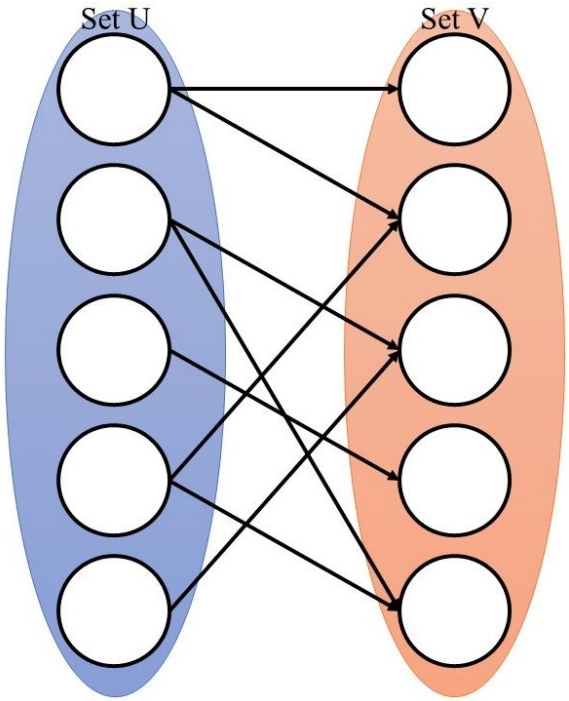

Figure 1.1: Bipartite Graph Example

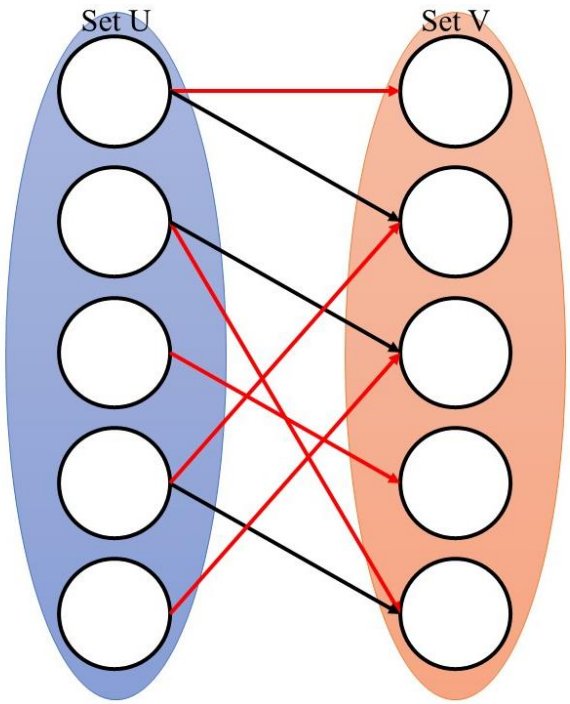

Figure 1.2: Bipartite Graph Matching Solution

can be formulated. Suppose *U* and *V* are the two sets of the bipartite graph and let *E* be the set of edges connecting these two sets. (i, j) corresponds to the arc that exists between nodes *i* and *j*. The decision variables are binary variables that indicate whether a given arc is in the maximum matching, denoted by Equation 1. The objective function for maximum matching is to maximize the number of arcs used in the matching, denoted by Equation 2. The constraints for this model are relatively simple. The sum of all arcs from *i* to *j* must be less than or equal to one for each *i*, shown in Equation 3. The sum of all arcs from *i* to *j* must be less than or equal to one for each *j*, as shown in Equation 4. In simpler terms, each node in set *U* can only be matched to one node in set *V*. The following is the formulation for the integer program.

Decision Variables

$$x_{ij} \in \{0,1\} \quad (1)$$

Objective Function

$$\max \sum_{(i,j)\in E} x_{ij} \quad (2)$$

Constraints

$$\sum_{(i,j)\in E} x_{ij} \leq 1, \forall\ i \in U \quad (3)$$

$$\sum_{(i,j)\in E} x_{ij} \leq 1, \forall\ j \in V \quad (4)$$

There are multiple types of maximum matchings that are possible for bipartite graphs. These include maximum-weight matching, which maximizes the weight of the arcs that are used, minimum-weight matching, which minimizes the weights of the arcs that are used, and maximum-cardinality matching, which is used for unweighted graphs. Bipartite

graph matchings can be applied to protein structure comparison [7] and pattern recognition and image processing [8]. In addition, bipartite graphs can be used for simple assignment problems, where each object in a subset needs to be assigned to an object in another subset of objects.

In addition to using bipartite graphs to solve assignment problems, a network flow approach can also be used for assignment problems involving time periods. Multicommodity flow networks are commonly used for transportation problems, and the corresponding linear program is a complex problem due to many variables and constraints in the model [9]. Due to the complexity of the model, branching and decomposition methods have been developed to determine an optimal solution in a less time-consuming way [10]. Multicommodity flow is an approach that solves problems with multiple goods leaving a source and flowing through a network to a target [11]. There are several associated optimization problems for multicommodity flow, which include minimum-cost, maximum, and maximum-concurrent multicommodity flow [10]. The minimum-cost multicommodity flow model will be discussed in depth regarding the proposed optimal classroom assignment model. Consider a directed graph with a set of arcs and nodes, including a source node and a target node. For every arc in the model, there is an associated capacity and cost. The goal of the minimum-cost multicommodity flow problem is to find an optimal path for each commodity to take that minimizes the total cost while meeting the demand of the commodity. Constraints for multicommodity flow problems include not exceeding the capacity of the arcs, flow conservation throughout the network, and flow conservation out of the source node and into the target node. The solving of this model

leads to an optimal solution that minimizes the total cost of transporting commodities across the network while ensuring all demand is met.

The following section will describe the details of the problem to be solved and the methodologies used to obtain an optimal solution, as well as provide examples of multicommodity flow graphs and a minimum-cost solution.

CHAPTER TWO
METHODOLOGIES

The first step in solving the classroom assignment problem involved data exploration and analysis. A list of all courses offered at the university was supplied by The University of Tennessee's registrar's office. Because this model only assigns courses to campus-hosted classrooms, which are classrooms maintained by the University and not the local academic departments, filtering the data had to be performed to obtain a list of courses that were only assigned to these campus-hosted rooms. After the data was filtered into courses that needed to be assigned classrooms, further analysis needed to be done regarding combining cross-listed courses. These courses are offered under different subject codes, but meet all meet at the same time. To combine these courses, the meeting time, professor, and current classroom assignment of a specific course were compared to the other courses to determine if it was cross-listed. If this condition holds, two different cases were evaluated, and one was applied to the combining of capacities for the given course. Case 1 involved combining the capacity of each course and giving the combined course a total capacity of the sum of these courses. Case 2 involved assigning the combined course a total capacity of one of the courses. After this step of data processing, the set of all courses that meet in campus-hosted spaces was ready for optimization.

The first phase of the classroom assignment optimization model includes a minimum-weight bipartite matching to assign courses to campus-hosted spaces in which the course's department has priority. Each department across campus has a set of campus-hosted classrooms where they hold priority, and courses from that department are assigned to a priority classroom if the enrollment capacity of the course fills at least 80% of the

classroom's seating capacity. It is important to note that this rule only applies to courses meeting during the standard time blocks. If a course does not meet during one of the standard time blocks, it will not be assigned during the first phase of the assignment model.

A bipartite graph was built for each time block and day to determine the optimal assignment of courses to priority rooms. The set of nodes on the left side of the bipartite graph includes all courses that meet during the specified standard time block on a given day and the nodes on the right side of the graph are all rooms with a priority department assignment. Arcs are drawn between a course and the rooms that it has priority in a given time block. For example, say Course 1 holds priority in Room 1, Course 2 holds priority in Rooms 2 and 3, and Course 3 holds priority in Rooms 2 and 3. Figure 2.1 shows the bipartite graph corresponding to this set of courses and rooms.

To solve the assignment problem, a maximum matching must be found between the course and the campus-hosted classrooms. To find the minimum-weight bipartite matching, weights must be added to the arcs first. For this model, the weight for each arc is equal to the capacity of the room minus the capacity of the course. For example, Course 1 has a capacity of 30, Course 2 has a capacity of 40, and Course 3 has a capacity of 44. Room 1 has a capacity of 35, Room 2 has a capacity of 45, and Room 3 has a capacity of 48. Figure 2.2 shows the new bipartite graph with the added weights for each arc.

The goal of the bipartite matching problem is to minimize the sum of the weights of the arcs used while maximizing the number of matchings present in the graph. This can be translated into a linear programming model by adapting the bipartite graph to a network graph. A source node is added with arcs to every course to be assigned in the given time

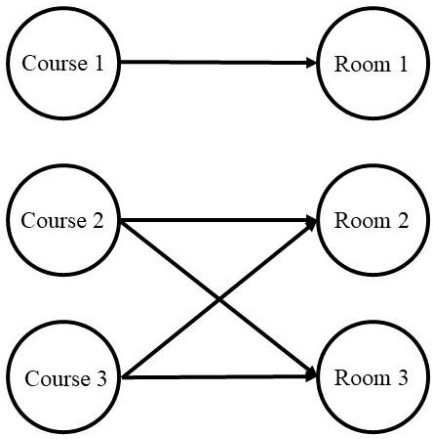

Figure 2.1: Bipartite Graph for Classroom Assignment

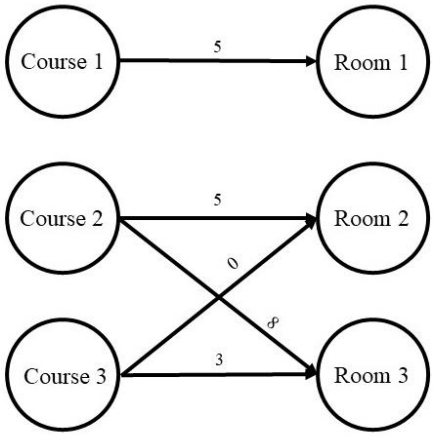

Figure 2.2: Bipartite Graph with Weights for Classroom Assignment

block, and a target node is added with arcs from every campus-hosted classroom with priority to the target node. This allows the bipartite matching to be written with an objective function and set of constraints. For bipartite graph matching, there are two sets; *N* corresponds to the set of all nodes in the network and *A* corresponds to the set of all arcs in the network. *C* corresponds to the set of all courses to be assigned and *R* corresponds to the set of all campus-hosted classrooms. In addition, several parameters are used in the model formulation; *s* corresponds to the source node and *t* corresponds to the target node. *M* is equal to the number of courses to be assigned, K_c is the capacity of course *c*, K_r is the capacity of campus-hosted classroom *r*. The weight of each arc is written as w_{ij}, which is equivalent to the difference between the capacity of room *j* and the capacity of course *i*. It is important to note that the weight of each arc must be greater than or equal to zero. This outcome is due to capacity constraints. A course cannot be assigned to a campus-hosted classroom that has a capacity smaller than the capacity of the course due to the requirement that every student must have a seat in the classroom. Once these parameters are defined, the objective function and constraints can be written. The following is the formulation for the associated linear program for the classroom assignment problem.

Decision Variables

$$x_{i,j} \in \{0,1\} \quad (1)$$

Objective Function

$$\min \sum_{(i,j)\in A} w_{i,j} x_{i,j} \quad (2)$$

Constraints

$$\sum_{\{i|(s,i)\in A\}} x_{s,i} = M \quad (3)$$

$$\sum_{\{i|(i,t)\in A\}} x_{i,t} = M \quad (4)$$

$$\sum_{\{(j,i)\in A\}} x_{j,i} - \sum_{\{(i,j)\in A\}} x_{i,j} = 0 \quad (5)$$

$$\sum_{i} x_{i,j} \leq 1 \; \forall \; j \; \in R \quad (6)$$

$$\sum_{j} x_{i,j} = 1 \; \forall \; i \; \in C \quad (7)$$

The decision variables for this model, denoted by Equation 1, are binary variables that indicate whether a given arc is used in the optimal solution. Equation 2 represents the objective function for this linear program. The goal of the objective function is to minimize the weights of the arcs used, which corresponds to minimizing the total excess capacity in occupied classrooms. The first constraint is displayed in Equation 3. This constraint ensures that the total number of used arcs leaving the source node must equal the number of courses to be assigned to a campus-hosted classroom for a given time block. The second constraint is displayed in Equation 4. Similar to the first constraint, this ensures that the total of used arcs entering the target node must equal the number of courses to be assigned to a campus-hosted classroom for a given time block. The third constraint, shown in Equation 5, ensures that flow is conserved throughout the network. The fourth constraint, displayed in Equation 6, ensures that each classroom is used at most once. The fifth constraint, displayed in Equation 7, ensures that every class receives an assignment. The last two constraints translate into only one course assigned to each campus-hosted classroom.

Figure 2.3 returns to the above example to give an example of the solution using the objective function and constraints listed above. For this example, there are two possible solutions. Course 2 could be assigned to Room 2 or 3, as well as Course 3. The sums of the

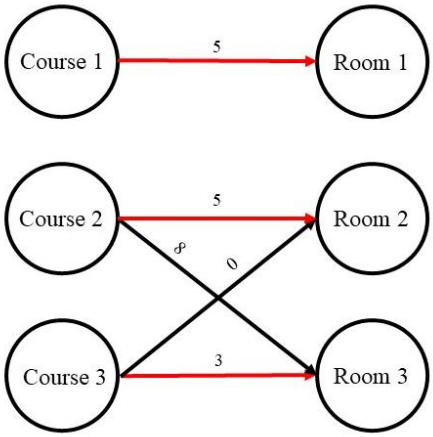

Figure 2.3: Bipartite Graph for Classroom Assignment Solution

weights are equal, and thus both solutions are optimal.

Sometimes, there are insufficient rooms to fulfill priority assignments for all courses in a given time block. In this case, dummy classroom nodes are added to the right side of the bipartite graph for each course to be assigned during a time block. These dummy nodes have a large weight, greater than the largest difference between a course and a possible room assignment in the graph. This weight ensures that all courses that could be assigned during a time block receive a priority room assignment, and only those courses that have no rooms to be assigned will be assigned to a dummy room.

When considering courses that do not meet during a standard time block, bipartite matching is not a feasible method for obtaining an optimal assignment of courses to campus-hosted classrooms. For example, a course might meet during the second standard time block on Monday and Wednesday. Another course might span the first and second time blocks on Wednesday. Since the bipartite matching problem is solved in order of days and then time blocks, a conflicting assignment could result when scheduling these two courses. Let no course be assigned to Room 1 in the first time block on Monday, and the first course be assigned to Room 1 in the second time block on Monday. This procedure leaves Room 1 available for assignment during the first time block on Wednesday. Let the second course be assigned to Room 1 during the first time block on Wednesday. The constraints of bipartite matching would require both the first and second courses to occupy Room 1 during the second time block on Wednesday. Thus, bipartite matching is not a feasible method for solving the classroom assignment problem if courses meet during non-standard time blocks. Thus, this model only allows for bipartite matching to be used to

assign courses that meet during standard time blocks on a given day. To consider courses that do not meet during standard time blocks, a multicommodity flow approach is used. Multicommodity flow allows an entire day to be solved at once, thus avoiding the problem of conflicting assignments during different time blocks. The following section explains the multicommodity flow approach and how the results from the priority assignment from bipartite matching are used in this approach.

The second phase of this model consists of a multicommodity network flow approach to assign the remaining courses to campus-hosted classrooms. The first step in creating the multicommodity flow model was the development of a directed graph. This graph consists of a depot or "home base" for the professors, a set of campus-hosted classrooms, and an office for each professor, all connected by a set of arcs. The set of arcs and nodes used to construct the graph was determined through data analysis of campus-hosted classrooms, professor office spaces, distances between classrooms and offices, and the capacity of courses to be assigned.

The graph for the classroom assignment model consists of a set of nodes that are needed for arc creation. The first node of the graph is a source node. This node is contained in a zeroth time block for each day and serves the purpose of hosting a central location for all professors to begin at before they flow throughout the network. In addition, all professors are assumed to be in their offices in the first time block and the $t + 1$ time block, where t is the last standard time block courses are offered on a given day. At the $t + 2$ time block is a target node. The purpose of this node is to have a central location where all professors return at the end of the day. These two nodes are important in creating

constraints for the optimization model. The last set of nodes needed to create this graph consists of all professor offices and campus-hosted classrooms for each time block. Thus the total number of nodes for this graph can be calculated using Equation 1.

$$(Number\ of\ Prof.Offices + Number\ of\ Classrooms) * Number\ of\ Time\ Blocks + 2 * Number\ of\ Prof.Offices + 2 \quad (1)$$

These nodes, plus the source and the sink nodes, make up the graph that professors will flow through on a given day. It should be noted that the nodes are created for each standard time block on a given day. These time blocks are predefined by the university and serve as a condition for assignment into campus-hosted classroom spaces.

For flow to occur, arcs must be generated between these nodes. For the arcs leaving the source node, each professor has at least one arc that connects to their office node or a campus-hosted classroom for the first time block on a given day. An arc is only generated between time blocks if the course in the $t + 1$ time block has a capacity that is at least 50% of that room's capacity and if the course is within a 0.7-mile, or fifteen-minute, walking distance of the professor's previous location. These arcs are generated between all nodes in each time block. It should be noted that a set of classrooms with a preset capacity limit was created to allow for courses whose capacities do not meet the 50% fulfillment requirement to be assigned to a room within this set. For example, if the preset capacity limit is equal to 15 and the maximum room size is set to 25, any course with a capacity of 15 or less can be assigned to any classroom with a capacity of 25 or less. Between the t time block and $t + 1$ time block, where t is the final time block where courses are offered, arcs are generated from every node back to their office. From the $t + 1$ time block to the t

+ 2 time block, an arc will be generated from every office to the target node. It should be noted that if a course is not within the standard time block, arcs are only generated for all possible room assignments for the first time block it occurs during and in the following time blocks it occurs during the arcs are only generated from the previous location to the same location in the current time block. This approach prevents courses that meet outside of the standard time blocks from being assigned to different classrooms for back-to-back standard time blocks. To demonstrate the drawing of this graph, Professor Jane Doe is used as an example and the course capacity constraint is set to 70%. On Mondays, Professor Doe teaches a course during the second standard time block and the third standard time block. The capacity of her first course is greater than 70% of the capacity of Rooms 2 and 3. The capacity of her second course is greater than 70% of the capacity of Rooms 1, 2, 3, and 4. Note that for this example, Monday consists of four standard time blocks. Figure 2.1 shows the graph that would be drawn for Professor Doe.

To determine what path will be taken by the professor, the arcs must have a weight associated with them. This weight is determined by the walking time between the nodes the arc connects and the room's capacity if a course is taught during the current time block. The walking distance is simply the time it takes to walk between the two locations. The value for the capacity of a room is converted to a percentage of excess capacity. This value was calculated using the following equation,

$$P_E = \frac{C_K}{C_R - C_K},$$

where C_K is the capacity of the course, C_R is the capacity of the room, and P_E is the percentage of excess capacity of a room, expressed as a value between 0 and 1. Note this

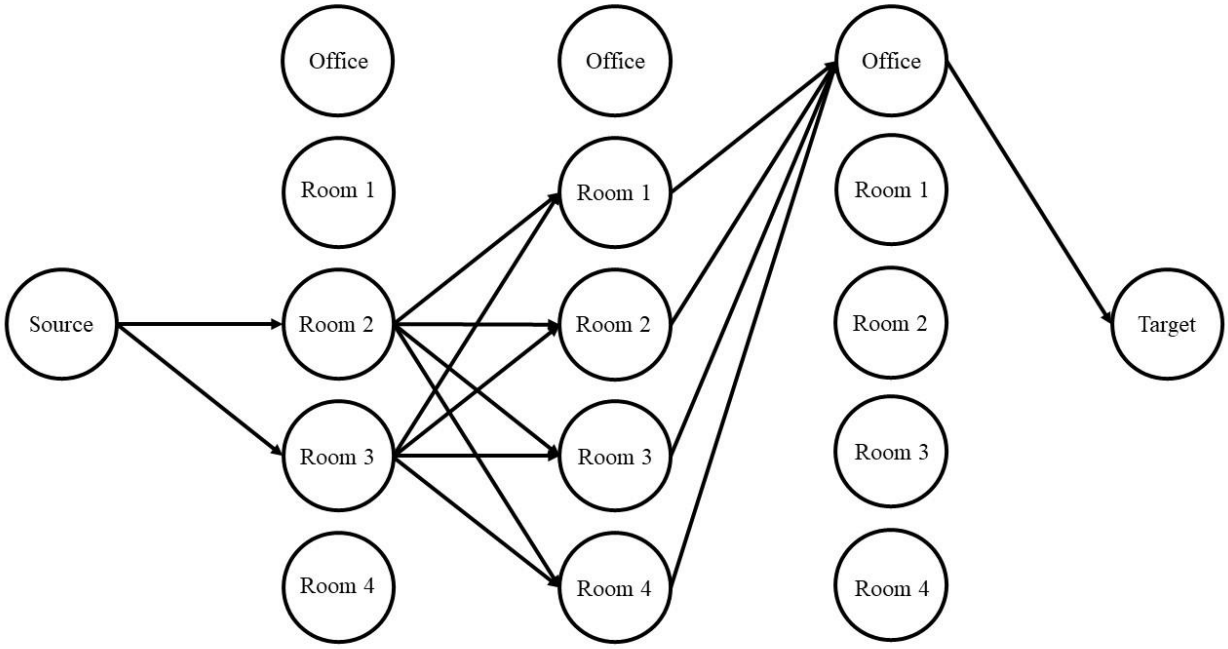

Figure 2.4: Classroom Assignment Directed Graph

value will always be less than 0.5 since arcs are only created if a course occupies at least fifty percent of a classroom's space. The excess capacity and walking time for each arc are multiplied by weights α and β which are determined by the importance of each element to the assignment process. This is based on the registrar and individual professors' requests. Returning to the example previously mentioned, the graph can now have weights added to the arcs. Rooms 1 and 4 have a capacity of 20 and Rooms 2 and 3 have a capacity of 18. Professor Doe's first course has a capacity of 13 and their second course has a capacity of 15. The walking times between the courses and her room are given in Table 2.1. With this information, the weights can be calculated for each arc. For example, the weight for the arc between Room 2 in time block two and Room 4 in time block three is calculated by the excess capacity in Room 4, which is 0.25, and the walking time between Rooms 2 and 4, which is 3 minutes. This calculation gives a total weight of 3.25 if each category has α and β values of one. The graph for Professor Doe with the corresponding weights is presented in Figure 2.2. The arc creation is performed for each professor teaching on a given day, as the model optimizes based on a full day of courses. The network graph exponentially grows as each professor is added to the graph. With each additional professor added to the graph, the optimization model takes longer to solve and results in a larger data preprocessing time.

To solve the problem, an optimization model must be built to determine which arcs will be used in the final solution. This problem can be written as an integer program. The parameters for this model are as follows. *s* represents the source node, and *r* represents the target node for the directed graph. The teaching day is represented as *m*, an element of the set of weekdays. The standard time block is represented as t_z, which indicates time block

Table 2.1: Sample Distance Matrix

	Room 1	Room 2	Room 3	Room 4	Doe's Office
Room 1	0	2	3	2	4
Room 2	2	0	7	3	2
Room 3	3	7	0	1	3
Room 4	2	3	1	0	6
Doe's Office	4	2	3	6	0

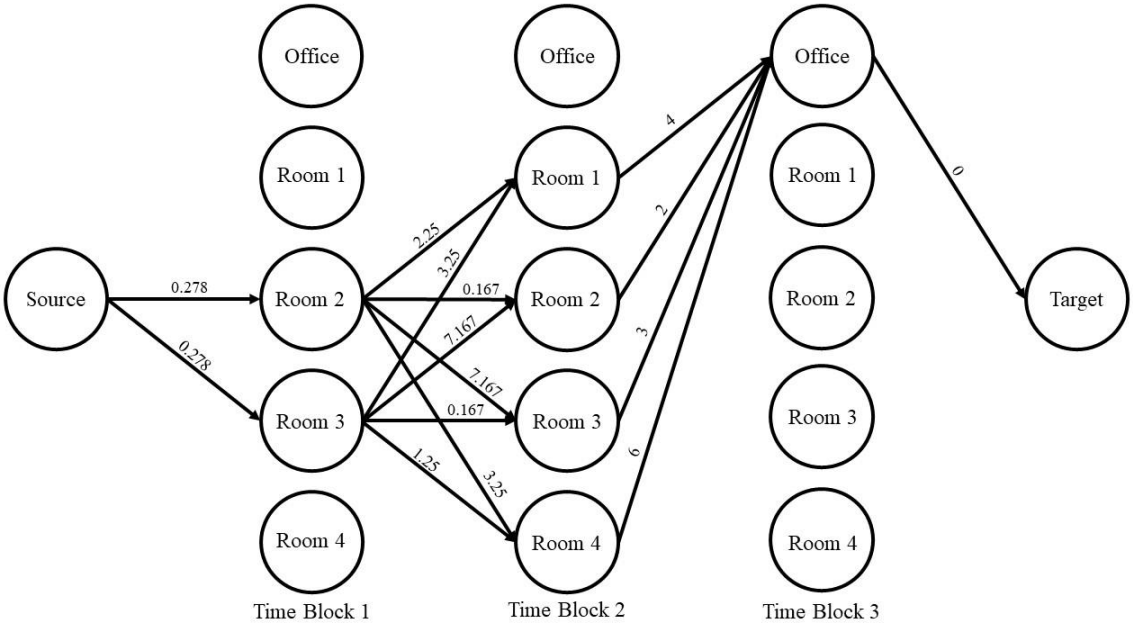

Figure 2.5: Classroom Assignment Directed Graph with Weights

t with index *z*. The set of all time blocks on day *m* is denoted as the set T_m. An individual course is represented by *k*, and the set of all courses is represented by *K*. The penalty weights for walking distance and capacity are represented by α and β. The reasonable percentage for whether a course is assigned to a room is represented by μ. The capacity of classroom *j* is represented by C_j, and the capacity of course *k* is represented by C_k. The percentage of excess capacity in classroom *j* if course *k* is assigned is represented by C_{jk}. This value is calculated using the following equation, $C_{jk} = \frac{C_j - C_k}{C_j}$.

U_{p,m,t_z} is a binary variable, where 1 indicates professor *p* teaches a course in time block t_z on day *m* and 0 otherwise. D_{ij} represents the distance from location *i* to location *j*. P_{dj} is a binary variable for priority, where 1 indicates department *d* does not have priority in classroom *j* and 0 indicates department *d* has priority in classroom *j*. The weight of each arc is represented by $w_{(\mathrm{p,m,(i,t_z),(j,t_{z+1})})}$. This value is calculated using the following equation,

$$w_{(\mathrm{p,m,(i,t_z),(j,t_{z+1})})} = \alpha D_{ij} + U_{p,m,t_{z+1}}(\beta C_{jk})\ \forall\ i \in PL_{p,m,t_z};\ j \in PL_{p,m,t_{z+1}};\ p \in \mathrm{M}_m;\ \mathrm{t}_z, \mathrm{t}_{z+1} \in \mathrm{T}_m\ ;\ m \in \{\mathrm{M, T, W, R, F}\}.$$

For example, the weight of the arc (Jane Doe, 'M', (Room 1, 1), (Room 2, 2)) can be calculated by using the above equation. Below is the calculation of the weight.

$$w_{(Jane\ Doe,\ \ M,\ \ (Room\ 1,1),\ \ (Room\ 2,2))} =$$
$$\alpha D_{(Room\ 1,\ \ Room\ 2)} + U_{Jane\ Doe,M,2}\left[\beta C_{(Room\ 2,\ \ MATH-001)}\right]$$

Note that the following constraints must hold for an arc to exist, $C_{jk} \geq 0$ and $C_k \geq \mu * C_j$.

The set of all nodes for day *m* in the graph is represented by N_m. These are formatted as

(classroom, time block, day) or (office, time block, day). N_m includes all campus-hosted classrooms, professors' office locations, and source and target nodes. An example of a classroom node is (Room 1, 1, M), which corresponds to Room 1 during the first time block on Monday. An example of a professor office node is (OFF-Jane Doe, 2, M), which corresponds to Jane Doe's office during the second time block on Monday. The set of all arcs in the graph for day *m* is represented by A_m. These arcs are formatted as (p, m, (i, t_z), (j, t_{z+1})). An example of an arc for Jane Doe is (Jane Doe, M, (Room 1, 1), (Room 2, 2)), which indicates Jane Doe is moving from Room 1 in the first time block to Room 2 in the second time block on Monday. The set of all professors teaching at least one course on day *m* is denoted by M_m. The set of all days that course *k* is offered is denoted by F_k. The set of all possible locations for professor p in the z^{th} time block of day m is denoted by PL_{p,m,t_z}.

The decision variables, objective function, and constraints for the optimization model are as follows.

Variables

$$x_{p,m,(i,t_z),(j,t_{z+1})} \in \{0,1\} \quad (1)$$

Objective Function

$$\sum_{(\mathrm{p,m,(i,t_z),(j,t_{z+1}))}\in A_m} \left[w_{(\mathrm{p,m,(i,t_z),(j,t_{z+1}))}}\, x_{(\mathrm{p,m,(i,t_z),(j,t_{z+1}))}}\right] \forall\, i \in PL_{p,m,t_z}; \quad (2)$$
$$j \in PL_{p,m,t_{z+1}}; p \in \mathrm{M}_m;\ \mathrm{t}_z, \mathrm{t}_{z+1} \in \mathrm{T}_m\ ;\ m \in \{\mathrm{M,T,W,R,F}\}$$

Constraints

$$\sum_{\{i|(\mathrm{p,m,(s,0),(i,1)})\in A_m\}} [x_{(\mathrm{p,m,(s,0),(i,1))}}] = |M_m|\ \forall\, i \in PL_{p,m,t_1};\ p \in \mathrm{M}_m; \quad (3)$$
$$m \in \{\mathrm{M,T,W,R,F}\}$$

$$\sum_{\left\{i\middle|\left(\mathrm{p,m,(i,t_{|T_m|}),(r,t_{|T_m|}+1)}\right)\in A_m\right\}} [x_{(\mathrm{p,m,(i,|T_m|),(r,|T_m|+1)}}] = |M_m|\ \forall \quad (4)$$
$$i \in PL_{p,m,t_z}; p \in \mathrm{M}_m;\ m \in \{\mathrm{M,T,W,R,F}\}$$

$$\sum_{\{(p,m,(i,t_z),(j,t_{z+1}))\in A_m\}} \left[x_{(p,m,(i,t_{z+1}),(j,t_{z+2}))}\right] - \sum_{\{(p,m,(j,t_{z+1}),(l,t_{z+2}))\in A_m\}} \left[x_{(p,m,(j,t_{z+1}),(l,t_{z+2}))}\right] = 0 \quad (5)$$

$\forall\, i \in PL_{p,m,t_z};\ j \in PL_{p,m,t_{z+1}};\ l \in PL_{p,m,t_{z+2}};\ p \in M_m;\ t_z, t_{z+1}, t_{z+2} \in T_m - \{0, T_{|T_m|}\};$
$m \in \{M, T, W, R, F\}$

$$\sum_{\{(p,m,(i,t_z),(j,t_{z+1}))\in A_m\}} \left[x_{(p,m,(i,t_z),(j,t_{z+1}))}\right] = 1 \quad (6)$$

$\forall\, i \in PL_{p,m,t_z};\ j \in PL_{p,m,t_{z+1}};\ p \in M_m;\ t_z, t_{z+1} \in T_m;\ m \in \{M, T, W, R, F\}$

$$x_{(p,m,(i,t_z),(j,t_{z+1}))} = x_{\left(p,m',(i,t_z),(j,t_{z+1})\right)} \quad (7)$$

$\forall\, i \in PL_{p,m,t_z};\ j \in PL_{p,m,t_{z+1}}; p \in M_m;\ t_z, t_{z+1} \in T_m;\ m, m' \in F_k; k \in K$

The decision variables for this model are all binary and indicate whether an arc is used. This then translates to which classroom a course occupies during a given time block on a given day. If this decision variable equals 1, course *k* will be assigned to classroom *j*. If it is equal to 0, the arc is not used, and thus the course will not be assigned to the corresponding classroom.

The objective function for the integer program is a minimization problem. The objective is to minimize the sum of the weights of the arcs used in the flow problem. This applies to the set of arcs composed of all possible locations for a professor during each time block for a given day. In turn, the objective function minimizes the excess capacity in occupied classrooms and the walking distance to classrooms for professors.

The five constraints for this model help guide the professors throughout the network. The first constraint, defined by Equation 3, ensures that the number of arcs leaving the source node equals the number of professors teaching during a given day. This value allows only the total number of professors to flow through the network to avoid having professors assigned to multiple classrooms during the first time block. The second

constraint, defined by Equation 4, ensures that the number of arcs entering the target equals the number of professors teaching during a given day. This value ensures that a professor is not assigned to multiple classrooms during the last standard time block on a given day. The third constraint, defined by Equation 5, is simply a flow conservation constraint. The sum of all arcs entering a node must equal the sum of all arcs leaving a node. This ensures that a professor flows throughout the network and ends at the target node. Note that this constraint does not hold for the zeroth time block or the t+1 time block of a given day because the source and target constraints apply to those time blocks. The fourth constraint, defined by Equation 6, states that the number of arcs entering a node must equal 1. A classroom can only be assigned to one course, and thus only one professor can enter that room during a given time block. The last constraint, defined by Equation 7, ensures that if a course meets on multiple days, the arc used on the first assignment day is used again during subsequent meeting days. Due to the university requirements, the course must occupy the same classroom each time it meets, thus requiring this constraint to be defined.

Solving this model will yield an optimal set of arcs to be used, which is equivalent to the route each professor will take throughout a given day. This information is used to determine which campus-hosted classroom will be occupied by a given course. The result will be a set of classrooms that minimizes the wasted excess capacity of occupied classrooms across campus and cuts down the walking distance for professors between classrooms and offices.

Returning to the example of Jane Doe, this model has a solution that is presented in Figure 2.6. The arcs in red are the arcs that will be used in the model for Jane Doe to

flow through. From the solution, it is apparent that Jane Doe will teach both her first course and second course in Room 2. The sum of all the weights used to route Jane Doe from the source node to the target node is equal to 2.445, which is the lowest possible sum of weights for the arcs that satisfy the constraints of the optimization model.

The following section will provide details on the results obtained from the model after being tested on Spring 2022 course data from The University of Tennessee. It also includes a discussion on the impact of these results and what they mean for future classroom assignment.

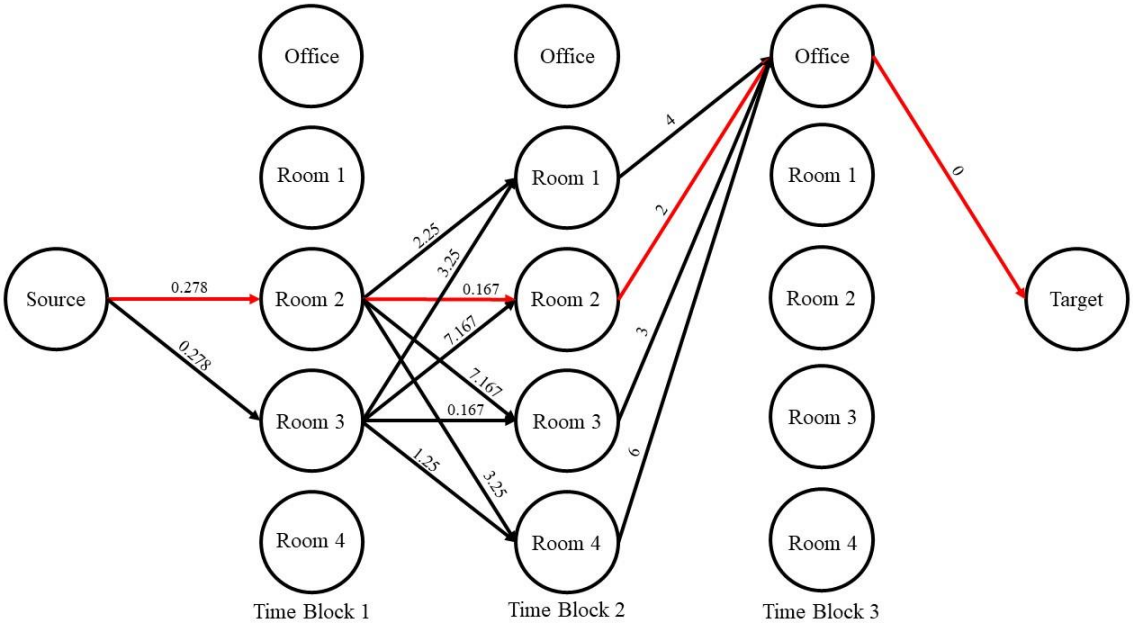

Figure 2.6: Minimum Cost Flow Solution for Classroom Assignment

CHAPTER THREE
RESULTS AND DISCUSSION

This optimization model was applied to Spring 2022 course data provided by the University of Tennessee's Registrar. The optimization model was solved using Gurobi Optimizer on a PC with Intel Core i7 8th generation processor and 16 GB of RAM. It should be noted that campus-hosted classroom information, priority department information, and professor office information were obtained through the University of Tennessee's Registrar. All walking distance and duration information was obtained through Google Maps API.

Some data preprocessing had to occur before solving the linear program for the University's official course data. First, when creating the arcs for the directed graph, if a professor did not have an office listed in the official university directory, they were assumed to have a "home base" where the department of the classes they taught the most is located. For example, if a professor taught two nuclear engineering courses during a given day and one industrial engineering course, they were assumed to have an office located in the home location of the nuclear engineering department.

For the optimal solution, the percent change in classroom assignments and the percent change in building assignments were calculated. The percent change in classroom assignments allows for the number of classroom changes to be evaluated and determine the effect of the departmental priority and capacity constraints. The percent change of building assignments, which is which courses changed in which building they were offered, was calculated to determine the effect of the walking distance constraint on classroom assignments. If a course was assigned to a campus-hosted classroom on a different floor

than the original assignment, it was assumed that it changed classrooms, but the building stayed the same. The following equations were used for the percent change of classrooms and buildings,

$\delta_{pcr} = \frac{R_{ch}}{K}$ and $\delta_{pcb} = \frac{B_{ch}}{K}$,

where δ_{pcr} is the percent change in classroom assignments, δ_{pcb} is the percent change in building assignments, R_{ch} is the total number of changed classroom assignments for a given semester, B_{ch} is the total number of changed building assignments for a semester, and K is the total number of courses offered for a given semester. A value of 88.02% was obtained for the percent change in classrooms and 64.73% was obtained for the percent change in buildings. These percentages indicate that there was not an optimal assignment of courses, given the requests from the registrar. To evaluate the capacity and distance changes, the percent average excess capacity in occupied rooms should be calculated, as well as the average walking distance for each professor.

The average percent excess capacity across all occupied classrooms for the new optimal assignment, as well as the average excess capacity across all classrooms for the original assignment from the Registrar's Office, were calculated to determine how effective the new model is at meeting the objectives. This value is calculated by first finding the percent excess capacity for each course, which is simply the difference between the assigned classroom's capacity and the course's capacity divided by the total capacity of the classroom. The values for each course are summed together and divided by the total number of courses to find the average percent excess capacity for the semester. This value is not calculated per day but as an average across all courses. A value of 22.97% was

obtained for the original assignment and a value of 12.17% was obtained for the optimal assignment. The average percent excess capacity decreased with the new model. This shows that this model will optimally assign courses to minimize the wasted space on campus, allowing for more effective utilization of campus-hosted space. By using classrooms that better fit the capacity of the course, resources are not wasted on classrooms that are not highly occupied. This outcome could in turn lead to monetary savings on energy costs. For example, a smaller classroom will likely have less lighting, which could lead to less money spent on energy costs. This decrease could in turn allow universities to relocate part of their budget to fund a variety of other projects. In addition to evaluating how excess capacity changed after optimization, it is also important to evaluate the change in walking time for professors. For the Spring 2022 semester, the walking distance for professors throughout their day was calculated, assuming a professor takes a route modeled by the network flow model. An average of this value was calculated by summing all walking distances and dividing by the total number of professors teaching on a given day. Table 3.1 summarizes this for Spring 2022 course data. The average walking distance for the Spring 2022 semester decreased from the original model to the optimal model for all meeting days. This outcome shows that the optimization model accurately assigns courses to classrooms that minimize professors' average walking time. Because the time between standard time blocks is between 20 and 25 minutes, time is a limited resource for professors who teach back-to-back courses. Decreasing the time it takes for professors to walk between classrooms gives them more time to prepare for their next course before beginning to teach. This time increase favors a better student experience, as the instructor has more time to

Table 3.1: Average Travel Distances per Day for Optimal and Original Assignments

Semester	Monday	Tuesday	Wednesday	Thursday	Friday
Spring 2022	0.36 miles optimally 0.79 miles originally	0.37 miles optimally 0.58 miles originally	0.36 miles optimally 0.79 miles originally	0.38 miles optimally 0.55 miles originally	0.43 miles optimally 0.67 miles originally

prepare for class and focus more completely on the class content and goals, instead of stressing about their commute to the course. This time increase also allows more time for the students to talk with the instructor before or after class, as the instructor is not having to use the entire time between time blocks to commute to their next course.

From the results presented in the above tables, it is apparent that a multicommodity flow approach is an effective way to assign courses to campus-hosted classrooms. For the tested course data, the optimal assignment led to a decrease in the average percent excess capacity for courses in campus-hosted classrooms and a decrease in the average walking distance for professors.

In addition to determining if this model optimally assigns courses to campus-hosted spaces, it is also important to determine if the university has enough space for increasing student enrollment. Thus, the model was tested assuming an overall student enrollment increase of 10%. Assuming all courses receive a ten percent increase in enrollment, the model was run using these new capacities. It was determined that the university would not be able to handle an enrollment increase of this size without increasing the number of campus-hosted classrooms, increasing the capacity of current classrooms on campus, or increasing the number of courses offered.

In addition to calculating metrics to determine if an optimal assignment was found, optimization time was calculated to determine how long this model takes to implement. The preprocessing time includes the amount of time spent generating the arcs and weights for each day that courses are offered and the amount of time spent splitting the arcs into arcs in and arcs out subsets for each node. Table 3.2 summarizes how many arcs were

generated for each semester of data, separated by weekday. As the days progress, the number of arcs tends to decrease. It is important to note that this is due to potential locations for a course only consisting of the optimal assignments on subsequent meeting days, where the standard groups are Monday, Wednesday, and Friday, and Tuesday and Thursday. As the number of arcs increases, the preprocessing time and solution time increase exponentially. For Spring 2022 semester data, 2.75 minutes were used to determine the assignments for priority courses in standard time blocks using a maximum matching. 25.53 minutes were needed to generate the set of all possible arcs for the multicommodity flow model and 132.19 minutes were required to generate constraints and solve the multicommodity flow model.

Because the size of the problem to be solved is relatively large, it is expected that the solution time will be high. For this model, 3062 courses needed to be assigned across five days, each consisting of at least nine time blocks, into a set of 236 campus-hosted classrooms. In addition, 324 instructors taught at least two courses that meet in back-to-back time blocks, exponentially increasing the number of arcs needed for each instructor in that set. It should be noted that this assignment problem is NP-hard. Although the solution time is relatively high, it is unlikely that this assignment will need to be completed more than once per semester. By implementing a model, the assignment process for courses goes from a span of several months of the process being performed by hand to approximately 2 hours to determine an optimal assignment. This significantly reduces time and eliminates human error in the assignment process.

Table 3.2: Number of Arcs Created per Day

Semester	Monday	Tuesday	Wednesday	Thursday	Friday
Spring 2022	2.65 million arcs	1.39 million arcs	40.0 thousand arcs	30.8 thousand arcs	575.9 thousand arcs

The following section revisits the introduction to the problem, gives a brief overview of the methods used to solve the classroom assignment problem, the results obtained once the model was implemented on a test dataset, and recommendations for further research on the classroom assignment problem.

CHAPTER FOUR
CONCLUSIONS AND RECOMMENDATIONS

The model developed in this project holds the potential to transform the way classroom assignment is completed for a university. By taking an innovative approach to how classroom assignment is performed, classrooms can be more effectively used to enhance the student experience. A multicommodity flow approach allowed for walking distance to be considered between classrooms and offices for professors, which has the potential to increase the faculty retention rate. This retention rate increases the potential for increased student-professor relationships. In addition, this model allows for more effective use of spaces, allowing more courses to be offered in person instead of via virtual meeting setting. Enhancing the student experience by offering most courses in person could lead to a higher student retention rate. Due to the COVID-19 pandemic, students are eager to return to a sense of normalcy, which includes having a schedule of in-person classes. By optimizing how campus-hosted spaces are used, students can return to schedules similar to those they had pre-pandemic. Advancements must be made in terms of classroom assignment to increase retention rates among students and professors and overall give students a positive learning experience.

Some further advancements could be made by using this model as a foundational framework for further research. Because of the complexity of the model, data preprocessing and solution times are large. Heuristics could be implemented to decrease the solve time but still find a near-optimal solution for assignment. These could include robust optimization and zone decomposition. In addition, several other factors could be included in the weights of the arcs. For example, if a given classroom does not contain the

technology requested by a professor, a penalty weight could be added to the arc. Another possibility is adding a penalty weight to an arc if the classroom type is not the desired type. For example, a professor teaching a seminar is likely to prefer a classroom that has a layout desirable for discussion instead of standard lecture hall seating. Several factors could be considered for the arc weights. This model allows for customization of the elements factored into the weights of the arcs, allowing universities to customize for their needs.

This model effectively assigned courses to campus-hosted spaces while addressing the common complaints from professors, students, and the registrar. It is recommended that further research be conducted in classroom assignment to optimize further a process common to all universities and colleges globally.

LIST OF REFERENCES

[1] A.E. Phillips, H. Waterer, M. Ehrgott, D.M. Ryan, “Integer programming methods for large-scale practical classroom assignment problems,” *Computers & Operations Research*, vol. 53, pp. 42-53, Jan. 2015. Accessed on: Aug. 15, 2022. [Online]. Available doi: 10.1016/j.cor.2014.07.012.

[2] H. Waterer, “A Zero-one Integer Programming Model for Room Assignment at the University of Auckland,” *Proceedings of the 1995 ORSNZ Conference.* 1995, pp. 63-69.

[3] A. Dammak, A. Elloumi, H. Kamoun, “Classroom assignment for exam timetabling,” *Advances in Engineering Software*, vol. 37, no. 10, pp. 659-666, Oct. 2006. Accessed on: October 2, 2022. [Online]. Available doi: 10.1016/j.advengsoft.2006.02.001.

[4] A. Elloumi, H. Kamoun, B. Jarboui, A. Dammak, “The classroom assignment problem: Complexity, size reduction and heuristics,” *Applied Soft Computing*, vol. 14, no. C, pp. 677-686, Jan. 2014. Accessed on: October 3, 2022. [Online]. Available doi: 10.1016/j.asoc.2013.09.003.

[5] M. W. Carter, “A Lagrangian relaxation approach to the classroom assignment problem,” *INFOR: Information Systems and Operations Research*, vol. 27, no. 2, pp. 230-246, 1989. Accessed on: September 25, 2022. [Online]. Available doi: 10.1080/03155986.1989.11732094.

[6] H. Martinez-Alfaro, G. Flores-Teran, “Solving the classroom assignment problem with simulated annealing,” *SMC ’98 Conference Proceedings, 1988 IEEE International Conference on Systems, Man, and Cybernetics*, 1998, pp. 3702-2708, vol. 4. Accessed on: September 25, 2022. [Online]. Available doi: 10.1109/ICSMC.1998.726655.

[7] W. Taylor, “Protein structure comparison using bipartite graph matching and its application to protein structure classification,” *Molecular & cellular proteomics : MCP*, vol. 1, no. 4, pp. 334–339. Apr. 2002. Accessed on October 1, 2022. [Online]. Available doi: 10.1074/mcp.t200001-mcp200.

[8] D. Conte, P. Foggia, C. Sansone, and M. Vento, "Graph matching applications in pattern recognition and image processing," *Proceedings 2003 International Conference on Image Processing*, pp. II-21, Sep. 2003. Accessed on: October 10, 2022. [Online]. Available doi: 10.1109/ICIP.2003.1246606.

[9] P. Mahey, M.C. de Souza, “Local optimality for multicommodity flow problems with separable piecewise convex costs,” *Operations Research Letters*, vol. 35, no. 2, pp. 221-226, Mar. 2006. Accessed on: October 10, 2022. [Online]. Available doi: 10.1016/j.orl.2006.02.005.

[10] K. Salimifard, S. Bigharaz, “The multicommodity network flow problem: state of the art classification, applications, and solution methods,” *Operations Research,* vol. 22, no. 1, pp. 1-47, Apr. 2022. Accessed on: October 10, 2022. [Online]. Available doi: 10.1007/s12351-020-00564-8.

[11] J.A. Tomlin, “Minimum-cost multicommodity network flows,” *Operations Research,* vol. 14, no. 1, pp. 45-51, Feb. 1966. Accessed on September 10, 2022. [Online]. Available doi: 10.1287/opre.14.1.45.

VITA

Hannah Smith was born in Tennessee and spent her entire childhood there. After high school graduation, she attended The University of Alabama. She earned her Bachelor of Science degree in Mathematics, with a concentration in Statistics and Optimization and a minor in General Business. Following college graduation, she enrolled at The University of Tennessee to obtain her Master of Science degree in Industrial and Systems Engineering. She served as both a Graduate Teaching Assistant and Graduate Research Assistant during her time at the University of Tennessee. After graduation, Hannah will pursue a career in data science.