Student Project Allocation Using Integer Programming

A. A. Anwar and A. S. Bahaj, *Member, IEEE*

***Abstract*—The allocation of projects to students is a generic problem in many universities within the U.K. and elsewhere, not only in engineering but also in various other disciplines. This paper defines the student project allocation problem explicitly by an objective function and a number of constraints. Two integer program models are presented, the first of which is a dynamic program. A general purpose solver is used to solve the models, and the input files are included in the Appendix. The models are computationally efficient and easily solved on a PC. Important issues in interpreting the model outputs are highlighted. As with any optimization problem it is possible for constraints to be too tight to permit any feasible solution. Application of the models is demonstrated by using data from the Department of Civil and Environmental Engineering, University of Southampton, for the academic year 2001–2002. The model has been used successfully to allocate Individual Project and Group Project to students and is likely to become the defacto method of allocation of projects in the future. This paper demonstrates how operations research techniques used widely in optimizing use of resources can be applied in education.**

***Index Terms*—Allocation model, group project, individual project, integer programing, University of Southampton.**

NOMENCLATURE

$D_{i,l}$	Binary integer that assumes a value of 1 if a student is of a discipline l.
$E_{l,j}$	Binary integer that assumes a value of 1 if a discipline l is suited to project j and 0 otherwise.
$F_{i,j}$	Binary integer that assumes a value of 1 if a student i is of a discipline that is a subset of the disciplines required for project j and 0 otherwise.
G_j	Number of students allocated any project j.
i	Index representing a student and can assume a value $1 \cdots N_S$.
j	Index representing a suggested project and can assume a value $1 \cdots N_P$.
l	Integer representing a discipline and can assume a value $1 \cdots N_D$.
L_j	Late penalty factor.
M_j	Maximum number of students permitted on any project j.
N_D	Total number of disciplines.
N_j	Minimum number of students permitted on any project j.
N_P	Total number of suggested projects.
N_S	Total number of students.
P_j	Binary variable which assumes a value of 1 if a project has been allocated and 0 otherwise.
$S_{i,j}$	Binary integer that assumes the value of 1 if the project j is selected by the student i, and 0 otherwise.
$X_{i,j}$	Binary decision variable for the ith student and jth project.
Z_a	Objective function variable for model 1a.
Z_b	Objective function for model 1b.
α	Large positive number.
Δ	Total number of projects tutored by a member of staff.

Manuscript received October 24, 2001; revised August 24, 2002.

The authors are with the Department of Civil and Environmental Engineering, University of Southampton, Highfield, Southampton SO17 1BJ, U.K. (e-mail: A.A.Anwar@soton.ac.uk).

Digital Object Identifier 10.1109/TE.2003.811038

I. BACKGROUND

ALLOCATION of projects to students as part of a degree course is common to most, if not all, universities. However, the exact requirements and procedures differ widely. Any project allocation model/algorithm must take into account the specific conditions under which it is being applied. For the purpose of this paper, the Department of Civil and Environmental Engineering, University of Southampton, is taken as a case study, and its particular conditions for project allocation are explained at some length. However, the models presented are sufficiently generic to be applied to many other situations. One of the models presented is also applied to the conditions described by Teo and Ho [1] for the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.

At the Department of Civil and Environmental Engineering, University of Southampton, the staff prepare project briefs, and a list of projects is presented to the students from which the students select four projects, ranked in order of preference. Each individual project is supervised by one member of the staff and is allocated to one student within the department. Students on the four-year course are expected to complete two group projects in their fourth year of study. These projects are carried out by a group of students from various departments within the Faculty[1] of Engineering and Applied Science. Group projects are proposed by one or more members of the staff, often from different departments within the faculty. A group project will have minimum and maximum number of students and may also specify the disciplines for which the project is suitable. In contrast to the individual project, a group project will be supervised by one or more members of the staff and is allocated to a group of students.

[1]Faculty in U.K. universities refers to an organizational unit consisting of a group of departments. This is in contrast to the USA, where faculty refers to academic staff.

The problem of project allocation to individual students is a generic problem, common to many universities albeit with somewhat different criteria from those outlined earlier. In [1], Teo and Ho described their approach to final-year projects in an electrical engineering undergraduate course at Nanyan Technological University, Singapore. To allocate projects, Teo and Ho coded a computer program AssignProj in C. Although the AssignProj program described in [1] does assign a project to all student groups, it does not attempt to optimize the allocation, i.e., it produces a possible feasible solution but not necessarily an optimum solution.

There are numerous techniques used for optimization problems, e.g., integer programming, simulated annealing, tabu search, genetic algorithms, etc. [2]. Comparison of the various techniques is outside the scope of this paper. However, allocation problems of the type described in this paper are polynomial, i.e., can be solved within polynomial time. Integer programming is the only technique that will necessarily solve to a global optimum if a feasible solution exists and, therefore, is the technique used in this paper. This paper shows how two integer programs (with the first using dynamic programming) can be used to solve this problem of project allocation. The advantage to using an integer programming approach is that the model will seek the global optimum.

II. Individual Project Model

The objective of this model is to make staff effort in the supervision of individual student projects as uniform as possible and, if possible, assign students their first choice of a project. For the purpose of the development of the model, this objective has been translated into making the number of student projects supervised by the staff within a department equal (if possible). The model in effect has dual goals, and a dynamic programming approach is adopted. The first model *Model 1a* minimizes the number of projects supervised by each staff member. The second model *Model 1b* then distinguishes between equivalent solutions from *Model 1a* to select the solution where the maximum number of students are allocated their first choice.

A. Model 1a—Minimizing Number of Projects Supervised by Each Member of the Staff

1) Decision Variable: A student can be allocated any one project suggested by any member of the staff. This allocation can be defined by

$$\begin{aligned} X_{i,j} &= 1 \qquad \text{if student } i \text{ is assigned the project } j \\ &= 0 \qquad \text{otherwise} \end{aligned} \tag{1}$$

where $X_{i,j}$ = binary decision variable for the ith student and jth project; i = index representing a student and can assume a value $1 \cdots N_S$; N_S = total number of students; j = index representing a suggested project and can assume a value $1 \cdots N_P$; and N_P = total number of suggested projects.

2) Objective Function: The objective function is to minimize the number of projects tutored by the staff, given by

$$\text{MINIMIZE} \qquad Z_a = \Delta \tag{2}$$

where Z_a is the objective function variable for Model 1a; Δ = total number of projects tutored by a member of the staff.

3) Constraints: Every student must be allocated one and only one project. This constraint is enforced by

$$\sum_{j=1}^{N_p} X_{i,j} = 1 \qquad \text{for every student } i. \tag{3}$$

Similarly a project can be allocated to at most one student

$$\sum_{i=1}^{N_S} X_{i,j} \leq 1 \qquad \text{for every project } j. \tag{4}$$

For every student, the project allocated to a student can only be from the subset of projects selected by the student

$$\sum_{j=1}^{N_P} (X_{i,j} \times S_{i,j}) = 1 \qquad \text{for every student } i \tag{5}$$

where $S_{i,j}$ is the binary integer that assumes the value of 1 if the project j is one of the projects selected by student i, and 0 otherwise. Introducing a variable to define whether a project has been allocated as

$$P_j = \sum_{i=1}^{N_S} X_{i,j} \qquad \text{for every project } j \tag{6}$$

where P_j is the binary variable which assumes a value of 1 if a project has been allocated to a student, and 0 otherwise. Of the subset of projects suggested by a staff member, the staff member will only tutor the project if it has been allocated to a student. Defining

$$T_{k,j} = G_{k,j} \times P_j \qquad \begin{aligned} &\text{for every project } j, \text{ and} \\ &\text{for every staff member } k \end{aligned} \tag{7}$$

where $T_{k,j}$ is the binary variable which assumes a value of 1 if a staff k will tutor a project j, and 0 otherwise; k is the integer representing a member of the staff and can assume a value $1 \cdots N_T$; N_T is the total number of staff; and, $G_{k,j}$ is the binary variable which assumes a value of 1 if a member of the staff k has suggested a project j, and 0 otherwise. The total number of projects tutored by any member of the staff is then given by

$$\sum_{j=1}^{N_P} T_{k,j} \leq \Delta \qquad \text{for every staff member } k. \tag{8}$$

B. Model 1b—Allocating Project According to Ranking as Assigned by Students

1) Decision Variable: The decision variable remains unchanged as defined in (1).

2) Objective Function: Rather than simply select a subset of all the projects suggested, students could rank their selection of projects to indicate preference among the selection. If students are asked to select a subset of four projects from the complete set of projects, the students could assign an integer of 4 to the project of their first choice (highest priority), 3 to the project of second choice, 2 to the project of third choice, etc. Hence, the variable $S_{i,j}$ defined for (5) becomes an integer variable (rather than a binary integer variable) and takes a value 1, 2, 3, or 4. The objective is to maximize the sum of the rank of allocated projects. To favor those students who submitted their choices by the stipulated deadline, a penalty factor is introduced in the objective function. The objective function is defined as

$$\text{MAXIMIZE} \qquad Z_b = \sum_{j=1}^{N_P} \sum_{i=1}^{N_S} (X_{i,j} \times S_{i,j} \times L_j) \tag{9}$$

where Z_b is the objective function for model 1b; and, L_j is the late penalty factor.

3) Constraints: Constraint (5) needs to be redefined given the variable $S_{i,j}$ is no longer a binary integer as

$$\sum_{j=1}^{N_P} (X_{i,j} \times S_{i,j}) \geq 1 \qquad \text{for every student } i. \tag{10}$$

Model 1b allocates students their first choice of project (as far as possible). However, this allocation must be achieved without disturbing the optimum uniformity of staff effort from Model 1a. Therefore, (8) becomes

$$\sum_{j=1}^{N_P} T_{k,j} \leq Z_a \qquad \text{for every staff member } k. \tag{11}$$

All other constraints remain the same as defined for Model 1a.

III. Group Project Model

The requirements of group projects are different from those of the individual project described earlier. The aforementioned model is applicable where a student or group of students form a group before making a ranked selection of projects. An alternative strategy is to have individual students make a ranked selection of projects, and then form groups from the individuals allocated the same project. A suitable group project is suggested by one or more members of the staff. If a project is allocated to students, the project will be tutored by one or more members of the staff. The maximum and minimum permissible numbers of students on each group project is also specified. Suitable disciplines for the project may also be specified, e.g., suitable for students of Civil and Mechanical Engineering only, etc. Students are not expected to form groups themselves. Rather, a student is required to select a subset of projects (normally three) from the complete set and is required to rank his/her choice. By optimally allocating students to a project, project groups are formed implicitly. There is no attempt to distribute staff effort uniformly with group projects since the staff normally suggest no more than two or three projects. Hence, a starting point for this model is Model 1b described earlier.

A. Model 2—Allocating Project and Creating Student Groups by Virtue of Allocated Project

1) Decision Variable: The decision variable remains unchanged as defined in (1).

2) Objective Function: The objective function remains unchanged as defined in (9).

3) Constraints: Similar to the individual project model, within the group project model every student must be allocated a project. This constraint as defined in (3) remains unchanged. For a group project, the project can be allocated to more than one student; hence, (4) is no longer valid. A group project must be allocated to a number of students such that this number is not greater than the maximum permissible number nor less than the minimum number permissible. If

$$G_j = \sum_{i=1}^{N_S} X_{i,j} \qquad \text{for every project } j \tag{12}$$

where G_j = number of students allocated any project j, then the inequalities

$$G_j \leq M_j \qquad \text{for every project } j \tag{13}$$

where M_j is the maximum number of students permitted on any project j, and

$$G_j \geq N_j \times P_j \qquad \text{for every project } j \tag{14}$$

where N_j is the minimum number of students permitted on any project j, constraining the number of students on each allocated project. In (14), the number of students allocated to a project need only be greater than the minimum group size if the project has, in fact, been allocated; hence, the product on the right-hand side (RHS) of the constraint. Constraint (6) is no longer valid since a project may be allocated to more than one student. The constraint can be replaced by

$$P_j \times \alpha \geq G_j \qquad \text{for every project } j \tag{15}$$

where α is the large positive number. The number should be selected judiciously to avoid making the model computationally inefficient [4]. Provided it is sufficiently large, (15) ensures that if a project is allocated to one or more students, the variable P_j assumes a value of 1; otherwise, since P_j is binary, it assumes a value of 0.

The constraint defined by (10) that a student must only be allocated a project from the subset of projects selected by the student is also valid for the group project. Similarly, (7) is also valid for the group project. Finally, (11) has no relevance in the group project model since no attempt is being made to make staff effort uniform.

Variable	Value	Reduced Cost
ALLOCATION(ALWAN_Z, 20)	1.000000	-3.000000
ALLOCATION(ATTWOOD_A, 18)	1.000000	-1.200000
ALLOCATION(BARKER_J, 60)	1.000000	-4.000000
ALLOCATION(BLAKE_S, 16)	1.000000	-0.8000000
ALLOCATION(BROOMFIELD_K, 11)	1.000000	-4.000000
ALLOCATION(CHU_R, 10)	1.000000	-4.000000
ALLOCATION(CRESSWELL_A, 51)	1.000000	-4.000000
ALLOCATION(CZASTKA_J, 23)	1.000000	-0.4000000
ALLOCATION(DAVIES_P, 13)	1.000000	-4.000000
ALLOCATION(DONOVAN_K, 55)	1.000000	-4.000000
ALLOCATION.....		
.....		
.....		

Fig. 1. Partial LINGO solution report for Model 1b.

An additional constraint in the group project model needs to be introduced to ensure that the students allocated a project are of suitable disciplines for the project, if

$$F_{i,j} = \sum_{l=1}^{N_D} (D_{i,l} \times E_{l,j}) \qquad \begin{array}{l} \text{for every student } i \\ \text{for every project } j \end{array} \quad (16)$$

where $F_{i,j}$ = binary integer that assumes a value of 1 if a student i is of a discipline that is a subset of the disciplines required for project j, and 0 otherwise; $D_{i,l}$ = binary integer that assumes a value of 1 if a student is of a discipline l; l = integer representing a discipline and can assume a value $1 \cdots N_D$; N_D = total number of disciplines; and $E_{l,j}$ = binary integer that assumes a value of 1 if a discipline l is suited to project j, and 0 otherwise. Then the inequality

$$F_{i,j} \geq X_{i,j} \qquad \begin{array}{l} \text{for every student } i \\ \text{for every project } j \end{array} \quad (17)$$

achieves the required constraint.

IV. Application

A. Individual Project Model

The individual project model was applied to data from the Department of Civil and Environmental Engineering, University of Southampton, Southampton, U.K., for the academic year 2001–2002. Twenty-two members of the staff suggested a total of 60 projects (note project #43 was withdrawn). On the average each member of the staff suggested three projects, although the maximum number of projects by a member of the staff was seven, and the minimum was one. The 39 students in their second year of their course of study were asked to select any four projects and rank their selection assigning an integer 4 to their first choice, 3 to their second choice, etc. Students were required to submit their selection by the May 22, 2001 and were informed that if they missed this deadline, they would stand less chance of getting the project of choice. A penalty of 20% was applied to the ranking for each day the students missed the deadline—up to a maximum of 80%.

The model was implemented in LINGO 6.0 for Windows [3], running on an Intel Celeron processor based PC (800 MHz with 128 mB RAM). Rather than place the data directly in the input file, the data were placed in a EXCEL spreadsheet file, and OLE features were used. Both models solve in under three seconds. The objective function for Model 1a is 3. Using this value for Model 1b produces the allocation of projects to students partly shown in Fig. 1 with all zero values suppressed. The columns titled "Value" and "Reduced Cost" have no significance for integer programs and can be ignored, [4]. The column titled "Variable" shows that the student ALWAN_Z has been allocated project #20. Similarly ATTWOOD_A, has been allocated project #18. The LINGO input files for both models are included in the Appendix.

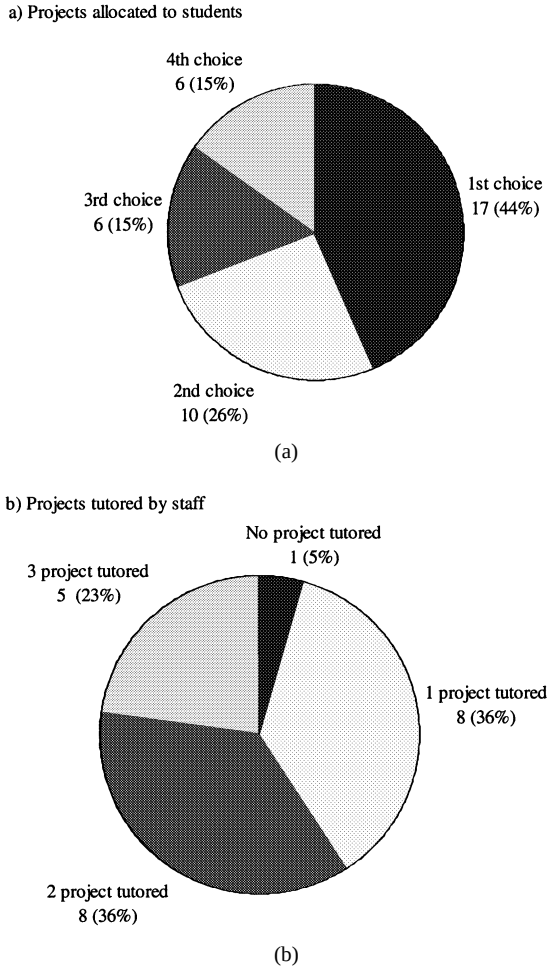

Fig. 2. Output from Model 1a and 1b. (a) Projects allocated to students. (b) Projects tutored by staff.

Fig. 2(a) shows the number of students receiving their first, second, third, and fourth choices. Seventy percent of all students

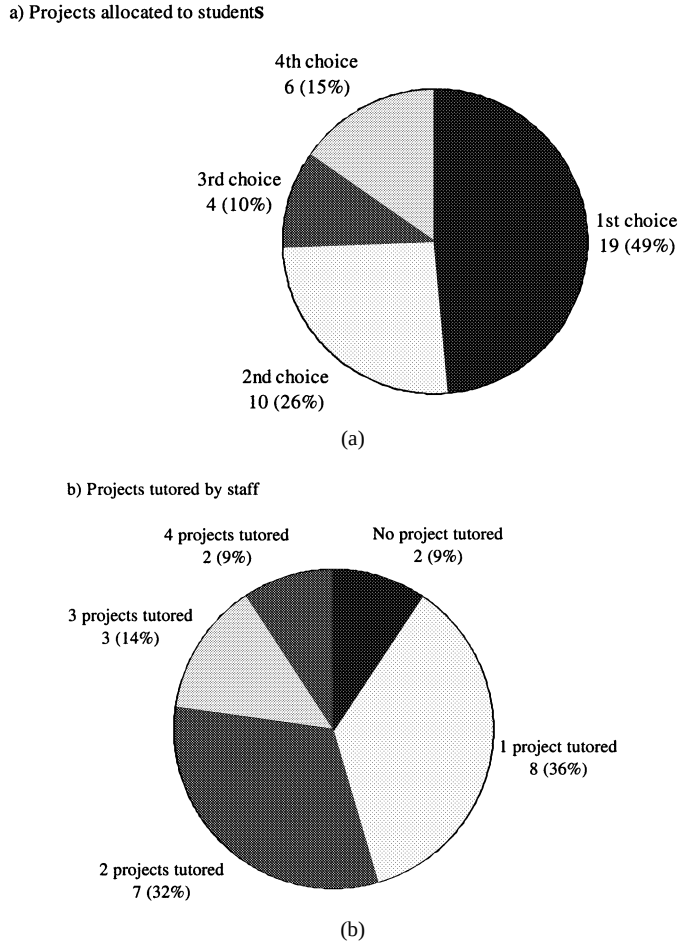

Fig. 3. Output from Model 1b only. (a) Projects allocated to students. (b) Projects tutored by staff.

received either their first or second choices. The distribution of projects among the staff is shown in Fig. 2(b). Only one member of the staff will not be supervising any student on his/her project because this staff member had suggested only one title which was selected by only one student as the fourth choice. The model allocated the student a higher ranked project.

It is instructive to run Model 1b directly without running Model 1a. This procedure can be achieved by removing (11), i.e., Model 1b is run directly without running Model 1a. By running Model 1b directly, more students will be allocated their first choice project. In fact, Fig. 3(a) shows that 49% of students are allocated their first choice, and 5% less students allocated the project of their third choice. However, the number of projects supervised by the staff is less uniform. Fig. 3(b) shows some staff are now supervising four projects, and two members of the staff are not supervising any projects at all.

B. Group Project Model

The group project model was applied to data collected by the Faculty of Engineering and Applied Sciences. The faculty is comprised of a number of departments. For the purpose of the group project the disciplines within the Faculty (which roughly translate to the various departments within the faculty) are shown in Table I.

TABLE I
DISCIPLINES WITHIN THE FACULTY

Discipline	Acronym
Electronics and Computer Science	ECS
Materials	MAT
Acoustics	ACS
Astro and Aeronautical Engineering	AAE
Civil Engineering	CIV
Electrical Engineering	ELE
Mechanical Engineering	MEC
Ship Science	SHS
Institute of Sound and Vibration	ISV
Environmental Engineering	ENE
Electromechanical Engineering	EME

For the academic year 2000–2001, there were 68 students within the Faculty of Engineering and Applied Science required to complete a group project. The students were asked to select three projects from a list of 30 projects suggested by 40 members of the staff. Students were also asked to rank their selection by assigning the integer 3 to their first choice, 2 to their second choice, and 1 to their third choice. In contrast to the individual project, penalties were not imposed if students missed the deadline. Most members of the staff suggested one project, but a few suggested two or three projects. Similarly, most projects were suggested by one member of the staff although a number were jointly suggested by two or three members of the staff. Each project specifies the minimum and maximum number of students and also the suitable disciplines from Table I.

The model was implemented in LINGO 6.0 for Windows (LINGO input file included in the Appendix). It reached a global optimum in approximately three seconds with an objective function value of 175. As an average over the number of students, the objective function is 2.57. A score of three would have indicated that all students obtained their first choice. The objective function for the application of the Individual Project cannot be interpreted directly in this manner because of the penalty factor. The group project model allocated a high proportion of students their first choice. In fact 68% (46 students) are allocated their first choice, with 15 students (22%) allocated their second choice, and only 10% (seven students) were allocated their third choice.

V. COMPARSION WITH ASSIGNPROJ

This section compares the allocation made by the integer program presented in this paper and the AssignProj by Teo and Ho [1]. Teo and Ho [1] described the allocation of projects to 660 students from a choice of 413 projects submitted by 180 members of the staff. Their practice is for students to form pairs and select 10 projects with an indication of the order of preference (ranking) of the choices. This model of allocation is in fact identical to that described in Model 1b earlier for 330 student pairs with the late penalty factor in the objective function (9) removed and constraint (11) also removed. Data was available for 372 student pairs selecting up to 10 projects from a set of 483 projects. Students had ranked their selection to indicate which project was the most preferred and which was the least

TABLE II
OUTPUT FROM ASSIGNPROJ

	AssignProj	Integer Prog.
Count of student pairs allocated 1st choice	204	153
Count of student pairs allocated 2nd choice	28	79
Count of student pairs allocated 3rd choice	13	49
Count of student pairs allocated 4th choice	20	30
Count of student pairs allocated 5th choice	13	27
Count of student pairs allocated 6th choice	8	13
Count of student pairs allocated 7th choice	14	8
Count of student pairs allocated 8th choice	14	9
Count of student pairs allocated 9th choice	8	2
Count of student pairs allocated 10th choice	5	2
Count of students not allocated any project	45	-

preferred. AssignProj was implemented in Java and allowed to run for 24 h as in the original work by Teo and Ho [1]. Table II summarizes the output from AssignProj. Of 372 student pairs, 45 were not allocated any project, i.e., strictly speaking the solution is infeasible. Teo and Ho [1] managed this condition by asking these students to reselect projects from the remaining unallocated projects; hence, these students were effectively allocated their eleventh, twelfth, thirteenth … choices. Table II also shows the output from the integer program. The integer program solves in 6.66 min (as compared to over 24 h runs for AssignProj). Table II shows that the integer program allocates all students a project. Fewer students are allocated their first choice than in the allocation by AssignProj (153 by the integer program as compared to 204 by AssignProj). However, the overall allocation by the integer program is superior. The objective function attains a value of 3135. As an average over a count of student pairs, the objective value is 8.427. In context, a score of 10 would indicate that every student pair was allocated their first choice. Although it is incorrect to calculate the objective value for an infeasible solution, if the infeasibility of the AssignProj allocation is ignored (i.e., the 45 student pairs who have not been allocated a project are ignored), the objective value averaged over a count of the student pairs from AssignProj is 7.454. To make a true comparison between these two methods, the student pairs would have to select more than 10 projects so that running AssignProj leads to a feasible solution, and these results are then compared with the integer program. Notwithstanding the infeasibility, the integer program finds a better solution than AssignProj. Furthermore, it finds this solution in approximately 6.50 min as opposed to 24 h required by AssignProj.

VI. CONCLUSION

Although the project allocation problem is a problem common to many universities in numerous disciplines, a review of published literature has shown it has not received the rigorous investigation that academics normally apply to their own specialties. This paper presents two optimization models to solve the project allocation to criteria and constraints imposed by an institution. The models are generic and can easily be adapted to suit the different criteria and constraints at different institutions. Model 1 solves the problem where students or student groups formed *a priori*, and then each group selects a project. In Model 2 students select a project, and the group is formed from those students allocated any one project. The two models represent very different philosophies of forming groups before selection of projects or forming groups by virtue of the allocation. One could argue in favor or against both of these philosophies, a subject beyond the scope of the current paper.

The models are computationally efficient and solve easily on a desktop PC running suitable software. The models have been formulated using OLE technology to keep the data in a spreadsheet which is easier to modify than a text file, especially when the amount of data becomes large. An additional advantage of this approach is that the models can be converted into a C or FORTRAN executable file and, therefore, can be executed directly without the need for any general purpose solver. The performance of the integer program is compared with AssignProj and is shown to be both computationally more efficient and superior in solution quality. The integer program has been successfully applied to a case study of 40 students and also to a case study of 372 students, verifying that the problem can be applied to small and large scale problems.

For the data presented, both models solved to a global optimum. As a result, there is no solution for which a higher objective function can be obtained; however, there may be solutions for which an equal objective function can be obtained. In practical terms, there may be a situation where the projects allocated to two students can be interchanged without affecting the objective function. In operations research this condition is often termed a "snake-eye" condition, and depending on the data, there can be a number of "snake eye" conditions in such allocation problems and integer programs in general.

For Model 2 in particular, there are a relatively large number of constraints. For certain data sets it is possible that no feasible solution exists. Only a detailed examination of the solution can determine which constraint is causing the infeasibility. That constraint may need to be relaxed e.g., by reducing the minimum number of students on a project, or by increasing the maximum number of students. Alternatively, students may be asked to select and rank five projects rather than just three so that the model has greater flexibility in allocating a student to a project.

If a problem does become infeasible (either Model 1 or Model 2), a strategy may be to rank all unselected projects for all students as 1 (instead of 0), and to rank selected projects with higher integers, i.e., 2, 3, 4 in increasing order of preference. Changing the input data in this manner will avoid the problem of infeasibility but may result in students being allocated projects they had never selected if it improves the objective function! Hence the methods suggested in the preceding paragraphs are preferable alternatives. It is certainly not advisable to resort to this strategy until it is clearly established that the problem is infeasible without this manipulation of data.

APPENDIX I
LINGO INPUT FILES

(Shown on pages 365 and 366.)

Individual Project—Model 1a

```
MODEL:
SETS:
                                STUDENT:                        PENALTY;
                                PROJECT:                        PROJECTSELECTED;
                                STAFF:                          ;
                                TUPLE1(STUDENT, PROJECT):       SELECTION, ALLOCATION;
                                TUPLE2(STAFF, PROJECT):         SUGGESTED, SUPERVISE;
ENDSETS
DATA:
                                STUDENT, PROJECT, STAFF, PENALTY, SELECTION, SUGGESTED =
                                @ole('c:\LINGO6\Models\data1.xls', 'student', 'project',
                                'staff', 'penalty', 'selection', 'suggested');
ENDDATA

MIN = TOTSUPERVISED;

@FOR(PROJECT(j): @SUM(TUPLE1(i,j):ALLOCATION(i,j)) = PROJECTSELECTED(j));
@FOR(TUPLE2(i,j): SUGGESTED(i,j)*PROJECTSELECTED(j) =SUPERVISE(i,j));
@FOR(STUDENT(i): @SUM(TUPLE1(i,j): ALLOCATION(i,j) = 1);
@FOR(PROJECT(j): @SUM(TUPLE1(i,j): ALLOCATION(i,j)) ≤ 1);
@FOR(STAFF(i): @SUM(TUPLE2(i,j):SUPERVISE(i,j)) ≤ TOTSUPERVISED);
@FOR(TUPLE1(i,j): @BIN(ALLOCATION(i,j)));
@FOR(STUDENT(i): @SUM(TUPLE1(i,j): ALLOCATION(i,j) * SELECTION(i,j)) ≥ 1);

END
```

Individual Project—Model 1b

```
MODEL:
SETS:
                                STUDENT:                        PENALTY;
                                PROJECT:                        PROJECTSELECTED;
                                STAFF:                          ;
                                TUPLE1(STUDENT, PROJECT):       SELECTION, ALLOCATION;
                                TUPLE2(STAFF, PROJECT):         SUGGESTED, SUPERVISE;
ENDSETS
DATA:
                                STUDENT, PROJECT, STAFF, PENALTY, SELECTION, SUGGESTED =
                                @ole('c:\LINGO6\Models\data backup.xls', 'student', 'project',
                                'staff', 'penalty', 'selection', 'suggested');
ENDDATA

MAX = @SUM(TUPLE1(i,j): ALLOCATION(i,j)*SELECTION(i,j)*PENALTY(i));

@FOR(PROJECT(j): @SUM(TUPLE1(i,j):ALLOCATION(i,j)) = PROJECTSELECTED(j));
@FOR(TUPLE2(i,j): SUGGESTED(i,j)*PROJECTSELECTED(j) = SUPERVISE(i,j));
@FOR(STUDENT(i): @SUM(TUPLE1(i,j): ALLOCATION(i,j)) = 1);
@FOR(PROJECT(j): @SUM(TUPLE1(i,j): ALLOCATION(i,j)) ≤ 1);
@FOR(STAFF(i): @SUM(TUPLE2(i,j):SUPERVISE(i,j)) ≤ 3);

!NOTE: RHS value of 3 is objective function from Model 1a;
@FOR(TUPLE1(i,j): @BIN(ALLOCATION(i,j)));
@FOR(STUDENT(i): @SUM(TUPLE1(i,j): ALLOCATION(i,j) * SELECTION(i,j)) ≥ 1);

END
```

Group Project Model

```
MODEL:
SETS:
                                STUDENT:                        ;
                                PROJECT:                        PROJECTSELECTED, MINGROUPSIZE,
                                                                MAXGROUPSIZE, GROUPSIZE;
                                STAFF:                          ;
                                DEPARTMENT:                     ;
                                TUPLE1(STUDENT, PROJECT):       SELECTION, ALLOCATION, SUITABLESTUD;
                                TUPLE2(STAFF, PROJECT):         SUGGESTED, SUPERVISE;
                                TUPLE3(DEPARTMENT, PROJECT):    SUITABLEDISCIP, MINIMUMSTUDENT;
                                TUPLE4(STUDENT, DEPARTMENT):    HOME;
ENDSETS
DATA:
                                STUDENT, PROJECT, STAFF, DEPARTMENT, MINGROUPSIZE, MAXGROUPSIZE,
                                SELECTION, SUGGESTED, SUITABLEDISCIP, MINIMUMSTUDENT, HOME
                                =@ole('c:\LINGO6\Models\gdpdata.xls',
                                'STUDENTNAME', 'PROJECTNAME', 'STAFFNAME', 'DEPTABBREV',
                                'MINGROUPSIZE', 'MAXGROUPSIZE', 'SELECTION', 'SUGGESTED',
                                'SUITABLEDISCIP', 'MINIMUMSTUDENT',
                                'HOME');
ENDDATA

MAX = @SUM(TUPLE1(i,j): ALLOCATION(i,j)*SELECTION(i,j));

@FOR(PROJECT(j):@SUM(TUPLE1(i,j):ALLOCATION(i,j)) = GROUPSIZE(j));
@FOR(PROJECT(j): PROJECTSELECTED(j) * 100 >= GROUPSIZE(j););
@FOR(TUPLE2(i,j): SUGGESTED(i,j)*PROJECTSELECTED(j) = SUPERVISE(i,j));
@FOR(STUDENT(i):@FOR(PROJECT(j):@SUM(DEPARTMENT(k):HOME(i,k)*SUITABLEDISCIP(k,j)) =
          SUITABLESTUD(i,j)));
@FOR(STUDENT(i):@SUM(TUPLE1(i,j): ALLOCATION(i,j)) = 1);
@FOR(PROJECT(j):@SUM(TUPLE1(i,j): ALLOCATION(i,j)) <= MAXGROUPSIZE(j););
@FOR(PROJECT(j):@SUM(TUPLE1(i,j): ALLOCATION(i,j)) >= MINGROUPSIZE(j)*PROJECTSELECTED(j));
@FOR(TUPLE1(i,j): @BIN(ALLOCATION(i,j)));
@FOR(PROJECT(j): @BIN(PROJECTSELECTED(j)));
@FOR(STUDENT(i): @SUM(TUPLE1(i,j): ALLOCATION(i,j) * SELECTION(i,j)) >= 1);
@FOR(STUDENT(i):@FOR(PROJECT(j): SUITABLESTUD(i,j) >= ALLOCATION(i,j)));

END
```

ACKNOWLEDGMENT

Trade names and company names are used in this paper solely for the purpose of providing specific information. Their mention does not constitute a guarantee or warranty or endorsement of the company or product by the author or by the Department of Civil and Environmental Engineering, University of Southampton, U.K. The authors would like to acknowledge Prof. C.Y. Teo and Dr. J. Ho, Nanyang Technological University, for providing some of the data used in this work, and also Dr. A.C. Lock , University of Southampton, for his assistance in manipulating this data.

REFERENCES

[1] C. Y. Teo and D. J. Ho, "A systematic approach to the implementation of final year project in an electrical engineering undergraduate course," *IEEE Trans. Educ.*, vol. 41, pp. 25–30, Feb. 1998.

[2] K. A. Dowland, "Genetic algorithms—A tool for OR?," *J. Operat. Res. Soc.*, vol. 47, pp. 550–561, 1996.

[3] *LINGO User's Guide; version 6.0*. Chicago, IL: Lindo Systems Inc., 1999.

[4] L. Schrage, *Optimization Modeling With LINGO*, IL: Lindo Systems Inc., 1999.

A. A. Anwar received the B.Sc. (honors) degree in civil engineering and the M.Sc. degree from the same university in water resources engineering from the N-W.F.P. University of Engineering and Technology, Peshawar, Pakistan, in 1988 and 1996, respectively. He received the M.Sc. degree in irrigation engineering and the doctorate from the University of Southampton, Southampton, U.K., in 1995 and 2001, respectively.

He worked for the Irrigation Department, Government of N-W.F.P. Pakistan, on water resources planning projects and rehabilitation projects. In 1994, he was awarded the Chevening Scholarship for postgraduate study in the U.K. He joined the Department of Civil and Environmental Engineering, University of Southampton, in 1997. He is currently a Lecturer and Undergraduate Admissions Tutor in this department. He has a keen interest in optimization from his work on the Indus Basin Model, and more generally in the application of IT to engineering problems.

A. S. Bahaj (M'99) received the B.Sc. (honors) degree in electrical engineering in 1976, and the Ph.D. degree in 1981, from the University of Southampton, Southampton, U.K.

He is now a Senior Lecturer and heads the Sustainable Energy Research Group, Department of Civil and Environmental Engineering. His main research and teaching interests are in renewable energy power generation. He has published more than 80 publications and has experience in managing and coordinating large research programs in energy and environmental studies. In particular, the use of solar energy in buildings, refrigeration, and in engineering for human development. This work is now being augmented by research into other renewable energy areas such as the utilization of marine currents for electrical power production and in the study of energy issues in the built environment. Some of this work feeds into the work of the International Energy Agency (IEA).