Pergamon

Omega, Int. J. Mgmt Sci. Vol. 25, No. 4, pp. 473–482, 1997
© 1997 Elsevier Science Ltd. All rights reserved
Printed in Great Britain
0305-0483/97 $17.00 + 0.00

PII: S0305-0483(97)00007-8

An Empirical Comparison of Heuristic and Graph Theoretic Methods for Creating Maximally Diverse Groups, VLSI Design, and Exam Scheduling

RR WEITZ

Seton Hall University, South Orange, NJ, USA

S LAKSHMINARAYANAN

MinMax Consulting, New York, NY, USA

(Received July 1996; accepted after revision January 1997)

Creating groups of maximum diversity, VLSI design (where the objective is to group highly connected modules onto the same circuit), and the final exam scheduling task of assigning exam blocks to exam days, are all mathematically equivalent. VLSI design and exam scheduling are clearly important problems. Forming maximally diverse groups, based upon multiple criteria, has immediate application in academic or training settings where it may be desired to create class sections, or project groups within classes, such that students are immersed in a diverse environment. This research empirically contrasts graph theoretic approaches drawn from VLSI design with heuristics originating from final exam scheduling and the maximum diversity group problem. The methods are tested on a 'real-world' data set and evaluated on the criteria of solution quality and computational resources required. A principal conclusion of this work is that an adaptation of a pair-wise exchange procedure drawn from the final exam scheduling literature outperforms a more sophisticated graph theoretic approach which has previously been shown to be a top performer for VLSI design. © 1997 Elsevier Science Ltd

Key words—graph theory, timetabling, heuristics, education

1. INTRODUCTION

THE FOLLOWING PROBLEMS may be shown to be mathematically equivalent: (1) forming maximally diverse groups (based upon multiple criteria) of any specified size from a given population, (2) the final exam scheduling sub-problem of assigning exam blocks to days, and (3) the VLSI design task of grouping 'highly connected' modules onto the same circuit. The problem has been shown to be NP-complete [1, 2], hence heuristic approaches are important for sizeable applications. Weitz and Lakshminarayanan [3] explored the performance of 'switching' heuristics drawn from the student workgroup and exam scheduling areas. This article empirically tests the best of these switching approaches with graph theoretic heuristics developed for the VLSI design task.

The utility of the exam scheduling and VLSI applications are clear. Forming maximally diverse groups has immediate application in academic or training settings where it may be desired to create class sections, or project groups within classes, such that students are immersed in a maximally diverse environment. Creating diverse student workgroups seems to have become increasingly common in business schools, apparently in response to the corporate realities of an increasingly diverse workforce and a growing reliance on teams as an organizational structure.

2. FORMAL REPRESENTATIONS OF THE PROBLEM

The problem may be formulated as a quadratic integer program as follows.

$$\max: \sum_{p=1}^{G}\sum_{i=1}^{N-1}\sum_{j>i}^{N} d_{ij}x_{ip}x_{jp} \qquad (1)$$

subject to:

$$\sum_{p=1}^{G} x_{ip} = 1 \text{ for } i = 1,2,\cdots,N \qquad (2)$$

$$\sum_{i=1}^{N} x_{ip} = S \text{ for } p = 1,2,\cdots,G \qquad (3)$$

Using student workgroup terminology, there are N students, G groups, and $N/G = S$ students in each group.

d_{ij} = Difference between students i and j.
x_{ip} = 1 if student i is in group p,
0 otherwise.

The problem as formulated resembles the well-known quadratic assignment problem [4], classically applied to assigning facilities to locations.

For the student group assignment problem, the d_{ij}'s may be obtained by summing the weighted contributions of all the criteria by which two students being compared differ. Weights reflect the relative importance of each criterion. Details of the student workgroup assignment problem and its implementation are discussed by Weitz and Jelassi [5]. For other applications, the d_{ij}'s are typically evident from the problem.

Lotfi and Cerveny [6] address the final exam scheduling problem at a large university by decomposing the problem into four sub-problems. First, exams are assigned to exam periods, or 'blocks', with the objective of minimizing the number of students with simultaneous exams. Second, blocks are assigned to exam days in such a way as to minimize the number of students with two or more exams per day. Third, the exam days and exam blocks are arranged within days to minimize the number of students with consecutive exams. Finally, exams are assigned to classrooms with the objective of maximizing space utilization.

The sub-problem of assigning exam blocks to days is mathematically equivalent to the minimization version of the student-group IP presented above. For this application,

d_{ij} = Number of students with exams in both block i and block j.
x_{ip} = 1 if exam block i is assigned to day p,
0 otherwise.

An equivalent graph theory representation is given by Feo, Goldschmidt and Khellaf in their work on the VLSI design problem.

> Consider a complete graph G = (V,E), with nonnegative edge weights, |V| = km, and integers k and m. The maximization version of the k-partition problem seeks to cluster the vertices of G into k sets of m nodes each, such that the total weight of the edges having both endpoints in the same cluster... is maximized. This problem is equivalent to minimizing the total weight of the edges having both endpoints in different clusters... [7, p. S170].

Using student workgroup terminology, the edge weights are the d_{ij}'s, the nodes are students. Partitioning the graph into sets is equivalent to allocating students to groups.

For consistency, across the three domains we will remain with the variable naming convention used for student workgroups—that is, N total students, G groups, and S students per group. (Above Feo and Khellaf use $|V|$ 'students', k 'groups' and m 'students per group'). Finally, we will use G_g when referring to a graph, to avoid any confusion with G groups.

3. LITERATURE REVIEW

In this section, we review the relevant literature on exam scheduling, group formation and VLSI design—the three mathematically equivalent applications outlined above and addressed in this empirical study. We also provide a brief description of research in related areas.

3.1. Exam scheduling

For the problem of assigning exam blocks to periods, the heuristic of Lotfi and Cerveny [6] starts with a random assignment and then uses a guided pairwise switching approach to improve solution quality. Weitz and Lakshminarayanan [8] corrected several errata in Lotfi and Cerveny's paper, and suggested an improvement to the heuristic.

For the same task, Arani and Lotfi [9] use a heuristic based on 'the usual' assignment problem. In their relatively small application, they use the output of their heuristic not as a solution in itself, but as an initial upper bound for a branch and bound procedure they use in solving an integer programming formulation of the problem. Their results indicate that the heuristic performed on average within 5% of the branch and bound solution.

3.2. Forming maximally diverse groups

Weitz and Jelassi [5] developed and implemented a heuristic-based, multiple-criteria decision support system for creating maximally diverse groups at the European Institute of Business Administration (INSEAD). (Their heuristic is method WJ described below.) A modified version was subsequently implemented at the Stern School of Business of New York University (NYU) [2]. As part of the NYU project, a bound for the heuristic was developed, several integer programming (IP) versions of the problem were formulated, and empirical trials were conducted comparing the performance of the heuristic with that of the IP approaches. Weitz and Lakshminarayanan [3] empirically contrasted five different heuristics, drawn from student-workgroup assignment and final exam scheduling applications. A principal conclusion of this work was that their modified version of Lotfi and Cerveny's approach consistently provided the best results with no computational penalty. (This is method LCW below.)

3.3. VLSI design: Graph theory

Here, the task is to group highly connected modules onto the same circuit. Feo and Khellaf [1] show that the problem is NP-complete, and devise a bounded, 'matching-based' set of heuristics using an approach drawn from graph theory. Two general methods are detailed: P1 is for even-sized groupings, P2 for odd. They show that if the 'edge weights' (in our case the d_{ij}'s) satisfy the triangle inequality (as do the student differences), the value of the optimal solution is at most $2(S-1)/S$ times the value of the solution provided by P1; for P2 this bound on the optimal solution is $2S/(S+1)$. When the triangle inequality does not hold, as in the final exam scheduling problem, the bounds are $S-1$ and S, respectively.

Feo and Khellaf do not report any empirical results for methods P1 and P2; however, empirical trials are conducted using an adaptation of P1 which, in its general form, requires a group size of 2^i, where i is an integer. Here, the adaptation was limited to problems of a fixed group sizes of four ($i = 2$), though a postprocessor exchange procedure can expand the group size to five. This adaptation of P1, called here P4, is given below. Feo and Khellaf tested P4 and two variations; the variations utilized different heuristics in an attempt to avoid the computational expense of finding maximum weight perfect matchings. The performance of P4 and the variations on ten small problem sets was compared with those of two probabalistic exchange procedures, N-step lookahead and simulated annealing. The matching-based variations were competitive with the probabalistic approaches in terms of solutions, and superior in terms of computation time. P4 and the variations were then compared on five modifications of each of these ten problems, the modifications being created by randomly altering each nonzero edge weight. Last, for five small, randomly generated problems they were compared with a branch and bound (optimal solution) approach and again they provided good quality solutions, with significantly reduced processing time. Overall the heuristic using the matching algorithm, that is P4, demonstrated superior performance compared with the variations, and is therefore used here.

In subsequent work, Feo, Goldschmidt and Khellaf [7] reconsider P4 and show that the value of the optimal solution is at most twice the value of the solution provided by the heuristic. (This is a tighter bound for the case where the triangle equality does not hold.) They also develop a different bounded heuristic (not considered here) for group sizes of three.

3.4. Related areas

The problem may be generalized as the task of assigning a set of items to a limited number of entities, under a specified set of constraints, with the intent of maximizing some overall utility function. The body of work in these areas is extensive; a survey of related assignment problems is provided by Lakshminarayanan and Weitz [2].

Related academic applications involve assigning students to groups, with a variety of objectives. Groups may be project teams, laboratories, or job interview slots; typically the problem has one or two objectives, for example, accommodating preferences or balancing groups based on criteria such as experience, previous academic performance, or nationality. Typically, mathematical programming or heuristic methodologies are utilized. The mathematical programming approaches include linear, integer, mixed linear/integer, multiple criteria and goal programming models. Heuristics tend to be fairly simple sequential procedures.

Kuo, Glover and Dhir [10] use several mathematical programming approaches for selecting a single maximally diverse group, based on multiple criteria, from a given population. Most closely related to this research is the work of Mingers and O'Brien [11, 12] who focus on creating similar student groups. (Dispersing each student characteristic as evenly as possible between groups implies maximizing the distribution of characteristics within groups, and is hence directly related to the maximum diversity problem.) Last, the problem under study here bears a resemblance to cellular manufacturing applications. Here the idea is to improve overall manufacturing performance by grouping parts into part families (based on similar processing requirements), and machines into 'cells' which can satisfy the processing requirements of one or more families.

For a more detailed review of the literature in these related areas see [3].

4. AN INTEGER BOUND

An integer upper bound superior to the relaxation of the integer programming formulation of the problem was developed by Lakshminarayanan and Weitz [2]. This bound is determined by considering the matrix of pairwise differences between all students, and calculating the difference metric using the largest $(S-1)$ differences for each student. Formally, the Lakshminarayanan and Weitz (LW) integer bound is given by

$$\tfrac{1}{2}\sum_{i=1}^{N}\sum_{j\in M(i)} d_{ij}$$

where $M(i)$ is the largest $(S-1)$ differences for student i.

One objective of this research is to compare empirically the LW bound with those provided by Feo and Khellaf [1] and Feo, Goldschmidt and Khellaf [7].

5. THE HEURISTICS

In this section, we describe the heuristics tested here in detail.

5.1. Student-assignment: Method 'Weitz–Jelassi' (WJ)

The basic heuristic implemented at INSEAD and NYU works by avoiding placing the most similar students in the same group. The first student, usually selected randomly, is placed in the first group. The heuristic then selects the student least different from the first student and places him/her in the next group. The model continues in this fashion, at each iteration taking the student least different to the previous student and placing him/her in the next group.

As the method is computationally quite simple, for any conceivable real-world circumstance it would be possible to run the method 'N-times', each time starting with a different student, and then select the best of the generated solutions. Though it has been shown previously to be outperformed by method LCW, it is included here as a 'baseline' measure.

5.2. Exam scheduling: Method 'Lotfi–Cerveny–Weitz' (LCW)

The following pairwise switching heuristic is based on the work of Lotfi and Cerveny ([6], Appendix B) it includes the corrections and modifications noted previously. We have replaced exam blocks with students, and exam days with groups, and switched the objective from minimization to maximization.

Input. The d_{ij}'s, for a set of N students, the number of groups to be formed (G) and the number of students per group (S).

1. Create an arbitrary initial solution.

$x_{ip} = 1$ for $i = (p-1)*S+1, (p-1)*S+2, \ldots, (p-1)*S+S$, and $p = 1, 2, \ldots, G$,
0 otherwise.

2. For the present assignment, determine the matrix R which specifies for all students i and all groups p, the contribution to the total difference (a) that is a result of student i currently being assigned to group p, or (b) that would result if student i (not currently in group p) were added to group p. Let $D = [d_{ij}]$ and $X = [x_{ip}]$, determine $R = DX$. Set $i = 0$, and Flag = false, and go to step 3.

3. Select (the next) student i as a potential candidate for a switch. Increment i. If $i > N$, go to step 6. Otherwise go to step 4.

4. Determine which (if any) student j should be switched with student i. In row i of X find column t with $x_{it} = 1$. For each column $q \neq t$, consider each row j, in which $x_{jq} = 1$, let $wj = (r_{iq} - r_{it}) + (r_{jt} - r_{jq}) - 2*d_{ij}$. Let $w_k = \max w_j$. If $w_k > 0$, (that is, the objective function increases if students i and j are switched) go to step 5, else go to step 3.

5. Revise X to incorporate the switch. Recalculate R. Set Flag to true to indicate that a switch has taken place. Let $x_{kq} = 0$, $x_{kt} = 1$, and $x_{iq} = 1$, and $x_{it} = 0$. Let $R = DX$, Flag = true and go to step 3.

6. If no students have been switched, stop. Otherwise reset the Flag and i, and go to step 3. If Flag = false, stop. No further improvement is possible. Otherwise, let Flag = false, $i = 0$, and go to step 3.

5.3. VLSI design: Feo–Khellaf procedure P1 (FKP1) (for even group sizes)

The Feo–Khellaf procedures rely on perfect weight matching procedures.

```
Input.   A complete graph $G_g$, $N = GS$, $S$ even, and a
         nonnegative weight $w_e$ ($d_{ij}$) for each edge $e$.
Step 1.  Find a maximum weight perfect matching $M$ in
         $G_g$.
Step 2.  DO $i = 1, G$
           Choose arbitrarily a set, $\Pi$, of $S/2$ nonselected
           edges of $S$.
           Mark the edges in $\Pi$ selected.
           Place in cluster $i$ the nodes incident to the edges
           in $\Pi$.
```

The maximum weight perfect matching problem may be solved, in $O(N^3)$ time, via a maximum weight perfect matching algorithm [13]. We used a publicly available program described by Johnson and McGeoch [14].

As this procedure assigns pairs of students to groups, it must be adapted for odd group sizes.

5.4. VLSI design: Feo–Khellaf procedure P2 (FKP2) (for odd group sizes)

```
Input.   A complete graph $G_g$, $N = GS$, $S$ odd, and a
         nonnegative weight $w_e$ ($d_{ij}$) for each edge $e$.
Step 1.  Find a maximum weight matching $M$ in $G_g$
         containing $(S-1)N/2S$ edges.
Step 2.  DO $i = 1, G$
           Choose arbitrarily a set, $\Pi$, of $(S-1)/2$
           nonselected edges of $M$. Mark the edges in $\Pi$
           selected. Place in cluster $i$ the nodes incident to
           the edges in $\Pi$, and any one node not incident
           to $M$ which is not already included in a cluster.
```

Following Feo and Khellaf [1, p. 185], we 'tricked' the maximum weight perfect matching algorithm for use here. The idea is simply to add to the existing set of students (and d_{ij} matrix) G 'dummy' students with very low difference values between them, and very high difference values between each of them and each actual student. (Adding the G students increases the group size for the algorithm to $S + 1$, an even number). The algorithm then finds the maximum weight perfect matching (with $G(S+1)/2\ d_{ij}$'s), G of which include d_{ij}'s pairing dummy students with actual students. Deleting these Gd_{ij}'s from the set, produces the desired $G(S-1)/2\ d_{ij}$'s.

The G students paired with the 'dummies' are then, according to Feo and Khellaf, assigned randomly, one to a group, to complete the partition. We decided to try an improvement whereby this assignment, rather than being done randomly, would be done optimally. Optimal here means that these G students are assigned such that the total additional diversity, beyond that of the $N - G$ students already assigned, is maximized. (This approach is similar to one used by Arani and Lotfi [9].) To implement this procedure we adapted a publicly available classical assignment problem subroutine [15].

5.5. VLSI design: Feo–Khellaf Heuristic 1 (FKP4) (for forming groups of size four)

```
Input.   A complete graph $G_g$, $N = 4G$, and a weight $w_e$
         ($d_{ij}$) for each edge $e$.
Step 1.  Find a maximum weight perfect matching $M_1$
         of $G_g$.
Step 2.  Construct $G_g'$ by contracting all the edges of
         $M_1$, and combine resulting parallel edges by
         summing their edge weights.
Step 3.  Find a maximum weight perfect matching $M_2$
         of $G_g'$.
Step 4.  Form the $k$ disjoint clusters of 4 vertices each
         by contracting all the edges of $M_2$.
```

6. EXPERIMENTS

We used a real-world dataset to test each of the above methods; our aim was to compare the methods based on the quality of the solutions they provided, and the computational resources required. Solution quality was measured by the total diversity of the resulting partition (given mathematically by the objective function of the mathematical program). CPU processing time was used to gauge the computational resources required.

6.1. The data

The above techniques were tested using Fall 1993 data from the Stern School at NYU. We deleted four arbitrarily selected students from the 1993 data set, leaving 360 students in the class.

NYU requirements (which appear to be fairly typical for this application), dictated partitioning the entire class into six blocks of 60 students each, and each block into ten working groups of six students each. We conducted additional trials on the following problems: partitioning ten students into two groups, 12 students into three and four groups, 60 students into 12 and 15 groups and 120 students into 24 and 30 groups. These problems were chosen as: (1) The smaller problems are on the order of the final exam scheduling and VLSI examples reported in the literature and (2) it was of interest to compare the methods on both large and small problems.

6.2. The Weitz–Jelassi heuristic

For WJ, the experiment proceeded in the following manner. (It might be helpful to consider the $N = 60$ student, $G = 15$ group, $S = 4$ students per group problem, noting that for the population of 360 students, there are a total of $360/6 = 6$ sets of 60 students.)

```
1. Randomize the list of students.
2. Select the first $N$ students.
3. DO $i = 1,N$
   (3a) Apply the heuristic to this set of $N$ students, starting with student $i$, with $G$ groups of size $S$ formed.
   (3b) For each group calculate the sum of the (different) pairwise differences; the sum of these $G$ within-group differences is the measure of the quality of the solution reached by the heuristic, starting with that particular student. (This is the objective function of the IP representation of the problem.)end DO[1]
4. Record the maximum of the $N$ objective function values.
```

The above steps are then repeated, at step two choosing the next set of N students in the list each time, for a total of $360/N$ sets.

In addition to the best WJ results, for each set the following data were collected: (1) the average solution quality for 100 arbitrary allocations and (2) the results of applying WJ with the objective of minimizing diversity. The 100 random allocations are provided as a baseline measure. The WJ heuristic can easily be adapted for minimizing diversity by directing it to select the most different student at each iteration; these ‘worst case’ results are provided to indicate the range of the distribution of possible solutions.

Detailed results (for all methods) for the 60 student, 15 group and 60 student, 12 group problems are provided in Table 1. (Detailed results for all problems may be obtained from the authors.) Summary results (over all sets) for all problems are in Table 2.

6.3. Other methods

In order to avoid effects due to the order of the students, the following procedure was followed for method LCW.

```
1. Start with the randomized list of students.
2. Select the first $N$ students.
3. DO $j = 1,100$
   (a) Apply method LCW and record the solution value.
   (b) Re-randomize the $N$ students in the current set.end DO
4. Calculate and record the average of the 100 objective function values for this set.
```

As before, this process is then repeated for the remaining sets.

Again, detailed results, in this case the averages of the 100 values for each set are reported in Table 1 for the 60 student, 15 group and 60 student, 12 group problems. Summary results over sets are provided in Table 2.

Method FKP1 was applied to each set for even-sized group problems; method FKP4 (power of two) was additionally used on those

[1]Note: This process yields N (not necessarily unique) partitions, one for each starting student, and N corresponding objective function values.

Table 1. Sample detailed results

No. students, No. groups $N = 60$, $G = 15$	WJ 'worst' (minimize diversity)	Average of 100 random partitions	WJ 'best' (maximize diversity)	LCW (average with 100 random starts)	FKP1	FKP4 power = 2	LW bound	FKP1 Bound	LW bound/ FKP1 bound	%Improv WJ worst to LCW	%Improv 100 avg. to LCW	%Improv WJ best to LCW	%Improv FKP1 to LCW	%Improv FKP4 to LCW	LCW/LW bound
Set															
1	41.9120	43.1065	49.1300	50.8532	48.6180	50.1020	68.7310	72.9270	0.9425	21.33	17.97	3.51	4.60	1.50	0.74
2	37.7740	37.0870	42.2960	43.4486	41.5220	42.4700	67.9000	62.2830	1.0902	15.02	17.15	2.72	4.64	2.30	0.64
3	34.9440	35.3158	39.6440	41.1821	37.7820	40.7700	66.2400	56.6730	1.1688	17.85	16.61	3.88	9.00	1.01	0.62
4	39.1740	38.8589	43.3760	45.4499	43.2220	44.7960	66.6780	64.8330	1.0285	16.02	16.96	4.78	5.15	1.46	0.68
5	41.0720	38.6812	44.5300	45.6713	42.9360	45.2840	65.4910	64.4040	1.0169	11.20	18.07	2.56	6.37	0.86	0.70
6	37.9040	39.8550	46.5660	47.9224	45.0160	47.4860	67.1440	67.5240	0.9944	26.43	20.24	2.91	6.46	0.92	0.71
Mean	38.7967	44.2570	45.7546	45.7546	43.1827	45.1513	67.0307	64.7740	1.0402	17.98	17.83	3.39	6.04	1.34	0.68
Std. dev.	2.5184	3.3182	3.3770	3.3770	3.6025	3.3612	1.1645	5.4038	0.0792	5.32	1.31	0.84	1.67	0.55	0.04

No. students, No. groups $N = 60$, $G = 12$	WJ 'worst' (minimize diversity)	Average of 100 random partitions	WJ 'best' (maximize diversity)	LCW (average with 100 random starts)	FKP2	FKP2 power = 2	LW bound	FKP2 Bound	LW bound/ FKP2 bound	%Improv WJ worst to LCW	%Improv 100 avg. to LCW	%Improv WJ best to LCW	%Improv FKP2 to LCW	%Improv FKP2 to LCW	LCW/LW bound
Set															
1	52.6800	57.6048	64.5840	66.0005	60.0620	62.8380	90.4440	100.1034	0.9035	25.29	14.57	2.19	9.89	5.03	0.73
2	45.9220	49.4406	54.9720	56.0485	54.4940	54.9260	89.1670	90.8233	0.9818	22.05	13.37	1.96	2.85	2.04	0.63
3	45.4540	46.5518	52.5140	53.2779	51.9620	52.2000	86.2690	86.6033	0.9961	17.21	14.45	1.45	2.53	2.06	0.62
4	47.1060	51.4838	57.3820	58.8503	56.9000	57.7280	87.7341	94.8333	0.9251	24.93	14.31	2.56	3.43	1.94	0.67
5	45.1280	51.9801	57.8520	59.1567	56.8180	57.7120	86.2060	94.6967	0.9103	31.09	13.81	2.26	4.12	2.50	0.69
6	47.7300	53.6636	60.1640	61.1340	57.0440	58.7300	88.4010	95.0733	0.9298	28.08	13.92	1.61	7.17	4.09	0.69
Mean	47.3367	51.7875	57.9113	59.0780	56.2133	57.3557	88.0368	93.6889	0.9411	24.78	14.07	2.01	5.00	2.95	0.67
Std. dev.	2.7994	3.7526	4.1870	4.3595	2.7338	3.5971	1.6597	4.5564	0.0385	4.81	0.46	0.42	2.92	1.30	0.04

Note: %Improv A to $B = ((B - A)/A) \times 100$

Table 2. Summary results[a]

Even-sized groups

No. students, No. groups, No. sets	WJ 'worst' (minimum diversity)	Average of 100 random partitions	WJ 'best' (maximum diversity)	LCW (average with 100 random starts)	FKP1	FKP4 power = 2	LW bound	FKP1 bound	LW bound/ FKP1 bound	%Improv WJ worst to LCW	%Improv 100 avg. to LCW	%Improv WJ best to LCW	%Improv FKP1 to LCW	%Improv FKP4 to LCW	LCW/LW bound
$N = 12$, $G = 3$, No. sets = 30	7.67	7.71	8.56	8.60	8.31	8.48	11.33	12.46	0.91	11.97	11.53	0.52	3.71	1.51	0.76
$N = 60$, $G = 10$, No. sets = 6	58.40	64.59	70.88	72.10	68.01		108.46	113.35	0.96	23.40	11.64	1.71	6.11		0.65
$N = 60$, $G = 15$, No. sets = 6	38.80	44.26	45.75	45.75	43.18	45.15	67.03	64.77	1.03	17.98	17.83	3.39	6.04	1.34	0.68
$N = 120$, $G = 30$, No. Sets = 3	50.41	57.08	66.71	69.19	64.73	68.60	104.44	97.09	1.08	37.34	21.21	3.68	6.94	0.86	0.66
$N = 360$, $G = 6$, No. sets = 1	2960.92	3398.58	3438.59	3442.58	3397.92		7283.20	6682.58	1.09	16.27	1.29	0.12	1.31		0.47
Averages (even-sized group)									1.01	21.39	12.70	1.88	4.82	1.24	0.65

Odd-sized groups

No. students, No. groups, No. sets	WJ 'worst' (minimum diversity)	Average of 100 random partitions	WJ 'best' (maximum diversity)	LCW (average with 100 random starts)	FKP2	FKP2 power = 2	LW bound	FKP2 bound	LW bound/ FKP2 bound	%Improv WJ worst to LCW	%Improv 100 avg. to LCW	%Improv WJ best to LCW	%Improv FKP2 to LCW	%Improv FKP2 to LCW	LCW/LW bound
$N = 10$, $G = 2$, No. sets = 36	6.98	8.52	9.12	9.12	8.88	8.97	11.51	14.80	0.78	32.28	7.08	0.07	2.73	1.79	0.79
$N = 12$, $G = 4$, No. sets = 30	4.32	5.12	5.98	6.05	5.88	6.02	7.90	8.82	0.89	40.87	18.06	1.22	2.78	0.51	0.76
$N = 60$, $G = 12$, No. sets = 6	47.34	51.79	57.91	59.08	56.21	57.36	88.04	93.69	0.94	24.78	14.07	2.01	5.00	2.95	0.67
$N = 120$, $G = 24$, No. sets = 3	67.09	76.47	85.34	88.13	83.64	85.15	138.24	139.39	0.99	31.60	15.22	3.25	5.35	3.48	0.64
Averages (odd-sized groups)									0.90	32.38	13.61	1.64	3.96	2.18	0.72
Averages (all groups)									0.96	26.27	13.10	1.77	4.44		0.68

[a] Results averaged over sets.

Note: %Improv. A to $B = ((B - A)/A) \times 100$

Table 3. Average solution times (cpu, s)

Even-sized groups

No. students, No. Groups, No. sets	WJ best solution	LCW (time for 100 runs/100)	FKP1	FKP4 power = 2
$N = 12$, $G = 3$, No. sets = 6	0.0170	0.0025	0.3717	0.7887
$N = 60$, $G = 10$, No. sets = 6	2.5617	0.2057	1.1317	
$N = 60$, $G = 15$, No. sets = 6	2.4733	0.2139	1.0550	2.8583
$N = 120$, $G = 30$, No. sets = 3	37.1067	1.7123	3.7067	9.9833
$N = 360$, $G = 6$	2875.7400	21.3049	72.6200	

Odd-sized groups

No. students, No. groups, No. sets	WJ best solution	LCW (time for 100 runs/100)	FKP2 random assign.	FKP2 optimal assign.
$N = 10$, $G = 2$, No. sets = 36	0.0106	0.0012	0.3461	0.3392
$N = 12$, $G = 4$, No. sets = 30	0.0140	0.0026	0.3587	0.3403
$N = 60$, $G = 12$, No. sets = 6	2.4450	0.2191	1.2950	1.2717
$N = 120$, $G = 24$, No. sets = 3	37.7633	1.6878	5.0867	5.1767

Note: FK and WJ figures are the sum of the times for each set, divided by the number of sets.
LCW figures are the sum of average times for 100 solutions for each set, divided by the number of sets.

problems with group size of four. Method FKP2 was applied to each set for odd-sized group problem; results were obtained for both the original random allocation of the last G students, and our optimal allocation modification.

Finally, bound results were obtained using our formulation, and those provided for P1 and P2 by Feo and Khellaf [1]. As the triangle inequality holds for the student workgroup problem, we used those (tighter) bounds.

In Table 1, the WJ methods are presented first, in increasing order of solution quality: 'worst', average and 'best'. LCW follows. All appropriate FK methods for that particular group size are then presented. Following, these solutions are the two bounds, LW and the appropriate FK. The ratio of these two bounds is then provided, as a means of determining which one is tighter. The next columns display the percentage improvement in solution quality gained by using LCW in place of each of the other methods. The last column displays the ratio of the LCW solution to the LC bound, providing a worst case measure for the difference between the solution provided by the heuristic, and the optimal solution.

7. SOLUTION TIMES

Table 3 provides solution times for each method. Values shown represent times to reach solutions for all sets for a given problem. The programs were coded in C and run on a nondedicated dual processor Sun SparcStation-20.

8. RESULTS

In viewing Table 2, the following may be observed.

- The best performing method is LCW; in evaluating average results over sets, LCW outperforms WJ and the FK methods in every case examined.
- On average, LCW performed approximately 26% better than WJ Worst, 13% better than a random approach, 2% better than WJ Best, over 4% better than Feo–Khellaf P1 and P2, over 2% better than our optimal assignment variation of P2, and over 1% better than Feo–Khellaf P4.
- Overall the Lakshminarayanan–Weitz (LW) bound was slightly tighter than the Feo–Khellaf bound. LCW performed at 75% of the bound for smaller problems to 66% of the bound for the larger problems.

With respect to computational resources, Table 3 indicates that:

- Method WJ requires the most resources, and as expected, solution times increase rapidly with problem size.
- Our optimal assignment variation of method P2 brings with it no computational penalty compared with the original 'random' approach.
- Method P4 takes 2–3 times as long as method P1.
- LCW takes the least amount of time. One run of LCW takes from 0.3% to 17% of the time required for P4, and from 0.3% to 46% of the time for methods P1 and P2. Typically these percentages increased with problem size. (The smaller problems here are comparable to those studied by Feo and Khellaf.)

It should be noted that for each problem LCW solutions provided here are averages of 100 runs, while LCW solution times provided are the average times for one run. The results indicate that over the long term (i.e. for multiple problems), LCW outperforms FK, and requires fewer resources. In order to determine the likelihood of a single LCW run outperforming FK, we calculated the standard deviations of the solutions provided by the 100 LCW runs for each set. (Imagine a standard deviation next to each mean LCW value given in Table 1.) For the smaller problems these standard deviations are in the range 0.01–0.02; for the larger problems, 0.03–0.04. For the larger problems ($N = 60$, 120), the means and standard deviations indicate a probability of virtually 100% that any single LCW solution will be better than that provided by the best FK solution. For the ten and 12 student problems this was the case for 59 of the 96 sets. For 19 of the remaining 37 sets, the data indicate that doing several runs of LCW and choosing the best of these solutions will yield a result better than FK 99% of the time. For 18 sets, the best FK solution outperformed LCW—in all cases only slightly; 11 of these sets were in the 12 student, four group problem where FK seemed to do disproportionally well.

9. CONCLUSIONS

This work addresses different approaches for solving a mathematical problem which spans a range of practical application. Weitz and Lakshminarayanan [3] showed that of the switching methods they investigated, their adaptation of a technique proposed by Lotfi and Cerveny [6] performed best, with no computational penalty. Feo and Khellaf [1] developed a set of graph theoretic methods, and in a series of trials on small problems showed that one of them (here, P4) performed within a few percentage points of simulated annealing and N-step lookahead. They concluded that P4 was superior in that the computational resources it required were significantly less than those for the other methods. The major conclusion of this work is that method LCW outperforms the graph theoretic approaches of Feo and Khellaf, including method P4. Further, method LCW is less computationally intensive. LCW has two other significant advantages: (1) compared with P4 it is a general method, not limited to group sizes of a power of two, and (2) relative to any of the graph theoretic approaches, it is simple to understand, implement, and explain to nontechnical personnel.

In order to generalize these results, additional research would have to be performed on a variety of datasets. At the moment, however, it would appear that method LCW is the approach of choice.

REFERENCES

1. Feo, T. and Khellaf, M., A class of bounded approximation algorithms for graph partitioning. *Networks*, 1990, **20**, 181–195.
2. Lakshminarayanan, S. and Weitz, R., Creating Groups Of Maximum Diversity: Contrasting Math Programming and Heuristic Approaches. Working paper, Stillman School of Business, Seton Hall University, South Orange, NJ 07079, 1994/revised January 1996.
3. Weitz, R. and Lakshminarayanan, S., An Empirical Comparison of Heuristic Methods for Creating Maximally Diverse Groups. Working paper, Stillman School of Business, Seton Hall University, South Orange, 1996.
4. Lawler, E. L., The quadratic assignment problem. *Management Science*, 1963, **9**, 586–599.
5. Weitz, R. R. and Jelassi, M. T., Assigning students to groups: a multi-criteria decision support system approach. *Decision Sciences*, 1992, **23**, 746–757.
6. Lotfi, V. and Cerveny, R., A final exam scheduling package. *Journal of the Operational Research Society*, 1991, **42**, 205–216.
7. Feo, T., Goldschmidt, O. and Khellaf, M., One-half approximation algorithms for the k-partition problem. *Operations Research*, 1992, **40 (suppl)**, S170–S173.
8. Weitz, R. and Lakshminarayanan, S., On a heuristic for the final exam scheduling problem. *Journal of the Operational Research Society*, 1996, **47**, 599–600.
9. Arani, T. and Lotfi, V., A three phased approach to final exam scheduling. *IIE Transactions*, 1989, **21**, 86–96.
10. Kuo, C.-C., Glover, F. and Dhir, K. S., Analyzing and modeling the maximum diversity problem by zero-one programming. *Decision Sciences*, 1993, **24**, 1171–1185.
11. Mingers, J. and O'Brien, F. A., Creating student groups with similar characteristics: a heuristic approach. *Omega*, 1995, **23**, 313–332.
12. O'Brien, F. A. and Mingers, J., The Equitable Partitioning Problem: A Heuristic Algorithm Applied to the Allocation of University Student Accommodation. Warwick Business School, Research Paper No. 187, June 1995.
13. Papadimitriou, C. H. and Steiglitz, K., *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Englewood Cliffs, NJ, 1982.
14. Johnson, D. S. and McGeoch, C. C., *Network Flows and Matching: First DIMACS Implementation Challenge, DIMACS Series in Discrete Mathematics and Theoretical Computer Science* **12**, American Mathematical Society, Providence, R.I., 1993. (The program is available via anonymous ftp to dimacs.rutgers.edu and is in: /pub/netflow/matching/weighted/solver-1.)
15. Carpaneto, G. and Toth, P., *ACM Transactions on Mathematical Software*, Vol. 6, 1980, pp. 104–111, (The program, ASSCT: a Fortran subroutine for solving the square assignment problem, is available via the internet at http://math.nist.gov/cgi-bin/gams-serve/list-module-components/TOMS/548/8587.html).

ADDRESS FOR CORRESPONDENCE: *Rob R Weitz, Stillman School of Business, Seton Hall University, South Orange, NJ 07079, USA.*