# A Novel Graph-Based Algorithm for Solving the Right Partner Problem

Youssef Harrath
*Department of Computer Science*
*College of Information Technology*
*University of Bahrain*
Kingdom of Bahrain
yharrath@uob.edu.bh

Abdul Fattah Salman
*Department of Computer Science*
*College of Information Technology*
*University of Bahrain*
Kingdom of Bahrain
amsalman@uob.edu.bh

Abdulla Alqaddoumi
*Department of Computer Science*
*College of Information Technology*
*University of Bahrain*
Kingdom of Bahrain
aqaddumi@uob.edu.bh

***Abstract*—In this research, the right partner problem is investigated and a new polynomial graph-based algorithm to solve it is proposed. The right partner problem belongs to the category of well-known assignment and matching problems. In the right partner problem, groups of persons are formed within one set, while in the assignment and matching problems, elements from two different sets are linked together. The authors propose modeling the problem using graphs with special weights on edges and nodes. In this paper, the authors applied and illustrated the developed algorithm to solve a real life instance of the problem by creating pairs from a given set of $n$ students. The solution is based on student choices of like and dislike relationships with other students and aims to maximize the degree of overall satisfaction of students' preferences.**

***Index Terms*—Right Partner, Matching Problem, Assignment Problem, Graph Theory, Grouping Problem, Spanning Tree**

## I. Introduction

This research aims at modeling and developing an efficient solution for the right partner problem based on one or more criteria. This problem belongs to the same category of the assignment and matching problems that are encountered in many real-life applications such as selecting roommates, husband/wife, machines/tasks, graduates/job vacancies, etc. In both the assignment and matching problems, elements from different sets are matched based on specific criteria. The right partner problem is another category of assignment problem where the elements to be grouped belong to one set. In the problem under investigation, the authors tackle a very common case in education where students are grouped in teams to do various assignments, projects, etc. Also, the team size is fixed to two since this is a very common practice in educational environment. To the best of our knowledge, this kind of problems is not studied before.

In this paper, a new graph-based approach to form pairs of students based on certain criterion is proposed. The students are asked to enlist their choices of students they prefer to work with (likes) and students they would like to avoid working with (dislikes). The proposed approach pairs the students in a way that satisfies their wishes as much as possible.

The right partner problem is defined as follows: given a list of $n$ students and each student provides two disjoint lists of his/her likes and dislikes among other students. The size of these lists are defined as parameters $l$ and $d$ respectively. The outcome of the proposed approach is $n/2$ pairs that maximize the degree of overall satisfaction of students' preferences. This problem can be modeled using graphs and solved using algorithms inspired from well-known techniques of the assignment and matching problems.

The rest of the paper is organized as follows. In section II, a literature review of similar work is presented and analyzed. Section III describes the proposed representation of the problem using graphs, presents the algorithm for its solution, and provides the complexity analysis. A deep discussion and analysis of the representation of the problem and the proposed algorithm are included in section IV

## II. Literature Review

Pairing students has its roots in both the assignment and matching problems. The assignment problem is a problem that tries to assign members from one group to members of another group. Examples of the assignment problem include assigning job applicants to companies, teachers to classes, and scheduling in parallel machines to name a few [1] and [2]. The assignment problem tries to make assignments in a way that maximizes the profit or benefit of such assignments. The Hungarian Algorithm solves the assignment problem in polynomial time [3]. The matching problem pairs members of two different groups together. A few examples of the matching problem are matching intern doctors to hospitals, marriage, and college admission [1] and [2].

The task of partitioning a large set of items into a collection of mutually disjoint subsets was tackled by [4]. The goal was to optimize a given objective (cost, penalty) while achieving the minimum number of subsets (groups). A generic selection hyper-heuristic framework containing a fixed set of reusable grouping low level heuristics and an unconventional move acceptance mechanism was proposed and proved to be efficient to solve the grouping problem. Reinforcement learning based local search (RLS) is used by [5] to solve the grouping problem as well. Job grouping problem is a special case of grouping problem where group size is variable. Elhag and Ozcan [8] used constraint programming to solve the assembly

978-1-5386-9207-3/18/$31.00 ©2018 IEEE

of electronic components on printed circuit boards, a practical industrial application of the job grouping problem.

The problem of stable matching and stable roommates in the presence of ties and incomplete lists was studied in [6] and [7]. The input of the indicated problem consists of two sets of different type of elements not mainly of the same size where each element in one set has to rank some elements of the other set. The target is to find the most stable matching.

In contrast to the reviewed research work, the solution to the right partner problem has fixed number of groups of similar size that satisfy certain criterion. Additionally, the members of the groups all belong to one set, while the matching and assignment problems match elements from different disjoint sets. To the best of our knowledge, no related work investigated the right partner problem as defined in this paper.

## III. Methodology

In this section, we discuss a real-life instance of the right partner problem for creating student teams based on their choices of possible partners. We represent the right partner problem as a weighted undirected graph and propose an algorithm to solve it.

### A. Problem Representation

The students and their likes and dislikes relationships are summarized and represented in a graph $G = (V, E)$, where, $V$ is the set of vertices representing the $n$ students and $E$ is the set of edges defining the relationships among the students as stated in two sets of likes and dislikes. To complete the graph, virtual edges are added to define neutral relationships for students having no likes or dislikes. This results in $n(n-1)/2$ edges in $E$. The likes and dislikes relations among the students is given numerically.

Four different options to represent the weights $w_{ij}$ of edges $(i, j) \in E$ are proposed as follows:

- In option 1, a weight of 2 is given for likes, a weight of -1 is given for dislikes, and a weight of 0 is given for neutral. This option emphasizes the priority of likes over the dislikes.
- In option 2, a weight of 1 is given for likes, a weight of -2 is given for dislikes, and a weight of 0 is given for neutral. In contrast to option 1, this option gives priority to dislikes.
- In option 3, a weight of 2 is given for likes, a weight of -2 is given for dislikes, and a weight of 1 is given for neutral. In this option the absolute weights of likes and dislikes are the same.
- In option 4, a weight of 1 is given for likes, a weight of -1 is given for dislikes, and a weight of 0 is given for neutral. Similar to option 3, this option has equal absolute values of likes and dislikes weights.

Normally, the problem is represented as a complete directed graph with $n(n-1)$ edges. Each directed edge $(i, j)$ is labeled with a weight $w_{ij}$ that defines the relationship between students $i$ and $j$ as described in the above-mentioned options. To simplify the problem graphical representation, we merged every pair of directed edges $(i, j)$ and $(j, i)$ into one undirected edge $(i, j)$ that has the weight equal to the sum of the weights $w_{ij} + w_{ji}$ as shown in Figure 1. Then, a set $W$ can be defined as $W = \{w_{ij}\}$ for every $1 \leq i, j \leq n$ such that $i \neq j$ and $w_{ij} = w_{ji}$. This step reduces the number of edges in the graph into $n(n-1)/2$ that results in reducing the complexity of any algorithm based on the graph.

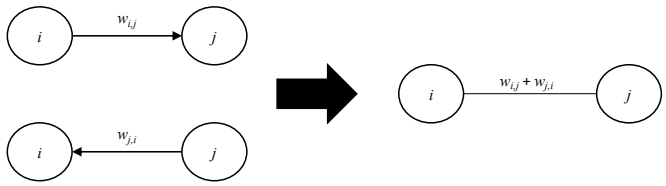


Fig. 1. Conversion of directed edges into undirected edges in the graph.

Based on the above discussion, option 4 of the proposed weights is discarded because it produces ambiguous situations such as when a student $i$ likes to be paired with a student $j$ and student $j$ dislikes to be paired with student $i$. This case produces a total weight of 0 for the edge $(i, j)$. Such weight of 0 can be obtained from bidirectional neutral relationships between two students.

In additional to the set $W$ of weights of edges, we calculate two other sets on graph nodes: $L$ of likes and $D$ of dislikes. The set $L$ can be defined as: $L = \{l_i\}$ for every node $1 \leq i \leq n$, where $l_i$ defines the number of students preferring to be paired with student $i$. The set $D$ can be defined as: $D = \{d_i\}$ for every node $1 \leq i \leq n$, where $d_i$ defines the number of students that prefer to not be paired with student $i$. So, the problem can be completely described by the complete undirected graph $G = (V, E, W, L, D)$.

### B. Problem Illustrating Example

To illustrate the proposed problem representation, we have a set of $n = 6$ students denoted by $A, B, C, D, E$, and $F$, number of likes $l = 3$, and number of dislikes $d = 1$. The details of every student set of like and dislike relationships are shown in table I.

TABLE I
TABLE OF LIKE AND DISLIKE RELATIONSHIPS

| Students | Likes | Dislikes |
|---|---|---|
| $A$ | $B, C, F$ | $E$ |
| $B$ | $A, C, E$ | $D$ |
| $C$ | $A, E, F$ | $D$ |
| $D$ | $A, B, C$ | $E$ |
| $E$ | $A, B, F$ | $D$ |
| $F$ | $B, D, E$ | $C$ |

The column labeled Neutral in Table II is deduced based on the like and dislike sets. While the numerical values $l_i$ are calculated as the number of members that like to work with a given member. The numerical values $d_i$ are calculated as the number of members that dislike to work with a given member.

TABLE II
TABLE OF CALCULATED NEUTRAL, LIKES AND DISLIKES RELATIONSHIPS

| Students | Neutral | $l_i$ | $d_i$ |
|---|---|---|---|
| $A$ | $D$ | 4 | 0 |
| $B$ | $F$ | 4 | 0 |
| $C$ | $B$ | 3 | 1 |
| $D$ | $F$ | 1 | 3 |
| $E$ | $C$ | 3 | 2 |
| $F$ | $A$ | 3 | 0 |

This problem instance is transformed into the graph illustrated in Figure 2. In this figure, the edge weight is calculated using option 3 of the weighting system. For example, $w_{AB} = 4$ since both $A$ and $B$ students like to be paired together, and $w_{DE} = -4$ since both $D$ and $E$ students dislike to be paired together. The pair of numbers on node $D$, $l_D = 1$ and $d_D = 3$ are obtained because only student $F$ likes to work with $D$ and students $B, C$, and $E$ do not prefer to be paired with $D$.

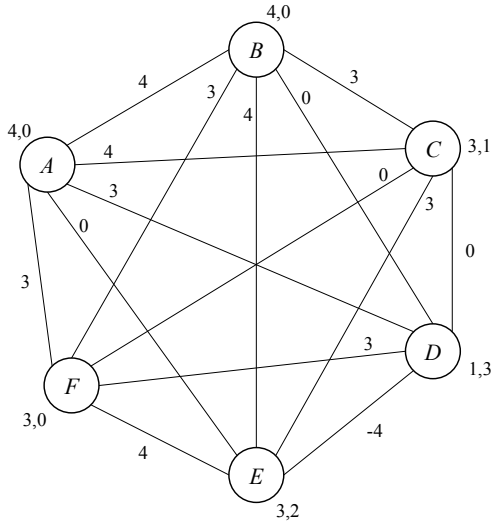


Fig. 2. Graph representation of the illustrating example.

### C. Proposed Algorithm

In this section we present the proposed algorithm to solve the right partner problem.

The algorithm takes as input a complete weighted graph $G$ discussed earlier in section III-A. The algorithm starts by making a copy $G^{'}$ of the original graph $G$. The algorithm produces a solution as a set $S$ of $n/2$ edges which is initially empty. The algorithm performs $n/2$ iterations during each an edge $e = (i, j)$ is carefully selected and added to $S$ as following:

- A set $V_1$ is formed by including all nodes $i$ that have maximum dislikes $d_i$.
- In case $V_1$ contains more than one node, a new set $V_2$ is formed to include all the nodes in $V_1$ that have the minimum likes $l_i$.

**Algorithm 1** Proposed algorithym for right partner problem

1: Input: $G^{'}(E^{'}, V^{'}, W^{'}, L^{'}, D^{'}) = G(E, V, W, L, D)$
2: Output: $S \subset E$ of $n/2$ disjoint edges with maximum total edge weight.
3: $S = \varnothing$
4: **for** $k = 1, ..., n/2$ **do**
5: Find $V_1$, set of nodes $i \in V^{'}$ with max $d_i$
6: Find $V_2$, set of nodes $i \in V_1$ with min $l_i$
7: Find $E_1 = \{(i, j) \in E^{'}, i \in V_2 or j \in V_2$ with max $w_{ij}\}$.
8: Select $(i, j) \in E_1 \mid i \in V_2$ and $j$ satisfies max $d_j$ followed by min $l_j$
9: $S = S \cup$ selected $(i, j)$
10: $V^{'} = V^{'} \setminus \{i, j\}$
11: $E^{'} = E^{'} \setminus \{e \in E^{'}$ incident to $i$ or $j\}$
12: Update $L^{'}$ and $D^{'}$ accordingly
13: **end for**

- All the edges incident to a node in $V_2$ and having the maximum weight are included in the set $E_1$.
- An edge from $E_1$ incident to a node in $V_2$ having the maximum dislike and minimum like is selected and added to the solution set $S$. If more than one edge satisfy the just indicated conditions, and edge is randomly selected.
- The graph $G^{'}$ is updated by removing the nodes $i$ and $j$ and the corresponding incident edges.
- The sets $L^{'}$ and $D^{'}$ of likes and dislikes are correspondingly updated.

### D. Illustration of Proposed Algorithm

In this section we illustrate the steps of the proposed algorithm to solve the problem presented in the graph in Figure 2.

In the first iteration of the algorithm, $V_2 = V_1 = \{D\}$ because $D$ has the maximum dislike weight of 3. After that, $E_1 = \{(A, D), (D, F)\}$ because both edges have the maximum weight of 3. Then, the edge $(D, F)$ is selected because the node $F$ has a smaller like weight than node $A$, although they have the same dislike weight which is checked first. The edge $(D, F)$ is inserted in the solution set $S$. After eliminating the edge $(D, F)$ and its incident nodes, the graph becomes as shown in figure 3.

In the second iteration of the algorithm, $V_2 = V_1 = \{E\}$ because $E$ has the maximum dislike weight of 1. After that, $E_1 = \{(B, E)\}$ because $(B, E)$ has the maximum weight of 4, which is selected. The edge $(B, E)$ is inserted in the solution set $S$. After eliminating the edge $(B, E)$ and its incident nodes, the graph becomes as shown in figure 4.

In the third iteration of the algorithm, the graph contains only the edge $(A, C)$ which will be selected and inserted in the solution set $S$ and the algorithm terminates.

The Solution set $S = \{(A, C), (B, E), (D, F)\}$ with a resulting weight of 11= 4 + 4 + 3.

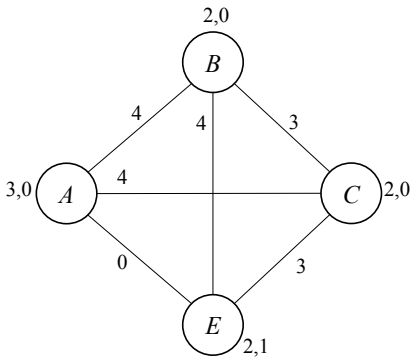


Fig. 3. Graph representation after the first iteration of the proposed algorithm.

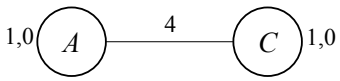


Fig. 4. Graph representation after the second iteration of the proposed algorithm.

### *E. Complexity of the Proposed Algorithm*

The complexity of the proposed algorithm is $O(n^3)$. In fact, the complexity of one iteration of the loop in step 4 is quadratic as the search in steps 5 and 6 is done in $O(n)$, while the search in steps 7 and 8 is done in $O(n^2)$ since there are $n(n-1)/2$ edges in $G$. Updating the graph by removing whether nodes or edges has a linear complexity $O(n)$. So, the overall complexity is $O(n^3)$ because the for loop iterates $(n-1)/2$ times.

## IV. Discussion and Analysis

To simplify the solution of the right partner problem, relationships among team members can be obviously limited to three types: likes, dislikes and neutral. Future studies can go further by considering more levels in each type of relationship such as strongly like/dislike, weakly like/dislike, etc. In addition to that, a relationship can be represented whether numerically or textually. Numeric values are more suitable for computational reasons since optimization problems mainly deal with minimization or maximization of numerical values. The solution of the studied problem, mainly is based on the applied edge weights. So, selecting the proper weights may affect the performance and accuracy of any suggested algorithm. Theoretically, the weights can be any three numeric values. However, using positive and negative values is more natural to represent such relationships. The simplest candidate values are +1, -1, and 0. But, these values may cause ambiguities. The authors suggested to include also +2 and -2 to avoid ambiguities as shown in section III-A.

The proposed algorithm is developed in a generic way and does not depend on the type of weighting system. In fact, most probably different optimal solutions are produced based on the options of weights. The suitable option of weights to use depends on the problem context. For example, if the target is to have a solution that maximizes the number of groups of compatible members (like, like), then option 1 or option 3 can be applied. However, if the aim is to minimize the number of groups of conflicting members (dislike, dislike), then option 2 is preferred.

The right partner problem can be modeled using graphs or mathematical equations. The authors opted to use graph-based representation of the problem following the majority of solutions to similar category of problems such as assignment and matching problems. The proposed algorithm is inspired from Prim's optimal spanning tree algorithm [9]. Our representation of the considered problem deals with three different weights: edge weights that represent the direct relationship between two persons, and node weights reflecting how much a member is liked or disliked by others. Prim's algorithm starts from any node and breaks the set of nodes into visited and non-visited nodes. It constructs iteratively an optimal solution by selecting an edge with the smallest weight that connects a visited node to a non-visited node and update the two sets accordingly. The essence of the proposed algorithm is to iteratively construct the solution set by selecting a certain edge each time. This is done by identifying a special set of nodes (persons) who other persons would prefer to avoid working with (nodes with maximum dislike weight followed by minimum like weight). After that, the edge with the maximum weight incident to one of the nodes in the previously identified set is selected and added to the solution set.

The advantages of the proposed algorithm are:

- Producing a solution is based on applying all input parameters of the problem such as: edge weights and nodes' like and dislike weights in a specific order, which we believe leads to very efficient solution.
- The complexity is polynomial ($O(n^3)$). This allows to use the proposed algorithm to solve large-sized real-life right partner problems in a short time.

## V. Conclusion and Future Work

This paper tackles a real life optimization problem in the educational environment where instructors assign projects, assignments, tasks to students. Sometimes, students need to work in pairs formed by the students themselves or enforced by the instructors. This way of forming the teams frequently creates conflicts among team members. The authors propose an algorithm to automate this process by taking into consideration student preferences in terms of like and dislike relationships. The algorithm was designed in a way to create the maximum number of teams with homogeneous members which helps in solving the assigned tasks in a more efficient way. The proposed solution is based on using graphs because of their advantages such as visualization, pool of available graph-based algorithms, simplicity of implementation, etc.

The solution is limited to forming teams having only two members. In reality, teamwork may require more members. So, the proposed algorithm can serve a core for designing more realistic algorithms where the size of the teams is a parameter. As a future work, the authors suggest to investigate

the efficiency of algorithms with different order of problem parameters. In addition, modeling and solving the problem using linear programming techniques would be a promising option.

## REFERENCES


[1] R. K. Ahyja and J. B. Orlin and T. L. Magnanti, Network flows: theory, algorithms, and applications, Prentice-Hall, 1993, pp.461–509.

[2] A. E Roth and M. Sotomayor, "Two-sided matching," Handbook of game theory with economic applications, vol. 1, pp. 485-541, 1992.

[3] H. W. Kuhn, "The Hungarian method for the assignment problem," Naval research logistics quarterly, vol. 2(1-2), pp. 83–97, 1955.

[4] A. Elhag and E. Ozcan, "A grouping hyper-heuristic framework: Application on graph colouring," Expert Systems with Applications, vol. 42, pp. 5491-5507, 2015.

[5] Y. Zhou, J. Hao, B. Duval, "Reinforcement learning based local search for grouping problems: A case study on graph coloring," Expert Systems with Applications, vol. 64, pp. 412–422, 2016

[6] A. Deeksha and G. Sushmita and R. Sanjukta and S. Saket and Z. Meirav, "Parameterized algorithms for stable matching with ties and incomplete lists," Theoretical Computer Science, vol. 723, pp. 1–10, 2018.

[7] D. Gale and M. Sotomayor , "Some remarks on the stable matching problem," Discrete Applied Mathematics, vol. 11(3), pp. 223–232, 1985.

[8] T. Knuutila and M. Puranen, and M. Johnsson, and O. Nevalainen, "Three perspectives for solving the job grouping problem," International Journal of Production Research, vol. 39(18), pp. 4261–4280, 2001.

[9] R. C. Prim, "Shortest connection networks And some generalizations," Bell System Technical Journal, vol. 36(6), pp. 1389–1401, 1957.