Available online at www.sciencedirect.com
ScienceDirect
European Journal of Operational Research 176 (2007) 774–793
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

Discrete Optimization

Assignment problems: A golden anniversary survey

David W. Pentico

A.J. Palumbo School of Business, John F. Donahue Graduate School of Business, Duquesne University, Pittsburgh, PA 15282-0180, USA

Received 19 January 2005; accepted 16 September 2005
Available online 18 November 2005

Abstract

Having reached the 50th (golden) anniversary of the publication of Kuhn's seminal article on the solution of the classic assignment problem, it seems useful to take a look at the variety of models to which it has given birth. This paper is a limited survey of what appear to be the most useful of the variations of the assignment problem that have appeared in the literature over the past 50 years. The intention here is not to identify every such paper (of which there have been hundreds) nor to identify the best solution procedure for each variation. Rather, the intention is to identify what these variations are and what they are called so as to make it easier for a researcher trying to develop some variation of the assignment problem for a particular application to find the relevant literature.

Keywords: Assignment problems; Integer programming

1. Introduction

Although the name "assignment problem" seems to have first appeared in a 1952 paper by Votaw and Orden [69], what is generally recognized to be the beginning of the development of practical solution methods for and variations on the classic assignment problem (hereafter referred to as the AP) was the publication in 1955 of Kuhn's article on the Hungarian method for its solution [38]. In recognition of the significance of this article, *Naval Research Logistics* reprinted it in honor of its 50th anniversary [39], along with a statement by Kuhn about the development of the Hungarian method [40] and a tribute by Frank [25] explaining the relationship between Kuhn's technique and its antecedents in the papers by two Hungarian mathematicians.

Over the past 50 years, many variations on the classic AP have been proposed, a fact that becomes immediately obvious if the key words "assignment problem" are entered into the search engine for the research database ABI/INFORM. Among the references found in such a search were ones to the bottleneck AP, the generalized AP, the

E-mail address: pentico@duq.edu

0377-2217/$ - see front matter
doi:10.1016/j.ejor.2005.09.014

quadratic AP, the semi-AP, and a variety of others. As might be expected, there were also references to papers that combined two or more of these basic variations.

Recognizing that there are researchers who have an interest in some variation on the classic AP, but do not know what the standard name of that variation is, thus making it more difficult to search for previous work on the topic, this paper is intended to provide a reasonably comprehensive survey of the different variations on the assignment problem that have appeared in the literature. The purpose here is not to identify the best solution procedure for each of these "adjective" APs. Papers describing new solution procedures appear constantly in the literature and are of interest primarily to those whose interest is in designing improved methods for these models' solution. A fairly recent survey of solution procedures, particularly those for the classic AP, may be found in Dell'Amico and Martello [17]. Rather, our purpose is to simply make researchers and analysts aware of what these variations are so that they can more easily determine which variation most closely matches their current problem. Thus, the remainder of this paper will be a series of brief descriptions of the various "adjective" APs identified, along with one or more references to each and, where possible, examples of the types of problems that they have been developed to address. In addition, to make it easier to read this paper, the notation used in the cited sources has, where necessary, been changed to try to make it as consistent as possible with the notation to be used in the presentation of the original or classic AP in the next section.

Assignment problems involve optimally matching the elements of two or more sets, where the dimension of the problem refers to the number of sets of elements to be matched. When there are only two sets, as will be the case for most of the variations we will consider, they may be referred to as "tasks" and "agents". Thus, for example, "tasks" may be jobs to be done and "agents" the people or machines that can do them. In its original version, the assignment problem involved assigning each task to a different agent, with each agent being assigned at most one task (a one-to-one assignment). While only two of the models to be discussed involve assigning multiple agents to a task [15,3], some of the models do assign multiple tasks to the same agent (a one-to-many assignment). The models to be discussed first, however, assign no more than one task to any given agent.

2. Models with at most one task per agent

2.1. The classic assignment problem

The original version of the assignment problem is discussed in almost every textbook for an introductory course in either management science/operations research or production and operations management. As usually described, the problem is to find a one-to-one matching between n tasks and n agents, the objective being to minimize the total cost of the assignments. Classic examples involve such situations as assigning jobs to machines, jobs to workers, or workers to machines.

The mathematical model for the classic assignment problem may be given as:

$$\text{Minimize} \quad \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$\text{Subject to:} \quad \sum_{i=1}^{n} x_{ij} = 1 \quad j = 1,\ldots,n,$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad i = 1,\ldots,n,$$

$$x_{ij} = 0 \text{ or } 1,$$

where $x_{ij} = 1$ if agent i is assigned to task j, 0 if not, and c_{ij} = the cost of assigning agent i to task j. The first set of constraints ensures that every task is assigned to only one agent and the second set of constraints ensures that every agent is assigned to a task. The basic mathematical structure of the problem makes the constraint that x_{ij} be binary unnecessary since there will automatically be an optimal linear programming solution in which all the x_{ij}s are either 0 or 1.

Minor modifications of the basic problem structure are easily handled with the standard solution procedures. A problem in which the objective

function is to be maximized can be easily converted into a minimization problem by either (a) multiplying all of the c_{ij}s by -1, or (b) replacing each c_{ij} by $c_{\max} - c_{ij}$, where $c_{\max}$ is the maximum of the c_{ij}s, thus converting the problem to one of minimizing "regret". A problem that is not balanced (i.e., one for which the numbers of tasks and agents differ) can be easily converted into a balanced problem by adding a sufficient number of "dummy" tasks or agents (whichever is in shorter supply) with costs of 0. One may also use non-zero costs for assignments using dummy tasks or agents to reflect differences based on which agents or tasks are not assigned.

It is worth noting that the classic AP is mathematically identical to the *weighted bipartite matching* problem from graph theory, so that results from that problem have been used in constructing efficient solution procedures for the classic AP.

2.2. The classic assignment problem recognizing agent qualification

In their work on a particular version of the assignment problem with side constraints (Section 2.13), Caron et al. [11] use a mathematical model for a variation of the classic AP in which there are m agents and n tasks, not every agent is qualified to do every task, and the objective is utility maximization:

$$\begin{aligned}
\text{Maximize} \quad & \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \\
\text{Subject to:} \quad & \sum_{i=1}^{m} q_{ij}x_{ij} \leqslant 1 \quad j = 1,\ldots,n, \\
& \sum_{j=1}^{n} q_{ij}x_{ij} \leqslant 1 \quad i = 1,\ldots,m, \\
& x_{ij} = 0 \text{ or } 1,
\end{aligned}$$

where $x_{ij} = 1$ if agent i is assigned to task j, 0 if not, $q_{ij} = 1$ if agent i is qualified to perform task j, 0 if not, and $c_{ij} =$ the utility of assigning agent i to task j (with $c_{ij} = 0$ if $q_{ij} = 0$). The first set of constraints ensures that no more than one qualified agent is assigned to any task and the second set of constraints ensures that no agent is assigned to more than one task. Note that even if m is greater than or equal to n it may not be possible to assign a qualified agent to every task or to give every agent a task for which it is qualified. Minimization problems can be handled using either of the methods suggested above for handling maximization problems for the standard classic AP.

2.3. The k-cardinality assignment problem

Dell'Amico and Martello [16] present a variation on the classic AP in which there are m agents and n tasks, but only k of the agents and tasks are to be assigned, where k is less than both m and n. The mathematical model for the k-cardinality assignment problem is:

$$\begin{aligned}
\text{Minimize} \quad & \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \\
\text{Subject to:} \quad & \sum_{i=1}^{m} x_{ij} \leqslant 1 \quad j = 1,\ldots,n, \\
& \sum_{j=1}^{n} x_{ij} \leqslant 1 \quad i = 1,\ldots,m, \\
& \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij} = k, \\
& x_{ij} = 0 \text{ or } 1,
\end{aligned}$$

where $x_{ij} = 1$ if agent i is assigned to task j, 0 if not, $c_{ij} =$ the cost of assigning agent i to task j, and k is the number of agent-task assignments that must be made.

Dell'Amico and Martello suggest as a potential application for this model the situation in which workers are to be assigned to machines, but only a subset of the workers and machines need to be assigned. They also suggest that it can be used to solve sub-problems in the problem of assigning time slots on a communications satellite being used to transmit information from m earth stations to n different earth stations. This problem is discussed in detail in Prins [56].

2.4. The bottleneck assignment problem

While the objective of the classic AP is to minimize or maximize the *sum* of the costs of the assignments of tasks to agents, the objective of

the bottleneck AP (BAP) is to minimize the *maximum* of the costs of the assignments. An example would be determining how to assign printing jobs to presses so as to minimize the time by which all the jobs would be completed. Examples given in [58] involve determining how to transport perishable goods from warehouses to markets without spoilage or military supplies from warehouses to command posts during an emergency. In either case, the objective is to minimize the time by which all the transfers have taken place.

The mathematical model for the BAP may be given as:

$$\text{Minimize } \max_{i,j}\{c_{ij}x_{ij}\} \text{ or Minimize } \max_{i,j}\{c_{ij}\,|\,x_{ij}=1\}$$

subject to the same constraints and definitions as in the classic AP. Unlike the classic AP, solving the linear programming version of this problem does not guarantee a binary solution, so that constraint is necessary.

Although they reference earlier work by Gross [33], the earliest discussion of the BAP found in the readily accessible published literature is in Ford and Fulkerson [22], in which they present a *maximin* version of the problem that involves assigning workers to jobs in such a way that the minimum efficiency of such assignments will be maximized. For example, quoting from [22], "the jobs might be those on an assembly line, and c_{ij} might represent the number of units per hour that man i can process if assigned job j. Then, for given assignment P, the rate of the assembly line is measured by the bottleneck $\min_i c_{i,P(i)}$, and thus we wish to maximize this..." (To help clarify this statement, P is a permutation of the indexes $1,2,\ldots,n$ which shows, in position i, the task j assigned to agent i. Thus, $c_{i,P(i)}$ is c_{ij} for the assignment of agent i to whatever task j is called for by permutation P.) This example provides the source of the name "bottleneck AP" for the *maximin* version of the AP problem, a designation which has been carried over to the minimax version.

2.5. The balanced assignment problem

While the basic BAP can be formulated to find the set of assignments that will either minimize the maximum value of an assignment or maximize the minimum value, neither approach considers the values of the other assignments. The *balanced* assignment problem, described in Martello et al. [47], attempts to recognize both objectives by minimizing the difference between the maximum and minimum assignment values. One example given is an American travel agency planning a program of trips to Europe with all the travelers, each of whom will take one of the trips, going and returning on the same pair of chartered flights; the objective is to minimize the difference between the dates of the going and return flights. Another example is choosing the suppliers for a multi-component product, the objective being to minimize the difference between the maximum and minimum expected failure times so that all the components will have to be replaced at around the same time. Duin and Volgenant [20] give additional examples based on (1) competitors cooperating on the construction of a network-tree for communication, with the objective of keeping the maximum and minimum costs of the construction and future maintenance of the individual incoming lines as close as possible, and (2) the assignment of patients with different degrees of severity of some "condition" to test groups for comparing different treatment approaches, with the objective that the maximum and minimum of the "average" conditions of the test groups should be as close as possible.

The model for the balanced assignment problem may be given as:

$$\text{Minimize} \quad \max_{i,j}\{c_{ij}|x_{ij}=1\} - \min_{i,j}\{c_{ij}|x_{ij}=1\}$$

subject to the same constraints and definitions as in the classic and bottleneck APs.

2.6. The minimum deviation assignment problem

Closely related to the balanced AP problem is the *minimum deviation* AP discussed by Gupta and Punnen [34] and by Duin and Volgenant [20]. The basic difference between the two models is that while the objective in the balanced AP is to minimize the difference between the maximum and minimum assignment costs, the objective in

the minimum deviation AP is to minimize the difference between the maximum and average assignment costs or, equivalently, to:

$$\text{Minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n}(\max_{p,q}\{c_{pq}x_{pq}\} - c_{ij})x_{ij}$$

or, if m and n differ,

$$\text{Minimize} \quad \min\{m,n\} \times \max_{p,q}\{c_{pq}x_{pq}\} - \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$$

subject to the usual classical AP constraint set.

Duin and Volgenant [20] give two examples of the use of the minimum deviation AP. In the first example a network is to be constructed by cutting a standard size edge-piece down to the sizes to be used for the individual edges. The objective is to minimize the waste from cutting. In the second example a project is to be completed by assigning the tasks to different machines, which will work on the tasks simultaneously, but are not available for assignment to the tasks of another project until all the tasks of the current project are completed. The objective is to minimize the idle time for the machines.

2.7. The lexicographic bottleneck problem

Related in concept to the balanced AP (Section 2.5), but carrying further the idea of recognizing the times or costs of assignments other than the longest or most costly one on which the BAP focuses, is the *lexicographic* bottleneck (or lex-bottleneck) assignment problem, discussed by Burkard and Rendl [8] and by Sokkalingam and Aneja [63]. Recognizing that there are generally many different sets of assignments that have the same minimum value for $\max\{c_{ij}x_{ij}\}$, the objective of the lexicographic bottleneck problem is to find the set of x_{ij}s meeting the constraints of the classic AP that will not only minimize the largest cost coefficient in the solution, but also minimize the second largest, the third largest, etc.

Notice that the objective here is quite different from the objectives of the balanced and minimum deviation APs discussed in the two preceding sections. In the lex-bottleneck AP the objective is to find the solution giving the smallest maximum cost, then the smallest next-to-maximum, etc. In both the balanced and minimum deviation APs we start with a small maximum cost, but then the smallest cost (in the balanced AP) or the second largest, etc., down through the smallest (minimum deviation AP) should be relatively *large* so that either the spread from top to bottom or the difference between the largest and the average costs are as small as possible.

2.8. The Σ_k-assignment problem

While the objective of the classic AP is to find a set of assignments that minimizes their sum and the objective of the bottleneck AP (BAP) is to find a set of assignments that minimizes the value of the largest one (and, therefore, trivially, its "sum"), in the Σ_k-assignment problem (Σ_k-AP), an algorithm for the solution of which is given in Grygiel [32], the objective is to find a set of assignments for which the sum of the k largest is minimized. Assuming that the numbers of agents and tasks are both n, the BAP and classic AP can then be viewed as special cases of the Σ_k-AP with $k = 1$ and $k = n$ respectively.

The Σ_k-AP differs from the k-cardinality AP discussed in Section 2.3, which also focuses on only k of the assignment values, in that in that problem only k out of the n possible assignments are actually made while in this problem all of the assignments are made but only the k largest are summed.

2.9. The semi-assignment problem

One of the key assumptions of the classic AP is that the tasks and the agents to which they are to be assigned are unique, which leads to a problem description matrix of the c_{ij}s in which every row is different from every other row and every column is different from every other column. It may be, however, that although all the agents are unique, some of the tasks are identical or vice versa. Kennington and Wang [36] list manpower planning, scheduling, capital budgeting and planning, and project planning as examples of areas in which this might be the case. From manpower planning, they give an example "in the area of Navy personnel

assignment [in which] a particular ship may require several radio operators with the same rank and skill level". The model for the situation in which there are m agents and n task groups ($n < m$), with d_j tasks in group j ($\sum_j d_j = m$), is:

$$\text{Minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$\text{Subject to:} \quad \sum_{i=1}^{m} x_{ij} = d_j \quad j = 1,\ldots,n,$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad i = 1,\ldots,m,$$

$$x_{ij} = 0 \text{ or } 1,$$

where $x_{ij} = 1$ if agent i is assigned to task group j, 0 if not, c_{ij} = the cost of assigning agent i to task group j, and, as noted above, d_j is the number of tasks in group j. As noted in Volgenant [66], taking advantage of the fact that there are identical columns or rows in the data matrix makes it possible to restructure the problem and the solution procedure so that very large problems of this type can be solved very quickly. (Note that if the data matrix for the problem has both identical rows *and* identical columns, the structure of the model is the same as that of a Transportation Problem.)

2.10. The categorized assignment problem

One of the key assumptions of all the models discussed above is that all of the tasks can be done simultaneously. In some cases, however, the jobs may be divided into categories or groups, with sequencing requirements either within each group or between groups. Punnen and Aneja [57] present generalizations and examples of both the AP and BAP under categorization, giving heuristic solution procedures for both. For both versions of the problem, the definition starts with the assumption that n jobs are to be assigned to n machines on a one-to-one basis. However, the jobs are partitioned into $r < n$ categories or sets: $S_1, S_2, \ldots, S_r$. Punnen and Aneja identify two cases or versions of the problem, depending on which type of sequencing or precedence relationship applies.

Case 1: If jobs in the same category must be processed in a particular sequence, but the different categories may be processed in parallel, then, using the earlier notation based on recognizing the solution of the problem as one of identifying a permutation that minimizes an objective function, the objective is:

$$\underset{P}{\text{Minimize}} \quad \max_k \sum_{j\in S_k} c_{P(j),j}$$

subject to the usual constraints. Notice that the objective here is to find a permutation P (an assignment of agents to tasks) that will minimize the maximum of the sums of the assignment values (usually times) across the m different sets. That is, the objective is to minimize the time at which the last set of tasks is finished if all sets start at the same time, where the completion time of a set is the sum of the times of its tasks. Thus, within each set we have the basic idea of a classic AP (minimize the sum), while across the sets we have the basic idea of a bottleneck AP (minimize the maximum). Seshan [60] gives as an example of this case "a machine shop having a number of similar (but not identical) machines. A number of turning jobs has to be processed on these machines, and each turning job requires more than one turning operation to be performed in a fixed sequence. The problem is to find an assignment of the operations to the machines, for which the total time for completing all the jobs is minimized".

Case 2: If jobs in the same category may be processed simultaneously, but the different categories must be processed in sequence, then the objective is:

$$\underset{P}{\text{Minimize}} \quad \sum_{k=1}^{r} \max_{j\in S_k}\{c_{P(j),j}\}$$

subject to the usual constraints. Notice that the objective here is to find a permutation P (an assignment of agents to tasks) that will minimize the sum of the maximum times across the m different sets. That is, since the sets must be done in sequence, but the tasks within each set may be done simultaneously, the objective is to find the assignments that will minimize the sum of the times for finishing all the sets when an individual set's finishing time is the largest of the times of its tasks. Thus, within each set we have the basic idea of a bottleneck AP (minimize the maximum), while

across the sets we have the basic idea of a classic AP (minimize the sum). Seshan [60] gives, as an example of this case, "an assembly line which can handle multiple products, e.g. radios of different models. The components required for the products are processed in a number of machines. The same types of components can be processed simultaneously but different types of components have to be processed in a particular sequence because of the assembly line requirements. The problem is to assign the components to the machines so that the time for processing all the components is minimized".

Note that a modeling limitation of both of these cases is the assumption that there will be a one-to-one matching of tasks and agents, i.e., that the solution will be a permutation P of the indexes. Recognizing that in both cases there are precedence relationships, either between tasks in a set (Case 1) or between the sets of tasks (Case 2), it should be obvious that, in many cases, an agent will be able to do more than one task. Aggarwal et al. [1] provide models and solution procedures that involve assigning the same agent to all of the tasks in a particular set (Case 1) or that allow the assignment of any agent to multiple tasks, one each from two or more sets (Case 2).

Calling it the *multiple bottleneck assignment problem*, Aneja and Punnen [2] discuss a variation on Case 1 for which the objective function is more like that for Case 2. The problem is the assignment of n operators to n machines that have been divided up into r independent flow lines in a production system. Each operator/machine combination has a particular output rate c_{ij} that depends on which operator i is assigned to machine j. Although the operations on flow line k have to be performed in a particular sequence, as in Case 1, the output of that line is determined, as discussed by Ford and Fulkerson [22] in Section 2.4, by the *minimum* output rate of any work station on that line, rather than by their sum. The problem of maximizing the overall output of the production system leads to the objective function:

$$\underset{P}{\text{Maximize}} \quad \sum_{k=1}^{r} \min_{j \in S_k} \{c_{P(j),j}\}.$$

2.11. Multi-criteria assignment problems

With the exception of the lexicographic bottleneck AP (Section 2.7), all the models considered so far, while possibly having different objectives, are alike in that each has a single objective. Even the lex-bottleneck AP involved only one type of objective, repeatedly applying the bottleneck criterion to identify from among the solutions to the BAP those that would minimize the second, third, etc. highest costs. In many cases, however, there are multiple decision criteria and it is important to find a solution that, in one way or another, recognizes all of them. There are two basic methods by which multiple criteria are recognized in decision models: (1) by combining them into a single criterion, and (2) by considering them separately and sequentially.

2.11.1. Combining multiple criteria into one

Geetha and Nair [27] simultaneously consider the objective functions of the classic AP and the bottleneck AP. Their example assumes that there are not only the costs c_{ij} of having the agents do the tasks (the sum of which is to be minimized—the classic AP), but also a "supervisory" cost that is based on the makespan or the maximum time to complete the longest task (the value of which is to be minimized—the bottleneck AP). Their objective function is:

$$\text{Minimize} \quad \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} + F \times \max\{t_{ij} | x_{ij} = 1\}$$

subject to the usual constraints, where c_{ij} is, as before, the cost of assigning agent i to task j, t_{ij} is the time required for agent i to complete task j, and F is the "supervisory" cost per period.

Scarelli and Narula [59], using the example of assigning referees to football matches in an Italian championship, develop a procedure for the situation where there are multiple independent criteria with weights assigned by the decision maker. Their method "allows the inclusion of the concept of indifference, preference and veto thresholds that must be taken into consideration in many of the multi-criteria models, especially in the presence of qualitative criteria", such as the compatibility/

incompatibility of an agent for a task and the priorities of each task and every worker.

Mehrez et al. [51] and Yuan et al. [72] consider what they call the *matching assignment problem*, which is basically the problem of matching applicants with jobs when both the applicants and the employers have their own preferences as to which applicant is to fill each job. An example is the National Resident Matching Program, which assigns graduates of medical schools to hospitals for internships. The authors compare the "fairness" of the stable assignment approach, which is a multi-stage approach driven by the preferences of the hospitals but allowing for rejection by the applicants, with the results of using a classical AP model in which the objective function coefficients, the sum of which is to be maximized, are obtained by multiplying terms based on the utilities for the assignments as expressed by both the applicants and the employers. The authors' analysis found that the multiplicative utility approach resulted in assignments that were not only more equitable in meeting the two sets of preferences (less biased toward the employers), but were also more efficient in matching each participant with his or her most preferred choice.

2.11.2. Considering multiple criteria sequentially

Preemptive goal programming is an extension of linear programming that considers multiple objectives separately and sequentially. Lee and Schniederjans [45] applied preemptive goal programming to the problem of determining how to reassign remedial education teachers from the schools at which they taught in the morning to the schools at which they were to teach during the afternoon. The objectives considered included the costs of travel between the schools (objective), the preferences (subjective) of the teachers for and against certain schools and of the school administrators for and against certain teachers, and the recommendations (subjective) of the teachers' supervisors as to who should be assigned where.

The concept of the lexicographic bottleneck AP model (Section 2.7) was extended by Volgenant [67] to consider the problem of finding solutions to certain graph theory problems, including APs, that optimize a first objective and then, from among those solutions, optimize a second objective, a third objective, etc. With r objectives to consider, Volgenant refers to this as the *r-lexicographic multi-objective problem* (r-LMOP). The objectives may be any combination selected from the set {min sum, max sum, min bottleneck, max bottleneck}. As an example of an r-LMOP assignment problem, Volgenant [67] suggests the problem of assigning "jobs to machines to minimize cost, delay, machine wear, error in accuracy, etc. in some priority order". Volgenant also states that this approach can be used to model and solve the problem of assigning agents to tasks while recognizing the seniority classes of the agents and the priorities of the tasks considered by Caron et al. [11] (Section 2.13).

2.12. The fractional assignment problem

Fractional programming is a particular type of non-linear programming in which the objective function to be optimized is the ratio of two other objective function expressions. Shigeno et al. [61] present a solution procedure for the *fractional assignment problem*, the model for which is:

$$\text{Minimize} \quad \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}}{\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij}}$$

subject to the usual constraints for the classic AP. Shigeno et al. allow for the possibility that not every agent is qualified to be assigned to every task (as in the classic AP recognizing agent qualification in Section 2.2). They also require the c_{ij}s to be integers and the d_{ij}s to be positive integers. A potential application is the maximization across all assignments of the value received per unit of time expended on the assignments, where c_{ij} is the negative of the value received from assigning agent i to task j and d_{ij} is the time agent i would take to perform task j.

2.13. The assignment problem with side constraints

In all the models discussed so far with the exception of the extension of the Categorized AP discussed in [1], the basic structure was the optimi-

zation of some objective function or functions subject to the two sets of constraints that each task is to be assigned to a single agent and each agent is to be assigned to at most one task. There may, however, also be side constraints, such as budgetary limitations, degree of technical training of personnel, the rank of personnel, or time restrictions, that limit the assignment of agents to tasks. The issue of whether an agent is qualified to be assigned to a particular task was addressed in the model for the classic assignment problem recognizing agent qualifications in Section 2.2. Mazzola and Neebe [49] present a general model for the assignment problem with side constraints that adds the following constraint or constraints to either the classic AP model in Section 2.1 or the classic AP recognizing agent qualifications model in Section 2.2:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} r_{ijk}x_{ijk} \leqslant b_k,$$

where r_{ijk} is the amount of resource k used if agent i is assigned to task j and b_k is the amount of resource k available. There may be as many such constraints as are necessary.

Foulds and Wilson [24] discuss a special type of this problem concerned with assigning tools to the slots in a tool carousel for a flexible manufacturing system. The constraints in this model recognize that (1) due to their different sizes, the different types of tools occupy different numbers of adjacent slots and (2) since there are zones in the carousel that may not be used, the sequences of adjacent slots into which the tools may be placed are of varying sizes. The objective is to position the tools in the slots so as to maximize the sum of adjacency ratings that reflect the desirability of locating specific pairs of tools next to one another in a specific order.

Building on the model for the assignment problem recognizing agent qualifications in Section 2.2, Caron et al. [11] and Volgenant [68] consider a variation of the AP with side constraints in which the constraints deal with (1) assigning agents only to tasks for which they are qualified, (2) recognizing seniority classes of the agents, which require that preference be given to assigning tasks to agents in higher seniority classes, and (3) recognizing priority classes for the tasks, which require that tasks with higher priorities be given preference when assigning agents. The objective is to maximize the utilities (or minimize the disutilities) of the assignments.

In Caron et al. [11] the seniority and priority constraints are recognized by modifying the objective function coefficients by incorporating appropriate weights for the seniority/priority classes to which the i and j indices refer and then solving the resulting classic AP. Volgenant [68] presents an alternative solution approach of lower complexity that involves solving a sequence of APs of increasing size. This approach can be easily modified to solve the bottleneck version of the problem.

Caron et al. [11] give an example involving the daily scheduling of nurses at a hospital. The nurses belong to either (1) the float team, who are under contract to the hospital and will be assigned to jobs, generally due to the absence of regular staff or an unexpected work overload, or, if no jobs are available, to training, or (2) the hospital's availability list. Nurses in the latter category will only be used if no qualified float-team nurse is available.

2.14. The quadratic assignment problem

The quadratic assignment problem (QAP) is most often discussed in relation to the problem of determining where to locate m facilities among n ($n \geqslant m$) given sites. With this setting in mind, the mathematical model may be given as:

$$\text{Minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{p=1}^{m}\sum_{q=1}^{n} c_{ijpq}x_{ij}x_{pq}$$

$$\text{Subject to:} \quad \sum_{i=1}^{m} x_{ij} \leqslant 1 \quad j = 1,\ldots,n,$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad i = 1,\ldots,m,$$

$$x_{ij} = 0 \text{ or } 1,$$

where $x_{ij} = 1$ if facility i is assigned to location j and 0 if it is not. The cost c_{ijpq} of the interaction

between facility i in location j and facility p in location q is determined as the product of two terms: (1) f_{ip}, the cost of moving items for one unit of distance from facility i to facility p, usually measured as the volume of traffic or "flow" between the two facilities, and (2) d_{jq}, the distance from j, the location of facility i, to q, the location of facility p. Hillier and Connors [35] discuss how to determine the values of the c_{ijpq} coefficients, depending on the relationships among the indexes and the more specific nature of the location problem. While the statement of the model may not even involve the definition of the x_{ij} decision variables, such as Drezner's formulation [19], which assumes that $m = n$ and states the problem as one of identifying the permutation P of $1, 2, \ldots, n$ that will minimize $\sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij}d_{P(i)P(j)}$ (using a combination of the notations used here and previously), the formulation given first makes clear the source of the name *quadratic* assignment problem.

While it is possible to set up and solve optimizations for small examples of the QAP, including one by Lawler [42] that transforms it into an equivalent, very large linear programming model that is, in essence, a classic AP with two sets of side constraints, the difficulty of optimizing for more realistically sized problems means that most of the solution procedures discussed in the literature are heuristics.

In an early survey by Burkard [7] of the QAP, applications discussed include determining where to put components on circuit boards, campus planning, and typewriter keyboard design. Section 1.2 of a monograph by Çela [14] gives references to additional applications, including ones in scheduling, parallel and distributed computing, statistical analysis, sports, chemistry, archeology, computer manufacturing, and transportation. Although they do not identify it as a QAP, Tsui and Chang [64] give an example of assigning the doors on the opposite sides of a shipping company's dock facility to incoming and outgoing trucks when the items in the incoming trucks are to be transported directly to the outgoing trucks and the objective is to minimize the total travel distance for the forklifts doing the transporting.

Determining assignments of departments to locations in a hospital with the objective of minimizing the *maximum* distance a doctor must travel to reach a patient leads to the *bottleneck* QAP [21].

2.15. The robust assignment problem

In the general field of decision theory there are four basic classifications of problems: (1) decision making under certainty, in which all parameter values are known; (2) decision making under risk, in which there are multiple sets of possibilities for the parameter values (scenarios is the term often used), with known probabilities for the different sets, (3) decision making under uncertainty, in which there are different scenarios but their probabilities are not known, and (4) fuzzy decision making, in which the possible scenarios are not all known. While there are considerable differences among the characteristics of the models discussed so far, and those remaining to be discussed, what they have in common is the assumption that all of the model's parameters are known with certainty. Kouvelis and Yu [37] discuss the formulation of and present a solution procedure for the assignment problem under uncertainty, which they call the *robust assignment problem*. The key concept behind this approach to solving the AP is the decision maker's recognition that, in Kouvelis and Yu's words, "for any potentially realizable scenario he/she has to live with the consequences on system performance of the decision made". Their choice of the term *robust* reflects the idea that the solution chosen remains close to optimal for any input data scenario to the model.

A key aspect of Kouvelis and Yu's approach to robust decision making is the choice of an appropriate *robustness criterion*, of which they identify three:

- The *absolute robust* decision is the set of x_{ij} values that minimizes the maximum total cost over all the possible parameter value scenarios. In the terminology of decision making under uncertainty, this is equivalent to the minimax cost decision. It is a very conservative approach as it is based on the anticipation that the worst might happen.

- The *robust deviation* decision is the set of x_{ij} values that gives the best worst-case deviation from optimality across all possible parameter value scenarios. This is equivalent to the minimax regret decision in the terminology of decision making under uncertainty. This approach recognizes, in Kouvelis and Yu's words, that "decisions that are made based on uncertain information are often evaluated ex post as if the actual scenario realization has been known in advance of the decision, [so the] decision maker is rightfully concerned not only with how a decision's performance varies with the various data scenario realizations, but also with how actual system performance under the decision compares with the optimal performance that could have been achieved if perfect information on the scenario realization had been available prior to the decision".
- The *relative robust* decision approach is based on the same "hindsight" concept as the robust deviation decision approach except that instead of focusing on the size of the deviation from optimality, it finds the solution that gives the best worst-case *percentage* deviation from optimality across all possible parameter value scenarios.

Kouvelis and Yu [37] give the model forms for the three versions of the robust assignment problem and present a branch-and-bound solution method that, in their evaluation study, performed well computationally for problems with up to 40 tasks and agents and 30 parameter value scenarios.

Under the heading Robust Layout Planning, Kouvelis and Yu [37] also give models for the quadratic assignment problem (QAP) under all three versions of robustness criteria just discussed.

3. Models with multiple tasks per agent

One type of problem that allows or requires assigning the same agent to more than one task, the multiple bottleneck assignment problem [2], was discussed in Section 2.10, when the categorized assignment problem was considered. In this section we will focus on other models with multiple tasks per agent, starting with different versions of the generalized assignment problem.

3.1. The generalized assignment problem

The most basic version of the AP that allows an agent to be assigned to multiple tasks is the generalized assignment problem or GAP. This model assumes, as in the classic AP, that each task will be assigned to one agent, but it allows for the possibility that an agent may be assigned more than one task, while recognizing how much of an agent's capacity to do those tasks each one would use. Thus the GAP is an example of a one-to-many assignment problem that recognizes capacity limits. Recognizing that a task may use only part of an agent's capacity (GAP) rather than all of it (AP), leads to the following model:

$$\text{Minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$\text{Subject to:} \quad \sum_{i=1}^{m} x_{ij} = 1 \quad j = 1,\ldots,n,$$

$$\sum_{j=1}^{n} a_{ij}x_{ij} \leqslant b_i \quad i = 1,\ldots,m,$$

$$x_{ij} = 0 \text{ or } 1,$$

where $x_{ij} = 1$ if agent i is assigned to task j, 0 if not, c_{ij} = the cost of assigning agent i to task j, a_{ij} is the amount of agent i's capacity used if that agent is assigned to task j, and b_i is the available capacity of agent i. The first set of constraints ensures that every task is assigned to only one agent and the second set of constraints ensures that the set of tasks assigned to an agent do not exceed its capacity. Note that although there are m constraints in the second set, one for each agent, only one "resource", the agent's capacity for doing these tasks, is being considered.

Cattrysse and Van Wassenhove, in their survey of algorithms for GAP [13], identify a variety of applications in which GAP has been used either directly or as a sub-problem within a broader model type. Applications referenced include vehicle routing, fixed charge location problems, grouping and loading for flexible manufacturing systems, scheduling projects, allocating storage

space, designing communication networks, scheduling payments on accounts, assigning software development tasks to programmers, assigning jobs to computers in networks, scheduling variable length TV commercials, and assigning ships to overhaul facilities. Campbell and Diaby [9] discuss the use of GAP as an approximation for the problem of allocating cross-trained workers to multiple departments when the benefits of assigning additional personnel to a department are given by a concave, rather than linear, function.

3.2. The multiple resource GAP

A logical extension of GAP is to recognize how multiple resources may limit the capacities of the agents. Revising the GAP model above to recognize multiple resources (MRGAP) simply involves adding additional constraints of the second type, one for each resource. Applications of MRGAP include recognizing vehicle routing in designing distribution systems [6], order selection recognizing early-tardy and subcontracting costs in multiple-period production planning for flexible manufacturing systems [44], developing a snow disposal plan for a city [10], scheduling the use of a space telescope [52], and determining how to allocate databases in a distributed computer system [55].

LeBlanc et al. [43] present an extension of MRGAP that includes recognition of setup times and costs in allowing for splitting production lots among different machines in developing production plans. Shtub and Kogan [62] present an extension of MRGAP to the case where demand varies over time and capacity assignments are dynamic. They show how this extended model can be used for strategic capacity planning.

3.3. The bottleneck GAP

In the same way that the minimum-sum objective function of the classic AP can be replaced by a minimax expression (the bottleneck AP or BAP), so there may be a minimax objective for the GAP, which leads to the Bottleneck Generalized Assignment Problem (BGAP). Mazzola and Neebe [50] identify two versions of the BGAP: (1) the Task BGAP, in which the objective is to minimize the maximum of the costs of the assignments made, and (2) the Agent BGAP, in which the objective is to minimize the maximum of the sum of the costs for an agent. Applications of the two models include facility location and machine scheduling. Martello and Toth [48] suggest that the Task BGAP could be useful in public sector modeling for situations such as assigning urban areas to service centers, like hospitals or schools, where an objective could be to minimize the maximum travel time between a center and the area being served.

3.4. The imbalanced time minimizing assignment problem

Aora and Puri [3] consider a variation on the BAP (Section 2.4) in which the number of agents (m) is less than the number of tasks to be done (n) and, since every task must be done, some agent or agents will have to be assigned to more than one task. As in the Agent BGAP model just discussed, all agents may start working on their tasks simultaneously, although an agent with more than one task must do them sequentially, as in the categorized assignment problem (Section 2.10). The objective is to minimize the time by which all the tasks are completed. Unlike the BGAP model, however, Aora and Puri do not recognize resource limits on the number of tasks to which an agent may be assigned.

The mathematical model for the imbalanced time minimizing assignment problem may be given as:

$$\text{Minimize} \quad \max_i \sum_{j=1}^{n} c_{ij}x_{ij}$$

$$\text{Subject to:} \quad \sum_{i=1}^{m} x_{ij} = 1 \quad j = 1,\ldots,n,$$

$$\sum_{j=1}^{n} x_{ij} \geqslant 1 \quad i = 1,\ldots,m,$$

$$x_{ij} = 0 \text{ or } 1.$$

3.5. The β-assignment problem

Chang and Ho [15] present three variations on the one-to-many assignment problem that they call

β-assignment problems (β-AP). In addition to the usual AP assumption that a task must be assigned to only one agent, they consider, as in Section 2.2, the restriction that not every agent is qualified to do every task, so that, since tasks can only be assigned to qualified agents, an agent may be assigned more than one task. Unlike the GAP (Section 3.1), these models do not recognize how much of an agent's capacity a task will take.

3.5.1. The cardinality β-assignment problem

Chang and Ho [15] first consider the problem of minimizing the maximum number of tasks assigned to any worker, which they call the *cardinality* β-AP, giving references to papers that apply this model to scheduling problems. With some change in notation to make it more consistent with the notation used elsewhere in this paper, their model for the cardinality β-AP is:

$$\begin{aligned}
&\text{Minimize} && \max_i \sum_{j\in T_i} x_{ij} \\
&\text{Subject to:} && \sum_{i\in A_j} x_{ij} = 1 \quad j = 1,\ldots,n, \\
& && x_{ij} = 0 \text{ or } 1,
\end{aligned}$$

where $x_{ij} = 1$ if agent i is assigned to task j, 0 if not, T_i is the set of tasks j that agent i is qualified to perform, and, correspondingly, A_j is the set of agents i that are qualified to perform task j. If, as in Section 2.2, Q is an $m \times n$ matrix of 0–1 elements q_{ij}, where $q_{ij} = 1$ if and only if agent i is qualified to perform task j, then T_i is the set of column indexes j for which $q_{ij} = 1$ and, correspondingly, A_j is the set of the row indexes i for which $q_{ij} = 1$.

Using the notation in Section 2.2, the model for the cardinality β-AP may be rewritten as:

$$\begin{aligned}
&\text{Minimize} && \max_i \sum_{j=1}^{n} q_{ij}x_{ij} \\
&\text{Subject to:} && \sum_{i=1}^{m} q_{ij}x_{ij} = 1 \quad j = 1,\ldots,n, \\
& && x_{ij} = 0 \text{ or } 1.
\end{aligned}$$

Following the notation in [15], let the optimal value of the objective function for this problem be k^*.

3.5.2. The bottleneck β-assignment problem

Next, considering that each agent-task assignment has an associated value, such as the profit to be derived from that assignment, Chang and Ho [15] consider the *bottleneck* β-AP, in which the objective is to identify, from among all optimal solutions to the cardinality β-AP, the one that will maximize the minimum weight of any assignment. Using both notation based on Chang and Ho and the notation in Section 2.2, the mathematical model for the bottleneck β-AP is:

$$\begin{aligned}
&\underset{j}{\text{Maximize}} \ \min_{i\in Aj}\{c_{ij}x_{ij}\} \text{ or } \underset{j}{\text{Maximize}} \ \min_i\{c_{ij}q_{ij}x_{ij}\} \\
&\text{Subject to:} \ \sum_{i\in A_j} x_{ij} \text{ or } \sum_{i=1}^{m} q_{ij}x_{ij} = 1 \quad j = 1,\ldots,n, \\
&\qquad \sum_{j\in T_i} x_{ij} \text{ or } \sum_{j=1}^{n} q_{ij}x_{ij} \leqslant k^* \quad i = 1,\ldots,m, \\
&\qquad x_{ij} = 0 \text{ or } 1,
\end{aligned}$$

where, in addition to those model components defined for the cardinality β-AP, we define c_{ij} as the value of having agent i assigned to task j. Note that the second set of constraints ensures that the maximum number of tasks to which any agent is assigned does not increase, although it does allow for the possibility that the number of tasks to which any particular agent will be assigned may increase.

3.5.3. The weighted β-assignment problem

Finally, Chang and Ho [15] consider the *weighted* β-AP, in which the objective is to identify, from among all solutions to the cardinality β-AP, the one that will maximize the sum of the values of the assignments. Again using both notation based on Chang and Ho and the notation in Section 2.2, the mathematical model for the weighted β-AP is:

$$\begin{aligned}
&\text{Maximize} && \sum_{i=1}^{m}\sum_{j\in T_i} c_{ij}x_{ij} \quad \text{or} \\
&\text{Maximize} && \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}q_{ij}x_{ij}
\end{aligned}$$

subject to the same set of constraints as those in the bottleneck β-AP.

3.6. The quadratic generalized assignment problem

Bokhari [5] considers a variety of problems in designing distributed and parallel processing systems that involve assigning the modules of a program, which can include storage, processing, and peripheral devices such as printers, over two or more interconnected computers so as to minimize some cost measure, which could be financial, time-based, or use some other metric. Allowing two or more modules to be assigned to the same processor makes this a one-to-many or generalized assignment problem, although the specific problem to be addressed may not involve recognizing resource limits, such as storage space, for the processors. What makes these problems quadratic is the need to recognize that the cost to transfer program execution from a module located on one computer to another module located on a different computer depends on what those modules are and the computers on which they are located. Bokhari shows how, in many problems with special, although commonly occurring, structures, solutions can be relatively easily determined using methods from graph theory.

4. Multi-dimensional assignment problems

All of the models discussed so far are two-dimensional or two-index models, meaning that the problem is one of optimally matching the members of two sets. In some cases, however, the problem is one of matching the members of three or more sets, with three being the most commonly encountered. For example, the problem could be one of matching jobs with workers and machines or one of assigning students and teachers to classes and time slots. (There is a considerable body of literature on the latter problem, generally called the school or university *timetabling* problem, which is not considered here. The interested reader is referred to de Werra [70] for a review of the timetabling literature up to 1985 and to Daskalai et al. [18] for a recent paper on the subject with additional references. Foulds and Johnson [23] describe an example of a functioning system.) Pierskalla [54] suggests an example "where a company realizes that its distribution system is in need of two new warehouses in the next three years and a third new warehouse in about five years. The company is not only faced with the problem of where to locate the warehouses but also when should the warehouses be built".

Gilbert and Hofstra [30] identify two (among several) different versions of the three-dimensional AP before discussing a more general multi-dimensional problem. The two three-dimensional APs discussed, which they suggest have the most potential for practical application, are called the *planar* and *axial*. Where the two model forms differ is in which combinations of assignment variables must add to 1.

In addition to discussing these two three-dimensional AP types and a more general multi-dimensional problem, Gilbert and Hofstra [30] identify special cases of the planar and axial versions of the three-dimensional AP that are relatively easy to solve (meaning that they are not NP-hard, but can be solved using polynomial time algorithms).

4.1. The planar three-dimensional assignment problem

Gilbert et al. [29] provided an example of the planar three-dimensional assignment problem that involved scheduling meetings between r vendors and s customers over a set of t time periods ($r \geqslant s \geqslant t$), where, during each time period, each customer is to meet one vendor and each vendor is to meet at most one customer. The mathematical model for the planar three-dimensional AP is:

$$\text{Minimize} \quad \sum_{i=1}^{r}\sum_{j=1}^{s}\sum_{k=1}^{t} c_{ijk}x_{ijk}$$

$$\text{Subject to:} \quad \sum_{i=1}^{r} x_{ijk} = 1 \quad \text{for each } j,k \text{ combination},$$

$$\sum_{j=1}^{s} x_{ijk} \leqslant 1 \quad \text{for each } i,k \text{ combination},$$

$$\sum_{k=1}^{t} x_{ijk} \leqslant 1 \quad \text{for each } i,j \text{ combination},$$

$$x_{ijk} = 0 \text{ or } 1 \quad \text{for all } i,j,k,$$

where $x_{ijk} = 1$ if vendor i meets with customer j during period k, 0 if not, and c_{ijk} is the cost of such

a meeting. The first set of constraints requires each customer to meet with exactly one vendor in each period; the second set of constraints allows each vendor to meet with at most one customer per time period; the third set of constraints prevents customers and vendors from meeting more than once. Besides the example that motivated this description, applications identified in [29] include scheduling teacher rotation among classes and scheduling interviews in job placement centers.

4.2. The axial three-dimensional assignment problem

Gilbert and Hofstra [30] give as an example of the axial three-dimensional AP the assignment of p jobs to q workers and r machines ($p \leqslant q \leqslant r$). Extending the basic ideas of the classic AP, each job must be assigned to some worker and machine and each worker and machine can be used on at most one job. The mathematical model for the axial three-dimensional AP (modified somewhat from that in [30] so that we may continue using subscript i to refer to the agent and subscript j to refer to the task) is:

$$\text{Minimize} \quad \sum_{i=1}^{q}\sum_{j=1}^{p}\sum_{k=1}^{r} c_{ijk}x_{ijk}$$

$$\text{Subject to:} \quad \sum_{i=1}^{q}\sum_{j=1}^{p} x_{ijk} \leqslant 1 \quad \text{for each } k,$$

$$\sum_{j=1}^{p}\sum_{k=1}^{r} x_{ijk} \leqslant 1 \quad \text{for each } i,$$

$$\sum_{i=1}^{q}\sum_{k=1}^{r} x_{ijk} = 1 \quad \text{for each } j,$$

$$x_{ijk} = 0 \text{ or } 1 \quad \text{for all } i, j, k.$$

The first set of constraints says that a machine can be used on at most one job; the second set of constraints says that a worker can do at most one job; the third set of constraints ensures that every job gets assigned to some worker and machine.

4.3. The three-dimensional bottleneck assignment problem

Assigning n workers to do n jobs on n machines (a one-to-one-to one match) with the objective of minimizing the time at which the last job is completed if all the jobs are started simultaneously is an example of the basic three-dimensional bottleneck assignment problem, which is essentially a straightforward generalization of the standard two-dimensional BAP (Section 2.4). Malhotra et al. [46] present the basic mathematical model and a solution procedure based on solving a sequence of two-dimension BAPs. Recognizing that there are the same numbers of workers, jobs, and machines, the mathematical model is:

$$\text{Minimize} \quad \max_{i,j,k}\{t_{ijk}x_{ijk}\}$$

$$\text{Subject to:} \quad \sum_{j=1}^{n}\sum_{k=1}^{n} x_{ijk} = 1 \quad \text{for each } i,$$

$$\sum_{i=1}^{n}\sum_{k=1}^{n} x_{ijk} = 1 \quad \text{for each } j,$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ijk} = 1 \quad \text{for each } k,$$

$$x_{ijk} = 0 \text{ or } 1 \quad \text{for all } i, j, k,$$

where $x_{ijk} = 1$ if worker i does job j on machine k, 0 otherwise, and t_{ijk} is the time that that particular worker-job-machine combination would take. The constraints ensure that each worker does exactly one job on one machine, each job is done by exactly one worker on one machine, and each machine is used by exactly one worker on one job.

Malhotra et al. [46] also consider a variant on the basic three-dimensional BAP in which the number of workers (m) exceeds the number of jobs (n) and the number of machines (p), but includes the requirement that every worker be assigned to exactly one job and machine. Since the number of workers exceeds the number of jobs, this means that more than one worker may be assigned to the same job. Also, since the number of workers exceeds the number of machines, more than one worker may be assigned to the same machine. Defining β_j to be the minimum number of workers that are to be using some machine to work on job j and α_k to be the minimum number of workers to be doing some job using machine k, the mathematical model for this variant of the basic three-dimensional BAP is:

$$\text{Minimize} \quad \max_{i,j,k}\{t_{ijk}x_{ijk}\}$$

$$\text{Subject to:} \quad \sum_{j=1}^{n}\sum_{k=1}^{p} x_{ijk} = 1 \quad \text{for each } i,$$

$$\sum_{i=1}^{m}\sum_{k=1}^{p} x_{ijk} \geqslant \beta_j \quad \text{for each } j,$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ijk} \geqslant \alpha_k \quad \text{for each } k,$$

$$x_{ijk} = 0 \text{ or } 1 \quad \text{for all } i,j,k.$$

The authors note that a necessary and sufficient condition for there to be a solution to this problem is that m, the number of workers, must be at least as large as the sum of the αs and at least as large as the sum of the βs. What this model does not allow for is the possibility that assigning more than one worker or machine to a particular job may reduce the time required to do that job.

Vartak and Geetha [65] present models that combine the ideas of the basic three-dimensional assignment problem with the ideas of job categorization and precedence relationships either within job categories or between job categories that were discussed in [57] (Section 2.10).

Geetha and Vartak [28] consider an extension of the basic three-dimensional BAP in which the number of workers (m) is less than the number of jobs and machines (both $n > m$). Assuming that each job is to be done on a different machine, it will be necessary for at least one of the workers to be assigned to more than one job/machine combination. A reasonable objective for this problem is to minimize the time that the last worker completes all of his or her assigned jobs. The mathematical model for this problem is:

$$\text{Minimize} \quad \max_{i} \sum_{j=1}^{n}\sum_{k=1}^{n} t_{ijk}x_{ijk}$$

$$\text{Subject to:} \quad \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ijk} = 1 \quad \text{for each } k,$$

$$\sum_{i=1}^{m}\sum_{k=1}^{n} x_{ijk} = 1 \quad \text{for each } j,$$

$$x_{ijk} = 0 \text{ or } 1 \quad \text{for all } i,j,k,$$

where $x_{ijk} = 1$ if worker i is assigned to do job j on machine k, 0 if not, and t_{ijk} is the time that worker i would take to do job j on machine k. The first constraint ensures that each machine will be used on only one job and the second constraint ensures that every job will be done. If, in addition, it is desired to set upper and lower bounds on the number of jobs a worker can be assigned, then the following additional constraint set may be added:

$$L_i \leqslant \sum_{j=1}^{n}\sum_{k=1}^{n} x_{ijk} \leqslant U_i \quad \text{for each } i,$$

where L_i and U_i are those upper and lower bounds. Geetha and Vartak [28] also consider an extension of this problem in which the number of workers (m) and the number of machines (p) are both less than the number of jobs (n) so that both some of the workers and some of the machines will have to be assigned to multiple jobs.

4.4. Multi-period assignment problems

An interesting, and frequently occurring, version of a three-dimensional assignment problem that does not exactly fit any of the basic forms discussed so far arises when it is necessary to assign agents (often staff members) to changing tasks or jobs over a time horizon.

4.4.1. Scheduling medical residents to rotations

Franz and Miller [26] discuss the problem of assigning medical residents to rotations at a teaching hospital and its associated clinics over a 12-month planning horizon. While a resident must be assigned to a rotation every month and may be assigned to only one rotation at a time, the number of months for which he or she is assigned to each rotation varies, with the time depending on both the resident and the rotation. Other constraints reflect certain structural requirements, such as having particular pairs of residents assigned to the same rotation at the same time. The objective is to maximize the expressed preferences of the residents as to when they are assigned to specific rotations and when they get their vacations.

Pentico [53] addressed the similar but more restricted problem of scheduling third-year medical

students through their clerkship program, a series of classes taken one at a time over a 45-week-long planning horizon. The classes, which are of varying lengths, start on specific dates and have both minimum and maximum enrollments. The objective is to simply find a feasible schedule for all the students.

4.4.2. Scheduling transit system drivers

A common problem in managing transit systems is scheduling vehicle drivers. While this is typically handled in the US by drivers bidding for routes on a seniority basis, transit systems in other parts of the world, particularly Europe, often have to develop schedules that would be considered more equitable in spreading the desirability or undesirability of assignment to particular routes and/or schedules. Carraresi and Gallo [12] address the two problems of, first, finding a list of shifts that meet all the service requirements while meeting constraints with respect to allowable combinations of working times on consecutive days and, second, assigning these shifts to drivers in such a way as to minimize the maximum of the sums of the undesirability weights for the drivers (a bottleneck objective).

Bianco et al. [4] extend the analysis of the driver scheduling problem to develop crew rosters, weekly or monthly patterns for crew schedules that specify, for each driver, the working days and the starting and stopping times by day that meet requirements for minimum time off. Again, the objective is balancing the workload.

4.4.3. Scheduling program modules to computers over time

Bokhari [5] extends his discussion of assigning program modules to the computers in a distributed processing system (Section 3.6) to consider, for programs with a tree structure, the timing of the execution of the modules in order to take advantage of the differing loads and processing costs on different computers at different times of the day.

4.5. Other types of multi-dimensional assignment problems

Gilbert and Hofstra [30] note that, as would be expected, there are multi-dimensional versions of many of the two-dimensional APs discussed here. In addition, as described in the following sub-sections, a number of specialized multi-dimensional APs have been developed.

4.5.1. The simultaneous assignment problem

Yamada and Nasu [71] discuss an application of a many-to-one three-dimensional AP that they call the *simultaneous assignment problem*. They discuss and provide both heuristics and algorithms for the problem of simultaneously assigning cadets at Japan's National Defense Academy to science/engineering departments and to branches of Japan's armed services, recognizing both the students' preferences (the objective function) and limits on the sizes of the classes and the numbers of cadets to be assigned to the different service branches. Letting the science/engineering departments and branches of the armed service play the roles of the jobs and machines in Malhotra et al.'s second version of the three-dimensional BAP discussed in Section 4.3 [46], the constraints for this problem would be similar to the constraints in that model, although there could be constraints for upper bounds as well as lower bounds. The objective function, however, would be to *maximize* the *sum* of the objective function terms rather than to minimize their maximum.

4.5.2. The multi-level generalized assignment problem

Glover et al. [31] and Laguna et al. [41] used an extension of the basic GAP model to determine the optimal lot sizing and machine loading for multiple products over a planning horizon. What differentiates this model from the basic GAP model (Section 3.1) and the MRGAP model (Section 3.2) is that the agents (machines) can perform the tasks (producing the products in specified quantities) at different efficiency levels (lot sizes) which have different costs that include the costs of setting up and operating the machines and carrying inventory. Each product is to be produced on only one machine, using a particular lot size. Recognizing that different lot sizes use different amounts of a machine's capacity leads to a set of resource utilization constraints, one per machine.

The mathematical model for the multi-level GAP with m machines, n products, and l lot sizes is:

$$\text{Minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{l} c_{ijk}x_{ijk}$$

$$\text{Subject to:} \quad \sum_{i=1}^{m}\sum_{k=1}^{l} x_{ijk} = 1 \quad j = 1,\ldots,n,$$

$$\sum_{j=1}^{n}\sum_{k=1}^{l} a_{ijk}x_{ijk} \leqslant b_i \quad i = 1,\ldots,m,$$

$$x_{ijk} = 0 \text{ or } 1,$$

where $x_{ijk} = 1$ if machine i produces product j using lot size k, 0 if not, c_{ijk} is the cost of doing this, a_{ijk} is the amount of machine i's capacity consumed in doing this, and b_i is the amount of capacity machine i has. The first set of constraints ensures that each product will be produced on only one machine using only one lot size and the second set of constraints ensures that machine capacities are not exceeded.

5. Summary

Assignment problems involve matching the elements of two or more sets in such a way that some objective function is optimized. Since the publication by Kuhn in 1955 [38] of the Hungarian Method algorithm for its solution, the classic AP, which involves matching the elements of two sets on a one-to-one basis so as to minimize the sum of their associated weights, has spawned a wide variety of derivatives, including problems with different or multiple objectives, problems that involve one-to-many or many-to-one matching, and problems that involve matching the elements of three or more sets. In this paper we have described and provided references to most of the variations of the classic AP that have appeared in the literature over the past 50 years.

References

[1] V. Aggarwal, V.G. Tikekar, L.-F. Hsu, Bottleneck assignment problems under categorization, Computers & Operations Research 13 (1) (1986) 11–26.

[2] Y.P. Aneja, A.P. Punnen, Multiple bottleneck assignment problem, European Journal of Operational Research 112 (1) (1999) 167–173.

[3] S. Aora, M.C. Puri, A variant of time minimizing assignment problem, European Journal of Operational Research 110 (2) (1998) 314–325.

[4] L. Bianco, M. Bielli, A. Mingozzi, S. Ricciardelli, M. Spadoni, A heuristic procedure for the crew rostering problem, European Journal of Operational Research 58 (2) (1992) 272–283.

[5] S.H. Bokhari, Assignment Problems in Parallel and Distributed Computing, Kluwer Academic Publishers, Norwell, MA, 1987.

[6] J.H. Bookbinder, K.E. Reece, Vehicle routing considerations in distribution system design, European Journal of Operational Research 37 (2) (1988) 204–213.

[7] R.E. Burkard, Quadratic assignment problems, European Journal of Operational Research 15 (3) (1984) 283–289.

[8] R.E. Burkard, F. Rendl, Lexicographic bottleneck problems, Operations Research Letters 10 (5) (1991) 303–308.

[9] G.M. Campbell, M. Diaby, Development and evaluation of an assignment heuristic for allocating cross-trained workers, European Journal of Operational Research 138 (1) (2002) 9–20.

[10] J.F. Campbell, A. Langevin, The snow disposal assignment problem, Journal of the Operational Research Society 46 (8) (1995) 919–929.

[11] G. Caron, P. Hansen, B. Jaumard, The assignment problem with seniority and job priority constraints, Operations Research 47 (3) (1999) 449–454.

[12] P. Carraresi, G. Gallo, A multi-level bottleneck assignment approach to the bus drivers' rostering problem, European Journal of Operational Research 16 (2) (1984) 163–173.

[13] D.G. Cattrysse, L.N. Van Wassenhove, A survey of algorithms for the generalized assignment problem, European Journal of Operational Research 60 (3) (1992) 260–272.

[14] E. Çela, The Quadratic Assignment Problem: Theory and Algorithms, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.

[15] G.J. Chang, P.-H. Ho, The β-assignment problems, European Journal of Operational Research 104 (3) (1998) 593–600.

[16] M. Dell'Amico, S. Martello, The k-cardinality assignment problem, Discrete Applied Mathematics 76 (1–3) (1997) 103–121.

[17] M. Dell'Amico, S. Martello, Linear assignment, in: M. Dell'Amico, F. Maffioli, S. Martello (Eds.), Annotated Bibliographies in Combinatorial Optimization, John Wiley & Sons Ltd., Chichester, England, 1997.

[18] S. Daskalai, T. Birbas, E. Housos, An integer programming formulation for a case study in university timetabling, European Journal of Operational Research 153 (1) (2004) 117–135.

[19] Z. Drezner, A new genetic algorithm for the quadratic assignment problem, INFORMS Journal on Computing 15 (3) (2003) 320–330.

[20] C.W. Duin, A. Volgenant, Minimum deviation and balanced optimization: A unified approach, Operations Research Letters 10 (1) (1991) 43–48.
[21] A.N. Elshafei, Hospital lay-out as a quadratic assignment problem, Operational Research Quarterly 28 (1) (1977) 167–179.
[22] L.R. Ford Jr., D.R. Fulkerson, Flows in Networks, Princeton University Press, Princeton, NJ, 1966 (Section II.6).
[23] L.R. Foulds, D.G. Johnson, SlotManager: A microcomputer-based decision support system for university timetabling, Decision Support Systems 27 (4) (2000) 367–381.
[24] L.R. Foulds, J.M. Wilson, On an assignment problem with side constraints, Computers & Industrial Engineering 37 (4) (1999) 847–858.
[25] A. Frank, On Kuhn's Hungarian method—a tribute from Hungary, Naval Research Logistics 52 (1) (2005) 2–6.
[26] L.S. Franz, J.L. Miller, Scheduling medical residents to rotations: Solving the large-scale multiperiod staff assignment problem, Operations Research 41 (2) (1993) 269–279.
[27] S. Geetha, K.P.K. Nair, A variation of the assignment problem, European Journal of Operational Research 68 (3) (1993) 422–426.
[28] S. Geetha, M.N. Vartak, The three-dimensional assignment problem with capacity constraints, European Journal of Operational Research 73 (3) (1994) 562–568.
[29] K.C. Gilbert, R.B. Hofstra, R. Bisgrove, An algorithm for a class of three-dimensional assignment problems arising in scheduling operations, Institute of Industrial Engineers Transactions 19 (1) (1987) 29–33.
[30] K.C. Gilbert, R.B. Hofstra, Multidimensional assignment problems, Decision Sciences 19 (2) (1988) 306–321.
[31] F. Glover, J. Hultz, D. Klingman, Improved computer-based planning techniques—Part II, Interfaces 9 (4) (1979) 12–20.
[32] G. Grygiel, Algebraic Σ_k-assignment problems, Control and Cybernetics 10 (3&4) (1981) 155–165.
[33] O. Gross, The bottleneck assignment problem, The Rand Corporation, Paper P-1630, 6 March 1959, presented at the RAND Symposium on Mathematical Programming (Linear Programming and Extensions), 16–20 March 1959.
[34] S.K. Gupta, A.P. Punnen, Minimum deviation problems, Operations Research Letters 7 (4) (1988) 201–204.
[35] F.S. Hillier, M.M. Connors, Quadratic assignment problem algorithms and the location of indivisible facilities, Management Science 13 (1) (1966) 42–57.
[36] J. Kennington, Z. Wang, A shortest augmenting path algorithm for the semi-assignment problem, Operations Research 40 (1) (1992) 178–187.
[37] P. Kouvelis, G. Yu, Robust Discrete Optimization and Its Applications, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1997.
[38] H.W. Kuhn, The Hungarian method for the assignment problem, Naval Research Logistics Quarterly 2 (1&2) (1955) 83–97 (original publication).
[39] H.W. Kuhn, The Hungarian method for the assignment problem, Naval Research Logistics 52 (1) (2005) 7–21 (50th anniversary reprint).
[40] H.W. Kuhn, Statement for Naval Research Logistics, Naval Research Logistics 52 (1) (2005) 6.
[41] M. Laguna, J.P. Kelly, J.L. Gonzalez-Velarde, F. Glover, Tabu search for the multilevel generalized assignment problem, European Journal of Operational Research 82 (1) (1995) 176–189.
[42] E.L. Lawler, The quadratic assignment problem, Management Science 9 (4) (1963) 586–599.
[43] L.J. LeBlanc, A. Shtub, G. Anandalingam, Formulating and solving production planning problems, European Journal of Operational Research 112 (1) (1999) 54–80.
[44] D.-H. Lee, Y.-D. Kim, A multi-period order selection problem in flexible manufacturing systems, Journal of the Operational Research Society 49 (3) (1998) 278–287.
[45] S.M. Lee, M.J. Schniederjans, A multicriteria assignment problem: A goal programming approach, Interfaces 13 (4) (1983) 75–81.
[46] R. Malhotra, H.L. Bhatia, M.C. Puri, The three dimensional bottleneck assignment problem and its variants, Optimization 16 (2) (1985) 245–256.
[47] S. Martello, W.R. Pulleyblank, P. Toth, D. de Werra, Balanced optimization problems, Operations Research Letters 3 (5) (1984) 275–278.
[48] S. Martello, P. Toth, The bottleneck generalized assignment problem, European Journal of Operational Research 83 (3) (1995) 621–638.
[49] J.B. Mazzola, A.W. Neebe, Resource-constrained assignment scheduling, Operations Research 34 (4) (1986) 560–572.
[50] J.B. Mazzola, A.W. Neebe, Bottleneck generalized assignment problems, Engineering Costs and Production Economics 14 (1) (1988) 61–65.
[51] A. Mehrez, Y. Yuan, A. Gafni, Stable solutions vs. multiplicative utility solutions for the assignment problem, Operations Research Letters 7 (3) (1988) 131–139.
[52] J. Nowakovski, W. Schwarzler, E. Triesch, Using the generalized assignment problem in scheduling the ROSAT space telescope, European Journal of Operational Research 112 (3) (1999) 531–541.
[53] D.W. Pentico, Computer scheduling of medical school clerkships, Computers & Education 4 (2) (1980) 139–143.
[54] W.P. Pierskalla, The multidimensional assignment problem, Operations Research 16 (2) (1968) 422–431.
[55] H. Pirkul, An integer programming model for the allocation of databases in a distributed computer system, European Journal of Operational Research 26 (3) (1986) 401–411.
[56] C. Prins, An overview of scheduling problems arising in satellite communications, Journal of the Operational Research Society 45 (6) (1994) 611–623.
[57] A.P. Punnen, Y.P. Aneja, Categorized assignment scheduling: A tabu search approach, Journal of the Operational Research Society 44 (7) (1993) 673–679.
[58] A. Ravindran, V. Ramaswami, On the bottleneck assignment problem, Journal of Optimization Theory and Applications 21 (4) (1977) 451–458.

[59] A. Scarelli, S.C. Narula, A multicriteria assignment problem, Journal of Multi-Criteria Decision Analysis 11 (2) (2002) 65–74.
[60] C.P. Seshan, Some generalisations of the time minimising assignment problem, Journal of the Operational Research Society 32 (6) (1981) 489–494.
[61] M. Shigeno, Y. Saruwatari, T. Matsui, An algorithm for fractional assignment problems, Discrete Applied Mathematics 56 (2–3) (1995) 333–343.
[62] A. Shtub, K. Kogan, Capacity planning by the dynamic multi-resource generalized assignment problem (DMR-GAP), European Journal of Operational Research 105 (1) (1998) 91–99.
[63] P.T. Sokkalingam, Y.P. Aneja, Lexicographic bottleneck combinatorial problems, Operations Research Letters 23 (1) (1998) 27–33.
[64] L.Y. Tsui, C.-H. Chang, An optimal solution to a dock door assignment problem, Computers & Industrial Engineering 23 (1–4) (1992) 283–286.
[65] M.N. Vartak, S. Geetha, Specially structured precedence constraints in three-dimensional bottleneck assignment problems, Journal of the Operational Research Society 41 (4) (1990) 339–344.
[66] A. Volgenant, Linear and semi-assignment problems: A core oriented approach, Computers & Operations Research 23 (10) (1996) 917–932.
[67] A. Volgenant, Solving some lexicographic multi-objective combinatorial problems, European Journal of Operational Research 139 (3) (2002) 578–584.
[68] A. Volgenant, A note on the assignment problem with seniority and job priority constraints, European Journal of Operational Research 154 (1) (2004) 330–335.
[69] D.F. Votaw, A. Orden, The personnel assignment problem, Symposium on Linear Inequalities and Programmng, SCOOP 10, US Air Force, 1952, pp. 155–163.
[70] D. de Werra, An introduction to timetabling, European Journal of Operational Research 19 (2) (1985) 151–162.
[71] T. Yamada, Y. Nasu, Heuristic and exact algorithms for the simultaneous assignment problem, European Journal of Operational Research 123 (3) (2000) 531–542.
[72] Y. Yuan, A. Mehrez, A. Gafni, Reducing bias in a personnel assignment process via multiplicative utility solution, Management Science 38 (2) (1992) 227–239.