ENGINEERING JOURNAL

Article

A Two-Step Tabu Search Heuristic for Multi-Period Multi-Site Assignment Problem with Joint Requirement of Multiple Resource Types

Siravit Swangnop and Paveena Chaovalitwongse*

Department of Industrial Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok 10330, Thailand
*E-mail: paveena.c@chula.ac.th (Corresponding author)

Abstract. An assignment problem has been extensively studied and applied in many industries. Variations of assignment problem have been proposed and appeared in literatures for many years. This paper extends the variation of assignment problem in the dimension of task and resource by proposing the joint requirement of multiple resource types in a multi-period multi-site assignment problem. The specific characteristic is that there are many multi-skill resource types and tasks require joint requirement of more than one resource type to operate. An application of this model can be found in healthcare industry, especially in clinic networks or hospital networks, which have many service locations, have many resource types such as doctors, nurses or medical equipments and definitely require more than one resource type for operations. This paper proposes a two–step Tabu search heuristic for multi-period multi-site assignment problem with joint requirement of multiple resource types. The specified neighborhood strategy, short-term memory and long-term memory are designed for the addressed problem in order to generate an efficient move to improve solutions. From computational study, solutions from Tabu search algorithm are compared with optimal solutions from CPLEX, and the result shows that, for small size problems, most solutions are close to optimal solutions (average gap = 2.2%), for medium size problems, the algorithm can provide good solutions in a short time comparing with CPLEX (average gap = 5.8%), and for large size problem, four out of five solutions from the proposed algorithm are better than solutions from CPLEX in a limit of time.

Keywords: Assignment problem, Tabu search, heuristic, health resource, joint requirement.

ENGINEERING JOURNAL Volume 18 Issue 3
Received 4 November 2013
Accepted 20 January 2014
Published 10 July 2014
Online at http://www.engj.org/
DOI:10.4186/ej.2014.18.3.83

DOI:10.4186/ej.2014.18.3.83

1. Introduction

An assignment problem has been extensively studied and applied in many industries, namely dairy [1], clothing [2], mining [3], airlines [4], automated manufacturing [5] and service industries [6]. First appearing in 1952 [7], the classic assignment problem is to find a one-to-one matching between n tasks and m agents and the objective function is to minimize the total cost. Over the past few decades, the classic assignment problem has been extended and many variations of the assignment problem are proposed, for example, variation in objective function such as maximizing profit[8, 9] or minimizing the maximum number of travelling time[10], variation in planning period such as three-dimensional assignment problem [11], multi-period assignment problems for medical residents [12], multi-period machine assignment [13] or variation in task and resource such as multi-resource generalized assignment problem [14], resource-constrained assignment scheduling [15], assignment problem with seniority and job priority constraints [16] and generalization of multi-resource generalized assignment problem [9].

In this study, we extend the variation of assignment problem in the dimension of task and resource by proposing the joint requirement of multiple resource types in a multi-period multi-site assignment problem. This specific characteristic is that there are multiple resource types and tasks require joint of more than one resource type for operations. This model can be found in healthcare industry, especially in clinic networks or hospital networks, which have many service locations, have many resource types such as physicians, nurses or medical equipments and require more than one resource type for operations. Only qualified resources can do tasks or treatments and allocating each resource to each site has different operation cost. The resource planners have to decide where their resources should be assigned to maximize total profit.

The remaining parts of this paper are organized as follows. The related work is reviewed in section 2. In section 3, the statement of problem and mathematical model are described. Then, Tabu search algorithm and computational experiment are presented in section 4 and 5. Finally, the conclusion and future works are summarized in section 6.

2. Related Work

In academic view, this kind of multi-period multi-site assignment problems can be found in a problem of emergency resource allocation [17-21] and a problem of health staff scheduling [22-26]. An emergency resource allocation is a problem of allocating resources such as health staff, equipments and medical supplies from possible depots or responds units to disaster sites in the disaster situations such as earthquakes, floods or hurricanes while health staff scheduling is a problem of assigning physician, nurses or aides to hospitals, wards or shifts. Both problems consider allocating or assigning resources to suitable shifts or sites as in the proposed problem and many of them consider multiple resource types.

For emergency resource allocation, their resources can be one or more than one type and time period for planning can also be one or more than one period. The performance of allocation processes and decisions in a few days after disaster strikes is an important key to reduce the number of fatalities [21]. Zhang et al. [17] proposed the model and algorithm for allocating multiple resources from emergency depots to disaster sites to fulfill all demands. Ozdamar et al. [18] proposed emergency logistics planning model in natural disaster, which consists of two decisions: allocating multiple commodities to the disaster sites and scheduling the vehicles. Tzeng et al. [20] designed a relief-distribution model for distributing relief items from collection points to transfer candidate depots and relief demand points. More details of this problem can be seen in Caunhye et al. [27] reviewing the optimization models in emergency logistics, Altay and Green Iii [28] surveying the existing OR/MS literatures in disaster operations managements and Fiedrich et al. [21] providing the definitions of core components in Emergency resource allocation problem.

For health staff scheduling, a scope of allocation and assignment can be limited in one ward/department [22, 23, 29-32], in many wards/departments in one hospital [24] or in many hospitals [25, 26, 33]. Most researchers have focused on nurse scheduling more than physician scheduling [34] and both staff are mostly considered separately. Ernst et al. [34] reviewed a problem of staff scheduling and rostering in which health staff scheduling is included. Trivedi and Warner [24] proposed algorithm for allocating available float pool of nursing personnel to various inpatient units in a hospital. The nursing personnel are divided into three types which are registered nurses, licensed practical nurses and aides and substituting of those staff is allowed with different performance. Aickelin and Dowsland [22, 29] and Dowsland [30] also considered the problem of nurse scheduling. In their models, there are sets of shift pattern and nurses are divided into many grades in which higher graded nurses can substitute nurses in lower grades. The decision is to assign nurses to shift patterns, which provides the different penalty cost in each shift pattern. For physician scheduling, Carter and Lapierre [23] studied the problem of physician scheduling in emergency room while Goyal and Yadav [33] developed mathematical model and heuristic for allocating physicians to various medical institutions. Other health staff scheduling models can be seen in [25, 26, 31, 32].

Most of these studies mainly focus on allocating or assigning resources to working sites or shifts. However, they usually consider assigning only one type of resource and if they consider multiple resources types, they do not concern joint requirement of resource types.

Another kind of problem related to the considered problem is the generalized assignment problem (GAP) in which there are many agents who has own capacity for doing tasks and they can do many tasks as long as they have enough capacity. In the past decades, many extensions of GAP are proposed and models concerning multiple resource types and joint of multiple resources are arisen. Gavish and Pirkul [14] proposed multi-resource generalized assignment problem (MRGAP), which extends GAP by allowing agents to have multiple resources and their resources are consumed when accomplishing their tasks. Mazzola and Neebe [15] defined the assignment problem with side constraints (APSC) which extends GAP as MRGAP but the resources in the system is not separated by individual. All agents can use the resources in the system until resources are out. Toktas et al. [35] combined the characteristics of the GAP and MRGAP with the APSC and generates two additional problems: collectively capacitated generalized assignment problem (CCGAP) and assignment problem with individual capacities (AIPC). Alidaee et al. [9] presented the assignment model that includes the model of MRGAP as special cases. This new model is called generalization of MRGAP (GMRGAP). An extension is that their tasks consist of many operations and if tasks are done, all its operations must be completed.

Also, although models in this problem consider multiple resources assignment and their resources can be assigned to many tasks, most models have the condition that each task must to be assigned to only one resource. There is only a model of Alidaee et al. [9] which studied deeply in assigning multiple resources to tasks. However, they have the different objective and do not consider a dimension of multiple period and multiple sites as in our models.

Although this kind of problem is in the focus of many researchers and have been extended in many areas, there are no any models considering multi-period multi-site assignment problems with joint requirement of multiple resource types as in the considered model. The objective of this research is to develop the mathematical model and the solution method based on Tabu search algorithm.

3. Problem Description

A multi-period and multi-site assignment problem determines where resources are and what resources do in each period. In our model, there are many resource types and tasks require multiple resource types for operation. All resources are allocated to the site with different operation cost, and each resource is limited to be in one site. Each task at sites provides different benefit, and resources are assigned to do tasks for total profit maximization.

A mathematical model can be written as follows:

Index

i = index for tasks; $i \in \{1, 2, 3, ..., I\}$

j = index for resources; $j \in \{1, 2, 3, ..., J\}$

p = index for periods; $p \in \{1, 2, 3, ..., P\}$

r = index for resource types; $r \in \{1, 2, 3, ..., R\}$

s = index for sites; $s \in \{1, 2, 3, ..., S\}$

Set

I_{ps} = set of task i occurring in site s in period p.

Parameters

g^{r}_{jpi} =1 if resource j in type r is qualified to do task i in period p.

= 200 otherwise. [Big M value]

b_{pi}^{r} = 1 if task i in period p requires resource type r.
= 0 otherwise.

B_{pi} = benefit when task i in period p is executed.

C_{js}^{r} = operation cost when resource j in type r is assigned to site s.

I_{ps} = set of task i occurring in site s in period p.

Decision variables

Y_{jpi}^{r} = 1 if resource j in type r is assigned to task i in period p.
= 0 otherwise.

Z_{js}^{r} = 1 if resource j in type r is assigned to site s.
= 0 otherwise.

W_{pi} = 1 if task i in period p is executed.
= 0 otherwise.

Objective function

$$\text{Maximize total profit} = \sum_{p=1}^{P}\sum_{i=1}^{I} B_{pi}W_{pi} - \sum_{r=1}^{R}\sum_{j=1}^{J}\sum_{s=1}^{S} C_{js}^{r}Z_{js}^{r} \quad (1)$$

Constraints

Qualification constraint: $$\sum_{i=1}^{I} g_{jpi}^{r}Y_{jpi}^{r} \le 1\,; \forall r \in \{1,...,R\}, j \in \{1,...,J\}, p \in \{1,...,P\} \quad (2)$$

Location constraint: $$\sum_{s=1}^{S} Z_{js}^{r} = 1\,; \forall r \in \{1,...,R\}, j \in \{1,...,J\} \quad (3)$$

Joint requirement constraint: $$\sum_{j=1}^{J} g_{jpi}^{r}Y_{jpi}^{r} = b_{pi}^{r}W_{pi}\;; \forall r \in \{1,...,R\}, p \in \{1,...,P\}, i \in \{1,...,I\} \quad (4)$$

Available task constraint: $$Z_{js}^{r} \ge Y_{jpi}^{r}\;; \forall r \in \{1,...,R\}, j \in \{1,...,J\}, p \in \{1,...,P\}, s \in \{1,...,S\}, i \in I_{ps} \quad (5)$$

The objective function, Eq. (1), maximizes the total profit, which is calculated from total benefits and total operation cost of all resources. Eq. (2) enforces that only qualified resources can do tasks and each resource is assigned to only one task per period. Eq. (3) enforces that each resource must be assigned to only one site. Eq. (4) states that only qualified resources can do tasks and tasks can be done when joint requirements of resources are satisfied, for example, if a task requires resource type 1 and 2, this task can be done (W_{pi} =1) when there are two qualified resources, selected from resource type 1 and 2, assigned to do this task. Each site has different tasks and resources can do only tasks in the site where they are assigned. Eq. (5) is used for enforcing this restriction.

4. Tabu Search Algorithm

Tabu search (TS) is a well-known meta-heuristic for solving a combinatorial optimization initiated by Glover [36-38]. A basic process for finding solutions by Tabu search algorithm is roughly divided into three steps: set an initial solution, find neighborhoods and select neighborhood to be new solution. The second and third steps are done iteratively until stopping criteria is met. Efficiency of Tabu search algorithm mainly depends on the structure of neighborhood, Tabu list and long term memory. Good neighborhood lets the algorithm find the best solution in a short time. Because moving to worse solutions is allowed, Tabu list is used to prevent algorithm from cycling or being stuck in a local optimum. Long term memory is usually used to identify the good or bad elements of the solutions or the unvisited regions and then provide the good direction of the next move.

Tabu search algorithm is widely applied in allocation, scheduling and assignment problem [5, 31, 39-43]. For problems whose decision can be divided into many steps as our model, one approach for developing algorithm is to separate the decision into many steps depending on characteristics of the problem and algorithms for each step are developed [32, 34, 43-45].

For our problem, as described in previous section, the decision of the model is to find the suitable sites (allocation) and suitable tasks (assignment) for resources. To develop algorithm for the considered problem, we separate the decision into two steps: allocating resource to sites and then assigning resources to tasks. We propose a two-step Tabu search algorithm for solving the considered problem: Main Tabu searh algorithm (MTS) for resource allocation in the first step and Sub Tabu search algorithm (STS) for resource assignment in the second step.

A structure of two-step Tabu search algorithm is illustrated in Fig. 1. The algorithm starts from generating an initial solution in Main Tabu searh algorithm (MTS). Then, a process of finding all neighborhoods is done. Each neighborhood is a set of resources which should be moved to some sites to provide a better solution. Because getting true objective function of all neighborhoods by using Sub Tabu search algorithm (STS) takes a lot of computational time, we have a process of reducing the number of neighborhood by selecting only some neighborhoods with some criteria to be candidates. After candidate list is generated, Sub Tabu Search Algorithm (STS) will be done to find a solution of resource assignment of each candidate and the true objective function will be calculated. In our STS, we design a specific Tabu list, neighborhood and diversification technique for getting better solution. After all candidates are calculated by STS, the best candidate is selected to be a new initial solution for MTS and the process of updating Tabu list and best known solution in MTS are done. The process is done iteratively until reaching the stopping criteria. The detail of MTS and STS is described as follows:

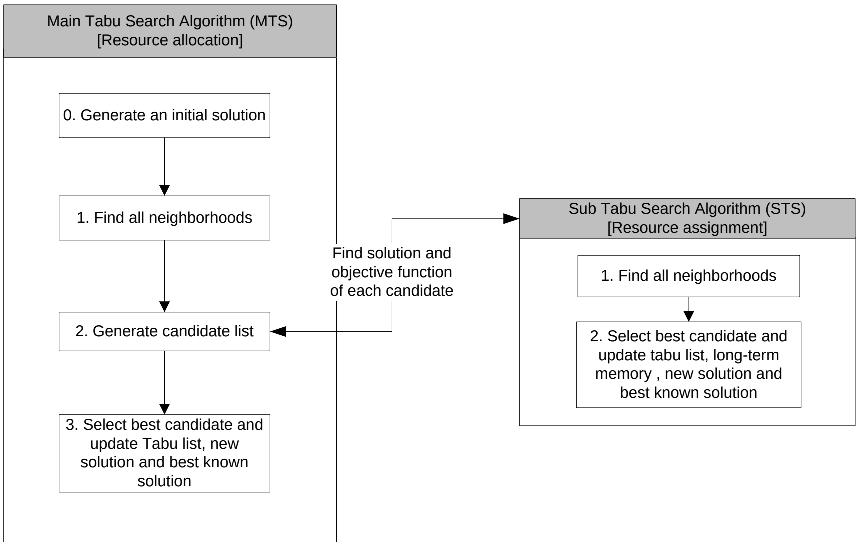
Main Tabu Search Algorithm (MTS)
[Resource allocation]
0. Generate an initial solution
1. Find all neighborhoods
2. Generate candidate list
3. Select best candidate and update Tabu list, new solution and best known solution
Find solution and objective function of each candidate
Sub Tabu Search Algorithm (STS)
[Resource assignment]
1. Find all neighborhoods
2. Select best candidate and update tabu list, long-term memory , new solution and best known solution

Fig. 1. Structure of two-step Tabu search algorithm.

4.1. Main Tabu Search Algorithm (MTS)

Details of Main Tabu Search Algorithm are described as follows:

Generate an initial solution: All resources are assigned to the site that has the lowest operation cost. Then, the problem is split up into many sub-problems. One sub-problem is a problem of one site and one period. Then, CPLEX is used to find an optimal solution of each sub-problem.

Find all neighborhoods: The objective of this step (MTS) is to move resource to the better site. Neighborhoods are generated from moving set of resources from sites to another or, in other words, changing the value of Z^r_{js} from 1 to 0 and the value of $Z^r_{js'}$ from 0 to 1. The destination site to which these resources are moved is the site where there are unassigned tasks (tasks that nobody does). The moved resources are the resource in each type that can do those unassigned tasks, which selects only one resource per type. For example, in Fig. 2, task i_1 in site s_2 is an unassigned task requiring resource type r_2 and r_4. The resource j_1 and j_2 of type r_2 and j_3 and j_4 of type r_4 can do this task. Suppose j_1 and j_3 are selected to move and, in initial solution, j_1 is in s_3 and j_3 is in s_5. Generating neighborhood is to move resource j_1 from site s_3 and j_3 from site s_5 to site s_2 or, in other words, to change the value of $Z^{r_2}_{j_1s_3}$ and $Z^{r_4}_{j_3s_5}$ from 1 to 0 and the value of $Z^{r_2}_{j_1s_2}$ and $Z^{r_4}_{j_3s_2}$ from 0 to 1. Other neighborhoods can be generated with the same concept which is moving j_1 and j_4 to site s_2, moving j_2 and j_3 to site s_2 and moving j_2 and j_4 to site s_2. This process will be done with all unassigned tasks to generate all neighborhoods.

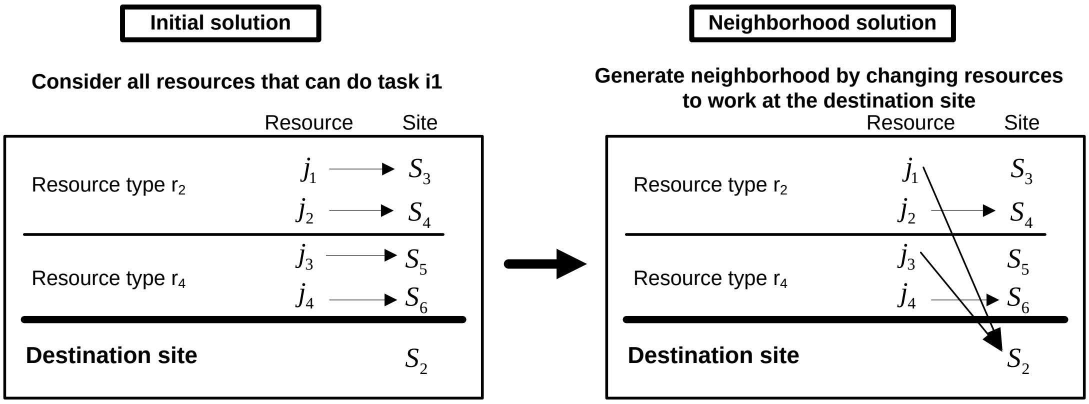

Fig. 2. Example of generating neighborhood in MTS.

Generate candidate list: Because there are a lot of neighborhoods in this step, we reduce the number of neighborhoods by selecting only some neighborhoods. We calculate surrogate objective function of each neighborhood, which consumes short computational time, and select only the best M neighborhoods to be the candidate in the candidate list. If there is more than one candidate generated from one task, only the candidate which has the highest surrogate objective function is considered to be in candidate list.

- Surrogate objective function = Tbg - Tbl + Toc
  - Tbg= Total possible benefit gain, which is the sum of highest benefits of unassigned tasks that each moved resource can do in each period in new site.
  - Tbl = Total benefit lost from moving all resource to new site.
  - Toc = Total additional operation cost from moving all resource to new site.

Tabu list: Tabu list is a short term memory used for preventing cycling. In our model, all moved resources from the best neighborhood are added to Tabu list. The resources in Tabu list are not allowed to be moved to other sites for N iterations.

Stopping rule: MTS will be run iteratively until reaching the maximum iteration W or the maximum computational time V.

4.2. Sub Tabu Search Algorithm (STS)

For each candidate, one or more resources are moved to the new site and then resources in some sites and periods are changed. The algorithm in this step is to assign resources in these sites and periods to unassigned tasks to get as much benefit as possible. Those sites and periods will be calculated by this algorithm one by one until all of them are considered.

An initial solution in this step is the solution from MTS. Details of Sub Tabu Search Algorithm are described as follows:

Find all neighborhoods: Because the objective of this step is to assign resource to unassigned tasks, the neighborhood is generated from selecting some resources to do unassigned tasks ($W_{pi} = 0$). Both available resources (resources which are not assigned to any tasks) and unavailable resources (resources which are assigned to some tasks) are able to be reassigned to do unassigned tasks. A set of resources which provides the minimum benefit lost from reassigning them to do the unassigned task is selected to be the neighborhood. The benefit lost of each neighborhood is calculated from the sum of benefit of all tasks which are cancelled because of changing resources from doing the tasks that they were assigned to doing new unassigned tasks. For example, Fig. 3, an unassigned task i_5 in period p_1 requires resource type r_2, which resource j_3 and j_4 can do, and type r_3, which resource j_5 and j_6 can do. Suppose, in initial solution, j_3 and j_4 do task i_1 and i_2 whose benefit is 100 and 50 respectively while j_5 and j_6 do task i_3 and i_4 whose benefit is 200 and 150 respectively. To generate neighborhood, we select resource j_4 and j_6 to do this unassigned task because they provide the minimum benefit lost from cancelling tasks ($B_{p_1 i_2} + B_{p_1 i_4} = 50 + 150 = 200$). The neighborhood is to change the value of $Y^{r_2}_{j_4 p_1 i_2}$, $W_{p_1 i_2}$, $Y^{r_3}_{j_6 p_1 i_4}$ and $W_{p_1 i_4}$ from 1 to 0 and change the value of $Y^{r_2}_{j_4 p_1 i_5}$, $Y^{r_3}_{j_6 p_1 i_5}$ and $W_{p_1 i_5}$ from 0 to 1. Qualified resources in this step must not only be able to do unassigned tasks but also be in the site which has those unassigned tasks. This step is done with all unassigned tasks to generate all neighborhoods. After having all neighborhoods, the neighborhood which has the highest objective function is selected to be a new initial solution. This process is done iteratively until the objective function does not improve for P iterations.

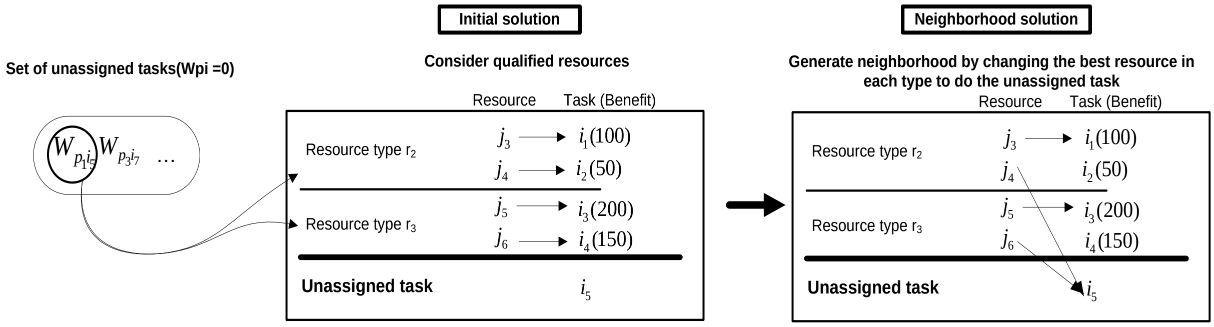

Fig. 3. Example of generating neighborhood in STS.

Tabu list: All changed resources and tasks from the best neighborhood are added to Tabu list. They are not allowed to be changed for Q iterations.

Diversification technique: This is a technique in Tabu search algorithm to enable the searching process to move and find neighborhoods in different area of solution space. We collect a frequency of tasks done in each iteration to long term memory. If the objective function of solution does not improve for D iterations, the benefit of each task will be divided by this frequency and the new benefit will be used to find the neighborhood. The process will be done for E iterations and then the long term memory will be reset.

5. Computational Experiments

A Tabu search algorithm is tested to evaluate the efficiency of the proposed algorithm to the considered problem. The algorithm was coded in C# 2010 and run on the Windows 7 Ultimate with Intel Core i5-2410M, CPU 2.30GHz and RAM 4GB. We compare our results with solutions from commercial optimization tool (ILOG CPLEX 12.1.0).

Test problems are generated into three different sizes. The first set of problem is a small size problem which takes short computational time. That is, CPLEX can find an optimal solution in a few second. The second set is a medium size problem which takes less than 4,000 seconds to find an optimal solution while the third set, a large size problem, takes more than 4000 seconds.

For the small size problem, the number of resource type is fixed to 2 (ratio of task that requires 1 type and 2 types is set to 25%: 75%). The number of period is set to 3 and 6 while the number of site is set to 5 and 10. A ratio of resource and task is set to 1:2 [6 resources: 12 tasks and 10 resources: 20 tasks] and ratio of resource that can do each task is set to 0.4. Operation cost and benefit are randomized uniformly between 500 to 2,000 and 400 to 4,000 respectively. For each problem set, 10 tests are generated.

For the medium size problem, the experiment is separated into 2 parts. A ratio of resource and task is varied in the first part while the number of resource type and the ratio of task that requires each type are varied in the second part. In the first part, the number of resource is varied from 10 to 16 and the ratio of resource and task is set to 1:2 and 1:4. The number of resource type, period and site are fixed to 2, 12 and 5 respectively. In the second part, the number of resource type is set to 2 and 3. The ratio of tasks that requires each resource type is varied from 0% to 100%. The number of resource, period, site and task are fixed to 14, 9, 5 and 30 respectively and ratio of resource that can do each task is set to 0.4. Operation cost and benefit are randomized uniformly between 2,000 to 10,000 and 400 to 4,000 respectively. For each problem set, 5 tests are generated.

For the large size problem, the number of resource, period, site and task are fixed to 20, 12, 5 and 60 respectively. The number of resource type is set to 4, 6, 8, 10 and 12 and ratio of resource that can do each task is set to 0.4. Operation cost is uniformly randomized between 2,000 to 10,000 for all problems while a benefit is uniformly randomized between 400 to 4,000 for 4, 6 and 8 resource types, 800 to 8,000 for 10 resource types and 1,200 to 12,000 for 12 resource types. The ranges of benefit in each problem size are different to maintain the value of objective function to be positive. For each problem set, 1 test is generated.

The details of all problem sizes are shown in Table1. The first seven columns are the description of tested problems, which is the size of problem, the set of problem, the number of resource, the number of task, the number of period, the number of site and the number of resource type, and the rest are the ratio of tasks that requires each resource type for operations.

Table 1. Details of all problem sizes.

Problem size	Problem set	Number of resources	Number of tasks	Number of periods	Number of sites	Number of resource types	Ratio of each resource type											
							1	2	3	4	5	6	7	8	9	10	11	12
Small problem	S1	6	12	3	10	2	0.25	0.75	-	-	-	-	-	-	-	-	-	-
	S2	6	12	6	5	2	0.25	0.75	-	-	-	-	-	-	-	-	-	-
	S3	10	20	3	10	2	0.25	0.75	-	-	-	-	-	-	-	-	-	-
	S4	10	20	6	5	2	0.25	0.75	-	-	-	-	-	-	-	-	-	-
Medium problem (Part1)	MA1.1	10	20	12	5	2	0.25	0.75	-	-	-	-	-	-	-	-	-	-
	MA1.2	12	24	12	5	2	0.25	0.75	-	-	-	-	-	-	-	-	-	-
	MA1.3	14	28	12	5	2	0.25	0.75	-	-	-	-	-	-	-	-	-	-
	MA1.4	16	32	12	5	2	0.25	0.75	-	-	-	-	-	-	-	-	-	-
	MA2.1	10	40	12	5	2	0.25	0.75	-	-	-	-	-	-	-	-	-	-
	MA2.2	12	48	12	5	2	0.25	0.75	-	-	-	-	-	-	-	-	-	-
	MA2.3	14	56	12	5	2	0.25	0.75	-	-	-	-	-	-	-	-	-	-
	MA2.4	16	64	12	5	2	0.25	0.75	-	-	-	-	-	-	-	-	-	-
Medium problem (Part2)	MB1.1	14	30	12	5	2	1.00	0	-	-	-	-	-	-	-	-	-	-
	MB1.2	14	30	12	5	2	0.75	0.25	-	-	-	-	-	-	-	-	-	-
	MB1.3	14	30	12	5	2	0.50	0.50	-	-	-	-	-	-	-	-	-	-
	MB1.4	14	30	12	5	2	0.25	0.75	-	-	-	-	-	-	-	-	-	-
	MB1.5	14	30	12	5	2	0	1.00	-	-	-	-	-	-	-	-	-	-
	MB2.1	14	30	9	5	3	1.00	0	0	-	-	-	-	-	-	-	-	-
	MB2.2	14	30	9	5	3	0.60	0.20	0.20	-	-	-	-	-	-	-	-	-
	MB2.3	14	30	9	5	3	0.20	0.60	0.20	-	-	-	-	-	-	-	-	-
	MB2.4	14	30	9	5	3	0.20	0.20	0.60	-	-	-	-	-	-	-	-	-
	MB2.5	14	30	9	5	3	0	0	1.00	-	-	-	-	-	-	-	-	-
Large problem	L1	20	60	12	5	4	0.10	0.20	0.30	0.40	-	-	-	-	-	-	-	-
	L2	20	60	12	5	6	0.05	0.10	0.15	0.15	0.20	0.35	-	-	-	-	-	-
	L3	20	60	12	5	8	0.05	0.05	0.10	0.10	0.15	0.15	0.20	0.20	-	-	-	-
	L4	20	60	12	5	10	0.01	0.04	0.05	0.05	0.06	0.10	0.14	0.15	0.20	0.20	-	-
	L5	20	60	12	5	12	0.01	0.01	0.03	0.03	0.06	0.06	0.09	0.09	0.13	0.13	0.18	0.18

All parameters of Tabu search algorithm are set according to size of the problem. From preliminary experiments, the suitable parameters for STS can be set as follows:

$$P = \text{ROUNDUP(6*(number of task),0)} \tag{6}$$

$$Q = \text{ROUNDUP}\big(\text{MIN}\big(0.7 * \text{(number of resource)}, 0.1 * \text{(number of task)}\big), 0\big) \tag{7}$$

$$D = \text{ROUNDUP(0.4*(number of task),0)} \tag{8}$$

$$E = 2 \tag{9}$$

For MTS, a suitable size of Tabu list (N) to prevent algorithm from being stuck in a local optimum is 3 or 4 depending on the problem size. The number of maximum iteration (V) and candidate (M) affect directly to the quality of the solution. A larger number of V and M increase the opportunity to find better solutions; however, it takes more computational time. V and M are limited to the suitable value according to the problem size and the computational time. All MTS and STS parameters are shown in Table 2. The parameters are divided into 2 groups: MTS and STS. In MTS, the Max iteration (W)/time (V), Tabu list (N) and Candidate list (M) is the maximum iteration or maximum time for running MTS, the number of iteration for keeping moved resources in Tabu list and the number of neighborhood in candidate list respectively. In STS, the Max iteration (P), Tabu list (Q) Tricker for Divert (D) and Duration for Divert (E) is the maximum iteration for running STS, the number of iteration for keeping changed resources and tasks in Tabu list, the number of iteration for using diversification technique and the duration for using diversification technique respectively.

Table 2. Parameters for all test problems.

Problem size	Problem set	Parameters						
		MTS			STS			
		Max iteration (W) /time (V)	Tabu list (N)	Candidate list (M)	Max iteration (P)	Tabu list (Q)	Tricker for Divert (D)	Duration for Divert (E)
Small problem	S1	5 seconds	3	4	8	1	1	2
	S2	5 seconds	3	8	15	1	1	2
	S3	5 seconds	3	8	12	1	1	2
	S4	5 seconds	3	4	24	1	2	2
Medium problem (Part1)	MA1.1	200 iterations	3	3	1	2	2	24
	MA1.2	200 iterations	3	3	1	2	2	29
	MA1.3	200 iterations	3	3	1	3	2	34
	MA1.4	200 iterations	3	3	1	3	2	39
	MA2.1	200 iterations	3	3	1	4	2	48
	MA2.2	200 iterations	3	3	1	4	2	58
	MA2.3	200 iterations	3	3	2	5	2	68
	MA2.4	200 iterations	3	3	2	6	2	77
Medium problem (Part2)	MB1.1	200 iterations	3	3	1	3	1	36
	MB1.2	200 iterations	3	3	1	3	1	36
	MB1.3	200 iterations	3	3	1	3	1	36
	MB1.4	200 iterations	3	3	1	3	1	36
	MB1.5	200 iterations	3	3	1	3	1	36
	MB2.1	80 iterations	3	6	1	3	1	36
	MB2.2	80 iterations	3	6	1	3	1	36
	MB2.3	80 iterations	3	32	1	3	1	36
	MB2.4	80 iterations	3	64	1	3	1	36
	MB2.5	80 iterations	3	170	1	3	1	36
Large problem	L1	100 iterations	4	32	2	5	2	72
	L2	100 iterations	4	32	2	5	2	72
	L3	100 iterations	4	32	2	5	2	72
	L4	100 iterations	4	32	2	5	2	72
	L5	100 iterations	4	32	2	5	2	72

*The value of all parameters in STS is calculated by equation (6) – (9).

The experiment of small size problems

In the small size problem, 4 problems are generated: S1, S2, S3 and S4. For S1 and S3, there are only few periods but many sites whereas, for S2 and S4, there are only few sites but many periods. The number of resource and task in S3 and S4 is more than in S1 and S2. Time for running MTS for all problems is limited to 5 seconds. The results of the experiment are illustrated in Table 3. The second column (#OPT by Tabu) shows the number of optimal solution found by Tabu search algorithm. The average optimal gap is shown in the third column, which is calculated from [(solution of CPLEX) – (solution of Tabu search)] *100 / (solution of CPLEX). The result shows that, for all the test problems (40 tests), 13 optimal solutions are found and the average optimal gap ranges from 0.6 to 4.0.

Table 3. The results from experiments of small size problems.

Problem set	#OPT by Tabu (10 tests)	Average gap (%)
S1	5	0.6
S2	7	1.5
S3	1	2.7
S4	0	4.0

The experiment of medium size problems [part1]

In the medium size problem [part1], the experiment is separated into two groups, and four problems per group are generated: MA1.1 to MA1.4 for the first group, and MA2.1 to MA2.4 for the second group. All parameters in both groups are the same except the number of task which is set by the ratio of resource and task (1:2 for the first group and 1:4 for the second group). The result of the experiment is shown in Table 4. The second column shows the computational time of CPLEX while the third column shows time to find the best solution of Tabu search algorithm. The forth to sixth column show the minimum optimal gap, maximum optimal gap and average optimal gap. The average optimal gap and computational time are plotted in Fig. 4. The result shows that the computational time of CPLEX considerably increases when the number and ratio of resource and task increase. Tabu search algorithm can find good solutions in a short time comparing with CPLEX and the quality of the solution remains good when the number and ratio of the resource and task increase.

Table 4. The results from experiments of medium size problems [part 1].

Problem set	CPLEX time (sec)	Tabu time(sec)	Minimum gap (%)	Maximum gap (%)	Average gap (%)
MA1.1	52	5	2.8	7.9	5.3
MA1.2	48	6	3.3	8.8	6.0
MA1.3	108	33	3.3	7.4	5.1
MA1.4	775	17	2.8	7.5	4.8
MA2.1	66	39	3.3	7.4	5.5
MA2.2	212	49	3.8	6.0	4.5
MA2.3	335	64	4.8	6.7	5.3
MA2.4	953	99	3.7	5.4	4.4

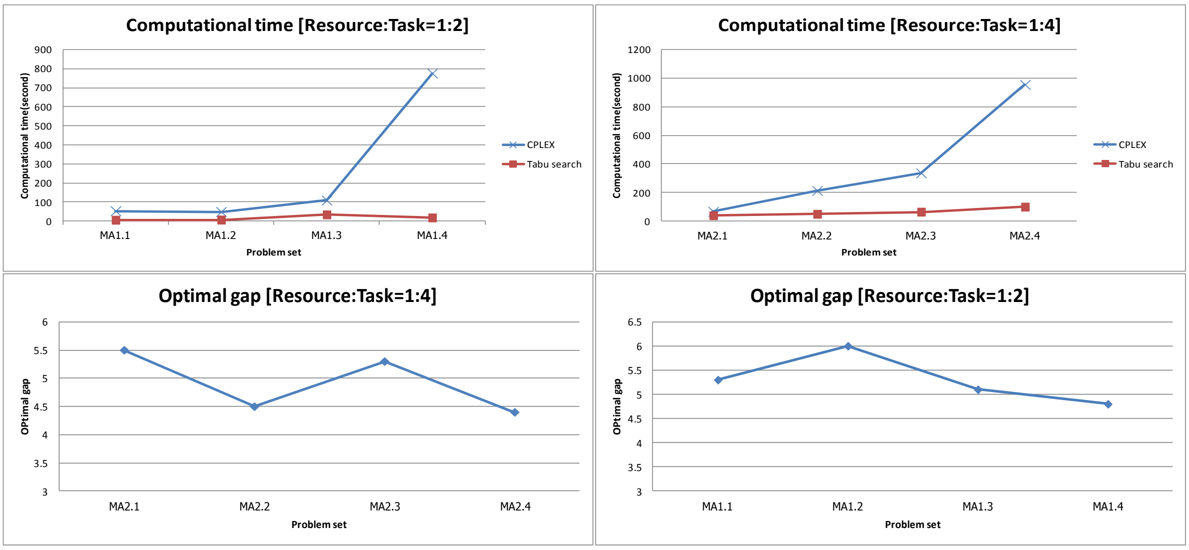

Fig. 4. Computational time and optimal gap of medium size problem [part 1].

The experiment of medium size problems [part2]

In the medium size problem [part2], the experiment is separated into two groups by the number of resource type, and five problems per group are generated: MB1.1. to MB1.5 for the first group and MB2.1 to MB2.5 for the second group. All parameters in both groups are the same except the number of resource type varied from 2 to 3 and the ratio of task that requires each type varied from 0% to 100%. The result of the experiment is shown in Table 5 and the computational time and optimal gap are illustrated in Fig. 5. The result shows that the computational time of CPLEX extremely increases when the number of resource type and the ratio increase (for this experiment, the number of resource type just increases by 1). The optimal gap slightly increases when the ratio increases. However, the algorithm can still provide good solutions in a short time comparing with the CPLEX.

Table 5. The results from experiments of medium size problems [part 2].

Problem set	CPLEX time (sec)	Tabu time(sec)	Minimum gap (%)	Maximum gap (%)	Average gap (%)
MB1.1	2	8	2.7	4.2	3.7
MB1.2	38	13	0.9	6.8	3.7
MB1.3	88	29	2.8	7.5	4
MB1.4	178	38	4.7	7.9	6.1
MB1.5	619	28	5.8	8.7	7.1
MB2.1	1	7	1.53	6.09	4.6
MB2.2	13	11	2.61	10.02	5.6
MB2.3	183	65	4.35	8.28	6
MB2.4	1334	120	5.11	11.35	8.1
MB2.5	2435	370	10.85	18.82	13.9

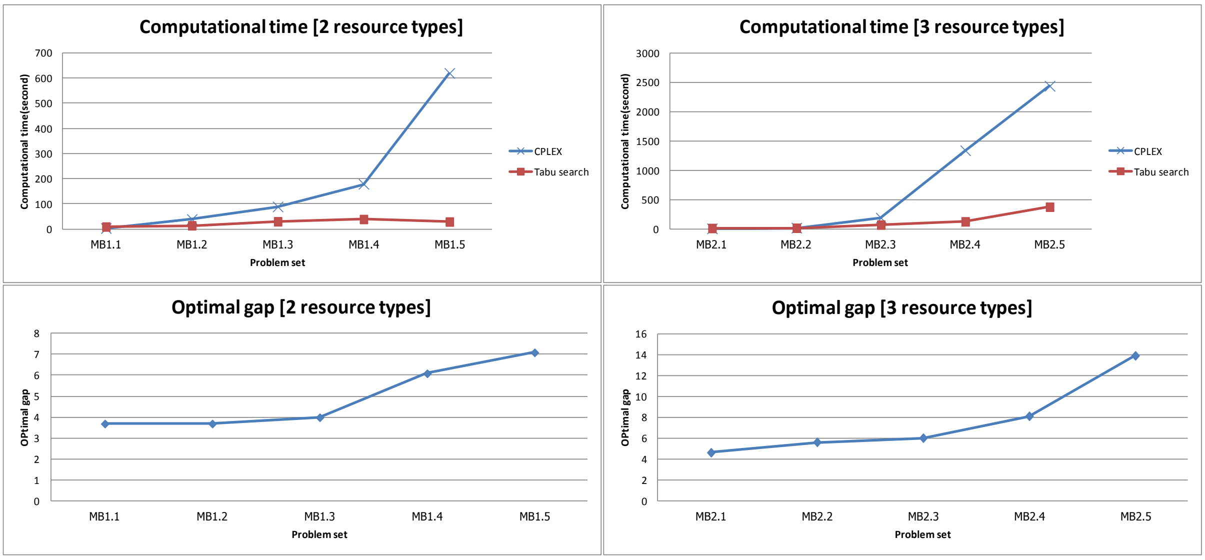

Fig. 5. Computational time and optimal gap of medium size problem [part 2].

The experiment of large size problems

In the large size problem, five problems are generated: L1, L2, L3, L4 and L5. Because finding an optimal solution for this problem set takes a lot of time, the computational time for running CPLEX is limited to 4,000 seconds while time for running MTS is limited to 100 iterations and the number of candidate list is fixed to 32. We know from the result of experiment in medium size problem that the computational time or complexity of the problem extremely increases when the number of resource type increases. In this experiment, the number of resource type is varied from 4 to 12, which is a large number comparing with the previous experiment (2 and 3 resource types), to evaluate the efficiency of the algorithm when being applied to high complexity problems. The computational time and optimal gap are shown in Table 6. The value of optimal gap in this problem set can be less than zero because running time of CPLEX is limited and solutions from CPLEX can be worse than the solutions from Tabu search algorithm. As shown in the Table5, the optimal gap of four out of five problems is less than zero, which means that four out of five solutions from Tabu search algorithm are better than solutions from CPLEX. Figure 6 illustrates the computational time and the best solution found in each time by CPLEX and Tabu search algorithm. Parts of CPLEX line that have no data means that CPLEX cannot find any feasible solution. As can be seen, when the number of resource type increases, time to find a feasible solution by CPLEX increases. In contrast to CPLEX, Tabu search algorithm can provide good feasible solutions in a very short computational time.

Table 6. The results from experiments of large size problems.

Problem set	Tabu time(sec)	Gap
L1	934	4.9
L2	269	-1.1
L3	217	-45.8
L4	203	-29.2
L5	705	-17.3

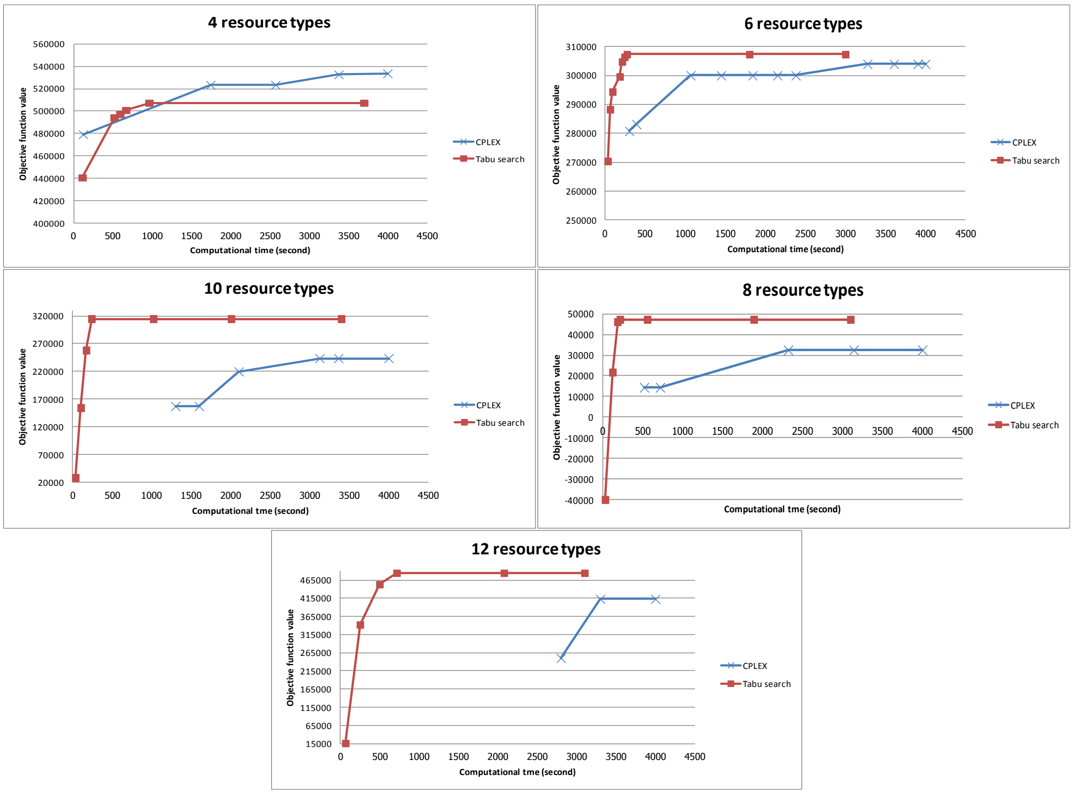

Fig. 6. Computational time and optimal gap of large size problem [part 2].

In summary, the complexity of problems drastically increases when the number of resource type and the number of task that requires joint of multiple resource types increase. Tabu search algorithm performs well in all problem sizes. Many solutions in small size problem are optimal and the average gap for all problems is 2.2%. In the medium size problem, good solutions can be found in a short time comparing with CPLEX (average gap = 5.8%). For the large size problem, the proposed algorithm clearly outperforms CPLEX. Most solutions from Tabu search algorithm are better than solutions from CPLEX and all best solutions can be found quickly comparing with CPLEX.

6. Conclusion

In most allocation and assignment models, only one resource type is considered. If multiple resources types are considered, they fail to consider joint requirement of multiple resource types. In this study, we have considered this aspect in a multi-period multi-site assignment problem (multi-period multi-site assignment problem with joint requirement of multiple resource types). This kind of problem can be found in planning multiple health resources in clinic or hospital network. The mathematical model and heuristic based on Tabu search algorithm was developed. The proposed Tabu search algorithm is comprised of two steps (two-step Tabu search algorithm). The first step aims to allocate resources to site while the second step means to assign resources to task. The computational experiments are done to evaluate the efficiency of the algorithm. Test problems are grouped into three sizes: small, medium and large size problems. The result shows that the developed algorithm provides good solutions in all problem sizes.

Although the algorithm performs well for all test problems, the quality of solution tends to be dropped when the ratio of task that requires more than one resource type increases. Future work should find ways to improve the quality of the solution when this ratio is high. In addition, the developed model does not allow their resources to be rotated, so another subject of future study is to develop model that allow their resources to be rotated to many sites.

Acknowledgements

The portion of this work was presented at 17th International Conference of the International Journal of Industrial Engineering – Theory, Applications, and Practice (IJIE) [46]. We wish to thank Prof. Rein Boondiskulchok and Sakol Suethanapornkul for guidance and feedback on the earlier drafts of this paper and anonymous reviewers for insightful comments.

References

[1] L. R. Foulds and J. M. Wilson, "A variation of the generalized assignment problem arising in the New Zealand dairy industry," *Annals of Operations Research*, vol. 69, no. 0, pp. 105-114, 1997.

[2] S. Hajri-Gabouj, "A fuzzy genetic multiobjective optimization algorithm for a multilevel generalized assignment problem," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 33, no. 2, pp. 214-224, 2003.

[3] G. H. Reynolds, "A shuttle car assignment problem in the mining industry," *Management Science*, vol. 17, no. 9, pp. 652-655, 1971.

[4] H. D. Sherali, E. K. Bish, and X. Zhu, "Airline fleet assignment concepts, models, and algorithms," *European Journal of Operational Research*, vol. 172, no. 1, pp. 1-30, 2006.

[5] S. Bilgin, and M. Azizoglu, "Operation assignment and capacity allocation problem in automated manufacturing systems," *Computers & Industrial Engineering*, vol. 56, no. 2, pp. 662-676, 2009.

[6] A. Corominas, R. Pastor, and E. Rodriguez, "Rotational allocation of tasks to multifunctional workers in a service industry," *International Journal of Production Economics*, vol. 103, no. 1, pp. 3-9, 2006.

[7] D. W. Pentico, "Assignment problems: A golden anniversary survey," *European Journal of Operational Research*, vol. 176, no. 2, pp. 774-793, 2007.

[8] C. Rainwater, J. Geunes, and H. Edwin Romeijn, "The generalized assignment problem with flexible jobs," *Discrete Applied Mathematics*, vol. 157, no. 1, pp. 49-67, 2009.

[9] B. Alidaee, H. Gao, and H. Wang, "A note on task assignment of several problems," *Computers & Industrial Engineering*, vol. 59, no. 4, pp. 1015-1018, 2010.

[10] A. Ravindran and V. Ramaswami, "On the bottleneck assignment problem," *Journal of Optimization Theory and Applications*, vol. 21, no. 4, pp. 451-458, 1977.

[11] K. C. Gilbert and R. B. Hofstra, *Multidimensional Assignment Problems*. Blackwell Publishing Ltd, 1988, pp. 306-321.

[12] L. S. Franz and J. L. Miller, "Scheduling medical residents to rotations: solving the large-scale multiperiod staff assignment problem," *Operations Research*, vol. 41, no. 2, pp. 269-279, 1993.

[13] X. Zhang and J. F. Bard, "A multi-period machine assignment problem," *European Journal of Operational Research*, vol. 170, no. 2, pp. 398-415, 2006.

[14] B. Gavish and H. Pirkul, "Efficient algorithms for solving multiconstraint zero-one knapsack problems to optimality," *Mathematical Programming*, vol. 31, no. 1, pp. 78-105, 1985.

[15] J. B. Mazzola and A. W. Neebe, "Resource-constrained assignment scheduling," *Operations Research*, vol. 34, no. 4, pp. 560-572, 1986.

[16] G. Caron, P. Hansen, and B. Jaumard, "The assignment problem with seniority and job priority constraints," *Operations Research*, vol. 47, no. 3, pp. 449-453, 1999.

[17] J.-H. Zhang, J. Li, and Z.-P. Liu, "Multiple-resource and multiple-depot emergency response problem considering secondary disasters," *Expert Systems with Applications*, vol. 39, no. 12, pp. 11066-11071, 2012.

[18] L. Ozdamar, E. Ekinci, and B. Kucukyazici, "Emergency Logistics Planning in Natural Disasters," *Annals of Operations Research*, vol. 129, no. 1-4, pp. 217-245, 2004.

DOI:10.4186/ej.2014.18.3.83

[19] J.-B. Sheu, "An emergency logistics distribution approach for quick response to urgent relief demand in disasters," *Transportation Research Part E: Logistics and Transportation Review*, vol. 43, no. 6, pp. 687-709, 2007.

[20] G.-H. Tzeng, H.-J. Cheng, and T. D. Huang, "Multi-objective optimal planning for designing relief delivery systems," *Transportation Research Part E: Logistics and Transportation Review*, vol. 43, no. 6, pp. 673-686, 2007.

[21] F. Fiedrich, F. Gehbauer, and U. Rickers, "Optimized resource allocation for emergency response after earthquake disasters," *Safety Science*, vol. 35, no. 1-3, pp. 41-57, 2000.

[22] U. Aickelin and K. A. Dowsland, "An indirect Genetic Algorithm for a nurse-scheduling problem," *Computers & Operations Research*, vol. 31, no. 5, pp. 761-778, 2004.

[23] M. W. Carter and S. D. Lapierre, "Scheduling emergency room physicians," *Health Care Management Science*, vol. 4, no. 4, pp. 347-360, 2001.

[24] V. M. Trivedi and D. M. Warner, "A branch and bound algorithm for optimum allocation of float nurses," *Management Science*, vol. 22, no. 9, pp. 972-981, 1976.

[25] W. J. Gutjahr and M. S. Rauner, "An ACO algorithm for a dynamic regional nurse-scheduling problem in Austria," *Computers & Operations Research*, vol. 34, no. 3, pp. 642-666, 2007.

[26] C. F. F. Costa Filho, D. A. Rivera Rocha, M. G. Fernandes Costa, and W. C. de Albuquerque Pereira, "Using Constraint Satisfaction Problem approach to solve human resource allocation problems in cooperative health services," *Expert Systems with Applications*, vol. 39, no. 1, pp. 385-394, 2012.

[27] A. M. Caunhye, X. Nie, and S. Pokharel, "Optimization models in emergency logistics: A literature review," *Socio-Economic Planning Sciences*, vol. 46, no. 1, pp. 4-13, 2012.

[28] N. Altay, and W. G. Green III, "OR/MS research in disaster operations management," *European Journal of Operational Research*, vol. 175, no. 1, pp. 475-493, 2006.

[29] U. Aickelin and K. A. Dowsland, "Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem," *Journal of Scheduling*, vol. 31, pp. 139-153, 2000.

[30] K. A. Dowsland, "Nurse scheduling with tabu search and strategic oscillation," *European Journal of Operational Research*, vol. 106, no. 2-3, pp. 393-407, 1998.

[31] E. Burke, P. Cowling, P. D. Causmaecker, and G. V. Berghe, "A memetic approach to the nurse rostering problem," *Applied Intelligence*, vol. 15, no. 3, pp. 199-214, 2001.

[32] C.-C. Tsai and S. H. A. Li, "A two-stage modeling with genetic algorithms for the nurse scheduling problem," *Expert Systems with Applications*, vol. 36, no. 5, pp. 9506-9512, 2009.

[33] S. K. Goyal and J. P. Yadav, "Allocation of doctors to health centres in Haryana State of India -- A case study," *The Journal of the Operational Research Society*, vol. 30, no. 5, pp. 427-431, 1979.

[34] A. T. Ernst, H. Jiang, M. Krishnamoorthy, and D. Sier, "Staff scheduling and rostering: A review of applications, methods and models," *European Journal of Operational Research*, vol. 153, no. 1, pp. 3-27, 2004.

[35] B. Toktas, J. W. Yen, and Z. B. Zabinsky, "Addressing capacity uncertainty in resource-constrained assignment problems," *Computers & Operations Research*, vol. 33, no. 3, pp. 724-745, 2006.

[36] F. Glover, "Future paths for integer programming and links to artificial intelligence," *Comput. Oper. Res.*, vol. 13, no. 5, pp. 533-549, 1986.

[37] F. Glover, "Tabu search-part 1," *ORSA Journal on Computing*, vol. 1, pp. 190-206, 1989.

[38] F. Glover, "Tabu search-part 2," *ORSA Journal on Computing*, vol. 2, pp. 4-32, 1990.

[39] A. P. Punnen and Y. P. Aneja, "A tabu search algorithm for the resource-constrained assignment problem," *The Journal of the Operational Research Society*, vol. 46, no. 2, pp. 214-220, 1995.

[40] J. Bard and L. Wan, "The task assignment problem for unrestricted movement between workstation groups," *Journal of Scheduling*, vol. 9, no. 4, pp. 315-341, 2006.

[41] J. Ferland, I. Berrada, I. Nabli, B. Ahiod, P. Michelon, V. Gascon, and É. Gagné, "Generalized assignment type goal programming problem: Application to nurse scheduling," *Journal of Heuristics,* vol. 7, no. 4, pp. 391-413, 2001.

[42] E. Burke, P. Causmaecker, and G. Berghe, "A hybrid tabu search algorithm for the nurse rostering problem," *Simulated Evolution and Learning*, Lecture Notes in Computer Science, pp. 187-194: Springer Berlin Heidelberg, 1999.

[43] K. A. Dowsland and J. M. Thompson, "Solving a Nurse Scheduling Problem with Knapsacks, Networks and Tabu Search," *The Journal of the Operational Research Society*, vol. 51, no. 7, pp. 825-833, 2000.

[44] H. Li and K. Womer, “Scheduling projects with multi-skilled personnel by a hybrid MILP/CP benders decomposition algorithm,” *Journal of Scheduling*, vol. 12, no. 3, pp. 281-298, 2009.

[45] L. Chen, A. Langevin, and D. Riopel, “A tabu search algorithm for the relocation problem in a warehousing system,” *International Journal of Production Economics*, vol. 129, no. 1, pp. 147-156, 2011.

[46] S. Swangnop and P. Chaovalitwongse, “A Tabu search heuristic for multi-period multi-site assignment problem with joint requirement of multiple resource types,” in *Proceedings of 17th International Conference on Industrial Engineering Theory, Applications and Practice (IJIE 2013)*, 2013, pp. 603-611.