Journal of Global Optimization (2005) 31: 17–43

Dynamic Generalized Assignment Problems with Stochastic Demands and Multiple Agent–Task Relationships

KONSTANTIN KOGAN[1], EUGENE KHMELNITSKY[2] and
TOSHIHIDE IBARAKI[3]
[1]*Department of Interdisciplinary Studies–Logistics, Bar Ilan University, Israel*
[2]*Department of Industrial Engineering, Tel-Aviv University, Ramat Aviv, 69978 Tel-Aviv, Israel (e-mail: xmel@eng.tau.ac.il)*
[3]*Department of Applied Mathematics and Physics, Faculty of Engineering, Kyoto University, Japan*

(Received 19 February 2002; accepted in revised form 12 March 2004)

Abstract. The assignment problem is a well-known operations research model. Its various extensions have been applied to the design of distributed computer systems, job assignment in telecommunication networks, and solving diverse location, truck routing and job shop scheduling problems.

This paper focuses on a dynamic generalization of the assignment problem where each task consists of a number of units to be performed by an agent or by a limited number of agents at a time. Demands for the task units are stochastic. Costs are incurred when an agent performs a task or a group of tasks and when there is a surplus or shortage of the task units with respect to their demands. We prove that this stochastic, continuous-time generalized assignment problem is strongly *NP*-hard, and reduce it to a number of classical, deterministic assignment problems stated at discrete time points. On this basis, a pseudo-polynomial time combinatorial algorithm is derived to approximate the solution, which converges to the global optimum as the distance between the consecutive time points decreases. Lower bound and complexity estimates for solving the problem and its special cases are found.

1. Introduction

The assignment problem is a classic operations research model, which deals with the optimal allocation of a number of tasks to a group of agents in such a way that each agent is given one task to perform and each task is assigned to one agent only (Hillier and Lieberman 1995). Due to the wide range of applications in modeling real-life events, it has gained much attention from the operations research community. The assignment problem and its various extensions were applied to the design of distributed computer systems (Gavish and Pirkul 1986) and solving diverse location problems (Ross and Soland 1977). It was also used as a model for truck routing problems (Murphy 1986), job assignment in computer networks (Balachandran 1976), and is useful for problems in the areas of job shop scheduling, cargo loading, ships and warehouse design (Gavish and Pirkul 1991).

Major extensions of the assignment problem are related to the types of the resource constraints which define different resource consumption modes, and the agent–task relationships by allowing an agent to perform multiple tasks and/or a task to be performed by a number of agents subject to the resource availability. Specifically, an extension of the assignment problem that considers side constraints on total resource consumption was addressed by Mazzola and Neebe (1986). A wider extension, where multiple tasks may be assigned to an agent, is usually referred to either as the generalized assignment problem or, if multiple resources are involved, as the multi-resource generalized assignment problem. These problems were extensively studied by Ross and Soland (1975, 1977), Nauss (1976), Martello and Toth (1977), Klastorin (1979), Balas and Zemel (1980), and Yagiura *et al.* (1999).

An evident restriction of the above static extensions of the assignment problem is that the sequence in which an agent performs its tasks cannot be taken into account. This sequence is essential when each task consists of a batch of units that are processed to meet a given time-dependent demand profile (i.e., processing ahead of the demand as well as backlogging incur earliness and tardiness costs, respectively). Such a generalization was introduced by Kogan *et al.* (1997) and referred to as the dynamic generalized assignment problem (DGAP). The DGAP is formulated as a continuous-time model, where each task consists of a number of units to be performed with a fixed rate under the single resource constraint that prohibits an agent to carry out more than one task at a time. The units of a task are allowed to be performed simultaneously by different agents with no limits. The objective is to minimize the costs of dynamic earliness and tardiness as well as the cost incurred when performing the tasks by agents. The focus of the paper is on a new dynamic extension of the assignment problem. Although the authors show the unimodularity of the dual problem, it is not exploited. Instead, a standard gradient-based solution algorithm with no control over accuracy and convergence is employed to illustrate the approach. Such a method becomes useless when the problem complicates. Therefore, the extension restricts to deterministic demands and to only agent-related constraints.

This paper proceeds with extending the DGAP to cope with stochastic environment and multiple agent–task relationships limiting, at every time point, both the number of agents performing a task and the number of tasks that can be performed by an agent. This is imposed with the aid of the total unimodularity of the resource constraint matrix. Such two types of resource constraints ensure that the problems under consideration are still in the framework of the assignment problem, but in the form of many-to-many agent–task relationship.

Thus, every agent in our generalized problem deals with stochastic demands encountered along the planning horizon, and is allowed to carry out a limited number of tasks at a time within its time-dependent capacity. Every task is processed by a limited number of agents specified by its time-dependent availability. We assume that the realization of stochastic demand and the number of the accomplished tasks are observable only by the end of a time period. Thus, given demand univariate density and autocorrelation functions for each task type, the determination of which type of tasks to perform and when to perform must be made under these uncertain conditions, before production commences. The well-known application of such a generalization is found in the stochastic environment of the flow shop scheduling of parallel workstations and flexible manufacturing cells, which can process a limited number of products at a time while a limited pool of material-holding devices is available. Since continuous review of product inventories is not always possible in the flow shop, inventory update is available only at the end of a given time period.

In this paper we take advantage of the unimodularity property to develop a new solution method which enables to control both the algorithm convergence and the solution accuracy. Thus, the contribution of the paper is twofold:

(i) the mathematical formulation of the problem includes general stohastic demands and agent–task relationships;

(ii) a new combinatorial algorithm which converges to the global optimum is developed.

In Section 2, some applications of the dynamic assignment problem are presented. The stochastic, dynamic generalized assignment problem (SDGAP) is formulated in Section 3. Basic properties of the continuous-time SDGAP are studied in Sections 4 and 5 and of the discrete-time SDGAP in Section 6. This study shows that though the properties of the optimal solutions derived for the DGAP (Kogan, *et al.* 1997) are no longer valid, the dual SDGAP can be reduced to a number of the classical deterministic assignment problems. Based on this result, a time-decomposition needle algorithm is proposed in Section 7 to approximate numerically the solution for various versions of the SDGAP. Section 8 discusses complexity of the algorithm, which is essentially polynomial time for a large class of problems, if computed on a fixed time grid, and its asymptotic convergence to the globally optimal solution. Section 9 presents computational experiments, while Section 10 summarizes the results.

2. Examples

In this section we consider two examples, which present two classes of problems where the SDGAP is straightforwardly applicable. Both classes of problems are characterized by stochastic demands, limited task–agent capacities, and periodic review of completed tasks. The first class is related to the communication problems where channels, computers, or computer memory have to be allocated to process the incoming calls, jobs, or requests. The goal is to minimize delays (queues of awaiting jobs) and processing costs. The second class is related to dynamic inventory management. Inventory management is a general philosophy, which aims at making the material flow between different sections smooth. The ultimate goal is to produce only the necessary products in a required amount without delay and unnecessary inventories.

2.1. DYNAMIC JOB ALLOCATION IN A COMPUTER NETWORK

A computer network is an interconnected group of independent computer systems that communicate with each other and share resources such as programs, data, hardware, and software. As a primary goal, some networks link functionally different and specialized computing facilities to avoid duplicating unique hardware configurations. Global job scheduling in such environments is not normally a difficult problem because most programs are designed for only one of these specialized systems and the network distributes over a large area the unique capabilities of each node. However, in a network of functionally similar nodes, the allocation of jobs can be a major problem. The nodes may be local or distributed over a wide geographical area. An example of this type of design is a set of cooperating, general purpose computers in which each machine (*agent*) can solve a number of the problems (*tasks*) submitted to the network, but where particular machines hold a competitive advantage over other computers on a specialized subset of the jobs. This implies that each job (*task*) can be performed by a limited number of machines (*agents*). Thus, given dynamic stochastic demands, the allocation of jobs is straightforwardly formulated as SDGAP. Moreover, load sharing between machines is common in the network design. Therefore, a program initiated at a node that is temporarily overloaded can be sent without delays to a temporarily underloaded node with similar functional capabilities. This implies that the tasks can be preempted at no cost and negligible setup time as assumed in the SDGAP.

2.2. SCHEDULING PRE-MEDIA PRINTING OPERATION IN A FLEXIBLE PRODUCTION SYSTEM

The key tasks of a pre-media printing operation are to receive photographs/ images and text material; scan images; combine these elements into a mail order catalog, which provides proper color and layout; and then generate the computer files and/or media which will enable printing presses to actually manufacture the catalogs. All tasks are performed with a large set of pages on one or a few ample computer workstations. Once a task is completed for a page (no matter how many pages are assigned to this task), the next task can be initiated for this page. Therefore, the operations can be considered as parallel (start-to-start relationship) and easily preemptive. Each key task is best handled by a number of workers (agents). However, workers are specialized in a limited number of tasks and may simultaneously perform them on computer workstations. Furthermore, negligible setup time is needed for workers to change from one task/operation to another (e.g., task setups, station-to-station walk times). Finally, pre-media operations suffer from high demand variability as well as extremely tight flow-time performance requirements. Therefore, scheduling the key tasks for the incoming materials over the workers, to avoid early completion of some pages at the cost of expensive delays arising from other pages, is of great importance. A simplified version of this example is studied in detail in Section 9.

3. Formulation of the SDGAP

Our extension of the DGAP deals with the allocation of a set of tasks J to a group of agents I over planning horizon T. Each agent can process a limited number of tasks and every task can be assigned to a limited number of agents at time t:

$$\sum_{j\in J} y_{ij}(t)\leqslant\beta_i(t),\quad i\in I \text{ (agent resource constraint)}, \tag{1}$$

$$\sum_{i\in I} y_{ij}(t)\leqslant\chi_j(t),\quad j\in J \text{ (task resource constraint)}, \tag{2}$$

where $\beta_i(t)$, $\chi_j(t)$ are the integer capacities of the agent and task resources available at time t, respectively, and $y_{ij}(t)$ is a decision variable:

$$y_{ij}(t)=\begin{cases}1, \text{ if agent } i \text{ performs task } j \text{ at time } t,\\ 0, \text{otherwise,}\end{cases}\qquad i\in I,\ j\in J, \tag{3}$$

integrable on $[0,T]$.

Every task is induced according to its stochastic demand $d_j(t)$ for which univariate density and autocorrelation functions are known. The dynamics of such an agent–task system is described by the total flow of task units performed by the agents at time t:

$$\dot{X_j}(t)=\sum_{i\in I} U_{ij}y_{ij}(t)-d_j(t),\quad j\in J, \tag{4}$$

where $X_j(t)$ is a continuous and differentiable function which represents the cumulative number of task j units at time t, over-performed (surplus) by time t if $X_j(t) > 0$ and under-performed if $X_j(t) < 0$ (shortage); U_{ij} is the intensity with which agent i is capable to perform task j. A natural boundary condition for these differential equations is that in the beginning of the planning horizon [0, T] the task levels are known:

$$X_j(0) = X_j^0, \quad j \in J. \tag{5}$$

For the system we are investigating, it is assumed that the realization of stochastic demand and the level of the accomplished tasks $X_j(t)$are observable only by the end of the planning horizon, T. Therefore, the decision has to be made under these uncertain conditions before the dynamic assignment starts.

The objective of the SDGAP is to minimize the expected total cost of shortages and surpluses as well as the cost of performing all tasks by the agents:

$$Z = E\left[\int_0^T \left(\sum_{i\in I}\sum_{j\in J} C_u(y_{ij}(t)) + \sum_{j\in J} C_x(X_j(t))\right) \mathrm{d}t\right] \to \min, \tag{6}$$

subject to constraints (1)–(5).

In this paper we assume, similar to the classical assignment problems, the linear cost functions for performing task j by agent i, i.e. $C_u(y_{ij}(t)) = c_{ij}^y y_{ij}(t)$ and the quadratic penalties for over- as well as under-performed tasks, $C_x(X_j(t)) = c_j^x X_j(t)^2$. The approach developed in this paper can be extended to apply to various symmetric (not only quadratic) penalty functions. Although such an extension is straightforward, it unavoidably leads to awkward mathematical expressions obscuring the main results and, therefore, it is not considered here.

Note that there is an asymmetry in agent–task dynamics. We study quite straightforward extension of the assignment problem which assumes that each type of task consists of a number of works to be completed with respect to a given demand. This is described by the task dynamics (4). Applications of this extension are well known and discussed in the paper. The model we suggest could be extended to symmetrically incorporate the agent dynamics. This would imply that we have different types of agents. The number of agents of each type depends on the number of task works, agents of this type are assigned to perform. This, in turn, would imply that the number of task works assigned to a type of agents influences the intensities with which each task is performed by the agents of this type. We do not consider such an extension of the assignment problem because of two major reasons. First of all, as it is shown in the paper, even the extension of the assignment problem to incorporate only task dynamics results in an NP-hard problem. Consequently, construction of an efficient algorithm for only this extension is quite challenging. Secondly, applications of the agent dynamics are found rarely in OR practice as compared to the task dynamics.

4. Equivalent Deterministic Assignment Problem and its Dual Formulation

Generally, a random process is completely determined by its multivariate distribution function. However, in practice, this function is rarely available. Therefore various types of series expansions and quasi-deterministic forms are commonly employed to represent the process (Van Trees 1968). In such a case, the process is described by a number of deterministic functions with randomized parameters. Widely used trigonometric series is an example of random process representation by deterministic trigonometric functions with random amplitude. This representation is very useful if the stochastic process is periodic and stationary, because the random amplitudes become uncorrelated. If this is not the case, more general orthogonal representations are utilized.

The well-known Karhunen–Loeve (KL) type of series is employed here to illustrate the approach for general random demands which are not necessarily stationary and periodic. Algazi and Sakrison (1969) showed that KL expansion is optimal not only in terms of minimizing mean-square error between the demand data and its truncated representation, but also minimizes a number of modes to describe the data for a given error. The optimality of KL reduces the amount of information required to represent statistically dependent data to a minimum. This crucial feature explains the wide usage of KL. Henceforth, a KL representation of demand is adopted in the paper to study the SDGAP and derive an equivalent deterministic problem.

If accuracy of such a representation is specified, then stochastic demands for the tasks can be expanded in a finite element KL series. If, however, the stochastic process is rational, then it can be exactly represented by a K-element KL expansion (Youla 1957). In this paper we limit our consideration to only rational demand processes:

$$d_j(t) = \sum_{k \in K} \xi_{jk} \varphi_{jk}(t), \quad j \in J, \ \ t \in [0, T], \tag{7}$$

where $\varphi_{jk}, k \in K$, are orthonormal deterministic sample functions and ξ_{jk} are independent random coefficients with expected values m_{jk}, standard deviations D_{jk}, and density functions $f_{jk}(\xi)$.

PROPOSITION 1. *Let the demand be presented by* (7). *Then, the problem* (1)–(6) *is equivalent to the following deterministic DGAP*:

$$Z = \int_0^T \left(\sum_{i \in I} \sum_{j \in J} c_{ij}^y y_{ij}(t) + \sum_{j \in J} c_j^x \left(\widehat{X}_j(t) - \sum_{k \in K} m_{jk} \hat{\varphi}_{jk}(t) \right)^2 \right) \mathrm{d}t \to \min \tag{8}$$

subject to constraints (1)–(3) *and*

$$\dot{\hat{X}}_j(t) = \sum_{i\in I} U_{ij} y_{ij}(t), \quad j \in J, \tag{9}$$

$$\widehat{X}_j(0) = X_j^0, \quad j \in J, \tag{10}$$

where

$$\hat{\varphi}_{jk}(t) = \int_0^t \varphi_{jk}(\tau)\,\mathrm{d}t, \quad j \in J, \;\; k \in K. \tag{11}$$

Proof. Introduce new variables $\hat{X}_j(t)$ as defined in (9) into the differential equation (4) with the demands (7):

$$\dot{X}_j(t) = \dot{\widehat{X}}_j(t) - \sum_{k\in K} \xi_{jk}\varphi_{jk}(t). \tag{12}$$

Next, by integrating equation (12) and introducing new functions $\hat{\varphi}_{jk}(t)$ as defined in (11) we obtain:

$$X_j(t) = \widehat{X}_j(t) - \sum_{k\in K} \xi_{jk}\hat{\varphi}_{jk}(t). \tag{13}$$

Equation (13) is then substituted into the objective (6) for which the expected value is calculated as follows:

$$\begin{aligned} Z &= E\left[\int_0^T \left(\sum_{i\in I}\sum_{j\in J} c_{ij}^y y_{ij}(t) + \sum_{j\in J} c_j^x \left(\widehat{X}_j(t) - \sum_{k\in K} \xi_{jk}\hat{\varphi}_{jk}(t)\right)^2\right) \mathrm{d}t\right] \\ &= \int_0^T \left(\sum_{i\in I}\sum_{j\in J} c_{ij}^y y_{ij}(t) + \sum_{j\in J} c_j^x \int_{\{\xi_{jk}\}} \left(\widehat{X}_j(t) - \sum_{k\in K} \xi_{jk}\hat{\varphi}_{jk}(t)\right)^2 \prod_{k\in K} f_{jk}(\xi_{jk})\,\mathrm{d}\xi_{jk}\right) \mathrm{d}t. \end{aligned} \tag{14}$$

The new objective now can be simplified by opening parentheses, replacing integrals of the independent random coefficients multiplied by their density functions with the corresponding moments. The resulting terms are again collected into the following final functional:

$$\begin{aligned} &\int_0^T \left(\sum_{i\in I}\sum_{j\in J} c_{ij}^y y_{ij}(t) + \sum_{j\in I} c_j^x \left(\hat{X}_j(t) - \sum_{k\in K} m_{jk}\hat{\varphi}_{jk}(t)\right)^2\right. \\ &\left. + \sum_{j\in I} c_j^x \left(\sum_{k\in K} D_{jk}^2 \hat{\varphi}_{jk}^2(t) + 2\sum_{k'\neq k''} m_{jk'}m_{jk''}\hat{\varphi}_{jk'}(t)\hat{\varphi}_{jk''}(t)\right)\right) \mathrm{d}t. \end{aligned}$$

Taking into account that

$$\int_0^T \sum_{j\in J} c_j^x \left(\sum_{k\in K} D_{jk}^2 \hat{\varphi}_{jk}^2(t) + 2\sum_{k'\neq k''} m_{jk'}m_{jk''}\hat{\varphi}_{jk'}(t)\hat{\varphi}_{jk''}(t)\right) \mathrm{d}t$$

is a given constant which does not influence the optimization, we finally obtain the objective function (8) subject to constraints (1)–(3), (9) and (10). □

4.1. DUAL FORMULATION OF THE SDGAP

Since the equivalent deterministic DGAP model meets the canonical form of an optimal control problem (Hartl *et al.* 1995, Dubovitsky and Milyutin 1981), where $X_j(t)$ are state variables and $y_{ij}(t)$ are decision variables, the maximum principle can be applied to develop a dual formulation of the problem. The maximum principle asserts that there exist continuous dual functions $\psi_j(t)$, so that the following dual differential equations hold:

$$\dot{\psi}_j(t) = 2c_j^x\left(\widehat{X}_j(t) - \sum_k m_{jk}\widehat{\varphi}_{jk}(t)\right), \quad \psi_j(T) = 0, \tag{15}$$

and the optimal $y_{ij}(t)$ is achieved by maximizing, for each t, the dual objective function (Hamiltonian function):

$$\begin{aligned} H(t) = &-\sum_{i\in I}\sum_{j\in J} c_{ij}^y y_{ij}(t) - \sum_{j\in J} c_j^x\left(\hat{X}_j(t) - \sum_{k\in K} m_{jk}\hat{\varphi}_{jk}(t)\right)^2 \\ &+ \sum_{j\in J}\psi_j(t)\sum_{i\in I} U_{ij}y_{ij}(t), \end{aligned} \tag{16}$$

subject to constraints (1)–(3).

5. Strong *NP*-hardness and Unimodality of the Equivalent DGAP

In this section we study two basic features of the equivalent DGAP: its complexity and unimodality. To prove *NP*-hardness, we consider a simplified version of the equivalent DGAP. Optimal solutions of this version are characterized by a no-switching policy, i.e. each agent performs only one task over the entire planning horizon, as stated in Lemma 1. Since the no-switching property holds for only a special version of the equivalent DGAP, the proof of the lemma is relocated to Appendix.

LEMMA 1. *On an optimal solution each agent performs no more than one task over the entire planning horizon if*

- *agent capacities are minimal*: $\beta_i(t) = \beta_i = 1,\ i \in I$ *and task capacities are unlimited:* $\chi_j(t) = \chi_j \geqslant |I|, j \in J$;
- *the task intensities are not identical*: $U_{ij} \neq U_{i'j'},\ i,\ i' \in I, j, j' \in J$;
- *all demands are static, deterministic, equal* $d_j(t) = d, j \in J$; *and large*

$$d > \frac{U_{ij}\sum_{i'\in I} U_{i'j}}{|U_{ij} - U_{ij'}|}, \quad \forall i \in I,\ j,j' \in J,$$

- *there are no initial task units*: $\hat{X}_j(0) = X_j^0 = 0, j \in J$;
- *there are no processing costs*: $c_{ij}^y = 0,\ i \in I, j \in J$;
- *all penalties for shortages and surpluses are equal*: $c_j^x = c, j \in J$.

Based on Lemma 1, Theorem 1 shows that the continuous-time equivalent DGAP is NP-hard.

THEOREM 1. *The equivalent DGAP is strongly NP-hard even* if

- *agent capacities are minimal*: $\beta_i(t) = \beta_i = 1,\ i \in I$ *and task capacities are unlimited*: $\chi_j(t) = \chi_j \geqslant |I|,\ j \in J$;
- *the task intensities are not identical*: $U_{ij} \neq U_{i'j'},\ i,\ i' \in I,\ j,\ j' \in J$.
- *all demands are static, deterministic, equal* $d_j(t) = d,\ j \in J$ *and large* $d > \frac{U_{ij} \sum_{i' \in I} U_{i'j}}{|U_{ij} - U_{ij'}|},\quad \forall i \in I,\ j,\ j' \in J,$
- *there are no initial task units*: $\widehat{X}_j(0) = X_j^0 = 0,\ j \in J$;
- *there are no processing costs*: $c_{ij}^y = 0,\ i \in I,\ j \in J$;
- *all penalties for shortages and surpluses are equal*: $c_j^x = c,\ j \in J$.

Proof. Given the conditions of Theorem 1, let us integrate the differential equation (9). Then by taking into account Lemma 1, we obtain:

$$\widehat{X}_j(t) - \widehat{X}_j(0) = \int_0^t \sum_{i \in I} U_{ij} y_{ij}(\tau)\ \mathrm{d}\tau \Rightarrow \widehat{X}_j(t) = t \sum_{i \in I} U_{ij} y_{ij}(0).$$

The expression obtained for $\widehat{X}_j(t)$ can now be substituted into the objective. The new simplified version of our assignment problem then takes the following form:

$$\sum_{j \in J} c \left(\sum_{i \in I} U_{ij} y_{ij}(0) - d \right)^2 \frac{T^3}{3} \to \min,$$

$$\sum_{j \in J} y_{ij}(0) \leqslant 1, \quad i \in I,$$

$$y_{ij}(0) = \{0, 1\}, \quad i \in I,\ j \in J.$$

Next, taking into account that the constants c and T do not influence the optimization and replacing $y_{ij}(0)$ with y_{ij}, we obtain the following assignment problem with a non-separable quadratic objective function:

$$\sum_{j \in J} \left(\sum_{i \in I} U_{ij} y_{ij} - d \right)^2 \to \min, \tag{17}$$

$$\sum_{j \in J} y_{ij} \leqslant 1, \quad i \in I, \tag{18}$$

$$y_{ij} \in \{0, 1\}, \quad i \in I,\ j \in J. \tag{19}$$

Any solution to this instance is a partition of the set of agents I into a number of disjoint subsets of agents A_j. In other words, by constraints (18) and

(19), all agents $i \in A_j$ process task j and no other agent processes the same task j. The objective is to minimize the total sum of squared differences between the performance rate of every subset of agents and the demand.

To prove that the formulated problem is *NP*-hard, we now present the minimum sum of squares (MSS) problem, which is proven to be *NP*-complete in the strong sense by transformation from 3-partition (Garey and Johnson 1991).

Problem MSS

Instance: Finite set $A = \{a_1, a_2, \ldots, a_n\}$, size s: $A \to Z^+$, positive integers $B \leqslant n$ and G.

Question: Can A be partitioned into B disjoint sets $A_1, A_2, \ldots, A_B$ such that

$$\sum_{l=1}^{B} \left(\sum_{a \in A_l} s(a) \right)^2 \leqslant G. \tag{20}$$

Let us reduce MSS into the above special case of the equivalent DGAP (17)–(19). Define $I = \{1, 2, \ldots, n\}$ and $J = \{1, 2, \ldots, B\}$. We also denote by A_j the subset of agents which process task j, i.e. the agents for which

$$y_{ij} = \begin{cases} 1 & \forall i \in A_j, \\ 0 & \forall i \notin A_j. \end{cases}$$

Next, by letting $U_{ij} = s(a_i)$, $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, B$ the MSS can be reformulated as follows:

$$\sum_{j=1}^{B} \left(\sum_{i \in A_j} U_{ij} \right)^2 \leqslant G.$$

Note that, by definition, $\sum_{j=1}^{B} \sum_{i \in A_j} U_{ij} = \sum_{i=1}^{n} s(a_i)$.

Finally by adding to the both sides of the last inequality the term:

$$\sum_{j=1}^{B} d^2 - 2d \sum_{j=1}^{B} \sum_{i \in A_j} U_{ij},$$

and introducing the new constant $G' = G + \sum_{j=1}^{B} d^2 - 2d \sum_{j=1}^{B} \sum_{i \in A_j} U_{ij}$ we obtain:

$$\sum_{j=1}^{B} \left(\sum_{i \in A_j} U_{ij} - d \right)^2 \leqslant G'. \tag{21}$$

By setting the demands for all tasks to be very large, and considering that there is no cost for processing tasks and the penalties for surpluses and shortages are equal, we can assume without loss of generality that all B tasks are processed in the optimal solution of the problem (17)–(19).

Consequently, by comparing inequality (21) and the objective (17), one can easily observe that the solution of (17)–(19) solves the original instance of MSS. This proves that the equivalent DGAP (17)–(19) is strongly *NP*-hard. □

Now let us define the relaxation of the equivalent DGAP by allowing decision variables to be continuous:

$$0 \leqslant y_{ij}(t) \leqslant 1, \tag{22}$$

rather than binary, $y_{ij}(t) \in \{0, 1\}$, as imposed by constraint (3). We wish to show that if the optimal solution $y_{ij}(t)$ of the relaxed DGAP satisfies the *integrality property*, i.e. $y_{ij}(t) \in \{0, 1\}$, $i \in I$, $j \in J$, $t \in [0, T]$, then an effective time-decomposition approximation algorithm can be developed to solve the original, *NP*-hard equivalent DGAP. The following theorem formalizes the fact that the relaxed DGAP possesses the integrality property.

THEOREM 2 (the integrality property). *The relaxed equivalent DGAP with integer capacities $\beta_i(t)$ and $\chi_j(t)$ has a binary optimal solution.*

Proof. To prove this theorem, we turn to the dual formulation of the equivalent DGAP (15) and (16). Note that the relaxed DGAP and its deterministic equivalent contain only linear constraints, while the objective is a sum of convex functions. Thus, the relaxed DGAP is unimodal. This means that a solution for the dual problem (i.e., $y_{ij}(t)$, which maximize the Hamiltonian (16)), is the globally optimal solution of the primal problem. Consequently, if the dual problem possesses the integrality property, i.e., it always has an optimal binary solution, then this binary solution is also optimal for the primal problem. Therefore, the proof of Theorem 2 is reduced to the proof that the dual problem with the relaxed binary constraint possesses the integrality property.

Let us combine only decision-dependent terms in the Hamiltonian (16). Then the dual problem takes the following form:

$$\sum_{i \in I} \sum_{j \in J} (U_{ij} \psi_j(t) - c_{ij}^{y}) y_{ij}(t) \rightarrow \max, \tag{23}$$

$$\sum_{j \in J} y_{ij}(t) \leqslant \beta_i(t), \quad i \in I, \tag{24}$$

$$\sum_{i \in I} y_{ij}(t) \leqslant \chi_j(t), \quad j \in J, \tag{25}$$

$$y_{ij}(t) \in [0, 1], \quad i \in I, \ j \in J. \tag{26}$$

According to the dual formulation, the objective (23) is maximized at every point of time t, independently, with $y_{ij}(t)$ being variables and $\psi_j(t)$ being parameters. By setting $c_{ij} = c^y_{ij} - U_{ij}\psi_j(t)$, one can immediately observe that the resulting problem (23)–(26) is the static assignment problem with relaxed 0–1 constraints and multiple agent–task relationships. That is, the resource constraints (24) and (25) are presented by a totally unimodular matrix. At each t, such problem always has an integer 0–1 optimal solution if the capacities are integer. □

Theorem 2 proves that the relaxed dual problem has optimal integer solutions for arbitrary dual variables $\psi_j(t)$. However, $\psi_j(t)$ which satisfies the dual equation (15) is unknown. Therefore, solving the dual problem (16) for a $\psi_j(t)$ in polynomial time does not reduce the complexity of the primal problem proved in Theorem 1. In the following, we suggest an algorithm to calculate such $\psi_j(t)$ for which both (15) and (16) hold, and thus the solution of the dual problem becomes the optimal solution of the primal.

6. The Discretized Equivalent DGAP

Prior to suggesting a pseudo-polynomial time numerical method, which operates on a fixed time grid, it is important to determine whether the discrete version of our problem can be exactly solved in polynomial time or not.

To present a discrete version of the equivalent DGAP, we consider Δ-grid which is constructed by $R+1$ equally distributed mesh points t_r, such that:

$$t_0 = 0, \quad t_{r+1} - t_r = \Delta, \quad r = 0, 1, \ldots, R-1, \quad t_R = T.$$

Then, the agent–task constraints take the following form:

$$\sum_{j\in J} y_{ij}(t_r) \leqslant \beta_i(t_r), \quad i \in I, \tag{27}$$

$$\sum_{i\in I} y_{ij}(t_r) \leqslant \chi_j(t_r), \quad j \in J, \tag{28}$$

$$y_{ij}(t_r) = \begin{cases} 1, \text{ if agent } i \text{ performs task } j \text{ at time } t_r, \\ 0, \text{ otherwise,} \end{cases} \quad i \in I, \; j \in J. \tag{29}$$

The differential equation is replaced with the difference equation:

$$\widehat{X}_j(t_{r+1}) - \widehat{X}_j(t_r) = \Delta \sum_{i\in I} U_{ij} y_{ij}(t_r), \quad \widehat{X}_j(t_0) = X^0_j, \quad j \in J. \tag{30}$$

The objective is

$$Z = \Delta \sum_r \left(\sum_{i\in I} \sum_{j\in J} c^y_{ij} y_{ij}(t_r) + \sum_{j\in J} c^x_j \left(\widehat{X}_j(t_r) - \sum_{k\in K} m_{jk} \hat{\varphi}_{jk}(t_r) \right)^2 \right) \to \min. \tag{31}$$

THEOREM 3. *The discretized equivalent DGAP is strongly NP-hard even if*

- *there is only one discrete time interval* $[0,T]$, *i.e., the distance between the two consecutive time points of the grid is* $\Delta = T$;
- *agent capacities are minimal*: $\beta_i(t) = \beta_i = 1$, $i \in I$, *and task capacities are unlimited*: $\chi_j(t) = \chi_j \geqslant |I|$, $j \in J$;
- *all demands are static, deterministic, very large and equal*: $d_j(t) = d$, $j \in J$;
- *there are no initial task units*: $X_j(0) = X_j^0 = 0, j \in J$;
- *there are no processing costs*: $c_{ij}^y = 0, i \in I, j \in J$;
- *all penalties for shortages and surpluses are equal*: $c_j^x = c$, $j \in J$.

Proof. Let us consider the difference equations describing a single time step $r = 0, 1$ in the discrete formulation of the equivalent DGAP with constant demands:

$$\widehat{X}_j(T) - \widehat{X}_j(0) = \Delta \sum_{i \in I} U_{ij} y_{ij}(0) \Rightarrow \widehat{X}_j(T) = T \sum_{i \in I} U_{ij} y_{ij}(0).$$

The expression obtained for $\widehat{X}_j(T)$ can now be substituted into the objective. The new simplified version of our assignment problem then takes the following form:

$$\sum_{j \in J} c \left(\sum_{i \in I} U_{ij} y_{ij}(0) - d \right)^2 T^3 \rightarrow \min,$$

$$\sum_{j \in J} y_{ij}(0) \leqslant 1, \quad i \in I,$$

$$y_{ij}(0) = \{0, 1\}, \quad i \in I, \ j \in J.$$

The rest of the proof is identical to that presented in Theorem 1. □

7. Time-decomposition Needle Algorithm

Time-decomposition methods have been proved to be effective for solving complex dynamic problems (Sousa and Pereira 1992, Khmelnitsky *et al.* 1995). The main idea of the proposed time-decomposition algorithm is to solve iteratively a number of the dual equivalent DGAP formulated on a Δ-grid with $R + 1$ equally distributed mesh points t_r.

At every iteration, a point of the grid is searched where the dual problem provides the maximal improvement in the objective value. At this point, a currently optimal set of unit decision variables is set for the minimum time interval Δ. Such decisions $y_{ij}(t) = 1$, $t = [t_r, t_{r+1}]$ of the minimum duration Δ are further referred to as needles. The procedure terminates when no improvement is attainable over the planning horizon.

NEEDLE ALGORITHM

Input: U_{ij}, c^y_{ij}, c^x_j, $\beta_i(t_r)$, $\chi_j(t_r)$, X^0_j, m_{jk}, $\hat{\varphi}_{jk}(t_r)$, T, Δ for $i \in I$, $j \in J$, $k \in K$, $r = 0, \ldots, R$.

Output: $y_{ij}(t_r)$, $\widehat{X}_{ij}(t_r)$ for $i \in I, j \in J$, $r = 0, \ldots, R$.

Step 1. Choose a feasible solution for the problem (27)–(31), e.g., $y^0_{ij}(t_r) = 0$, $i \in I, j \in J$, $r = 0, \ldots, R$. Set the objective value Z at infinity.

Step 2. Integrate (from left to right) the primal difference equation (30) with its initial condition and given $y^0_{ij}(t_r)$ as the initial-value problem to determine task levels $\widehat{X}_j(t_r), j \in J$, $r = 1, \ldots, R$.

Step 3. For the obtained task levels and decision variables calculate the objective function (31). If it is improved, then go to the next step. Otherwise halt; the solution has been found.

Step 4. Integrate (from right to left) the dual difference equation

$$\psi_j(t_{r-1}) = \psi_j(t_r) - 2\Delta c^x_j \left(\widehat{X}_j(t_r) - \sum_k m_{jk} \hat{\varphi}_{jk}(t_r) \right), \quad \psi_j(t_R) = 0 \tag{32}$$

with given task units $\widehat{X}_j(t_r)$ as the terminal-value problem to find dual variables $\psi_j(t_r), j \in J$, $r = 0, \ldots, R-1$.

Step 5. For the obtained dual and decision variables calculate the decision-dependent term of the Hamiltonian $H_y(t_r) = \sum_{i \in I} \sum_{j \in J} y^0_{ij}(t_r) \times (U_{ij}\psi_j(t_r) - c^y_{ij})$, $r = 0, \ldots, R$.

Step 6. At each point t_r of Δ-grid, solve the dual equivalent DGAP as the corresponding static assignment problem with the objective function

$$H^*_y(t_r) = \max_{y_{ij}(t_r)} H_y(t_r) = \max_{y_{ij}(t_r)} \sum_{i \in I} \sum_{j \in J} y_{ij}(t_r)(U_{ij}\psi_j(t_r) - c^y_{ij}),$$

s.t. constraints (27)–(29). Denote the solution of this problem at time t_r as $y^*_{ij}(t_r)$.

Step 7. Find the point of time r^* where the variation of the Hamiltonian is maximum: $r^* = \arg\max_r \delta H_y(t_r)$, where $\delta H_y(t_r) = H^*_y(t_r) - H_y(t_r)$. Set needles of width Δ as follows:

$$y_{ij}(t_r) = \begin{cases} y^*_{ij}(t_r), & \text{if } r = r^*, \\ y^0_{ij}(t_r), & \text{otherwise}, \end{cases} \qquad i \in I, \; j \in J.$$

Step 8. Set $y^0_{ij}(t_r) = y_{ij}(t_r)$, $i \in I, j \in J$, $r = 0, \ldots, R$, and return to Step 2.

8. Needle Algorithm Complexity

We start with the convergence of the proposed algorithm to the unique optimal solution (see Theorem 2), and then proceed to studying its complexity for three important practical SDGAP cases.

PROPOSITION 2 (asymptotic convergence). *The Needle algorithm performed on a Δ-grid converges to the optimal solution of the continuous-time equivalent DGAP as $\Delta \to 0$.*

Proof. Let us restart the algorithm several times while decreasing Δ. More exactly, let $\Delta_1 > 0$ be fixed and choose $\Delta_{p+1} = \Delta_p/2$ for $p = 1, 2, \ldots$. As a result, we obtain a sequence of solutions each of which is a local optimum of the discrete-time problem with respect to the selected initial set of feasible and finite-dimensional decision variables. Thus, to prove the proposition, we have to show that each sequence of the locally optimal solutions of the discrete-time problem converges to the global solution of the continuous-time problem. Suppose that this is not true and there exist two sequences of the locally optimal solutions of the discrete-time problem, S_p^1 and S_p^2 for $p = 1, 2, \ldots$, such that

$$\lim_{p\to\infty} (Z_p^1 - Z_p^2) = A > 0, \tag{33}$$

where Z_p^1 and Z_p^2 are the corresponding sequences of the objective values.

Consider continuous-time solutions S_p^3 and S_p^4 which are built on the basis of S_p^1 and S_p^2 as follows:

- the decision variable $y_{ij}(t)$ of S_p^3 equals the decision variable $y_{ij}(t_r)$ of S_p^1,

$$y_{ij}(t) = y_{ij}(t_r) \text{ for } t_r \leqslant t < t_{r+1},\ i \in I,\ j \in J,\quad r = 0, \ldots, R-1,\quad p = 1, 2, \ldots, \tag{34}$$

- the decision variable $y_{ij}(t)$ of S_p^4 equals the decision variable $y_{ij}(t_r)$ of S_p^2,

$$y_{ij}(t) = y_{ij}(t_r) \text{ for } t_r \leqslant t < t_{r+1},\quad i \in I,\ j \in J,\ r = 0, \ldots, R-1,\quad p = 1, 2, \ldots. \tag{35}$$

Denote by Z_p^3 and Z_p^4 the objective values of the solutions S_p^3 and S_p^4. Note, the more we restart the algorithm, the closer the discrete-time optimal solution approaches a continuous-time solution with infinite-dimensional binary decision variables. Since each of the discrete-time locally optimal solutions S_p^1 and S_p^2 satisfies the discrete-time form of the optimality conditions (see Step 6 of the algorithm), S_p^3 and S_p^4 determined by (34) and (35) must satisfy the continuous-time form of the same conditions (23)–(26) as $p \to \infty$. However, the continuous-time problem is unimodal, which implies:

$$\lim_{p\to\infty} (Z_p^3 - Z_p^4) = 0. \tag{36}$$

Furthermore, from (34) and (35), it immediately follows that:

$$\lim_{p\to\infty} (Z_p^1 - Z_p^3) = 0 \quad\text{and}\quad \lim_{p\to\infty} (Z_p^2 - Z_p^4) = 0. \tag{37}$$

By considering (36) and (37) together, we conclude that $\lim_{p\to\infty}(Z_p^1 - Z_p^2) = 0$ which contradicts (33). □

COROLLARY 1 (lower bound). *Assume that the optimal value Z_p of the equivalent DGAP is calculated by the Needle algorithm performed on a Δ_p-grid. Also let Z_p^R be the optimal value for the relaxed DGAP on the same time grid, i.e. constraints* (3) *are replaced with* (22). *Then*

$$Z_p^R \leqslant Z_p \text{ and } \lim_{p\to\infty} Z_p = \lim_{p\to\infty} Z_p^R.$$

Proof. The relaxed DGAP with a fixed Δ_p is a quadratic programming problem, for which feasible solutions are not necessarily binary. Therefore, it does not cover all feasible solutions for the equivalent DGAP, i.e. $Z_p^R \leqslant Z_p$. However, Proposition 2 says that $\lim_{p\to\infty} Z_p = Z_\infty$ holds, where by Z_∞ we denote the value of the objective function for the continuous-time problem. By the same argument, we have $\lim_{p\to\infty} Z_p^R = Z_\infty^R$. Furthermore, Theorem 2 says that $Z_\infty = Z_\infty^R$ holds. Thus, $\lim_{p\to\infty} Z_p = \lim_{p\to\infty} Z_p^R$. □

Note that the above lower bound is necessary to assess whether Δ_p is chosen properly, so that a required accuracy is guaranteed. When the differential equations are replaced with the corresponding difference equations on a Δ_p-grid, the lower bound Z_p^R is obtained in polynomial time from the relaxed DGAP solvable as a quadratic programming problem with linear constraints.

PROPOSITION 3. *The Needle algorithm on a Δ-grid with $R+1$ equally distributed mesh points ($\Delta = T/R$) requires at most $(V/\varepsilon)R^3$ iterations, where*

$$V = \frac{\Delta}{T^3} \sum_{r=1}^{R} \left(\sum_{j\in J} c_j^X \left(X_j^0 - \sum_{k\in K} m_{jk} \varphi_{jk}(t_r) \right)^2 \right), \tag{38}$$

$$\varepsilon = \min_{\substack{\delta y_{ij}\in\{1,0,-1\}\\ \delta y_{i'j}\in\{1,0,-1\}}} 2 \left(\sum_{i\in I} \sum_{j\in J} \sum_{i'\in I} c_j^x U_{ij} U_{i'j} \delta y_{ij} \delta y_{i'j} \right)^+, \tag{39}$$

and operator α^+ is determined as: $\alpha^+ = \infty$ *if* $\alpha \geqslant 0$ *and* $\alpha^+ = |\alpha|$ *if* $\alpha < 0$.

Proof. We consider a small variation of the decision variables ($\Delta \to 0$). Therefore, the terms of higher orders of Δ than the first order, are negligible.

The variation of the objective function in the first order of Δ on an iteration of the algorithm is due to the needles set by Step 7 at point t_{r^*}:

$$\delta Z = \Delta \sum_{i\in I} \sum_{j\in J} c_{ij}^y \delta y_{ij}(t_{r^*}) + \Delta \sum_{r=r^*+1}^{R} \sum_{j\in J} \left(2c_j^x \left(\widehat{X}_j(t_r) - \sum_{k\in K} m_{jk} \hat{\varphi}_{jk}(t_r) \right) \delta \widehat{X}_j(t_r) \right), \tag{40}$$

where $\delta y_{ij}(t_r)$ can be equal either to 1, or to 0, or to -1; and $\delta\widehat{X}_j(t_r)$ is the corresponding variation of the task levels $\widehat{X}_j(t_r)$,

$$\delta\widehat{X}_j(t_r) = \Delta\sum_{i\in I} U_{ij}\delta y_{ij}(t_{r^*}) \quad \text{for all } r > r^*, \quad j\in J. \tag{41}$$

By integrating (32), we obtain

$$\psi_j(t_r) = -\Delta\sum_{r'=r+1}^{R}\left(2c_j^x\left(\widehat{X}_j(t_{r'}) - \sum_{k\in K} m_{jk}\hat{\varphi}_{jk}(t_{r'})\right)\right), \tag{42}$$

and substituting (41) and (42) into (40), we obtain

$$\delta Z = \Delta\sum_{i\in I}\sum_{j\in J}\delta y_{ij}(t_{r^*})(c_{ij}^y - U_{ij}\psi_j(t_{r^*})). \tag{43}$$

According to (42) and (43), the minimum deviation of $U_{ij}\psi_j(t_{r^*})$ from c_{ij}^y occurs when at the previous mesh point t_{r^*-1} the following holds:

$$\widehat{X}_j(t_{r^*-1}) = \sum_{k\in K} m_{jk}\hat{\varphi}_{jk}(t_{r^*-1}) \quad \text{and} \quad U_{ij}\psi_j(t_{r^*-1}) = c_{ij}^y.$$

In this case, at point t_{r^*}:

$$\widehat{X}_j(t_{r^*}) - \sum_{k\in K} m_{jk}\hat{\varphi}_{jk}(t_{r^*}) = \Delta\sum_{i\in I} U_{ij}\delta y_{ij}(t_{r^*})$$

and

$$U_{ij}\psi_j(t_{r^*}) - c_{ij}^y = 2c_j^x\Delta^2 U_{ij}\sum_{i'\in I} U_{i'j}\delta y_{i'j}(t_{r^*}). \tag{44}$$

In all other cases the deviation of $U_{ij}\psi_j(t_{r^*})$ from c_{ij}^y is evidently $O(\Delta)$, rather than $O(\Delta^2)$ as in this case. By substituting (44) in (43), we obtain the minimum improvement of the objective as

$$\min|\delta Z| \geqslant \min_{\substack{\delta y_{ij}\in\{1,0,-1\}\\ \delta y_{i'j}\in\{1,0,-1\}}} 2\Delta^3\left(\sum_{i\in I}\sum_{j\in J}\sum_{i'\in I} c_j^x U_{ij}U_{i'j}\delta y_{ij}\delta y_{i'j}\right)^+.$$

Operator α^+ excludes the needles which result in deterioration of the objective, $\delta Z \geqslant 0$.

Given $\Delta \neq 0$, the maximum number of iterations, required to calculate the optimal solution under the slowest improvement in the objective at every iteration, is straightforwardly defined as follows:

$$n = \frac{Z_1 - Z_2}{\varepsilon\Delta^3} = \frac{Z_1 - Z_2}{\varepsilon T^3}R^3, \tag{45}$$

where the minimal value Z_2 of the objective is a non-negative number and the maximal value of the objective Z_1 from which the algorithm starts is obtained by setting all decision variables at zero:

$$Z_1 = \sum_{r=1}^{R}\left(\sum_{j\in J} c_j^x\left(X_j^0 - \sum_{k\in K} m_{jk}\hat{\varphi}_{jk}(t_r)\right)^2\right)\Delta.$$

The proof is completed by replacing: $V = Z_1/T^3$ in (45) and setting $Z_2 = 0$. □

We are now able to estimate the worst-case complexity of the Needle algorithm for special versions of the equivalent DGAP found in practice.

THEOREM 4. *The Needle algorithm solves the equivalent DGAP on a* Δ-*grid with* $R+1$ *equally distributed mesh points* ($\Delta = T/R$) *in the computation time* $\mathrm{O}(V/\varepsilon R^4 a^3)$ (*capacity-dependent estimate*)

$$a = 2\max_r \left\{ \sum_i \beta_i(t_r) + \sum_j \chi_j(t_r) \right\},$$

or $\mathrm{O}(V/\varepsilon R^4 a^4 \log a)$ *time* (*capacity-independent estimate*),

$$a = 2|I| + 2|J|,$$

where V *and* ε *are determined by* (38) *and* (39), *respectively.*

Proof. Step 6 of the algorithm is the most complex step. The integration of the primal and dual equations (30) and (32) as well as of the objective function (31) evidently requires $\mathrm{O}(IJR)$ operations. In step 6, the static dual-assignment problem with multiple agent–task relationship is solved R times. To reduce this problem to the classical one with one-to-one agent–task relationship, we simply break down at every time point t_r every agent i into $\beta_i(t_r)$ agents and every task j into $\chi_j(t_r)$ tasks and add slack variables (dummy agents and dummy jobs) to produce equalities from inequality constraints (1) and (2). Then the new set of agents becomes equal to the new number of tasks, which is $\sum_i \beta_i(t_r) + \sum_j \chi_j(t_r)$. Thus, the maximal number of nodes we obtain for the classical assignment is: $a = 2\max_{t_r}\{\sum_i \beta_i(t_r) + \sum_j \chi_j(t_r)\}$ while the maximal number of arcs is $b = \frac{a}{2} \cdot \frac{a}{2}$. The best available, strongly polynomial time bound for the classical assignment problem is $\mathrm{O}(ab + a^2 \log a)$ (e.g., Ahuja *et al.* (1993)).

By taking into account that the assignment problem is solved R times at each iteration and that the maximal number of iterations is derived in Proposition 2, one can readily obtain $\mathrm{O}(V/\varepsilon R^4(a^3/4 + a^2 \log a))$ from where the capacity-dependent estimate stated in this theorem immediately follows.

To obtain a capacity-independent estimate, we utilize the fact that the multiple agent–task assignment is a special case of the minimum cost flow problem, which contains no transshipment nodes. Similar to the previous estimate, we use slack variables to convert inequality constraints (1) and (2) into equalities and to ensure the mass balance $\sum_i \beta_i(t_r) = \sum_j \chi_j(t_r)$. Finally, we take the best available time bound for this problem $\mathrm{O}((b \log a)(b + a \log a))$, where $a = 2|I| + 2|J|$ and $b = \frac{a}{2} \cdot \frac{a}{2}$, and apply the same argument as the case of capacity-dependent estimate. Thus, we obtain the complexity as $\mathrm{O}(V/\varepsilon R^4(\frac{a^2}{4}\log a)(\frac{a^2}{4}+a\log a))$, from which the capacity-independent estimate follows immediately. □

Theorem 4 considers a general case of the SDGAP where the number of tasks and the number of agents are limited by the agent and task resource constraints (1) and (2), respectively, which imply $\chi_j(t) \leqslant |I|$ and $\beta_i(t) \leqslant |J|$. The following two corollaries are concerned with very important special cases of the dynamic assignment problem applied to scheduling flexible manufacturing systems. One is the case of scheduling a number of the workstations (agents) processing a limited number of parts (tasks) $(\beta_i(t) \leqslant |J|)$, when there is no limit on the number of the workstations simultaneously processing the same part $(\chi_j(t) \geqslant |I|)$. The other is a classical preemptive scheduling of machines which process at most one product at a time $(\beta_i(t) = 1)$ to minimize dynamic tardiness and lateness. In both special cases, the complexity of the algorithm significantly decreases, compared with the general result of Theorem 4.

COROLLARY 2. *The Needle algorithm solves the equivalent DGAP on a* Δ-*grid with* $R+1$ *equally distributed mesh points with unlimited task* (*or agent*) *capacities in*

$$\mathrm{O}\big(V/\varepsilon R^4|I||J|\log|J|\big)\, \textit{time if}\ \chi_j(t) \geqslant |I|,$$

$$\mathrm{O}\big(V/\varepsilon R^4|I||J|\log|I|\big)\ \textit{time if}\ \beta_i(t) \leqslant |J|.$$

Proof. The difference from the equivalent DGAP is that the Needle algorithm, in Step 6, solves the simplified assignment problem R times:

$$\sum_{i\in I}\sum_{j\in J}(U_{ij}\psi_j(t) - c^y_{ij})y_{ij}(t) \to \max,$$

$$\sum_{j\in J} y_{ij}(t) \leqslant \beta_i(t), \quad i \in I,$$

$$y_{ij}(t) \in [0,1], \quad i \in I,\ j \in J.$$

Since there is no constraint that correlates different agents, this problem can be decomposed into $|I|$ independent optimization problems, each of which is solvable by the following greedy procedure.

> For every agent i: Sort all tasks in a non-increasing order of positive gradients $\partial H/\partial y_{ij} = (U_{ij}\psi_i(t) - c^y_{ij})$. Let the number of positive gradients be $n_i(t)$. Then, in the constructed sequence of tasks, assign the first $\min\{\beta_i(t), n_i(t)\}$ tasks to the agent $i : y_{ij}(t) = 1$, and $y_{ij}(t) = 0$ for the other tasks in the sequence.

By taking into account that sorting requires $\mathrm{O}(|J|\log|J|)$ time for every agent i, we readily obtain the complexity stated in the corollary. □

COROLLARY 3. *The Needle algorithm solves the equivalent DGAP on a Δ-grid with $R+1$ equally distributed mesh points with unlimited task (or agent) capacities and a single agent (or task) capacity in*

$$\mathrm{O}\left(\frac{V}{\varepsilon}R^4|I|(|J|+1)\right) \text{ time if } \beta_i = 1 \text{ and } \chi_j(t) \geqslant |I|,$$

$$\mathrm{O}\left(\frac{V}{\varepsilon}R^4|J|(|I|+1)\right) \text{ time if } \chi_j = 1 \text{ and } \beta_i(t) \geqslant |J|.$$

Proof. The difference form of the equivalent DGAP is again the assignment problem to be solved in step 6. The new assignment problem we obtain in the dual space is:

$$\sum_{i\in I}\sum_{j\in J}(U_{ij}\psi_j(t) - c_{ij}^y)y_{ij}(t) \to \max,$$

$$\sum_{j\in J} y_{ij}(t) \leqslant 1, \quad i \in I,$$

$$y_{ij}(t) \in [0,1], \quad i \in I, \ j \in J.$$

This problem is also decomposed into $|I|$ independent optimization problems, each of which is solvable in a straightforward manner by trying an agent to perform every single task ($|J|$ operations) and do not perform any task (one operation) and comparing the objective value. This requires a total of $|I|(|J|+1)$ operations. □

9. Computational Results

In this section we present an example and computational results of the experiments conducted in order to assess the accuracy of the algorithm for various time grids Δ.

We illustrate a model for the assignment problem found in a copy center at a university bookstore. There are six machines (agents) of four different types. Each machine can copy one or more of the following page sizes (tasks): A4 ($j=1$), Legal ($j=2$), B4 ($j=3$), A3 ($j=4$). Since each machine can carry out only one task at a time, the agent resource is $\beta_i(t) \equiv 1$, $i = 1,\ldots,6$. The task resources are set at $\chi_1(t) = \chi_3(t) = \chi_4(t) \equiv 2$ and $\chi_2(t) \equiv 4$. Every task can hardly be assumed to be demanded in a steady and pure deterministic fashion. The demand forecasts given below were calculated by SPSS nonlinear autoregression analysis performed on the data of the copy center:

$$d_1(t) = 2 + \xi_1(-0.25t^4 + 4.55t^3 - 25t^2 + 44t),$$

$$d_2(t) = 5 + \xi_2(-0.235t^4 + 4.47t^3 - 26t^2 + 40.4t),$$

$$d_3(t) = \xi_3(-0.25t^4 + 4.2t^3 - 20t^2 + 40t),$$

$$d_4(t) = \xi_4(0.33t^3 - 7.2t^2 + 40t),$$

where the expected values of the random coefficients ξ_j are, respectively, $m_1 = 0.05$, $m_2 = 0.05$, $m_3 = 0.02$, $m_4 = 0.03$. Initial task levels are given as follows: $X_1^0 = X_3^0 = 0$, $X_2^0 = 1, X_4^0 = 2$; and all cost coefficients for task surpluses and shortages are set at 25 cost units.

The time it takes to copy a page on each of the six machines and the cost per page incurred when working on the machines are summarized in Table 1. The problem of assigning tasks along the working hours is typical of this type of operation. The task units (in 1000 pages) along with the optimal sequence for performing them found by our algorithm for $\Delta = \frac{1}{2}$ ($T = 10$), as well as the expected values of the corresponding demands (in 1000 pages per hour), are shown in Figure 1. Assignment of four different page sizes (tasks) over the planning horizon for every copy machine is shown by boxes of four black-white intensities.

Table 2 presents computation times and deviation of the objective value (Relative Optimality Gap) obtained on an IBM PC-586 computer (120 MHz, REM 32 Mb) when applying the Needle algorithm and mathematical programming (*NLP*) package GAMS. The table compares the needle-based optimal solutions with corresponding lower bounds (Needle *VS* Relaxation) for various assignment problems, where *IJ* is an integrated index obtained as the product of the number of agents and tasks and $IJ(R+1)$ is the total number of variables.

Table 2 shows the average running time and the convergence of the algorithm based on several hundreds computational experiments. The experi-

Table 1. Time and cost per page on each of the copy machines

		Page size							
Machine number,	Machine type	A4		Legal		B4		A3	
i		Time	Cost	Time	Cost	Time	Cost	Time	Cost
1	A	1	2	1.3	2.5	1.5	2.7	2.0	3.0
2	A	1	2	1.3	2.5	1.5	2.7	2.0	3.0
3	B	0.8	2.5	1.0	2.5	1.2	2.7	1.8	2.8
4	C	1.3	1.6	1.5	2.3	1.8	2.3	2.0	2.5
5	C	1.3	1.6	1.5	2.3	1.8	2.3	2.0	2.5
6	D	1.5	1.5	2.0	2.0	2.3	2.0	2.3	2.5

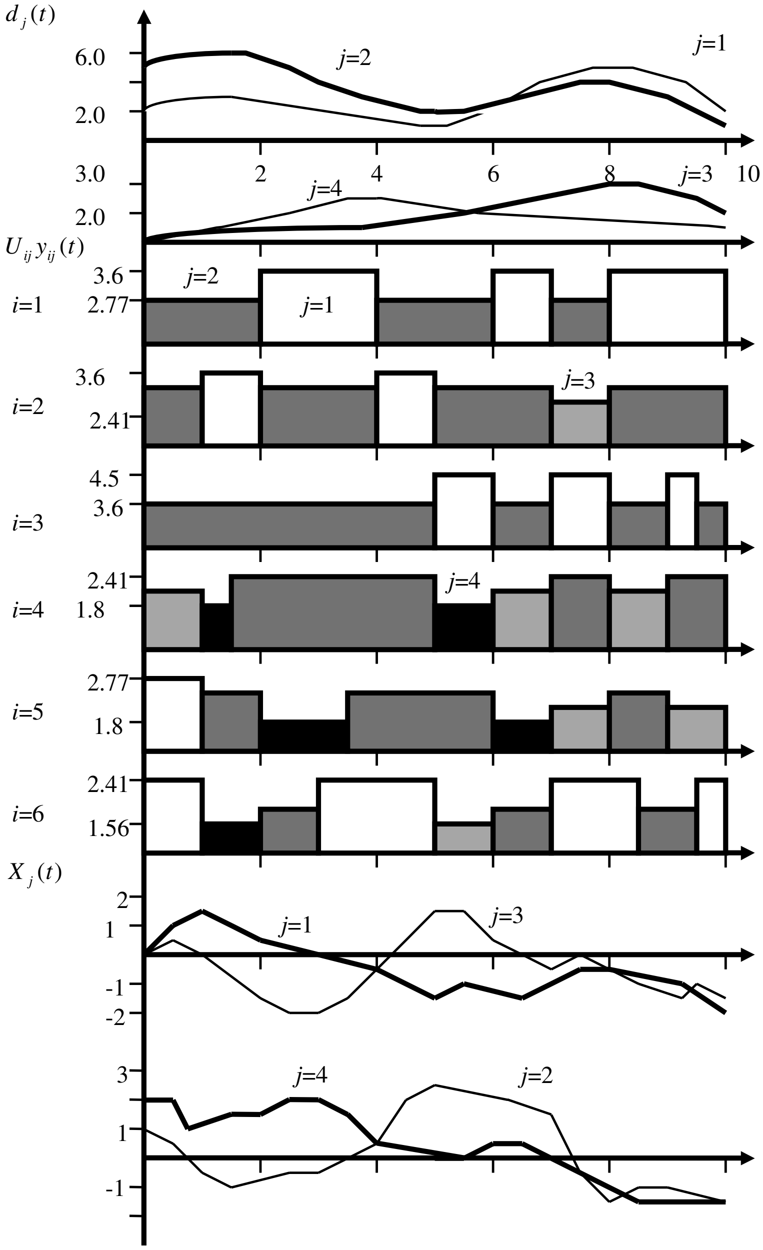

Figure 1. Solution and the expected demands for the university shop.

ments were conducted on the planning horizon of $T = 10$ time units with linear demand functions $d_j(t) = \xi_j(a_j + b_j t)$. The demand parameters for each I and production rates were chosen randomly, a_j within the range [2I, 5I], b_j within the range $[-\frac{1}{5}I, \frac{1}{5}I]$, and U_{ij} within the range [0, 10]. The

Table 2. Computational results

Period, Δ	Variables	Optimality gap (%)	CPU time (min)	
	IJ(*R*+*1*)	Needle *VS* relaxation	Needle solution	Relaxed solution
IJ = 4				
1	44	34.0	0.03	0.91
1/2	84	19.1	0.09	0.92
1/4	164	9.3	0.33	1.11
1/8	324	3.9	1.38	1.40
1/16	644	1.9	6.90	3.28
IJ = *10*				
1	110	46.5	0.04	1.10
1/2	210	25.3	0.21	1.13
1/4	410	11.3	0.52	1.41
1/8	810	4.4	3.30	5.52
1/16	1610	2.0	14.7	13.1
IJ = *15*				
1	165	61.1	0.07	1.02
1/2	315	32.9	0.35	1.21
1/4	615	15.1	1.37	1.55
1/8	1215	5.3	6.10	6.31
1/16	2415	2.4	19.2	15.3

expected values of the random coefficients ξ_j were chosen equal to 1 for each j. The uniform distribution was also used to generate the capacities $\beta_i(t)$ and $\chi_j(t)$ as constants for each I and J in the range of [0,*J*] and [0,*I*], respectively.

Table 2 also shows that the optimality gap between the suggested Needle algorithm and the corresponding lower bound shortens from almost 50% to a few percents as Δ reduces from 1 to 1/16 time unit.

10. Summary and Conclusions

A dynamic generalization of the well-known assignment problem, SDGAP, is introduced to incorporate important factors such as time, multiple agent–task relationships and stochastic demands for the task units. Although this assignment problem is *NP*-hard, it is revealed by applying the maximum principle that the problem has an integrality property and its dual formulation is polynomially solvable at every time point. Moreover, the dual problem is a classical assignment with multiple agent–task relationships. Based on these results, a needle time-decomposition algorithm is proposed, in which the classical assignment is solved as a subroutine to approximate the optimal solution in the dual space. The algorithm runs in pseudo-polynomial time for various versions of the SDGAP and converges to the globally optimal solution as the size of the time grid decreases.

Appendix

LEMMA A.1. *On an optimal solution each agent performs no more than one task over the entire planning horizon if*

- *agent capacities are minimal*: $\beta_i(t) = \beta_i = 1,\ i \in I$, *and task capacities are unlimited*: $\chi_j(t) = \chi_j \geqslant |I|,\ j \in J$;
- *the task intensities are not identical*: $U_{ij} \neq U_{i'j'},\ i, i' \in I, j, j' \in J$;
- *all demands are static, deterministic, equal*: $d_j(t) = d, j \in J$ *and large*

$$d > \frac{U_{ij} \sum_{i' \in I} U_{i'j}}{|U_{ij} - U_{ij'}|}, \quad \forall i \in I,\ j, j' \in J,$$

there are no initial task units: $\widehat{X}_j(0) = X_j^0 = 0, j \in J$;

- *there are no processing costs*: $c_{ij}^y = 0,\ i \in I, j \in J$;
- *all penalties for shortages and surpluses are equal*: $c_j^x = c, j \in J$.

Proof. First note, given conditions of the lemma, the dual formulation (15)–(16) simplifies to

$$\dot{\psi}_j(t) = 2c(\widehat{X}_j(t) - dt), \psi_j(T) = 0. \tag{A.1}$$

$$H(t) = -\sum_{j \in J} c(\widehat{X}_j(t) - dt)^2 + \sum_{j \in J} \psi_j(t) \sum_{i \in I} U_{ij} y_{ij}(t). \tag{A.2}$$

By maximizing the Hamiltonian with respect to $y_{ij}(t)$, we immediately find the following optimal behavior:

$$y_{ij}(t) = \begin{cases} 1, \text{if } \psi_j(t) > 0 \text{ and } U_{ij}\psi_j(t) > U_{ij'}\psi_{j'}(t), j \neq j', \\ 0, \text{if } \psi_j(t) < 0 \text{ or } \exists j' \neq j, U_{ij}\psi_j(t) < U_{ij'}\psi_{j'}(t), \\ y_{ij} \in \{0, 1\}, \text{ otherwise.} \end{cases} \tag{A.3}$$

Next, based on (A.3), we prove by contradiction that no agent switches between tasks an infinite number of times. Indeed, such infinite switching (chattering) of agent i can occur only when

$$U_{ij}\psi_j(t) = U_{ij'}\psi_{j'}(t) \tag{A.4}$$

for a pair of tasks j and j' on an interval of time. By differentiating twice the last condition and taking into account equations (9) and (A.1), we obtain that on this time interval

$$U_{ij}\left(\sum_{i' \in I} U_{i'j} y_{i'j}(t) - d\right) = U_{ij'}\left(\sum_{i' \in I} U_{i'j'} y_{i'j'}(t) - d\right).$$

Resolving the last equation in d results in

$$d = \frac{U_{ij} \sum_{i' \in I} U_{i'j} y_{i'j}(t) - U_{ij'} \sum_{i' \in I} U_{i'j'} y_{i'j'}(t)}{U_{ij} - U_{ij'}}. \tag{A.5}$$

By taking into account the fact that $\max y_{ij}(t) = 1$, $\min y_{ij}(t) = 0$ and $U_{ij} \neq U_{ij'}$, we find that condition $d > U_{ij} \sum_{i' \in I} U_{i'j} / |U_{ij} - U_{ij'}|$, $\forall i \in I$, j, $j' \in J$ of Lemma A.1 ensures (A.5) never holds. Therefore, (A.4) cannot hold on an interval of time that contradicts our assumption that agent i switches an infinite number of times.

Finally, we prove that no agent switches between tasks a finite number of times. To this end, it is sufficient to show that no task is performed by different agents over the planning horizon. The proof is again by contradiction. Let task j be initially performed by agent i and at time s agent i' starts performing the task until time p, $p \leqslant T$. Assume that this sequence is optimal and let $a = U_{ij} - d$ and $b = U_{i'j} - d$. The cost incurred by task j over interval $t \in [0, p]$ is

$$J = \int_0^p c(\widehat{X}_j(t) - td)^2 \mathrm{d}t = c\left(\frac{1}{3}a^2 s^3 + (p-s)\left(a^2 s^2 + abs(p-s) + \frac{1}{3}b^2 (p-s)^2\right)\right).$$

In order to find the best switching point, s, for the described sequence, we differentiate J with respect to s

$$J'_s = c(p-s)(a-b)((2a-b)s + bp).$$

Since switching infinite number of times has been proven to be impossible, $p - s > 0$, therefore, J'_s does not equal zero on interval $s \in (0, p)$. Thus, no change of agents when performing task j can be optimal over the planning horizon, which contradicts our initial assumption. □

References

Ahuja, R.K., Magnati, T.L. and Orlin, J.B. (1993), *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall, Englewood Cliffs, NJ.

Algazi, V.R. and Sakrison, D.J. (1969), On the Optimality of the Karhunen–Loève Expansion, *IEEE Transactions on Information Theory* 15, 319–321.

Balachandran, V. (1976), An integer generalized transportation model for optimal job assignment in computer networks, *Operations Research* 24, 742–759.

Balas E. and Zemel, E. (1980), An algorithm for large zero–one Knapsack problems, *Operations Research* 28, 1130–1154.

Dubovitsky, A.Y. and Milyutin, A.A. (1981), Theory of the Maximum Principle. In: Levin, V.L. (ed.), *Methods of Theory of Extremum Problems in Economy*, pp. 6–47, Nauka, Moscow.

Hartl, R.F., Sethi, S.P. and Vickson, R.G. (1995), A survey of the maximum principles for optimal control problems with state constraints, *SIAM Review* 37(2), 181–218.

Hillier F.S. and Lieberman, G.J. (1995), *Introduction to Operations Research*, McGraw-Hill.

Garey M.R. and Johnson, D.S. (1991), *Computers and Intractability: A guide to the Theory of NP-Completeness*, W.H. Freeman and Company, New York.

Gavish, B. and Pirkul, H. (1986), Computer and database location in distributed computer systems, *IEEE Transactions on Computers* 35(7), 583–590.

Gavish, B. and Pirkul, H. (1991), Algorithms for the multi-resource generalized assignment problem, *Management Science* 37(6), 695–713.

Klastorin, T.D. (1979), An effective subgradient algorithm for the generalized assignment problem, *Computers and Operations Research* 6, 155–164.

Khmelnitsky, E., Kogan, K. and Maimon, O. (1995), A maximum principle based combined method for scheduling in a flexible manufacturing system, *Discrete Events Dynamic Systems* 5, 343–355.

Kogan, K., Shtub A. and Levit, V. (1997), DGAP–The dynamic generalized assignment problem, *Annals of Operations Research* 69, 227–239.

Martello, S. and Toth, P. (1977), An upper bound for the zero–one Knapsack problem and branch and bound Algorithm, *European Journal of Operational Research* 1, 169–175.

Mazzola, J.B. and Neebe, A.W. (1986), Resource-constrained assignment scheduling, *Operations Research* 34(4), 560–572.

Murphy, R.A. (1986), A Private Fleet Model with Multi-Stop Backhaul, *Working Paper*, 103, Optimal Decision Systems, Green Bay, WI.

Nauss, R.M. (1976), An efficient algorithm for the 0–1 Knapsack problem, *Management Science* 23, 27–31.

Ross, G.T. and Soland, R.M. (1975), A branch and bound algorithm for the generalized assignment problem, Math. Programming 8, 91–103.

Ross, G.T. and Soland, R.M. (1977), Modeling facility location problems as generalized assignment problems, *Management Science* 24, 345–357.

Sousa, J.B. and Pereira, F.L. (1992), A hierarchical framework for the optimal flow control in manufacturing systems. In: *Proceedings of the Third International Conference on Computer Integrated Manufacturing*, Troy, pp. 278–286.

Van Trees, H.L. (1968), *Detection, Estimation and Modulation Theory*, Part I, John Wiley & Sons.

Yagiura, M., Yamaguchi, T. and Ibaraki, T. (1999), A variable depth search algorithm for the generalized assignment problem. In: Voss, S., Martello, S., Osman, I.H., and Roucairol, C., (eds.), *Meta-heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer Academic Publishers, Boston, pp. 459–471.

Youla, D.C. (1957), The solution of a homogeneous Wiener–Hopf integral equation occurring in the expansion of second-order stationary random functions, *IEEE Transactions on Information Theory*, 187–193.