Journal of Mathematical Modeling
Vol. 10, No. 3, 2022. pp. 367-385. Research Article

JMM

Multi-agent single machine scheduling problem with transportation constraints

Mohsen Ziaee[†*], Mahdi Imanparast[‡], Vahid Khodabakhshi[†]

[†]*Department of Industrial Engineering, University of Bojnord, Bojnord, Iran*
[‡]*Department of Computer Science, University of Bojnord, Bojnord, Iran*
Email(s): ziaee@ub.ac.ir, m.imanparast@ub.ac.ir, vahidkh74@yahoo.com

Abstract. A multi-agent single machine scheduling problem with transportation constraints is studied. We assume that there are several independent agents placed in different geographical locations, each of them has several orders and each order includes different types of products. We use a simple and effective model to obtain maximum profit of the products. To have desired on-time deliveries, the minimization of the transportation costs and total tardiness costs are considered as objective functions. The main idea of this research is to develop a simple and integrated scheduling and transportation model which can be applied in many factories, chain stores, and so on. In order to solve this problem, a mixed integer linear programming (MILP) model is presented. Moreover, since solving large instances of the proposed MILP model is very time-consuming, a heuristic algorithm is presented. Implementing of two approaches on a variety of datasets show that the heuristic algorithm can provide good-quality solutions in very short time.

Keywords: Single machine scheduling problem, transportation constraints, mixed integer linear programming, heuristic algorithm.
AMS Subject Classification 2010: 90-XX, 90C05.

1 Introduction and problem description

In many production and non-production environments, the response time to customer demand is a very important issue and considered as a competitive advantage. One of the important factors affecting on the response time is the transportation time. Therefore, many factories and researchers focus on the integrated production planning/scheduling and transportation systems [9–11, 15, 24, 25] . A review of studies on scheduling and transportation problems can be found in [2, 8, 30]. Researches on this subject either do

*Corresponding author.
Received: 6 August 2021/ Revised: 25 September 2021 / Accepted: 25 October 2021
DOI: 10.22124/jmm.2021.20310.1769

not consider the vehicle routing models or use these models leading to more complex problems. Multi-agent scheduling, introduced by Baker and Smith [3], and observed in many real-world problems [21], and has attracted a large amount of attention during the past decade [7, 12, 13, 20, 22, 26, 28, 29]. In this environment, there are several agents that compete on the usage of common resources (machines); each agent has several orders and has its own objective function. In this paper, we use a very simple and effective method to find best paths, instead of using more complex vehicle routing models. We generalize this integrated scheduling and transportation problem to a multi-agent single machine scheduling and transportation problem. We also assume that all orders of a given agent are delivered in a same geographical location. In order to have desired on-time deliveries, the minimization of total tardiness cost is considered as one of our objective functions. Moreover, sequence dependent setup times are taken into account, i.e. if two consecutively processed products are of different types, then a setup time between them will be required. These two problem features, i.e. the total tardiness objective and the sequence dependent setup times are widely used in the literature (see [1, 5, 16, 17, 23, 27]). Buyers (agents or customers) are often interested in using quantity discounts in which the supplier will decrease a products unit price if their customers order quantity exceeds some specified breakpoint. The two most common types of quantity discount schedules are as follows [6]:

1. *all-unit discounts* in which every bought item is reduced in price,
2. *incremental discounts* in which the price reduction is applied to only those bought items above the corresponding breakpoint.

In this paper, we use the all-unit discount and assume that each product type has its own particular discount model and discount intervals. In the literature, few papers deal with the distinct discount models in scheduling problems. To the best of our knowledge, only Lu et al. [31] and Lu et al. [32] investigate the single-machine scheduling problem with outsourcing and distinct discount models. Because of uncertainty in the customer demand for future planning periods and to prevent inventory shortages and to have less setups, safety stocks are often taken into account, which are additional items over the stock required to response the demand of customers ([4, 18, 19]) . In this paper, it is assumed that for each product type, a safety stock must be held, where its amount should not be less than a fixed and known lower limit. Also, the weighted sum of safety stocks is considered as an objective (see Section 2). In general, this problem is strongly NP-hard [14], and the results of literature review show that our studied problem with all the above mentioned features has never been studied in the past, and is a very practical problem that can be applied in many production and non-production environments such as packing and distribution centers, chain stores, and so on. The remainder of this paper is organized as follows: In Subsection 1.1, basic concepts and definitions of the problem are presented. Then, a mixed integer linear programming (MILP) model to optimally solve the problem and an efficient heuristic algorithm to solve the problem are proposed in Section 2. The performance of the proposed MILP model and the proposed heuristics algorithm is evaluated by some test problems in Section 3. Concluding remarks are given in Section 4.

1.1 Problem definition

The studied problem can be defined as follows. A single factory (machine/facility/distributor) has m independent agents (customers) placed in different geographical locations. Each agent a $(a = 1, 2, \ldots, m)$ has to_a orders and each order includes several product (job) types. lb_{aop} is the minimum required number of products p for order o of agent a, determined by the agent, i.e. the factory can send (buy) more than

lb_{aop} to the agent as a surplus buy, but the total number of surplus products of type p for agent o should not be greater than a predetermined limit ub_{ap}. In each order o of agent a, there are n_{aop} products of type p. Processing (producing/packing/) time of one unit of product type p is t_p. The parameter $t_{pp'}$ is the setup time between two consecutive products p and p'. Obviously, if two consecutively processed products are of the same type, then the setup time between them is zero. Due to simplicity of the model, we assume that all same-type products of each order are processed continuously and consecutively. There are V vehicles for transporting the products to the agents. The capacity of each vehicle is given and denoted by $vcap_v$, and capacity needed by one unit of product type p is cap_p. The problem has multiple objectives as follows:

(1) One of the objectives of the problem is to maximize the total profit of the products, without considering transportation and tardiness costs, since the problem under study is an integrated scheduling and transportation problem and also considers tardiness costs. In calculating each product profit, a discrete discount model is taken into account. We assume that each product p has D_p discount intervals and e_{pd_p}'s $(d_p = 1,2,\ldots,D_p)$ are corresponding break points. This means that if the factory sells $ND_{ap}^{d_p}$ products of type p to agent a which is in interval $[e_{pd_p}, e_{pd_p+1}]$, then its one-unit profit is equal to $profit_p^{d_p}$.

(2) Another objective is the minimization of the transportation costs which is the sum of the fixed costs of vehicles selected and used in the system. It is assumed that each selected vehicle is applied for only one travel from the factory to several agents. Moreover, we assume that the transportation times are negligible in comparison with the processing times of the products, and so the orders delivery times are equal to their completion times. In this paper, instead of using transportation planning models and finding shortest paths which leads to a much more complex model, we use a new, simple and effective constraint in which, two agents a and a' can be consecutively served by the same vehicle, if and only if the distance between them is less than or equal to a given limit $(dist_{max})$.

(3) The third objective is to minimize the total tardiness costs. Each order o of agent a has a due date due_{ao} and if its delivery time is greater than this due time, then a penalty cost tc_{ao} must be paid for each unit-time tardiness.

Since objectives (1), (2) and (3) have the same dimension; we merge them as a single objective.

(4) In this objective, we look for solutions in which, less number of agents are served by each vehicle, or less number of vehicles are applied to transport the orders of each agent. By using this objective, the number of loading and unloading and the distance traveled by each vehicle is minimized.

(5) The fifth objective is to maximize the weighted sum of safety stocks. Weight (importance coefficient) of safety stock of product p, denoted by σ_p, is determined by its price, demand, weight, volume and other factors. Greater value of σ_p means that more safety stock should be held. However, each product p has a lower bound $lb.ss_p$, i.e. its safety stock should not be less than this lower bound.

The problem is to determine:

- How many products of different types are produced and delivered for each agent, taking into account its lower and upper limits (lb_{aop} and ub_{ap}) and discount intervals?

- The processing of each product type of each order is started at what time?
- Which vehicles are selected and used for transporting the orders to the agents?
- Which orders are transported by each used vehicle?
- How much safety stock is held in the system for each product type?

A comprehensive MILP model and a heuristic algorithm are developed in the next section. Other assumptions considered in this paper are stated as follows:

- The products are independent of each other,
- The processing of the products is done without any interruption, i.e. no preemption is allowed,
- The processing of all the products can be started at time zero,
- The factory (machine) is continuously available during the planning horizon,
- All parameters of the problem are deterministic and there is no randomness.

2 The proposed solution approaches

In this section, we present two approaches to solve this problem, one based on an MILP optimization and the other based on a heuristic algorithm.

2.1 Proposed mathematical model

In this subsection, a MILP model is presented for the problem. The notations used in this model are defined as follows.

Indices and parameters:

m: number of agents,
n: total number of products,
to_a: number of orders of agent a,
a: index of agents, $a = 1,2,\ldots,m$,
p: index of products, $p = 1,2,\ldots,n$,
o: index of orders, $o = 1,2,\ldots,k$,
v: index of vehicles, $v = 1,2,\ldots,V$,
d_p: index of discount intervals of product p, $d_p = 1,2,\ldots,D_p$,
obj: index of objective functions, $obj = 1,2,3$,
$dist_{max}$: maximum allowed distance between each two agents consecutively served by the same vehicle,
σ_p: weight (importance coefficient) of safety stock of product p,
$g_{aa'}$: a binary parameter taking value 1 if the distance between two agents a and a' is less than or equal to $dist_{max}$ and 0 otherwise,
t_p: processing time of product p,
$vcap_v$: capacity of vehicle v,
cap_p: capacity needed by one unit of product p,
due_{ao}: due time of order o of agent a,
$profit_p^{d_p}$: profit of product p in discount interval d,

tc_{ao}: penalty of unit-time tardiness of order o of agent a,

fc_v: fixed cost of vehicle v,

$st_{pp'}$: setup time between two consecutively processed products p and p',

$lb.ss_p$: lower bound of safety stock level of product p,

e_{pd_p}: dth price break point of product p,

lb_{aop}: minimum required number of product p for order o of agent a,

ub_{ap}: an upper bound on the surplus number of products p sold to agent a,

M: a big number.

Decision variables:

C_{aop}: completion time of product p of order o of agent a,

N_{aop}: number of product p produced and delivered for order o of agent a,

DT_{ao}: delivery time of order o of agent a,

TAR_{ao}: tardiness of order o of agent a,

$ND_{ap}^{d_p}$: number of product p of agent a which is sold at price of discount interval d,

SS_p: value of safety stock of product p,

H_v: a binary variable taking value 1, if vehicle v is selected and used and 0 otherwise,

Y_{aov}: a binary variable taking value 1, if order o of agent a is transported by vehicle v and 0 otherwise,

$Y1_{av}$: a binary variable taking value 1, if one or more orders of agent a is transported by vehicle v, and 0 otherwise,

$\alpha_{aopa'o'p'}$: a binary variable taking value 1, if the processing of product p of order o of agent a precedes the processing of product p' of order o' of agent a', and 0 otherwise,

$X_{ap}^{d_p}$: a binary variable taking value 1, if product p of agent a is sold at price of discount interval d and 0 otherwise.

The proposed mixed integer non-linear programming (MINLP) model is given below:

$$max \quad Z = [\sum_{a}\sum_{p}\sum_{d_p} ND_{ap}^{d_p} \times profit_{p}^{d_p}] - \sum_{a}\sum_{o}^{to_a} tc_{ao} \times TAR_{ao} - \sum_{v} fc_v \times H_v, \tag{1}$$

$$min \quad u_1 = \sum_{v}\sum_{a} Y1_{av}, \tag{2}$$

$$max \quad u_2 = \sum_{p} \sigma_p \times SS_p, \tag{3}$$

such that:

$$\sum_{a}\sum_{o\leqslant to_a}\sum_{p} cap_p \times N_{aop} \times Y_{aov} \leqslant vcap_v; \qquad \forall v, \tag{4}$$

$$DT_{ao} \geqslant C_{a'p'o'} - M(2 - Y_{aov} - Y_{a'o'v}); \qquad \forall a', o', v, o \leqslant a, p', to_a, to_{a'}, \tag{5}$$

$$C_{aop} \leqslant C_{a'o'p'} - ((N_{a'o'p'} - SS_{p'})t_{p'}) - st_{pp'} + M(1 - \alpha_{aopa'o'p'}); \qquad \forall a, p, o \leqslant to_a, a', p', o', p < p', to_{a'}, \tag{6}$$

$$C_{a'o'p'} \leqslant C_{aop} - ((N_{aop} - SS_p)t_p) - st_{p'p} + M \times \alpha_{aopa'o'p'}); \qquad \forall a, p, o \leqslant to_a, a', p', o', p < p', to_{a'}, \tag{7}$$

$$TAR_{ao} \geqslant DT_{ao} - due_{ao}; \qquad \forall a, o \leqslant to_a, \tag{8}$$

$$Y_{aov} \leqslant H_v; \qquad \forall a, o \leqslant to_a, v, \tag{9}$$

$$\sum_{a} Y1_{av} - 1 \leqslant \sum_{a'}\sum_{a} g_{aa'} \times Y1_{a'v} \times Y1_{av}; \qquad \forall v, \tag{10}$$

$$Y_{aov} \leqslant Y1_{av}; \qquad \forall a, o \leqslant to_a, v, \tag{11}$$

$$\sum_{v} Y_{aov} = 1; \qquad \forall a, o \leqslant to_a, \tag{12}$$

$$N_{aop} \geqslant lb_{aop}; \qquad \forall a, p, o \leqslant to_a, \tag{13}$$

$$\sum_{o\leqslant to_a} N_{aop} \leqslant \sum_{o\leqslant to_a} lb_{aop} + ub_{ap}; \qquad \forall a, p, \tag{14}$$

$$\sum_{o\leqslant to_a} N_{aop} = \sum_{d} ND_{ap}^{d_p}; \qquad \forall a, p, \tag{15}$$

$$SS_p \geqslant lb.ss_p; \qquad \forall p, \tag{16}$$

$$(e_{pd_p} - e_{pd_p-1})X_{ap}^{d_p} \leqslant ND_{ap}^{d_p-1} < (e_{pd_p} - e_{pd_p-1})X_{ap}^{d_p-1}; \qquad \forall a, p, d_p, \tag{17}$$

$$TAR_{ao}, SS_p, N_{aop}, ND_{ap}^{d_p}, DT_{ao}, C_{aop} \geqslant 0; \qquad \forall a, p, o \leqslant to_a, \tag{18}$$

$$Y_{aov}, Y1_{av}, H_v, \alpha_{aopa'o'p'}, X_{ap}^{d_p} \in \{0, 1\}; \qquad \forall v, a, p, o \leqslant to_a, d, a', p', o' \leqslant to_{a'}. \tag{19}$$

The relations (1), (2) and (3) are the objective functions. The first objective function (1) is to maximize the total profit taking into account the selling price discounts, the total tardiness costs and the total vehicle costs. The second objective function (2) is the minimization of the number of agents served by each vehicle; and the third objective function (3) is the weighted sum of safety stocks. Constraint (4) is related to the capacity of vehicles and ensures that the total volume of the products carried by each vehicle does not exceed its capacity. Constraint (5) computes the delivery times of orders; the delivery time of each order of each agent is equal to the maximum completion time of the products belonging to this order. Constraints (6) and (7) guarantee that processing of the products do not overlap. Constraint (8) calculates the tardiness of orders. According to constraint (9), orders can be assigned to vehicle v only if the vehicle is selected to be used ($H_v = 1$). By constraint (10), two agents a and a' can be consecutively served by the same vehicle, if and only if the distance between them does not exceed the maximum allowed distance ($dist_{max}$). Constraint (11) is a relationship between variables Y_{aov} and $Y1_{av}$; based on these constrains that the variable $Y1_{av}$ takes value 1, if at least one order of agent a is transported by vehicle v. Constraint (12) ensures that order o of agent a is transported by only one vehicle. Constraint (13) ensures that the number of product p for order o of agent a is not less than its lower limit (lb_{aop}). Constraint (14) takes care of the requirement that the total number of surplus products of type p for agent o is not greater than its upper limit (ub_{ap}). Constraint (15) computes the total number of product type p that is sold to agent a. Constraint (16) guarantees that the value of safety stock of each product type p is not less than its lower bound ($lb.ss_p$). Constraint (17) indicates the discount limit, which represents

the amount of produces that are provided at that discount interval. Constraint (18) is a non-negativity constraint, and constraint (19) defines the binary variables. Overall, in the proposed MINLP model, the maximum number of variables and constriants are

$$(2n+V+2)(\sum_{a=1}^{m} to_a)+2m(\sum_{p=1}^{n} D_p)+n+V+mV+(\frac{n^2-n}{2})(\sum_{a=1}^{m} to_a)^2,$$

and

$$2V+nV(\sum_{a=1}^{m} to_a)^2+2(\frac{n^2-n}{2})(\sum_{a=1}^{m} to_a)^2+(2V+n+2)(\sum_{a=1}^{m} to_a)+2mn+n+m(\sum_{p=1}^{n} D_p),$$

respectively.

2.2 Mathematical model linearization

Now, we linearize constraints (4) and (10) as follows and convert the MINLP model to an MILP model. For constraint (4), we use the following two constraints instead of them:

$$\sum_{v} N'_{aopv} \geqslant N_{aop} \tag{20}$$

$$N'_{aopv} \leqslant M \times Y_{aov} \tag{21}$$

We can also linearize constraint (10) in the proposed model by replacing the non-linear term $Y1_{a'v} \times Y1_{av}$ in constraint (10) with a variable $Y2_{aa'v}$ and adding the following constraints to the model as follows:

$$Y1_{a'v}+Y1_{av}-1 \leqslant Y2_{aa'v} \tag{22}$$

$$Y1_{a'v}+Y1_{av} \geqslant 2 \times Y2_{aa'v} \tag{23}$$

Example 1. A numerical example: In order to demonstrate the performance of the proposed mathematical model, a numerical example is solved by the model and by using CPLEX solver of GAMS software. In this example, it is assumed that there are 4 agents, each of which can have up to 3 orders, along with 6 different product types and 7 vehicles. The values of parameters for this example are given as follows:

$$dist_{max}=500, \quad M=10000, \qquad t_p=[5 \quad 5 \quad 4 \quad 6 \quad 5 \quad 2],$$
$$cap_p=[4 \quad 3 \quad 7 \quad 5 \quad 5 \quad 3], \qquad \sigma_p=[1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1],$$
$$vcap_v=[1000 \quad 1500 \quad 2500 \quad 3000 \quad 3500 \quad 3800 \quad 4200],$$
$$fc_v=[1000 \quad 1500 \quad 3000 \quad 3500 \quad 4000 \quad 4300 \quad 4500],$$
$$lb.ss_p=[10 \quad 10 \quad 10 \quad 10 \quad 10 \quad 10], \qquad to_a=[1 \quad 3 \quad 2 \quad 1],$$

$$g_{aa'} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad due_{ao} = \begin{bmatrix} 500 & - & - \\ 500 & 550 & 800 \\ 650 & 500 & - \\ 390 & - & - \end{bmatrix}, \quad tc_{ao} = \begin{bmatrix} 300 & - & - \\ 260 & 200 & 250 \\ 500 & 260 & - \\ 350 & - & - \end{bmatrix},$$

$$profit_p^{d_p} = \begin{bmatrix} 1800 & 1700 & - & - & - \\ 800 & 700 & 600 & 550 & - \\ 1500 & 1400 & 1300 & 1200 & - \\ 1100 & 1000 & 800 & 700 & 600 \\ 1000 & 950 & - & - & - \\ 2000 & 1800 & - & - & - \end{bmatrix}, \quad e_{pd_p} = \begin{bmatrix} 0 & 70 & - & - & - \\ 0 & 20 & 50 & 100 & - \\ 0 & 10 & 50 & 90 & - \\ 0 & 20 & 30 & 50 & 90 \\ 0 & 80 & - & - & - \\ 0 & 70 & - & - & - \end{bmatrix},$$

$$ub_{ap} = \begin{bmatrix} - & - & 10 & 5 & 10 & - \\ 10 & 30 & - & - & 10 & - \\ 10 & 10 & - & 20 & - & 20 \\ - & 10 & - & 30 & 20 & - \end{bmatrix}, \quad st_{pp'} = \begin{bmatrix} 0 & 3 & 3 & 2 & 0 & 1 \\ 3 & 0 & 6 & 5 & 0 & 1 \\ 3 & 6 & 0 & 2 & 0 & 1 \\ 2 & 5 & 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix},$$

$$lb_{aop} = \begin{bmatrix} - & - & 9 & 5 & 4 & - \\ - & - & - & - & 5 & 2 \\ - & 3 & 11 & 13 & - & - \\ 1 & 8 & 2 & - & 4 & 4 \\ 10 & 12 & - & - & - & - \\ 5 & 5 & - & 3 & 10 & 9 \\ 1 & 1 & - & 9 & 10 & - \end{bmatrix}.$$

The results of solving the example by CPLEX solver of GAMS software are as follows:

$$TAR_{ao} = \begin{bmatrix} 17.000 & - & - \\ 35.333 & 0 & 0 \\ 0 & 79.500 & - \\ 19.000 & - & - \end{bmatrix}$$

$$SS_p = (10 \quad 10 \quad 10 \quad 10 \quad 49695.500), \quad Z = 279193.333, \quad u_1 = 4.0, \quad u_2 = 49745.5$$

In this solution, vehicles 1, 2 and 3 are selected and used to transport the orders of agents (see Figure 1).

2.3 Proposed heuristic algorithm

Since the proposed model has many numbers of variables and constraints, solving it directly by GAMS is time consuming, thus in this subsection, a heuristic algorithm is proposed to solve the problem. The details of the algorithm are presented in Algorithm 1. It should be noted that this algorithm improves only the first objective function (Z) which is the most important one. The algorithm is based on a greedy process and implemented in three phases, and the maximum amount of profit obtained from these three phases and corresponding solution is considered as its output. In the first phase of the above algorithm, the minimum amount of products (lb_{aop}) is assigned to each order and corresponding total profit (Z_1) is

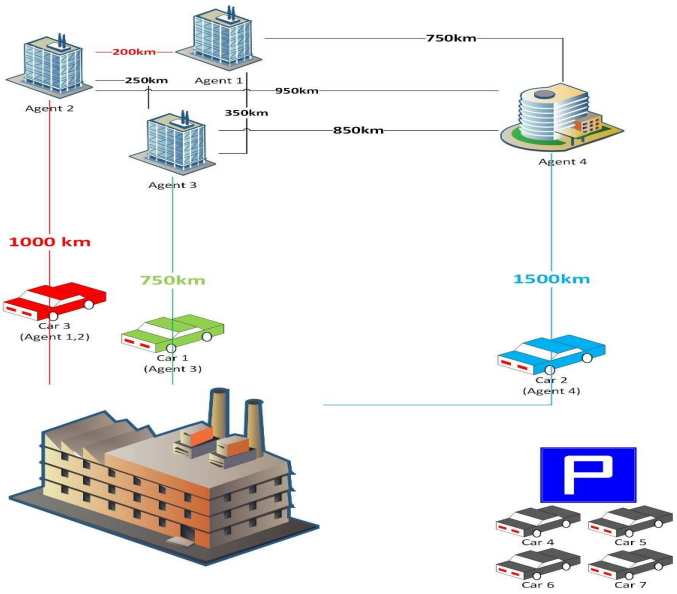

Figure 1: Optimal solution of the numerical example (selected vehicles and their routes).

calculated. In the second phase, the minimum amount of products (lb_{aop}) plus the maximum amount of products (ub_{ap}) is assigned to each order and corresponding total profit (Z_2) is calculated. Finally, in the third phase, starting with the minimum amount of products (lb_{aop}) assigned to each order, in an iterative process, one unit is added to a product of an order selected to be sent (if it does not exceed the maximum allowed number of that product per agent) at each step. This is done by using an auxiliary matrix am, which is a zero matrix in first iteration, and only one unit is added to the sum of its elements in each next iteration (if the value of each matrix element does not exceed the value of corresponding element in matrix ub_{ap}). Each of these iterations is considered as a feasible solution and its objective value (total profit) is calculated. Z_3 is considered as the best objective value obtained from this phase. At the end, the best objective value obtained from the above three phases and corresponding solution is considered as the output of the algorithm.

We use the following auxiliary symbols in the algorithm:

$T_{o_i}, T_{max-o_i}, T_{am-o_i}$: processing time of order o_i in phases 1, 2 and 3, respectively,

$C_{o_i}, C_{max-o_i}, C_{am-o_i}$: the total capacity of order o_i in phases 1, 2 and 3, respectively,

$[am]_{A\times P}$: auxiliary matrix of size $A \times P$,

am_{ap}: element ap of matrix am.

The above values are obtained from the following equations:

$$T_{o_i} = \sum_a \sum_p lb_{ao_ip} \times t_p, \qquad C_{o_i} = \sum_a \sum_p lb_{ao_ip} \times cap_p,$$

$$T_{max-o_i} = \sum_a \sum_p (lb_{ao_ip} + ub_{ap}) \times t_p, \qquad C_{max-o_i} = \sum_a \sum_p (lb_{ao_ip} + ub_{ap}) \times cap_p,$$

$$T_{am-o_i} = \sum_a \sum_p (lb_{ao_ip} + am_{ap}) \times t_p, \qquad C_{am-o_i} = \sum_a \sum_p (lb_{ao_ip} + am_{ap}) \times cap_p.$$

We use these values at the beginning of each phase of the algorithm as a criterion to select and assign an order to a vehicle.

Algorithm 1. *Heuristic Algorithm*

Input:

a: index of agents, $a = 1,2,\ldots,m$,
p: index of products, $p = 1,2,\ldots,n$,
o: index of orders, $o = 1,2,\ldots,k$,
v: index of vehicles, $v = 1,2,\ldots,V$,
d_p*: index of discount intervals of product p,* $d_p = 1,2,\ldots,D_p$,
t_p*: processing time of product p,*
$vcap_v$*: capacity of vehicle v,*
cap_p*: capacity needed by one unit of product p,*
due_{ao}*: due time of order o of agent a,*
$profit_p^{d_p}$ *: profit of product p in discount interval d,*
tc_{ao}*: penalty of unit-time tardiness of order o of agent a,*
fc_v*: fixed cost of vehicle v,*
e_{pd_p} *: dth price break point of product p,*
lb_{aop}*: minimum required number of product p for order o of agent a,*
ub_{ap}*: an upper bound on the surplus number of product p sold to agent a,*
$g_{aa'}$*: a binary parameter taking value 1, if the distance between two agents a and* a' *is less than or equal to* $dist_{max}$ *and* 0 *otherwise,*

Output:

Z_A*: first objective function (Z).*

/* Phase 1 (Steps 1-4) */

1: Set $Z_1 = 0$.
2: Assign the minimum number of products (lb_{aop}*) to each order o of agent a and then compute the total processing time* $\{T_{o_1}, T_{o_2}, \ldots, T_{o_k}\}$ *and the total capacity* $\{C_{o_1}, C_{o_2}, \ldots, C_{o_k}\}$ *required for all k orders* $\{o_1, o_2, \ldots, o_k\}$.
3: ***for*** *i=1 to k*
Find order o_i *with minimum processing time value (*T_{o_i}*). Then, find a vehicle* v_t *whose capacity is nearest to the capacity of order* o_i.
Compute $Z_1 = Z_1 + \sum_p ND_{ap}^{d_p} \times profit_p^{d_p} - fc_{v_t} - tc_{ao_i} \times TAR_{ao_i}$, *where* $ND_{ap}^{d_p}$ *is the number of product p of agent a, which is sold at price of discount status d.*
4: ***end for***
/* Phase 2 (Steps 5-8) */
5: Set $Z_2 = 0$.
6: Assign the minimum (lb_{aop}*) plus the maximum (*ub_{ap}*) number of products to each order and then compute the total processing time* $\{T_{max-o_1}, T_{max-o_2}, \ldots, T_{max-o_k}\}$ *and the total capacity* $\{C_{max-o_1}, C_{max-o_2}, \ldots, C_{max-o_k}\}$ *required for each order,*
7: ***for*** $i = 1$ *to k*
Find order o_i *with minimum processing time value (*T_{max-o_i}*). Then, find a vehicle* v_t *whose capacity is nearest to the capacity of order* o_i.
Given the discount status d, compute $Z_2 = Z_2 + \sum_p ND_{ap}^{d_p} \times profit_p^{d_p} - fc_{v_t} - tc_{ao_i} \times TAR_{ao_i}$.
8: ***end for***
/* Phase 3 (Steps 9-19) */
9: Define auxiliary matrices $[am]_{A\times P}$, $[am']_{A\times P}$ *and set* $[am]_{A\times P} = 0$.
10: Set $Z_3 = Z_1$, $Z' = Z_3$.
11: ***while*** $(\sum_i^A \sum_j^P am_{ij} \leqslant \sum_i^A \sum_j^P ub_{ij})$
12: Set $Z_3 = 0$.
13: Construct a matrix am', such that $\sum_i^A \sum_j^P am'_{ij} \leqslant \sum_i^A \sum_j^P am_{ij} + 1$.
14: Set $[am]_{A\times P} = [am']_{A\times P}$.
15: Assign the minimum number of products (lb_{aop}*) plus the auxiliary matrix (*am_{ap}*) to each order o*

and then compute the total processing time $\{T_{am-o_1}, T_{am-o_2}, \ldots, T_{am-o_k}\}$ *and the total capacity* $\{C_{am-o_1}, C_{am-o_2}, \ldots, C_{am-o_k}\}$ *required for each order.*

16: ***for*** $i = 1$ *to* k

Find order o_i *with minimum processing time value* (T_{am-o_i}). *Then, find a vehicle* v_t *whose capacity is nearest to the capacity of order* o_i.

Given the discount status d, compute $Z_3 = Z_3 + \sum_p ND_{ap}^{d_p} \times profit_p^{d_p} - fc_{v_t} - tc_{ao_i} \times TAR_{ao_i}$.

17: ***end for***

18: ***if*** $(Z_3 \geqslant Z')$ ***then*** *set* $Z' = Z_3$.

19: ***end of while***

20: Set $Z_3 = Z'$.

21: $Z_A = Max(Z_1, Z_2, Z_3)$.

22: ***Return*** Z_A *as output.*

Now, we analyze the time complexity of the proposed algorithm. Let the total number of orders be $N = k$. In all three phases of the algorithm, the calculations of processing times (T_{o_i}) and capacities (C_{o_i}) for all orders take a time of $O(N)$. As mentioned earlier, in the first and second phases, for each order, a suitable vehicle is found at each step and the order is assigned to it; this has the running time of $O(N)$ for each order, and so needs time of $O(N^2)$ for all orders. The third phase also needs the same time, except that the while loop is repeated τ times, therefore the running time of the third phase will be $O(\tau N^2)$, where $\tau = \sum_i^A \sum_j^P ub_{ij}$, and since it is a constant value, the total running time of the heuristic algorithm will be $O(N^2)$.

3 Experimental results

In this section, we compare the computational results of two approaches (i.e., MILP and the heuristic algorithm) on a variety of datasets. In these datasets, the number of agents (m) is from 2 to 20, the number of orders (k) is from 1 to 5, the number of products (n) is from 4 to 50, and the number of available vehicles (V) is from 3 to 25. The values of parameters for these datasets are random integers from the uniform distribution in ranges given in Table 1. We implement our proposed heuristic algorithm in C++ programming language. The mathematical model is also run in GAMS software. Both the mathematical model and the heuristic algorithm are run on an Intel (R) Core (TM) i7 CPU and 12 GB RAM computer with the Windows 10 operating system.

These results of computational studies are presented in Tables 2, 3 and 4. In these tables, column m represents the number of agents, column k represents the maximum number of orders for each agent, column n represents the number of products, and column V represents the number of available vehicles. The results of solving the mathematical model by GAMS software and the heuristic algorithm are given in columns GAMS and heuristic, respectively, in terms of the value of the objective function (Z), the execution time (Time (s)) and Gap. The column (Gap percent) shows the percentage error of the objective function value of the corresponding method (i.e. GAMS or heuristic) that is calculated as follows:

$$Gap = \frac{(Z_{Best} - Z_M)}{Z_{Best}} \times 100.$$

In the above relation, Z_M is the objective function value of the corresponding method (i.e. GAMS or heuristic) and Z_{Best} is the best objective value obtained by the two approaches for the considered instance. All running times in seconds are also reported in column Time(s). Since optimally solving the larger size instances by GAMS software was very time-consuming, the running time of each instance was limited

Table 1: The ranges of parameter values for generated datasets.

Parameter	Range
t_p	1-50
cap_p	10-50
$vcap_v$	1000-10000
fc_v	1000-10000
to_a	1-5
$profit_p^{d_p}$	1000-5000
ub_{ap}	10-100
lb_{aop}	1-50
$st_{pp'}$	0-10
due_{ao}	100-1000
σ_p	1-4
D_p	1-5
$dist_{max}$	500
tc_{ao}	100-1000
$lb.ss_p$	10-100
e_{pd_p}	0-200

to 3600 CPU seconds. In Tables 2, 3 and 4, non-bold rows denote the optimal solutions obtained by the GAMS and **bold rows** denote the best solutions obtained by the GAMS after 3600 seconds.

Herein, we statistically compare the results of the heuristic and the GAMS by some statistical experiments performed by the statistical software SPSS. First, using the results of the three instances set (i.e. instances 1-30, 31-60 and 61-100), three one-way analysis of variance (ANOVA) tests are performed to test the null hypothesis that the means of the two approaches (i.e. the heuristic and the GAMS), are equal at a significance level of 5 percent. The results of these ANOVA tests are given in Tables 5, 6 and 7, respectively. We also perform the same ANOVA test for all 100 instances and report its results in Table 8. As it can be seen from all these tables, there is no significance difference between the means of the two approaches for all three instance sets. Taking into account that the running time of the heuristic is much less than the GAMS and less than 2 seconds, we can conclude that the heuristic has better performance and can be applied to generate good-quality and competitive solutions for real-world problems in very short execution time.

To see the differences between the two approaches (i.e. the heuristic and the GAMS) for different values of m, k, n and V, we now investigate them graphically. Figures 2, 3, 4 and 5 show a graphical comparison of the average *Gap* percent of the two methods for different values of these parameters. Therefore, for each value of these parameters (i.e. m, k, n and V), the *Gap* percent reported in these figures is average *Gap* percent of instances corresponding to that value among all 100 instances. As it can be seen from these figures, the results of heuristic algorithm are totally better than those of GAMS. In Figures 2, 4 and 5, the heuristic generates better solutions than the GAMS for larger size instances, but for Figure 3, the GAMS has better performance for larger instances. As shown in this figure, the heuristic does not generate good-quality solutions for larger values of k.

Table 2: Experimental results for the datasets with 2, 3 and 4 agents.

					GAMS			Heuristic		
Instance	m	k	n	V	Z_G	Time(s)	Gap	Z_H	Time(s)	Gap
1		1	4	3	67700	0.03	0.00	67700	0	0.00
2		2	5	3	169450	15	0.00	169450	0	0.00
3		2	6	3	219700	330	0.00	218400	0	0.59
4		2	8	3	186450	35	0.00	186450	0	0.00
5	2	2	10	3	171500	584	0.00	171500	0	0.00
6		2	12	3	198800	2481	0.00	189600	0	4.63
7		3	4	3	140800	0.08	0.00	140800	0	0.00
8		3	5	4	170500	253	0.00	165000	0	3.23
9		3	6	4	192200	1892	0.00	189500	0	1.40
10		**3**	**8**	**4**	**176300**	**3600**	**0.00**	**170000**	**0**	**3.57**
11		1	4	3	165320	0.6	0.00	165320	0	0.00
12		1	5	3	195000	7.3	0.00	195000	0	0.00
13		1	6	3	191000	526	0.00	189500	0	0.79
14		2	5	4	200450	34	0.00	200450	0	0.00
15	3	2	6	4	201700	766	0.00	194300	0	3.67
16		2	8	4	220140	2432	0.00	210600	0	4.33
17		**2**	**10**	**4**	**215950**	**3600**	**0.00**	**200950**	**0**	**6.95**
18		**3**	**6**	**4**	**81690**	**3600**	**0.00**	**76690**	**0**	**6.12**
19		**3**	**7**	**4**	**101800**	**3600**	**0.00**	**95210**	**0**	**6.47**
20		**3**	**8**	**4**	**95760**	**3600**	**0.00**	**92220**	**0**	**3.70**
21		1	4	4	227500	3	0.00	227500	0	0.00
22		1	5	4	230220	111	0.00	228300	0	0.83
23		1	6	4	235500	648	0.00	229500	0	2.55
24		2	4	5	192600	3116	0.00	183600	0	4.67
25	4	**2**	**6**	**5**	**226400**	**3600**	**0.00**	**212200**	**0**	**6.27**
26		**2**	**8**	**5**	**239220**	**3600**	**0.00**	**215330**	**0**	**9.99**
27		**3**	**5**	**5**	**95520**	**3600**	**0.00**	**90500**	**0**	**5.26**
28		**3**	**6**	**5**	**71540**	**3600**	**0.00**	**70520**	**0**	**1.43**
29		**3**	**8**	**5**	**72480**	**3600**	**0.00**	**68050**	**0**	**6.11**
30		**3**	**10**	**5**	**105500**	**3600**	**0.00**	**95500**	**0**	**9.48**

In summary, the results of computational experiments show that the proposed heuristic algorithm can generate good-quality and competitive solutions, even for large size instances, in very short running time (less than 2 seconds).

4 Conclusions

The main idea of this research is to develop a simple and integrated scheduling and transportation model which can be applied in many factories, packing and distribution centers, chain stores, and so on. We present a mixed integer linear programming model for the problem. Due to the high complexity of the problem, we also develop a new heuristic algorithm to solve large–scale instances of the problem. The

Table 3: Experimental results for the datasets with 5, 6 and 7 agents.

					GAMS			Heuristic		
Instances	m	k	n	V	Z_G	Time(s)	Gap	Z_H	Time(s)	Gap
31		1	4	4	242300	6.8	0.00	242300	0	0.00
32		2	4	4	331480	152.5	0.00	331480	0	0.00
33		2	6	4	372600	336	0.00	370020	0	0.69
34		2	8	4	255000	24.1	0.00	255000	0	0.00
35	5	3	10	4	299350	921	0.00	280500	0	6.30
36		**3**	**12**	**4**	**322470**	**3600**	**0.00**	**311000**	**0**	**3.56**
37		3	4	5	301450	2.7	0.00	301450	0	0.00
38		4	5	5	277600	119	0.00	273200	0	1.59
39		4	6	5	194420	2855	0.00	180150	0	7.34
40		**4**	**8**	**5**	**196000**	**3600**	**0.00**	**185500**	**0**	**5.36**
41		1	4	4	285220	20.9	0.00	285220	0	0.00
42		1	5	4	287720	89.2	0.00	287720	0	0.00
43		2	6	4	272320	302	0.00	271150	0	0.43
44		2	8	5	355000	652	0.00	342600	0	3.49
45	6	3	6	5	175520	1502	0.00	163200	0	7.02
46		3	8	5	186400	2157	0.00	174000	0	6.65
47		**3**	**10**	**5**	**188450**	**3600**	**0.00**	**173520**	**0**	**7.92**
48		**4**	**6**	**6**	**152300**	**3600**	**0.00**	**139500**	**0**	**8.40**
49		**4**	**7**	**6**	**155520**	**3600**	**0.00**	**148500**	**0**	**4.51**
50		**4**	**8**	**6**	**164800**	**3600**	**0.00**	**156900**	**0**	**4.79**
51		1	4	5	215000	7.7	0.00	215000	0	0.00
52		1	5	5	220800	25.3	0.00	220800	0	0.00
53		2	6	5	187500	860	0.00	181500	0	3.20
54		**2**	**4**	**5**	**95020**	**3600**	**0.00**	**86500**	**0**	**8.97**
55	7	**2**	**6**	**6**	**112500**	**3600**	**0.00**	**108080**	**0**	**3.93**
56		**2**	**8**	**6**	**153600**	**3600**	**0.00**	**143600**	**0**	**6.51**
57		**3**	**6**	**6**	**336000**	**3600**	**0.00**	**318900**	**0**	**5.09**
58		**3**	**8**	**6**	**300850**	**3600**	**0.00**	**286600**	**0**	**4.74**
59		**4**	**8**	**6**	**122000**	**3600**	**0.00**	**113000**	**0**	**7.38**
60		**4**	**10**	**6**	**152550**	**3600**	**0.00**	**139200**	**0**	**8.75**

results of computational experiments show that the proposed heuristic algorithm has good performance and can obtain good–quality solutions in comparison with the solutions obtained by MILP model in GAMS software, especially for large–scale instances. Some directions for future work are suggested as follows:

– Solving the model with different meta–heuristic algorithms,

– Taking into account other discount models such as continuous discount models,

– Applying the vehicle routing models to more decrease the transportation costs,

– Applying the inventory control models,

– Applying the time periods.

Table 4: Experimental results for the datasets with 8 to 20 agents.

					GAMS			Heuristic		
Instances	m	k	n	V	Z_G	Time(s)	Gap	Z_H	Time(s)	Gap
61		1	6	6	320750	166	0.00	319900	0	0.27
62		2	6	6	355300	702	0.00	345000	0	2.90
63		2	8	6	352000	1524	0.00	343500	0	2.41
64		**3**	**8**	**7**	**272500**	**3600**	**0.00**	**261100**	**0**	**4.18**
65	8	**3**	**10**	**7**	**288500**	**3600**	**0.00**	**269420**	**0**	**6.61**
66		**4**	**8**	**7**	**312500**	**3600**	**0.00**	**300800**	**0**	**3.74**
67		**4**	**9**	**7**	**313400**	**3600**	**0.00**	**301700**	**0**	**3.73**
68		**4**	**10**	**7**	**355500**	**3600**	**0.00**	**340500**	**0**	**4.22**
69		**5**	**6**	**8**	**186620**	**3600**	**0.00**	**169800**	**0**	**9.01**
70		**5**	**8**	**8**	**214500**	**3600**	**0.00**	**200240**	**0**	**6.65**
71		1	6	7	155000	330	0.00	155000	0	0.00
72		2	6	7	125000	2444	0.00	116500	0	6.80
73		**2**	**8**	**7**	**147750**	**3600**	**0.00**	**144000**	**0**	**2.54**
74		**3**	**8**	**7**	**227000**	**3600**	**0.00**	**215050**	**0**	**5.26**
75	9	**3**	**10**	**8**	**76200**	**3600**	**1.68**	**77500**	**0**	**0.00**
76		**4**	**8**	**8**	**166500**	**3600**	**0.00**	**159000**	**0**	**4.50**
77		**4**	**10**	**8**	**175800**	**3600**	**0.00**	**166200**	**0**	**5.46**
78		**5**	**8**	**8**	**262500**	**3600**	**0.00**	**250700**	**0.1**	**4.50**
79		**5**	**9**	**9**	**368200**	**3600**	**0.46**	**369900**	**0.5**	**0.00**
80		**5**	**10**	**9**	**357500**	**3600**	**0.00**	**350000**	**0.4**	**2.10**
81		1	8	9	122400	412	0.00	122400	0	0.00
82		**2**	**8**	**9**	**102500**	**3600**	**0.00**	**99500**	**0**	**2.93**
83		**2**	**10**	**9**	**122250**	**3600**	**6.14**	**130250**	**0**	**0.00**
84		**3**	**8**	**9**	**151000**	**3600**	**0.00**	**150050**	**0.3**	**0.63**
85		**3**	**10**	**9**	**169300**	**3600**	**1.86**	**172500**	**0.8**	**0.00**
86	10	**4**	**6**	**10**	**295000**	**3600**	**0.00**	**290000**	**0.8**	**1.69**
87		**4**	**8**	**10**	**288560**	**3600**	**2.35**	**295500**	**0.8**	**0.00**
88		**4**	**10**	**10**	**342200**	**3600**	**8.13**	**372500**	**0.2**	**0.00**
89		**5**	**8**	**10**	**186000**	**3600**	**6.74**	**199450**	**0.5**	**0.00**
90		**5**	**10**	**10**	**180400**	**3600**	**4.80**	**189500**	**0.8**	**0.00**
91		**1**	**20**	**10**	**1000020**	**3600**	**5.21**	**1055000**	**0**	**0.00**
92		**2**	**20**	**10**	**1338000**	**3600**	**12.15**	**1523000**	**0.2**	**0.00**
93	12	**1**	**20**	**15**	**1922000**	**3600**	**10.42**	**2145500**	**0.1**	**0.00**
94		**2**	**25**	**15**	**1565800**	**3116**	**8.89**	**1718500**	**0**	**0.00**
95	15	**1**	**25**	**18**	**2336430**	**3600**	**8.70**	**2559200**	**0**	**0.00**
96		**2**	**30**	**18**	**2002790**	**3600**	**14.18**	**2333650**	**1.2**	**0.00**
97	17	**2**	**25**	**20**	**1738300**	**3600**	**6.84**	**1866000**	**0.9**	**0.00**
98		**3**	**30**	**20**	**3237700**	**3600**	**16.32**	**3869200**	**1.8**	**0.00**
99	20	**2**	**40**	**25**	**4292230**	**3600**	**10.79**	**4811150**	**1.1**	**0.00**
100		**3**	**50**	**25**	**Out of memory**	**3600**	**–**	**3620050**	**1.6**	**0.00**

References

[1] A. Aydilek, H. Aydilek, A. Allahverdi, *Minimising maximum tardiness in assembly flowshops with setup times*, Int. J. Prod. Res. **55** (2017) 7541–7565.

Table 5: Results of one-way ANOVA for the first instance set (i.e. instances 1-30).

Source	DF	SS	MS	F	P
Factor	1	141.067	147.067	30.000	0.000
Error	29	132.000	4.000		

Table 6: Results of one-way ANOVA for the second instance set (i.e. instances 31-60).

Source	DF	SS	MS	F	P
Factor	1	226.000	226.000	44.000	0.000
Error	29	147.000	5.099		

Table 7: Results of one-way ANOVA for the second instance set (i.e. instances 61-100).

Source	DF	SS	MS	F	P
Factor	1	264.000	264.000	1.000	0.045
Error	39	5566.000	142.000		

Table 8: Results of one-way ANOVA for all 100 instances.

Source	DF	SS	MS	F	P
Factor	1	19.000	19.000	0.000	0.003
Error	99	6459.079	65.000		

[2] Y. Adulyasak, J.F. Cordeau, R. Jans, *The production routing problem: a review of formulations and solution algorithms*. Comput. Oper. Res. **55** (2015) 141–152.

[3] K.R. Baker, J.C. Smith, *A multiple–criterion model for machine scheduling*. J. Scheduling **6** (2003) 7–16.

[4] Y. Boulaksil, *Safety stock placement in supply chains with demand forecast updates*, Oper. Res. **3** (2016) 27–31.

[5] R. Chen, J. Yuan, *Unary NP–hardness of single–machine scheduling to minimize the total tardiness with deadlines*, J. Scheduling **22** (2019) 595–601.

[6] R.R. Chen, L.W. Robinson, *Optimal multiple–breakpoint quantity discount schedules for customers with heterogeneous demands: all–unit or incremental?* IIE Trans. **44** (2012) 199–214.

[7] C.Y. Cheng, S.F. Li, K.C. Ying, Y.H. Liu, *Scheduling jobs of two competing agents on a single machine*, IEEE ACCESS, **7** (2019) 98702–98714.

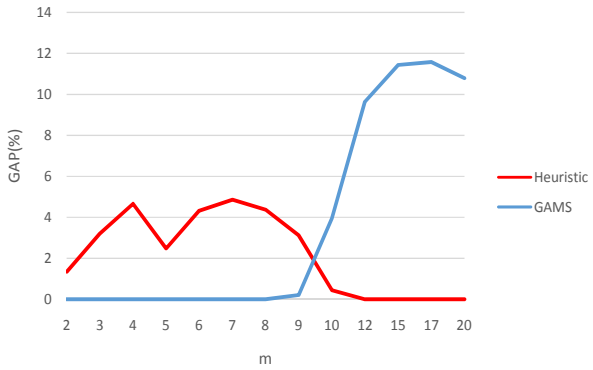

Figure 2: Comparison of the average *Gap* percent of the proposed heuristic algorithm and the GAMS software for different values of *m* (number of agents).

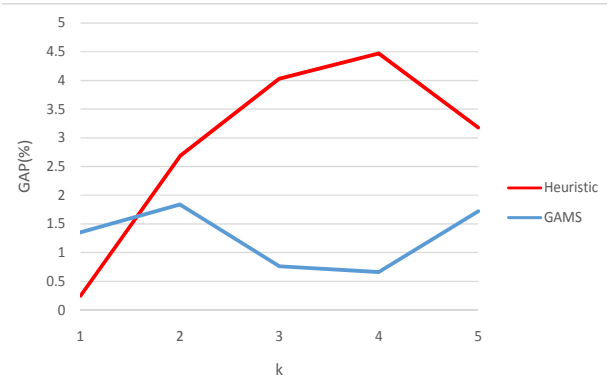

Figure 3: Comparison of the average *Gap* percent of the proposed heuristic algorithm and the GAMS software for different values of *k* (maximum number of orders per agent).

[8] Z.L. Chen, *Integrated production and outbound distribution scheduling: review and extensions*, Oper. Res. **58** (2010) 130–148.

[9] M. Dai, D. Tang, A. Giret, M.A. Salidol, *Multi–objective optimization for energy–efficient flexible job shop scheduling problem with transportation constraints*, Robot. Cim–Int. Manuf. **59** (2019) 143–157.

[10] M. Fabri, H. Ramalhinho, M.C. de Souza, M.G. Ravetti, *The Lagrangean relaxation for the flow shop scheduling problem with precedence constraints, release dates and delivery times*, J. Adv. Transport. **2019** (2019) 3176074 .

[11] X. Feng, Z. Xu, *Integrated production and transportation scheduling on parallel batch–processing machines*. IEEE ACCESS **7** (2019) 148393–148400.

[12] A. Gharaei, F. Jolai, *A branch and price approach to the two–agent integrated production and distribution scheduling*, Comput. Ind. Eng. **136** (2019) 504–515.

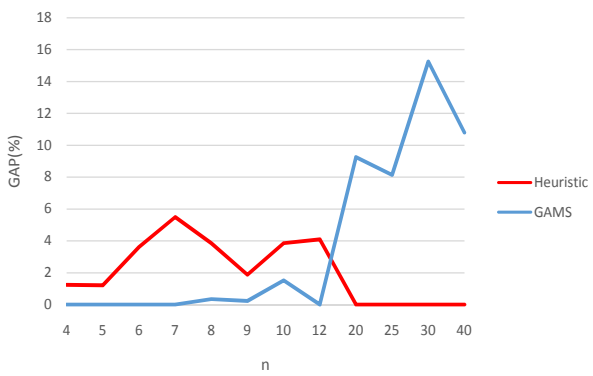

Figure 4: Comparison of the average *Gap* percent of the proposed heuristic algorithm and the GAMS software for different values of n (number of products)

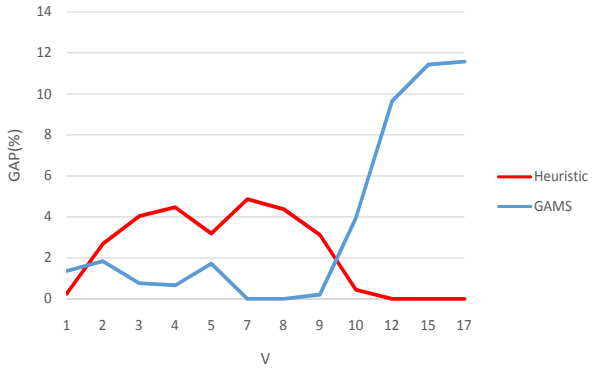

Figure 5: Comparison of the average Gap percent of the proposed heuristic algorithm and the GAMS software for different values of V (number of available vehicles).

[13] M. Gu, X. Lu, J. Gu, *An approximation algorithm for multi–agent scheduling on two uniform parallel machines*, Opt. Lett. **13** (2019) 907–933.

[14] J. Hartmanis, *Computers and intractability: A guide to the theory of NP–completeness* (Michael R. Garey and David S. Johnson), SIAM Rev. **24** (1982) p.90.

[15] X. He, S. Dong, N. Zhao, *Research on rush order insertion rescheduling problem under hybrid flow shop based on NSGA–III*, Int. J. Prod. Res. **58** (2020) 1161–1177.

[16] J. Heger, J. Branke, *Dynamic adjustment of dispatching rule parameters in flow shops with sequence dependent set–up times*, Int. J. Prod. Res. **54** (2016) 6812–6824.

[17] O. Herr, A. Goel, *Minimising total tardiness for a single machine scheduling problem with family setups and resource constraints*, Eur. J. Oper. Res. **248** (2016) 123–135.

[18] P. Jonsson, S.A. Mattsson, *An inherent differentiation and system level assessment approach to inventory management: A safety stock method comparison*, Int. J. Logist. Manag. **30** (2019) 663–680.

[19] C.W. Kang, M. Ullah, B. Sarkar, *Optimum ordering policy for an imperfect single–stage manufacturing system with safety stock and planned backorder*, Int. J. Adv. Manuf. Tech. **95** (2018) 109–120.

[20] V. Kaplanoglu, *Multi–agent based approach for single machine scheduling with sequence–dependent setup times and machine maintenance*, Appl. Soft. Comput. **23** (2014) 165–179.

[21] P. Liu, M. Gu, G. Li, *Two–agent scheduling on a single machine with release dates*. Comput, Oper. Res. **111** (2019) 35–42.

[22] D. C. Li, P. H. Hsu, *Solving a two–agent single–machine scheduling problem considering learning effect*, Comput. Oper. Res. **39** (2012) 1644–1651.

[23] S. Mustu, T. Eren, *The single machine scheduling problem with sequence–dependent setup times and a learning effect on processing times*, Appl. Soft Comput. **71** (2018) 291–306.

[24] H. Pang, *Modeling and optimizing of distributed multi–factory production and transportation coordinative scheduling problem*, In Chinese Control And Decision Conference (2019) 3848–3853.

[25] L. Liu, W. Li, K. Li, X. Zou, *A coordinated production and transportation scheduling problem with minimum sum of order delivery times*, J. Heuristics **26** (2020) 33–58.

[26] M. Imanparast, V. Kiani, *A practical heuristic for maximum coverage in large-scale continuous location problem*, J. Math. Model. **9**(4) (2021) 555–572.

[27] J. Pei , X. Liu , P.M. Pardalos , W. Fan , S. Yang, *Scheduling deteriorating jobs on a single serial–batching machine with multiple job types and sequence–dependent setup times*, Ann. Oper. Res. **249** (2017) 175–195.

[28] N.H. Tuong, J.C. Billaut, *Single–machine multi–agent scheduling problems with a global objective function*, J. Scheduling **15** (2012) 311–321.

[29] Y. Yin, *Two–agent single–machine scheduling with release times and deadlines*, Int. J. Ship. Trans. Logist. **5** (2013) 75–94.

[30] D.Y. Wang, O. Grunder, A.E. Moudni, *Integrated scheduling of production and distribution operations: A review*, Int. J. Ind. Syst. Eng. **19** (2015) 94–122.

[31] L. Lu, L. Zhang, J. Zhang, L. Zuo, *Single machine scheduling with outsourcing under different fill rates or quantity discount rates*, Asia Pac. J. Oper. **37** (2020) p.1950033.

[32] L. Lu, L. Zhang, J.W. Ou, *In–house production and outsourcing under different discount schemes on the total outsourcing cost*, Ann. Oper. Res. **298** (2021) 361–374.