International Journal of
Operations Research

International Journal of Operations Research Vol. 18, No. 1, 11-27 (2021)

Multi-Customer Joint Replenishment Problem with Districting Consideration and Material Handling Capacity Constraints

Ming-Jong Yao[1*], Yu-Liang Lin[2], Bo-Kai Jhang[3]

[1,2,3]Department of Transportation and Logistics Management
National Yang Ming Chiao Tung University, Hsinchu 30010, Taiwan

Received August 2020; Revised December 2020; Accepted March 2021

Abstract: This study investigates the multi-customer Joint Replenishment Problem with districting considerations and material handling capacity constraints (MJRPDC-MHCC). Solving MJRPDC-MHCC is important to the manager before signing a contract with a third-party logistics service provider since it obtains a logistics plan providing lot-sizing and districting decisions. We propose a solution approach based on genetic algorithm (GA) as its framework and incorporate a heuristic that generates a feasible replenishment schedule for each districting setting. Following our numerical experiments, we observe that the average runtime of our approach grows in a cubic order with respect to both the number of customers and the number of items. Therefore, the proposed GA-based solution approach is recommended as a useful decision-support tool.
Keyword — Joint replenishment, districting problem, material handling, capacity constraints, genetic algorithm

1. INTRODUCTION

Enterprises take advantage of advanced logistic services and inventory control to improve supply chain management and to strengthen their core competencies for long-term profits; see Chopra and Meindl (2015) and Simchi-Levi, Chen, and Bramel (2014). This study investigates a logistics planning problem integrating lot-sizing and districting considering limited available capacity.

Joint replenishment has been popularly adopted to reduce the unit setup cost of order picking, packaging, preparation, and dispatch via economies of scale. The joint replenishment problem (JRP) is concerned with the determination of the lot size and the delivery schedule of n items of goods supplied from a single supplier so as to minimize the average total cost (typically, including setup and holding cost) over an infinite (and continuous) planning horizon. Usually, there are two categories of setup costs: (i) a major setup cost, A_0 , for collectively handling a subset of items and (ii) a minor setup cost, a_i , processing for each item i. It is common that a major setup involves a considerable amount of time and costs, and managers utilize JRP to synchronize the replenishment schedule of each item i to share the major setup cost to balance the inventory holding cost of each item.

JRP has been studied for more than half of a century since the early work of Shu (1971). Cohen-Hillel and Yedidsion (2018)show that the periodic JRP is strongly NP-hard recently. Researchers, e.g., Fung and Ma (2001), Goyal (1974), Van Eijs (1993), Viswanathan (2002), etc., proposed solution approaches for JRP by enumerating combinations of the basic cycle time B and its multiples, k_i of item i, to search for an optimal solution. Lee and Yao (2003) investigate the optimality structure of the objective function for JRP under the power-of-two (PoT) policy that mandates $k_i = 2^p$ for some $p \in \mathbb{N}$. They illustrate that the optimal objective function is piece-wise convex and propose a search algorithm to obtain a global optimum of JRP under the PoT policy. Their theoretical analysis establishes important foundation to investigate the general integer (GI) policy that requires each k_i to be a positive integer. One may refer to Khouja and Goyal (2008) and Bastos, Mendes, Nunes, Melo, and Carneiro (2017) for a thorough review of JRP. The logistic managers may take advantage of the joint replenishment and extend it to cases with multiple customers if many customers order the same group of items from a single supplier; for example, a chain of branches or retail stores in an enterprise group. We call such an extension of JRP a multi-customer joint replenishment problem (MJRP). Following some practical concerns such as geography, transportation, or fleet size, managers usually divide the customers into various mutually exclusive zones of a planning region. Then, they need to solve MJRP to coordinate the replenishment for customers in the same zone. Some studies proposed solution approaches for solving MJRP. For instance, Chan,

*Corresponding author's e-mail: ilinwang@mail.ncku.edu.tw

http://doi.org/10.6886/IJOR.202103_18(1).0002

Cheung, and Langevin (2003) solve MJRP in each zone corresponding to eight predetermined groups (Figure 1.1) using a genetic algorithm.Yao, Lin, Y., Lin, and Fang (2020) bring up a search algorithm that effectively solves MJRP for a given zone.

Yao et al. (2020) investigate a multi-customer JRP with districting consideration (MJRPDC), which determines an optimal districting setting such that the average total cost of MJRP for all zones is minimized. Note that most of the related studies in the literature solve MJRP for given partitions, while MJRPDC views the districting as a decision variable and divides customers into a fixed number of groups before solving MJRP.

As emphasized in Yao et al. (2020), MJRPDC is of particular importance to a company that outsources its transportation and delivery operations to a third-party logistics (3PL) service provider. A company must not only negotiate the freight rate in the contract with a 3PL service provider, but also offer a districting plan with a replenishment schedule to cover all customers before signing an outsourcing contract.

Ricca, Scozzari, and Simeone (2013) present three commonly used principles of districting:

(i) Contiguity: A zone is said to be contiguous if it is possible to travel between any two territorial units without leaving the zone.

(ii) Population equity: A partition is said to be in population equity if the number of desired units such as eligible voters and sales potential in each zone is roughly the same.

(iii) Compactness: A zone is said to be compact if it is geographically round shaped.

Yao et al. (2020) considers population equity and compactness principles in their study. To solve MJRPDC, they use a GA-based solution approach that integrates with a search algorithm for solving MJRP for a given zone to evaluate the performance of each districting setting. This study is an extension of MJRPDC investigated in Yao et al. (2020). The manager solves MJRPDC before negotiating a 3PL contract with the 3PL service provider. Then, the 3PL service provider plans a specified area (for example, an assigned dock) in its warehouse for the logistics operations, namely, sorting, packaging and loading items, of a particular zone. The availability of material handling resources such as labors or lifting equipment are essential constraints in the implementation. To fulfill customers' demand, the 3PL service provider reserves some specific amount of capacity of the material handling equipment for contract customers. Therefore, before negotiating with a contract customer, the manager needs to incorporate a set of material handling capacity constraints into the decision-making scenario of MJRPDC to coordinate the lot-sizing and replenishment for all the customers of each zone. The integration of MJRPDC with material handling capacity constraints is surely original in the literature. Most of the studies on JRP consider no restrictions on the replenishment operations of all items. However, managers make their logistics plans subject to many constraints due to limitations in transportation, budget, and physical space in practice. Goyal (1975) is the first study that introduced the JRP with budget constraints, and he proposed a heuristic based on the Lagrangian multipliers for solving the problem. Hoque (2006) included the considerations of storage capacity, transport capacity, and budget limitation in the JRP. Later, Moon and Cha (2006) investigated a JRP considering a budget constraint and solved it using a GA. Yao (2010) dealt with the JRP under the warehouse-space restrictions in a distribution center. Amaya, Carvajal, and Castano (2013) studied the JRP with budget limitations and proposed a heuristic based on linear programming. Ongkunaruk, Wahab, and Chen (2016) considered a JRP with defective items and the restrictions such as shipment, budget, and transportation capacity constraints. However, the researches in the literature are different from MRJP with material handling capacity constraints in this study (which will be presented in Section 2).

Similar to Yao et al. (2020), this study takes into account the principles of population equity and compactness because population size defines demand and compactness involves transportation distance and cost. The proposed GA-based framework to solve the districting problem is similar to that presented in Yao et al. (2020). Then, we propose a heuristic for solving MJRP considering Material Handling Capacity Constraints (MHCC) which plays a critical role in the evaluation of the performance of each districting setting. We use a small-size example showing a districting decision from the proposed solution approach. We also conduct a sensitivity analysis of the parameters corresponding to the demand equity and compactness constraints. This study contributes to the following two aspects. First, it presents an efficient search algorithm to solve MJRP under the GI policy. Second, by incorporating the search algorithm for MJRP, we propose a GA-based solution method to solve MJRPDC considering MHCC, and it may assist the decision-makers in obtaining a logistic plan before signing a contract with a 3PL service provider.

The rest of the paper is presented as follows. Section 2 introduces the decision-making scenario and the mathematical models for MJRPDC considering MHCC. Section 3 uses the local minimum obtained by the search algorithm of Yao et al. (2020) for unconstrained MJRP as the candidate solutions. Section 4 proposes a feasibility-testing procedure based on Yao (2001) heuristic to generate a feasible replenishment schedule, and fix an infeasible solution as obtaining no feasible schedule. Section 5 presents computational experiments showing the average runtime of our approach grows in a cubic order with respect to both the number of customers and the number of items. Section 6 concludes the paper with practical implications and remarks on future extensions.

2. MATHEMATICAL MODELS

We first introduce a decision-making scenario for the multi-customer JRP with districting considerations and material handling capacity constraints (abbreviated as MJRPDC-MHCC). Then, we review a mathematical model for MJRPDC. Finally, given a districting setting, we formulate a mathematical model for the multi-customer JRP (MJRP) considering MHCC.

2.1 Decision-making scenario

The manager needs to manage two issues when dealing with MJRPDC and MJRPDC-MHCC:

1. Districting: Divide all of the customers into p mutually exclusive zones.
2. Lot-sizing: Coordinate the replenishment of multiple items in each zone.

Regarding the issue of "districting", the manager should start with deciding the location (i.e., the coordinates) of "centroids" of p zones, and a centroid is not necessary to set at any customer. Then, each customer will be assigned to one and only one zone based on these centroids.

Figure 2.1 shows two zoning examples in a planning region with 3 zones and 10 customers; see also, Yao et al. (2020). For the left-side example, the manager first determines the coordinates of three centroids, e.g., (x_1, y_1) , (x_2, y_2) and (x_3, y_3) , and then, customers 1 to 4 are assigned to zone 1, customers 5 and 6 to zone 2, and the others to zone 3. The manager takes into accounts two concerns when making the districting decision:

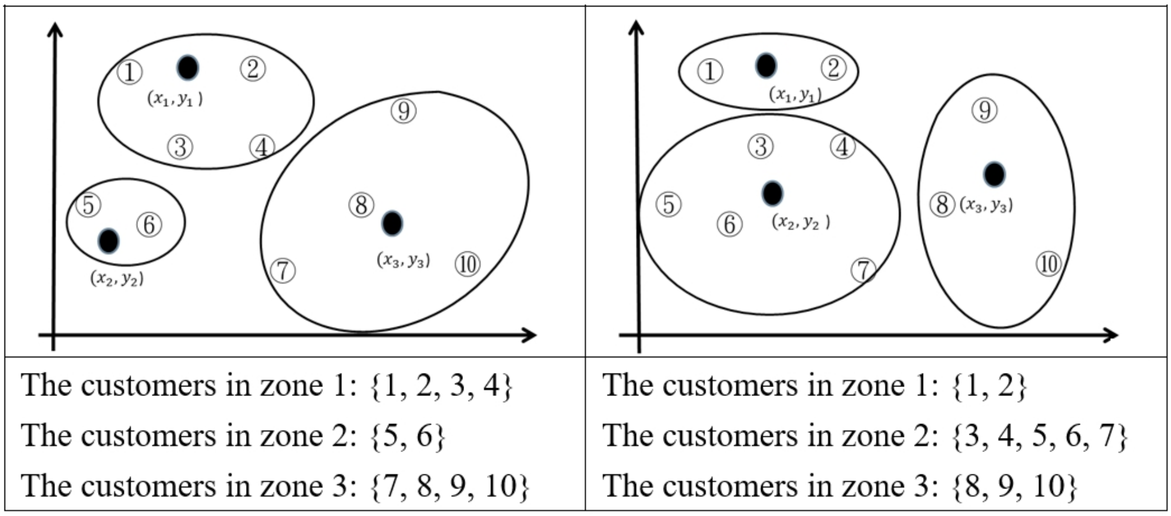

Figure 2.1: Two districting examples with 3 zones and 10 customers (Yao, et al., 2020)

1. Compactness: The distance between any customer and the corresponding centroid does not exceed M_d km.
2. Demand equity: The total demand of any zone is no more than α times the average demand of all zones.

Beside of districting, the manager needs to deal with the "lot-sizing" problem by coordinating the replenishment of the customers in the same zone to minimize the average total costs for each zone (called a multi-customer JRP), and to meet the material handling capacity constraints.

The objective function of the multi-customer JRP in each zone include the (major and minor) setup/ordering cost and the inventory holding cost, but similar to Chan et al. (2003), it does not take account of the transportation cost since we assume that a third-party-logistics (3PL) service provider handles the logistics of replenishment between the supplier and all the customers. Following the pricing mechanism of 3PL (see Haniefuddin et al., 2013), the transportation cost may be based on either sales revenue or the volume/weight of the commodities, and it turns to be a constant since the annual demand of each item is known and fixed. Therefore, we do not include the transportation cost in the objective function in this study.

For a given districting setting, the manager coordinates the lot-sizing and replenishment for all the customers in a zone z using a basic cycle time denoted by B_z . The replenishment cycle time of item i for customer j is a positive integer multiplier k_{ij} of B_z , i.e., $k_{ij}B_z$. Taking the right-hand example in Figure 3.1, we have the replenishment cycle time of item i for Customer 5 (in zone 2) is $k_{i5}B_2$. The set of material handling capacity constraints arise from the 3PL contract negotiation between the company and the 3PL service provider. Referring to practical operations in distribution centers, the 3PL service provider usually employs a specified operating area (sometimes, also a designated dock) in its warehouse for the logistics operations (including sorting, packaging and loading items for the customers) of a particular zone. The material handling equipment is critical in the implementation of these logistics operations. To guarantee feasibility of a replenishment schedule for a zone z, the logistics operations may use no more than the available capacity of the material handling equipment for zone z. When negotiating with a contract customer, 3PL service provider must reserve an amount of capacity of the material handling equipment. The manager of the company must take into account the set of material handling capacity constraints when coordinating the lot-sizing and replenishment for all the customers in a zone z.

2.2 The model for MRJPDC

This section reviews the model for MJRPDC as follows.

Parameters

q	Number of items
n	Number of customers
A_0	Major setup cost
a_{ij}	Minor setup cost of item i for customer j
h_{ij}	Holding cost of item i per unit, per unit time for customer j
d_{ij}	Demand rate of item i for customer j
p	Number of zones
M_d	Maximal allowed distance between any customer and its corresponding centroid
α	Balanced-factor among zones
$(\bar{x}_j, \bar{y}_j)$	Coordinate of customer j

Indices

i	item index, $i \in \{1, 2, \ldots, q\}$
j	customer index, $j \in \{1, 2, \ldots, n\}$
z	zone index, $z \in \{1, 2, \ldots, p\}$

Decision variables

B_z	Basic cycle time in zone z
k_{ij}	Multipliers of basic cycle time of item i of customer j
r_{jz}	$\begin{cases} = 1, \text{if customer } j \text{ is assigned to zone } z \\ = 0, \text{otherwise} \end{cases}$
(x_z, y_z)	Coordinates of the centroid of zone z
N_z	Subset of customers assigned to zone z, i.e., $N_z \equiv \{j : r_{jz} = 1, j = 1, \ldots, n\}, z = 1, \ldots, p$

Assumptions

1. The replenishment for each item is instantaneous, and the replenishment cycle time of each item is constant.
2. No shortages are permitted.
3. A third-party logistics (3PL) service provider is responsible for the replenishment. Therefore, we take into accounts neither the vehicle routing, nor the transportation costs.

Model Formulation

$$TC([B_z]_{1\times p},[k_{ij}]_{q\times n},[r_{jz}]_{n\times p},[(X^z,Y^z)]_{z=1}^{p}) = min\sum_{z=1}^{p}\frac{A_0}{B_z}+\sum_{z=1}^{p}\sum_{j=1}^{n}\sum_{i=1}^{q}r_{jz}(\frac{a_{ij}}{k_{ij}B_z}+\frac{h_{ij}d_{ij}k_{ij}}{2}B_z) \tag{2.1}$$

Subject to

$$\sum_{z=1}^{p}r_{jz}=1, j=1,\dots,n \tag{2.2}$$

$$\sum_{i=1}^{q}\sum_{j=1}^{n}d_{ij}r_{jz}\leq\frac{\alpha}{p}\sum_{i}\sum_{j}d_{ij}, z=1,\dots,p \tag{2.3}$$

$$r_{jz}\sqrt{(\bar{x}_j-x_z)^2+(\bar{y}_j-y_z)^2}\leq M_d, j=1,\dots,n; z=1,\dots,p \tag{2.4}$$

$$r_{jz}\in\{0,1\}, j=1,\dots,n; z=1,\dots,p \tag{2.5}$$

$$(x_z,y_z)\in\mathbb{R}_{+}^{2}, z=1,\dots,p \tag{2.6}$$

$$k_{ij}\in\mathbb{Z}^{+}, i=1,\dots,q; j=1,\dots,n \tag{2.7}$$

$$B_z>0, z=1,\dots,p \tag{2.8}$$

The objective function in (2.1) is to minimize the average total costs that include the major setup cost, the minor setup cost and inventory holding cost per unit of time. Equations (2.2) ensure that each customer can be assigned to a single zone. The inequalities in (2.3) mandate that the total demand of all the customers in each zone cannot exceed α times of the average total demand of all zones. The constraints in (2.4) make sure that a customer j may be assigned to zone z, only if the distance between customer j and the centroid of zone z is no more than M_d km. Constraints (2.5) - (2.8) define the domain of the decision variables.

2.3 The model for MRJP considering MHCC

Given any zone z in a districting setting, we formulate a mathematical model for multi-customer JRP (MJRP) considering MHCC.

Before presenting the model, we re-define some notation to simplify the formulation. We define an index $\hat{i}$ for an artificial item corresponding to the demand of an item i for a customer j. Table 2.1 shows the redefinition of $\hat{i}$ using a small example in which three customers (Customer 1, 7 and 9) in zone 1 demand for two items (Item 1 and 2), or $N_1\in\{1,7,9\}$ with $n=2$.

Suppose that each customer demands for all items. The index $\hat{i}$ runs from 1 to $\sum_{z=1}^{p}n\cdot|N_z|$. For simplified notation, we define $\hat{n}\equiv\sum_{z=1}^{p}n\cdot|N_z|$. We may redefine notations of the minor setup cost ($a_{\hat{i}}$), the demand rate ($d_{\hat{i}}$), the holding cost ($h_{\hat{i}}$), respectively. For instances, it holds that $a_{11}=a_{\hat{i}}$, $d_{29}=h_{\hat{6}}$ and $h_{17}=h_{\hat{2}}$ following Table 2.1.

Table 2.1: An example of redefinition of index $\hat{i}$

$\hat{i}$	Original definition
1	Item 1 ($i=1$) demanded by Customer 1 ($j=1$)
2	Item 1 ($i=1$) demanded by Customer 7 ($j=7$)
3	Item 1 ($i=1$) demanded by Customer 9 ($j=9$)
4	Item 2 ($i=2$) demanded by Customer 1 ($j=1$)
5	Item 2 ($i=2$) demanded by Customer 7 ($j=7$)
6	Item 2 ($i=2$) demanded by Customer 9 ($j=9$)

We define some additional notations before presenting the mathematical model.

Parameters

$\bar{k}i$: Largest integer multiplier of item $\hat{i}$

$r_{\hat{i}}$: Processing time (of material handling) of item $\hat{i}$ per unit

H_z : Available capacity of the material handling equipment for zone z

Index

t : period in a unit of basic cycle time, $t = 1, 2, 3, \ldots, lcm\{k_{\hat{i}}\}$

Decision variables

$$y_{\hat{i},k,\varphi(k,t)} = \begin{cases} 1, \text{if item } \hat{i} \text{ is replenished in the } \varphi(k,t)\text{-th period among the first } k \text{ periods} \\ 0, \text{otherwise.} \end{cases}$$

Assumptions

The sum of total processing time (of material handling) cannot exceed the available capacity of the material handling equipment for zone z in each period.

Model Formulation

Given a particular zone z in a districting setting, the mathematical model of MJRP considering MHCC is presented as follows.

Minimize

$$\Gamma([y_{\hat{i},k,\varphi(k,t)}]_{\substack{\hat{i}=1,2\ldots,\sum_{z=1}^{p} n\cdot|N_z| \\ k=1,2\ldots,\bar{k}_{\hat{i}} \\ t=1,2,\ldots,lcm\{k_{\hat{i}}\}}}, B_z) = \frac{A_0}{B_z} + \sum_{\hat{i}=1}^{\sum_{z=1}^{p} n\cdot|N_z|} \sum_{k=1}^{\bar{k}_{\hat{i}}} \sum_{t=1}^{k} \left(\frac{a_{\hat{i}} y_{\hat{i},k,\varphi(k,t)}}{kB_z} + \frac{kB_z d_{\hat{i}} h_{\hat{i}} y_{\hat{i},k,\varphi(k,t)}}{2}\right) \tag{2.9}$$

Subject to

$$\sum_{\hat{i}=1}^{\sum_{z=1}^{p} n\cdot|N_z|} \sum_{k=1}^{\bar{k}_{\hat{i}}} kBd_{\hat{i}} r_{\hat{i}} y_{\hat{i},k,\varphi(k,t)} \leq H_z, t = 1, 2, \ldots, lcm\{k_{\hat{i}}\} \tag{2.10}$$

$$\sum_{k=1}^{\bar{k}_{\hat{i}}} \sum_{t=1}^{k} y_{\hat{i},k,\varphi(k,t)} = 1, \hat{i} = 1, 2, \ldots, \sum_{z=1}^{p} n \cdot |N_z| \tag{2.11}$$

$$k_{\hat{i}} = \sum_{k=1}^{\bar{k}_{\hat{i}}} k \cdot \sum_{t=1}^{k} y_{\hat{i},k,\varphi(k,t)}, \hat{i} = 1, 2, \ldots, \sum_{z=1}^{p} n \cdot |N_z| \tag{2.12}$$

where

$$y_{\hat{i},k,\varphi(k,t)} = \begin{cases} 1, \text{if item } \hat{i} \text{ is replenished in the } \varphi(k,t)\text{-th period among the first k periods} \\ 0, \text{otherwise.} \end{cases} \tag{2.13}$$

and

$$\varphi(k,t) = \begin{cases} t \bmod k & , \text{if } t \neq \xi \cdot k, \xi \in \mathbb{N}^+ \\ k & , \text{if } t = \xi \cdot k, \xi \in \mathbb{N}^+ \end{cases} , t = 1, 2, \ldots, lcm\{k_{\hat{i}}\} \tag{2.14}$$

$$y_{\hat{i},k,\varphi(k,t)} \in \{0,1\}, \hat{i} = 1, 2, \ldots, \sum_{z=1}^{p} n \cdot |N_z|; k = 1, 2 \ldots, \bar{k}_{\hat{i}}; t = 1, 2, \ldots, lcm\{k_{\hat{i}}\} \tag{2.15}$$

$$B_z > 0 \tag{2.16}$$

The objective function in eq. (2.9) is to minimize the average total costs, including major setup cost, minor setup cost, and holding costs. The constraints in (2.10) mandate the sum of total processing time (of material handling) cannot exceed the available capacity of the material handling equipment for zone z in each of the $lcm\{k_{\hat{i}}\}$ periods.

The equations in (2.11) and (2.12) determine the starting period and the replenishment cycle time of an item $\hat{i}$, respectively, with $y_{\hat{i},k,\varphi(k,t)}$ in (2.13) and the function $\varphi(k,t)$ in (2.14) . Equations (2.15) and (2.16) express the domain of the decision variables B_z and $y_{\hat{i},k,\varphi(k,t)}$. Table 2.2 presents an example of using $y_{\hat{i},k,\varphi(k,t)}$ for deciding the replenishment cycle time and cyclic replenishment scheduling of an item $\hat{i}$ with $\bar{k}_{\hat{i}} = 3$.

Table 2.2: Using $y_{\hat{i},k,\varphi(k,t)}$ for deciding the replenishment cycle time and scheduling

Multiplier	Periods in the unit of a basic cycle time		
	1^{st} period	2^{nd} period	3^{rd} period
$k=1$	$y_{\hat{i},1,1}$		
$k=2$	$y_{\hat{i},2,1}$	$y_{\hat{i},2,2}$	
$k=3(=\bar{k}_{\hat{i}})$	$y_{\hat{i},3,1}$	$y_{\hat{i},3,2}$	$y_{\hat{i},3,3}$

When $k=1$, there is only one choice, that is to start the replenishment in the first period (and repeat replenishing in each period thereafter). For the case of $k=2$, one may start the replenishment in either the first or the second period. In general, there are a total of $2^{\bar{k}_{\hat{i}}} - 1 = 2^3 - 1 = 7$ possible choices of (replenishment cycle time–starting period) combination. Eq. (2.11) shows that one may choose one and only one out of these choices. Suppose that $y_{\hat{i},2,2} = 1$ in this example. Then, eq. (2.12) shall obtain a multiplier $k_{\hat{i}} = 2$ (or equivalently, a replenishment cycle time of $k_{\hat{i}}B = 2B$) for item $\hat{i}$.

Consider an example with only two items with $k_{\hat{1}} = 2$ and $k_{\hat{2}} = 3$. The whole replenishment schedule repeats every $lcm(k_{\hat{1}}, k_{\hat{2}}) = 6$ periods. (By the same token, the whole replenishment schedule repeats every $lcm\{k_{\hat{i}}\}$ periods in the general case.) If the starting period of item $\hat{2}$ is period 2 (among the first $k_{\hat{2}} = 3$ periods), i.e., $y_{\hat{2},3,2} = 1$, we have the value of $\varphi(k,t) = \varphi(3,5) = 5$ mod 3 = 2 at the 5^{th} period, which means $y_{\hat{2},3,\varphi(3,5)} = y_{\hat{2},3,2} = 1$ following (2.14). Therefore, the function $\varphi(k,t)$ keeps an item being replenished after a fixed cycle time.

The model in (2.9) - (2.16) precisely expresses the problem of MJRP considering MHCC, especially, in the replenishment scheduling of items. However, we are not able to directly use the model for decision-making because of the following reasons.

1. The values of $k_{\hat{i}}$ are dependent variables of the decision variables $y_{\hat{i},k,\varphi(k,t)}$.
2. There are a total of $lcm\{k_{\hat{i}}\}$ constraints for each of (2.10), (2.14) and (2.15), respectively.
3. The total number of constraints in the model stays unknown since it depends on the decision variables $y_{\hat{i},k,\varphi(k,t)}$.

We will utilize part of the model in the determination of a feasible replenishment schedule in the proposed solution approach discussed in Section 4 later.

3. AN INTEGRATED GA-BASED SOLUTION APPROACH

We propose a solution approach for solving MJRPDC-MHCC in this section. First, we introduce the coding-encoding mechanisms of the chromosome and the initialization procedure. Then, we discuss the fitness evaluation of chromosomes and the genetic operators (namely, selection, crossover, and mutation, etc.) and the termination conditions in GA. Figure 3.1 illustrates an integrated framework of the proposed GA-based solution approach.

3.1 Coding-encoding mechanisms and initialization

We employ the same coding-encoding mechanisms of the chromosome as Yao et al. (2020) in this study since it enjoys greater flexibility to meet the compactness and distance allowance restrictions when solving the districting problem.

Given a maximum allowed unit distance, $\bar{\mu}$ (e.g., 0.5 km), we cover the service region by a reasonable number of discrete grid points with a unit of grid width/length of no more than $\bar{\mu}$ km. We denote the minimum number of grids needed to represent the coordinates of the service region on the l-axis as GP_l , where $l \in \{x, y\}$:

$$GP_l = \lceil \frac{l_{max} - l_{min}}{\bar{\mu}} \rceil \tag{3.1}$$

where l_{min} and l_{max} are the smallest and largest coordinate values of the customers' locations on the l axis, where $l \in \{x, y\}$. Let β_l be the number of bits required for the representation on the l axis:

$$\beta_l = \lceil \log_2(GP_l + 1) \rceil, l \in \{x, y\} \tag{3.2}$$

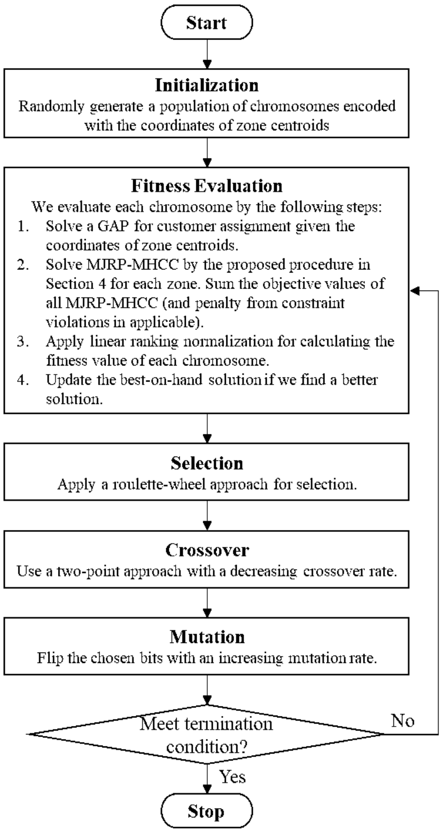

Figure 3.1: Integrated framework of the proposed GA-based approach

We assure that the grid width/length on the l axis is no greater than $\bar{u}$ following (3.1) and (3.2) where

$$u_l = \frac{l_{max} - l_{min}}{2^{\beta_l} - 1}, l \in \{x, y\} \tag{3.3}$$

An encoding mechanism converts the coordinate value on the l axis of a zone centroid into binary strings $(b_{\beta_l} \quad b_{\beta_l - 1} \quad \ldots \quad b_1 \quad b_0)$ for the chromosome representation in our GA. We may decode the coordinate value on the l axis of a zone, denoted as $\tilde{l}$, by

$$\tilde{l} = l_{min} + (\sum_{i=0}^{\beta_l} b_i \cdot 2^i)\mu_l, l \in \{x, y\} \tag{3.4}$$

The initialization process of the GA randomly generates binary strings, representing the coordinates of the zone centroids, for the chromosomes in the initial population. Our encoding mechanism ensures the locations of all the zone centroids keep inside of the service region.

3.2 Evaluation of chromosomes

Referring to Yao et al. (2020), we employ an evaluation procedure to evaluate the fitness for each chromosome using the following four steps:

1. Solve a generalized assignment problem (GAP) in (3.5)-(3.8) to assign customers to all zones based on the decoded locations of the zone centroid.

$$\text{Minimize} \quad \sum_{j=1}^{n}\sum_{z=1}^{p} c_{jz} r_{jz} \tag{3.5}$$

Subject to

$$\sum_{i=1}^{q}\sum_{j=1}^{n} d_{ij} r_{jz} \leq \frac{\alpha}{p}\sum_{i}\sum_{j} d_{ij}, z = 1, \ldots, p \tag{3.6}$$

$$\sum_{z=1}^{p} r_{jz} = 1, j = 1, \ldots, n \tag{3.7}$$

$$r_{jz} \in \{0, 1\}, j = 1, \ldots, n; z = 1, \ldots, p \tag{3.8}$$

The objective function in (3.5) is to minimize the total distance of all customers to their assigned zone centroid, where c_{jz} is the distance between customer j and the centroid of zone z. The constraints in eq. (3.6) are identical to (2.3), which mandate the demand equity principle. The constraints in Eq. (4.7), the same as eq. (2.2), ensure that each customer is assigned to one and only one zone. We do not include the compactness constraint in Inequality (2.4) in the GAP to allow our GA searching for an optimal solution via illegal chromosomes as intermediate structures. Our GA applies an extremely large penalty function whenever encounter any violation in the compactness constraint. Similar to Yao et al. (2020), we use the heuristic of Jeet and Kutanoglu (2007) for solving efficiently the GAP.

2. Calculate the average total cost given the set of customers assigned to each zone by solving MJRP with MHCC for each zone with the given set of customers using the proposed procedure in Section 4. Then, sum up the average total costs, as well as the penalties for the constraint violation of each zone if applicable, for all the zones.

3. Calculate the fitness value for the chromosome by applying $linear\ ranking\ normalization$ (see Pohlheim (1999)), where all the chromosomes in a population are ranked and stored in a temporary list, $temp$.
We denote the ranking of the chromosome i in a population as R_i and calculate its normalized fitness value, $eval$, by

$$eval_{R_i} = 2 - SP + \frac{2(SP-1)(R_i-1)}{PS-1} \tag{3.9}$$

where PS is the size of a population and takes a value of the selection pressure (SP) in the range of [1.0, 2.0]. The best-fit chromosome takes the first portion of the ranking list and is ranked the highest ($R_i = PS$), whereas the least-fit chromosome reserves the last portion of the list and is ranked the lowest ($R_i = 1$).

4. Update the best-on-hand solution if a better solution appears.

3.3 GA operators and termination conditions

Before forming the population in the next generation, GA uses selection to pick the chromosomes from the current population to undergo genetic operators. $Tournament\ selection$ and $roulette\ wheel\ selection$ are most popular among the selection mechanisms used in GA (see Michalewicz (1996)). Tournament selection matches a few randomly chosen chromosomes for "tournaments", and the winner of each tournament (the one with the best fitness) goes for crossover. The proposed GA adopts the roulette wheel selection in which a probability of selection is associated with the fitness level of each chromosome.

Crossover exchanges genetic information of two parents to generate new offspring in the next generation and explores the solution space to search for new solutions. Single-point crossover, two-point (or k-point crossover), and Uniform crossover are most popular when using binary-encoding as the data structure in GA (see Michalewicz (1996)). Since our numerical experiments show no significant difference among these three operators, we choose a $two-point$ $crossover$ in the proposed GA. The operator first randomly picks two cut points are from both parent chromosomes and swaps the bits in between the two cut points between the parent organisms. The proposed GA uses a dynamic crossover probability since a higher crossover probability at the beginning of the evolutionary process enhances diversity among the chromosomes improving the exploration of the solution space. Alternatively, a lower crossover probability assists in keeping information across generations before the evolution terminates. Therefore, the crossover probability decreases linearly from one generation to the next in our GA. One may refer to Srinivas (1994) and Zhang, Chung, S., and Lo (2007), etc., for the promotion of using dynamic (or adaptive) probabilities.

Mutation conducts a minor random deviation to increase diversity in the evolutionary process since it randomly changes some genes in a chromosome before being passed on to the next generation. Similar to the crossover probability, the proposed GA uses a dynamic setting of mutation probability that increases linearly after a fixed number of generations. This improves diversity and exploit solutions in its neighborhood, particularly when chromosomes turn to be too similar to each other close to the end of evolutionary process.

The proposed GA terminates when the evolutionary process shows no further improvement. For example, our GA procedure stops when the best-on-hand solution was not updated over the last 100 generations.

4. HEURISTIC FOR SOLVING MJRP CONSIDERING MHCC

We propose a heuristic for solving MJRP Considering MHCC in this section. The section includes three parts. The first part reviews the search algorithm proposed by Yao et al. (2020) for solving the (unconstrained) MJRP as a framework of our heuristic. Then, we present the procedure for generating a replenishment schedule considering MHCC. The third part presents our heuristic that picks the local minimum with not only a feasible replenishment schedule, but also the minimum objective function value as the heuristic solution for MJRP considering MHCC.

4.1 Review of the search algorithm for solving MJRP

We review the search algorithm for solving MJRP in a given zone z, proposed by Yao et al. (2020) since the search algorithm will serve as a framework of our heuristic that solves MJRP considering MHCC. The overview of the search algorithm is presented as follows. First, we define a search range within a lower and an upper bound on the B_z-axis, denoted by $\underline{B}_z$ and $\bar{B}_z$, respectively. Note that $\underline{B}_z$ and $\bar{B}_z$ are derived based on the rationale that the solutions beyond $\underline{B}_z$ and $\bar{B}_z$ are no better than the optimal solution located in $[\underline{B}_z, \bar{B}_z]$. The proposed algorithm searches along the B_z-axis from the upper bound ($\bar{B}_z$) to the lower bound ($\underline{B}_z$).

Then, we obtain the junction points, i.e., $\delta_{ij}(m)$, in $[\underline{B}_z, \bar{B}_z]$, sort them to get a non-decreasing sequence $\cdots \leq w^2 \leq w^1 \leq w^0 = \bar{B}_z$, and take the sorted sequence as the road map to proceed with the search. We divide the search range into sub-intervals where each sub-interval $[w^l, w^{l-1}]$ is defined by two neighboring points in the sorted sequence. Following the junction point analysis, we are able to obtain the vector of optimal multipliers, $\bar{\mathbf{K}}$, and the local minimum $\breve{B}_z^*(\bar{\mathbf{K}})$. We take the local minimum as a candidate for the optimal solution in each sub-interval, and update the best-on-hand solution if applicable. By examining all the local minima that exist in $[\underline{B}_z, \bar{B}_z]$, we secure an optimal solution for MJRP in a given zone z.

One may refer to Yao et al. (2020) for the details of the search algorithm.

4.2 Generate a replenishment schedule considering MHCC

We present a procedure that generates a feasible replenishment schedule considering MHCC in this section. Our proposed procedure is based on Yao (2001) heuristic for solving Peak Load Minimization Problem in Cyclic Production. Note that it shares a common characteristic of cyclic replenishment with MJRP considering MHCC. But, we need to take care of the constraints in (2.10) that mandate the sum of total processing time (of material handling) cannot exceed the available capacity of the material handling equipment for zone z in each of the $lcm\{k_{\hat{i}}\}$ periods when solving MJRP considering MHCC.

Our heuristic for generating a replenishment schedule considering MHCC follows the Feasibility Testing Procedure for the ELSP in Yao (2001) in which it takes into account the capacity constraints of a production facility in the Economic Lot Scheduling Problem (ELSP). We use Proc Gen Schedule to test the feasibility of a given combination $(K(B), B)$ for MJRP considering MHCC. We denote $L^*(K(B), B)$ as the peak load, which corresponds to the largest value of the sum of total processing time (of material handling) among all the $lcm\{k_{\hat{i}}\}$ periods. Apparently, if it holds that $L^*(K(B), B) \leq H_z$ for all the $lcm\{k_{\hat{i}}\}$ basic periods, the replenishment schedule is feasible. We define an indicator ϕ in $Proc\ FT$ to indicate if a feasible production schedule is obtained when $\phi = 1$; otherwise, $\phi = 0$. Let

$L(W^F)$ and W^F be the peak load and the replenishment schedule obtained by Proc Gen Schedule. We present the flowchart of our heuristic, Proc Gen Schedule, in Figure 4.1.

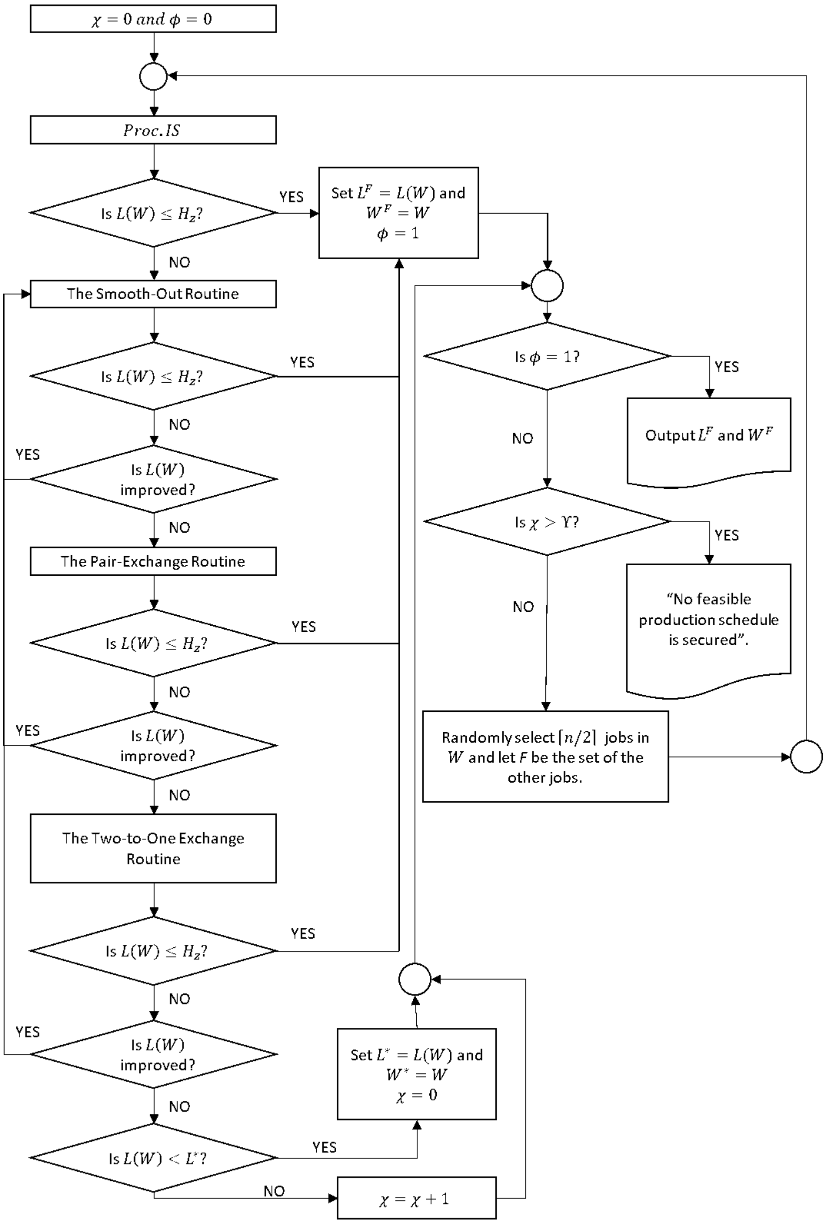

Figure 4.1: Flowchart of Proc Gen Schedule

Further discussion about the solution framework is as follows. In the beginning of Proc Gen Schedule, an $Initial\ Schedule\ Procedure(Proc\ IS)$ is used to obtain an initial schedule W by emulating the LPT rule that iteratively assigns replenishment lot size to the customer with the longest σ_j (i.e., the longest processing time of material handling) among the unassigned customers until all customers are assigned.

After an initial schedule W is obtained, one starts with the next run of local search (or re-optimization) by employing three subroutines, namely, the $Smooth-Out\ Routine$, the Pair-Exchange Routine, and the $Two-to-One\ Exchange\ Routine$ to reduce the peak load in W as much as possible. The terms "Smooth-Out" and "Exchange" indicate the essence of local search, namely, moving-out or exchanging the processing time of replenishment lots produced in the peak-load period to reduce the peak load in the schedule. To prevent from getting stuck in a local minima, we randomly select half of the customers (i.e., $\lfloor \frac{n}{2} \rfloor$), and re-assign their replenishment schedule. One may refer to Yao (2001) for the details of the three subroutines.

The termination of Proc Gen Schedule is based on the value of χ as the number of $consecutive$ times that the procedure was not able to improve $L^*(W^*)$ where $L^*(W^*)$ is the minimal peak load obtained by the procedure using W^* and W^* is the minimal peak load schedule obtained in the heuristic. A threshold value γ is set for the termination, i.e., one stops Proc Gen Schedule when $\chi > \gamma$. (We set $\gamma = 3$ in our numerical experiments.)

Following the numerical experiments in Yao (2001), Proc Gen Schedule is efficient since its run time grows in a cubic order of the problem size, i.e., it is approximately $O(n^3)$ algorithm.

4.3 Proposed heuristic for solving MJRP considering MHCC

The proposed heuristic uses the search algorithm proposed by Yao et al. (2020) for solving MJRP as a framework of our heuristic; please refer to Section 4.1 and Algorithm 2. Taking each local minimum, say the l-th local minimum, as a candidate solution $(\breve{B}_z^*(\bar{\mathbf{K}}), \bar{\mathbf{K}})$, we aim to generate a feasible replenishment schedule for this local minimum by Proc Gen Schedule discussed in Section 4.2. If we are able to obtain a feasible replenishment schedule for $(\breve{B}_z^*(\bar{\mathbf{K}}), \bar{\mathbf{K}})$, we calculate the objective function value by $TC_z^l(\breve{B}_z^*(\bar{\mathbf{K}}), \bar{\mathbf{K}})$. If Proc Gen Schedule generates no feasible replenishment schedule, we will find a larger value of B, denoted by $\tilde{B}$, so that $(\tilde{B}, \bar{\mathbf{K}})$ may obtain a feasible replenishment schedule, set $\breve{B}_z^*(\bar{\mathbf{K}}) = \tilde{B}$, and calculate $TC_z^l(\breve{B}_z^*(\bar{\mathbf{K}}), \bar{\mathbf{K}})$. Then, we try to update the best-on-hand solution if $TC_z^l(\breve{B}_z^*(\bar{\mathbf{K}}), \bar{\mathbf{K}})$ secures a better objective function value. After examining all the local minimum, we report the best-on-hand solution as our heuristic solution.

Algorithm 2.

Step1: Initialization

i. Find an initial upper bound: Calculate T_{cc} and $max_{ij}\delta_{ij}(1)$ by (4.1) and (4.2).
Set $\bar{B}_z = max\{T_{cc}, max_{ij}\delta_{ij}(1)\}$.

$$T_{cc} = \sqrt{\frac{A_0 + 2\sum_{i=1}^{q}\sum_{j=1}^{n} a_{ij}}{\sum_{i=1}^{q}\sum_{j=1}^{n} h_{ij}d_{ij}}} \tag{4.1}$$

$$\delta_{ij}(m) = \sqrt{\frac{1}{m(m+1)}}\sqrt{\frac{2a_{ij}}{h_{ij}d_{ij}}}, \forall m \in \mathbb{N} \tag{4.2}$$

ii. The initial solution of the optimal multipliers $\bar{\mathbf{K}}(\bar{B}_z)$ can be derived from (4.3). Then, we calculate the initial solution $TC_z(\bar{B}_z, \bar{\mathbf{K}}(\bar{B}_z))$ by (4.4), and set $\check{\Phi}_z^* \leftarrow TC_z(\bar{B}_z, \bar{\mathbf{K}}(\bar{B}_z))$.

$$\bar{k}_{ij}(B_z) = \lceil -\frac{1}{2} + \frac{1}{2}\sqrt{1 + \frac{8a_{ij}}{h_{ij}d_{ij}B_z}} \rceil \tag{4.3}$$

where $\lceil x \rceil$ denotes that the ceiling function maps x to the least integer greater than or equal to x.

$$TC_r(B_z, [k_{ij}]_{q\times n}) = \frac{A_0}{B_z} + \sum_{j\in N_z}\sum_{i=1}^{q}(\frac{a_{ij}}{k_{ij}B_z} + \frac{h_{ij}d_{ij}k_{ij}}{2}B_z) \tag{4.4}$$

iii. Find an initial lower bound: Let $\underline{B}_z = 0$. Set $\Phi_z^U = \check{\Phi}_z^*$, and calculate β_1 and β_2 by (4.5) and (4.6), respectively. Update $\underline{B}_z$ by (4.7).

$$\beta_1 = \frac{2A_0}{\Phi_z^U} \tag{4.5}$$

$$\beta_2 = \frac{A_0}{\check{\Phi}_z^* - \sum_{j\in N_z}\sum_{i=1}^{q}\sqrt{2a_{ij}h_{ij}d_{ij}}} \tag{4.6}$$

$$\underline{B}_z = max\{\beta_1, \beta_2\} \tag{4.7}$$

Step2: Further improve the bounds by the following iterative procedures:

i. Improve the upper bound:
  (a) Let $B_z^b \leftarrow \bar{B}_z$
  (b) Calculate $\bar{\mathbf{K}}(\bar{B}_z)$ by (4.3) and $\breve{B}_z(\bar{\mathbf{K}}(\bar{B}_z))$ by (4.8), respectively.

$$\breve{B}_z(\bar{\mathbf{K}}(\bar{B}_z)) = \sqrt{\frac{2 + (\sum_{j\in N_z}\sum_{i=1)}^{q}\frac{a_{ij}}{k_{ij}(\bar{B}_z)}}{\sum_{j\in N_z}\sum_{i=1}^{q} h_{ij}d_{ij}\bar{k_{ij}}(B_z)}} \tag{4.8}$$

(c) Set $\bar{B}_z \leftarrow \check{B}_z(\bar{\mathbf{K}}(\bar{B}_z))$.
(d) If $B_z^b = \bar{B}_z$, go to ii.; otherwise, go to Stepi.(a).

ii. Improve the lower bound:

(a) Let $B_z^a \leftarrow \underline{B}_z$
(b) Calculate $\bar{\mathbf{K}}(\underline{B}_z))$ by (4.3) and $\check{B}_z(\bar{\mathbf{K}}(\underline{B}_z))$ by (4.8), respectively.
(c) Set $\underline{B}_z \leftarrow \check{B}_z(\bar{\mathbf{K}}(\underline{B}_z))$.
(d) If $B_z^a = \underline{B}_z$, go to Step3; otherwise, go to Stepii.(a).

Step3: Set $l = 1$ and $\check{\Phi}_z^* \leftarrow \infty$. Sort the junction points in a non-increasing order:

$$\cdots \leq w^2 \leq w^1 \leq w^0 = \bar{B}_z \tag{1}$$

Step4: Solve the sub-problem (P^l) by the following steps:

i. Given $\bar{\mathbf{K}}$, solve the sub-problem (P^l) in (4.9).

$$(P^l) \begin{array}{ll} \min\limits_{B_z} & TC_z^l(B_z, [\bar{k}_{ij}]_{q\times n}) = \frac{A_0}{B_z} + \sum\limits_{j\in N_z}\sum\limits_{i=1}^{q}(\frac{a_{ij}}{\bar{k}_{ij}B_z} + \frac{h_{ij}d_{ij}\bar{k}_{ij}}{2}B_z) \\ s.t. & B_z \in [w^l, w^{l-1}] \end{array} \tag{4.9}$$

ii. Obtain $\check{B}_z^*(\bar{\mathbf{K}})$ by (4.10), and calculate $TC_z^l(\check{B}_z^*(\bar{\mathbf{K}}), \bar{\mathbf{K}})$.

$$\check{B}_z^*(\bar{\mathbf{K}}) = \begin{cases} \check{B}_z(\bar{\mathbf{K}}) & \text{, if } \check{B}_z(\bar{\mathbf{K}}) \in [w^l, w^{l-1}] \\ w^l & \text{, if } \check{B}_z(\bar{\mathbf{K}}) < w^l \\ w^{l-1} & \text{, if } \check{B}_z(\bar{\mathbf{K}}) > w^{l-1} \end{cases} \tag{4.10}$$

iii. Try to generate a feasible replenishment schedule for the candidate $(\check{B}_z^*(\bar{\mathbf{K}}), \bar{\mathbf{K}})$ by Proc Gen Schedule.

(a) If Proc Gen Schedule finds a feasible replenishment schedule for $(\check{B}_z^*(\bar{\mathbf{K}}), \bar{\mathbf{K}})$, we calculate $TC_z^l(\check{B}_z^*(\bar{\mathbf{K}}), \bar{\mathbf{K}})$.
(b) If Proc Gen Schedule obtains no feasible replenishment schedule for $(\check{B}_z^*(\bar{\mathbf{K}}), \bar{\mathbf{K}})$, we find $\widetilde{B}$ by (4.11) for the period with the maximal load, set $\check{B}_z^*(\bar{\mathbf{K}}) = \widetilde{B}$, and calculate $TC_z^l(\check{B}_z^*(\bar{\mathbf{K}}), \bar{\mathbf{K}})$.

$$\widetilde{B} = H_z \Big/ \sum_{\hat{i}=1}^{\sum_{z=1}^{p} n\cdot|N_z|} \sum_{k=1}^{\bar{k}_{\hat{i}}} kBd_{\hat{i}}r_{\hat{i}}y_{\hat{i},k,\varphi(k,t)} \tag{4.11}$$

Step5: Update the best-on-hand solution. If $TC_z^l(\check{B}_z^*(\bar{\mathbf{K}}), \bar{\mathbf{K}}) < \check{\Phi}_z^*$, then $\check{\Phi}_z^* \leftarrow TC_z^l(\check{B}_z^*(\bar{\mathbf{K}}), \bar{\mathbf{K}})$, $\check{B}_z^* \leftarrow \check{B}_z^*(\bar{\mathbf{K}})$, and $\check{\mathbf{K}}^* \leftarrow \bar{\mathbf{K}}$. Also, set $\Phi_z^U = \check{\Phi}_z^*$, and calculate β_1 and β_2 by (4.5) and (4.6), respectively. Update $\underline{B}_z$ by $\underline{B}_z \leftarrow \max\{\beta_1, \beta_2\}$ if $\underline{B}_z < \max\{\beta_1, \beta_2\}$.

Step6: Stop criterion. If $w^{l-1} < \underline{B}_z$, go to Step 7; otherwise, determine $(\widetilde{i}, \widetilde{j})$ by (4.12), then update $\bar{\mathbf{K}} = \bar{\mathbf{K}} \backslash \{\bar{k}_{\widetilde{ij}}\} \cup \{\bar{k}_{\widetilde{ij}} + 1\}$, set $l \leftarrow l + 1$, and go to Step 4.

$$(\widetilde{i}, \widetilde{j}) = \arg\max_{i,j}\{\delta_{ij}(\bar{k}_{ij}) < w^l | i = 1, \ldots, q, j \in N_z\}. \tag{4.12}$$

Step7: Output the optimal solution $\mathbf{K}^* \leftarrow \check{\mathbf{K}}^*$, $B_z^* \leftarrow \check{B}_z^*$, and $\Phi_z^* \leftarrow \check{\Phi}_z^*$

5. NUMERICAL EXPERIMENTS

We conduct some numerical experiments to verify our theoretical results and demonstrate our proposed algorithm in this section. The first part presents the settings of our numerical experiments. Then, the second part uses a small-size example showing a districting decision of 10 customers from our proposed GA. The third part investigates the runtime growth of the proposed solution approach with respect to the problem size.

5.1 Settings of experiments

We use a personal computer with a microprocessor of Intel® Core™ i5 CPU 750 @ 2.67GHz and 24GB memory for our numerical experiments.
The instances of our numerical experiments are randomly generated using the following guidelines.

1. The coordinates of the customers: We employ the set R (uniformly distributed in the planning region) in Solomon (1987) and randomly choose among the coordinates among the 100 nodes of the benchmark problems.

2. The parameters related to inventory control and material handling: We refer to Van Eijs (1993) to set the range of demand rate d_{ij} , holding cost h_{ij} , and minor setup cost a_{ij} . Also, we include the range of the processing time and the capacity limit of material handling equipment. Table 5.1 shows the ranges of the parameters.

Table 5.1: Range of the parameters

Parameters	Minimum	Maximum
Demand rate d_{ij}	100	1000000
Minor setup cost a_{ij}	1	51
Holding cost h_{ij}	0.2	2
Processing time of MH r_i	0.00001	0.00009
Available capacity of MH equipment for zone z H_z	8 hours/day	
Major setup cost A	200	

Table 5.2 presents the parameters of our GA. We find the best settings of parameters by evaluating the trade-off between the improvement in solution quality and runtime. To allow our GA for more ability of exploration, we set a lower penalty of 100,000 for the first 50 generations when there exists any violation in compact principle constraints. To force GA to search in the feasible solution space, we increase the value of penalty to 10,000,000 (which is 100 times of the anterior setting) after the 50^{th} generations.

Table 5.2: Parameters of GA

Parameters	Value of setting
Population size(pop_size)	40
Crossover rate(P_e)	0.6
Mutation rate(P_m)	0.1
Selection Pressure(SP)	1.5
No. of consecutive generations without update	100

5.2 A small size example

A small size example with 10 customers ($n = 10$), 10 items ($q = 10$), and 3 zones ($p = 3$) MJRPDC considering MHCC is solved by the proposed GA-based approach in this section.

The planning region of this example refers to a real-world case of a leading bank with its branches scattered across Taipei City and New Taipei City in northern Taiwan. This bank uses more than 50 forms for its daily operations such as applications for personal loans, credit cards, and remittance requests (but, we pick only 10 forms for this example). The proposed GA solves this small-size problem for 30 times, and we show the districting decision of the best solution in Figure 5.1.

The centroids of three zones are located at (15.29, 12.11) , (4.88, 3.55), and (12.55, 3.74), respectively. The corresponding sets of customers in the three zones are $N_1 = \{2, 3, 7\}$, $N_2 = \{1, 5, 8, 10\}$, and $N_3 = \{4, 6, 9\}$.

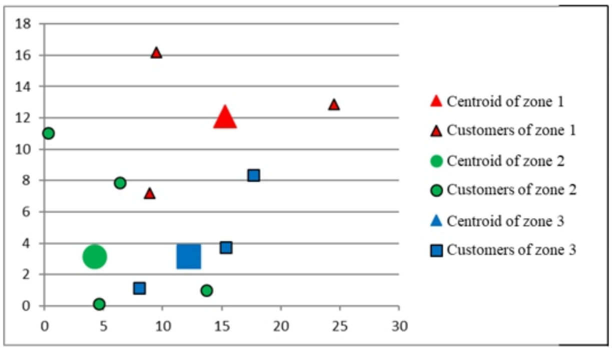

Figure 5.1: A small example with 10 customers and 10 items

5.3 Experiments on runtime growth

Two categories of experiments investigates the runtime growth with respect to the problem size in this subsection: (1) the growth of runtime with respect to the number of $customers$, and (2) the growth of runtime with respect to the number of $items$. For each instance, we run our GA for 30 times and save the optimal objective values as well as the average runtime.

We tested a total of eight levels for the number of customers, from 10 to 80, and randomly generated 30 instances with 10 items for each level in the first category of experiments. Table 5.3 summarizes the results of the average number of generations before termination and the average runtime.

Table 5.3: Runtime Growth with respect to number of customers

No. of customers	Average runtime(sec)
10	373.47
20	833.50
30	1676.35
40	3025.26
50	6403.60
60	5742.27
70	8521.13
80	13144.16

We analyze the growth of the average runtime with respect to the number of customers using polynomial regressions (see Neter et al., 1996). The curve fitting demonstrates that the average runtime grows in a cubic order with a 95%confidence level, as shown in Figure 5.2. Therefore, we may conclude that the proposed GA-based approach is effective for solving MJRPDC considering MHCC. The decision-makers may make observations on the impact of these parameters on the average total costs by testing different combinations of α, M_d , and p within a reasonable run time. These experiments are crucial for the managers before signing a contract with 3PL service providers.

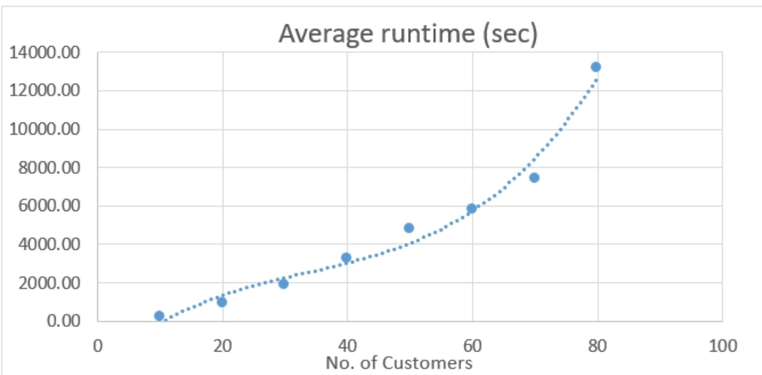

Figure 5.2: Runtime growth with respect to number of customers

In the second category of experiments, we observe the runtime growth with respect to the number of items, which serves as another important factor of the problem size. We collect runtime data from 30 randomly generated instances with 10 customers from each of 6 levels of the number of items, from 10 to 60 items. Table 5.4 presents the results of the average runtime. Again, we apply polynomial regressions for analysis and observe that the average runtime of our solution approach for the number of items grows in a cubic order with a 95%confidence level. Consequently, our results suggest that the proposed GA is efficient for solving MJRPDC considering MHCC.

Table 5.4: Runtime Growth with respect to number of items

No. of items	Average runtime
10	378.85
20	2010.35
30	6285.33
40	17844.95
50	26427.03
60	69737.53

6. CONCLUSIONS

This study investigates the multi-customer JRP with districting considerations and material handling capacity constraints (MJRPDC-MHCC) which is an extension of the multi-customer JRP (MJRP) with districting consideration. A 3PL service provider generally uses a specified area/designated dock in its warehouse for the logistics operations like sorting, packaging, and loading items for the customers of a particular zone. Therefore, when negotiating with a contract customer, the 3PL service provider must reserve an amount of capacity of the material handling equipment to guarantee the feasibility of a replenishment schedule for each zone. Therefore, we formulate a mathematical model for MJRP that takes into account the set of material handling capacity constraints. We propose a GA-based framework that solves MJRPDC with compactness and demand equity constraints by referring to Yao et al. (2020). We use the local minima obtained by the search algorithm of Yao et al. (2020) for unconstrained MJRP as the candidate solutions. Then, we employ a feasibility-testing heuristic proposed by Yao (2001) to generate a feasible replenishment schedule, and fix an infeasible solution as obtaining no feasible schedule. From our numerical experiments, we observe that the average runtime of our approach grows in a cubic order with respect to both the number of customers and the number of items. It is helpful and efficient for a logistic manager to use the proposed GA-based solution approach under different combinations of scenario or parameter settingsand examine the corresponding impact on the average total costs within a reasonable runtime. Therefore, we recommend the proposed GA-based solution approach as a useful decision-support tool for logistic managers before negotiating a service contract with a 3PL service provider.

REFERENCES

Amaya, C. A., Carvajal, J., & Castano, F. (2013). A heuristic framework based on linear programming to solve the constrained joint replenishment problem (c-jrp). *International Journal of Production Economics*, *144*, 243-247.

Bastos, L. d. S. L., Mendes, M. L., Nunes, D. R. d. L., Melo, A. C. S., & Carneiro, M. P. (2017). A systematic literature review on the joint replenishment problem solutions: 2006-2015. *Production*, *27*, e20162229.

Chan, C. K., Cheung, B. K. S., & Langevin, A. (2003). Solving the multi-buyer joint replenishment problem with a modified genetic algorithm. *Transportation Research Part B: Methodology 37*, 291–299.

Chopra, S., & Meindl, P. (2015). *Supply chain management: Strategy, planning and operations*. United States of Americaa: Pearson Education.

Cohen-Hillel, T., & Yedidsion, L. (2018). The periodic joint replenishment problem is strongly np-hard. *Mathematics of Operations Research*, *43*, 1269–1289.

Fung, R. Y. K., & Ma, X. (2001). A new method for joint replenishment problems. *Journal of Operational Research Society*, *52*, 358–362.

Goyal, S. K. (1974). Optimum ordering policy for a multi item single supplier system. *Operations Research Quarterly*, *25*, 293–298.

Goyal, S. K. (1975). Analysis of joint replenishment inventory systems with resource restriction. *Journal of the Operational Research Society*, *26*, 197-203.

Hoque, M. (2006). An optimal solution technique for the joint replenishment problem with storage and transport capacities and budget constraints. *European Journal of Operational Research*, *175*, 1033-1042.

Jeet, V., & Kutanoglu, E. (2007). Lagrangian relaxation guided problem space search heuristics for generalized assignment problems. *European Journal of Operational Research*, *182*, 1039–1056.

Khouja, M., & Goyal, S. (2008). A review of the joint replenishment problem literature: 1989–2005. *European Journal of Operational Research*, *186*, 1-16.

Lee, F. C., & Yao, M. J. (2003). A global optimum search algorithm for the joint replenishment problem under power-of-two policy. *Computers and Operations Research*, *30*, 1319–1333.

Michalewicz, Z. (1996). *Genetic algorithms + data structures = evolution programs,3rd ed.* Berlin, Heidelberg: Springer-Verlag.

Moon, I., & Cha, B. (2006). The joint replenishment problem with resource restriction. *European Journal of Operational Research*, *173*, 190-198.

Ongkunaruk, P., Wahab, M. I. M., & Chen, Y. (2016). A genetic algorithm for a joint replenishment problem with resource and shipment constraints and defective items. *International Journal of Production Economics*, *175*, 142-152.

Pohlheim, H. (1999). Evolutionary algorithms: Overview, methods and operators. Retrieved from `http://www.geatbx.com/`

Ricca, F., Scozzari, A., & Simeone, B. (2013). Political districting: from classical models to recent approaches. *Annals of Operations Research*, *204*, 271–299.

Shu, F. T. (1971). Economic ordering frequency for two items jointly replenished. *Management Science*, *17*(6), B-406–BB-410.

Simchi-Levi, D., Chen, X., & Bramel, J. (2014). *Logic of logistics: Theory, algorithms, and applications for logistics management. springer series in operations research and financial engineering.* New York: Springer.

Srinivas, . P. L. M., M. (1994). Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Transactions on System, Man, and Cybernetics*, *24*(4), 656–667.

Van Eijs, M. J. G. (1993). Journal of the operational research society. *European Journal of Operational Research*, *44*(2), 185–191.

Viswanathan, S. (2002). On optimal algorithms for the joint replenishment problem. *Journal of the Operational Research Society*, *53*, 1286–1290.

Yao, M. J. (2001). The peak load minimization problem in cyclic production. *Computers and Operations Research*, *28*, 1441-1460.

Yao, M. J. (2010). A search algorithm for solving the joint replenishment problem in a distribution center with warehouse-space restrictions. *International Journal of Operations Research*, *7*(2), 45-60.

Yao, M. J., Lin, Y., J., Lin, Y. L., & Fang, S. C. (2020). An integrated algorithm for solving multi-customer joint replenishment problem with districting consideration. *Transportation Research Part E*, *138*, to appear.

Zhang, J., Chung, S., H., & Lo, W. (2007). Clustering-based adaptive crossover and mutation probabilities for genetic algorithms. *IEEE Transactions on Evolutionary Computation*, *11*(3), 326–335.