How to Use Neural-enhanced Math-heuristic Deploy Multi-Tote Storage and Retrieval Systems

ABSTRACT

Keywords:

1. Introduction

2. Literature Review

2.1 Robotic Warehouse System

2.2 Joint Optimisation Decision Making

2.3 Learning Methodology for Combination Optimisation

3. Problem Description and Formulation

3.1 MTSR System with Sorting Conveyor

3.2 Three-Stage Separate Decision Method

3.3.1 Stage 1 Model：Assigning Orders to Picking Stations and Sequence

在第一阶段，考虑到订单的拣选时间和拣选站的订单容量上限，将订单分配给拣选站，该模型决策订单应该被分配给哪个拣选站以及订单的开始拣选时间和结束拣选时间。The variables are defined as follows:

Input parameters：

t_o^{order}:订单o在拣选站的拣选时间

Sets：

O:订单的集合

P:拣选站的集合

Parameters:

b_1:拣选站缓存的订单数量上限

M: A sufficiently large positive number

Variables:

$z_{o,p}$: 订单o是否被分配给第p个拣选站，若是则$z_{o,p}=1$，否则$z_{o,p}=0$

α_{o_1,o_2}: 订单o_1开始执行时订单o_2是否已经开始执行，若是则$\alpha_{o_1,o_2}=1$，否则$\alpha_{o_1,o_2}=0$

β_{o_1,o_2}: 订单o_1开始执行时订单o_2是否还未完成执行，若是则$\beta_{o_1,o_2}=1$，否则$\beta_{o_1,o_2}=0$

γ_{o_1,o_2}: 订单o_1开始执行时订单o_2是否正在执行，若是则$\gamma_{o_1,o_2}=1$，否则$\gamma_{o_1,o_2}=0$

δ_{o_1,o_2}：订单o_1开始执行时订单o_2是否正在同一个拣选站同时执行，若是则$\delta_{o_1,o_2}=1$，否则$\delta_{o_1,o_2}=0$

T_o^s:订单o的开始执行时间

T_o^e:订单o的结束执行时间

T_1:所有订单最大的结束执行时间

Mathematical model:

$$min \quad T_1 \tag{3-1}$$

$$T_1 \geq T_o^e, \forall o \in O \tag{3-2}$$

$$T_o^s + t_o^{order} \leq T_o^e, \forall o \in O \tag{3-3}$$

$$\sum_{p=1}^{P} z_{o,p} = 1, \forall o \in O \tag{3-4}$$

$$T_{o_1}^s - T_{o_2}^s \leq \alpha_{o_1,o_2} * M, \forall o_1, o_2 \in O \tag{3-5}$$

$$T_{o_2}^e - T_{o_1}^s \leq \beta_{o_1,o_2} * M, \forall o_1, o_2 \in O \tag{3-6}$$

$$\gamma_{o_1,o_2} \geq \alpha_{o_1,o_2} + \beta_{o_1,o_2} - 1, \forall o_1, o_2 \in O \tag{3-7}$$

$$\delta_{o_1,o_2} \geq \gamma_{o_1,o_2} + z_{o_1,p} + z_{o_2,p} - 2, \forall o_1, o_2 \in O, \forall p \in P \tag{3-8}$$

$$\sum_{o_2}^{O} \delta_{o_1,o_2} \leq b_1, \forall o_1 \in O \tag{3-9}$$

$$z_{o,p} \in \{0,1\}, \forall o \in O, p \in P \tag{3-10}$$

$$\alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2} \in \{0,1\}, \forall o_1, o_2 \in O \tag{3-11}$$

$$T_o^s, T_o^e \geq 0, \forall o \in O \tag{3-12}$$

$$T_1 \geq 0 \tag{3-13}$$

目标函数（3-1）表示最小化所有订单的最大完成时间，约束条件（3-2）表示所有订单的最大完成时间，约束条件（3-3）表示订单开始执行时间与结束执行时间之间的关系，约束条件（3-4）表示所有订单都需要完成，约束条件（3-5）-（3-8）是为了正确限制决策变量$\alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}$，约束条件（3-9）表示拣选站同时执行订单数量的上限，约束条件（3-10）-（3-13）表示定义的变量。

3.3.2 Stage 2 Model：Assigning Totes to Picking Stations and Sequence

在第二阶段，考虑到料箱的拣选时间和拣选站的料箱容量上限，将料箱分配给拣选站，该模型决策料箱应该被分配给哪个拣选站以及料箱的到达拣选站的时间、开始拣选时间和结束拣选时间。The variables are defined as follows:

Input parameters：

t_i^{toteIn}:料箱到达输送机入口的时间

$z_{o,p}$: 订单o是否被分配给第p个拣选站，若是则$z_{o,p}=1$，否则$z_{o,p}=0$

Sets：

O:订单的集合

P:拣选站的集合

P_1:所有料箱的第一次取货点集合，也是所有料箱的集合

Parameters：

p_i:料箱i需要的拣选时间，与料箱待选的 SKU 数量有关

$OT_{i,o}$: order cluster-tote (OT) pairs, 若料箱i属于订单o，则$OT_{i,o}=1$，否则$OT_{i,o}=0$

t_p^1:输送机入口到拣选站p所需要的时间

t_p^2:拣选站p到输送机出口的时间

b_2:拣选站缓存的料箱数量上限

Variables：

$y_{i,p}$:第i个料箱是否被分配给第p个拣选站，若是则$y_{i,p}=1$，否则$y_{i,p}=0$

$f_{i,j,p}$:第i个料箱是否先于料箱j到达拣选站p，若是则$f_{i,j,p}^2=1$，否则$f_{i,j,p}^2=0$

I_i:第i个料箱到达输送机入口的时间

$T_{i,p}^a$:料箱i到达拣选站p的时间

$T_{i,p}^s$:料箱i在拣选站p开始拣选的时间

$T_{i,p}^e$:料箱i在拣选站p结束拣选的时间

T_2:所有料箱最大的结束执行时间

Mathematical model：

$$min \quad T_2 \tag{3-14}$$

$$T_2 \geq T_{i,P}^e + t_P^2, \forall i \in P_1 \tag{3-15}$$

$$I_i \geq t_i^{toteIn}, \forall i \in P_1 \tag{3-16}$$

$$\sum_{i=1}^{P_1} \sum_{p=1}^{P} y_{i,p} = \sum_{i=1}^{P_1} \sum_{o=1}^{O} OT_{i,o}, \forall o \in O \tag{3-17}$$

$$OT_{i,o} * z_{o,p} \leq y_{i,p}, \forall i \in P_1, \forall o \in O, \forall p \in P \tag{3-18}$$

$$T_{i,1}^a = I_i + t_1^1, \forall i \in P_1 \tag{3-19}$$

$$T_{i,p}^s \geq T_{i,p}^a, \forall i \in P_1, \forall p \in P \tag{3-20}$$

$$T_{i,p}^e \geq T_{i,p}^s, \forall i \in P_1, \forall p \in P \tag{3-21}$$

$$T_{i,p}^{a} = T_{i,p-1}^{e} + t_{p}^{1} - t_{p-1}^{1}, \forall i \in P_1, \forall p \in 2, \ldots, P \quad (3-22)$$

$$T_{i,p}^{e} - T_{i,p}^{s} = p_i * y_{i,p}, \forall i \in P_1, \forall p \in P \quad (3-23)$$

$$T_{i,p}^{a} - T_{j,p}^{a} \leq \left(1 - f_{i,j,p}\right) * M, \forall i, j \in P_1, \forall p \in P \quad (3-24)$$

$$T_{i,p}^{a} - T_{j,p}^{a} \geq -f_{i,j,p} * M, \forall i, j \in P_1, \forall p \in P \quad (3-25)$$

$$T_{j,p}^{s} - T_{i,p}^{e} \geq \left(f_{i,j,p} + y_{i,p} + y_{j,p} - 3\right) * M, \forall i, j \in P_1, \forall p \in P \quad (3-26)$$

$$T_{i,p}^{s} - T_{i,p}^{a} \leq (b_2 - 1) * p_i, \forall i \in P_1, \forall p \in P \quad (3-27)$$

$$y_{i,p} \in \{0,1\}, \forall i \in P_1, \forall p \in P \quad (3-28)$$

$$f_{i,j,p} \in \{0,1\}, \forall i, j \in P_1, \forall p \in P \quad (3-29)$$

$$I_i \geq 0, \forall i \in P_1 \quad (3-30)$$

$$T_{i,p}^{a}, T_{i,p}^{s}, T_{i,p}^{e} \geq 0, \forall i \in P_1, \forall p \in P \quad (3-31)$$

$$T_2 \geq 0 \quad (3-32)$$

目标函数（3-14）表示最小化所有料箱到达输送机出口的最大时间。约束条件（3-15）表示料箱到达输送机出口的时间大于等于料箱在最后一个拣选站完成拣选的时间加上最后一个拣选站到输送机出口的时间。约束条件（3-16）表示料箱达到输送机入口的时间。约束条件（3-17）表示所有的料箱都要完成拣选。约束条件（3-18）表示只有当料箱i属于订单o，并且该订单o被分配给了拣选站p，那么该料箱i才能被分配给拣选站p。约束条件（3-19）表示料箱到达第一个拣选站的时间。约束条件（3-20）和（3-21）表示料箱在拣选站的开始拣选时间要大于料箱到达拣选站的时间，而料箱在拣选站的结束拣选时间要大于等于料箱到达拣选站的时间。约束条件（3-22）表示料箱到达下一个拣选站的时间等于料箱在上一个拣选站结束拣选的时间加上两个拣选站之间的路程时间。约束条件（3-23）表示若料箱被分配给了拣选站，那么结束拣选时间等于开始拣选时间加上料箱的拣选时间，否则结束拣选时间等于开始拣选时间。约束条件（3-24）和（3-25）表示若料箱i比料箱j先到达拣选站p，那么$f_{i,j,p}^{2} = 1$，否则$f_{i,j,p}^{2} = 0$。约束条件（3-26）表示若料箱i比料箱j先到达拣选站p，并且它们都被分配给了拣选站p，那么料箱j的开始拣选时间要大于等于料箱i的结束拣选时间。约束条件（3-27）表示每个拣选站缓存的料箱数量不能超过上限。约束条件（3-28）-（3-32）表示定义的变量。

3.3.3 Stage 3 Model：Assigning Totes to Robots and Robot Scheduling

在第三阶段，考虑到 AMRs 的旅行时间，将待出入库的料箱分配给 AMRs，并且 AMRs 一次性可以携带多个料箱，我们还需为 AMRs 规划路由，该模型决策待出入库的料箱应该被分配给哪个 AMRs 以及 AMRs 访问这些料箱点的顺序。The variables are defined as follows:

Input parameters：

$t_i^{toteOut}$:料箱到达输送机出口的时间

Sets：

K:AMRs 的集合

P_1:所有料箱的第一次取货点集合，也是所有料箱的集合

P_2:所有料箱的第二次取货点集合，也是输送机的出口

D_1:所有料箱的第一次送货点集合，也是输送机的入口

D_2:所有料箱的第二次送货点集合，也是所有料箱存储位置

W:所有 AMRs 的起点集合

N:所有点的集合$P_1 \cup P_2 \cup D_1 \cup D_2 \cup W$，包括两次取货点、两次送货点和 AMRs 的起点

Parameters：

q_i:点i的载重，其值为 1，因为我们将每个料箱视为重量为 1 的货物

Q: AMRs 的最大载重，其值为 8，因为每个 AMRs 最多可以搭载 8 个料箱

n:待拣选料箱的总数量

t_{ij}: AMRs 从料箱点i到料箱点j需要的行驶时间

s_i:料箱点i需要的服务时间，与料箱在货架中所处的层数成正比

e_i:料箱i最早的出库时间

l_i:料箱i最晚的出库时间

Variables：

x_{ij}:是否有机器人从料箱点i前往料箱点j，若是则$x_{ij} = 1$，否则$x_{ij} = 0$

u_i^k:机器人k是否经过料箱点i

Q_i:机器人在料箱点i时的载重

T_i:完成料箱点i任务的时间

T_{D_1}:所有D_1点最大的任务完成时间

T_{D_2}:所有D_2点最大的任务完成时间

T_3:所有机器人最大的任务完成时间

Mathematical model：

$$min \quad T_3 \tag{3-33}$$

$$T_3 \geq T_{D_1} \tag{3-34}$$

$$T_3 \geq T_{D_2} \tag{3-35}$$

$$T_{D_1} \geq T_i, \forall i \in D_1 \quad (3-36)$$

$$T_{D_2} \geq T_i, \forall i \in D_2 \quad (3-37)$$

$$t_i^{toteOut} \leq T_{i+n}, \forall i \in P_1 \quad (3-38)$$

$$\sum_{j=1,j\neq i}^{N} x_{ij} = \sum_{j=1,j\neq i}^{N} x_{ji}, \forall i \in N \quad (3-39)$$

$$\sum_{k=1}^{K} \sum_{j=1,j\neq i}^{N} x_{ij} \geq 1, \forall i \in P_1 \cup P_2 \cup D_1 \cup D_2 \quad (3-40)$$

$$u_i^k - u_j^k \geq M(x_{ij} - 1), \forall i \in N, \forall j \in N, \forall k \in K \quad (3-41)$$

$$u_i^k - u_j^k \leq M(1 - x_{ij}), \forall i \in N, \forall j \in N, \forall k \in K \quad (3-42)$$

$$u_i^k = 1, \forall i \in W, i = W_k, \forall k \in K \quad (3-43)$$

$$\sum_{k=1}^{K} u_i^k = 1, \forall i \in N \quad (3-44)$$

$$u_i^k = u_{i+2n}^k, \forall i \in P_1 \cup P_2, \forall k \in K \quad (3-45)$$

$$\sum_{j=1,j\neq i}^{N} x_{ij} \leq 1, \forall i \in W, i = W_k \quad (3-46)$$

$$Q_j \geq Q_i + q_i - Q(1 - x_{ij}), \forall i \in N \ \ i \neq j, \forall j \in P_1 \cup P_2 \cup D_1 \cup D_2 \quad (3-47)$$

$$0 \leq Q_i \leq Q, \forall i \in N \quad (3-48)$$

$$T_j \geq T_i + t_{ij} + s_i - M(1 - x_{ij}), \forall i \in N \ \ i \neq j, \forall j \in P_1 \cup P_2 \cup D_1 \cup D_2 \quad (3-49)$$

$$T_{i+2n} \geq T_i + t_{i,i+2n} + s_i, \forall i \in P_1 \cup P_2 \quad (3-50)$$

$$e_i \leq T_i \leq l_i, \forall i \in N \quad (3-51)$$

$$x_{ij} \in \{0,1\}, \forall i, j \in N \quad (3-52)$$

$$u_i^k \in \{0,1\}, \forall i \in N, \forall k \in K \quad (3-53)$$

$$Q_i \geq 0, \forall i \in N \quad (3-54)$$

$$T_i \geq 0, \forall i \in N \quad (3-55)$$

$$T_{D_1}, T_{D_2} \geq 0 \quad (3-56)$$

目标函数（3-33）表示最小化D_1点与D_2点中的最大任务完成时间，它是由约束条件（3-34）和（3-35）计算而来。约束条件（3-36）和（3-37）分别表示到达D_1点和D_2点的最大完成时间。约束条件（3-38）表示P_2点到达的时间要大于等于料箱到达输送机出口的时间。约束条件（3-39）表示 AMRs 在对料箱进行取货和送货过程中的流平衡约束。约束条件（3-40）表示 AMRs 需要完成所有的取货或送货任务。约束条件（3-41）-（3-45）表示同一个任务只能使用同一辆 AMRs 进行完成。约束条件（3-46）表示 AMRs 从其起点最多出发一次。约束条件（3-47）-（3-48）表示 AMRs 的载重约束，即搬运料箱数量的上限。约束条件（3-49）表示 AMRs 到达下一个目标点的时间要大于等于到达上一个点的时间加上上一个点的服务时间以及两个点之间的路程时间。

约束条件（3-50）表示料箱到达终点的时间要大于等于其到达起点的时间加上起点到终点的路程时间以及起点的服务时间。约束条件（3-51）表示料箱的完成时间必须满足时间窗的约束。约束条件（3-52）-（3-56）表示定义的变量。

3.3 An Integrated Decision Method

本小节提供了一个集成模型，称之为 JO-OTR 模型，用于联合优化分配订单任务至拣选站、料箱出入库的任务分配和机器人调度、以及料箱在输送机上的调度。为了建立集成模型，我们需要从目标函数和约束条件两个视角出发，通过三个阶段决策模型之间的联系，将所有的目标和约束进行整合。

3.3.1 Objective

集成模型的目标是尽快完成输入系统的所有订单，即最小化最大完成时间，而三个阶段的决策模型一共有四个最小化最大完成时间：所有订单的最大完成时间T_1、所有料箱到达输送机出口的最大完成时间T_2、机器人完成所有任务的最大完成时间T_3。由于集成模型是一个顺序决策过程，所以我们可以得到如下的关系表达式。

$$T_3 \geq T_2 \geq T_1 \tag{3-57}$$

由此，我们可以得出集成模型的优化目标为最小化机器人完成所有任务的最大完成时间，如下式所示。

$$min \quad T \tag{3-58}$$

$$T \geq T_i, \forall i \in N \tag{3-59}$$

3.3.2 Constraints

集成模型的约束是包含了三个阶段决策的所有约束，以及将三个阶段决策关联起来的衔接约束，而衔接约束正是每个阶段决策的输入参数，比如说t_o^{order}、t_i^{toteIn}、$t_i^{toteOut}$等，我们只需要将这些参数转换成衔接两个阶段决策的衔接约束，即可将孤立的决策整合成集成的模型。

我们在第一阶段决策中引入输入参数t_o^{order}的目的是为了获取订单的开始执行时间和结束执行时间，而实际上它与第三阶段决策中料箱的开始执行时间和结束执行时间有关，可将其转换为如下两个约束。

$$t_o^s - T_{i,p}^s \leq \left(3 - OT_{i,o} - y_{i,p} - z_{o,p}\right) * M, \forall i \in P_1, \forall o \in O, \forall p \in P \tag{3-60}$$

$$t_o^e - T_{i,p}^e \geq \left(OT_{i,o} + y_{i,p} + z_{o,p} - 3\right) * M, \forall i \in P_1, \forall o \in O, \forall p \in P \tag{3-61}$$

其中，约束条件 28 表示订单的开始执行时间小于等于其对应的料箱的开始执行时间；约束条件 29 表示订单的结束执行时间大于等于其对应的料箱的结束执行时间。

我们在第二阶段决策中引入输出参数t_i^{toteIn}的目的是为了获取料箱到达输送机入口的时间，而实际上它与第三阶段决策中D_1点的到达时间一致，可将其转换为如下的约束条件。

$$I_i \geq T_{i+2n}, \forall i \in P_1 \tag{3-62}$$

约束条件 30 表示料箱到达输送机入口的时间大于等于其到达D_1点的时间。

我们在第三阶段决策中引入输入参数$t_i^{toteOut}$的目的是为了获取P_2点可以开始执行的时间，而实际上它与第三阶段决策中料箱到达输送机出口的时间一致，可将其转换为如下的约束条件。

$$T_{i,P}^e + t_P^2 \leq T_{i+n}, \forall i \in P_1 \tag{3-63}$$

约束条件 30 表示P_2点的开始执行时间大于等于料箱到达最后一个拣选站的时间加上从最后一个拣选站到输送机出口的时间。

3.3.3 Mathematical model

综上所述，我们用约束条件（3-61）和（3-62）替代约束条件（3-3），用约束条件（3-62）替代约束条件（3-16），用约束条件（3-63）替代约束条件（3-38），以及将公式（3-58）作为最终的目标函数，并添加约束条件（3-59）得到集成的数学模型，如下所示。

$$min \quad T \tag{3-64}$$

$s.t.$

$$(3-4)-(3-12); (3-17)-(3-31); (3-39)-(3-55);$$

$$T \geq T_i, \forall i \in N \tag{3-65}$$

$$t_o^s - T_{i,p}^s \leq \left(3 - OT_{i,o} - y_{i,p} - z_{o,p}\right) * M, \forall i \in P_1, \forall o \in O, \forall p \in P \tag{3-66}$$

$$t_o^e - T_{i,p}^e \geq \left(OT_{i,o} + y_{i,p} + z_{o,p} - 3\right) * M, \forall i \in P_1, \forall o \in O, \forall p \in P \tag{3-67}$$

$$T_{i,P}^e + t_P^2 \leq T_{i+n}, \forall i \in P_1 \tag{3-68}$$

$$I_i \geq T_{i+2n}, \forall i \in P_1 \tag{3-69}$$

$$T \geq 0 \tag{3-70}$$

目标函数（3-64）是为了最大限度地缩短所有任务完成的最大时间。约束条件（3-4）-（3-12）属于第一阶段决策。约束条件（3-17）-（3-31）属于第二阶段决策。约束条件（3-39）-（3-55）属于第三阶段决策。约束条件（3-66）和（3-67）是衔接第一阶段决策和第二阶段决策的约束。约束条件（3-68）是衔接第二阶段决策和第三阶段决策的约束。约束条件（3-69）是衔接第三阶段决策和第二阶段决策的约束。约束条件（3-70）表示定义的变量。

4. Mathematical Programming Based Model Decomposition

在本小节中，我们为集成模型 JO-OTR 及与其类似的多项决策联合优化的大规模混合整数规划模型提供了一种基于数学规划来特殊设计的，进行模型分解的框架，我们称之为基于数学规划的模型分解 MP-MD。复杂的多项决策联合优化的模型中，我们会发现，有若干的决策变量对模型的影响程度远高于模型中其它的一些变量，比如在 JO-OTR 模型中的x_{ij}，y_{ip}，z_{op}，我们可以将这些变量统称为关键决策变量。在一个完整的模型中来同时优化这些复杂的关键决策变量，会使模型规模变得极大从而不可求解，所以我们就在思考有没有一种方法可以让这些关键决策变量先在各自的子问题中单独的进行求解，再统一所有的子问题结果并作用于主问题中，得到最终原问题的解决方案。

正是这个动机，促使我们为类似于 JO-OTR 这种多项决策联合优化的大规模 MIP 提出了 MP-MD 框架，它是一种对决策变量进行旋转固定并求解的方法。它主要的流程描述如下：首先，对于一个复杂的 MIP 我们需要识别其中的关键决策变量；然后，根据关键决策变量识别的结果，构造对应的基于逻辑的主问题；紧接着，我们需要基于原问题，为不同的关键决策变量构造对应的松弛模型，我们称这个松弛模型为子问题；最后，我们还可以通过推导原问题或子问题的上下界（在 JO-OTR 中是下界，因为它是最小化问题）和进行子问题关键决策变量的初始设置来加速求解。接下来，就将详细地阐述我们是如何应用 MP-MD 框架求解 JO-OTR 的。

4.1 Logic-based Master Problem

由于 MP-MD 的子问题与关键变量是一一对应的，并且每个子问题的求解都需要固定其它的关键变量，所以 MP-MD 的主问题是通过逻辑约束来限制每次迭代优化的子问题以及决策当前是否已经求解得到最优解。每次主问题的输入是上一次迭代优化的子问题及其目标函数值，输出则是下一次需要迭代优化的子问题，主问题的模型如下。

主问题：

$$min \sum_{s=1}^{S} \theta_s \cdot \mathbb{F}_s^S(\mathcal{X}_s) \tag{4-1}$$

$s.t.$

$$\sum_{s=1}^{S} \theta_s = 1, \forall s \in S \tag{4-2}$$

$$\theta_s \leq 1 - \vartheta_s, \forall s \in S \tag{4-3}$$

$$\theta_s \in \{0,1\}, \forall s \in S \quad (4-4)$$

ϑ_s表示最近一次迭代优化的子问题是s，$\mathcal{X}_s$表示子问题s的关键变量，$\mathbb{F}_s^S(\mathcal{X}_s)$表示子问题$s$在关键变量$\mathcal{X}_s$情况下的最优目标函数值，我们需要决策$\theta_s$来确定下一次迭代优化的子问题，所以目标函数 0 表示我们将选择当前情况下$\mathbb{F}_s^S(\mathcal{X}_s)$最小的子问题$s$来进行优化，约束条件 1 表示每次我们只能选择一个子问题，约束条件 2 表示我们不能旋转最近一次已经迭代优化的子问题，约束条件 3 表示定义的决策变量θ_s。

Theorem 1. Logic-based Master Problem 将在有限多次迭代后终止并获得最优解，终止时 Logic-based Master Problem 的最优解既是各子问题的最优解，也是原问题的最优解，即$\mathbb{F}^{\mathrm{O}} = \mathbb{F}^{\mathrm{M}} = \mathbb{F}_s^S(\mathcal{X}_s)$。

Proof.

4.2 Relaxation-based Subproblem

对于任意子问题s，它的模型都是在固定除$\mathcal{X}_s$之外所有的关键变量而获得的，它是在原模型的基础上松弛掉除$\mathcal{X}_s$之外所有的关键变量，被松弛掉的$\mathcal{X}_s$都将以固定的值$\mathcal{X}_s'$传入模型的约束之中。在 JO-OTR 中，3 个子问题的模型如下所示。

4.2.1 Relaxed Model for Sub-problem 1

子问题 1：固定x_{ij}和y_{ip}为x_{ij}'和y_{ip}'

$$min \quad \mathbb{F}_1^S(z_{op}) \quad (4-5)$$

$s.t.$

$(3-4)-(3-12); (3-19)-(3-22); (3-24)-(3-25); (3-27); (3-29)-(3-31);$

$(3-43)-(3-45); (3-48); (3-50)-(3-51); (3-53)-(3-55); (3-68)-(3-69);$

$$\mathbb{F}_1^S(z_{op}) \geq T_i, \forall i \in N \quad (4-6)$$

$$\sum_{i=1}^{P_1}\sum_{p=1}^{P} y_{ip}' = \sum_{i=1}^{P_1}\sum_{o=1}^{O} OT_{i,o}, \forall o \in O \quad (4-7)$$

$$OT_{i,o} * z_{o,p} \leq y_{ip}', \forall i \in P_1, \forall o \in O, \forall p \in P \quad (4-8)$$

$$T_{i,p}^e - T_{i,p}^s = p_i * y_{ip}', \forall i \in P_1, \forall p \in P \quad (4-9)$$

$$T_{j,p}^s - T_{i,p}^e \geq \left(f_{i,j,p} + y_{ip}' + y_{jp}' - 3\right) * M, \forall i,j \in P_1, \forall p \in P \quad (4-10)$$

$$\sum_{j=1,j\neq i}^{N} x_{ij}' = \sum_{j=1,j\neq i}^{N} x_{ji}', \forall i \in N \quad (4-11)$$

$$\sum_{k=1}^{K}\sum_{j=1,j\neq i}^{N} x_{ij}' \geq 1, \forall i \in P_1 \cup P_2 \cup D_1 \cup D_2 \quad (4-12)$$

$$u_i^k - u_j^k \geq M(x'_{ij} - 1), \forall i \in N, \forall j \in N, \forall k \in K \quad (4-13)$$

$$u_i^k - u_j^k \leq M\left(1 - x'_{ij}\right), \forall i \in N, \forall j \in N, \forall k \in K \quad (4-14)$$

$$\sum_{j=1, j\neq i}^{N} x'_{ij} \leq 1, \forall i \in W, i = W_k \quad (4-15)$$

$$Q_j \geq Q_i + q_i - Q\left(1 - x'_{ij}\right), \forall i \in N \ \ i \neq j, \forall j \in P_1 \cup P_2 \cup D_1 \cup D_2 \quad (4-16)$$

$$T_j \geq T_i + t_{ij} + s_i - M\left(1 - x'_{ij}\right), \forall i \in N \ \ i \neq j, \forall j \in P_1 \cup P_2 \cup D_1 \cup D_2 \quad (4-17)$$

$$t_o^s - T_{i,p}^s \leq \left(3 - OT_{i,o} - y'_{ip} - z_{o,p}\right) * M, \forall i \in P_1, \forall o \in O, \forall p \in P \quad (4-18)$$

$$t_o^e - T_{i,p}^e \geq \left(OT_{i,o} + y'_{ip} + z_{o,p} - 3\right) * M, \forall i \in P_1, \forall o \in O, \forall p \in P \quad (4-19)$$

$$\mathbb{F}_1^S\left(z_{op}\right) \geq 0 \quad (4-20)$$

4.2.2 Relaxed Model for Sub-problem 2

子问题 2：固定x_{ij}和z_{op}为x'_{ij}和z'_{op}

$$min \quad \mathbb{F}_2^S\left(y_{ip}\right) \quad (4-21)$$

$s.t.$

$$(3-5)-(3-7);(3-9);(3-11)-(3-12);(3-43)-(3-45);(3-17)-(3-31);$$

$$(3-48);(3-50)-(3-51);(3-53)-(3-55);(3-68)-(3-69);$$

$$\mathbb{F}_2^S\left(y_{ip}\right) \geq T_i, \forall i \in N \quad (4-22)$$

$$\sum_{p=1}^{P} z'_{op} = 1, \forall o \in O \quad (4-23)$$

$$\delta_{o_1,o_2} \geq \gamma_{o_1,o_2} + z'_{o_1p} + z'_{o_2p} - 2, \forall o_1, o_2 \in O, \forall p \in P \quad (4-24)$$

$$t_o^s - T_{i,p}^s \leq \left(3 - OT_{i,o} - y_{ip} - z'_{op}\right) * M, \forall i \in P_1, \forall o \in O, \forall p \in P \quad (4-25)$$

$$t_o^e - T_{i,p}^e \geq \left(OT_{i,o} + y_{ip} + z'_{op} - 3\right) * M, \forall i \in P_1, \forall o \in O, \forall p \in P \quad (4-26)$$

$$\mathbb{F}_2^S\left(y_{ip}\right) \geq 0 \quad (4-27)$$

4.2.3 Relaxed Model for Sub-problem 3

子问题 3：固定y_{ip}和z_{op}为y'_{ip}和z'_{op}

$$min \quad \mathbb{F}_3^S\left(x_{ij}\right) \quad (4-28)$$

$s.t.$

$$(3-5)-(3-7);(3-9);(3-11)-(3-12);(3-19)-(3-22);(3-24)-(3-25);$$

$$(3-27);(3-29)-(3-31);(3-39)-(3-55);(4-7)-(4-10);(4-23)-(4-24);$$

$$\mathbb{F}_3^S(x_{ij}) \geq T_i, \forall i \in N \quad (4-29)$$

$$t_o^s - T_{i,p}^s \leq \left(3 - OT_{i,o} - y'_{ip} - z'_{op}\right) * M, \forall i \in P_1, \forall o \in O, \forall p \in P \quad (4-30)$$

$$t_o^e - T_{i,p}^e \geq \left(OT_{i,o} + y'_{ip} + z'_{op} - 3\right) * M, \forall i \in P_1, \forall o \in O, \forall p \in P \quad (4-31)$$

$$\mathbb{F}_3^S(x_{ij}) \geq 0 \quad (4-32)$$

4.3 Initial Set of Variables

4.3.1 Initial Set of the Order Assignment and Sequence

订单的指派和排序主要有两个目的，第一个目的是为了让相似结构的订单能够指派到同一拣选站上，第二个目的是为了让在同一拣选站上订单执行顺序前后的订单都与该订单相似，以此来达到使用最少的料箱数量满足尽可能多的订单的目的。所以，订单的指派和排序决策的初始解构造包含了两部分，第一部分采用**公式 1** 来评估订单的相似性并将其分为若干类别指派至拣选站，第二部分采用 **Qi** 提出的贪婪算法来对指派至每个拣选站上的订单进行排序，完整的算法伪代码如 Algorithm1 所示。

$$\psi_{i_1,i_2} = \frac{O_{i_1}^T O_{i_2}}{O_{i_1}^T O_{i_1} + O_{i_2}^T O_{i_2} - O_{i_1}^T O_{i_2}}$$

We define the similarity of two orders as the proportion of the number of common SKUs to the number of all SKUs that the two orders contain. 其中，ψ_{i_1,i_2}表示订单i_1和i_2之间的相似性，$O_{set_{ik}}$表示订单i是否包含第k种 SKU，因此$O_{i_1} = \left(O_{set_{i1}}, O_{set_{i2}}, \ldots, O_{set_{iK}}\right)^T$。

Algorithm 1: Initial solution of the order assignment and sequence.	
Input:	$O, P_1, P, OT_{i,o}$
Output:	$z_{o,p}, \alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}$
1	Randomly select P orders as representatives of the initial clusters.
2	$for\ o \in O$:
3	For each order, calculate its similarity to each cluster representation ψ_{o,o_p}.
4	Assign orders to the most similar clusters: $max\left(\psi_{o,o_p}\right) \rightarrow z_{op} = 1$.
5	Calculate the average similarity of the cluster as a new cluster representative.
6	$end\ for$.
7	$for\ p \in P$:
8	Select the order with the largest SKU as the first order o_p to be executed.
9	$end\ for$.
10	$for\ p \in P$:
11	Calculate O_p according to z_{op}
12	$for\ o \in O_p$:
13	Calculate ψ_{o,o_p} and select $max\left(\psi_{o,o_p}\right) \rightarrow o$ as the next order to be executed.
14	Replace o_p with o。
15	$end\ for$.
16	$end\ for$.
Output:	$z_{o,p}, \alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}$

4.3.2 Initial Set of the Tote Assignment and Sequence

料箱的指派和排序主要的目的是为了确定料箱需要前往哪些拣选站以及料箱出库的顺序是怎样的。所以，我们在根据订单的指派确定料箱的指派之后，依据当前需要的紧急程度来确定每个料箱出库的先后顺序，所谓料箱的当前需要的紧急程度就是描述在当前情况下，拣选站上正处于拣选站槽口中的所有订单与其绑定的次数，如**公式 3** 所示，越多的订单需要该料箱，该料箱的紧急程度越大，就越应该先进行出库，完整的算法伪代码如 Algortihm2 所示。

$$\omega_i = \sum_{p=1}^{P} OT_{i,o} \cdot z_{o,p}, \forall o \in O_{on}$$

其中，O_{on}表示当前情况下，正处于各个拣选站槽口上的订单。

Algorithm 2: Initial solution of the tote assignment and sequence.	
Input:	$O, P_1, P, OT_{i,o}, z_{o,p}, \alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}$
Output:	$y_{i,p}, I_i, f_{i,j,p}$
1	$i = 0$.
2	while $i \le P_1$:
3	$for\ \ i \in P_1$:
4	Calculate the number of bindings per bin: ω_i.
5	$end\ \ for$.
6	The $max(\omega_i)$ is used as the currently executed tote.
7	Update order status at each picking station.
8	$i = i + 1$.
9	$end\ \ while$.
Output:	$y_{i,p}, I_i, f_{i,j,p}$

4.3.3 Initial Set of the Robot Allocation and Scheduling

机器人任务分配及调度的目的是为了确定哪些料箱任务由哪些机器人去执行，并且机器人执行这些料箱任务的具体路由是什么。由于其与路径问题十分的相似，因此我们采用 Adapted nearest neighbor (ANN)【1】进行初始解的构造，它是一种快速简单的贪婪选择策略，实验证明它在路径相关的问题中表现相当的好。从定义的或者随机指定的初始节点开始，ANN 为机器人指定最近的一个未访问节点作为下一个节点，直到所有节点都包含在路径中。针对本文提出的问题，我们只需要保证每次插入节点的满足优先级和 AMRs 的容量限制，便可以确保构建的初始解始终是可行的，完整的算法伪代码如 Algortihm3 所示。

Algorithm 3: Initial solution of the robot allocation and scheduling.	
Input:	$K, P_1, P_2, D_1, D_2, W, N, I_i, f_{i,j,p}$
Output:	$x_{i,j}, u_i^k, Q_i, T_i$
1	$n = 0,\ q_k = 0$.
2	while $n \le P_1 + P_2$:
3	Select a P_1 or P_2, and assign it to an AMRs.
4	$for\ \ k \in K$:
5	The current location of AMRs is p.
6	$if\ \ q_k \le Q$:
7	When the load of the AMRs is less than Q, a P_1 or P_2 closest to the current position of the AMRs is selected to join in the path of that AMR.
8	$x_{p,p_1}\ \ or\ \ x_{p,p_2} = 1, u_{p_1}^k\ \ or\ \ u_{p_2}^k = 1, Q_{p_1}\ \ or\ \ Q_{p_2} = q_k$.
9	$n = n + 1, q_k = q_k + 1$.

10	*else*:
11	Add D_1 or D_2 that already has a P_1 or P_2 counterpart in the path of the AMRs.
12	$q_k = 0$.
13	*end if*.
14	*end for*.
15	*end while*.
Output:	$x_{i,j}, u_i^k, Q_i, T_i$

4.4 Lower Bounds for Original Problem

集成模型 JO-TOR 的目标值我们用T来表示，它表示所有料箱中最后一个被完成料箱的完成时间，即所有料箱的完成时间中最大的哪个料箱所花费的时间。每个料箱的开始时间为其离开储位的时刻，然后它会随着机器人前往输送机的入口，接着它会在输送机上的拣选站内完成拣选站并前往输送机的出口，最后它会随着机器人前往自己的储位。我们假设料箱的两类起终点集合分别为P_1, P_2, D_1, D_2，机器人的集合为R，机器人起点的集合为W，拣选站的集合为P，机器人r从其起点w前往料箱起点p_1的时间为t_{wp_1}，料箱从储位前往输送机入口的时间和从输送机出口返回储位的时间分别为$t_{p_1d_1}$和$t_{p_2d_2}$，料箱在输送机上输送时间为t_c，那么每个料箱的完成时间t_{p1}可以表示为如下式所示。

$$t_{p1} = t_{wp_1} + t_{p_1d_1} + t_c + t_{p_2d_2}$$

由于我们推导的是集成模型 JO-TOR 的下界，所以机器人r从其起点w前往料箱起点p_1的时间我们取最小值，如下式所示。同理可得，料箱在输送机上的时间我们也取最小值，如下式所示。所以，每个料箱的完成时间t_{p1}可以进一步修改为如下式所示。

$$\min_{w \in W}\{t_{wp_1}\}$$

$$t_c = t_P^1 + t_P^2 + p_{p1}$$

$$t_{p1} = \min_{w \in W}\{t_{wp_1}\} + t_{p_1d_1} + t_P^1 + t_P^2 + p_{p1} + t_{p_2d_2}$$

所以集成模型 JO-TOR 的下界为所有料箱完成时间的最大值，如下式所示。

$$LB_1 = \max_{p_1 \in P_1}\{\min_{w \in W}\{t_{wp_1}\} + t_{p_1d_1} + t_P^1 + t_P^2 + p_{p1} + t_{p_2d_2}\}$$

进一步的，我们将机器人的数量考虑进来，我们发现当机器人数量少于料箱数量时，会存在一个机器人需要完成多个料箱情况的出现，此时集成模型 JO-TOR 的下界就变成了所有机器人中最后一个完成任务的机器人的完成时间，我们用t_1^r表示机器人r的进行取货的时间，用t_2^r表示机器人r进行送货的时间，那么机器人r的完成时间如下式所示。

$$t_r = t_1^r + t_2^r + t_c$$

所以集成模型 JO-TOR 的下界为所有机器人完成时间的最大值，如下式所示。

$$LB_2 = \max_{r \in R}\{t_r\}$$

我们注意到，同时计算多个机器人的t_1^r或t_2^r是一件很困难的事情，因此，我们根据机器人数量和料箱数量，将其转换为寻找一个最小的t_1^r和t_2^r来计算LB_2，具体见公式 3 至公式 4。首先，我们计算出每个机器人平均需要完成的料箱任务数量$\bar{n}$，然后，计算每个机器人完成这$\bar{n}$个料箱任务至少需要多少时间，我们取其中最小的时间作为LB_2，并且在这个过程中我们不考虑机器人的载重问题，显然此时所需的时间一定为集成模型 JO-TOR 的下界。

$$min \quad t_1^r$$

$s.t.$

$$t_1^r \geq T_i, \forall i, j \in W \cup P_1 \cup D$$

$$\sum_{j \in W \cup P_1} v_{ji} \geq \tau_i, \forall i \in P_1$$

$$\sum_{j \in P_1 \cup D} v_{ij} \geq \tau_i, \forall i \in P_1$$

$$\sum_{j \in P_1} v_{ij} = 1, i = W$$

$$\sum_{j \in P_1} v_{ji} = 1, i = D$$

$$\sum_{i \in P_1} \tau_i \geq \bar{n}$$

$$T_j \geq T_i + t_{ij} + s_i - M(1 - v_{ij}), \forall i, j \in W \cup P_1 \cup D$$

$$\tau_i \in \{0,1\}, \forall i \in P_1$$

$$v_{ij} \in \{0,1\}, \forall i, j \in W \cup P_1 \cup D$$

$$T_i \geq 0, \forall i \in W \cup P_1 \cup D$$

其中，τ_i表示料箱i是否分配给该机器人，v_{ij}表示是否从点i前往点j，T_i表示机器人到达点i的时间。目标函数表示最小化机器人r的取货时间t_1^r，约束 1 表示机器人的完成时间，约束 2-3 表示料箱点的出度和入度，约束 4-5 表示起点或终点的出度或入度，约束 6 表示时间递推约束，约束 7-9 是决策变量的取值范围。

通过上述的模型，我们可以计算出每个机器人的完成时间t_1^r，同理可得，也可以计算出每个机器人的t_2^r，它们是完全一样的问题。所以，集成模型 JO-TOR 的下界就可以设置为最小的t_1^r和t_2^r，如公式 10 所示。

$$LB_2 = \min_{r \in R}\{t_1^r\} + t_c + \min_{r \in R}\{t_2^r\}$$

5. Neural-enhanced Revolving Math-heuristic Algorithm for Integrated Method

本节详细阐述了一种针对我们提出的 MP-MD 方法特殊设计的一种“神经增强的旋转数学启发式算法”（DeepRMHA），以在短时间内解决大规模的集成模型 JO-TOR。DeepRMHA 是一种具有迭代固定变量的两层解决方案框架，图 1 说明了 DeepRMHA 的核心思想，我们将变量划分为了多个类别，并且以旋转的方式临时和迭代的固定一些变量。其中，主问题的数学启发式框架是为了控制算法的迭代和结束，以及确定固定和松弛的关键变量，子问题的神经增强的子算法是为了求解几个子问题，即松弛的子问题模型。

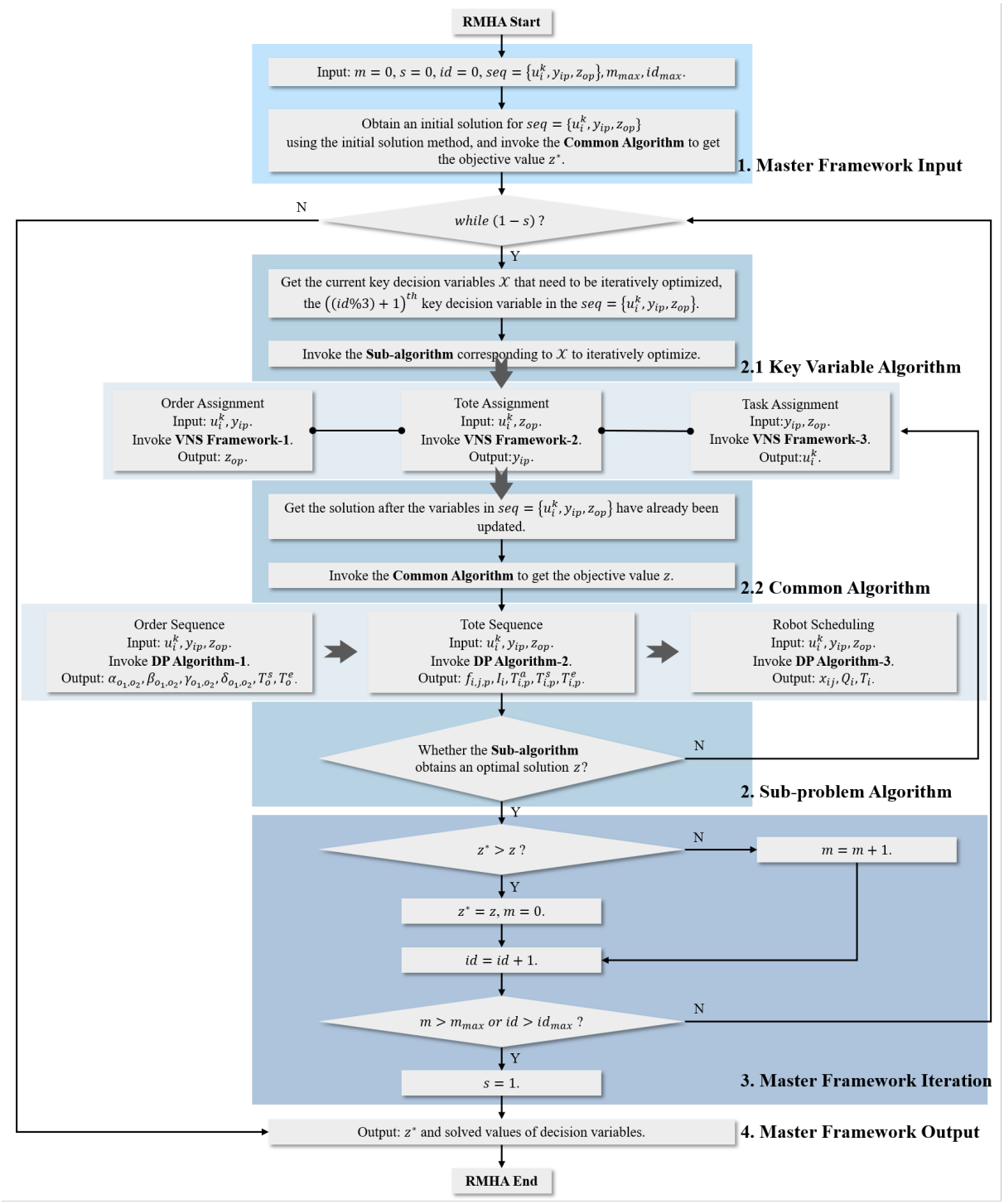

Figure 1. RMHA framework.

5.1 Math-heuristic Framework for Master-problem

主问题的数学启发式框架在 DeepRMHA 中的作用是确定每次固定的关键变量、更新问题的上下界、以及控制算法的开始和结束。首先，算法会进行解决方案的初始化，然后，算法会迭代固定关键变量并每次更新解决方案的上下界，最后，达到迭代次数上限或者满足退出条件则退出算法并输出最终的解决方案，它的伪代码如算法 4 所示。

Algorithm 4: Math-heuristic framework for master-problem.	
Input:	$m, s, id, seq, m_{max}, id_{max}$.
Output:	Optimal objective value z^* and solved values of decision variables.
1	$m = 0,\ s = 0,\ id = 0,\ seq = (u_i^k, y_{ip}, z_{op})$.
2	Obtain an initial solution for $seq = (u_i^k, y_{ip}, z_{op})$ using the initial solution method, and invoke the **Common Algorithm** to get the objective value z^*.
3	$while\ (1 - s)$:
4	Get the current key decision variables $\mathcal{X}$ that need to be iteratively optimized, the $((id\%3) + 1)^{th}$ key decision variable in the $seq = (u_i^k, y_{ip}, z_{op})$.
5	Invoke the **Sub-algorithm** corresponding to $\mathcal{X}$ to iteratively optimize.
6	Obtain the **Sub-algorithm** optimised solution z.
7	$if\ (z < z^*)$:
8	$z^* = z,\ m = 0$.
9	$else$:
10	$m = m + 1$.
11	$end\ if$.
12	$if\ (m \geq m_{max}\ \ or\ \ id \geq id_{max})$:
13	$s = 1$.
14	$end\ if$.
15	$end\ while$.
Output:	Optimal objective value z^* and solved values of decision variables.

5.2 Sub-algorithms for Sub-problems

在 DeepRMHA 中，我们需要针对三个子问题的松弛模型分别设计三个子算法进行求解，而每个子问题的松弛模型我们将分为两部分求解，第一部分是基于可变邻域搜索 VNS 的框架来求解每个子问题的关键决策变量值的邻域，得到关键决策变量值的邻域和另外两个固定决策变量的值之后，我们设计了一个基于动态规划的、通用的算法来求解模型中剩下的其它整数决策变量和连续型决策变量，Common Algorithm 的详细内容见 5.3。

对于每个子问题的关键决策变量，它们都属于指派问题，因此我们基于指派问题的 VNS 框架设计了一个求解三个子问题的通用型 Sub-problems，它主要包括初始解的生成、邻域的 Shaking 和 Local search、新解的接受和输出等部分，它的伪代码如 Algorithm 5 所示。

Algorithm 5: Sub-algorithm for sub-problem.	
Input:	$z_{o,p}, y_{i,p}(z_{o,p}, u_i^k\ \text{or}\ y_{i,p}, u_i^k),\ count_{max},\ iter_{max}$, a set of neighbourhood structures $N_k\ \ for\ \ k = 1,2,\dots,K$ for shaking, a set of neighbourhood structures $N_l\ \ for\ \ l = 1,2,\dots,L$ for local search.
Output:	Optimal objective value z^* and $u_i^k(y_{i,p}\ \text{or}\ z_{o,p})$ and $\alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}, I_i, f_{i,j,p}, x_{i,j}, Q_i, T_i$.
1	Initialize $u_i^k(y_{i,p}\ \text{or}\ z_{o,p})$ according to the solution obtained in the previous step.
2	Invoke **Common Algorithm**, obtain $\alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}, I_i, f_{i,j,p}, x_{i,j}, Q_i$, and obtain objective value z of the initial solution.
3	Set $z^* = z,\ z_{min} = z$.
4	Set $count = 0,\ iter = 0$.
5	$while\ \ iter \leq iter_{max}$:
6	$k = 1$.
7	$while\ \ k \leq K$:
8	Shaking:
9	$u_i^{k'}(y'_{i,p}\ \text{or}\ z'_{o,p}) = Shake(u_i^k(y_{i,p}\ \text{or}\ z_{o,p}), k)$, obtain updated $u_i^{k'}(y'_{i,p}\ \text{or}\ z'_{o,p})$, and invoke **Common Algorithm**, obtain objective value z'.
10	Local search:
11	$l = 1$.
12	$while\ \ l \leq L$:
13	$u_i^{k''}(y''_{i,p}\ \text{or}\ z''_{o,p}) = LS(u_i^{k'}(y'_{i,p}\ \text{or}\ z'_{o,p}), l)$.

14	Invoke **Common Algorithm**, obtain objective value z''.
15	$if\ \ z'' \le z'$:
16	$u_i^{k'}(y'_{i,p}\ \text{or}\ z'_{o,p}) = u_i^{k''}(y''_{i,p}\ \text{or}\ z''_{o,p}),\ l = 1.$
17	$else$:
18	$l = l + 1.$
19	$end\ \ if.$
20	$end\ \ while.$
21	Move or not.
22	$if\ \ z'' \le z$:
23	$u_i^k(y_{i,p}\ \text{or}\ z_{o,p}) = u_i^{k''}(y''_{i,p}\ \text{or}\ z''_{o,p}),\ z_{min} = z'',\ k = 1.$
24	$else$:
25	$k = k + 1.$
26	$end\ \ if.$
27	$end\ \ while.$
28	$if\ \ z_{min} \le z^*$:
29	$z^* = z_{min},\ count = 0.$
30	$else$:
31	$count = count + 1.$
32	$end\ \ if.$
33	$if\ \ count \le count_{max}$:
34	$break.$
35	$end\ \ if.$
36	$iter = iter + 1.$
37	$end\ \ while.$
Output:	Optimal objective value z^* and $u_i^k(y_{i,p}\ \text{or}\ z_{o,p})$ and $\alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}, I_i, f_{i,j,p}, x_{i,j}, Q_i, T_i.$

5.3 Dynamic Programming Based Common Algorithm for All Sub-problems

在 5.2 节中的子算法里需要多次调用 Common Algorithm，而这个 Common Algorihm 实际上已知三个关键决策变量$u_i^k, y_{i,p}, z_{o,p}$的值之后，求解其余剩余决策变量的值的一个过程，而这些剩余变量的求解也可以分为三个部分：1）已知u_i^k求解$x_{i,j}$；2）已知$y_{i,p}, I_i$求解$f_{i,j,p}$；3）已知$z_{o,p}, T_{i,p}^a, T_{i,p}^s, T_{i,p}^e$求解$T_o^s, T_o^e, \alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}$。它们之间求解的先后顺序为：先求解部分 1，得到机器人的路径以及机器人到达各任务点的时间，从而得到料箱在仓库中的流动情况；然后，根据料箱的流动情况，再求解部分 2，得到料箱在输送机上的流动情况；最后，根据料箱在输送机上的流动情况，求解部分 3，得到订单在各拣选站上的拣选情况；最终，完成所有决策的求解。

除了上述三部分内容需要求解的决策变量之外，Common Algorithm 还需要求解问题中剩下的所有决策变量的值，比如说一些时间变量、辅助变量等，它的伪代码如 Algorithm 6 所示。

Algorithm 6: Dynamic programming based Common Algorithm for sub-algorithm.	
Input:	$z_{o,p}, y_{i,p}, u_i^k.$
Output:	Objective value z and $\alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}, I_i, f_{i,j,p}, x_{i,j}, Q_i, T_i.$
1	Based on u_i^k and equations (5-1)-(5-8), invoke the **Dynamic Programming Algorithm 1** and calculate $x_{i,j}$.
2	$for\ \ i,j \in N$:
3	$if\ \ i! = j$:
4	According to $Q_j \ge Q_i + q_i - Q(1 - x_{ij})$, calculate Q_i.
5	According to $T_j \ge T_i + t_{ij} + s_i - M(1 - x_{ij})$ and $T_{i+2n} \ge T_i + t_{i,i+2n} + s_i$, calculate T_i.
6	$end\ \ if.$
7	$end\ \ for.$
8	$for\ \ i \in P_i$:

9	According to $I_i \geq T_{i+2n}$, calculate I_i.
10	$end\ \ for$.
11	Based on $y_{i,p}, I_i$ and equations (3-19)-(3-27), invoke the **Dynamic Programming Algorithm 2** and calculate $T^a_{i,p}, T^s_{i,p}, T^e_{i,p}, f_{i,j,p}$.
12	$for\ \ i \in P_i$:
13	$if\ \ T^e_{i,P} + t^2_P \leq T_{i+n}$ is not satisfied:
14	$z = +\infty$, return.
15	$end\ \ if$.
16	$end\ \ for$.
17	$for\ \ p \in P, i \in P_i$:
18	$if\ \ T^s_{i,p} - T^a_{i,p} \leq (b_2 - 1) * p_i$ is not satisfied:
19	$z = +\infty$, return.
20	$end\ \ if$.
21	$end\ \ for$.
22	Based on $z_{o,p}, T^a_{i,p}, T^s_{i,p}, T^e_{i,p}$ and equations (3-5)-(3-8), (3-60)-(3-61), invoke the **Dynamic Programming Algorithm 3** and calculate $T^s_O, T^e_O, \alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}$.
23	$for\ \ o_1 \in O$:
24	$if\ \ \sum_{o_2}^{O} \delta_{o_1,o_2} \leq b_1$ is not satisfied:
25	$z = +\infty$, return.
26	$end\ \ if$.
27	$end\ \ for$.
28	$for\ \ i \in N$:
29	$z \geq T_i$.
30	$end\ \ for$.
Output:	Objective value z and $\alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}, I_i, f_{i,j,p}, x_{i,j}, Q_i, T_i$.

Common Algorithm 中所需的三个动态规划算法分别对应三个部分，所以接下来我们将逐一分析这三个部分，并且为每一部分设计一个动态规划算法。

第一部分是已知u_i^k求解$x_{i,j}$，对于每个机器人k的决策来说，我们可以把起点同时设置为终点，将其转换为一个带访问顺序约束和资源限制约束的 TSP，数学模型如（5-1）-（5-8）所示。

$$min \sum_{i=1}^{N+1} \sum_{j=1}^{N+1} d_{ij}\, x_{ij} \tag{5-1}$$

$$s.t. \sum_{i=1}^{N} x_{ij} = 1, \forall j \in \{2,3,\dots,N+1\}, i \neq j \tag{5-2}$$

$$\sum_{j=2}^{N+1} x_{ij} = 1, \forall i \in \{1,2,\dots,N\}, i \neq j \tag{5-3}$$

$$\mu_i - \mu_j + N x_{ij} \leq N - 1, \forall i \in \{1,2,\dots,N\}, \forall j \in \{2,3,\dots,N+1\}, i \neq j \tag{5-4}$$

$$\mu_j - \mu_i + N \omega_{ij} \geq N + 1, \forall i \in \{1,2,\dots,N\}, \forall j \in \{2,3,\dots,N+1\}, i \neq j \tag{5-5}$$

$$x_{ij} \in \{0,1\}, \forall i, j \in \{1,2,\dots,N+1\} \tag{5-6}$$

$$\omega_{ij} \in \{0,1\}, \forall i, j \in \{1,2,\dots,N+1\} \tag{5-7}$$

$$\mu_i \geq 0, \forall i \in \{1,2,\dots,N+1\} \tag{5-8}$$

其中，N是点的个数，1 为机器人的起点，$N+1$为机器人的虚拟的终点（起点），d_{ij}为点i到点j的距离，目标函数（5-1）表示最小化行驶距离，约束条件（5-2）和（5-3）分别表示入度为 1 和出度为 1，约束条件（5-4）用于破子环，约束条件（5-5）表示遵循访问顺序，约束条件（5-6）-（5-8）表示决策变量的取值范围。

为了求解这个模型，我们为它设计了一种动态规划算法，状态转移方程如（5-9）所示，伪代码如 Algorithm 7 所示。

Algorithm 7: Dynamic programming algorithm 1.

Input: Graph $G=(V,A)$, start node $s=1$.
Output: Optimal routing r^*.

$for\ \ i=2,3,\dots,|V|:$
  $f(i,\emptyset)\leftarrow d_{i,s}$.
  The next node of $(i,\emptyset)\leftarrow s$.
$end\ \ for$.
$for\ \ k=V,V-1,V-2,\dots,2:$
  $for\ \ current\ \ node\ \ i=2,3,\dots,|V|:$
    $for\ \ S_l\subset V\backslash\{i\}\ \ and\ \ |S_l|=|V|-k:$
      Calculate the current load Q_i of the robot.
      $if\ \ i\ \ is\ \ delivery\ \ node\ \ and\ \ i's\ \ pick\ \ up\ \ node\ \ \in S_l:$
        $break$.
      $end\ \ if$.
      $if\ \ Q_i>Q:$
        $break$.
      $end\ \ if$.
      $f(i,S_l)=min(d_{i,v}+f(v,S_l-\{v\}))$.
      The next node of $(i,S_l)\leftarrow \underset{v}{arg}\ \underset{v\in S_l}{min}(d_{i,v}+f(v,S_l-\{v\}))$.
    $end\ \ for$.
  $end\ \ for$.
$end\ \ for$.
$S_l\leftarrow V-\{s\}$.
$f(s,S_l)\leftarrow \underset{v\in S_l}{min}(d_{s,v}+f(v,S_l-\{v\}))$.
The next node of $(s,S_l)\leftarrow \underset{v}{arg}\ \underset{v\in S_l}{min}(d_{i,v}+f(v,S_l-\{v\}))$.
Obtain the optimal routing r^* by checking the next node from the current node.

Output: Optimal routing r^* and optimal objective $f(s,S_l)$.

第二部分是已知$y_{i,p},I_i$求解$T^a_{i,p},T^s_{i,p},T^e_{i,p},f_{i,j,p}$，它是一个简单的排序问题，我们根据公式（3-19）-（3-27）来构建我们的前向动态规划，它伪代码如 Algorithm 8 所示。

Algorithm 8: Dynamic programming algorithm 2.

Input: $y_{i,p},I_i$.
Output: $T^a_{i,p},T^s_{i,p},T^e_{i,p},f_{i,j,p}$.

According to I_i, sort the totes and get $seq_{tote}=\{s_1,s_2,\dots,s_n\}$.
$for\ \ i\in seq_{tote}:$
  $for\ \ p\in P:$
    $for\ \ j\in(i+1,seq_{tote}):$
      $f_{i,j,p}=1$.
    $end\ \ for$.
    $if\ \ p=1:$
      According to $T^a_{i,1}=I_i+t^1_1$, calculate $T^a_{i,1}$.
      $if\ \ i=1:$
        $T^s_{1,1}=T^a_{1,1}$.
      $else$:
        $if\ \ T^a_{i,1}\geq T^s_{i-1,1}:$
          $T^s_{i,1}=T^a_{i,1}$.
        $else$:
          $T^s_{i,1}=T^s_{i-1,1}$.
        $end\ \ if$.
      $end\ \ if$.
      According to $T^e_{i,1}-T^s_{i,1}=p_i*y_{i,1}$, calculate $T^e_{i,1}$.
    $else$:
      According to $T^a_{i,p}=T^e_{i,p-1}+t^1_p-t^1_{p-1}$, calculate $T^a_{i,p}$.
      $if\ \ i=1:$
        $T^s_{1,p}=T^a_{1,p}$.
      $else$:
        $if\ \ T^a_{i,p}\geq T^s_{i-1,p}:$
          $T^s_{i,p}=T^a_{i,p}$.

26	$else$:
27	$T_{i,p}^{s} = T_{i-1,p}^{s}$.
28	$end\ \ if$.
29	$end\ \ if$.
30	According to $T_{i,p}^{e} - T_{i,p}^{s} = p_i * y_{i,p}$, calculate $T_{i,p}^{e}$.
31	$end\ \ if$.
32	$end\ \ for$.
33	$end\ \ for$.
Output:	$T_{i,p}^{a}, T_{i,p}^{s}, T_{i,p}^{e}, f_{i,j,p}$.

第三部分是已知$z_{o,p}, T_{i,p}^{a}, T_{i,p}^{s}, T_{i,p}^{e}$求解$T_o^s, T_o^e, \alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}$的过程，对于每个拣选站$p$来说，这部分决策的内容依然如第二部分内容一样，是一个简单的排序问题，我们根据公式（3-5）-（3-8）以及（3-60）-（3-61）来构建我们的前向动态规划，伪代码如 Algorithm 9 所示。

Algorithm 9: Dynamic programming algorithm 3.	
Input:	$z_{o,p}, T_{i,p}^{a}, T_{i,p}^{s}, T_{i,p}^{e}$.
Output:	$T_o^s, T_o^e, \alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}$.
1	$for\ \ o \in O, p \in P$:
2	$if\ \ z_{o,p} == 1$.
3	$T_o^s = min_{i \in P_1}(OT_{io} \cdot T_{i,p}^{s})$.
4	$T_o^e = max_{i \in P_1}(OT_{io} \cdot T_{i,p}^{e})$.
5	$end\ \ if$.
6	$end\ \ for$.
7	$for\ \ o_1, o_2 \in O$:
8	According to $T_{o_1}^{s} - T_{o_2}^{s} \le \alpha_{o_1,o_2} * M,\ T_{o_2}^{e} - T_{o_1}^{s} \le \beta_{o_1,o_2} * M, \forall o_1, o_2 \in O,\ \gamma_{o_1,o_2} \ge \alpha_{o_1,o_2} + \beta_{o_1,o_2} - 1, \forall o_1, o_2 \in O$, calculate $\alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}$.
9	$end\ \ for$.
10	$for\ \ o_1, o_2 \in O, p \in P$:
11	According to $\delta_{o_1,o_2} \ge \gamma_{o_1,o_2} + z_{o_1,p} + z_{o_2,p} - 2$, calculate δ_{o_1,o_2}.
12	$end\ \ for$.
Output:	$T_o^s, T_o^e, \alpha_{o_1,o_2}, \beta_{o_1,o_2}, \gamma_{o_1,o_2}, \delta_{o_1,o_2}$.

6. Numerical Experiments

6.1 Experiment Settings

我们为各种实验随机生成了一系列的标准算例，包括了小规模算例、中规模算例和大规模算例三部分，如表 5、表 6、表 7 所示。每个标准算例的设置包括了五点要素，仓库布局长或宽方向上块的数量W和L、待出库料箱的总数N、待完成的订单任务总数O、ACR 的数量R和拣选站的数量P。其中，仓库中每一个块的长度是 18，宽度是 2，高度是 10，所以每个块都包括了 360 个料箱储位。

Table 1. Information on instances.

S-s ID	W	L	N	O	R	P	M-s ID	W	L	N	O	R	P	L-s ID	W	L	N	O	R	P
I-1	2	4	2	2	2	2	II-1	6	8	12	10	6	4	III-1	10	12	30	20	10	5
I-2	2	4	3	2	2	2	II-2	6	8	14	10	6	4	III-2	10	12	60	30	15	5
I-3	2	4	4	2	2	2	II-3	6	8	16	10	6	4	III-3	10	12	90	40	20	5
I-4	4	4	5	5	4	4	II-4	8	8	18	15	8	6	III-4	12	12	110	50	25	10
I-5	4	4	6	5	4	4	II-5	8	8	20	15	8	6	III-5	12	12	140	60	30	10
I-6	4	4	7	5	4	4	II-6	8	8	22	15	8	6	III-6	12	12	170	70	35	10
I-7	6	6	8	8	6	6	II-7	10	8	24	20	10	8	III-7	14	12	210	80	40	15
I-8	6	6	9	8	6	6	II-8	10	8	26	20	10	8	III-8	14	12	240	90	45	15
I-9	6	6	10	8	6	6	II-9	10	8	28	20	10	8	III-9	14	12	270	100	50	15

6.2 Comparison of the Three Stage Separate Decision Method and the Integrated Decision Method

6.2.1 Small-scale Instance

Table 2. Gurobi and MPMD results for small-scale instances.

Instance ID	Gurobi				MPMD						
	UB_G	LB_G	T_G	G^*	UB_M	LB_M	T_M	Δ_{UB}^{Obj}	Δ_{LB}^{Obj}	Δ_G^{Time}	M^*
I-1	573.00	573.00	0.05	10/10	573.00	573.00	0.09	0.00%	0.00%	0.04	10/10
I-2	573.00	573.00	0.23	10/10	573.00	573.00	0.24	0.00%	0.00%	0.01	10/10
I-3	597.40	597.40	2.27	10/10	597.40	597.40	1.48	0.00%	0.00%	-0.79	10/10
I-4	678.00	678.00	1.37	10/10	678.00	678.00	0.48	0.00%	0.00%	-0.89	10/10
I-5	683.50	683.50	6.38	10/10	683.50	683.50	0.71	0.00%	0.00%	-5.67	10/10
I-6	683.50	683.50	32.61	10/10	683.50	683.50	0.94	0.00%	0.00%	-31.67	10/10
I-7	684.00	684.00	34.07	10/10	684.00	684.00	3.44	0.00%	0.00%	-30.63	10/10
I-8	684.00	684.00	137.31	10/10	684.00	684.00	13.90	0.00%	0.00%	-123.41	10/10
I-9	684.00	684.00	341.76	10/10	684.00	684.00	13.19	0.00%	0.00%	-328.57	10/10
Average	648.93	648.93	61.78	10/10	648.93	648.93	3.83	0.00%	0.00%	-57.95	10/10

Note. $G^*(M^*)$*: the number of optimal solutions obtained by Gurobi (MPMD);* $\Delta_{UB}^{Obj} = (UB_G - UB_M)/UB_G$*;* $\Delta_{LB}^{Obj} = (LB_G - LB_M)/LB_G$*;* $\Delta_G^{Time} = T_G - T_M$.

Table 3. RMHA, R* and G* Results for small-scale instances.

Instance ID	Integrated algorithm MPMD		Integrated algorithm RMHA		3 stages rule-based algorithm R^3		3 stages greedy algorithm G^3		Gap		
	O_M	T_M	O_R	T_R	O_{R^3}	T_{R^3}	O_{G^3}	T_{G^3}	Δ_R^{Obj}	$\Delta_{R^3}^{Obj}$	$\Delta_{G^3}^{Obj}$
I-1	573.00	573.00	573.00	9.44	577.00	0.00	576.00	0.00	0.00%	-0.70%	-0.52%
I-2	573.00	573.00	573.00	10.32	658.00	0.00	663.00	0.00	0.00%	-14.83%	-15.71%
I-3	597.40	597.40	597.40	12.30	659.00	0.00	693.60	0.00	0.00%	-10.31%	-16.10%
I-4	678.00	678.00	678.00	15.25	699.00	0.00	706.00	0.00	0.00%	-3.10%	-4.13%
I-5	683.50	683.50	683.50	17.45	767.00	0.00	832.00	0.00	0.00%	-12.22%	-21.73%
I-6	683.50	683.50	683.50	18.52	767.00	0.00	832.00	0.00	0.00%	-12.22%	-21.73%
I-7	684.00	684.00	684.00	18.20	741.60	0.00	710.00	0.00	0.00%	-8.42%	-3.80%
I-8	684.00	684.00	684.00	19.02	741.60	0.00	755.00	0.00	0.00%	-8.42%	-10.38%
I-9	684.00	684.00	684.00	18.97	788.00	0.00	762.00	0.00	0.00%	-15.20%	-11.40%
Average	648.93	648.93	648.93	15.50	710.91	0.00	725.51	0.00	0.00%	-9.55%	-11.80%

Note. $\Delta_R^{Obj} = (O_M - O_R)/O_M$*;* $\Delta_{R^3}^{Obj} = (O_M - O_{R^3})/O_M$ $\Delta_{G^3}^{Obj} = (O_M - O_{G^3})/O_M$.

6.2.2 Medium-scale Instance

Table 4. Gurobi and MPMD results for medium-scale instances.

Instance ID	Gurobi				MPMD						
	UB_G	LB_G	T_G	G^*	UB_M	LB_M	T_M	Δ_{UB}^{Obj}	Δ_{LB}^{Obj}	Δ_G^{Time}	M^*
II-1	892.00	892.00	742.02	10/10	892.00	892.00	83.30	0.00%	0.00%	-658.72	10/10
II-2	\	856.60	3600.00	0/10	892.00	882.30	3600.00	\	-2.91%	0.00	0/10
II-3	\	849.50	3600.00	0/10	892.10	892.10	1854.32	\	-4.78%	0.00	10/10
II-4	\	848.00	3600.00	0/10	934.70	891.30	3600.00	\	-4.86%	0.00	0/10
II-5	\	904.00	3600.00	0/10	937.60	895.90	3600.00	\	0.90%	0.00	0/10
II-6	\	843.00	3600.00	0/10	941.00	898.80	3600.00	\	-6.21%	0.00	0/10
II-7	\	834.00	3600.00	0/10	931.00	899.20	3600.00	\	-7.25%	0.00	0/10
II-8	\	831.00	3600.00	0/10	931.00	897.00	3600.00	\	-7.36%	0.00	0/10
II-9	\	830.00	3600.00	0/10	940.40	897.00	3600.00	\	-7.47%	0.00	0/10
Average	\	854.23	3282.45	1/9	921.31	893.96	3015.29	\	-4.44%	-73.19	2/9

Table 5. RMHA, R* and G* Results for medium-scale instances.

Instance ID	Integrated algorithm MPMD	Integrated algorithm RMHA	3 stages rule-based algorithm R^3	3 stages greedy algorithm G^3	Gap

	O_M	T_M	O_R	T_R	O_{R^3}	T_{R^3}	O_{G^3}	T_{G^3}	Δ_R^{Obj}	$\Delta_{R^3}^{Obj}$	$\Delta_{G^3}^{Obj}$
II-1	892.00	83.30	893.40	26.97	1179.00	0.00	910.00	0.00	-0.16%	-32.17%	-2.02%
II-2	892.00	3600.00	894.50	36.11	1361.60	0.00	950.00	0.00	-0.28%	-52.65%	-6.50%
II-3	892.10	1854.32	898.40	39.27	1361.60	0.00	970.50	0.00	-0.71%	-52.63%	-8.79%
II-4	934.70	3600.00	937.30	47.08	1306.00	0.00	1134.00	0.00	-0.28%	-39.72%	-21.32%
II-5	937.60	3600.00	938.30	60.23	1313.20	0.00	1126.00	0.00	-0.07%	-40.06%	-20.09%
II-6	941.00	3600.00	940.40	71.24	1313.20	0.00	1123.80	0.00	0.06%	-39.55%	-19.43%
II-7	931.00	3600.00	932.10	106.38	1216.30	0.00	1098.00	0.00	-0.12%	-30.64%	-17.94%
II-8	931.00	3600.00	938.60	74.11	1233.00	0.00	1077.10	0.00	-0.82%	-32.44%	-15.69%
II-9	940.40	3600.00	940.60	92.04	1400.00	0.00	1108.00	0.00	-0.02%	-48.87%	-17.82%
Average	921.31	3015.29	923.73	61.49	1298.21	0.00	1055.27	0.00	-0.26%	-40.91%	-14.54%

6.2.3 Large-scale Instance

Table 6. Gurobi and MPMD results for large-scale instances.

Instance ID	Gurobi				MPMD						
	UB_G	LB_G	T_G	G^*	UB_M	LB_M	T_M	Δ_{UB}^{Obj}	Δ_{LB}^{Obj}	Δ_G^{Time}	M^*
III-1	\	975.00	3600.00	0/10	1131.20	975.00	3600.00	\	0.00%	0	0/10
III-2	\	1023.00	3600.00	0/10	1198.50	1023.00	3600.00	\	0.00%	0	0/10
III-3	\	\	3600.00	0/10	1410.90	1033.00	3600.00	\	\	0	0/10
III-4	\	\	3600.00	0/10	1627.50	1019.00	3600.00	\	\	0	0/10
III-5	\	\	3600.00	0/10	1618.10	\	3600.00	\	\	0	0/10
III-6	\	\	3600.00	0/10	1914.40	\	3600.00	\	\	0	0/10
III-7	\	\	3600.00	0/10	1959.00	\	3600.00	\	\	0	0/10
III-8	\	\	3600.00	0/10	1893.20	\	3600.00	\	\	0	0/10
III-9	\	\	3600.00	0/10	1907.80	\	3600.00	\	\	0	0/10
Average	\	\	3600.00	0/10	1628.96	\	3600.00	\	\	0	0/10

Table 7. RMHA, R* and G* Results for large-scale instances.

Instance ID	Integrated algorithm MPMD		Integrated algorithm RMHA		3 stages rule-based algorithm R^3		3 stages greedy algorithm G^3		Gap		
	O_M	T_M	O_R	T_R	O_{R^3}	T_{R^3}	O_{G^3}	T_{G^3}	Δ_R^{Obj}	$\Delta_{R^3}^{Obj}$	$\Delta_{G^3}^{Obj}$
III-1	1131.20	3600.00	1128.40	70.09	1784.00	0.00	1422.00	0.00	0.25%	-57.71%	-25.71%
III-2	1198.50	3600.00	1166.20	80.06	2882.00	0.00	1564.00	0.00	2.70%	-140.47%	-30.50%
III-3	1410.90	3600.00	1184.70	202.18	3141.00	0.00	1708.20	0.01	16.03%	-122.62%	-21.07%
III-4	1627.50	3600.00	1207.00	786.61	4492.00	0.00	1963.00	0.02	25.84%	-176.01%	-20.61%
III-5	1618.10	3600.00	1213.60	1207.16	3848.00	0.00	1765.30	0.02	25.00%	-137.81%	-9.10%
III-6	1914.40	3600.00	1277.40	1802.32	4374.00	0.00	1916.00	0.04	33.27%	-128.48%	-0.08%
III-7	1959.00	3600.00	1313.50	1902.06	4574.00	0.00	2007.00	0.05	32.95%	-133.49%	-2.45%
III-8	1893.20	3600.00	1330.20	1987.27	4472.00	0.00	1941.00	0.08	29.74%	-136.21%	-2.52%
III-9	1907.80	3600.00	1274.30	2028.35	4412.00	0.00	1910.00	0.12	33.21%	-131.26%	-0.12%
Average	1628.96	3600.00	1232.81	1118.46	3775.44	0.00	1799.61	0.04	24.32%	-131.77%	-10.48%

6.3 Impact of Order, tote, robot Composition and Warehouse Configurations

6.3.1 Impact of Order Composition

6.3.2 Impact of Tote Composition

6.3.3 Impact of Robot Composition

6.3.4 Impact of Warehouse Configurations

7. Conclusion and Future Research