Computers and Electronics in Agriculture 158 (2019) 313–325

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag

Original papers

Adaptive large neighborhood search for scheduling sugarcane inbound logistics equipment and machinery under a sharing infield resource system

Rapeepan Pitakaso[a], Kanchana Sethanan[b,*]

[a] Metaheuristics for Logistic Optimization Laboratory, Department of Industrial Engineering, Faculty of Engineering, Ubon Ratchathani University, Thailand
[b] Research Unit on System Modeling for Industry, Department of Industrial Engineering, Faculty of Engineering, Khon Kaen University, Thailand

ARTICLE INFO

Keywords:
Adaptive Large Neighborhood Search (ALNS)
Sharing infield resource system
Sugarcane
Scheduling
Routing

ABSTRACT

This paper presents the ALNS metaheuristics, employing the idea of DE to solve the mechanical harvester assignment and routing problem with time windows (HARPTW) to maximize the total area serviced by a mechanical harvester under a sharing infield resource system. The effective ALNS is designed to solve large-scale problems integrating the mechanical harvester assignment problem (HAP) and the mechanical harvester routing problem (HRP). The newly developed destroy and repair methods are unique and effective. Additionally, four new formulas have been developed to calculate the probability to accept the worse solution using linear and parabola functions instead of the exponential function that is used mostly in the literature. The numerical results show that the parabola function, which uses the information about the solution quality, outperforms all other proposed heuristics. This demonstrates that the proposed heuristics are very efficient and are not only useful for reducing the infield operations costs of small growers, but also for efficient management of the inbound logistics equipment and machinery of the sugarcane supply system.

1. Introduction

Sugarcane is an important crop for Thailand's economy. Presently, the Thai sugar industry is faced with difficulties of increasing production costs. They are thus searching for solutions to improve the profitability for both sugarcane growers and the sugar mill industry. In Thailand, the cost of sugarcane harvesting and transportation comprises a large portion of the Thai sugarcane total production cost, with the average cost of harvesting accounting for 66% of the total labor cost, or equivalent to 35% of the total cost. The average cost for sugarcane transportation was 2.79 US$/ton in 2003 (Office of Agricultural Economics, 2003). Nearly half of the total cost is devoted to harvesting and transportation.

Presently, infield machinery use, particularly in harvesting operations, has increased, since labor wages have risen greatly, and less labor is available for manual infield work (Neungmatcha and Sethanan, 2015). In many parts of the world where sugarcane production is steadily increasing, the harvesting mode has switched from manual harvesting to the use of mechanical harvesters (Salassi and Champagne, 1998). This is likely to be the most crucial factor in reducing future sugarcane production costs (Ahmed and Alam-Eldin, 2015), since it can complete the harvest faster with more sugarcane harvested per unit of time than manual harvesting and loading. However, a mechanical harvester is very expensive. The fuel costs have risen significantly faster, over the past many years, than the growth in the sugarcane price paid to growers for sugarcane delivered to the mill (Sethanan and Neungmatcha, 2016).

Most sugarcane growers in Thailand (approximately 80%) are small-scale growers. Since most of them do not normally possess mechanical harvesters, at present these growers rent a mechanical harvester from outside people who own mechanical harvesters. However, the outside people are not under contract to the mill, which usually leads to paying unfair hiring costs, in cash, for harvesting, loading and transporting by the small growers, resulting in high debts for the small-scale growers.

These special problems prevailing in this industry relate to its having limited resources and intense competition. As a result of resource constraints, the operating costs have increased greatly, especially for the inbound logistics process that involves planting, harvesting, and transportation, all of which need cost reduction. In practice, the small-scale growers are not able to manage all procedures effectively, because of their lack of bargaining power and inadequate resources, which may eventually force them to give up growing sugarcane or to shift to other economic crops. The Office of Agricultural

* Corresponding author at: Research Unit on System Modeling for Industry, Industrial Engineering Department, Faculty of Engineering, Khon Kaen University, Khon Kaen 40002, Thailand.
E-mail addresses: enrapepi@ubu.ac.th (R. Pitakaso), skanch@kku.ac.th (K. Sethanan).

https://doi.org/10.1016/j.compag.2019.02.001
Received 14 September 2018; Received in revised form 28 January 2019; Accepted 1 February 2019
Available online 21 February 2019

Economics and the Ministry of Agriculture and Cooperatives recommend the sharing of mechanical harvesters among growers within a village for further savings in cost for small-scale growers, and also to increase the utilization of the mechanical harvesters.

The mill factories, therefore, play an important role in liaising between the small, medium or larger scale growers and third-party inbound logistics providers, under the mill contract in their service areas, in terms of allocation and scheduling of available resources and guaranteeing fair service prices. Hence, in this resource sharing system, small-scale growers will get mechanical harvester services from the medium and larger growers, or third-party inbound logistics providers who own mechanical harvesters, rather than spend a lot of money to own mechanical harvesters themselves. The medium and large-scale growers, or third-party inbound logistics providers, can service the small-scale growers as long as their mechanical harvesters are completely available aside from harvesting their own sugarcane or for others. If the mills could manage the needs of small-scale growers in terms of time period and a number of mechanical harvesters required and the availability of mechanical harvesters belonging to third-party inbound logistics providers or medium and large-scale growers more efficiently, the cost of sugar production by small-scale growers and the mills would be lower. The medium and large-scale growers or third-party inbound logistics providers (i.e., agents), would not only get paid on time as contracted but could also earn more revenue from sharing their available mechanical harvesters with others.

Thus, this research considers the allocation and routing of the available mechanical harvesters to service the needs of small-scale growers. In the mechanical harvester allocation problem (HAP), it is decided when and where each mechanical harvester should be assigned to which sugarcane fields, while in the mechanical harvester routing problem (HRP), the route of each mechanical harvester is decided. A key input to the HAP is the total time available for each mechanical harvester, and also the total number of mechanical harvesters available in each time period. These inputs depend on the harvest schedule assigned to sugarcane fields for the previous periods, a decision that belongs to the HRP. Thus, the two problems are closely connected, and it is therefore likely that higher quality solutions can be obtained by solving a model integrating the two problems. Since each sugarcane field should be harvested differently at each time period, due to its commercial cane sugar (CCS) value, a harvesting scheduling at a proper time is required to increase CCS yields. In this paper, we consider the mechanical harvester assignment and routing problem with time windows (HARPTW) to maximize the total areas serviced by the mechanical harvesters under a resource sharing system.

However, there are no publications on the HARPTW, and there is to date no research that has applied Adaptive Large Neighborhood Search algorithms (ALNS) to solve the HARPTW. The ALNS was proposed by Ropke and Pisinger (2006). It extends the LNS by allowing repair operators and multiple destroy to be used within the same search. A destroy operator is chosen at each iteration, on the basis of choosing probabilities which are adjusted dynamically during the search process, according to the performance obtained by the operators in the previous iterations (Mancini, 2016). Recently, the ALNS has been applied in various research areas. For example, Muller (2011) applied the ALNS to solve the resource-constrained project scheduling problem. Aksen et al. (2014) applied the ALNS to solve the selective and periodic inventory routing problem, and Monroy-Licht et al. (2017) used the ALNS for the rural postman problem with time windows. Due to the attractive features of Adaptive Large Neighborhood Search, this research focused on the implementation of this methodology to solve the HARPTW with the expectation of maximizing the total areas serviced by a resource sharing system.

The main contributions of this paper are fourfold. First, the HAP and HSP are integrated. Secondly, we present 8 destroy and 4 repair methods. Normally, a good solution of a problem that combines assignment and routing problems is obtained from an algorithm that continuously constructs the solution. Information from the assignment and routing phases is needed for exchange with each other. Therefore, if we destroy the connection of the assignment and the routing phases by the destroy and repair methods, the solution can get worse. We design our algorithms based on the preservation of this attribute. Our algorithm consists of two phases, which are to construct the sequences of the sugarcane fields and the mechanical harvesters; then the construction of the complete algorithm will be performed. The destroy method has been done to determine the sequence of the sugarcane fields for each mechanical harvester, then the repair method that includes the various construction methods has been executed. Finally, a good and effective solution from the destroy and repair method will be obtained. As a consequence, we perform the destroy method with the incomplete solution and we hide the algorithm to transform incomplete solutions to complete solutions, and therefore our destroy and repair methods are unique and effective. Thirdly, we present four new formulas to accept the worse solution to be the starting solution for the search to calculate the probability to accept the worse solution. These formulas use linear and parabola functions instead of the exponential function that is mostly used in the literature. The parabola function allows a high chance of acceptance probability in the beginning and the last phase of the algorithms, so that it has a better chance to escape from a local optimum. Fourthly, adaptive large-scale neighborhood search originally constructs the solution (complete solution) then iteratively applies destroy and repair methods to improve the solution quality. In our research, the destroy method has been made iteratively in the incomplete solution, such as destroy the list of customers in the VRP problem that has not yet been routed, then the repair method has been applied. For the repair method, a good constructive algorithm is hidden inside it to get a good solution. The final solution will be more flexible, since the destroy and repair methods have been executed only in the complete solution.

The remainder of this paper is organized as follows. In Section 2, a brief literature review of the HARPTW problem and its variants is presented. In Section 3, the mathematical model is developed. In Section 4, the ALNS for the HARPTW problem is presented. Computational results are discussed in Section 5. Finally, the conclusion and the future research direction are presented in the last section.

2. Literature review

Recently, the feedstock supply to sugar mills has received significant attention in the academic literature. Most work on the sugarcane industry has been divided into value chain optimization, harvest scheduling and transportation (Lamsal, 2014; Sethanan and Neungmatcha, 2016). However, most work on the sugarcane industry relating to sugarcane mechanical harvesters has been done in the last 10 years on mechanical harvester scheduling. The route planning of a sugarcane mechanical harvester in the sugarcane supply system has been studied to manage the supply system for the mill to ensure continuous feed and low operational costs, such as in the research study of Jiao et al. (2005), Higgins (2006), Díaz and Pérez (2002), Le Gal et al. (2009) and Sethanan and Neungmatcha (2016). Yet, to the best of our knowledge, none of the literature addresses the integration of assignment and scheduling of sugarcane mechanical harvesters to the sugarcane fields, under the limitations of the mechanical harvester availability and the CCS value of sugarcane fields.

Harvesting scheduling is an interesting topic in the optimization literature. Various research studies have been done to solve the problem to meet the objective. Recently, Kusumastuti et al. (2016) published a review of harvest planning. Unfortunately, in this work, they could not find a model which involves routes, processing times, clusters and time-windows simultaneously. In the area of mechanical harvester scheduling/routing (HRP), various research studies have been done to solve the problem to meet the objectives. Grunow et al. (2007) aim at preserving a constant supply while minimizing the associated costs. The

entire planning problem is structured in a hierarchical fashion: (1) cultivation of the haciendas, (2) harvesting and (3) dispatching the harvesting crews and equipment. Jena and Poggi (2013) determined the routing of cutting machines during harvest periods of sugarcane using mixed integer programming techniques.

Recently, Sethanan and Neungmatcha (2016) studied the increasing sugarcane harvesting efficiency of the path planning of the mechanical harvester, involving direction and field accessibility constraints, with two objective functions being the minimization of harvested distance and maximization of sugarcane yield, which are conflicting and must be considered simultaneously. The variant of the particle swarm optimization combining gbest, lbest and nbest social structures (MO-GLNPSO) was developed to solve sugarcane mechanical harvester route planning (MHRP) to make efficient choices with multi-objectives. Later, Cerdeira-Pena et al. (2017) studied a variant of the traveling salesman problem with additional constraints of clusters, time windows and processing times in cities by applying this model to a real-world problem related to an agricultural cooperative that needs to optimize the routes of several mechanical harvesters. Two different heuristic algorithms based on tabu search and the simulated annealing philosophy were developed. He et al. (2018) studied the operational model to determine the optimal combine mechanical harvester scheduling for fragmental farmlands, with the objective of minimizing the wheat harvesting period. The minimal difference in harvesting time among combine-mechanical harvesters is considered as a constraint. The hybrid algorithm consisting of the tabu search (TS) and operators provided by a genetic algorithm is proposed to identify the optimal schedule that achieves the objectives of minimizing the harvesting period and minimizing the differences of harvesting time among combine mechanical harvesters.

The HARPTW problem is an NP-hard problem (non-deterministic polynomial-time hardness) for which there is no known polynomial algorithm, so there is difficulty in finding the optimal solution. Thus, the heuristic and/or evolutionary algorithms are potential alternatives to be applied to guide the search more effectively. Recently, the ALNS method is becoming very popular because of its simplicity of implementation, as well as its ability to find a good solution. It has been applied in various research areas such as the vehicle routing problems (see Bruglieri et al., 2015; Chen et al., 2018; Ribeiro and Laporte, 2012; Wen et al., 2016; Thevenin and Zufferey, 2014; François et al., 2016; Li et al., 2016; Dayarian et al., 2015; Hintsch and Irnich, 2018; Demir et al., 2012), scheduling problems (see Liu et al., 2017; Rifai et al., 2016; Thevenin and Zufferey, 2018; Lusby et al., 2016), Location-routing problems (see Koç, 2016), Generalized Assignment Problems (see Yagiura et al., 2004), and Traveling Salesman Problems (see Smith and Imeson, 2017).

Even as the performance of the ALNS has increased continuously, the hybridization of the ALNS and other methods has been used to enhance the performance of the pure ALNS. New papers on the hybridization of the ALNS have been published recently. For instance, ALNS is integrated into the variable neighborhood search (LNS) algorithm usually in the shaking phase, such as Stenger et al. (2013) and Li et al. (2015). The neighborhood strategy is chosen using the roulette wheel method based on the success rate of each neighborhood. Recently, Alinaghian and Shokouhi (2018) developed the hybrid adaptive large neighborhood search (ALNS) and the variable neighborhood search (LNS) algorithms to solve the multi-depot multi-compartment vehicle routing problem. The results show the good performance of the proposed hybrid algorithm. Likewise, Sze et al. (2016) proposed an ALNS incorporating LNS as a diversification strategy and applied it to the capacitated vehicle routing problem. A simple and flexible data structure and a neighborhood reduction scheme are embedded. The proposed hybrid algorithm produces very competitive results.

Additionally, ALNS is also integrated with other heuristics. For example, Žulj et al. (2018) developed a metaheuristic hybrid heuristic based on tabu search and ALNS (i.e., ALNS/TS) to solve the order-batching problem. The ALNS/TS shows significant advantages on the larger instances of the existing benchmark sets, and can solve the newly generated large-scale instances. Qu and Bard (2012) developed a greedy randomized adaptive search procedure (GRASP) for pickup and delivery problems with transshipment. In this paper, adaptation of various insertion and removal algorithms was specialized to accommodate transshipments.

Review of past research shows that despite the importance of the integration of assignment and scheduling of mechanical sugarcane harvesters to the sugarcane fields under the limitations of the mechanical harvester availability and the CCS value of sugarcane fields, this aspect of the problem has remained neglected. Therefore, this paper contributes to the literature by developing the mathematical model for the HARPTW for small-scale problems. Considering the NP-Hardness of the problem, an effective ALNS that employs the idea of Differential Evolution (DE) is designed to solve large-scale problems. Over the last decade, DE, first introduced by Storn and Price (1997), has been one of the best evolutionary algorithms, and is used extensively in various fields, since its features make it very attractive for numerical optimization (Dechampai et al., 2017).

3. Problem statement and mathematical model formulation

3.1. Problem statement

As outlined in Fig. 1, in each time period, we have a pre-specified number of mechanical harvesters available and also a number of sugarcane fields to be serviced. Then we schedule the mechanical harvesters to harvest the sugarcane fields by moving them to service the sugarcane fields as scheduled. After servicing the first sugarcane field, a mechanical harvester can move to the next field directly, or go back to the base if it runs out of service time. Each sugarcane field can have time windows due to the maturity of the sugarcane to be harvested or the working condition of the fields.

From Fig. 1, we have 2 sets of mechanical harvesters and 11 sugarcane fields to be harvested. The set of mechanical harvesters is not to be assigned to a sugarcane field if all conditions are not met, such as no sugarcane fields have low CCS value, no mechanical harvester is available as needed, or no predefined constraints are met. The mathematical model of the problem is presented in Section 3.2.

3.2. Mathematical formulation

Based on the characteristics of the HARPTW problem, the mathematical model is constructed. The details of indices, parameters, decision variables, objective function and constraints are as follows.

Indices

i, k	indices for sugarcane field, $i, k = 1\ldots I$
z	index for mechanical harvester, $z = 1\ldots Z$

Parameters

T_i^E	Earliest arrival time of mechanical harvester in field i
T_i^L	Latest arrival time of mechanical harvester in field i
D_{zi}^1	Distance to move mechanical harvester z to field i
D_{ik}^2	Distance between field i and k
R_i	Area of sugarcane field i (rai)
S_z	Harvest speed of mechanical harvester z (minutes per rai)
H_z	Maximum available harvest time of a mechanical harvester z in one day (minutes)
T_z	Traveling rate per kilometer of mechanical harvester z

Decision Variables

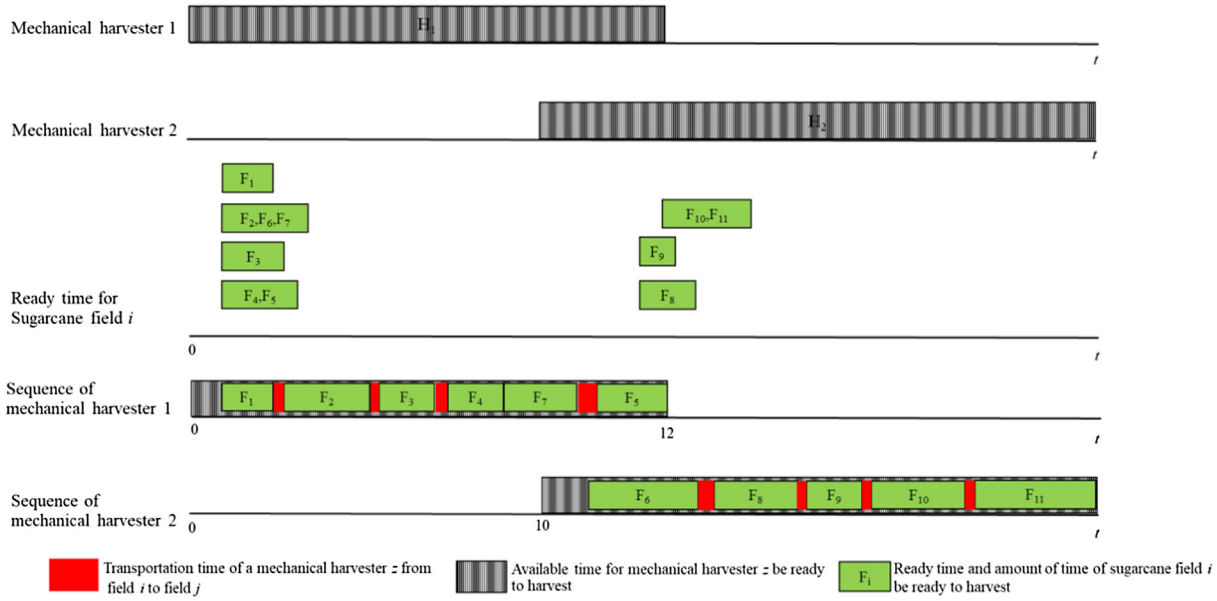

Fig. 1. Illustration of the mechanical harvester sequence representing a solution of the HARPTW problem.

X_{zik}	$\begin{cases}1 & \text{if mechanical harvester } z \text{ is traveling from } i \text{ to } k\\ 0 & \text{otherwise}\end{cases}$
Y_i	$\begin{cases}1 & \text{if field } i \text{ is visited by at least one mechanical harvesterz}\\ 0 & \text{otherwise}\end{cases}$
C_z	Start time of mechanical harvester z
W_{zi}	Start time of mechanical harvester z at field i
A_{zi}	Arrival time of mechanical harvester z at field i
E_{zi}	$\begin{cases}1 & \text{if field } i \text{ is assigned to mechanical harvester set } z\\ 0 & \text{otherwise}\end{cases}$
U_{zk}	Accumulated time used at field k by mechanical harvester z
Q_{zk}	Time used to harvest field k using mechanical harvester z
F	The total areas harvested by mechanical harvester under a harvesting-sharing system

Objective function

$$Maximize F = \sum_{zik} R_i X_{zik} \tag{1}$$

Subject to

$$\sum_{k} D^1_{zk} \times T_z \times X_{z1k} + \sum_{k} D^1_{zk} \times T_z \times X_{zk1} + \sum_{i,k} D^2_{ik} \times T_z \times X_{zik} + \sum_{i} \frac{R_i}{S_z} \times E_{zi} + \sum_{i} W_{zi} + C_z \leq H_z \quad \forall z = 2\cdots Z \text{ and } i \neq k$$

i = 1 is a dummy node that represents the node where z starts travelling (2)

$$A_{zk} \geq A_{zi} + (W_{zi} \times X_{zik}) + (T_z \times D^2_{ik} \times X_{zik}) + E_{zi} \times \frac{R_i}{S_z} + (T^L_i \times E_{zi}) \times (1 - X_{zik}) \quad \forall i = 2\cdots I, k = 2\cdots I \text{ and } i \neq k, \forall z = 1\cdots Z \tag{3}$$

$$A_{z1} = C_z \quad \forall z = 1\cdots Z \tag{4}$$

$$W_{z1} = C_z \quad \forall z = 1. ..Z \tag{5}$$

$$T^E_i \leq A_{zi} \leq T^L_i \quad \forall i = 2. ..I, \forall z = 1. ..Z \tag{6}$$

$$\sum_{z} E_{zi} \leq Y_i \quad \forall i = 2\cdots I \tag{7}$$

$$X_{zik} \leq E_{zi} \quad \forall i = 2\cdots I, k = 2\cdots I \text{ and } i \neq k, \forall z = 1\cdots Z \tag{8}$$

$$\sum_{i} X_{zik} - \sum_{i} X_{zki} = 0 \quad \forall k = 2\cdots I \text{ and } i \neq k, \forall z = 1\cdots Z \tag{9}$$

$$\sum_{z,k} X_{zik} = Y_i \quad \forall i = 2\cdots I \text{ and } i \neq k \tag{10}$$

$$\sum_{z,k} X_{zki} = Y_i \quad \forall i = 2\cdots I \text{ and } i \neq k \tag{11}$$

$$U_{zk} \geq (U_{zi} + Q_{zk} - H_z + (H_z \times (X_{zki} + X_{zik}) - (Q_{zk} + Q_{zi}) \times X_{zki}) \quad \forall i = 2\cdots I, k \geq 1 \text{ and } i \neq k \tag{12}$$

$$U_{zk} \leq H_z - (H_z - Q_{zk}) \times X_{z1k} \quad \forall k = 2\cdots I \text{ and } k \text{ is first stop} \tag{13}$$

$$U_{zk} \geq Q_{zk} + \sum_{i=2}^{I} Q_{zi} \times X_{zik} \quad \forall k = 2\cdots$$

I and k is not the first stop (14)

$$X_{zik} \geq 0 \tag{15}$$

$$Y_i \geq 0 \tag{16}$$

$$C_z \geq 0 \tag{17}$$

$$W_{zi} \geq 0 \tag{18}$$

$$A_{zi} \geq 0 \tag{19}$$

$$E_{zi} \geq 0 \tag{20}$$

$$U_{zk} \geq 0 \tag{21}$$

$$Q_{zk} \geq 0 \tag{22}$$

$$F \geq 0 \tag{23}$$

The objective function (3.1) is to maximize the agricultural area of the sugarcane in units of rai. Constraint 3.2 is used to confirm that in one day the time used to travel from z to all fields, harvest time and start time must not exceed the maximum time available for each mechanical harvester. Constraints 3.3 indicate the arrival time of z at field i while 3.6 is used to verify that the arrival time of mechanical harvester

z at field i must lie within field i time windows. Constraints 3.4 and 3.5 show the arrival and start time at the location that z is in are just its start and start time. Constraints 3.7 and 3.8 indicate that the tour between i and k can occur only when field i is used by mechanical harvester z. Constraints 3.9–3.11 indicate when the tour must enter i and must leave i. Constraint 3.12 is the cumulative amount of time used to harvest field k using mechanical harvester z, which must be higher than the cumulative time used to harvest field i when k is harvested after i on the same route. 3.13 addresses that when k is the first stop, the cumulative harvest time at field k is equal to the time used to harvest field k using mechanical harvester z. Finally, 3.14 shows that the cumulative time used to harvest on the route of mechanical harvester z at field k is equal to the cumulative time used to harvest all of the fields that are on the route of z, plus the time used to harvest field k using mechanical harvester z. 3.15–3.23 are used to guarantee that the decision variables must be greater than or equal to zero.

In the HARPTW problem, Lingo v.11 for Windows was used to find the optimal solution. Although the optimal solution was obtained for the problem, the computational time was excessive. Attempts to solve the medium- or large-size problems were unsuccessful since they required too much CPU time. Therefore, an effective ALNS that employs the idea of DE was developed to obtain a near-optimal solution for realistic problems.

4. The proposed heuristics

4.1. The ALNS and development of 8 newly invented destroy methods and 4 repair methods

When the size of the problem becomes too large and too complicated to be solved by exact solution methods (i.e. mathematical model), in this study, Adaptive Large Neighborhood Search (ALNS) is applied. The general procedure of ALNS is presented in Fig. 2. From this figure, we can describe the operation as follows.

Step1: Generate an initial solution. In this study, the initial solution is randomly constructed.
Step2: The destroy method is randomly selected according to the performance of the proposed destroy method. An effective destroy method has a higher probability of being selected. Then go to Step 3.
Step3: The repair method is selected to return the feasible solution to the algorithm.
Step4: For acceptance of the new solution, the better solution is accepted but a worse solution is accepted with a certain probability.

Then redo steps (2) to (4) until the stopping criteria are fulfilled.

To improve the algorithm, in this study, 8 newly invented destroy methods and 4 repair methods are presented, while 2 more acceptance methods are presented beside the exponential probability that is frequently used in ALNS research. The destroy methods have been designed to remove the entities from the current solution based on the possibility to reduce the total cost of the problem. The techniques that are to be used to select the entities can be randomly selected, removing the most costly entity or removing the cheapest entities. The procedures of the 8 destroy methods are detailed as follows:

Destroy 1: FirstCS/BestCS/O1/In routine
Step 1: Calculate the amount of resources used (RS) by all servers.
Step 2: Order the servers according to their RS in increasing order and call it O1.
Step 3: Calculate the slack in resources used (S-RS) of all other servers. S-RS is calculated from the difference of RS of the current assignment when exchanging the clients assigned to the server which is in the first rank (FirstCS) and the target server.
Step 4: Remove clients FirstCS and the Clients that have the maximum S-RS (BestCS).

Destroy 2: RandO1/In Routine
Step 1: Randomly order the servers and call it O1.
Step 2: Calculate the slack in resources used (S-RS) of all other servers. S-RS is calculated from the difference of RS of the current assignment when exchanging the clients assigned to the server which is in the first rank (RandFirstCS) and the target server.
Step 3: Remove clients RandFirstCS and the Clients that have the maximum S-RS (BestCS).

Destroy 3: FirstCS/BestCS/O1/De routine
Step 1: Calculate the amount of resources used (RS) by all servers.
Step 2: Order the servers according to their RS in decreasing order and call it O1.
Step 3: Calculate the slack in resources used (S-RS) of all other servers. S-RS is calculated from the difference of RS of the current

Pseudocode for maximization of the harvest area under a sharing mechanical harvester system
Input : V_t
For t=1 : T
Randomly generate an initial solution V_t update $V_t^* = V_t$ and $f(V_t^*) = f(V_t)$
If number of iterations/solution quality less than predefined number
then do
Generate candidate solution V_t' by
Randomly select the destroy method
- Select Destroy Method (select the destroy method using probability)
- Perform the selected destroy method.

Randomly select the repair method
- Select Repair Method
- Perform the selected repair method.

Update V_t by

$$V_t = \begin{cases} V_t' \ \ if\ f(V_t') \geq f(V_t) \\ V_t' \ or\ V_t\ with\ \ prob\ 1-5\ if\ \ f(V_t') \leq f(V_t) \end{cases}$$

Update $f(V_t^*)\ \ if\ f(V_t') \geq f(V_t)$
Else end the algorithm
Output : V_t

Fig. 2. General Procedure of ALNS.

assignment when exchanging the clients assigned to the server that is in the first rank (FirstCS) and the target server.
Step 4: Remove clients FirstCS and the Clients that have the maximum S-RS (BestCS).

Destroy 4: RandO1/De routine
Step 1: Randomly order the servers and call it O1.
Step 2: Calculate the slack in resources used (S-RS) of all other servers. S-RS is calculated from the difference of RS of the current assignment when exchanging the clients assigned to the server which is in the first rank (RandFirstCS) and the target server.
Step3: Remove clients RandFirstCS and the Clients that have the minimum S-RS (BestCS).

Destroy 5: PCS/O2/1 routine
Step1: Find O2 which is the order of the clients which are ranked using their resources used in decreasing order. Please note that only the assigned clients are sorted.
Step2: Find the server that services the clients that are in the first rank of order O2 and call it BestCS.
Step3: Find the position in the route of BestCS of the first rank in order O2 and call it PCS.

Destroy 6: PCS/O2/2 routine
Step 1: Find O2, which is the order of the clients that are ranked using their resources used in decreasing order. Please note that only the assigned clients are sorted.
Step2: Find the servers that service the clients which are in the first and the second rank of order O2 and call them BestCS1 and BestCS2.
Step 3: Find the positions in the route of BestCS1 and BestCS2 of the first rank in order O2 and call them PCS1 and PCS2.

Destroy 7: ATT/1 routine
Step 1: Randomly select one client out of I clients. The probability of the client to be selected is calculated from its attractiveness score (AT). The selected client is called TaBuList.
Step 2: Find the server that services the selected client and call it BestCS.
Step 3: Find the position in the route of BestCS and call it PCS.

Destroy 8: ATT/2 routine
Step 1: Randomly select two clients out of I clients. The probability of a client to be selected is calculated from its attractiveness score (AT). The selected client is called TaBuList1 and TaBuList2
Step 2: Find the servers that service the selected clients and call them BestCS1 and BestCS2.
Step 3: Find the positions in the route of BestCS1 and BestCS2 and call them PCS1 and PCS2.

After the entities are removed from the solution, an effective method which forms a feasible solution after the destroy method has been used is needed. Four repair methods have been designed based on the idea of various local search techniques. These repair methods are: **ExRoute, PartialReRoute, ReRouteTabu, and FixedReRoute**. To retain the searching capacity, the scores of all destroy and repair methods have been updated. We present the two newly invented formulas to accept the worse solution to be the current solution to destroy and repair in the next iteration. These two formulas are designed based on the idea to increase the diversification behavior of the proposed heuristics. These formulas include the parabola and linear acceptance formulas so that there is more variety to accept the worse solutions. The procedures of the repair methods are presented as follows:

Repair 1: ExRoute
Step 1: Exchange all clients that are in FirstCS and BestCS (if possible according to all constraints).
Step 2: Re-route unassigned clients into FirstCS and BestCS (if possible).

Repair 2: PartialReRoute
Step 1: Sort the servers in decreasing order according to their resources used and call it order O3.
Step 2: Decide the number of clients that will be re-routed by Eq. (24).

$$M = \text{Ceil}[0.5 \times \text{N} \times \exp^{\frac{(-0.01 \times MaxIt)}{MaxIt - It}}] \quad (24)$$

where M is the number of servers that will be re-routed, MaxIt = maximum iterations that will be executed, I is the current iteration and N is the number of mechanical harvesters.

Step 3: Reroute all un-assigned clients and the clients that are in the last M servers in order O3.

Repair 3: ReRouteTabu
Step 1: Rank all clients that are not in the TaBuList/TaBuList1/TaBuList2 in decreasing order according to the resources used and call it ListA.
Step 2: Put TaBuList/TaBuList1/TaBuList2 after the last field in ListA and call it ListB.
Step 3: Re-Route List B.

Repair 4: FixedReRoute
Step 1: Fix the route of clients that have position before position PCS/PCS1/PCS2 of BestCS/BestCS1/BestCS2.
Step 2: Re-reroute all clients that are not in BestCS/BestCS1/BestCS2.

4.2. Demonstration of the ALNS application

To demonstrate the application of the ALNS, three mechanical harvesters which have 10 working hours per day, and 10 sugarcane fields are given in order to simplify the problem. If the server represents the mechanical harvester, the clients represent the sugarcane fields, and the resources used are replaced by Field size in order to make the generic framework work with the proposed problem. Details of the proposed algorithm using 8 destroy and 4 repair methods are as follows.

Step 1: The initial solution is randomly constructed 10 times, and the best solution among all solutions is selected as the starting solution (V, V^*, z(V) and $z(V^*)$) when x is the current solution, and z (V) is the objective function of the current solution. V^* and $z(V^*)$ are the best solution and the best objective function.

From Fig. 3, there are 10 sugarcane fields which have areas of 80, 120, 140, 89, 90, 160, 140, 130, 180 and 170 rai (1 rai = 0.16 Ha), and there are three mechanical harvesters which have a pre-specified maximum working time which is 10 h per day. When constructing the solution, the time windows of each field need to be considered. The traveling time from the current position of the mechanical harvester to the first field, the harvest time, the traveling time between fields which are on the same route, and the traveling time from the last field to the base station of the mechanical harvester are included in the time used by the mechanical harvester, and this amount of time must not exceed 600 min. Fig. 3 shows that the best solution is V2 ($V = V^* = V2$) mechanical harvester No. 2 due to it has the maximum number of rai that can be cut in one day in which mechanical harvester 1 cuts sugarcane F2, F3 and F10, mechanical harvester 2 cuts F4 and F6 while mechanical harvester 3 cuts fields 1 and 9. This harvest plan generates a 939 rai harvest area.

Step 2: In this step, we will destroy the current solution by using one out of 8 destroy routines.
Step 3: After the destroy method has been used, the repair method is needed to return the feasibility of the solutions. If the server represents the mechanical harvester, the clients represent the sugarcane fields and the resources used are replaced by field size in

Solution 1			Solution 2			Solution 3		
Harvester 1	Harvester 2	Harvester 3	Harvester 1	Harvester 2	Harvester 3	Harvester 1	Harvester 2	Harvester 3
F1	F2	F8	F3	F4	F1	F8	F4	F5
F6	F5	F9	F2	F6	F9	F9	F1	F7
		F10	F10				F3	
760 rai			939 rai			849 rai		

V=V2

Solution		
Harvester 1	Harvester 2	Harvester 3
F3	F4	F1
F2	F6	F9
F10		
939 rai		

Note: $F_1, F_2, \ldots, F_9$ = Sugarcane Fields, **1 rai = 0.16 Hectare**, V = Solution

Fig. 3. Examples of 3 randomly constructed solutions (V1, V2, and V3).

order to make the generic framework work with the proposed problem. Details of using 8 destroy and 4 repair methods are as follows.

Destroy 1: FirstCS/BestCS/O1/In

Step 1: Calculate the amount of area (RS) that the mechanical harvester can harvest.
Step 2: Order the mechanical harvester according to their RS by increasing order and call it O1.
Step 3: Calculate the slack of area that can be harvested by the mechanical harvester (S-RS). S-RS is calculated from the difference of RS of the current assignment when exchanging the field assigned to the mechanical harvester that is in the first rank (FirstCS) and the target mechanical harvester.
Step 4: Remove field FirstCS and the field that has the maximum S-RS (BestCS).

Destroy 2: RandO1/In

Step 1: Randomly order the servers and call it O1.
Step 2: Calculate the slack of area that can be harvested by the mechanical harvester (S-RS). S-RS is calculated from the difference of RS of the current assignment and when exchanging the field assigned to the mechanical harvester that is in the first rank (FirstCS) and the target mechanical harvester.
Step 3: Remove field FirstCS and the field that has the maximum S-RS (BestCS).

Destroy 3: FirstCS/BestCS/O1/De

Step 1: Calculate the amount of area (RS) that the mechanical harvester can harvest.
Step 2: Order the mechanical harvesters according to their RS in decreasing order and call it O1.
Step 3: Calculate the slack of area that can be harvested by the mechanical harvester (S-RS). S-RS is calculated from the difference of RS of the current assignment when exchanging the field assigned to the mechanical harvester that is in the first rank (FirstCS) and the target mechanical harvester.
Step 4: Remove field FirstCS and the field that has the maximum S-RS (BestCS).

Destroy 4: RandO1/De

Step 1: Randomly order the servers and call it O1.
Step 2: Calculate the slack area that can be harvested by the mechanical harvester (S-RS). S-RS is calculated from the difference of RS of the current assignment when exchanging the field assigned to the mechanical harvester that is in the first rank (FirstCS) and the target mechanical harvester.
Step 3: Remove field FirstCS and the field that has the minimum S-RS (BestCS).

Destroy 5: PCS/O2/1

Step 1: Find O2, which is the order of the fields that are ranked from their available area in decreasing order. Please note that only the assigned fields are sorted.
Step 2: Find the mechanical harvester that harvests the fields that are in the first rank of order O2 and call it BestCS.
Step 3: Find the position in the route of BestCS of the first rank in order O2 and call it PCS.

Destroy 6: PCS/O2/2

Step 1: Find O2, which is the order of the fields which are ranked by their available area in decreasing order. Please note that only the assigned fields are sorted.
Step 2: Find the mechanical harvester that harvests the fields that are in the first and second rank of order O2 and call them BestCS1 and BestCS2.
Step 3: Find the position in the route of BestCS1 and BestCS2 of the first rank in order O2 and call them PCS1 and PCS2.

Destroy 7: ATT/1

Step 1: Randomly select one field out of I fields. The idea is that a different field should have a different attractiveness to be selected at every iteration. The attractiveness score will be added to the current attractiveness score if the field is the best solution. Otherwise, the attractiveness score will remain the same.

For an example in an arbitrary iteration, there are 10 fields and the current attractiveness score (AS) is as shown in Table 1. In the current iterations, the best solution is the vector shown in Fig. 3. Thus, fields F1, F2, F3, F4, F6, F9, and F10 will increase by 2 scores. The result of the attractiveness score of the current iteration is shown in Table 1.

Then we apply the popular Roulette wheel selection process to select one of these fields to be the position that we will start to re-assign the route. The selected field is called TaBuList.

Step 2: Find the mechanical harvester that services the selected field

Table 1
Added score example of the destroy 7 routines.

Field	AS (t)	Added score	AS(t + 1)	Field	AS (t)	Added score	AS(t + 1)
F1	50	2	52	F6	96	2	98
F2	90	2	92	F7	60	0	60
F3	40	2	42	F8	46	0	46
F4	80	2	82	F9	81	2	83
F5	85	0	85	F10	81	2	83

and call it BestCS1 and BestCS2. If F6 is selected then BestCS will be mechanical harvester 2 (Fig. 1).
Step 3: Find the position in the route of BestCS and call it PCS. If F6 is selected then PCS will be 2.

Destroy 8: ATT/2

Step 1: Randomly select two fields (called AF1, AF2) out of I fields. The idea is that different fields should have a different attractiveness to be selected. At every iteration, the attractiveness score will be added into the current attractiveness score if the field is in the best solution. Otherwise, the attractiveness score will remain the same.

For an example in an arbitrary iteration, there are 10 fields and the current attractiveness score (AS) is as shown in Table 2. If in the current iteration the best solution is the vector shown in Fig. 3, fields F1, F2, F3, F4, F6, F9, and F10 will increase by 2 scores. The result of the attractiveness score of the current iteration is shown in Table 2.

Then we apply the popular Roulette wheel selection process to select one of these fields to be the position that we will start to re-assign the route. The selected field is called TaBuList1 and TaBuList2.

Step 2: Find the mechanical harvester that services the selected field and call it BestCS1 and BestCS2. If F6 and F9 are selected then BestCS1 and BestCS2 will be mechanical harvesters 2 and 3 (Fig. 1) consequently.
Step 3: Find the position in the route of BestCS1 and BestCS2 and call it PCS1 and PCS1
Then PCS1 and PCS2 will be 2 and 2 consequently.

Repair 1: ExRoute.

Step 1: Exchange all fields that are in FirstCS and BestCS (if possible according to the time windows and time constraints).
Step 2: Re-route unassigned fields into FirstCS and BestCS (if possible).

Repair 2: PartialReRoute

Step1: Sort mechanical harvesters in decreasing order according to total rai that is cut by a particular mechanical harvester, and call it order O3.
Step2: Decide the amount of mechanical harvester that will be re-routed by Eq. (24).
This equation is based on a reduced number of re-routing mechanical harvesters when the iteration is higher, so that at the beginning of the execution of the algorithm it can search a wider space.
Step 3: Reroute all un-assigned fields and the fields that are in the last M mechanical harvesters in order O3.

Table 2
Score adding example of destroy 8 routines.

Field	AS (t)	Added score	AS(t + 1)	Field	AS (t)	Added score	AS(t + 1)
F1	50	2	52	F6	96	2	98
F2	90	2	92	F7	60	0	60
F3	40	2	42	F8	46	0	46
F4	80	2	82	F9	81	2	83
F5	85	0	85	F10	81	2	83

Repair 3: ReRouteTabu

Step 1: Rank all fields that are not in the TaBuList, TaBuList1, and TaBuList2 in decreasing order according to the size of the field (rai) and call it ListA.
Step 2: Put TaBuList, TaBuList1, and TaBuList2 after the last field in ListA and call it ListB.
Step 3: Re-Route List B.

Repair4: FixedReRoute

Step 1: Fixed field before PCS, PCS1, PCS2 of BestCS, BestCS1, BestCS2 consequently.
Step 2: Re-reroute all fields that are not in BestCS, BestCS1 or BestCS2.

Please note that if Destroy methods 1–4 are selected, only Repair methods 1 and 2 can be applied and when Destroy methods 5–8 are selected only repair methods 3 and 4 can be used. We can conclude the matching of using Destroy and Repair methods as in Table 3.

The Destroy and Repair methods are randomly selected using their attractiveness. The attractiveness is updated during the simulation execution. The procedure of the updating is explained as follows.

The attractiveness of Destroy and Repair Methods (AT)

The probability to select one out of the Destroy and Repair methods is calculated from the current score of each Destroy and Repair method. The selected Destroy and Repair methods in a particular iteration are allowed to add a new score into their current scores. The new score is assigned using the following rules.

(1) Add 10 scores when the new best solution is found in that iteration.
(2) Add 8 scores when the solution found in that iteration is better than the accepted solution in the previous iteration.
(3) Add 6 scores when the solution found in that iteration is worse than that of the accepted solution in the previous iteration, but it is accepted as the next current solution using Formulas 1–5.
(4) Add 4 scores when the solution found is not accepted by Formulas 1–5.

Then the current score is updated and Roulette wheel selection is used to pick one of the Destroy and Repair methods. For example, if the current score of the Destroy methods 1–8 is 221, 432, 123, 451, 324, 225, 254 and 310, the total score for all methods is 2140. We use the total score to find the probability of selecting each Destroy method by use (score of each Destroy method)/(total score) and the result is 0.09, 0.18, 0.05, 0.19, 0.14, 0.10, 0.11 and 0.13. Then find the cumulative probabilities of all methods which are 0.09, 0.28, 0.33, 0.52, 0.66, 0.76, 0.87 and 1.00. Finally, a random number is generated. For example, if the random number is 0.56, then Destroy method 5 will be selected. This procedure can be applied to the Repair method as well.

Step 4: The acceptance of a worse solution

If the newly generated solution is better than that of the current solution, it is automatically accepted to be the next current solution. A worse solution will sometimes be accepted according to some probability function. In this study, 5 acceptance functions are applicable to choose for use which are Formulas 1 to 5.

Formula 1:

$$p = \exp^{-\frac{(Z(V')-Z(V))}{T\times K}} \quad (25)$$

Table 3
Matching of destroy and repair methods.

List of destroy methods	Available repair methods	List of destroy methods	Available repair methods
FirstCS/BestCS/O1/In	ExRoute PartialReRoute	PCS/O2/1	ReRouteTabu FixedReRoute
RandO1/In	ExRoute PartialReRoute	PCS/O2/2	ReRouteTabu FixedReRoute
FirstCS/BestCS/O1/De	ExRoute PartialReRoute	ATT/1	ReRouteTabu FixedReRoute
RandO1/De	ExRoute PartialReRoute	ATT/2	ReRouteTabu FixedReRoute

in which p is the probability to accept the worse solution when V′ is the current solution and Z(V′) is the objective function of V′. V is the currently used solution and Z(V) is the objective function of V. T is a predefined temperature, as in the Simulated Annealing algorithm, and K is the predefined parameter.

Formula 2:

$$p = 1 - \exp^{-\left[\left(\frac{Z(V)-Z(V')}{Z(V)}\right)^2 + (It - \frac{MaxIt}{2})^2\right]} \quad (26)$$

where *It* is the current iteration and *MaxIt* is the predefined maximum number of iterations.

Formula 3:

$$p = 1 - exp^{-\left[(It-\frac{MaxIt}{2})^2\right]} \quad (27)$$

Formula 4:

$$p = 1 - \frac{It}{MaxIt} \quad (28)$$

Formula 5:

$$p = 1 - \frac{[\frac{Z(V)-Z(V')}{Z(V)} + \frac{It}{MaxIt}]}{2} \quad (29)$$

Formula 1 is a simple formula used in the Simulated Annealing algorithm, while Formulas 2 and 3 are adapted from the parabola function, where the very first and last iterations have the highest probability to accept the worse solution. Formula 2 takes account of the solution quality (Z(V′)) to be the attractiveness to be selected as well, while Formula 3 only uses the current iteration as the attractiveness. Formulas 4 and 5 are more or less the same as Formulas 2 and 3, but use linear functions instead of an exponential function to calculate the probability. In this research, Formulas 1 to 5 are selected to be used in the proposed algorithm and the performance compared.

5. Computational framework and results

5.1. Computational framework

The case study mill uses approximately 15,000 tons of sugarcane per day, with 3000 contract growers that are separated into 10 sub-regions. Currently, the sugar mill owns 80 mechanical harvesters in total. Each machine can harvest 15–25 tons per day depending on its performance, size, and capacity.

In order to test the model, the Adaptive Large Neighborhood Search (ALNS) employed the idea of DE in the mechanical harvester assignment and routing problem with time windows (HARPTW) to maximize the total area serviced by the harvesting-sharing system. It was also validated by comparing the solutions with the optimal solution obtained by Lingo v.11 for Windows software and the proposed algorithms. The performance of the proposed methods was tested using 3 groups of problem instances and one real case study. Details of generated data are shown in Table 4.

Table 4 shows that 21 problem instances were used to test the algorithm in total, which includes the real case study. The research presents 4 new acceptance of worse solution criteria and 1 popular formula, thus 5 sub-algorithms are presented as ALNS-1 (original), ALNS-2, ALNS-3, ALNS-4, and ALNS-5 which use Formulas 1, 2, 3, 4 and 5 respectively. The algorithms have been coded in C++ and run on a computer notebook with Intel® CoreTM i5-2410 M 2.3 GHz and 4 GB memory. The stopping criteria have been set according to the problem size. For small instances, the time after which the optimal solution was found is presented, while in the medium, large and case study problems, computational time was used, which was set to be 10, 20, and 60 min respectively. The experiment has been executed 5 times and the best solution is recorded. Due to the optimal solution not being found within the acceptable computational time for the medium and large (including the case study) sized problems, the time limit of the run time in Lingo v.11 for Windows was set to be 240 h for the medium sized instances and 480 h for the large-sized instances (including the case study). The best and bound of the solutions generated are reported and compared with those that are generated by the proposed heuristics.

Table 4
Details of data generated for every group of problem instances.

Group of data	Data name	No. of fields	Size of fields (rai)	No. of Harvesters	Total area (rai)
Small	S-1	8	5–100	3	150–600
	S-2	10		4	
	S-3	10		4	
	S-4	12		3	
	S-5	10		4	
Medium	M-1	15	10–150	7	150–600
	M-2	15		8	
	M-3	20		8	
	M-4	20		7	
	M-5	25		9	
	M-6	25		10	
	M-7	25		10	
	M-8	30		12	
Large	L-1	50	10–150	20	400–1500
	L-2	55		20	
	L-3	60		22	
	L-4	60		23	
	L-5	65		25	
	L-6	65		25	
	L-7	70		25	
Case study	C-1	321	5–200	80	120,000

Remark: 1 rai = 0.16 Ha.

5.2. Computational results

Computational results are presented in terms of solution quality and computational speed. Table 5 shows the optimal solutions obtained from the mathematical model, and the best values (i.e., harvested areas and the amount of CPU time required to execute the proposed algorithms) of the proposed algorithms for small-size problems. For the medium-sized instances, computational results are presented in Table 6 and the results of statistical tests for all methods are shown in Table 7. Tables 8 and 9 present the best values of the proposed algorithms and statistics results among all methods for large-size problems, respectively.

From Table 5, we can see that all proposed algorithms can find the optimal solution in a shorter computational time. We use the stopping

Table 5
Result of small instances.

Test problem	Optimal	Time	ALNS-1		ALNS-2		ALNS-3		ALNS-4		ALNS-5	
			Area	Time	Area	Time	Area	Time	Area	Time	Area	Time
S-1	314	379.8	314	2.1	314	2.1	314	2.2	314	2.5	314	2.5
S-2	363	328.8	363	2.5	363	2.3	363	2.2	363	2.2	363	2.2
S-3	291	4620.0	291	2.3	291	1.9	291	2	291	2.4	291	2.4
S-4	385	8580.0	385	2.4	385	1.8	385	2.9	385	2.3	385	2.3
S-5	258	612.6	258	2.6	258	1.9	258	2.05	258	2.1	258	2.1
c.time(avg.)		2904.24	2.38		2.0		2.27		2.3		2.3	

Remark: c.time(avg.) = computational average for all instances, time = computational time used to find the optimal solution (second), area = area harvested in 1 day (rai), 1 rai = 0.16 Ha.

criteria as the time that the proposed algorithms can find the optimal solution recorded in this Table. We found that ALNS-2 which uses the acceptance of best solution Formula 2 is the one that has the lowest computational time when compared with all other heuristics. The ALNS-2 uses 1452.12 times (2904.24/2 = 1452.12) less computational time than that of Lingo v.11 for Windows, while generating the same solution. From Table 6, the result is similar to the small size test instances. ALNS-2 can find 100% best solutions among all proposed heuristics. Comparing with the Best solution generated by Lingo v.11 for Windows, ALNS-2 generates better solutions while using 1440 times ((240 × 60)/10) less time than Lingo v.11 for Windows. Comparing the proposed heuristics with the Bound, the proposed heuristics are 0.25–4.97% away from the Bound generated by Lingo v.11 for Windows while using 14,400 min, which is relatively low. This means that we can conclude that our proposed heuristic is effective for medium size test instances. The statistical test was performed using the paired *t*-test at a confident interval of 95%. The result is shown in Table 7. In this table, the signs ≥, = and ≤ indicate that the solution of an algorithm is significantly greater, equal or less than that of the compared algorithm. Numbers in parentheses are the p-value of that comparison. The results in Table 7 show that "Best" can generate worse solutions than that of all proposed heuristics, while the "Bound" is significantly higher than that of all others. It is noted that the "Bound" could contain infeasible solutions. That is why it has better solutions than the others. Comparing only the proposed heuristics, ALNS-2 is significantly the best heuristic out of all proposed heuristics. From the statistical result, the proposed heuristics that use Formula 1 (ALNS-1) are not different from Formula 5 (ALNS-5), and Formula 3 (ALNS-3) is not different from Formula 4 (ALNS-4). This means that the acceptance formula using an exponential function (Formula 1) is not different from the linear function (Formula 5). The parabola function that did not use the quality of the current solution to determine the probability of accepting a worse solution can reduce the quality of the proposed heuristic to be as good as a function that takes care of the quality of the current solution.

For the experiment of the large size test instances, the results are shown in Table 8. From this table, the ALNS-2, which is the adaptive large-scale neighborhood search that uses Formula 1 as the best case acceptance, can find the best solution among all other proposed heuristics in about 3 out of 4 test instances. ALNS-2 can find 0.17–3.90% from the Bound generated by Lingo v.11 for Windows using 1440 times ((480 × 60)/20) less computational time. The paired *t-test* is used to check if the proposed heuristic is statistically different than that of the "Best" and the "Bound". The results are shown in Table 9. This Table reveals that Formula 1 (ALNS-1) is not significantly different from Formula 5 (ALNS-5). In the last test instances (i.e., S-5), Formula 1 is also not different from Formula 4 (ALNS-4). This means that the exponential formula does not perform differently from the linear function, but the parabola function Formulas 2 (ALNS-2) and 3 (ALNS-3) generate a significantly better solution than all of the other proposed heuristics.

To investigate the performance of the proposed algorithms, the percentage differences (PD) of the solutions obtained by the algorithms with respect to those of the mathematical model (i.e. optimal solutions or "Bound" which is the bound obtained within the predefined time limit) were determined by Eq. (30). The percentage difference obtained for all test instances is shown in Table 10. From this table, we can see that using the acceptance worst case, Formula 1 generates the lowest difference from the best solution, which is only 5.402%, while the other acceptance formulas generate solutions which lay between 1.782 and 5.346% from the best solution:

Table 7
Results of statistical test for all methods.

Method	Bound	ALNS-1	ALNS-2	ALNS-3	ALNS-4	ALNS-5
Best	≤(0.00)	≤(0.035)	≤(0.000)	≤(0.022)	≤(0.004)	≤(0.028)
Bound		≥(0.000)	≥(0.000)	≥(0.000)	≥(0.000)	≥(0.000)
ALNS-1			≤(0.001)	≤(0.026)	≤(0.001)	=(0.156)
ALNS-2				≥(0.020)	≥(0.035)	≥(0.001)
ALNS-3					=(0.631)	≤(0.024)
ALNS-4						≤(0.003)

Table 6
Computational results of medium size of test instances.

Test problem	No. of fields	No. of harvesters	Total harvested area in predefined time (rai)						
			Best	Bound	ALNS-1	ALNS-2	ALNS-3	ALNS-4	ALNS-5
M-1	15	7	450	497	464	484	469	475	464
M-2	15	8	451	528	489	515	505	505	496
M-3	20	8	574	684	640	650	653	648	643
M-4	20	7	490	548	511	534	506	529	511
M-5	25	9	717	792	755	788	755	759	756
M-6	25	10	740	796	769	790	780	772	770
M-7	25	10	748	791	776	789	784	784	774
M-8	30	12	820	892	862	881	876	870	864

Remark: Best = best solution generated by Lingo v.11 on Windows within 240 h, Bound = solution bound reported by Lingo v.11 for Windows within 240 h, 1 rai = 0.16 Ha.

Table 8
Computational result of large-sized instances and the case study.

Test problem	No. of fields	No. of harvesters	Total harvested area in predefined time (rai)						
			Best	Bound	ALNS-1	ALNS-2	ALNS-3	ALNS-4	ALNS-5
L-1	50	20	1850	2015	1972	1985	1950	1953	1972
L-2	55	20	1890	2140	2104	2116	2130	2105	2050
L-3	60	22	1882	2298	2185	2294	2267	2190	2176
L-4	60	23	1974	2310	2035	2220	2113	2057	2048
L-5	65	25	2011	2424	2231	2389	2289	2287	2269
L-6	65	25	2032	2421	2242	2391	2302	2251	2239
L-7	70	25	2208	2540	2349	2459	2381	2350	2340
Case study	321	113	12,015	13,420	13,220	13,338	13,110	13,210	13,002

Remark: Best = best solution generated by Lingo v.11 for Windows within 480 h, Bound = solution bound reported by Lingo v.11 within 480 h, 1 rai = 0.16 Ha.

Table 9
Statistical test for the large-size test instances.

Method	Bound	ALNS-1	ALNS-2	ALNS-3	ALNS-4	ALNS-5
Best	≤(0.00)	≤(0.050)	≤(0.017)	≤(0.021)	≤(0.043)	≤(0.046)
Bound		≥(0.000)	≥(0.000)	≥(0.000)	≥(0.000)	≥(0.000)
ALNS-1			≤(0.002)	≤(0.017)	=(0.217)	=(0.755)
ALNS-2				≥(0.002)	≥(0.004)	≥(0.012)
ALNS-3					≥(0.012)	≥(0.004)
ALNS-4						≤(0.021)

Table 10
Percent difference of the solution generated by the proposed algorithms and the bound solution (optimal solution or the bound solution found by Lingo v.11 for Windows).

Instance	Bound solution	% Difference from the bound solution				
		ALNS-1	ALNS-2	ALNS-3	ALNS-4	ALNS-5
M-1	497	6.640	2.616	5.634	4.427	6.640
M-2	528	7.386	2.462	4.356	4.356	6.061
M-3	684	6.433	4.971	4.532	5.263	5.994
M-4	548	6.752	2.555	7.664	3.467	6.752
M-5	792	4.672	0.505	4.672	4.167	4.545
M-6	796	3.392	0.754	2.010	3.015	3.266
M-7	791	1.896	0.253	0.885	0.885	2.149
M-8	892	3.363	1.233	1.794	2.466	3.139
L-1	2015	2.134	1.489	3.226	3.077	2.134
L-2	2140	1.682	1.121	0.467	1.636	4.206
L-3	2298	4.917	0.174	1.349	4.700	5.309
L-4	2310	11.905	3.896	8.528	10.952	11.342
L-5	2424	7.962	1.444	5.569	5.652	6.394
L-6	2421	7.394	1.239	4.915	7.022	7.518
L-7	2540	7.520	3.189	6.260	7.480	7.874
Case study	13,420	1.490	0.611	2.310	1.565	3.115
% Average		**5.346**	**1.782**	**4.011**	**4.383**	**5.402**

$$PD = \frac{|Bound - Algo|}{Bound} \times 100 \quad (30)$$

where

Bound = the solution bound generated from Lingo v.11 for Windows which can be both the optimal solution or the bound solution within the predefined time limit,
Algo = the solution generated by the proposed heuristics.

Additionally, we performed experiments on the adaptation of the best solution of the proposed heuristics using Formulas 1–5. The simulation used 1185 iterations. The best solutions generated from each formula were then collected, and the simulation results are shown in Fig. 4. This Figure shows that the acceptance of worse criteria using Formula 2 slowly changes the simulation result, and it changes the solution throughout the simulation run until it obtains the best solution. Formula 3 performs well in the first half of the simulation runs. Then it does not change the solution in the last part of the simulation run, which can make the solution worse than that of Formula 2. These two formulas are the best formulas to accept the worse solution due to giving a higher chance to accept the worse solution in the first part of the simulation to explore a greater search area. The chance of acceptance of the worse solution decreases until the lowest chance is in the middle of the simulation run, and starts to increase the chance of acceptance of the worse solution again after the first half. The second increase we decided to allow was for the algorithm to escape from a local optimum, by accepting the chance of a solution more easily, which can increase the search efficiency. Formula 2 takes account of the difference of the worse solution and the current solution, while Formula 3 only uses the effect of the iteration that makes it behave differently.

6. Conclusions and future developments

This research focuses on the mechanical harvester assignment and routing problem with time windows (HARPTW) to maximize the total areas serviced by a mechanical harvester under a sharing infield resources system. A mixed-integer programming model that can handle small-size problems was proposed. For large-scale problems, an effective ALNS that employs the idea of Differential Evolution (DE) is firstly designed to solve the problems integral to the mechanical harvester assignment problem (HAP) and mechanical harvester routing problem (HRP).

In our research, to develop the ALNS employing the idea of DE, the destroy method has been made iterative in the incomplete solution, such as destroying the list of sugarcane fields that have not yet been scheduled, then the repair method has been applied, in which a good constructive algorithm is hidden, to get a good solution. The final solution will be more flexible than if the destroy and repair methods have been executed only in the complete solution. Our algorithm has two phases, which are constructing the sequences of the sugarcane fields and the mechanical harvesters. Then the construction of the complete algorithm will be performed. Additionally, four new formulas were developed to calculate the probability of accepting a worse solution using a linear and parabola function, instead of the exponential function that is mostly used in the literature. The results of the proposed method show that the parabola function that uses the information of the solution quality outperforms all the other proposed heuristics. In this paper, mathematical and heuristics models were developed to solve the infield machinery sharing system which are easily adaptable, and should prove to be beneficial to other sugar industries, including other similar agro-food sectors in Thailand and around the world by reducing the costs of provision of agricultural infield machinery. Most importantly, the machinery sharing system can create sustainable sugar production by reducing infield machinery costs for small-size growers, increasing returns to infield machine owners, and also maintaining the right amount of supply for the sugar industry.

However, there is still much opportunity to extend our work in

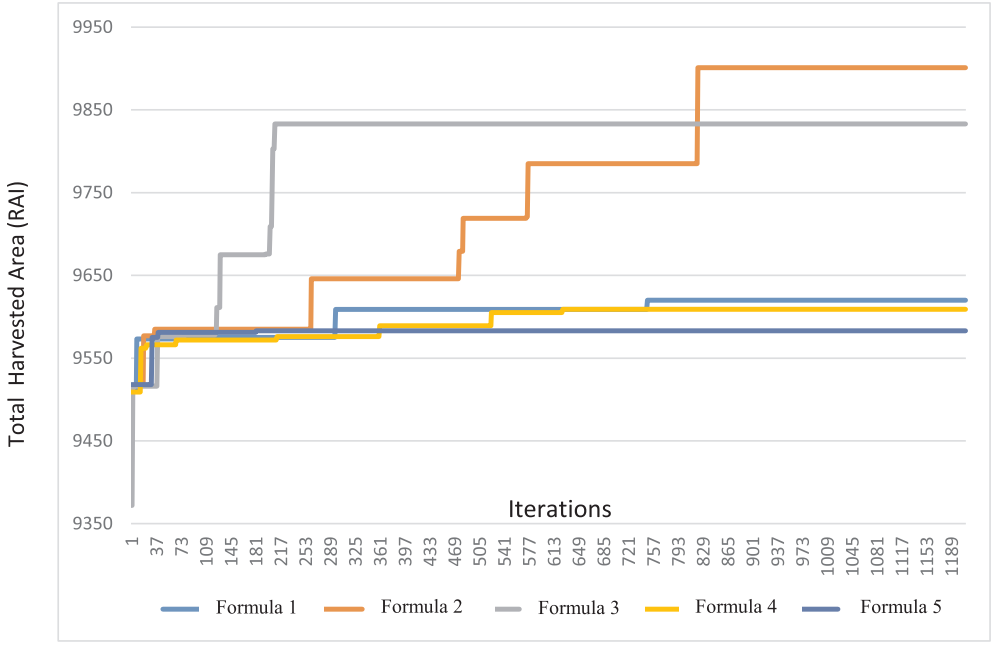

Fig. 4. Development of the best solution along the simulation using different acceptance Formulas 1–5.

many ways. Future development will be to investigate mechanical harvester conditions in terms of breakdowns which can be considered as a fuzzy variable. We believe that this issue can be added to our study to model real-world problems, and will be a valuable extension. As earlier stated, even though the proposed methods are very efficient and widely used, developing new destroy and repair methods including new formulas to accept the worse solution to be the starting solution for the search will be a valuable extension. Hence, research to determine the solutions of the problem should be carried out in future work by using metaheuristics or hybrid methods to compare the strengths of various approaches in solving problems of this nature.

Acknowledgement

This research was supported by the Research Unit on System Modeling for Industry, Khon Kaen University, Thailand (Grant No. 1/2560). The authors would also like to thank Mr. Ian Thomas for critical review of the manuscript.

Appendix A. Supplementary material

Supplementary data to this article can be found online at https://doi.org/10.1016/j.compag.2019.02.001.

References

Ahmed, A.E., Alam-Eldin, A.O., 2015. An assessment of mechanical harvester vs manual harvesting of the sugarcane in Sudan – the case of Sennar Sugar Factory. J. Saudi Soc. Agric. Sci. 14 (2), 160–166.
Aksen, D., Kaya, O., Salman, F.S., Tüncel, Ö., 2014. An adaptive large neighborhood search algorithm for a selective and periodic inventory routing problem. Eur. J. Oper. Res. 239 (2), 413–426.
Alinaghian, M., Shokouhi, N., 2018. Multi-depot multi-compartment vehicle routing problem solved by a hybrid adaptive large neighborhood search. Omega 76, 85–99.
Bruglieri, M., Pezzella, F., Pisacane, O., Suraci, S., 2015. A variable neighborhood search branching for the electric vehicle routing problem with time windows. Electron. Notes Discrete Math. 47, 221–228.
Cerdeira-Pena, A., Carpente, L., Amiama, C., 2017. Optimised forage mechanical harvester routes as solutions to a traveling salesman problem with clusters and time windows. Biosyst. Eng. 164, 110–123.
Chen, S., Chen, R., Wang, G.G., Gao, J., Sangaiah, A.K., 2018. An adaptive large neighborhood search heuristic for dynamic vehicle routing problems. Comput. Electr. Eng. 67, 596–607.
Dayarian, I., Crainic, T.G., Gendreau, M., Rei, W., 2015. A column generation approach for a multi-attribute vehicle routing problem. Eur. J. Oper. Res. 241 (3), 888–906.
Dechampai, D., Tanwanichkul, L., Sethanan, K., Pitakaso, R., 2017. A differential evolution algorithm for the capacitated VRP with flexibility of mixing pickup and delivery services and the maximum duration of a route in poultry industry. J. Intell. Manuf. 28 (6), 1357–1376.
Demir, E., Bektaş, T., Laporte, G., 2012. An adaptive large neighborhood search heuristic for the pollution-routing problem. Eur. J. Oper. Res. 223 (2), 346–359.
Díaz, J.A., Pérez, I.G., 2002. Simulation and optimization of sugar cane transportation in harvest season. In: Proceedings of the 2000 Winter Simulation Conference, pp. 1114–1117.
François, V., Arda, Y., Crama, Y., Laporte, G., 2016. Large neighborhood search for multi-trip vehicle routing. Eur. J. Oper. Res. 255 (2), 422–441.
Grunow, M., Günther, H.O., Westinner, R., 2007. Supply optimization for the production of raw sugar. Int. J. Prod. Econ. 110 (1–2), 224–239.
He, Pengfei, Li, Jing, Wang, Xin, 2018. Wheat harvest schedule model for agricultural machinery cooperatives considering fragmental farmlands. Comput. Electron. Agric. 145, 226–234. https://doi.org/10.1016/j.compag.2017.12.042.
Higgins, A., 2006. Scheduling of road vehicles in sugarcane transport: a case study at an Australian sugar mill. Eur. J. Oper. Res. 170 (3), 987–1000.
Hintsch, T., Irnich, S., 2018. Large multiple neighborhood search for the clustered vehicle-routing problem. Eur. J. Oper. Res. 270 (1), 118–131.
Jena, S.D., Poggi, M., 2013. Harvest planning in the Brazilian sugar cane industry via mixed integer programming. Eur. J. Oper. Res. 230 (2), 374–384.
Jiao, Z., Higgins, A.J., Prestwidge, D.B., 2005. An integrated statistical and optimisation approach to increasing sugar production within a mill region. Comput. Electron. Agric. 48 (2), 170–181.
Koç, Ç., 2016. A unified-adaptive large neighborhood search metaheuristic for periodic location-routing problems. Transport. Res. Part C: Emerg. Technol. 68, 265–284.
Kusumastuti, R.D., van Donk, D.P., Teunter, R., 2016. Crop-related harvesting and processing planning: a review. Int. J. Prod. Econ. 174, 76–92.
Lamsal, K., 2014. Sugarcane harvest logistics (Doctor of Philosophy). Retrieved at https://ir.uiowa.edu/etd/1349.
Le Gal, P.Y., Le Masson, J., Bezuidenhout, C.N., Lagrange, L.F., 2009. Coupled modelling of sugarcane supply planning and logistics as a management tool. Comput. Electron. Agric. 68 (2), 168–177.
Li, J., Pardalos, P.M., Sun, H., Pei, J., Zhang, Y., 2015. Iterated local search embedded adaptive neighborhood selection approach for the multi-depot vehicle routing problem with simultaneous deliveries and pickups. Expert Syst. Appl. 42 (7), 3551–3561.
Li, Y., Chen, H., Prins, C., 2016. Adaptive large neighborhood search for the pickup and delivery problem with time windows, profits, and reserved requests. Eur. J. Oper. Res. 252 (1), 27–38.
Liu, X., Laporte, G., Chen, Y., He, R., 2017. An adaptive large neighborhood search metaheuristic for agile satellite scheduling with time-dependent transition time. Comput. Oper. Res. 86, 41–53.
Lusby, R.M., Schwierz, M., Range, T.M., Larsen, J., 2016. An adaptive large neighborhood search procedure applied to the dynamic patient admission scheduling problem. Artif. Intell. Med. 74, 21–31.
Mancini, S., 2016. A real-life multi depot multi period vehicle routing problem with a heterogeneous fleet: Formulation and adaptive large neighborhood search based matheuristic. Transport. Res. Part C: Emerg. Technol. 70, 100–112.
Monroy-Licht, M., Amaya, C.A., Langevin, A., 2017. Adaptive large neighborhood search algorithm for the rural postman problem with time windows. Networks 70 (1),

44–59.
Muller, L.F., 2011. An adaptive large neighborhood search algorithm for the multi-mode RCPSP. DTU Manage. Eng. 3, 25.
Neungmatcha, W., Sethanan, K., 2015. Optimal mechanical harvester route planning for sugarcane field operations using particle swarm optimization. KKU Eng. J. 42 (2), 125–133.
Office of Agricultural Economics, 2003. Agricultural Production: Sugarcane. Office of Agricultural Economics, Bangkok, Thailand.
Qu, Y., Bard, J.F., 2012. A GRASP with adaptive large neighborhood search for pickup and delivery problems with transshipment. Comput. Oper. Res. 39 (10), 2439–2456.
Ribeiro, G.M., Laporte, G., 2012. An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. Comput. Oper. Res. 39 (3), 728–735.
Rifai, A.P., Nguyen, H.T., Dawal, S.Z.M., 2016. Multi-objective adaptive large neighborhood search for distributed reentrant permutation flow shop scheduling. Appl. Soft Comput. 40, 42–57.
Ropke, S., Pisinger, D., 2006. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transport. Sci. 40 (4), 455–472.
Salassi, M.E., Champagne, L.P., 1998. A spreadsheet-based cost model for sugarcane harvesting systems. Comput. Electron. Agric. 20 (3), 215–227.
Sethanan, K., Neungmatcha, W., 2016. Multi-objective particle swarm optimization for mechanical harvester mechanical harvester route planning of sugarcane field operations. Eur. J. Oper. Res. 252 (3), 969–984.
Smith, S.L., Imeson, F., 2017. GLNS: An effective large neighborhood search heuristic for the generalized traveling salesman problem. Comput. Oper. Res. 87, 1–19.
Stenger, A., Vigo, D., Enz, S., Schwind, M., 2013. An adaptive variable neighborhood search algorithm for a vehicle routing problem arising in small package shipping. Transport. Sci. 47 (1), 64–80.
Storn, R., Price, K., 1997. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. 11 (4), 341–359.
Sze, J.F., Salhi, S., Wassan, N., 2016. A hybridisation of adaptive variable neighbourhood search and large neighbourhood search: application to the vehicle routing problem. Expert Syst. Appl. 65, 383–397.
Thevenin, S., Zufferey, N., 2014, April. Variable neighborhood search for a scheduling problem with time window penalties. In: Proceedings of the 14th International Workshop on Project Management and Scheduling (PMS 2014), Munich, Germany.
Thevenin, S., Zufferey, N., 2018. Learning variable neighborhood search for a scheduling problem with time windows and rejections. Discrete Appl. Math. https://doi.org/10.1016/j.dam.2018.03.019.
Wen, M., Linde, E., Ropke, S., Mirchandani, P., Larsen, A., 2016. An adaptive large neighborhood search heuristic for the electric vehicle scheduling problem. Comput. Oper. Res. 76, 73–83.
Yagiura, M., Iwasaki, S., Ibaraki, T., Glover, F., 2004. A very large-scale neighborhood search algorithm for the multi-resource generalized assignment problem. Discrete Optim. 1 (1), 87–98.
Žulj, I., Kramer, S., Schneider, M., 2018. A hybrid of adaptive large neighborhood search and tabu search for the order-batching problem. Eur. J. Oper. Res. 264 (2), 653–664.