Row Allocation Negotiation for a Fleet of Strawberry Harvesting Robots

Madeline Mapes
Department of Mechanical and Aerospace Engineering,
University of Central Florida,
Orlando, FL 32816
e-mail: madeline.mapes@knights.ucf.edu

Yunjun Xu[1]
Professor
Mem. ASME
Department of Mechanical and Aerospace Engineering,
University of Central Florida,
Orlando, FL 32816
e-mail: yunjun.xu@ucf.edu

Cooperative small harvesting robots, mimicking a group of human pickers, have the potential to significantly reduce labor dependence in strawberry production. A tri-layered algorithm is investigated to effectively assign rows to robots with each robot incentivized to maximize its total number of picked strawberries within the fleet's harvesting time. The proposed algorithm consists of a decentralized local auction and negotiation strategy as the primary phase with a centralized fallback algorithm that guarantees an assignment. The salient features of the algorithm are reduced communication time, scalability, constant time complexity in the decentralized phase, and ease of implementation. The proposed algorithm is evaluated in a Monte Carlo simulation and the superior performance (e.g., significantly reduced computational time) is observed when compared with a centralized approach. It is expected that this row negotiation algorithm can address an important gap in strawberry harvesting via cooperative, small harvesting robots.

[DOI: 10.1115/1.4054644]

Keywords: autonomous systems, control applications, decentralized control, multivehicle systems

1 Introduction

High production costs, increasing populations, and labor shortages have driven the demand for automation in the agricultural industry [1–3]. Many operations in agricultural fields, such as weed control [4] and harvesting [5], have been or are being automated. Specialty crops, such as strawberries, are a research and engineering focal point of automation [6]. However, it is challenging to use robotics in some strawberry field operations which require more control and dexterity, such as harvesting, than grains or field crops [7]. To date, most strawberries are manually harvested, accounting for over 25% of the total production costs [7,8]. In Florida, many strawberry growers reported labor shortages while crops are in season [8], resulting in some of the yield being abandoned. Although autonomous harvesting has shown promising results in greenhouse conditions [9–11], advancements in field settings are quite limited. To increase the harvesting efficiency, industry players such as Agrobot[2] and Harvest Croo[3] developed robotic platforms with multiple independent harvesters that span the width of several strawberry rows. However, these approaches have foreseen inadequacies in the long run such as transportation difficulty, prone to single-point-of-failure, significant downtime impact, low platform flexibility, and low adaptation to field variations. It becomes natural to utilize cooperative, small robotic platforms in strawberry harvesting, analogous to a group of human pickers. Nevertheless, multirobot systems must address additional technique challenges, one of which is effectively scheduling a robot to a row in real-time.

So far, all research in scheduling can be broadly categorized as centralized or decentralized. To the best of our knowledge, centralized approaches are popular in solving robotic platform scheduling problems in agricultural fields, mainly due to the possibility that not many robots, if multiple robots are present, would work in one field at the same time [12–18]. For example, an automatic system for vineyards [12] utilizes a central computer to assign paths for robots to irrigate high-priority vines. Restricted by the layout of vineyards, robots can only change rows at row headlines [12,16]. In the case of an herbicide-spraying robot, task sets are assigned from a central computer that generates random solutions, improving at each iteration and converging on a set of assignments [13]. Using a cost matrix to assign tasks in a centralized approach can account for external factors like crop rotation, weather constraints, and urgency to harvest [14]. Agricultural surveillance drones can have flight paths calculated and assigned via a centralized computer [18], sometimes assisted by negotiations among agents [17].

Decentralized approaches such as bundle algorithms, greedy algorithms, and auction algorithms are widely seen in other application domains [19–21]. Choi et al. [20] proposed a two-layer decentralized algorithm to optimally assign multiple tasks to each agent with applications in office cleaning and building emergency surveying [19]. In those studies [19,20], each agent generates a bundle list via an auction algorithm in Phase 1, and conflicts are resolved via a consensus approach in Phase 2. Auction algorithms involve a bidding process where tasks are assigned based on the associated values [21], and the process is overseen by a centralized computer to reduce communication time and achieve consensus.

In this paper, a tri-layered negotiation algorithm is proposed for real-time scheduling consisting of a decentralized bidding process between neighboring robots (layer 1 and layer 2) and, if needed, a centralized algorithm afterward (layer 3). In layer 1, robots determine if they are able to continue harvesting. Layer 2 contains two algorithms: one for robots bidding to help neighbors and another for robots overseeing the bidding process initiated by its neighbors. If a solution cannot be obtained in the decentralized layer, the Hungarian Algorithm [22] is utilized in layer 3 to assign tasks to each robot requesting a new assignment. Robots are incentivized to cooperate like human pickers to harvest the field in minimal time while collecting as many strawberries as they can. However the robots are subjected to motion limitations due to row size and their own capabilities.

The work in this paper proposes a row allocation negotiation algorithm to address an important gap in strawberry harvesting

[1]Corresponding author.
[2]https://www.agrobot.com/e-series
[3]https://www.harvestcroo.com/technology

Manuscript received November 24, 2021; final manuscript received May 20, 2022; published online June 10, 2022. Assoc. Editor: Levi DeVries.

via cooperative, small harvesting robots. The technical contributions of the study are summarized here. (i) The decentralized part of the algorithm is scalable. The complexity of the centralized part is $O(N^3)$, but the probability of using it is low. (ii) The computational time is significantly reduced as compared to centralized approaches. (iii) The algorithm is easy to implement.

The remainder of the paper is organized as follows. In Sec. 2, the row allocation problem for robots in a strawberry field is defined with a list of assumptions. The algorithm is discussed in Sec. 3, and its communication, convergence, and computational complexity analyses are given in Sec. 4. The proposed approach is validated in a Monte Carlo simulation and its performance is discussed in Sec. 5. Conclusions are given in Sec. 6.

2 Problem Definition and Assumptions

There are a total of N_s rows in a strawberry field. For a typical field, two lines of trees are planted in every row, with mostly uniform distances between trees. There are N_{hr} harvesting robots in the fleet, each with identical functionalities and physical characteristics.

Figure 1 depicts a potential configuration of robots in a field, in which robots, their velocities, and the rows they occupy are shown along with descriptive information about the dimensions of the rows and the sets they belong to.

Definition 1. *The N_s rows are grouped into three sets: set Ω_n includes unharvested rows, set Ω_u includes rows currently under harvesting, and set Ω_h includes rows that have already been harvested.*

Definition 2. *r^+ and r^- denote the rows that are the right and left neighbors of row* r *in the same set, respectively.*

Definition 3. *The row that robot i is currently in is denoted as row(i), and the number of robots in row r is defined as num(r).*

The following assumptions and constraints are considered in the scheduling algorithm development.

Assumption 1. *Similar to a human picker, the total number of strawberries robot i harvested $n_{i,st}$ is proportional to the overbed length traveled $L_{i,h}$ as $n_{i,st} \propto L_{i,h}$, $i = 1, \ldots, N_{hr}$. Therefore, each robot will try to maximize this length within the total harvesting time. This assumption incentivizes a robot to help harvest a neighboring robot's row instead of going to a new, unharvested row that is far away from its current position.*

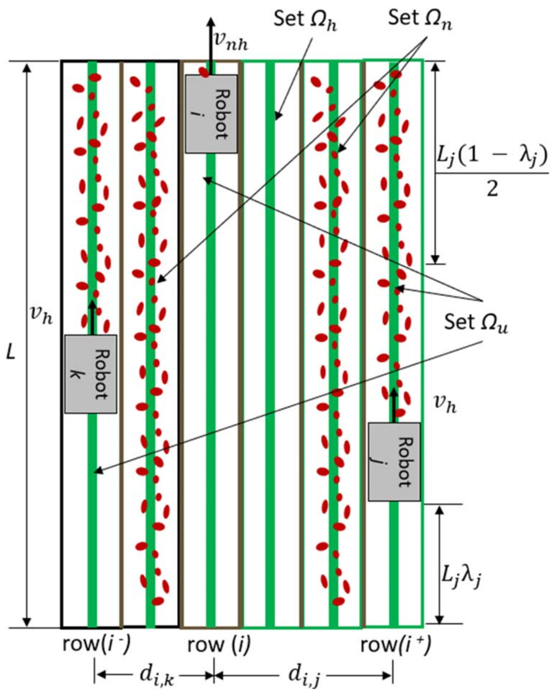

Fig. 1 Sketch of a farm and harvesting robot fleet configuration

Assumption 2. *Robot i, $i = 1, \ldots, N_{hr}$, has constant but distinct nonharvesting and harvesting speeds $v_{i,nh}$ and $v_{i,h}$ ($v_{i,nh} \geq v_{i,h}$), respectively.*

Assumption 3. *Robot i, $i = 1, \ldots, N_{hr}$, will continue to harvest for as long as they can, e.g., they still have sufficient battery life and rows are available for harvesting ($\Omega_n \neq \emptyset$ or $\Omega_u \neq \emptyset$).*

Assumption 4. *Robot i has a finite basket capacity C_i, $i = 1, \ldots, N_{hr}$ (i.e., the maximum number of strawberries) and can monitor the current number of strawberries in the basket $n_{i,b}$. The robot delivers the basket to the collection truck when the basket is full: $\varsigma_i = n_{i,b}/C_i = 1$. Basket delivery times are calculated by dividing the distance between a robot and the collection truck by its delivery speed.*

Assumption 5. *A robot always delivers its basket to the collection truck after completing a row and returns to the current row before requesting a new assignment.*

Assumption 6. *Robot i can calculate the distance to the end of row(j), meaning $d_{i,j}$, $\forall j \in \Omega_n \cup \Omega_u$. The distance to each row is the known width of each row multiplied by the number of rows between the robot and the row of interest. The turning distance is relatively small, thus omitted here.*

Assumption 7. *All robots in the fleet spend the same amount of time T in the farm. In reality, all robots in a fleet will be transported at the same time between a field and a storage facility. This implies that no robot should be idle if the overall objective is to minimize the harvesting time.*

Assumption 8. *There are collection trucks at both ends of the rows, allowing a robot to harvest up or down a row and deliver its basket to the truck on the closer end, assuring no motion conflict between two robots in the same row.*

Assumption 9. *No more than two robots are allowed per row (harvesting in opposite directions), meaning num(r) ≤ 2. A robot will deliver the basket to the collection truck on its side after finishing a row, according to Assumption 8.*

Assumption 10. *There are many nonharvesting events, such as moving toward a collection truck, repairing the powertrain, and replacing the onboard battery. Only the nonharvesting related traveling is modeled, and all the other nonharvesting operations will be neglected.*

The objective of the algorithm is to allocate a row to a robot when necessary in a decentralized fashion for the fleet to finish harvesting in the minimum time, i.e., $\min\{T\}$. Based on Assumption 7, $T_i = T = T_{i,h} + T_{i,nh}$, $i = 1, \ldots, N_{hr}$, denotes the time robot i spent on the farm including the times spent in harvesting operations $T_{i,h}$ and nonharvesting operations $T_{i,nh}$. Following Assumptions 1 and 2, $T_i = L_{i,h}/v_{i,h} + L_{i,nh}/v_{i,nh}$, in which $L_{i,nh}$ is the total distance robot i travels in nonharvesting operations. To minimize T_i, we seek to minimize the time when robot i is in nonharvesting operations. This can be done by reducing the distance traveled with constant speeds (Assumptions 2 and 10), reinforcing the incentive for robots to help nearby neighbors instead of traveling to a new row further away from its current row. Therefore, the performance index for the allocation algorithm to minimize is

$$\sup\{T_{1,nh}, T_{2,nh}, \ldots, T_{N_{hr},nh}\} \tag{1}$$

The optimization is subject to the constraints implied in the aforementioned assumptions. The nearest-neighbor topology is chosen for the communication considering typical commercial strawberry farm size, off-the-shelf wireless communication range, and number of harvesting robots in a fleet. Robot i, $i = 1, \ldots, N_{hr}$, can only communicate with its neighboring robots in $row(i^+)$ and $row(i^-)$, as well as the truck when necessary. The maximum number of bid iterations n_{req} among neighboring robots is set to two, but can be adjusted by users.

Table 1 Algorithm 1 in decision layer 1

```
1   Read in information from an onboard file
2   if “no battery”, “repairs are needed”, etc.
3       Stop harvesting, wait for maintenance
4   else if $\lambda_i = 100\%$
5       Deliver strawberries to a truck
6       If $\exists$ rows in $[row(i)^-, row(i)] \cup [row(i), row(i)^+]$, and these rows $\in \Omega_n$
7           go to the nearest row which belongs to $\Omega_n$; break; end
8       Enter Algorithm 2 in Decision Layer 2
9   else if $\zeta_i = 1$
10      Deliver strawberries to a truck, go back to the current position of row($i$) Continue harvesting
11  else if “any neighboring robot sent an offer”
12      Enter Algorithm 3 in Decision Layer 2
13  else
14      Continue harvesting
15  end
```

3 A Decentralized, Auction-Based Row Allocation Algorithm

The algorithm is divided into three decision layers comprised four algorithms. The first layer’s algorithm executes every time a robot finishes harvesting a distance equal to its body length l_i. If robot i receives a request from its neighbor in $row(j) \in [row(i)^+, row(i)^-]$, it will not process the request until it finishes its current harvesting distance (e.g., $l_i = 1m$).

3.1 Algorithm 1 in Layer 1. The algorithm begins in layer 1 (the pseudo code in Table 1), which is triggered when robot i harvests a distance l_i and reads in information from its file to decide if it should continue harvesting, etc. (line 1 of Table 1). Data in the file, updated by robot i itself, its neighboring robots, and/or the collection trucks, contain information about the farm, the progress, and condition of the robot fleet, and the inquiries sent by its neighboring robots. A robot will check for conditions that render it unable to harvest (lines 2 and 3 of Table 1). If robot i is at the end of its row, i.e., $\lambda_i = 100\%$ (line 4 of Table 1), it delivers its basket to the collection truck and returns to its row (Assumption 5). Robot i will determine which row to harvest next (lines 6–8 of Table 1) by either going to a new nearby row or entering Algorithm 2 to offer help to neighboring robots. Line 9 of Table 1 comes directly from the basket capacity definition in Assumption 4. If a robot receives an offer for a neighboring robot to come help (line 11 of Table 1), it enters Algorithm 3 to determine if this offer will be accepted. Otherwise, robot i will continue harvesting (line 14 of Table 1, Assumption 3).

3.2 Algorithm 2 in layer 2. In Algorithm 2, robot i initiates the bidding process by sending time incentives to its neighboring robots. Time incentives are used as currency in an auction process, where smaller time incentives are more valuable. The neighboring robots that receive time incentives will oversee the auction and determine the appropriate outcomes. We define n_{req} as the number of times robot i offers to help its neighbor. Robot i offers to help one of its neighboring robots in $row(j) \in [row(i)^+, row(i)^-]$ and waits long enough for robot j to respond (e.g., t_c ranging between 1 and 3 s) (line 5 of Table 2). The time incentive used in Algorithm 2 is calculated as follows. Per Assumption 5, a robot has an empty basket when calculating time incentives, therefore it does not need to factor in basket delivery times. The remaining fraction of row(j) that has not been harvested yet is defined as $L_j\,(1 - \lambda_j)$, where L_j is the length of $row(j)$ and λ_j is the percentage of the overbed length harvested. The distance robot i covers in transit is the distance between robot i and the end of row(j), $d_{i,j}$ (Assumption 6), plus $L_j(1 - \lambda_j)/2$. Furthermore, the time needed for both robot i and robot j to finish harvesting the row where robot j is in is counted in the time incentive. This yields the time incentive for robot i to help robot j, $t_i^j = [d_{i,j} + L_j(1 - \lambda_j)/2]/v_{i,nh} + L_j(1 - \lambda_j)/(v_{i,h} + v_{j,h})$, consistent with Assumption 9. If robot i receives positive responses (indicating it has won the bid) from both neighbors, the neighbor corresponding to the smaller t_i^j will receive help from robot i, based on Eq. (1), and it will update Ω_n, Ω_h, and Ω_u. If neither neighbor sends a positive response initially, robot i will offer to help a second time ($n_{req} = 2$). If it has not won a bid in the second iteration, it will enter the next decision layer. The bid offering process is shown in lines 5–16 of Table 2.

3.3 Algorithm 3 in layer 2. In Algorithm 3 (Table 3), robot i has finished harvesting a section of length l_i and checks to see if a neighboring robot in $row(j) \in [row(i)^+, row(i)^-]$ has offered to help (line 9 of Algorithm 1). Since the time incentives sent by the robot(s) on the far side of a row will be larger than that of the one(s) just on the near side and its motion is constrained by the field layout, it will decline the help offer immediately, and only focus on selecting between those that are closest to itself. Robot i compares the time incentive t_j^i to the time it would spend harvesting the remainder of the row by itself t_i^e and informs robot j if it won or not. If more than one time incentive was received from robots in different rows and robot j declines to come help after winning a bid, robot i will select the other time incentive t_j^i accordingly. If the

Table 2 Algorithm 2 in decision layer 2

```
1   If there is a row $k \in \Omega_n$ between row($i$) and $row(j)$, $j = [i^+, i^-]$
2       go to row $k$; break; end
3   Set $n_{req} = 1$
4   while $n_{req} \le 2$
5       Calculate and send $t_i^j$ to robot $j$, $j = [i^+, i^-]$, and wait for $t_c$ seconds
6       if a positive response is received from both robots $j = i^+$ and $j = i^-$
7           Select the smaller one of $[t_i^{i^+}, t_i^{i^-}]$
8           Inform robot $j$ it will receive help, and the other robot will not
9           Update $\Omega_n$, $\Omega_h$, and $\Omega_u$
10          Go to row($j$); break
11      else if only one neighboring robot in row($j$) sends a positive response
12          Inform robot $j$ it will receive help
13          Update $\Omega_n$, $\Omega_h$, and $\Omega_u$
14          Go to row($j$); break; end
15      else if $n_{req} = 2$
16          Enter Algorithm 4 in Decision Layer 3; break
17      end
18      $n_{req} = n_{req} + 1$
19  end
```

Table 3 Algorithm 3 in decision layer 2

```
1   Set $n_{req} = 1$
2   while $n_{req} \le 2$
3       Read in the time incentives $t_j^i$
4       if there are two robots in either row($j$) that send $t_j^i$
5           Inform the robot(s) on the far end of row($j$) that it has lost the bid
6       for remaining $t_j^i$ received, starting with the smaller $t_j^i$, $j = [i^+, i^-]$
7               if $t_j^i < t_i^e$
8                   Inform robot $j$ it won the bid
9                   Wait for $t_c$ seconds
10                  if robot $j$ confirms it will help
11                  Inform the other robot that it losts the bid; break; end
12              end
13          else
14              Inform robots $j$ they lost the bid; break
15          end
16      end
17      $n_{req} = n_{req} + 1$
18  end
```

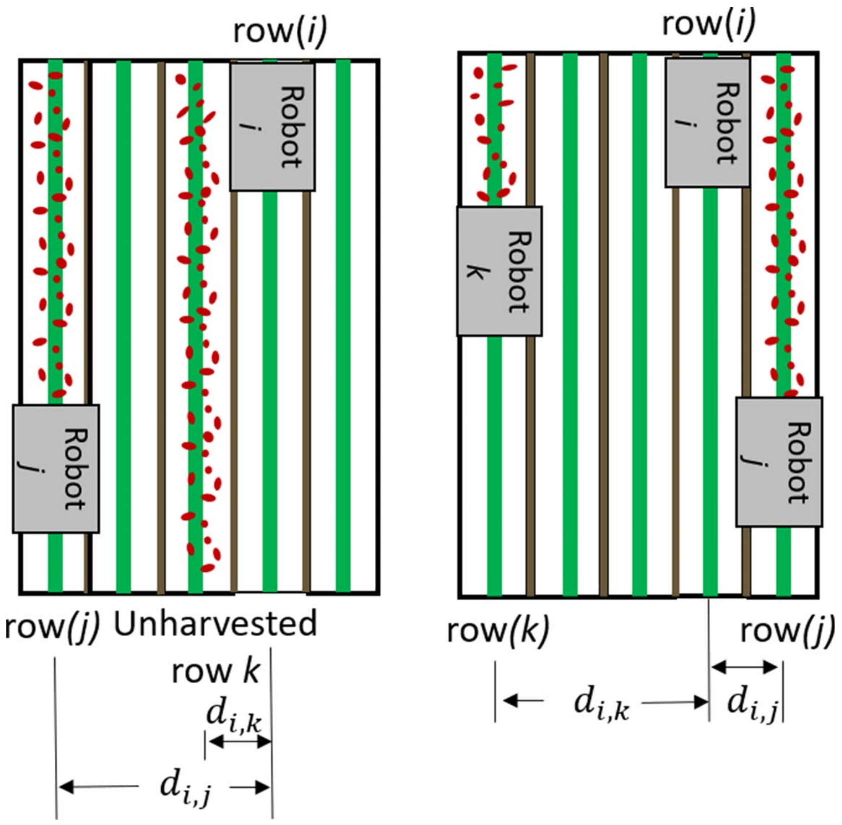

Fig. 2 (Scenario on the left) Robot *i* selects unharvested row *k* as the new assignment. (Scenario on the right) Robot *i* wins the bid to help robot *j* and selects row(*j*) as the new assignment.

second neighboring robot also declines to help, robot i will not be helped. A robot offering help will repeat the offer a second time if it fails to win a bid in the first iteration, therefore it is necessary for robot i to check for a second offer before resuming Algorithm 1. In this algorithm, the time incentive value for robot i to harvest its current row by itself is computed as $t_i^e = L_i(1-\lambda_i)/v_{i,h}$. It is noting that the harvesting time is included in the time incentive calculated in Algorithm 2 so to compare with robot i itself in Algorithm 3.

Figure 2 is shown with two example scenarios of a robot selecting rows affiliated with the lowest travel time as an unharvested row (left), and a partially harvested row occupied by a neighbor (right).

3.4 Algorithm 4 in layer 3.

Algorithm 4 begins when a robot fails to receive an assignment from its neighbors and requests an assignment from the collection truck. The truck's computer will wait t_w (e.g., between 1 and 3) seconds after receiving the first inquiry to allow other robots to submit requests before sending out the list of rows in Ω_n and Ω_u (line 3 of Table 4). The computer will then wait for t_c seconds for the robots to calculate and send time incentives. The time incentives are used to determine the row assignments via the Munkres Assignment Algorithm [22] in line 5 of Table 4. The collection truck computer will then update sets Ω_n, Ω_h, and Ω_u, shown in line 7. As mentioned before, Algorithm 4 is centralized; however, this algorithm has a low chance of being used. If the candidate row j belongs to Ω_n, the time incentive for robot i to move from its current row end to that row is $t_i^j = \gamma^{|j-row(i)|} d_{i,j}/v_{i,nh}$. Mimicking a human picker (Assumption 1), the discount factor $\gamma > 1$ is used to show the hesitation of a robot to go to a remote row. If the candidate row j belongs to Ω_u, the time incentive for robot i to move from its current row end to the remaining middle of row j is $t_i^j = \gamma^{|j-row(i)|} d_{i,j}/v_{i,nh} + L_j(1-\lambda_j)/2/v_{i,nh}$.

Here, the tri-layered, decentralized auction-based row allocation algorithm is abbreviated as DARA, and the decision process is shown in Fig. 3.

Table 4 Algorithm 4 in decision layer 3

1	Receive a row request from a robot
2	Wait for t_w seconds for more robots to send requests
3	Send out Set Ω_n to robots that requested a new row
4	Wait t_c to receive $t_i^j, j \in \Omega_n$ from robots requesting a new row
5	Optimal assignment using the Munkres Assignment Algorithm [22]
6	Send the row assignments to all robots that requested a new row
7	Update Sets Ω_n, Ω_h, and Ω_u

4 Algorithm Analyses

4.1 Communication Complexity.

Information will be exchanged among robots and/or trucks periodically. A robot will not update its global information such as the row sets until it reaches a truck. In the communication complexity analysis (the worst case scenario), we assume that any information regardless of its type (e.g., Booleans, integers, or floating point numbers) is stored as one unit.

In Algorithm 1 (Table 5), robot i will read information from its onboard file, which is updated by the collection trucks and its neighbors. The only information that will be transmitted from robot i to a truck is the robot index (one unit) and a Boolean (one unit) indicating if repairs, a battery replacement, etc., are needed.

In Algorithm 2, robot i sends an array of three units (a time incentive, its corresponding row, and the robot index) to a maximum of two available neighbors. The neighboring robots respond by each

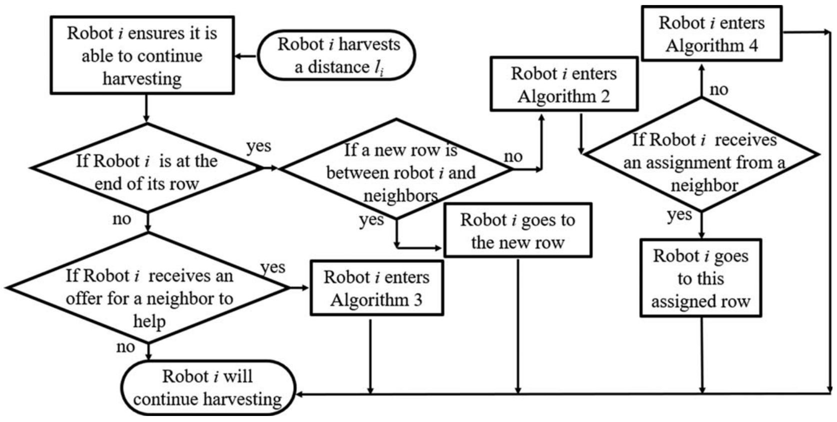
Robot i ensures it is able to continue harvesting
Robot i harvests a distance l_i
Robot i enters Algorithm 4
Robot i enters Algorithm 2
If Robot i is at the end of its row
yes
If a new row is between robot i and neighbors
no
no
If Robot i receives an assignment from a neighbor
no
yes
Robot i goes to the new row
yes
If Robot i receives an offer for a neighbor to help
yes
Robot i enters Algorithm 3
Robot i goes to this assigned row
no
Robot i will continue harvesting

Fig. 3 Interactions among decision layers

Table 5 Communication complexity

Layer#	To neighbors	From neighbors	Received by a truck	Sent by a truck
1	0	0	2	0
2	16	8	3	0
3	0	0	N_r+3	N_r+3

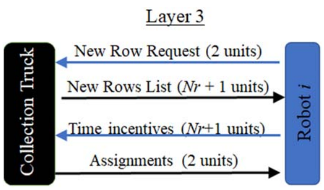
Layer 3
Collection Truck
Robot i
New Row Request (2 units)
New Rows List (Nr + 1 units)
Time incentives (Nr+1 units)
Assignments (2 units)

Fig. 4 Communications between robot *i* and truck in Algorithm 4 (layer 3)

sending two units of information (a Boolean and its index). If no bid is won, robot i resends the offers as three units of information and receives two units of information from each neighbor. Robot i then responds to each of its neighbors with two units of information and informs the truck of the row status changes by sending three units of information (the row that has been completed, the row it is moving to, and its index).

In Algorithm 3, robot i evaluates up to four time incentives sent by neighbors. It will immediately respond to two neighbors with a Boolean and an index (two units) to remove them from the bidding process and will send an additional two-unit response to the robot with the smallest time incentive. Robot i then receives a Boolean response and index (two units). Two units will be sent to its other neighbor if the initial response declines the offer, prompting it to send a two-unit response.

In Algorithm 4, as shown in Fig. 4, the collection truck receives a robot index and an integer prompting the algorithm to execute (two units) from a maximum of N_{hr} robots. Then the truck will send N_r integers indicating which rows are unharvested (N_r is the number of rows in Ω_n) and the truck index to each of the N_{hr} robots. It will then receive the robot index and N_r time incentives corresponding to the unharvested rows from the N_{hr} robots and will send back two units of information to each robot containing the robot index and an integer representing the new row assignment.

4.2 Convergence Analysis.

Consider a general instance where n_1 robots, $1 \le n_1 \le N_{hr}$, finished their rows and requested new assignments at around the same time. According to DARA, a robot requesting a new row will always receive an assignment if $\Omega_n \ne \varnothing$ or $\Omega_u \ne \varnothing$. If $\Omega_n = \varnothing$ and $\Omega_u = \varnothing$, the robot will wait in the farm until all robots finished their current rows.

The maximum time for the algorithm to converge is defined as the time needed for all the related robots (n_1 robots, their neighbors, and the trucks) to execute the algorithms including communication time, code execution time, and waiting time. Two cases are considered here. In case 1, a robot is seeking an assignment; in case 2, a robot receives a message from its neighbor offering help. The time needed for a truck to run its algorithm is included in these two cases.

Case 1: The robot enters Algorithm 2 to offer help to its neighbors and will enter Algorithm 4 if it does not win a bid. The Munkres Assignment Algorithm [22] in layer 4 guarantees an assignment for each robot requesting a new row if one is available. Thus, the robot will have a fixed number of instances of communication to obtain an assignment. The waiting time and the code execution time in all three layers are fixed and upper bounded. Therefore, the maximum time for a robot to obtain a new assignment is finite.

Case 2: The robot enters Algorithm 3 overseeing two iterations of the bidding process for up to four neighbors. In each iteration, a robot completes a fixed number of instances of communication for each neighbor. Combining the initial reception of help messages and the instances of rejecting robots on the far sides, there are also a fixed total instances of communication. The process finishes after the second iteration regardless of the results of the bidding process. Therefore, a robot receiving a help offer will oversee the bidding process in finite time.

Combining these two cases, the algorithm will converge in finite time.

4.3 Computational Complexity.

The computational complexities of the two cases mentioned in Sec. 4.2 are analyzed here.

Case 1: The CPU time used in calculating the time incentives is constant since only addition and multiplication operations are used. Communicating information about new rows is of constant complexity. Calculating time incentives corresponding to these rows is of $O(n_2)$ complexity, where n_2 is the number of rows related to n_1 robots and is bounded by the size of the strawberry field. The complexity of the centralized process, the Munkres Assignment Algorithm, has a $O(n_3^3)$ time complexity [22], where n_3 is the larger of the number of rows in Ω_n and Ω_u and the number of robots requesting a new assignment. Therefore, the upper bound of time complexity for case 1 is $O(n_3^3)$.

Case 2: The robot receives a help inquiry and oversees the bidding process, which is upper bound by the complexity of Algorithm 3 process. Reading in the time incentives, comparing these values, and communicating the results, each have constant time

Table 6 Settings in the Monte Carlo simulation

	Values
# of stops per row	100
Width of each row (m)	2
Distance between the truck and the nearest row (m)	2
Basket capacity (# of strawberries)	500
Delivery speed (m/s)	2.5
Harvesting speed (m/s)	0.1

Table 7 Independent variables in the Monte Carlo simulation

	Values
h (boolean)	{0,1}
η (%) (floating)	[0,100]
ζ (%) (integers)	[0,100]
$row(i)$ (integers)	[0,30]
λ of each robot (%) (integers)	[0,100]
# of harvesting robots	[5,80]
# of rows	[30,480]
γ	2

complexities. The process is limited to two iterations, resulting in an overall constant time complexity.

Combining both cases, the complexity of the centralized part in DARA is $O(n_3^3)$, while that of the decentralized part is $O(n_1)$. The chances of using the centralized part are low.

5 Simulation Results and Discussions

5.1 Simulation Setup. The Monte Carlo simulation aims to evaluate the ability of DARA to produce a new row assignment within the constraints. Table 6 lists the constant settings related to a simulated farm and harvesting robot fleet.

Different scheduling scenarios are mimicked by different combinations of the independent variables shown in Table 7. For example, a "repair" event is triggered when the Boolean representing a mechanical failure, h, is set to 1 or the battery life η (between 0% and 100%) is below a threshold. The determination of h is biased in favor of 0, as robots are not expected to fail frequently. Variables such as ζ, λ, and η are assigned with a normal distribution centered at 50%. When the variable indicating the percent fullness of a robot's basket, ζ, is at 100%, it will trigger the robot to deliver the basket to a nearby collection truck. The number of robots in the fleet is proportionate to the size of the farm ranging between 30 and 480 rows.

Each simulated robot is assigned to a randomly selected row; however, no more than two robots will be assigned to the same row. To reliably trigger a row assignment event, we let robot i always be at the end of its row, with other robots in the fleet given a random overbed position λ ranging between 0% and 100% of the row length. Since robots are likely to progress through the field near the rest of the fleet, robot i in $row(i)$ has two available neighbors, with other robots being randomly assigned to nearby rows.

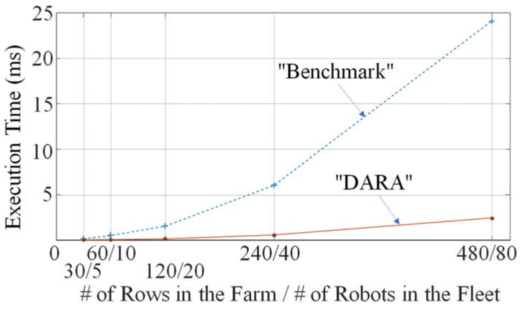

Fig. 5 CPU time complexity comparison

Ω_u, Ω_n, Ω_h, and time incentives are the dependent variables, which are computed based on the independent variables in Table 7. Set Ω_u is determined by $row(i)$ in Table 7 and set Ω_n includes rows randomly selected from all the rows excluding those in set Ω_u.

5.2 Results and Discussions. The Monte Carlo simulation of DARA is executed for 100,000 trials and compared to a benchmark central process (Benchmark). Benchmark utilizes only the centralized Munkres Assignment Algorithm [22] to award assignments from a set of available rows and is evaluated under the same conditions as DARA. The actual row assignment for each simulation run is not shown which is not crucial in demonstrating the advantages of the algorithm against the benchmark algorithm.

In Table 8, Benchmark and DARA assignments and execution times are observed for varying farm and fleet sizes. DARA assignments are globally optimal when they match the Benchmark assignments, which occurs a majority of the time. Discrepancies are caused by DARA assigning rows within the scope of its neighbors and Benchmark assigning unharvested rows outside of this range (e.g., the number of robots in the fleet) in cases where the overbed distance to travel to the starting point of an partially harvested row is significant. In these cases, DARA assignments are locally optimal. The results of the simulation indicate that, at minimum, the locally optimal assignment is obtained using DARA when compared to Benchmark with 100% of trials resulting in successful row assignments. Figure 5 shows the average execution times from Table 8 as the numbers of rows and robots increase, with DARA outperforming Benchmark and displaying much better scalability. It is worth noting that the waiting times are not included in the execution times.

Remark 1. The percentage of mismatches between DARA and Benchmark assignments reduces as γ increases since the increase of γ encourages Benchmark assignments to be within the

Table 8 Results of the Monte Carlo simulation

		Average execution time (ms)		Match (%) – high quality		Nonmatch (%)–Low Quality
# of Rows	# of Robots	DARA	Benchmark	Both assign neighboring row (%)	Both assign new rows (%)	Nonmatch (%)–Low Quality DARA to neighbor Benchmark to new row (%)
30	5	0.061	0.169	72.3	7.80	19.9
60	10	0.081	0.548	80.7	5.10	14.2
120	20	0.164	1.564	80.1	7.10	12.8
240	40	0.590	6.456	78.9	8.80	12.3
480	80	2.449	24.066	78.1	10.0	11.9

Table 9 Effects of the discount factor on the Monte Carlo simulation

	Match		Nonmatch
Gamma	Both assign neighboring row (%)	Both assign new row (%)	DARA to neighbor, benchmark new row (%)
1.1	66.9	7.35	25.7
1.5	80.9	7.12	12.0
2	86.6	7.14	6.25
2.5	90.9	7.33	1.78
3	91.1	7.19	1.77

neighboring range. This effect is shown in Table 9. In this case, there are 120 rows in the field, with the row width and length to be 2.5 m and 50 m, respectively. By increasing the discount factor, the cases where the DARA results differ from the Benchmark results can be reduced to less than 2% of total cases.

Remark 2. Increasing the width of each row reduces the percentage of mismatches between DARA and Benchmark assignments. Larger traveling distances between rows results in Benchmark preferring to assign rows within the neighboring range.

Remark 3. DARA and Benchmark assign rows outside of the neighboring range more often for smaller fleet sizes since the decreased span of the neighboring range reduces the traveling distance to get to new rows.

Remark 4. The DARA algorithm is designed based on the nearest-neighbor topology. If communication devices with wider ranges are adopted in the harvesting robot fleet, all-to-all communication topology and thus centralized algorithms can be used to achieve a better optimality with some latency. One of the future tasks is the detailed calculation of latency and bandwidth when using different communication topologies.

The applicability of algorithm is mainly constrained by the assumptions listed in Sec. 2. As one example, if Assumption 1 is invalid, meaning the total number of strawberries harvested is not proportional to the length a robot covered, the result achieved via this algorithm may only be a feasible solution. In this case, a stochastic optimization algorithm will need to be considered, which is beyond the scope of this study. Additionally, if collection trucks are only present on one side of the field (against Assumption 8), the distance calculation will be different. In this case, the algorithm is still applicable, but the results will be different.

6 Conclusions

The row assignment problem for a fleet of cooperative, small harvesting robots in strawberry fields is studied including local negotiations and a centralized fallback algorithm. The algorithm allocates a new row to a robot based on the time the robot predicts it will take to complete the row. The algorithm is validated in a Monte Carlo simulation and compared to a benchmark centralized algorithm. The results show that the algorithm finds local optimal solutions and has a lesser computational cost with superior scalability.

Acknowledgment

This work was supported by the National Science Foundation (Grant No. 1924622).

Conflict of Interest

There are no conflicts of interest.

References

[1] Liu, X., Xu, Y., Engel, B. A., Sun, S., Zhao, X., Wu, P., and Wang, Y., 2021, “The Impact of Urbanization and Aging on Food Security in Developing Countries: The View From Northwest China,” J. Cleaner Prod., **292**, p. 126067.

[2] Reddy, N. V., Reddy, A. V. V. V., Pranavadithya, S., and Kumar, J. J., 2016, “A Critical Review on Agricultural Robots,” Int. J. Mech. Eng. Technol., **7**(4), pp. 183–188.

[3] Tiffen, M., 2003, “Transition in Sub-saharan Africa: Agriculture, Urbanization and Income Growth,” World Dev., **31**(8), pp. 1343–1366.

[4] Wu, X., Aravecchia, S., Lottes, P., Stachniss, C., and Pradalier, C., 2020, “Robotic Weed Control Using Automated Weed and Crop Classification,” J. Field Rob., **37**(2), pp. 322–340.

[5] Yaghoubi, S., Akbarzadeh, N. A., Bazargani, S. S., Bazargani, S. S., Bamizan, M., and Asl, M. I., 2013, “Autonomous Robots for Agricultural Tasks and Farm Assignment and Future Trends in Agro Robots,” Int. J. Mech. Mechatron. Eng., **13**(3), pp. 1–6.

[6] Defterli, S. G., Shi, Y., Xu, Y., and Ehsani, R., 2016, “Review of Robotic Technology for Strawberry Production,” Appl. Eng. Agric., **32**(3), pp. 301–318.

[7] Feng, Q., Zheng, W., Qiu, Q., Jiang, K., and Guo, R., 2012, “Study on Strawberry Robotic Harvesting System,” 2012 IEEE International Conference on Computer Science and Automation Engineering, Zhangjiajie, China, May 25–27, pp. 134–140.

[8] Biswas, T., Wu, F., and Guan, Z., 2018, “Labor Shortages in the Florida Strawberry Industry: FE1041/FE1041, 7/2018,” EDIS, UF/IFAS Extension, 5.

[9] Rajendra, P., Kondo, N., Ninomiya, K., Kamata, J., Kurita, M., Shiigi, T., Hayashi, S., Yoshida, H., and Kohno, Y., 2009, “Machine Vision Algorithm for Robots to Harvest Strawberries in Tabletop Culture Greenhouse,” Eng. Agric. Environ. Food, **2**(1), pp. 24–30.

[10] Ko, M. H., Ryuh, B., Kim, Y. C., Suprem, A., and Mahalik, N. P., 2014, “Autonomous Greenhouse Mobile Robot Driving Strategies From System Integration Perspective: Review and Application,” IEEE/ASME Trans. Mechatron., **20**(4), pp. 1705–1716.

[11] Xiong, Y., Ge, Y., Grimstad, L., and From, P. J., 2020, “An Autonomous Strawberry-Harvesting Robot: Design, Development, Integration, and Field Evaluation,” J. Field Rob., **37**(2), pp. 202–224.

[12] Thayer, T. C., Vougioukas, S., Goldberg, K., and Carpin, S., 2020, “Multirobot Routing Algorithms for Robots Operating in Vineyards,” IEEE Trans. Autom. Sci. Eng., **17**(3), pp. 1184–1194.

[13] Conesa-Munoz, J., Bengochea-Guevara, J. M., Andujar, D., and Ribeiro, A., 2015, “Efficient Distribution of a Fleet of Heterogeneous Vehicles in Agriculture: A Practical Approach to Multi-path Planning,” 2015 IEEE International Conference on Autonomous Robot Systems and Competitions, Vila Real, Portugal, Apr. 9–15.

[14] Plessen, M. G., 2019, “Coupling of Crop Assignment and Vehicle Routine for Harvesting Planning in Agriculture,” Artif. Intell. Agri., **2**, pp. 99–109.

[15] Thayer, T. C., Vougioukas, S., Goldberg, K., and Carpin, S., 2018, “Routine Algorithms for Robot Assisted Precision Irrigation,” 2018 IEEE International Conference on Robotics and Automation, Brisbane, Australia, May 21–25.

[16] Kan, X., Thayer, T. C., Carpin, S., and Karydis, K., 2021, “Task Planning on Stochastic Aisle Graphs for Precision Agriculture,” IEEE Rob. Autom. Lett., **6**(2), pp. 3287–3294.

[17] Zhang, Y., Feng, W., Shi, G., Jiang, F., Chowdhury, M., and Ling, S. H., 2020, “UAV Swarm Mission Planning in Dynamic Environment Using Consensus-Based Bundle Algorithm,” Sensors, **20**(8), p. 2307.

[18] Kulbacki, M., 2018, “Survey of Drones for Agriculture Automation From Planting to Harvest,” 2018 IEEE 22nd International Conference on Intelligent Engineering Systems, Las Palmas de Gran Canaria, Spain, June 21–23.

[19] Johnson, L. B., Choi, H., and How, J. P., 2016, “The Role of Information Assumptions in Decentralized Task Allocation: A Tutorial,” IEEE Control Syst. Mag., **36**(4), pp. 45–58.

[20] Choi, H., Brunet, L., and How, J. P., 2009, “Consensus-Based Decentralized Auctions for Robust Task Allocation,” IEEE Trans. Rob., **25**(4), pp. 912–926.

[21] Naparstek, O., Zafaruddin, S. M., Leshem, A., and Jorswieck, E. A., 2019, “Distributed Energy Efficient Channel Allocation,” IEEE Trans. Green Commun. Netw., **3**(4), pp. 1152–1166.

[22] Kuhn, H. W., 1955, “The Hungarian Method for the Assignment Problem,” Naval Res. Logistics Q., **2**(1–2), pp. 83–97.