# Optimally Scheduling Complex Logistics Operations Involving Acquisition, Elaboration and Action Tasks

Lohic Fotio Tiotsop, Antonio Servetti, Enrico Masala
*Control and Computer Engineering Department*
*Politecnico di Torino*
Torino, Italy
lohic.fotiotiotsop@polito.it, antonio.servetti@polito.it, enrico.masala@polito.it

***Abstract*—Optimization algorithms can dramatically improve the efficiency of logistics operations. However, when complex tasks are involved, there is a significant lack of good models and optimized solutions. For instance, when it is necessary to perform more than one operation at each specific point in space, potentially depending on the outcome of others, finding optimal solutions for the operator movements may become extremely difficult. This work addresses such context, in which one operator is required to perform acquisition, elaboration and action tasks while visiting a set of positions, with complex sequential dependencies. An example could be an automatic operator, i.e., an unmanned ground vehicle (UGV), that is required to help in precision agriculture activities. For clarity's sake, we will mostly focus on the precision agriculture scenario, but our model and results can be readily applied to the other contexts described in more details in the paper, e.g., scheduling of robot operations or repair companies. After formulating the general problem analytically as an integer linear programming problem, we propose an algorithm that is guaranteed to find the global minimum cost solution in terms of time needed to complete the operations. Tests on instances of the optimization problems with a different number of nodes show the efficiency of the proposed algorithms in terms of computational time.**

***Index Terms*—logistics optimization, efficient path planning, UGV navigation, integer linear programming**

## I. Introduction

Improving the efficiency of logistics operations is becoming increasingly important in modern world. However, real world operations are often more complex than the ones described in traditional logistics models, e.g., travelling salesman problem (TSP) and vehicle routing problem (VRP), which have been intensively studied in literature [1], [2]. In fact, in real world situations the operator often needs to perform, at each specific location, different tasks that might depend on its previous activity, e.g., in terms of movements, thus introducing strong dependencies which often increase the complexity of the problem so that it cannot be solved in reasonable time.

In this work, we focus on the increasingly important precision agriculture context in which an unmanned ground vehicle (UGV), i.e., a rover, has to perform tasks, at given location, that can not always be defined in advance because they depends on the condition of the plants. Thus the UGV may need to report difficult situations to a decision maker, e.g., a human operator or some sophisticated software tools that cannot be put onboard in the robot, that can then decide the correct action that the robot has to execute. Often the transmission can be performed from the same location of the operation, but it is also necessary to consider that in rural areas it often happens that successful data communication can only be achieved in certain locations in the field. Unfortunately, this key aspect is still neglected in current literature that focuses, instead, on fully automated operations by robots. In this context, we aim to address all the previous shortcomings by proposing a more comprehensive optimization model than the ones investigated in the literature [3] which also includes the necessity to report to a central location and get instructions on the task to be done before proceeding to perform each action.

However, the model proposed in our work can also be applied to other context besides precision agriculture, as it will be briefly detailed in the beginning of the problem formulation. In general, our proposed model and solutions can be used for efficiently planning operations requiring data acquisition at given locations, data elaboration or transmission, and finally an action. In practice, we assume to have a rough, preliminary knowledge about the terrain and how a UGV can move in the area, i.e., obstacle positions and possible navigation paths are known. This can be obtained by previous knowledge or by means of a survey, e.g., made using UAV or other similar technologies [4]–[6]. During the survey, points of interests (POIs) are also identified, to be visited by the UGV in order to gather information, decide what to do and act accordingly. Since the decision of the action to perform in each POI might be difficult to take only relying on the on-board processing capabilities or simply might require human judgment not available through algorithms, we assume that after the UGV has gathered information by moving to the POI and taking, e.g., a picture, it must communicate with a central location to get an answer about the farming action to perform from, e.g., a human operator. The UGV must then perform the action returning to the POI. Note that we expect that in rural areas communications might be difficult. Therefore, in our scenario, we assume that we roughly know that in certain places on the terrain communication is possible, whereas in others this is not possible at all.

The objective of this work is to model and efficiently find the optimal solution of the navigation problem of the UGV that minimize the time needed to carry out all the activities in the POIs, subject to the following constraints. Before carrying out a task in a POI, the UGV must i) move to the POI and collect data about its state, ii) move to a place where the collected data can be transmitted and instructions are received about what to to at the POI, iii) go again to the POI and perform the action. Clearly. the UGV does not have to perform the three steps consecutively for each single POI, but it can move to multiple POIs before visiging a place where it can transmit the information and receive the instructions, and then it can go back to each one of them, in any order, to carry out the operations as instructed.

The results presented in this work are twofold. First, an integer linear programming (ILP) formulation of the problem is presented. This allows to solve the problem with any generic ILP solver. However, the time required to solve even small problem instances can be significant. Therefore, we propose an optimized algorithm can significantly outperform the ILP formulation. A prototype implementation of such algorithm is shown to significantly outperform the ILP model on the same instances of the problem. Also, its usefulness is shown by solving a problem whose geometry is derived from an actual precision agriculture problem.

The paper is organized as follows. Section II reviews related work in the field, then a new model for logistics operations involving acquisition, elaboration/transmission, and action tasks is presented in Section III in the form of an integer linear programming problem. Due to the the high complexity of the problem, in Section IV we propose a new branch and bound algorithm specifically tailored to such type of problem. Section V shows some experimental results followed by conclusions in Section VI.

## II. Related work

Agricultural applications rely more and more on sensing technologies in order to perform crop monitoring [7], [8]. One of the most common ways to acquire information is through imaging sensors, for instance in the visible spectrum [9] or at other wavelengths [10]. Image data can then be elaborated to detect cultivation type, e.g., vineyard [11], or to detect single plants and potential anomalies [12]. A more comprehensive overview of such technologies can be found in [13]. Once potential issues are identified, a set of POIs can be extracted and then, on the basis of the cultivation map, a UGV can move in such positions in order to perform a more detailed analysis, e.g., taking closer pictures or perform local measures with sensors, to later perform some specific physical action [14], [15]. For the purpose of solving the UGV movement problem, the cultivation field can be abstracted in the form of a graph with nodes (i.e., the POIs) connected by edges whose weight represent the cost, in terms of time, needed to move from one point to the other.

From the theoretical point of view, graph visiting problems are well studied. Examples are the travelling salesman problem (TSP) or more in general the vehicle routing problem (VRP) and its variants. Consider a set of vehicles that can transport goods and a graph whose nodes represent a set of customers, each one characterized by a given demand, and the edges report the cost that should be sustained to move from a costumer to another. The VRP aims at determining which costumers should be served by any vehicle and the schedule of the operations of any vehicle in order to minimize the total cost satisfying the capacity constraint of each vehicle and the demand of each costumer. If the operations are performed by a single vehicle the problem becomes the TSP. Both the VRP and the TSP have been intensively studied in literature [1], [2] and their related works represent an important background knowledge available about graph visiting problems.

A large number of papers in literature investigate the problem of reducing field work time assimilating the scheduling of agricultural tasks to graph visiting problem. In [16]–[18] the authors explain how the VRP can be used to gain efficiency in field logistics. In particular in [16] the numerical experiments show that it is possible to save up to 32% of the time if the operations are guided by the solution of the proposed VRP instead of an intuitive approach. The TSP is used in [19] to schedule the farming activities in fields characterized by the presence of a large number of obstacles.

Despite these works aim at minimizing the time needed to complete the assigned farming tasks, similarly to our navigation problem, they are substantially different in the fact that they do not consider the communication aspect and thus the interaction with an external decision maker, e.g., a human operator or any other external tool, whose presence coordinates the operations and addresses critical situations. Moreover, they do not consider that certain locations have to be visited twice, i.e., the first time to acquire data and the second time to perform an action.

## III. Mathematical formulation

In this section we give a formal definition of the optimization problem including the objective and the constraints. Before describing all the details of the problem, we highlight that the presented model can also be useful in other contexts. Here we provide two examples. In case of scheduling of industrial robot operations, often a large number of jobs have to be done, located in different places, and a human supervisor may just have a rough knowledge of what is the exact job to be done at each location. Hence, robots should move to each location, so that information is collected before the human decides what has to be done. In addition, other places could have to be visited by the robots to collect instruments necessary to perform the jobs. Another example is the logistics operations scheduling of repair companies. Operators often need to visit customer locations to identify the problem, then either fix it directly or go to warehouses to get, e.g., the needed replaceable parts, and finally moving back to the customer location to finalize the repair.

For the case of precision agriculture, in our work we assume that a picture has been used to derive a graph that represents the field in which the UGV must operate. The task of creating a graph from a picture by detecting the different types of areas and how they are connected and which is the optimal movement path between different areas is a problem well addressed in literature (see [20]–[23]). Therefore, here we assume that such graph is available.

Note also that the problem formulation does not address the tasks that have to be performed at each visited node, since we assume there is no interdependency between the tasks and all have to be performed according to the received instructions. Therefore, the time taken by performing each task is simply an additive term which has no dependency on the UGV path.

Let denote by $\mathcal{A}$ the set of POIs to visit, $\mathcal{B}$ the set of nodes where wireless communication can be established, $\mathcal{K}$ the set of displacements of the UGV, $\mathcal{E}$ the set of edges connecting the different nodes, $t_{ij}$ the time required to cover the edge $(i,j) \in \mathcal{E}$, $A$, $B$, $E$ and $K$ respectively the cardinality of $\mathcal{A}$, $\mathcal{B}$, $\mathcal{E}$ and $\mathcal{K}$. We then consider the boolean variables $x_{ij}^k$ equal to 1 if during its k-th displacement the UGV moves along the edge $(i,j) \in \mathcal{E}$ and $y_i^k$ equal to 1 if during the k-th displacement the UGV transfers the data collected from the node $i \in \mathcal{A}$. The model is formulated as follows

$$\min_{x,y} \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{E}} x_{ij}^k t_{ij} \tag{1}$$

subject to:

$$\sum_{\{j\in\mathcal{A}\cup\mathcal{B}:(s,j)\in\mathcal{E}\}} x_{sj}^1 = \sum_{\{i\in\mathcal{A}\cup\mathcal{B}:(i,s)\in\mathcal{E}\}} x_{is}^K = 1, \tag{2}$$

$$\sum_{(i,j)\in\mathcal{E}} x_{ij}^k = 1 \quad \forall k \in \mathcal{K}, \tag{3}$$

$$\sum_{k\in\mathcal{K}} y_i^k = 1 \quad \forall i \in \mathcal{A}, \tag{4}$$

$$\sum_{k\in\mathcal{K}} \sum_{\{i\in\mathcal{A}\cup\mathcal{B}:(i,j)\in\mathcal{E}\}} x_{ij}^k \geq \begin{cases} 2 & \forall j \in \mathcal{A}\setminus\mathcal{B}, \\ 1 & \forall j \in \mathcal{A}\cap\mathcal{B}, \end{cases} \tag{5}$$

$$x_{ij}^{k-1} \leq \sum_{\{i\in\mathcal{A}\cup\mathcal{B}:(j,i)\in\mathcal{E}\}} x_{ji}^k \qquad \forall (i,j)\in\mathcal{E} \;\; \forall k \in (\mathcal{K}\setminus\{1\}), \tag{6}$$

$$y_i^k \leq \sum_{\{j\in\mathcal{B}:(i,j)\in\mathcal{E}\}} x_{ij}^k \qquad \forall i \in \mathcal{A} \;\; \forall k\in\mathcal{K}, \tag{7}$$

$$y_j^k \leq \sum_{1\leq t\leq k} \sum_{\{i\in\mathcal{A}\cup\mathcal{B}:(i,j)\in\mathcal{E}\}} x_{ij}^t \;\; \forall j \in \mathcal{A} \;\; \forall k\in\mathcal{K}, \tag{8}$$

$$y_j^k \leq \sum_{k<t\leq K} \sum_{\{i\in\mathcal{A}\cup\mathcal{B}:(i,j)\in\mathcal{E}\}} x_{ij}^t \;\; \forall j \in \mathcal{A}\setminus\mathcal{B} \;\; \forall k\in\mathcal{K}, \tag{9}$$

where the variables $x$ and $y$ are binary: $x_{ij}^k \in \{0,1\}, y_i^k \in \{0,1\}$.

Eq. (1) requires the minimization of the total time required to perform all the operations, Eq. (2) imposes that the path of the UGV should start and end at a given node (later denoted by $S$). The constraints in Eq. (3) ensure that each displacement of the UGV corresponds to one edge in the solution and Eq. (4) requires that the data collected from any node $i \in \mathcal{A}$ are transmitted once and only once. The constraints in Eq. (5) ensure that the UGV visits, i) at least twice, the nodes of interest in the set $\mathcal{A}\setminus\mathcal{B}$ where it is not possible to communicate and ii) at least once, the other nodes of interest in the set $\mathcal{A}\cap\mathcal{B}$. Eq. (6) requires the solution to be a continuous path. The requirement that the UGV might transfer from a node only if such node belongs to the set $\mathcal{B}$ is given by Eq. (7). Eq. (8) requires that the UGV transfers the information related to a given node in the set $\mathcal{A}$ only if such node has already been visited, Eq. (9) requires that, once the information corresponding to the visit of a node in the set $\mathcal{A}\setminus\mathcal{B}$ where communication is not possible has been transmitted, the UGV must visit the node again once more, due to the specific requirements of the problem we are addressing.

As claimed before, the model in Eq. (1)-(9) con actually be used to schedule any logistics operations requiring data acquisition, elaboration/transmission and action. It is in fact enough to assimilate the set $\mathcal{A}$ to the set of jobs or client locations and the set $\mathcal{B}$ to stores or warehouses locations where one needs to go before finalizing the work at the $\mathcal{A}$ locations.

## IV. Proposed solution

The main reason of the difficulty of the model in Eq. (1)-(9) it is that it includes the set $\mathcal{K}$ of displacements whose cardinality $K$ should actually represents the number of edges present in the optimal solution. This is different from the case of many graph visiting optimization problems such as the TSP or VRP where such number is known in advance. In other words, the value of $K$ is not known a priori and thus the exact number of decision variables necessary to find the optimal solution is unknown. This issue could be solved by assigning a very large value to $K$ and allowing the UGV not to move to a new node to match the $K$ value. Unfortunately, the number of binary variables in the problem increases with $K$, therefore the complexity of the problem easily increases. The problem can still be solved by adopting a column generation approach (CGA) whose implementation requires to formulate a restrictive master problem (RMP) associated to the problem in Eq. (1)-(9) and progressively add new columns and thus new variables to the RMP until the minimum number of variables necessary to get the optimal solution is reached. More details about the CGA can be found in [24], [25]. The main drawback of this approach is the difficulty of getting a RMP associated to a given integer linear programming problem that guarantees both the effectiveness and the efficiency of the CGA. In this paper we focus our attention on an alternative approach presented in the next Section.

### A. Proposed graph visiting algorithm

Let us denote by $A_1$,$A_2$,..., $A_A$ the A nodes to be visited. While the UGV is moving through the graph, the state of the UGV itself is completely defined by

$$\mathbb{S} = (n_1, n_2, \ldots, n_A, U, C)$$

where $U$ is the node the UGV is visiting, $C$ is the total cost of the edges traversed so far and

$$n_i = \begin{cases} 0 & \text{if } A_i \text{ has not yet been visited;} \\ 1 & \text{if } A_i \text{ has been visited;} \\ 2 & \text{if the data collected at } A_i \text{ has been transmitted;} \\ 3 & \text{if the task required at } A_i \text{ has been performed.} \end{cases}$$

The state $\mathbb{S} = (n_1, n_2, \ldots, n_A, U, C)$ is said to be a feasible solution of the problem if $U = S$ and

$$n_i = 3 \quad \forall i \in \{1, 2, \ldots, A\}.$$

The state $\mathbb{S} = (n_1, n_2, \ldots, n_A, U, C)$ is branched taking into consideration all possible displacements of the UGV from the node $U$ to any other node connected to $U$ by one edge and listing all new possible states. Repeating such process on the new states leads to the construction of the state tree.

The algorithm implements two main operations: branch and bound. The first one (branch) consists in expanding the state tree by generating new states from its leaves, and the second one (bound) is the pruning procedure that consists in closing new leaves of the state tree that can not lead to the optimal solution. The pruning is based on the following two rules:

1) if the cost of any state among those to be expanded (the leaves) is greater than the cost of any already computed feasible solution, such state is closed, i.e., it is no more considered for branching;
2) if any state among those to be branched differs from a state $\mathbb{S}$ already reached just for its cost, such state is closed if its cost is greater than the one of $\mathbb{S}$.

While expanding the state tree, leaves can be considered in several different orders by means of a priority value. The simplest strategy to assign such a value is to use a monotonically decreasing value each time a new leaf is created. In such a way leaves are considered in creation order. Other more advances strategies are possible. One of them will be defined and investigated in Section V.

When branching a leaf $\mathbb{S}$ it may happen that all the states generated from $\mathbb{S}$ are cancelled when the two pruning rules are applied. In this case $\mathbb{S}$ is closed and no more considered. Note that closed states also include feasible solutions which clearly do not need to be further expanded. The expansion of the state tree and the pruning procedure are then iteratively applied until all the leaves of the state tree are closed. In order to formally present the algorithm we introduce the following definitions:

- $\mathcal{G}$ the graph associated to the problem;
- $\mathcal{S}_T$ the state tree;
- $N_U$ the number of nodes adjacent to the node $U$ in the graph $\mathcal{G}$;
- $\mathcal{L}_g$ the list of all new states generated when branching the leaf having highest priority;
- $\mathcal{L}_p$ the list of all the states in $\mathcal{L}_g$ that are not pruned after the function $Prune()$ has been called;
- $\mathcal{F}$ the list containing the closed states;
- $\{\}$ the empty list
- $Pop(\mathcal{L})$ a function that takes as input a list of states and returns the state $\mathbb{S}$ with highest priority;
- $Branch(\mathbb{S}, \mathcal{G})$ a function that takes as input a state $\mathbb{S} = (n_1, n_2, \ldots, n_A, U, C)$ and the graph, branches the received state and returns a list of $N_U$ states, one for each of the adjacent nodes to $U$;
- $Insert(\mathcal{L}_p, \mathcal{S}_T)$ a function that inserts in the state tree the list of states received as input;
- $Prune(\mathcal{S}_T, \mathcal{L}_g)$ a function that takes as input the state tree $\mathcal{S}_T$ and the list of newly generated states $\mathcal{L}_g$ then eventually prunes some states from $\mathcal{S}_T$ and $\mathcal{L}_g$ according to the aforementioned pruning rules and returns the list of remaining states from $\mathcal{L}_g$;
- $GetSolution(\mathcal{S}_T)$ a function that extracts from the state tree the branch that leads to the leaf state that is a feasible solution with the minimum cost.

The algorithm consists of the steps defined in Algorithm 1.

It is worth noting that the proposed algorithm solves the problem *without making any assumption* about the value of $K$. The value of $K$ can be simply computed at the end by just counting the number of edges present in the solution.

## V. Numerical results and discussion

The numerical experiments conducted in this section aim at comparing the complexity of solving the integer linear programming problem in Eq. (1)-(9) with both commercially available solvers and our branch and bound algorithm specifically designed for this problem. Given that the proposed model is considerably different from well studied vehicle routing problems, no benchmark instances are available in literature. Hence we generated though simulation many realistic modest size instances and provided up to one hour of computational time to commercial solvers to solve each instance. More precisely, we considered the map of a vineyard of the Langhe area in Piedmont (Italy). Then, we identified 26 areas in which potential actions may be required or communication is possible, and 32 edges representing feasible paths between them. To generate different problem instances, we randomly selected $N$ areas from the 26 available ones to form the set

**Algorithm 1** Compute Optimal Path

```
$\mathcal{S}_T = \{(0, 0, \ldots, S, 0)\}$
$\mathcal{F} = \{\}$
**while** $((\mathbb{S} = Pop(\mathcal{S}_T \setminus \mathcal{F})) \neq \{\})$ **do**
    $\mathcal{L}_g = Branch(\mathbb{S}, \mathcal{G})$
    $\mathcal{L}_p = Prune(\mathcal{S}_T, \mathcal{L}_g)$
    **if** $(\mathcal{L}_p == \{\})$ **then**
        $\mathcal{F} = \mathcal{F} \cup \mathbb{S}$
    **else**
        $Insert(\mathcal{L}_p, \mathcal{S}_T)$
    **end if**
**end while**
$Sol = GetSolution(\mathcal{S}_T)$
```

$\mathcal{A}$ of POIs and the set $\mathcal{B}$ of areas covered by wireless signal, i.e., where communication is possible. In the experiments, we considered 10 different values of $N$, ranging from 7 to 16. For each value of $N$ we generated 5 instances and hence a total number of 50 instances have been generated. The graph in Fig. 1 represents a sample of the graphs obtained for $N = 16$. The *squares* represent nodes in $\mathcal{A}$, the *diamonds* represent nodes in $\mathcal{B}$, the *stars* represent nodes in $\mathcal{A} \cap \mathcal{B}$ and the *circles* represent the nodes not selected in which the UGV just transits.

In order to quantify the efficiency of our algorithm with respect to those implemented in commercial solvers we first run our proposed algorithm on each instance and, once the optimal path is obtained with the associated cost, we compute the exact value of $K$. The integer linear programming model of the problem is then solved by means of the integer linear programming solver available in the CPLEX software [26] using three different estimations of $K$ including the one that yields the optimal solution. In order to further improve the efficiency of the proposed algorithm, we designed a slightly different version of the algorithm (named *advanced* version in the following) in which the priority when expanding the state tree is set to the inverse of the cost of the state, thus the state tree is always expanded from the leaf state with the lower cost. Following the minimum cost path, and not a path given by the ordering of the nodes, the pruning function is more efficient and a larger number of states are discarded by the prune rules because the algorithm already found an equivalent state with a lower cost.

The time taken by the CPLEX solver considering the exact value $\hat{K}$ of $K$ as well as an overestimation of $\hat{K}$ equal to $\hat{K}+2$ and $\hat{K}+4$ is shown in Table I. When $K = \hat{K}$, CPLEX requires the lowest time, as expected, and provides, of course, the same solution of our algorithm. However, note that the CPLEX solver, despite being a highly optimized commercial software, starts to require more than one hour even when $N$ is relatively low. For $N > 11$ (not reported in the table) even using the optimal value of $\hat{K}$ requires more than an hour. Note also that, in general, if the optimal $K$ value is unknown,

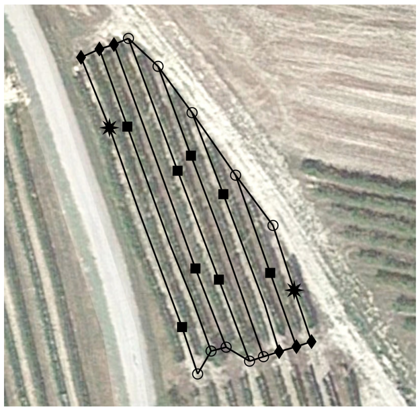

Fig. 1. An example of undirected graph that represents the POIs and the available paths for the UGV on a real vineyard for the case $N = 16$ nodes.

TABLE I
AVERAGE COMPUTATIONAL TIME (SECONDS) NEEDED TO SOLVE THE 5 INSTANCES (FOR EACH $N$) OF THE PROBLEM BY THE CPLEX SOLVER CONSIDERING DIFFERENT ESTIMATIONS OF K.

| | *CPLEX Solver* | | |
|---|---|---|---|
| $N$ | $T_{\hat{K}}$ | $T_{\hat{K}+2}$ | $T_{\hat{K}+4}$ |
| 7 | 0.125 | 0.197 | 0.301 |
| 8 | 0.145 | 0.238 | 0.280 |
| 9 | 5.471 | 22.653 | 70.873 |
| 10 | 270.9 | 2136.7 | >3600 |

as it is in a realistic case, solving the problem with CPLEX requires to overestimate $K$ which yields further complexity. With just a small overestimation (by 4 units), CPLEX can be more than one order of magnitude slower. Despite the fact that our proposed algorithm and CPLEX requires different inputs (e.g., $K$ is needed for CPLEX), we believe that this comparison is useful to understand the order of magnitude of the time required to solve the problem using our approach and a model based on integer linear programming.

Table II presents the average computational times needed to solve all the instances using respectively the normal ($T_{nv}$) and the advanced ($T_{av}$) version of the proposed algorithm. Our algorithm, in particular the advanced one, requires a limited amount of seconds even for relatively large instances of the problem, as shown in Table II. Also, note that our algorithm, to speed up development, has been implemented in the *python* scripting language, which is not particularly optimized for speed. Therefore it could potentially be made faster by rewriting it in, e.g., a language that can compile to native code. Therefore, it can be concluded that the proposed algorithm is better suited for this type of problems because it is relatively fast as well as it does not require to guess the $K$ value that yields to the optimal solution.

## VI. CONCLUSIONS

In this work we proposed a novel integer linear programming model aiming at scheduling the activities of an operator that needs to perform repeated acquisition, elaboration/communication and action tasks. We propose a new problem formulation that is also general enough to be applied to other industrial scheduling problems or logistics problems

TABLE II
AVERAGE COMPUTATIONAL TIME (SECONDS) NEEDED TO SOLVE THE PROBLEM BY THE NORMAL ($T_{nv}$) AND ADVANCED ($T_{av}$) VERSIONS OF THE PROPOSED ALGORITHM.

| | *proposed* | |
|---|---|---|
| $N$ | $T_{nv}$ | $T_{av}$ |
| 7 | 0.002 | 0.001 |
| 8 | 0.005 | 0.003 |
| 9 | 0.03 | 0.01 |
| 10 | 0.19 | 0.03 |
| 11 | 0.87 | 0.12 |
| 12 | 2.67 | 0.36 |
| 13 | 9.41 | 1.15 |
| 14 | 40.26 | 3.56 |
| 15 | 124.43 | 11.34 |
| 16 | 377.11 | 35.47 |

of, e.g., repair companies. After highlighting the difficulties of finding the solution of such complex problems using traditional solvers, we tackled such difficulties by proposing a new algorithm specifically tailored the the problem characteristics. Results showed that the proposed algorithm is relatively fast in finding the optimal solution also on actual problems extracted from a real world case. Future work will focus on deriving effective heuristics from the proposed exact algorithm that could be used to efficiently solve larger instances of the proposed model.

## VII. Acknowledgements

This work has been supported in part by the Politecnico di Torino Interdepartmental Center for Service Robotics (PIC4SeR) https://pic4ser.polito.it.

## References

[1] E. Fadda, R. Tadei, G. Perboli, and L. F. Tiotsop, "The multi-path traveling salesman problem with dependent random cost oscillations," in *Seventh Intl. Workshop on Freight Transportation and Logistics (ODYSSEUS)*, Cagliari, Italy, Jun. 2018, pp. 368–371.

[2] B. Eksioglu, A. Volkan. Vural, and A. Reisman, "The vehicle routing problem: A taxonomic review," *Computers & Industrial Engineering*, vol. 57, pp. 1472–1483, 2009.

[3] A. Bechar and C. Vigneault, "Agricultural robots for field operations: Concepts and components," *Biosystems Engineering*, vol. 149, pp. 94 – 111, 2016.

[4] C. Zecha, J. Link, and W. Claupein, "Mobile sensor platforms: Categorisation and research applications in precision farming," *Journal of Sensors and Sensor Systems*, vol. 2, no. 1, pp. 51–72, 2013.

[5] S. Candiago, F. Remondino, M. De Giglio, M. Dubbini, and M. Gattelli, "Evaluating multispectral images and vegetation indices for precision farming applications from UAV images," *Remote Sensing*, vol. 7, no. 4, pp. 4026–4047, 2015.

[6] V. Lukas, J. Novák, L. Neudert, I. Svobodova, F. Rodriguez-Moreno, M. Edrees, and J. Kren, "The combination of UAV survey and landsat imagery for monitoring of crop vigor in precision agriculture," *ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 41, pp. 953–957, 2016.

[7] A. Castellini, A. Farinelli, G. Minuto, D. Quaglia, I. Secco, and F. Tinivella, "EXPO-AGRI: Smart automatic greenhouse control," in *IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 2017, pp. 1–4.

[8] J. Dong, J. G. Burnham, B. Boots, G. Rains, and F. Dellaert, "4D crop monitoring: Spatio-temporal reconstruction for agriculture," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3878–3885.

[9] A. Chang, J. Jung, M. M. Maeda, and J. Landivar, "Crop height monitoring with digital imagery from unmanned aerial system (UAS)," *Computers and Electronics in Agriculture*, vol. 141, pp. 232–237, 2017.

[10] S. Khanal, J. Fulton, and S. Shearer, "An overview of current and potential applications of thermal remote sensing in precision agriculture," *Computers and Electronics in Agriculture*, vol. 139, pp. 22–32, 2017.

[11] L. Comba, P. Gay, J. Primicerio, and D. R. Aimonino, "Vineyard detection from unmanned aerial systems images," *Computers and Electronics in Agriculture*, vol. 114, pp. 78–87, 2015.

[12] J. Primicerio, G. Caruso, L. Comba, A. Crisci, P. Gay, S. Guidoni, L. Genesio, D. Ricauda Aimonino, and F. P. Vaccari, "Individual plant definition and missing plant characterization in vineyards from high-resolution UAV imagery," *European Journal of Remote Sensing*, vol. 50, no. 1, pp. 179–186, 2017.

[13] A. Matese and S. F. Di Gennaro, "Technology in precision viticulture: A state of the art review," *International Journal of Wine Research*, vol. 7, pp. 69–81, 2015.

[14] J. Das, G. Cross, C. Qu, A. Makineni, P. Tokekar, Y. Mulgaonkar, and V. Kumar, "Devices, systems, and methods for automated monitoring enabling precision agriculture," in *IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE, 2015, pp. 462–469.

[15] W. Pei, Y. Lan, L. Xiwen, Z. Zhiyan, Z. Wang, and Y. Wang, "Integrated sensor system for monitoring rice growth conditions based on unmanned ground vehicle system," *International Journal of Agricultural and Biological Engineering*, vol. 7, no. 2, p. 75, 2014.

[16] H. Seyyedhasani and J. S. Dvorak, "Using the vehicle routing problem to reduce field completion times with multiple machines," *Computers and Electronics in Agriculture*, vol. 134, pp. 142 – 150, 2017.

[17] ——, "Dynamic rerouting of a fleet of vehicles in agricultural operations through a dynamic multiple depot vehicle routing problem representation," *Biosystems Engineering*, vol. 171, pp. 63 – 77, 2018.

[18] D. Bochtis and C. Sørensen, "The vehicle routing problem in field logistics: Part II," *Biosystems Engineering*, vol. 105, no. 2, pp. 180 – 188, 2010.

[19] K. Zhou, A. L. Jensen, C. Sørensen, P. Busato, and D. Bothtis, "Agricultural operations planning in fields with multiple obstacle areas," *Computers and Electronics in Agriculture*, vol. 109, pp. 12 – 22, 2014.

[20] F. Nex and F. Remondino, "UAV for 3D mapping applications: a review," *Applied geomatics*, vol. 6, no. 1, pp. 1–15, 2014.

[21] J. Torres-Sánchez, J. M. Peña, A. I. de Castro, and F. López-Granados, "Multi-temporal mapping of the vegetation fraction in early-season wheat fields using images from UAV," *Computers and Electronics in Agriculture*, vol. 103, pp. 104–113, 2014.

[22] G. Sona, D. Passoni, L. Pinto, D. Pagliari, D. Masseroni, B. Ortuani, and A. Facchi, "UAV multispectral survey to map soil and crop for precision farming applications," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 41, pp. 1023–1029, 2016.

[23] J. Primicerio, G. Caruso, L. Comba, A. Crisci, P. Gay, S. Guidoni, L. Genesio, D. Ricauda Aimonino, and F. P. Vaccari, "Individual plant definition and missing plant characterization in vineyards from high-resolution UAV imagery," *European Journal of Remote Sensing*, vol. 50, no. 1, pp. 179–186, 2017.

[24] W. E. Wilhelm, "A technical review of column generation in integer programming," *Optimization and Engineering*, vol. 2, no. 2, pp. 159–200, Jun 2001.

[25] Y. Zhao, T. Larsson, E. Rönnberg, and P. M. Pardalos, "The fixed charge transportation problem: a strong formulation based on lagrangian decomposition and column generation," *Journal of Global Optimization*, vol. 72, no. 3, pp. 517–538, Nov 2018.

[26] IBM, "CPLEX." [Online]. Available: https://www.ibm.com/products/ilog-cplex-optimization-studio