Scheduling Contractors’ Farm-to-Farm Crop Harvesting Operations

C B Basnet *
L R Foulds *
J M Wilson #

University of Waikato
New Zealand

# Loughborough University
United Kingdom

Abstract

The harvesting of renewable resources from an operations scheduling viewpoint is introduced and a harvesting scenario arising in the agricultural context involving a commercial contracting enterprise that travels from farm to farm harvesting crops is discussed. The paper is an extension of previous work by two of the authors from the one-farm to the multi-farm case. In both cases the duration of each operation is dependent upon the combination of constrained resources allocated to it, equipment and worker allocation is restricted, and minimum or maximum time lags on the start and completion of operations may be imposed. The present case incorporates harvesting at more than one farm and thus the sequence in which the farms are visited and the inter-farm travel times must be taken into account. We report on a harvesting scheduling model and solution procedures designed specifically for large-scale versions of the multi-farm case. The computational times experienced in solving general instances of the model of small to medium practical size by a commercial integer programming package are encouraging. Greedy and tabu search heuristics which are capable of solving problems of relatively large dimensions in reasonable computational time are also included. The authors believe that the model and the solution techniques developed represent a useful addition to the farm crop contractor’s tool kit.

Key Words Harvesting, contractors, farm-to-farm, operations scheduling, integer programming, tabu search.
1. Introduction

The focus of this paper is on the scheduling of the operations that are carried out by contractors who harvest crops at various farms. The duration of each operation is dependent upon the combination of constrained resources allocated to it. Also, equipment and worker allocation is restricted. Further, minimum or maximum time lags on the start and completion of operations may be imposed. The present case incorporates harvesting at more than one farm and thus inter-farm travel times must also be taken into account. As far as the authors are aware, none of the solution techniques reported to date in the open literature appear to be applicable to large-scale numerical instances of the harvesting scenario considered in the present paper. This is because the scenario analysed here involves operation durations that are dependent upon the resources allocated to them, farm-to-farm travelling times, and time lags.

Nearly all harvesting processes have resource constraints because of limits on costs, time, worker skill levels, and machines. Much project scheduling literature focuses largely on timing issues without taking into account the link between resource availability and the process schedule. The fact that some resources might not be available in sufficient quantities when desired will almost certainly affect the final schedule. Operation duration delays often occur at critical times because some mix of labour or machines is not available at sufficient levels for some operations to be carried out within the time period dictated by the adopted schedule. Therefore, if these resources are not managed properly costs may increase due to harvesting completion delays.

The presence of a minimum time lag for a given operation implies that the operation cannot start before a specified time has elapsed after the completion of one of its predecessor operations. The presence of a maximum time lag for a given operation implies that an operation must be started at the latest some number of time periods after the completion of one of its predecessor operations. The set-up times for various types of machines and the drying and inspection time necessary between some pairs of operations sometimes create minimum or maximum time lags in any feasible schedule in the harvesting scenario under study. Time lags may exist in the form of imposed constraints due to technological considerations. In other cases they arise naturally in the scheduling process similar to activity slack time in project management. Minimum time lags may arise for instance due to the existence of compulsory release dates or ready times. Maximum time lags may arise for instance if operations have to be carried out in sequence without delay or due to operation completion deadlines that must be met to achieve overall process duration minimization. The presence of a combination of both minimum and maximum time lags for a
given pair of operations result in a time window with a conditional start time related to the completion time of a preceding operation. The importance of time lags is underlined via a numerical example given later.

The remainder of this paper is organized as follows. Section 2 discusses the scheduling of farm-to-farm harvesting operations in general and Section 3 reviews the current relevant scheduling literature. Sections 4, 5, and 6 present respectively an integer programming model for the farm-to-farm contractors’ harvesting operations scheduling problem with constrained resources, a report on its application and related solution procedures and our conclusions.

2. Scheduling Operations for Farm-to-farm Crop Harvesting

2.1 Problem definition and assumptions

The scenario to be studied involves a contracting company that travels from farm to farm to harvest crops. The primary objective of the harvesting process to be discussed in this paper is to specify the activity of each actual machine of each type and its crew in each time period in order to minimize the duration of the entire harvesting process at a given set of farms. Thus, set up, processing and inter-farm travel times must be taken into account. However, in many harvesting operations scheduling scenarios such as the present one the paucity of dedicated machines and trained crews capable of carrying out some operations often give rise to binding constraints on the duration of the overall harvesting process. It is assumed that

- The set of farms to be visited is known,

- All inter-farm travel times are known,

- The costs of the various operations will not affect the order in which the farms are visited, nor the order in which the operations are carried out at each farm (which is the same for all farms),

- Each operation can be performed by only one type of machine. (That is, any machine of a given type can perform exactly one type of operation. Thus, the number of machine types is equal to the number of operations.),

- The inter-farm transportation of any actual machine may involve the skipping of some farms. That is, each physical machine is not necessarily transported to each farm, and

- For each farm to be visited
The harvesting operations are performed in a given sequence, one at a time,

- The set up of all machines of each type that are actually utilized on a farm is carried out simultaneously, without interruption,

- While any operation (apart from the last) is being performed, the machines to be used for its succeeding operation can either be set up at the same time or once the operation is completed,

- Each operation must be started immediately after the completion of the set up of its associated machines, and

- Each operation must be performed, from start to completion, without interruption.

In certain cases the weather can affect the duration of operations and the time lags between operations. However, the effects of the weather are neglected in the model to be constructed in this paper. This is because the practical case study in which two of the authors were involved has relatively stable, warm, dry weather year-round which is reasonably common in harvesting scenarios concerning wheat and similar crops in countries such as Australia. It is the experience of the authors that in such situations the exclusion of weather effects has little or no affect on the validity of the model.

The input data needed for the analysis of a harvesting scheduling scenario include the number of machines and their crews that are available to perform each operation, the set-up time for each type of machine, all inter-farm travel times, the sequence of the operations to be performed, and for each farm, the imposed time lags between operations and the resource level allocation-operation duration relationships. (That is, how much if at all, operation processing times are reduced by the allocation of additional machines.) The matters just mentioned are made more precise in the model given in Section 4.

### 2.2 An example of farm-to-farm crop harvesting

Rape seed is often harvested to be crushed to produce canola oil. In order to harvest the rape seed a normal harvesting process involves the operations of swathing and then threshing. The meaning of these terms will become self-evident in the subsequent discussion. Each harvesting machine has a dedicated crew who set it up and then use it to perform the associated operation. The time lags imposed indicate minimum or maximum gaps in time that must occur between the end of swathing and the start of threshing. For each operation although machine set up time is constant processing time is dependent upon the number of machines used.
The harvesting starts with swathing in which the rape seed plants are cut down and left in the field in which they were grown. The swathed rape seed then requires drying which involves rotating the swathed plants lying on the ground. Finally, the rape seed is threshed and transferred into containers to be transported for either storage or processing. The fact that the swathed crop must be dried between swathing and threshing often induces minimum and maximum times lags between these operations. This is because the swathed crop must be dried for a minimum time to allow effective threshing but can be dried for only up to a maximum time to prevent rotting. The set-up time for each type of machine has to be established which involves start up, warm up, adjustment and lubrication. Further, the duration of each operation is dependent on field size and the number of machines allocated to it. For example, swathers and threshers of average capacity are capable of swathing and threshing 15 and 25 hectares of rape seed per hour respectively. The literature related to the scheduling scenarios similar to the one just described is now reviewed.

3. A Literature Survey

The review of the literature is divided into sections on the key elements of the farm crop harvesting scenario under study, namely resource constrained project scheduling, minimum and maximum time lags and resource leveling. Also, because it is somewhat similar to farm crop harvesting, work on forestry stand harvesting scheduling is also reviewed.

3.1 Resource constrained project scheduling

There is literature available on the use of traditional project scheduling models for scheduling activities in a resource-constrained environment by according priorities to different activities (see for example, Moder et al. [1983]). Approaches to scheduling resource-constrained projects comprise exact procedures (often based on integer programming) and heuristic procedures. A number of branch and bound algorithms have been proposed for various resource-constrained scheduling problems including those by Bartusch et al. [1988], Dorndorf et al. [2000] and Moehring et al. [1998]. Reviews of many of these methods have been provided by Brucker et al. [1999].

3.2 Minimum and maximum time lags and resource levelling

Some of the literature referenced above deals with minimum time lags (Bartusch et al. [1988] and Brucker et al. [1999]). There is less literature available on the additional requirement of maximum time lags than there is on minimum time lags. An exact procedure has been devised by De Reyck and
Herroelen [1998] though it does not appear capable of solving problems of practical size in reasonable computational time. Resource levelling involves attempting to minimize the fluctuations in requirements for resources so that they will be utilized as uniformly as possible throughout the process by rescheduling the harvesting operations to achieve more even usage of their resource requirements and to avoid resource usage peaks and valleys. Resource levelling techniques attempt to keep the usage of each resource within feasible limits so as to avoid unnecessary fluctuations in resource usage over time. Resource levelling often lengthens the overall harvesting completion time. As with resource constrained scheduling both algorithms and heuristics have been reported for resource levelling. Methods that aim to produce optimal schedules have been constructed by Easa [1997], Bandelloni et al. [1994], Demeulemeester [1995], Younis and Saad [1996] and Savin et al. [1996]. The last technique is based on a neural network model. Heuristic methods based on either simple shifting or priority rules have been reported by Moder et al. [1983], Harris [1990] and Takamoto et al. [1995]. This last paper reports on a large-scale case study. A model and solution procedures for large-scale numerical instances of the particular farm-to-farm harvesting scenario described earlier is developed in Section 4.

3.3 Forest stand harvesting scheduling

Scheduling of commercial forest stand harvesting operations has been the focus of some research (see for example, Kirkby et al. [1980], Manley et al. [1991], Nelson et al. [1991], Weintraub et al. [1994], McNaughton [1995], Murray and Church [1995] and McNaughton et al. [1998].). However the models, techniques and software just reported are mainly concerned with considerations that are particular to forest harvesting including adjacency or catchment constraints. However, there are a number of factors arising in farm crop harvesting that cannot be easily addressed by these approaches including maximum time lags, operation interdependency, time and activity-dependent criticality, conflicting priorities, and the mutual exclusivity, partial allocation, sharing and the substitution of resources.

An attempt to address some of these issues by reporting on exact and approximate methods for crop harvesting at a single farm has recently been made by Foulds and Wilson [2005]. The authors discuss the harvesting of farm crops from an operations research scheduling viewpoint. They report on harvesting operations scheduling models and solution procedures designed specifically for an actual case study. The results represent significant improvements over the schedules that were traditionally used. The computational times experienced in solving general instances of the model of practical size by a commercial integer programming package are encouraging.
However, as the authors of the present paper are unaware of any approaches available in the open literature that are directly relevant to the multi-farm problem under study they believe that there is much scope for the development of project scheduling techniques that are effective for this scenario. The present paper represents an extension of the techniques developed in Foulds and Wilson [2005] to large-scale instances of the multi-farm case. A model of this scenario is now developed.

4. Operations Scheduling for Harvest Duration Minimization

4.1 Development of the model

The description of the model is begun by introducing the necessary notation. Later in this section various solution procedures that were developed for the model are reported.

**Parameters**

- $H =$ The total number of farms to be serviced,
- $n =$ the number of machine types/operations,
- $m_i =$ the number of individual machines of type $i$ available, $i = 1, 2, ..., n,$
- $(1, 2, \ldots, n) =$ the sequence in which the operations must be performed (the same at all farms),
- $M =$ A relatively large constant.

**Input data**

- $q_{gh} =$ the time to transport any machines from farm $g$ to farm $h$; $g, h = 1, 2, \ldots, H,$
- $d_{ih}^{\text{min}} =$ the minimum time lag between operations $i$ and $i+1$ at farm $h$; $h = 1, 2, \ldots, H,$
- $d_{ih}^{\text{max}} =$ the maximum time lag between operations $i$ and $i+1$ at farm $h$; $h = 1, 2, \ldots, H,$
- $p_{ijh} =$ the time needed to carry out operation $i$ at farm $h$ if $j$ machines of type $i$ are used,
- $t_i =$ the set up time of machines of type $i$.

**Decision Variables**

- $s_{ih} =$ the start time of operation $i$ at farm $h, i = 1, 2, \ldots, n; h = 1, 2, \ldots, H,$
- $e_{ih} =$ the completion time of operation $i$ at farm $h, i = 1, 2, \ldots, n; h = 1, 2, \ldots, H,$
\[ x_{ijgh} = \begin{cases} 1, & \text{if machine } j \text{ of type } i \text{ is sent from farm } g \text{ to } h; \\ 0, & \text{otherwise}, \ i = 1, 2, \ldots, n; j = 1, 2, \ldots, m_i; g, h = 1, 2, \ldots, H; g \neq h, \end{cases} \]

\[ y_{ijh} = \begin{cases} 1, & \text{if } j \text{ machines of type } i \text{ are used at farm } h; \\ 0, & \text{otherwise}, \ i = 1, 2, \ldots, n; j = 1, 2, \ldots, m_i; h = 1, 2, \ldots, H. \]

\[ Z = \begin{cases} \text{The objective function value.} \end{cases} \]

A dummy farm, \( h = 0 \), is introduced. Initially all machines are stored at farm zero, but can be moved instantly to any farm.

**The Model**

**Objective**

Minimize \[ Z \quad (4.1) \]

**Constraints**

\[ Z \geq e_{nh} \quad h = 1, 2, \ldots, H \quad (4.2) \]

(The makespan is no smaller than the end time of the last operation at all farms)

\[ \sum_{h=1}^{H} x_{ij0h} = 1, \ i = 1, 2, \ldots, n; j = 1, 2, \ldots, m_i \quad (4.3) \]

(All machines are sent from farm zero to some farm.)

\[ \sum_{g=1}^{H} x_{ijgh} - \sum_{l=1}^{H} x_{ijlh} \geq 0, \ i = 1, 2, \ldots, n; j = 1, 2, \ldots, m_i; h = 1, 2, \ldots, H \quad (4.4) \]

(Machines sent from farm \( h \) must have arrived there previously.)

\[ e_{ig} - s_{ih} + (q_{gh} + t_i + M)x_{ijgh} \leq M, \ i = 1, 2, \ldots, n; j = 1, 2, \ldots, m_i; g = 1, 2, \ldots, H; h = 1, 2, \ldots, H, \quad (4.5) \]

(The start time of an operation at a farm is determined by the completion time of this operation at farms that send the associated machines to it.)

\[ \sum_{j=1}^{m_i} \sum_{g=0}^{H} x_{ijgh} - \sum_{j=1}^{m_i} y_{ijh} \geq 0, \ i = 1, 2, \ldots, n; h = 1, 2, \ldots, H \quad (4.6) \]
(The number of machines type $i$ available for use at farm $h$ cannot exceed the number that have arrived at farm $h$.)

\[ e_{ih} - s_{ih} - \sum_{j=1}^{m_i} p_{ijh} y_{ijh} \geq 0 \quad i = 1, 2, \ldots, n; \quad h = 1, 2, \ldots, H. \]  

(4.7)

(The completion time of an operation at a farm is determined by its start and processing times.)

\[ s_{i+1,h} - e_{ih} - d_{ih}^{\text{min}} \geq 0, \quad i = 1, 2, \ldots, (n-1); \quad h = 1, 2, \ldots, H. \]  

(4.8)

(The start time of each operation must allow for the minimum time lag after the completion of the previous operation.)

\[ s_{i+1,h} - e_{ih} - d_{ih}^{\text{max}} \leq 0, \quad i = 1, 2, \ldots, (n-1); \quad h = 1, 2, \ldots, H. \]  

(4.9)

(The start time of each operation must allow for the maximum time lag after the completion of the previous operation.)

\[ \sum_{j=1}^{m_i} y_{ijh} = 1, \quad i = 1, 2, \ldots, n; \quad h = 1, 2, \ldots, H \]  

(4.10)

(A positive number of machines must be used for each operation at each farm.)

4.2 Development of solution procedures from the model

It is clear that the above model expressed by (4.1)-(4.10) is an integer programme (IP) containing mainly zero-one variables. Numerical examples of the size normally encountered in practice (typically up to six farms and three operations) can be solved to optimality using current commercial IP codes. However, relatively large numerical instances are beyond the capability of current IP methods and require heuristic techniques. To this end, heuristics based on a simple greedy strategy and on Tabu Search (TS) have been developed. TS is a general improvement metaheuristic procedure for solving optimization problems which is designed to escape the trap of local optimality by applying a local search procedure at each step of a general iterative search process. For an introduction to TS the reader is referred to the book by Glover and Laguna [1997].

TS was chosen as the general approach to developing an effective heuristic for the problem at hand for the following reasons. It has been used successfully to solve a wide variety of problems including the
quadratic assignment problem (Taillard [1991]), job-shop scheduling (Dell’Amico and Trubian]) and vehicle routing (Semet and Taillard [1993] and Taillard et al. [1997]). Another adaptation of TS (to a particular supply chain management problem) has been reported by Rochat and Semet [1994] who created a TS list by devising a novel relaxation of constraints and also employed an interesting intensification strategy. As the model of Rochat and Semet has similarities to that of the model in Section 4.1 above, the authors of the present article have made significant modifications to Rochat and Semet’s procedure so that it is suitable for the present model.

A Greedy Heuristic

For this heuristic the farms are renumbered so that the sequence $h = 1, 2, ..., H$, corresponds to a Traveling Salesman Problem (TSP) path through all farms. Given the dimensions of numerical problems commonly encountered in practice the farthest insertion procedure can be used for this purpose in reasonable computational time. The reader is referred to a description of the TSP and solution procedures for it by Applegate et al. [1998]. The first step of the greedy heuristic is to set the number of machines assigned to each operation $i$, $i = 1, 2, ..., n$, at each farm, to be equal to $m_i$, the maximum number of machines available.

Algorithm

For each farm $h$, in the order $h = 1, 2, ..., H$

For each operation $i$, in the order $i = 1, 2, ..., n$

Allocate the maximum available number of machines of type $i$.

Set up the allocated machines and have them begin operation $i$ at the earliest feasible time.

If $h < H$

As soon as operation $i$ is completed, transport all of the machines involved to farm $h + 1$.

Next $i$

Next $h$

The TS procedure is now explained in terms of the numerical problem introduced earlier, but generalization to generic problems is straightforward.

A Tabu Search Procedure

The order in which the farms (denoted by the indices 1, 2, ..., $H$) are scheduled is termed the scheduling order and is denoted by $S$, where

$S = \{I_1, I_2, ..., I_H\}$, such that when $I_k = h$, farm $h$ is the $k^{th}$ farm to be scheduled, $h = 1, 2, .. , H$.

In generating a schedule farms are considered in the order specified in $S$. The following is also defined:

$u_{ih} =$ the number of machines allocated to farm $h$ for operation $i$, $i = 1, 2, ..., n; h = 1, 2, ..., H$; and

$U = (u_{ih})_{n \times H}$. 

The objective function $Z(S, U)$ is the makespan corresponding to a scheduling order $S$ and a machine allocation regime $U$. That is, the completion time of the last operation on the last farm according to $S$ and $U$. For simplicity $Z(S, U)$ is denoted by $Z$.

Schedule generation

Given a scheduling order $S$ of farms and an allocation of machines $U$ to all farms for all operations a non-delay schedule can be developed in the following manner.

1. A Gantt chart is built up chronologically. It displays the activity (set up, operation, idle or transport) of each physical machine in each time period starting from time zero.

2. In allocating the machines, farms are considered in the order that is given in $S$, the scheduling order. That is, a farm $I_k$ is scheduled with respect to all the $n$ operations. Then the next farm $I_{k+1}$ in $S$ is considered for scheduling. For each operation $i$ at a farm $h$ all the individual machines for that operation are listed in order of their feasible earliest arrival time at the farm. The first $u_{ih}$ machines from this list are then allocated to farm $h$ and are scheduled to be transported to farm $h$ without delay after their current operation.

3. Setting up and operation of a particular machine in a farm can start only after all the allocated number of machines of that type has arrived at a farm. The transportation, set up and operation is carried out without inserting delays. Delays are inserted only to the extent necessary to enforce the minimum time lags.

4. If an operation is delayed beyond the maximum time lag because the machines for that operation are not available in time the earlier operations for that farm are delayed to the required extent so that the maximum time lag constraint is satisfied. This may have a follow-on effect on the schedule of earlier operations at the farm.

5. The schedule is built up, one farm at a time, in the scheduling order $S$, and within a farm, one operation at a time, with respect to the technological constraints. This chronological build-up of the schedule continues until all the operations of all the farms are completed.
The search space

Given a scheduling order of farms $S$ and a particular allocation of machines to farms $U$ it is straightforward to show that the above schedule generation procedure always results in a feasible schedule. The search space in our TS search is the set of all schedules generated by the set of scheduling orders of farms and the set of machine allocations. A move transforms an element of the search space to another element. In the TS procedure there are the following two possible moves. At each iteration of the search all the moves (of both types) are evaluated and the move with the best objective function is selected for execution.

1. A change in the scheduling order so that two farms which are next to each other in $S$ swap their priorities. Only farms next to each other ($I_k, I_{k+1}$) are considered for a swap. The total number of moves of this type to be considered at each iteration is $(H - 1)$.

2. A change in the allocation of machines for an operation in a farm so that the number of machines allocated to a particular operation ($u_{ih}$) increases or decreases by exactly one. This allocation must be at least one and at most it can be the total number of machines for that operation. Since an allocation can go up or down the maximum number of moves to be considered at each iteration is $2nH$.

The TS procedure

As all the generated schedules are always feasible there are no penalty parameters in the TS procedure. The procedure is carried out in the normal TS manner, examining all the possible moves, and making the move that yields the lowest objective function until a given number of iterations have been performed. The particular $S$ and $U$ that yield the lowest overall objective over the entire search (and the associated schedule) are returned as the best solution. Any move that is implemented during the search is placed in a tabu list. Moves included in this list are prohibited unless such a move actually improves the overall objective. The tabu list is a first-in-first-out list: as new moves are added to the list, earlier moves become available for implementation. The length of the tabu list is updated dynamically. If the current objective decreases after a move, the length of the tabu list is incremented by one; the length is decremented by one otherwise. However, the length is bound by set upper and lower limits. An intensification strategy is also followed, which is described below. In evaluating a move the schedule does not always need to be generated *ab initio* – if a move involves farm $I_k$, only the schedule of farms $I_k, I_{k+1} ...I_H$ need to be
regenerated. The TS procedure requires as input an initial solution giving the sequence in which the farms are visited and the number of machines to be allocated to perform each operation at each farm. If no better input is known the output of applying the greedy heuristic described above can be used.

The intensification strategy

The intensification strategy comprises a prohibition of all moves associated with particular farms. When the objective does not improve for a given number of iterations, farms are randomly partitioned into sub-sets of two farms (that are not necessarily adjacent) and moves associated with one sub-set are prohibited for a fixed number of iterations. The prohibition is applied to all the sub-sets turn by turn. The intensification strategy is stopped as soon as the objective improves. If the objective does not improve, the partition size is increased to 4, 6, 8,…, and so on, until the partition size exceeds half the number of farms. If the objective still does not improve the normal TS procedure is resumed.

The solution techniques previously developed is now applied to a numerical problem.

5. Application of the Solution Techniques

5.1 An illustrative problem

The solution techniques described in the previous section were applied to the following numerical problem with input data (in hours) given in Tables 1, 2, and 3.

Tables 1, 2, and 3 about here.

Following the notation defined in the previous section the following parameters are set

\[ n = 2, \text{ (the number of operations)}, \]
\[ m_1 = 3, m_2 = 3, \text{ (the number of machines/crews of type 1 and type 2, respectively)}, \]
\[ H = 3, \text{ (the number of farms)}, \]
Swathing ~ operation 1,
Harvesting ~ operation 2,
The operation precedence order \(<1, 2>\), and
\[ q_{12} = 1, q_{23} = 8, q_{13} = 7 \text{ (the inter-farm transport times)}. \]
Initially, all machines are located at farm one. Although these data are fictitious, they are similar to the actual data. The above methods were applied to the numerical instance of the model with the above input data. The resulting schedules are displayed in Figures 1, 2, and 3.

The greedy heuristic, described earlier, was applied to the numerical problem and produced the solution shown in Figure 1. The TS procedure described earlier was also applied. The initial solution for the TS procedure was the solution produced by the greedy heuristic. Figure 2 shows an intermediate solution found by the TS procedure. The final solution of the TS procedure is shown in Figure 3. The IP code also generated the same (optimal) solution.

**Figures 1, 2, and 3 about here.**

### 5.2 Testing the heuristics

The IP approach can solve problems involving up to 6 farms and 3 operations fairly rapidly but becomes cumbersome on larger problems. Hence the greedy heuristic and the TS procedure were programmed in Microsoft Visual C++ run on a PC with 3.2 GHZ and 504 MB RAM and tested on larger numerical instances. Ten problems were generated for \((H, n)\) combinations of \(H = 3\) to 20; and \(n = 2\) to 5. The smaller problems were also solved by the IP code, running on a Sun Solaris computer.

**Problem generation**

Problems were randomly generated with the following distributions

- Setup time uniform between 2 and 5,
- Number of machines for an operation uniform between 2 and 4,
- Minimum time lag uniform between 3 and 5,
- Maximum time lag uniform between 5 and 10.
- One-machine-operation durations were uniformly distributed between 2 and 15. The operation time for using a number of machines was generated by dividing the duration by the number of machines.
- Farm locations were generated on a Cartesian co-ordinates grid with the distance from the origin being distributed uniformly between 0 and 10 along both axes. Distances between farms were calculated as the Euclidean distance.

The solution found by the greedy heuristic was used as the input initial solution for the TS procedure. The number of iterations was set to be equal to \((\text{the total number of machines}) \times (\text{the number of farms}) \times (10)\). The minimum size of tabu list was set at \((\text{the total number of machines}) \times (\text{the number of farms}) / (3)\), subject to a minimum of 10. The maximum size was set at \((\text{the total number of machines}) \times (\text{the number of machines})\).
farms)/(2.5); the intensification strategy was turned on when the objective did not improve in (the total number of machines)*(the number of farms)*(2) iterations and was discontinued after (the total number of machines)*(the number of farms)/(5) iterations.

The results are illustrated in Table 4.

**Table 4 about here**

As can be seen in Table 4, the search space for the IP code quickly gets too large as the problem size increases. The TS procedure is able to improve on the greedy heuristic to the extent of about 10% of the makespan at a cost of significantly higher (but still modest) run time. Obviously the tabu search takes longer run time since it needs to assess all the moves at each iteration of the search. The larger problems require about 26 seconds each for the TS procedure. The paper is now summarized below and some conclusions are presented.

**6. Summary and Conclusions**

The scheduling of contractors’ farm-to-farm crop harvesting operations has been discussed and it has been concluded that the scheduling of such harvesting operations is a significantly different scenario from those represented by the scheduling models currently available in the literature. The differences come about because the duration of each operation is dependent upon the combination of constrained resources allocated to it, equipment allocation is restricted, minimum or maximum time lags on the start and completion of operations can be imposed and the fact that inter-farm travel times must be taken into account. An illustrative problem was discussed involving the harvesting of rape seed and a model of the rape seed harvesting scenario was developed. A greedy heuristic and a tabu search procedure were developed from the model. A commercial integer programming code as well as these heuristics were applied to the data gathered for the illustrative problem to construct harvesting operations schedules. Furthermore, the computational times experienced in solving general instances of the model of relatively large dimensions by the tabu search procedure are encouraging. Thus, because one of the authors used the IP model to assist a group of farms and enabled the clients to facilitate planning for the first time in an efficient manner by providing a decision support system, the authors believe that the models and procedures reported represent useful additions to the harvesting contractor’s toolkit.

**References**


<table>
<thead>
<tr>
<th>Operation</th>
<th>Swathing</th>
<th>Threshing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machine Availability</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Set up Times</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

**Processing Times**

<table>
<thead>
<tr>
<th>Farm 1</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Machine</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>2 Machines</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>3 Machines</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Farm 2</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Machine</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>2 Machines</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>3 Machines</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Farm 3</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Machine</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>2 Machines</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>3 Machines</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

*Table 1 Swathing and threshing parameters*
<table>
<thead>
<tr>
<th>Farm</th>
<th>Minimum Time Lags</th>
<th>Maximum Time Lags</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

**Table 2 The time lags between swathing and threshing**

<table>
<thead>
<tr>
<th>Farm-to-Farm</th>
<th>1 to 2</th>
<th>2 to 3</th>
<th>1 to 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport Time</td>
<td>1</td>
<td>8</td>
<td>7</td>
</tr>
</tbody>
</table>

**Table 3 The inter-farm transport times**

<table>
<thead>
<tr>
<th>Problem size (number of farms, number of operations)</th>
<th>IP</th>
<th>Greedy Heuristic</th>
<th>Tabu Search</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average makespan</td>
<td>Average Number of nodes</td>
<td>Averagemakespan</td>
</tr>
<tr>
<td>3, 2</td>
<td>28.7</td>
<td>54.1</td>
<td>37.3</td>
</tr>
<tr>
<td>5, 3</td>
<td>31.3</td>
<td>8420.2</td>
<td>70.4</td>
</tr>
<tr>
<td>6, 3</td>
<td>42.8</td>
<td>6386632</td>
<td>77.0</td>
</tr>
<tr>
<td>10, 4</td>
<td>*</td>
<td>127.0</td>
<td>0</td>
</tr>
<tr>
<td>10, 5</td>
<td>*</td>
<td>142.8</td>
<td>0</td>
</tr>
<tr>
<td>15, 4</td>
<td>*</td>
<td>168.0</td>
<td>0</td>
</tr>
<tr>
<td>15, 5</td>
<td>*</td>
<td>178.1</td>
<td>0</td>
</tr>
<tr>
<td>20, 4</td>
<td>*</td>
<td>209.2</td>
<td>0</td>
</tr>
<tr>
<td>20, 5</td>
<td>*</td>
<td>222.0</td>
<td>0</td>
</tr>
</tbody>
</table>

*A solution could not be found within 3 hours, the cut-off time*

**Table 4 Comparison of the results produced by the IP code and the heuristics**
Figure 1  The solution generated by the greedy heuristic (and initial solution for the tabu search procedure)

Figure 2  An intermediate solution generated by the tabu search procedure (at its 19th iteration)
Figure 3. The final solution generated by the tabu search procedure and the optimal IP solution