J Sched (2010) 13: 111–121
DOI 10.1007/s10951-009-0135-7

Production setup-sequencing and lot-sizing at an animal nutrition plant through ATSP subtour elimination and patching

Alistair R. Clark · Reinaldo Morabito · Eli A.V. Toso

Published online: 16 October 2009

Abstract This paper considers the usefulness of a production lot sizing and scheduling model at an animal nutrition plant with sequence-dependent setup times. The model covers multiple periods and is based on the asymmetric travelling salesman problem (ATSP). It is applied initially to the case where the setup state is zeroed between periods, and then revised to model the carryover of the setup state from one period to the next. An iterative solution procedure based on subtour elimination is applied, and then enhanced by the inclusion of a subtour patching procedure. Case-based tests with actual plant data show that the subtour elimination is practicably fast where the setup state is zeroed between periods, but needs the patching procedure when the setup state is preserved, as is the situation at the plant. In this latter case, the subtour elimination and patching can be very fast, showing the method's viability for operational lot sizing and sequencing in animal nutrition plants of the kind studied. Tests on perturbed plant data show that further algorithmic development is needed to tackle certain challenging variants found in other plants.

A.R. Clark (✉)
Bristol Institute of Technology, University of the West of England, Frenchay Campus, Bristol BS16 1QY, UK
e-mail: Alistair.Clark@uwe.ac.uk

R. Morabito
Production Engineering Department, Universidade Federal de São Carlos, Campus de São Carlos, Rodovia Washington Luís (SP-310), Km 235, São Carlos, SP 13565-905, Brazil
e-mail: Morabito@ufscar.br

E.A.V. Toso
Production Engineering Department, Universidade Federal de São Carlos, Campus de Sorocaba, Rodovia João Leme dos Santos (SP-264), Km 110, Sorocaba, SP 18052-780, Brazil
e-mail: Eli@ufscar.br

Keywords Lot sizing · Production scheduling · Sequence-dependent setup times · Asymmetric travelling salesman problem · Animal nutrition industry · Case-study

1 Introduction

In many manufacturing systems, production lots correspond to specific orders and so have a predetermined size. However, a product may instead feed into many distinct orders with different deadlines. In such a situation, it can make economic sense to relate the product's lot-sizes to its total demand aggregated from the different orders, particularly when there is a setup cost or time charged for each lot. Such setups are often sequence-dependent, that is, the size of the setup charge depends on the product processed immediately beforehand in the sequence of production lots.

A good example is provided by the subject of this paper, namely, Anifeed, a Brazilian animal nutrition company (whose real name has been altered to protect its identity). Anifeed's production schedule needs to specify the lot sizes and setup sequence of many different feed mixes. The capacity-time needed for a setup is sequence-dependent, so that scheduling production to efficiently use capacity and still meet demand results in a problem which can be computationally too complex to solve exactly. The temptation is to deal with lot sizing and sequencing independently of each other, but in the animal nutrition industry, as in similar processes, this creates difficulties in being able to flexibly meet changing market demand and order due dates within the available production capacity. Thus an integrated modelling approach is needed, but there still remains the challenge of how to solve such a model effectively.

Anifeed currently produces about 200 animal nutrition supplements which can be grouped into about 20 families.

Products within the same family do not contaminate each other and have the same production time per batch. All animal nutrition supplements follow the same basic production route, and make use of the same key resources: silos, dosing machines, pre-mix machines, a single mixer, and post-mix packaging. The amount of time spent at these operations varies somewhat between product families. The mixer is considered to be the bottleneck that determines productive capacity and flow.

Technically, the mixer must only be at least half-full to ensure efficient mixing, but there are advantages to producing a full batch. Anifeed faces demand quantities that are a whole number of batches and so wishes to work with integer production and inventory quantities. Thus lot sizes are constrained to be a whole number of batches.

Some product families can cause contamination of other families, and so the production line must sometimes be cleaned, resulting in substantial setup time of about 100 minutes and consuming scarce production time. The amount of cleaning can be minimised by the effective sequencing of production lots. More information about the process is available in Toso (2008) (including the test data) and in Toso et al. (2009).

The demand in a particular period often exceeds capacity, and so overtime is frequently worked to satisfy demand. The animal nutrition market is highly competitive, and so delivery delays to clients must be avoided if possible. A further possibility to avoid delivery delays is to produce some feeds ahead of demand when slack capacity is available. However, animal nutrition products are perishable and so cannot be produced too far in advance. Anifeed aims not to produce feeds more than about a month in advance of shipping to clients.

This paper presents a modelling and solution approach for Anifeed's lot sizing and scheduling problem based on two formulations related to the Asymmetric Travelling Salesman Problem (ATSP), and tests both in the context of Anifeed's production environment. The two ATSP-based models take into account asymmetric non-triangular setup times, a feature of many animal nutrition plants, including at Anifeed. However, they assume at most one lot per family per period, which is not necessarily true when setups are non-triangular (discussed in Sect. 3 below). Being case-oriented, this paper does not develop a completely optimal model (a subject for future theoretical research).

The first ATSP-based model, denoted *Independent Sequences*, represents the situation where it is possible to prepare setups between productive periods. The second model, denoted *Dependent Sequences*, considers the more difficult situation where the setup state is carried over between periods which is the case at Anifeed. Optimal solution methods are then presented, based on iterative subtour elimination and patching. After outlining an alternative model and solution approach for experimental comparison, computational tests are carried out on all models and methods using real data from Anifeed. To gain further insights, additional tests were carried out on perturbed data with random variations. The results are discussed from a practical perspective within the context of the animal nutrition and similar industries. The paper concludes with a discussion and pointers for future research.

2 Review of previous research

Research into lot sizing and scheduling research has been carried out for many years, as shown by the surveys of Wolsey (1995), Drexl and Kimms (1997), Karimi et al. (2003) and Jans and Degraeve (2008) and recent work by Luche et al. (2009) and Toledo et al. (2009). However, lot-sizing and sequencing has been studied by a smaller number of researchers (Clark and Clark 2000; Gupta and Magnusson 2005; Araújo et al. 2007, 2008).

The Capacitated Lot-Sizing Problem (CLSP) in Drexl and Kimms (1997), for example, does not include sequencing decisions. Smith-Daniels and Smith-Daniels (1986) formulated models for the CLSP with sequence-dependent set-up times, but their solution methods were restricted to small problems. Arosio and Sianesi (1993) proposed a complex heuristic algorithm for simultaneous lot-sizing and sequencing on a single machine with sequence-dependent set-ups, but made no comparisons with optimal solutions in their computational tests. This paper does make such comparisons for a mid-sized real problem.

The General Lotsizing and Scheduling Problem (GLSP), developed by Fleischmann and Meyr (1997), minimises inventory and sequence-dependent setup costs on a single machine with finite capacity, allowing multiple setups in each single "large-bucket" time period. The GLSP was extended by Meyr (2000) who formulated the General Lotsizing and Scheduling Problem—Setup Times (GLSP-ST) model for simultaneous lotsizing and scheduling on a single production line with sequence-dependent setup times. He divided the planning periods into a predetermined number of "small-bucket" micro-periods which contain at most one setup. Meyr did not model the backlogging of demand (a consequence of insufficient capacity). The wide-ranging review on capacitated lot sizing by Karimi et al. (2003) notes that there is little literature on problems with backlogging.

The Asymmetric Travelling Salesman Problem (ATSP) has been very extensively researched (Lawler et al. 1985; Laporte 1994; Junger et al. 1995; Carpaneto et al. 1995; Zhang 1997; Glover et al. 2001; Cirasella et al. 2001; Johnson et al. 2002; Buriol et al. 2004) and can be adapted to model the problem of sequencing a set of lots with sequence dependent setups between them. For example, Salomon et al. (1997) transformed the Discrete Lotsizing and

Scheduling Problem with sequence-dependent set-up costs and set-up times (DLSPSD) into a TSP with Time Windows (TSPTW) and used dynamic programming to solve it. These authors found that the ATSP approach is sensitive to factors such as instance size, inventory costs, and tightness of capacity. Pochet and Wolsey (2006) also suggested ATSP formulations as an alternative for modelling single-period production sequencing problems.

In Sect. 3 of this paper, we use the ATSP to reformulate a multi-period capacitated lotsizing and sequencing problem. As will be shown below, the adaptation is not direct since the production system is often already setup for a particular product family (i.e. starting at a given city) and some families might not be produced in a given period if the demand is sufficiently small or the capacity tight.

A method that has been found to be successful in practice for optimally solving the ATSP is to quickly solve the corresponding Assignment Problem (AP) as a linear programme, identify the resulting subtours, and then resolve the AP, explicitly prohibiting these subtours. The method carries on iteratively in this manner until no subtours result. It can be used heuristically (and its convergence rate sometimes accelerated) by patching the subtours into a single tour at each iteration (Karp 1979; Karp and Steele 1985; Frieze and Dyer 1990; Frieze et al. 1995; Zhang 1997), thus providing a feasible solution (and an upper bound). Section 4 of this paper adapts the subtour elimination method to lot sequencing, firstly over single periods, and then over multiple periods with setup carryover between periods to deal with the lot-scheduling problem of animal nutrition companies such as Anifeed. An extension of the method then successfully uses the patching heuristic to greatly accelerate convergence to an optimal solution.

Section 5 computationally assesses the models and methods. This section also compares the results with Anifeed's own practice using their own production data and with the GLSP approach tested by this paper's authors in Toso (2008) and Toso et al. (2009). The latter paper uses *relax-&-fix* (Wolsey 1998), a flexible and often fast heuristic approach used by many researchers (Dillenberger et al. 1994; Clark and Clark 2000; Clark 2003; Ferreira et al. 2009). Section 5 also seeks to obtain insight into the performance of the ATSP methods in a wider range of plants still representative of the animal nutrition industry by perturbing the Anifeed data in a controlled manner. Section 6 concludes by drawing together the case-based discussion and identifying some remaining challenges for industrial operationalisation and future research.

3 Modelling approaches

To help Anifeed schedule production more efficiently, two related models are now formulated as mixed integer linear programmes (MIP). The models minimise inventory, backorders and overtime, three criteria of major importance to the company, while keeping within available capacity and overtime limits.

The following indices are used:

i Product family, $i = 1, \dots, N$

t Time period, $t = 1, \dots, T$

where

$N =$ the number of families

$T =$ the number of periods in the scheduling horizon

The input data required by the models are:

C_t Available capacity time in period t

p_i Time needed to produce one batch of family i

lm_i Minimum lot size of family i (an integer number of batches)

h_i Cost of holding one period's inventory of family i

co_t Unit cost of overtime for period t

st_{ij} Setup time needed to changeover from family i to family j

d_{it} Forecast of demand for family i at the end of period t (an integer number of batches)

I_{i0} Inventory of family i at the start of the scheduling horizon

u_t Upper limit on the number of overtime hours permitted in period t

The decisions output by the models are:

I_{it}^+ Inventory of family i at the end of period t

I_{it}^- Backlogs of family i at the end of period t

q_{it} Production lot size of family i in period t (an integer number of batches). Since demand occurs only at the end of the periods, each family need only be produced once (or not at all) in any period

y_{ijt} $= 1$ if the production setup is to be changed over from family i to family j in period t, otherwise $= 0$

O_t Number of overtime hours needed in period t

3.1 Independent sequences: setup preparation between periods

In many companies, the setup of the initial product produced in a period takes place in the elapsed time gap between planning periods (for example, at the weekend), without eating into productive capacity. This situation is modelled first. The carryover of a setup configuration from one period to the next is modelled afterwards.

Let i_0 represent a "phantom" family from which there is zero setup time to any other family. If the setup state at the start of a period is i_0, then production of any other family can begin immediately since $st_{i_0,i} = 0$ for all families i. In reality, the setup to the first family may not have time zero, but since it occurs between periods, it should not be included in capacity requirement calculations (such as in Constraints (4) below). This is the only purpose of family

i_0, so setups to it from any family are prohibited by setting $st_{i,i_0} = \infty \; \forall i$. In addition, it has zero production time and demand, i.e. $p_{i_0} = 0$ and $d_{i_0,t} = 0 \forall t$.

Independent sequences model The objective function heavily penalises backlogs of demand, and minimises the costs of inventory and overtime:

$$\text{Minimise} \quad \sum_i \sum_t h_i \left(I_{it}^+ + M I_{it}^- \right) + \sum_t co_t O_t, \tag{1}$$

where M is a number large enough to impose effective backlogs penalties.

Differently to Meyr (2000), the objective function (1) does not consider setup costs, the reason being that setups basically consume just labour and time, neither of which incurs an immediate direct cost in companies such as Anifeed. Setup times are not directly penalised in the objective function (1), but indirectly though their use of overtime. Thus the minimisation of overtime in the objective function will help to prevent superfluous setups via the capacity constraints (4) below. However, for scheduling clarity, the fewer setups the better, so their minimisation is explicitly included in the objective function (1) via a relatively small penalty term:

$$\alpha \sum_i \sum_j \sum_t st_{ij} y_{ijt}, \tag{2}$$

where α is a constant small enough not to override the other terms in (1). Note that the presence of the setup times st_{ij} as a factor means that only setups y_{ijt} that involve cleaning time are actually penalised.

Constraints (3) balance inventory, backlogs, production and demand over consecutive weeks:

$$I_{i,t-1}^+ - I_{i,t-1}^- - q_{it} - d_{it} - I_{it}^+ - I_{it}^- \quad \forall i, t. \tag{3}$$

The capacity constraints (4) take into account the setup times as well as actual production times, and the possibility of a limited amount of overtime:

$$\sum_i p_i q_{it} + \sum_i \sum_j st_{ij} y_{ijt} \le C_t + O_t \quad \forall t. \tag{4}$$

Constraints (5) ensure that production of a family can occur in a period only if the line is set up accordingly:

$$p_i q_{it} \le (C_t - u_t) \sum_j y_{jit} \quad \forall t, i \ne i_0. \tag{5}$$

Constraints (6) enforce a minimum lot size and are needed as setup times do not always satisfy the triangular inequality:

$$q_{it} \ge lm_i \sum_j y_{jit} \quad \forall t, i \ne i_0. \tag{6}$$

To better understand why Constraints (6) are needed, consider a family i whose production contaminates that of family k unless a thorough cleaning occurs as part of the substantial setup time st_{ik}. In the animal nutrition industry, such cleaning can sometimes occur during the production of an intermediate family j whose setup time st_{ij} from i and setup time st_{jk} to k are jointly short enough that $st_{ik} > st_{ij} + st_{jk}$. Thus the triangular inequality $st_{ik} \le st_{ij} + st_{jk}$ does not hold in this case. Without Constraints (6) to impose sufficient production of j to allow proper cleaning of i's contaminants, an optimal schedule could setup from i to k via zero production of j rather than directly.

Note that the disobeying of the triangular inequality implies that it could be optimal in certain circumstances for any intermediate "cleansing" family j to be produced in more than one lot within the same period. The model's assumption of at most one lot per family per period would not hold in such a situation. The development of an appropriate optimal model is not tackled in this paper, but flagged as a subject for future research.

Note also that the triangular inequality always holds for the zero-valued setup times from phantom family i_0, but that Constraints (6) will, in effect, not apply to the first family produced in a period given such zero-valued setups.

Constraints (7) prohibit setups between the same family:

$$y_{iit} - 0 \quad \forall i, t. \tag{7}$$

The first setup will be from the "phantom" family i_0 (with setup time zero). Constraints (8) permit a setup to a family in a period only if there is a setup from family i_0:

$$\sum_j y_{i_0,j,t} \ge \sum_k y_{kit} \quad \forall i \ne i_0, t. \tag{8}$$

Constraints (9) permit a setup from a family (other than i_0) only if it has been setup to:

$$\sum_i y_{ijt} \ge \sum_k y_{jkt} \quad \forall j \ne i_0, t. \tag{9}$$

Constraints (10) prohibit more than one setup from a family, i.e. at most one lot can be produced of each family:

$$\sum_j y_{ijt} \le 1 \quad \forall i, t. \tag{10}$$

Constraints (11) prohibit family subtours in each period:

$$\sum_{i \to j \in S} y_{ijt} \le |S| - 1 \quad \forall \text{ subtours } S, t, \tag{11}$$

as adapted from the ATSP subtour exclusion constraints in Orman and Williams (2004) and Carpaneto et al. (1995). By summing the reverse as well as the forward arcs of each subtour, Constraints (11) can be replaced by:

$$\sum_{i\to j\in S}(y_{ijt}+y_{jit})\le |S|-1 \quad \forall \text{ subtours } S,t, \tag{12}$$

which are at least as strong as (11) for the following reason: for a given subtour S, any combination of the variables $\{y_{ijt}, y_{jit} \mid i\to j\in S\}$ that disobeys Constraints (12) will either (i) be S, (ii) be its reverse subtour, or (iii) have a family $i\in S$ for which $y_{ijt}=1$ for two $j\in S$. All three of these possibilities are prohibited.

Note the huge number of subtour prohibition Constraints (12). However, the vast majority will not be binding at the model's optimal solution so, imposing the constraints selectively just for those subtours that occur, a series of Assignment-type problems can be solved, as shown later in this paper.

Constraints (13) impose limits on overtime working:

$$0\le O_t\le u_t \quad \forall t. \tag{13}$$

Constraints (14) prohibit the inventory and backlogs variables to be negative:

$$I^+_{it}, I^-_{it}, \ge 0 \quad \forall i,t. \tag{14}$$

Constraints (15) ensure the setup changeover y_{ijt} variables are binary:

$$y_{ijt}=0 \quad \text{or} \quad 1 \quad \forall i,j,t. \tag{15}$$

Constraints (16) require the production to be a whole number of batches:

$$q_{it}\ge 0 \quad \text{and integer } \forall i,t. \tag{16}$$

Given that $st_{i,i_0}=\infty\ \forall i$, Constraints (17) below are redundant, but can be added to speed up the solution time. They ensure that there is no setup to the "phantom" family i_0 already setup at the start of each period:

$$y_{i,i_0,t}-0 \quad \forall i,t. \tag{17}$$

3.2 Dependent sequences model: setup carryovers between periods

Until now in this paper, the initial setup state at the start of each period t has been the phantom family i_0 rather than a variable. This is in line with the context and practice of many companies where the initial setup can occur between planning periods without eating into capacity. For example, in periods of less demand, the Anifeed plant operates from Monday to Saturday only, enabling the first family produced in a period to be setup in advance on Sunday by maintenance personnel.

However, many companies (including Anifeed in periods of peak demand) restart production with the same setup state as at the end of the previous period. In this case, the carryover of a setup state from the end of one period to the start of the next can [and should] be treated as a variable. This is modelled as follows.

Dependent Sequences Model: Declare the following new variable:

z_{it} = 1 indicates that family i is the setup state at the start of period t, otherwise = 0. In other words, family i was the last to be produced in period $t-1$.

The existing setup configuration z_{i1} is known and so is a parameter rather than a variable, whereas $z_{i,T+1}$ is a variable that represents the setup state at the end of the time horizon.

The objective function (1) with (2) and Constraints (3), (4), (7) and (12)–(16) remain unchanged from model AtspAnifeed. However, other constraints must be modified or added in order to model the carryover of a setup state from the end of one period to the start of the next, as follows.

Clearly only a single family can be the setup configuration at the start of a period, enforced by a new constraint:

$$\sum_i z_{it}-1 \quad t=2,\dots,T+1. \tag{18}$$

Constraints (5) are replaced by (19) below, still ensuring that production of a family can occur in a period only if the line is set up accordingly. Note that Constraints (19) permit the production of the family already setup at the start of period t:

$$p_i q_{it}\le (C_t+u_t)\left(z_{it}+\sum_j y_{jit}\right) \quad \forall i,t. \tag{19}$$

Constraints (6), which enforce a minimum lot size, need not apply to the family already setup at the start of a period, and so are replaced by (20) below:

$$q_{it}\ge lm_i\left(\sum_j y_{jit}-z_{it}\right) \quad \forall i,t. \tag{20}$$

Constraints (8) are replaced by (21), permitting a setup to a family only if it is set up from, or is the family already setup at the start of the next period:

$$\sum_i y_{ijt}\le \sum_k y_{jkt}-z_{j,t+1} \quad \forall j,t. \tag{21}$$

Constraints (9) are replaced by (22), permitting a setup from a family only if it has been setup to or is already setup at the start of a period:

$$z_{jt}-\sum_i y_{ijt}\ge \sum_k y_{jkt} \quad \forall j,t. \tag{22}$$

A combined and possibly tighter alternative to (21) and (22) is constraint set (23):

$$z_{jt} - \sum_i y_{ijt} - \sum_k y_{jkt} - z_{j,t+1} \quad \forall j, t. \tag{23}$$

The redundant Constraints (17) are replaced by the now necessary Constraints (24) which prohibit a setup to the family already setup at the start of a period:

$$1 - z_{it} \geq \sum_j y_{jit} \quad \forall i, t. \tag{24}$$

Similarly, Constraints (25) prohibit a setup from the family already setup at the start of the next period:

$$1 - z_{i,t+1} \geq \sum_j y_{ijt} \quad \forall i, t. \tag{25}$$

Constraints (24) and (25) are inequalities, rather than equalities, as some families may not need to be produced every period.

Constraints (26) require a setup from the family already setup at the start of the period:

$$z_{it} \leq \sum_j y_{ijt} \quad \forall i, t. \tag{26}$$

Note that Constraint (10) is now included in (25).

The rare case of no setups at all in a period did not need to be modelled at Anifeed in (26), but if required at other plants then the double Constraint (27) replaces (25) and (26):

$$z_{it} \leq \sum_j y_{ijt} - z_{i,t+1} \leq 1 \quad \forall i, t. \tag{27}$$

4 Solution approaches

4.1 ATSP subtour elimination

The optimal solution to Sect. 3 ATSP models without Constraints (12) will consist of zero or more subtours and a single sequence starting with the family already setup at the start of the period and ending with the last family setup in the period. In the following solution method, the subtours that arise in one period are prohibited in all periods in subsequent iterations.

Method ATSP-*Subtour*:

Repeat {
  Solve Model AtspAnifeed or AtspCarryover prohibiting only those subtours encountered so far in any period;
  Comment: provides a *lower* bound to optimal solution;
  For $t = 1, \ldots, T$ **do** {
    Identify any subtours in period t and specifically prohibit them in all periods from now using Constraint (12); }
  Resolve;
} **While** subtours exist in any period;
Comment: the solution is now optimal.

Note that an eventual feasible solution with no subtours will not feature a circular tour in each period, but rather a single unbroken sequence of up to N families starting with the family already setup at the start of the period and ending with the last family setup in the period.

4.2 Patching as well as eliminating subtours

A fast procedure that can give good and often near-optimal results for the ATSP is to solve an assignment problem and then use a *patching heuristic* (Karp 1979; Karp and Steele 1985; Frieze and Dyer 1990; Frieze et al. 1995; Zhang 1997) to gather (i.e. patch) the optimal assignment subtours into a single salesman tour.

The following extension of Method AtspCarryover patches the subtours in each period to obtain a single unbroken sequence, and thus a feasible solution, at each iteration.

Method ATSP-*Patching:*

Repeat {
  Solve the model with no subtour constraints;
  Comment: provides a *lower* bound to optimal solution.
  For $t = 1, \ldots, T$ **do** {
    Identify any subtours in period t and specifically prohibit them in all periods from now on using Constraint (12); }
  Resolve;
  For $t = 1, \ldots, T$ **do** {
    Patch any subtours in period t into a single unbroken sequence; }
  Fix the T single-tour sequences and resolve;
  Comment: The resulting feasible solution provides an *upper* bound to the optimal solution of the model.
  Unfix the single-tour sequences;
} **While** lower bound < upper bound or subtours exist in any period;
Comment: the solution is now optimal.

The patching operation gradually joins the single sequence and the subtours together so that the increase in total setup time is minimised, starting with the two subtours or sequence that have the most families, as in Karp (1979) and Karp and Steele (1985).

5 Computational tests

Tests were first carried out using real data in order to obtain Anifeed-based insights into the effectiveness of the models rather than to assess them rigorously. Each data set covered a month of Anifeed production thus permitted a comparison with the company's own scheduling.

The computational experiments evaluated the models' efficiency of performance and resulting schedules, comparing them not only with those practised by Anifeed but also with the best-performing method of the GLSP approach in Toso (2008) and Toso et al. (2009). This latter method uses *relax-&-fix* (Wolsey 1998) and is denoted GLSP-RF. It relaxes the integer lot-size variables q_{it}, solves the model, permanently fixes the values of the binary setup changeover variables y_{ijt}, restores the integrality constraints to the lot-size variables, and solve the partially-fixed model. Relax-& -fix can be an effective method, as in Ferreira et al. (2009), but its solutions are of varying quality and speed.

Follow-on tests were then carried out altering the data parameters so as to evaluate models' performance in situations different from those at Anifeed.

The models and methods were implemented in the AMPL mathematical programming language (Fourer et al. 2003). The MIPs were solved using CPLEX 9.1 (Ilog 2004). The tests were run on a Sun V208 Dual Opteran 252 with 1.5 GHz of RAM. Each MIP solved within the methods was allowed to run for a maximum of one hour of CPU time.

5.1 Tests with Anifeed data

The data from Anifeed comprised nine distinct month-long sets of weekly demand for 26 families over $T = 4$ weeks with 16 hours per week of available overtime u_t. The sequence-dependent inter-family setup times st_{ij} were either 100 or 0 minutes (rounded down from negligible near-zero times). The family production times p_i and inventory holding costs h_i ranged between 0.2 & 0.6 hours per batch and 102 & 922 monetary units per period respectively. The capacity C_t, overtime limit u_t and overtime costs were constant and equal to 64 hours, 16 hours and 859.2 monetary units per hour, respectively. The complete data are available online at Toso (2008).

For reasons of confidentiality, the data was slightly modified but maintained the proportions of demand among product families. Two of the data sets represented rainy and dry months in Brazil, while the other 7 sets were consecutive months, allowing a wider comparison of the models' results with Anifeed's practice. Rainy season demand is spread over a larger number of families than in the dry season, but with less demand per family. About five families dominated most of the demand, and a few had zero or very small demand in any month. The demand each week also varied within each family. Overtime is needed if a lot-for-lot production policy were followed, as Anifeed tended to do.

Applying the GLSP-RF and ATSP-Subtour methods to the *Independent Sequences* model gave the solutions in Table 1. The columns show the one-month data set used, the solution value after a maximum of one hour of computing time, the computing time used, the number of subtour-elimination iterations, and the total number of subtours that has to be explicitly prohibited. All demand over the planning horizon was met with no backorders. Note that the ATSP method found an optimal solution within the one-hour time limit (except for month 7 where the optimality gap is less than 1/4%) whereas the GLSP-RF *relax-&-fix* method found just one optimal solution. The mean GLSP-RF and ATSP times are comparable at around than 20 minutes. Note that the ATSP method's objective value is at least as good as that of the GLSP-RF method (and strictly better for months 1 and 7).

Table 2 shows the solutions achieved by the *Dependent Sequences* model using the GLSP-RF, ATSP-Subtour and ATSP-Patching methods. Note that while the GLSP-RF method is on average 5 times faster than the ATSP subtour elimination, its mean gap is over 7% even though its mean solution value is within 1% of the mean optimal value

Table 1 Results on the *Independent Sequences* model

	GLSP-RF			ATSP-Subtour				
Month	Obj. func.	CPU time	Gap (%)	Obj. func.	CPU time	Iterations	Subtours	Gap (%)
Rainy	3028	2860	3.28	3028	336	60	240	0
Dry	15425	779	0.18	15425	9	28	86	0
Month 1	4029	706	1.08	4023	3542	118	587	0
Month 2	7655	2531	4.69	7655	2011	194	916	0
Month 3	3207	968	0.99	3207	460	160	686	0
Month 4	3378	234	5.31	3378	104	97	419	0
Month 5	13056	28	0	13056	43	113	486	0
Month 6	9611	790	0.23	9611	399	14	76	0
Month 7	9088	2579	6.09	9076	3600	4	20	0.22
Mean	7609	1275	2.43	7607	1167	88	391	0.02

Table 2 Results on the *Dependent Sequences* model

	GLSP-RF			ATSP-Subtour					ATSP Patching				
Month	Obj. func.	CPU time	Gap (%)	Obj. func.	CPU time	Itera-tions	Sub-tours	Gap (%)	Obj. func.	CPU time	Itera-tions	Sub-tours	Gap (%)
Rainy	3453	273	13.9	3445	2374	63	267	0	3445	9	1	5	0
Dry	16616	132	7.29	16616	0.93	3	15	0	16616	9	1	9	0
Month 1	4312	132	6.69	4312	643	61	333	0	4312	472	1	7	0
Month 2	8176	258	10.5	8176	1098	62	369	0	8176	196	1	5	0
Month 3	4328	240	7.69	4281	273	176	766	0	4281	279	1	8	0
Month 4	3378	183	0.1	3378	264	106	499	0	3378	1	1	5	0
Month 5	13510	117	3.12	13510	3600	61	299	0.07	13510	4	1	6	0
Month 6	10019	128	4.23	10019	1082	48	250	0	10019	2	1	5	0
Month 7	10726	484	13.03	10237	848	81	405	0	10237	7	1	7	0
Mean	8280	216	7.39	8219	1131	73	356	0.01	8219	109	1	6	0

of 8219. The ATSP subtour elimination method optimally solved all data sets, with the exception of month 5, requiring a mean of 1131 seconds, 73 iterations and 356 subtours to converge to optimality. Interestingly, months *Rainy*, 2 and 5 have a higher demand of the more contaminating families and, along with month 6, are the data sets that took most effort to solve. This insight needs further research to assess whether it indicates a general pattern. The final MIP solved to optimality within the time limit, thus guaranteeing optimality of the no-subtours solution.

The most encouraging result of all is the very fast convergence afforded by the use of the patching algorithm at each iteration. In fact, the algorithm always gave a patched solution with the same value as the unpatched subtour solution. In other words, the first iteration's upper and lower solution bounds were identical, thus proving the optimality of the patched solution. This resulted in solution times between 1 and 472 seconds (almost exclusively used to solve the initial MIP), with a mean of 108 seconds, over ten times faster than simply using subtour elimination. However, the convergence after just one iteration is quite possibly due to the existence of many alternative optimal solutions, resulting from the 0 or 100-minute setup times. Bearing this in mind, the next set of computational tests in Sect. 5.2 below included randomly varying the setup times.

How do these results compare with scheduling practice at Anifeed? At the start of this research, the company was very cooperative in providing staff time, demand data and production information. However, the following year, Anifeed changed its production system from that described in a *Document of Understanding* agreed the previous year, and was unable to continue collaborating with the research. This prevented on-site validation of the models, but was resolved by comparing the GLSP/ATSP schedules with a heuristic that reflected Anifeed's scheduling methodology, namely, to produce each week's forecast demand in that week, adopting a *chase strategy* (Nahmias 1995), incurring overtime if need be.

Lot sequencing at Anifeed was carried out by the plant floor, in an intuitive manner using a greedy approach. Given that inter-family setup times were either zero or a constant 100 minutes, this was not difficult to do optimally after all the lot-sizes had been fixed. If there was spare capacity, then some future production of a family could be anticipated to take advantage of a current setup to that family. If there was too much demand to produce with the available overtime, then some production would be delayed. Assuming that the plant floor implements this approach effectively, it is algorithmically equivalent to fixing that only those families with a non-zero demand in a given period will be produced [and so sequenced] in that period and then optimally solving the *Independent Sequences* model. Anifeed's scheduling methodology was simulated and compared to our model results. This showed that the company's scheduling practices resulted in a mean objective function value that was more than a fifth higher than that from the *Independent Sequences* ATSP method. Both the company's practice and the ATSP method avoided demand backlogs, but the former resulted in an average of 40% more overtime and 8% more finished inventory compared to the latter.

To sum up, the ATSP-based models provide schedules that are better than the GLSP-RF method and those practiced by Anifeed, have optimality guarantees and run in acceptable computing times, especially the ATSP-Patching method.

5.2 Further insights by testing with perturbed Anifeed data

Further tests were carried out to better evaluate the usefulness of the ATSP methods in contexts similar to Anifeed's but still representative of the animal nutrition industry. To do so, the Anifeed data was perturbed as follows:

(a) *Higher inventory costs*. Anifeed's inventory costs are relatively small compared to its expenditure on overtime. As a result, the model's schedules tend to anticipate production rather than temporarily create extra capacity. How do the models perform if inventory costs are gradually increased so as to reflect Anifeed's strategy of trying to produce each week just the forecast demand?
(b) *Tighter capacity*. What if capacity in all periods was decreased by 10% and 20%?
(c) *Different patterns of demand.* The mix of demand in the animal nutrition industry varies considerably from one period to another. To assess how this might impact on the models, three scenarios were tested: (1) demand is concentrated in the last period, (2) demand for a family is randomly varied between its observed minimum and maximum levels, and (3) demand for a family is randomly varied between minimum and maximum levels that is the same for all families.
(d) *Number of batches*. Anifeed requires an integer number of batches to be produced in each lot. Suppose this requirement was relaxed so that fractions of batches could be produced?
(e) *Setup times*. Anifeed's setup times are asymmetric, non-triangular, and zero between 75% of families. To the assess the impact of these characteristics, three scenarios were tested: (1) restore the small non-zero setups that had previously been rounded down to zero, (2) use only symmetric triangular setup times, and (3) include a greater proportion of non-zero setups.

The test results in Table 3 show the percentage of times the best solution identified over all methods (including the GLSP-RF) was found by the method in hand, the percentage of instances for which a feasible solution was found, and the computing time.

Table 3 shows that for the *Dependent Sequences* model the ATSP-Patching method found best solutions in 98% of the instances while this fell to 77% for the ATSP-Subtours methods. All three methods in Table 3 found feasible solutions in almost every instance. Regarding the 5 types of perturbation (a)–(e), the tests revealed that:

(a) *Inventory costs* have to increase substantially before the *Dependent Sequences* schedule tends to resemble that of Anifeed with its greater use of overtime. Interestingly, solution quality remained high while computing time fell to 10 seconds or less for the *Independent Sequences* ATSP-Subtours method and *Dependent Sequences* ATSP-Patching methods.
(b) *Tighter capacity* with decreases of 10% and 20% resulted in overtime costs increasing by a factor of 6 and 12, respectively. However, the ATSP-based methods continued to perform well, quickly finding optimal solutions in all instances.
(c) *Different patterns of demand*. Not surprisingly, demand Scenario 1 resulted in greatly increased inventory costs, but the models found optimal solutions in all instances. In Scenario 2, no big changes were observed, except for the *Dependent Sequences* ATSP-Subtours method which found optimal solutions in only a fifth of instances. However, Scenario 3's identical mean demand for all families resulted in a greater amount of time spent on setups. A fifth of Scenario 3's instances were not solvable by any of the methods. Among the other instances, the ATSP-Patching method performed best, always obtaining an optimal solution in a mean time of 98 seconds, contrasting with the *Dependent Sequences* ATSP-Subtours method which in all three scenarios required much more computing time than the tests with actual Anifeed data.
(d) *Number of batches*. Permitting fractional batches results in optimal solutions being found in much less time.
(e) *Setup times*. The perturbation of setup times causes a noticeable deterioration in *Dependent Sequences* performance for Scenarios 1 and 3, not surprisingly given that there number of non-zero setups had increases. Neither method identified an optimal solution within the time limit. For Scenario 2 (symmetric setup times), ATSP-Patching performed comparatively well and fast, though slower than for the actual asymmetric data. The ATSP-Subtour method performed much worse than ATSP-Patching for all 3 scenarios. For the harder *Dependent Sequences*, the performance was worse than for the original data, while for the easier *Independent Sequences* models, the performance was seemingly better.

In general the perturbed data required more computing time of all methods, including the ATSP-Patching which, nevertheless, remained by far the fastest method.

6 Conclusions

This paper presented two multi-period lot sizing and scheduling models suitable for operational use when setup times are sequence-dependent and may also be asymmetric and non-triangular. The *Independent Sequences* model is suitable when inter-period setups can be performed without consuming productive capacity, thus delinking lot-sequencing within successive periods. The *Dependent Sequences* model reflected the more challenging case when a production line restarts production with the same setup state as at the end of the previous period. The carryover of a setup state from the end of one period to the start of the next was treated as a variable rather than via the phantom family of the *Independent Sequences* model. A single-period ATSP solution method based on iterative subtour elimination was presented and then simultaneously applied over multiple periods to

Table 3 Results using perturbed data

Tests		*Dependent Sequences* ATSP-**Subtours** Best Soln. (%)	Feas. Soln. (%)	CPU time (s)	ATSP-**Patching** Best Soln. (%)	Feas. Soln. (%)	CPU time (s)	*Independent Sequences* ATSP-**Subtours** Best Soln. (%)	Feas. Soln. (%)	CPU time (s)
Actual data		100	100	1131	100	100	108	100	100	1167
(a)	+10%	100	100	3574	100	100	553	100	100	2393
	+50%	67	100	3085	100	100	88	100	100	1133
	+100%	67	100	2457	100	100	6	100	100	244
	+200%	100	100	1199	100	100	9	100	100	195
(b)	−10%	100	100	1104	100	100	97	100	100	23
	−20%	100	100	1112	100	100	2	100	100	107
(c)	Scen. 1	100	100	121	100	100	1	100	100	0
	Scen. 2	20	100	3522	100	100	10	100	100	1419
	Scen. 3	60	80	–	90	80	1488	90	80	–
(d)	non-int.	100	100	796	100	100	15	100	100	591
(e)	Scen. 1	22	44	–	89	100	3584	100	78	–
	Scen. 2	89	89	–	100	100	381	100	100	938
	Scen. 3	11	100	2104	89	100	1265	78	100	135
	Mean	77	95	1874	98	99	476	98	97	603

solve both models. The integration of ATSP subtour elimination and patching can together dramatically accelerate the convergence from both above (with feasible solutions) and below (with infeasible solutions) to a provably optimal solution, compared to the use of subtour elimination to converge from below only (via infeasible solutions). In addition, the use of patching to produce feasible solutions at each iteration means that the user can prematurely stop the algorithms before proving optimality and yet still have a feasible solution (with a guarantee of distance from optimality) to use practically.

Tests using actual data from Anifeed, a Brazilian animal nutrition manufacturer, showed operational viability and quality of the models. When there was inter-period cleansing without setup carryover, the ATSP subtour elimination method was usually able to produce a provably optimal solution. The adaptation to include setup carryover resulted in a larger MIP that took much longer to solve. However, the patching of subtours to produce a feasible solution and upper bound enabled the setup carryover model to converge extremely rapidly.

Testing with perturbed data, to represent departures from the Anifeed situation, revealed that variations in inventory costs, tightness of capacity, patterns of demand, fractional batches, and setup times, can impact negatively on the quality and computing time of the schedule solutions. The tests provided case-study insights, but were not extensive enough to reach firm conclusions. This motivates future and more thorough research into factors impacting the performance of the models and methods.

In addition, it is clear that the perturbed tests show the need to accelerate of solution methods for the *Dependent Sequences* model by, for example, relaxing demand and lot-size integrality within a relax-&-fix ATSP solution method in the manner successfully applied to difficult instances and the GLSP approach in Toso et al. (2009). An alternative method that needs to be explored is to incorporate subtour elimination Constraints (12) and patching within the linear programmes at nodes in a single branch-&-cut tree search (Kang et al. 1999), instead of solving the relaxed problem, without Constraints (12), to integer optimality and then eliminating specific subtours with (12). Moreover, new models need to be developed that recognise that the disobeying of the triangular inequality implies that it could be optimal in certain circumstances for any intermediate cleansing family to be produced in more than one lot within the same period.

Acknowledgements We wish to record our appreciation of the referees' helpful feedback that resulted in a better article. In addition, we acknowledge the financial support of Brazil's *CNPq* and the UK's *EPSRC*. We also thank Anifeed for providing data and collaborating with this research.

References

Araújo, S. A., Arenales, M. N., & Clark, A. R. (2007). Joint rolling-horizon scheduling of materials processing and lot-sizing with sequence-dependent setups. *Journal of Heuristics*, *13*(4), 337–358.

Araújo, S. A., Arenales, M. N., & Clark, A. R. (2008). Lot-sizing and furnace scheduling in small foundries. *Computers and Operations Research*, *35*, 916–932.

Arosio, M., & Sianesi, A. (1993). A heuristic algorithm for master production scheduling generation with finite capacity and sequence dependent setups. *International Journal of Production Research*, *31*, 531–553.

Buriol, L., Franca, P. M., & Moscato, P. (2004). A new memetic algorithm for the asymmetric traveling salesman problem. *Journal of Heuristics*, *10*(5), 483–506.

Carpaneto, G., Dell'Amico, M., & Toth, P. (1995). Exact solution of large scale asymmetric travelling salesman problems. *ACM Transactions on Mathematical Software*, *21*(4), 394–409.

Cirasella, J., Johnson, D. S., McGeoch, L. A., & Zhang, W. (2001). The asymmetric traveling salesman problem: Algorithms, instance generators, and tests. In *Springer lecture notes in computer science* (Vol. 2153, pp. 32–59). Berlin: Springer.

Clark, A. R. (2003). Optimization approximations for capacity constrained material requirements planning. *International Journal of Production Economics*, *84*(2), 115–131.

Clark, A. R., & Clark, S. J. (2000). Rolling-horizon lot-sizing when setup times are sequence-dependent. *International Journal of Production Research*, *38*(10), 2287–2308.

Dillenberger, C., Escudero, L. F., Wollensak, A., & Zhang, W. (1994). On practical resource allocation for production planning and scheduling with different setup products. *European Journal of Operational Research*, *5*(2), 275–286.

Drexl, A., & Kimms, A. (1997). Lot sizing and scheduling—survey and extensions. *European Journal of Operational Research*, *99*, 221–235.

Ferreira, D., Morabito, R., & Rangel, S. (2009). Solution approaches for the soft drink integrated production lot sizing and scheduling problem. *European Journal of Operational Research*, *196*, 697–706.

Fleischmann, B., & Meyr, H. (1997). The general lotsizing and scheduling problem. *OR Spektrum*, *19*(1), 11–21.

Fourer, R., Gay, D. M., & Kernighan, B. W. (2003). *AMPL—a modelling language for mathematical programming* (2nd Ed.). N. Scituate: Duxbury Press. http://www.ampl.com/.

Frieze, A. M., & Dyer, M. E. (1990). On a patching algorithm for the random asymmetric travelling salesman problem. *Mathematical Programming*, *46*, 361–378.

Frieze, A. M., Karp, R. M., & Reed, B. (1995). When is the assignment bound tight for the asymmetric traveling-salesman problem? *SIAM Journal on Computing*, *24*(3), 484–493.

Glover, F., Gutin, G., Yeo, A., & Zverovich, A. (2001). Construction heuristics for the asymmetric tsp. *European Journal of Operational Research*, *129*, 555–568.

Gupta, D., & Magnusson, T. (2005). The capacitated lot-sizing and scheduling problem with sequence-dependent setup costs and setup times. *Computers and Operations Research*, *32*(4), 727–747.

Ilog (2004). CPLEX 9.1 user's manual, ILOG S.A. www.cplex.com.

Jans, R., & Degraeve, Z. (2008). Modeling industrial lot sizing problems, a review. *International Journal of Production Research*, *46*(6).

Johnson, D. S., Gutin, G., McGeoch, L. A., Yeo, A., Zhang, W., & Zverovich, A. (2002). Experimental analysis of heuristics for the asymmetric travelling salesman problem—algorithms, instance generators and tests. In G. Gutin & A. P. Punnen (Eds.), *The traveling salesman problem and its variations*. Dordrecht: Kluwer.

Junger, M., Reinelt, G., & Rinaldi, G. (1995). The traveling salesman problem. In M. O. Ball, T. L. Magnanti, C. L. Monma, & G. L. Nemhauser (Eds.), *Handbooks in operations research and management science: Vol. 7. Network models* (pp. 225–330). Amsterdam: North Holland. Chap. 4.

Kang, S., Malik, K., & Thomas, L. J. (1999). Lotsizing and scheduling on parallel machines with sequence-dependent setup costs. *Management Science*, *45*(2), 273–289.

Karimi, B., Fatemi Ghomia, S. M. T., & Wilson, J. M. (2003). The capacitated lot sizing problem, a review of models and algorithms. *Omega*, *31*, 365–378.

Karp, R. M. (1979). A patching algorithm for the nonsymmetric traveling-salesman problem. *SIAM Journal on Computing*, *8*(4), 561–573.

Karp, R. M., & Steele, J. M. (1985). Probabilistic analysis of heuristics. In E. L. Lawler, J. K. Lenstra, A. H. G. Rinnoy Kan, & D. B. Shmoys (Eds.), *The traveling salesman problem—a guided tour of combinatorial optimization*. Chichester: Wiley.

Laporte, G. (1994). The travelling salesman problem, an overview of exact and approximate algorithms. *European Journal of Operational Research*, *59*, 231–247.

Lawler, E. L., Lenstra, J. K., Rinnoy Kan, A. H. G., & Shmoys, D. B. (1985). *The traveling salesman problem—a guided tour of combinatorial optimization*. Chichester: Wiley.

Luche, J. R. D., Morabito, R., & Pureza, V. (2009). Combining process selection and lot sizing models for the production scheduling of electrofused grains. *Asia-Pacific Journal of Operations Research*, *26*(3).

Meyr, H. (2000). Simultaneous lot-sizing and scheduling by combining local search with dual optimization. *European Journal of Operational Research*, *120*, 311–326.

Nahmias, S. (1995). *Production and operations analysis*. Homewood: Irwin.

Orman, A. J., & Williams, H. P. (2004). *A survey of different integer programming formulations of the travelling salesman problem* (Working paper LSEOR 04.67). London School of Economics, Deptartment of Operational Research, LSE, London.

Pochet, Y., & Wolsey, L. A. (2006). *Production planning by mixed integer programming*. Berlin: Springer.

Salomon, M., Solomon, M., Van Wassenhove, L. N., & Dumas, Y. (1997). Solving the discrete lot sizing and scheduling problem with sequence dependent set-up costs and set-up times using the travelling salesman problem with time windows. *European Journal of Operational Research*, *100*, 494–513.

Smith-Daniels, V. L., & Smith-Daniels, D. E. (1986). A mixed integer programming model for lot sizing and sequencing packaging lines in the process industries. *IIE Transactions*, *18*(3), 278–285.

Toledo, C. F. M., França, P. M., Morabito, R., & Kimms, A. (2009). A multi-population genetic algorithm to solve the synchronized and integrated two-level lot-sizing and scheduling problem. *International Journal of Production Research*, *47*(11), 3097–3119.

Toso, E. A. V. (2008). *Dimensionamento e seqüenciamento de lotes de produção na indústria de suplementos para nutrição animal*. PhD thesis, Department of Production Engineering, Universidade Federal de São Carlos, Brazil. Complete text and data online at http://200.136.241.56/htdocs/tedeSimplificado/tde_busca/arquivo.php?codArquivo=1924.

Toso, E. A. V., Morabito, R., & Clark, A. R. (2009). Lot-sizing and sequencing optimisation at an animal-feed plant. *Computers and Industrial Engineering*, *57*(3), 813–821.

Wolsey, L. A. (1995). Progress with single-item lot-sizing. *European Journal of Operational Research*, *86*, 395–401.

Wolsey, L. A. (1998). *Integer programming*. New York: Wiley.

Zhang, W. (1997). A note on the complexity of the asymmetric travelling salesman problem. *Operations Research Letters*, *20*, 31–38.