Solving Lotsizing Problems on Parallel Identical Machines Using Symmetry Breaking Constraints

Raf Jans

ERIM REPORT SERIES *RESEARCH IN MANAGEMENT*	
ERIM Report Series reference number	ERS-2006-051-LIS
Publication	September 2006
Number of pages	20
Persistent paper URL	
Email address corresponding author	rjans@rsm.nl
Address	Erasmus Research Institute of Management (ERIM) RSM Erasmus University / Erasmus School of Economics Erasmus Universiteit Rotterdam P.O.Box 1738 3000 DR Rotterdam, The Netherlands Phone: + 31 10 408 1182 Fax: + 31 10 408 9640 Email: info@erim.eur.nl Internet: www.erim.eur.nl

Bibliographic data and classifications of all the ERIM reports are also available on the ERIM website:
www.erim.eur.nl

ABSTRACT AND KEYWORDS	
Abstract	Production planning on multiple parallel machines is an interesting problem, both from a theoretical and practical point of view. The parallel machine lotsizing problem consists of finding the optimal timing and level of production and the best allocation of products to machines. In this paper we look at how to incorporate parallel machines in a Mixed Integer Programming model when using commercial optimization software. More specifically, we look at the issue of symmetry. When multiple identical machines are available, many alternative optimal solutions can be created by renumbering the machines. These alternative solutions lead to difficulties in the branch-and-bound algorithm. We propose new constraints to break this symmetry. We tested our approach on the parallel machine lotsizing problem with setup costs and times, using a network reformulation for this problem. Computational tests indicate that several of the proposed symmetry breaking constraints substantially improve the solution time, except when used for solving the very easy problems. The results highlight the importance of creative modeling in solving Mixed Integer Programming problems.
Free Keywords	Mixed Integer Programming, Formulations, Symmetry, Lotsizing
Availability	
Classifications	

SOLVING LOTSIZING PROBLEMS ON PARALLEL IDENTICAL MACHINES USING SYMMETRY BREAKING CONSTRAINTS

RAF JANS
RSM Erasmus University, PO Box 1738, 3000 DR Rotterdam, The Netherlands
Email: rjans@rsm.nl; Tel.: +31 10 408 2774

August 31, 2006

Abstract

Production planning on multiple parallel machines is an interesting problem, both from a theoretical and practical point of view. The parallel machine lotsizing problem consists of finding the optimal timing and level of production and the best allocation of products to machines. In this paper we look at how to incorporate parallel machines in a Mixed Integer Programming model when using commercial optimization software. More specifically, we look at the issue of symmetry. When multiple identical machines are available, many alternative optimal solutions can be created by renumbering the machines. These alternative solutions lead to difficulties in the branch-and-bound algorithm. We propose new constraints to break this symmetry. We tested our approach on the parallel machine lotsizing problem with setup costs and times, using a network reformulation for this problem. Computational tests indicate that several of the proposed symmetry breaking constraints substantially improve the solution time, except when used for solving the very easy problems. The results highlight the importance of creative modeling in solving Mixed Integer Programming problems.

Keywords: Mixed Integer Programming: Formulations; Symmetry; Lotsizing

1. Introduction

The aim of lotsizing is to determine the timing and level of production for several products over a specified discrete time horizon. Future demand is known and must be met at minimum cost. Machine capacity is a scarce resource in most manufacturing environments. Usually lotsizing models assume that the products are made on one single machine. However, in many cases a manufacturer has access to multiple machines or production lines, which can be used in parallel.

In this paper, we consider the case of parallel identical machines in a single stage. As the machines are identical, they all have the same capacity available. For each product, the setup and production cost and time are also identical on each of the machines. The machines are flexible as they can all produce the complete set of items. Parallel machines complicate the problem as we not only have to determine the timing and level of production, but we also have to assign production lots to machines. In the case of identical parallel machines, the problem is further complicated due to symmetry, which results in the existence of many alternative optimal solutions. Given a solution, i.e. a proposal for the timing and level of production and an assignment of products to machines, a different solution with the same total cost can be created by just renumbering the machines. It is known that this symmetry will slow down the branch-and-bound algorithm due to unnecessary duplication. To counter this problem, symmetry breaking constraints can be added to the Mixed Integer Programming (MIP) formulation.

The contribution of this paper is twofold. First, we extend the network formulation of Eppen and Martin (1986) for lotsizing problems on a single machine to the case of parallel machines. The network formulation has the advantage that it has a smaller IP gap compared to the regular formulation. This allows us to solve large problems from a standard test set in our computational experiments. This contribution adds to the research on tighter formulations. Belvaux and Wolsey (2001) and Wolsey (2002) indicate the importance of good formulations. They claim that many practical lotsizing problems can be solved using general purpose MIP software if tight formulations are used. Such formulations result from insights into polyhedral properties of the formulation.

Second, we explore the issue of symmetry. This is an issue that, to the best of our knowledge, has not been discussed before in the lotsizing literature. We propose several symmetry breaking constraints for the lotsizing problem with identical parallel machines and evaluate them in a computational experiment. This research indicates that explicitly considering this symmetry by imposing hierarchical constraints, results in improved models and is an important factor in solving parallel machine lotsizing problems using commercial branch-and-bound software. This is our main contribution and it adds to previous research done on symmetry breaking by Sherali and Smith (2001).

2. Literature Review

Many different versions and extensions of the basic lotsizing problem have been studied extensively in the literature. For a general review on lotsizing models and algorithms, we refer the reader to some recent review articles and books (Jans and Degraeve 2007[a,b], Pochet and Wolsey 2006). In this section, we focus on the single stage, parallel machine problem. Note that the parallel machine lotsizing problem is different from the lotsizing problem with multiple resources. In this latter model, a product consumes capacity from many different resources simultaneously such as machine capacity, labor and tools and a separate capacity constraint is imposed for each of the resource types (e.g. Stadtler 2003).

The practical relevance of parallel machine lotsizing is supported by examples of its application in various industries such as pharmaceuticals (De Matta and Guignard 1995), tile manufacturing (De Matta and Guignard 1994[a]), the tire industry (Jans and Degraeve 2004), injection molding (Dastidar and Nagi 2005), the alloy foundry industry (Dos Santos-Meza et al. 2002) and multi-layer-ceramics (Dillenberger et al. 1994).

In lotsizing models, a distinction is usually made between big bucket and small bucket models. This distinction also extends to the parallel machine case. In big bucket models, a machine can produce several different product types in the same time period. This is the Capacitated Lotsizing Problem (CLSP). In the small bucket models a single mode constraint is imposed, indicating that at most one product type can be made on a specific machine in one time period. In cases where the production quantities can be anywhere between zero and the capacity, this is called the Continuous Setup Lotsizing Problem (CSLP). The parallel machine case is discussed in Ahmadi et al. (1992), De Matta and Guignard (1995), and Dastidar and Nagi (2005). In the Discrete Lotsizing and Scheduling Problem (DLSP), an all-or-nothing production policy is assumed: if you decide to produce in a specific time period, it must be at full capacity. Formulations with parallel machines are discussed in Salomon et al. (1991), De Matta and Guignard (1994[a], 1994[b]), Dumoulin and Vercellis (2000), Jans and Degraeve (2004). The Proportional Lotsizing and Scheduling Problem (PLSP) allows that a maximum of two different items can be produced in each time period. There is still at most one setup per period, but the setup from the previous period can be carried over to the next period. This problem can also be extended to parallel machines (Kimms and Drexl 1998).

Özdamar and Barbarosoğlu (1999) and Özdamar and Birbil (1998) consider a big bucket, capacitated lotsizing problem with parallel machines. They refer to the assignment of the items to machines as the loading problem. In their models, they assume that the lots cannot be split among several machines. So in one specific period, an item can be produced on one machine at most. We consider a more general model in the sense that we allow an item to be produced on two or more machines simultaneously. Our model is the same as the basic model for the multi-machine problem as defined by Belvaux and Wolsey (2001). They consider reformulations and cutting planes for a variety of lotsizing models, but they do not consider the issue of symmetry.

Madan and Gilbert (1992) propose a lotsizing model with parallel machines. However, there is only a general setup if a product is produced in a specific period, irrespective of whether this is done on one or more machines. In our proposed model, we have a separate setup cost for each machine if an item is produced on more than one machine simultaneously. Kang et al. (1999) propose a column generation approach to solve the CHES problems, which are complicated lotsizing problems with non-identical parallel machines, sequence-dependent setup costs and sales. Also Clark and Clark (2000), Clark (2003) and Meyr (2002) consider lotsizing with sequence dependencies on parallel machines. Production planning problems with parallel machines but no setups are considered by Leachman and Carmon (1992) and Hung and Cheng (2002). As no setup costs or time are taken into account, the model can be formulated as a linear program. Their problem is complicated by the fact that an item has to go through several process steps.

In this paper, we will look specifically at models with identical machines, as they exhibit a lot of symmetry in the solution representation. We specifically consider big bucket problems with setup costs and times. The reason for this focus is that small bucket models do not suffer from the same symmetry problem in their solution representation as big bucket models. Because of the single mode constraint, the setup variables can be modeled as general integers to indicate how many machines are used to produce one item type in a period. Also sequence-dependencies reduce the symmetry, as the machines cannot be renumbered independently in each period anymore. When no setups are present, the capacity of identical machines can be aggregated and the problem of symmetry is avoided in this way.

3. Mathematical Programming Formulations

We consider the basic big bucket, single stage, parallel machine lotsizing model with setup times and setup costs. The machines are identical, meaning that they have the same capacity available and for a specific item, the variable production time and cost and the setup time and cost are the same on each machine. We use the basic model defined by Belvaux and Wolsey (2001), who consider the general case of non-identical machines. We have the following sets, variables and parameters:

Sets:

P	Set of products, $= \{1, 2, \ldots, n\}$,
T	Set of periods, $= \{1, 2, \ldots, m\}$,
M	Set of parallel identical machines, $= \{1, 2, \ldots, q\}$,

Input Parameters:

d_{it}	Demand for item i in period t,
sd_{itl}	Sum of the demand for item i from period t until period l,
sc_i	Setup cost for item i,
vc_i	Variable production cost for item i,

hc_i	Holding cost for item *i*,
vt_i	Variable production time for item *i*,
st_i	Setup time for item *i*,
cap_t	Capacity available in period *t* on each machine,
fc_i	Cost for one unit of initial inventory for item *i*,

Decision variables:

x_{ikt}	Production level for item *i* in period *t* on machine *k*,
y_{ikt}	= 1 if there is a setup for item *i* in period *t* on machine *k*; 0 otherwise,
s_{it}	Inventory for item *i* at the end of period *t*,
s_{i0}	Initial inventory for item *i*.

The regular formulation is then as follows:

$$Min \sum_{k\in M}\sum_{i\in P}\sum_{t\in T}\left(sc_i y_{ikt} + vc_i x_{ikt}\right) + \sum_{i\in P}\sum_{t\in T} hc_i s_{it} + \sum_{i\in P} fc_i s_{i0} \qquad (1)$$

$$s.t. \quad s_{i,t-1} + \sum_{k\in M} x_{ikt} = d_{it} + s_{it} \qquad \forall i \in P,\ \forall t \in T \qquad (2)$$

$$x_{ikt} \le sd_{itm} y_{ikt} \qquad \forall i \in P,\ \forall t \in T,\ \forall k \in M \qquad (3)$$

$$\sum_{i\in P}\left(st_i y_{ikt} + vt_i x_{ikt}\right) \le cap_t \qquad \forall t \in T,\ \forall k \in M \qquad (4)$$

$$x_{ikt}, s_{it} \ge 0; y_{ikt} \in \{0,1\} \qquad \forall i \in P,\ \forall t \in T,\ \forall k \in M \qquad (5)$$

The objective (1) is to minimize the total cost of setups, production and inventory. For each item, the production on the parallel machines in each period is summed and this amount is available to satisfy demand (2). If an item is produced on a specific machine, a setup is incurred (3). If an item is simultaneously made on several machines, a setup is necessary for each of these machines. The available capacity on each machine is limited. Variable production time and setup times are both taken into account in order to calculate the actual capacity utilization (4). In order to deal with infeasible problems, we allow that initial inventory s_{i0} is available from an external source at a high cost fc_i (Vanderbeck 1998). Note that aggregating the capacity over the machines will not provide an equivalent formulation. The aggregate capacity constraint cannot properly account for multiple setups in cases where the production of one item is done on more than one machine.

It is well known that the regular formulation (1)-(5) in the *x* and *y* variables provides a poor lower bound. Several approaches have been proposed to strengthen the formulation. Eppen and Martin (1986) reformulate the lotsizing problem as a network problem and show that the LP relaxation of this formulation has an integer solution for the uncapacitated single item problem. This reformulation is actually the network formulation of the Dynamic Programming algorithm of Wagner and Whitin (1958), which uses the property that there is an optimal solution where production is done for an integer number of periods. The variable zv_{itl} is a binary variable which takes the value of 1 if production in period *t* covers the full demand for period *t* up to period *l* (Figure 1). Eppen and Martin further prove that network formulation is also a valid reformulation for the capacitated multi-item case and can be used to obtain tighter lower bounds. The zv_{itl} variables are no longer defined as binary. We extend the network reformulation to the parallel machine case. The arcs are duplicated for each machine available, adding an extra index to the variable. Figure 2 shows a network with 2 machines.

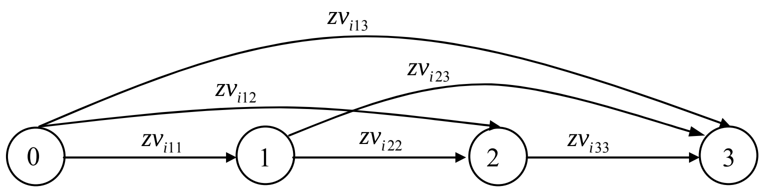

Figure 1. Network representation for one item *i*, 3 periods and 1 machine

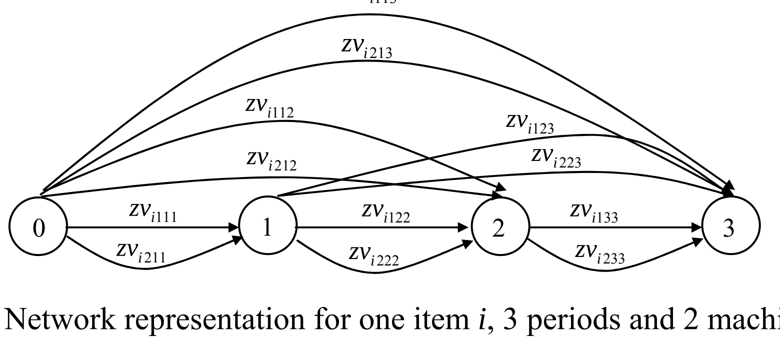

Figure 2. Network representation for one item *i*, 3 periods and 2 machines

The mathematical formulation is as follows:

$$Min \sum_{i \in P} \sum_{t \in T} \sum_{k \in M} sc_i y_{ikt} + \sum_{i \in P} \sum_{k \in M} \sum_{t \in T} \sum_{l=t}^{m} cv_{itl} zv_{iktl} + \sum_{i \in P} \sum_{t \in T} ci_{it} w_{it} \tag{6}$$

s.t.

$$1 = \sum_{l=1}^{m} \sum_{k \in M} zv_{ik,1,l} + \sum_{l=1}^{m} w_{il} \qquad \forall i \in P, \tag{7}$$

$$w_{i,t-1} + \sum_{s=1}^{t-1} \sum_{k \in M} zv_{iks,t-1} = \sum_{l=t}^{m} \sum_{k \in M} zv_{iktl} \qquad \forall i \in P, \forall t \in T \backslash \{1\}, \tag{8}$$

$$\sum_{t=1}^{m} \sum_{k \in M} zv_{iktm} = 1 \qquad \forall i \in P, \tag{9}$$

$$\sum_{l=t}^{m} zv_{iktl} \leq y_{ikt} \qquad \forall i \in P, \forall k \in M, \forall t \in T, \tag{10}$$

$$\sum_{i \in P} st_i y_{ikt} + \sum_{i \in P} \sum_{l=t}^{m} vt_i sd_{itl} zv_{iktl} \leq cap_t \qquad \forall k \in M, \forall t \in T, \tag{11}$$

$$zv_{iktl} \geq 0; \; y_{ikt} \in \{0,1\} \qquad \forall i \in P, \forall k \in M, \forall t, l \in T, l \geq t \tag{12}$$

The variable zv_{iktl} indicates the fraction produced of product *i* on machine *k* according to the production plan where production in period *t* satisfies demand from period *t* up to *l* (Eppen and Martin 1989). Likewise, the variable w_{it} indicates the fraction of the initial inventory plan for product *i* where demand is satisfied for the first *t* periods (Jans and Degraeve 2004). Note that we left out the w_{it} variables in Figures 1 and 2 in order to simplify the network graphs. The link with the variables of the regular formulation is as follows:

$$x_{ikt} = \sum_{l=t}^{m} sd_{itl} \, zv_{iktl}$$

$$s_{i0} = \sum_{l=1}^{m} sd_{i,1,l} \, w_{il}$$

On every arc, a cost cv_{itl} is defined as the total cost for producing in period *t* the demands for period *t* until *k* and the according inventory holding cost:

$$cv_{itl} = vc_i sd_{itl} + \sum_{s=t+1}^{l} \sum_{u=t}^{s-1} hc_u d_{is}$$

Constraints (7)-(9) are the conservation of flow equalities for the shortest path network. Sending a unit flow through the network is equivalent to imposing that demand must be met without backorders. We further have the setup forcing constraint (10) and capacity constraint (11). We use the network reformulation with initial inventory (Jans and

Degraeve 2004b). This formulation has two advantages. First, as explained earlier, problems are always feasible since initial inventory can be purchased at a high cost. Second, the regular network formulation, without initial inventory, does not correctly model the problem if the demand in the first period is zero. The reason is that in this case z_{ik11}, the arc to produce in period 1 the demand for period 1, should be set to one in order to have a feasible flow. However, this will trigger a setup through constraint (10), even if nothing is produced. This problem is solved by introducing initial inventory. If demand in the first period is zero, the arc to use initial inventory to cover demand for period 1 will be activated. The cost of this arc is zero as there is no demand. Further, there is no setup associated with such an initial inventory arc, and hence there is also no setup cost. Note that the network reformulation can easily be extended to include non-identical machines by introducing machine specific capacities, setup times and costs and variable production times and costs.

4. Symmetry breaking constraints

The existence of identical parallel machines results in the existence of many alternative solutions. In the formulation (1)-(5) and (6)-(12), there is a production and setup variable for each product-machine-period combination. Given a specific feasible solution, an alternative solution can be constructed by renumbering the machines. This leads to a different assignment of items to individual machines. However, as all machines are identical, this represents globally the same solution. This is illustrated in Figure 3 for a lotsizing problem with 3 machines, 4 products and 2 periods. The two plans are globally the same and differ only by the numbering of the identical machines.

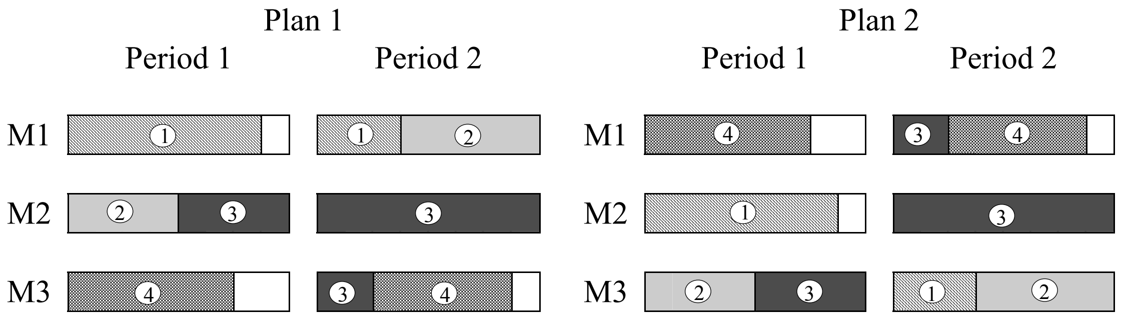

Figure 3: Alternative solutions for the lotsizing problem with identical machines

We further investigate the number of alternative solutions, given a specific solution. Suppose that we have k identical machines and m time periods. In a period, we can renumber the machines, giving $k!$ alternative solutions. As there is no interdependency on one machine between two periods, we can do this for each period independently, giving $(k!)^m$ alternative optimal solutions in total. This no longer holds if the periods are not independent, e.g. when sequence dependencies or setup carry-overs are present. Symmetry is also not a problem for unrelated machines, as we cannot renumber the machines to obtain alternative solutions.

A few papers have been published that explore the issues of symmetry in MIP models which are solved by standard branch-and-bound methods. The large number of alternative solutions will lead to problems in the branch-and-bound algorithm (Sherali and Smith, 2001). A path in the branch-and-bound tree leading to one of the alternative optimal solutions cannot be pruned as the lower bound will indicate that this is a valid path to investigate. Sherali and Smith (2001) assert that considering symmetry is an important modeling concept that is, however, not used very often. They illustrate the use of various symmetry breaking constraints on 3 different problems. Degraeve et al. (2002) explore symmetry in the IP formulation of a fixed charge cutting stock and layout problem in the fashion industry.

Symmetry breaking constraints aim to exclude alternative solutions. The first type of symmetry breaking constraints that we investigate for the lotsizing problem are lexicographic ordering constraints (Sherali and Smith 2001, Degraeve et al. 2002). We impose the constraint that if item 1 is produced in period t, it must be on the first machine(s). This is achieved by the following constraints:

$$y_{11t} \geq y_{12t} \geq y_{13t} \geq \dots$$

After imposing these constraints, we have a subset of machines on which item 1 is produced and a subset on which item 1 is not produced. If this does not result in a complete ordering, we impose a further hierarchical condition. In cases where there is a tie for item 1 (i.e. y_{1kt} and $y_{1,k+1,t}$ have the same value), we impose a further ordering on the production of item 2:

$$2y_{11t} + y_{21t} \geq 2y_{12t} + y_{22t} \geq 2y_{13t} + y_{23t} \geq \dots$$

In cases where there is still no complete ordering, the tie will be broken by looking at product 3:

$$4y_{11t} + 2y_{21t} + y_{31t} \geq 4y_{12t} + 2y_{22t} + y_{32t} \geq 4y_{13t} + 2y_{23t} + y_{33t} \geq \ldots$$

We continue this reasoning for the next items. This comes down to assigning a unique number to each possible configuration of setups on a machine, and next order the machines by decreasing value of this number. We use coefficients which are powers of two in order to ensure the uniqueness of the assigned number. We investigated three ways of implementing these lexicographic constraints. First of all, we impose all the lexicographic ordering constraints:

(SBC1) $$\sum_{j=1}^{i} 2^{(i-j)} y_{j,k-1,t} \geq \sum_{j=1}^{i} 2^{(i-j)} y_{jkt} \qquad \forall i \in P, \forall k \in M \setminus \{1\}, \forall t \in T$$

We can also only use the final ordering constraint, which includes all the items. This constraint suffices to impose a unique ordering as it assigns a unique number to each possible setup configuration on a machine.

(SBC2) $$\sum_{i=1}^{n} 2^{(n-i)} y_{i,k-1,t} \geq \sum_{i=1}^{n} 2^{(n-i)} y_{ikt} \qquad \forall k \in M \setminus \{1\}, \forall t \in T$$

We can also impose only a subset of the lexicographic ordering constraints. Using only the first constraints involving only item one, will impose a partial ordering.

(SBC3) $$y_{1,k-1,t} \geq y_{1kt} \qquad \forall k \in M \setminus \{1\}, \forall t \in T$$

Other ways of imposing a partial ordering can be obtained by assigning a number to each machine that is not necessarily unique. This can be achieved by using the structure of SBC2, but with different coefficients. The idea of imposing a hierarchy on the sum of the product indices or the sum of the squares of the product indices is similar to an idea used by Sherali et al. (2001) for a network design problem.

(SBC4) $$\sum_{i=1}^{n} i\, y_{i,k-1,t} \geq \sum_{i=1}^{n} i\, y_{ikt} \qquad \forall k \in M \setminus \{1\}, \forall t \in T$$

(SBC5) $$\sum_{i=1}^{n} i^2\, y_{i,k-1,t} \geq \sum_{i=1}^{n} i^2\, y_{ikt} \qquad \forall k \in M \setminus \{1\}, \forall t \in T$$

Yet other ways of breaking the symmetry can be achieved by ordering the machines according to some natural logic such as a decreasing total setup cost per machine (SBC6), decreasing total cost per machine (SBC7) or decreasing capacity utilization (SBC8):

(SBC6) $$\sum_{i=1}^{n} sc_i \; y_{i,k-1,t} \geq \sum_{i=1}^{n} sc_i \; y_{ikt} \qquad \forall k \in M \setminus \{1\}, \forall t \in T$$

(SBC7) $$\sum_{i=1}^{n} sc_i \; y_{i,k-1,t} + \sum_{i=1}^{n} \sum_{l=t}^{m} cv_{itl} \, zv_{i,k-1,tl} \geq \sum_{i=1}^{n} sc_i \; y_{ikt} + \sum_{i=1}^{n} \sum_{l=t}^{m} cv_{itl} \, zv_{iktl}$$

$$\forall k \in M \setminus \{1\}, \forall t \in T$$

(SBC8) $$\sum_{i=1}^{n} st_i \; y_{i,k-1,t} + \sum_{i=1}^{n} \sum_{l=t}^{m} vt_i sd_{itl} \, zv_{i,k-1,tl} \geq \sum_{i=1}^{n} sc_i \; y_{ikt} + \sum_{i=1}^{n} \sum_{l=t}^{m} vt_i sd_{itl} \, zv_{iktl}$$

$$\forall k \in M \setminus \{1\}, \forall t \in T$$

The effectiveness of these eight symmetry breaking constraints (SBC) will be tested in a computational experiment.

5. Computational Experiments

5.1. Discussion of the data sets used

As far as we know, no standard data test sets are available for the big bucket, parallel machine lotsizing problem. Therefore, we adapted an existing standard data set which was originally set up to test the single machine capacitated lotsizing problem with setup times (Trigeiro et al. 1989). This standard set is used in many computational experiments as a benchmark test set. We used the problem set F1 to F40. These problems all have 6 products and 15 periods. For F1 to F20 and F21 to F40, the original capacity level was set at 728 and 1064 respectively. For each of the 40 problem instances we created parallel machine problems with various capacity levels and number of machines available. The choice of the capacity levels was based on preliminary tests to have a broad range of easy and difficult problems. As such, each original single machine test problem resulted in 46 parallel machine test problems for the set F1-F20 and 61 test problems for the F21-F40 set. As a result, 2140 different test problems were created. The machine (M) and capacity (CAP) levels can be found in Table 1 and 2. Each of the 920 test problems in F1-F20 was solved with the 8 different symmetry breaking formulations (SBC1 to SBC8) and the base case (SBC0) where no symmetry breaking constraints are present. Based on the results from this first experiment, we decided to solve the 1220 test problems from F21-F40 only with the best 3 models (SBC1 to 3) and the base case (SBC0). The experiments required 200 days of computing in total.

5.2. Numerical Results

The experiments were performed on a 3.0 GHz computer with an Intel Pentium 4 processor. CPLEX 9.1.3 was used with the default setting. Only the MIP optimality gap was strengthened from 0.01% to 0.001%. We solved all the test problems using the network reformulation (6)-(12). All computation times are reported in seconds. We report two decimals for times below 1 second and 1 decimal for times below 100 seconds. We set a maximum time of 3600 seconds for each problem. In Tables 1 and 2, we present the CPU times for each combination of machine (M) and capacity (CAP) level and for various formulations (SBC0 to SBC8). The capacity level refers to the capacity available on each machine. For each machine-capacity combination, the CPU times are averaged over 20 problem instances. The smallest CPU time per level is indicated in bold.

Table 1: Average CPU times in seconds for the F1-F20 set for different machine and capacity levels

M	CAP	SBC0	SBC1	SBC2	SBC3	SBC4	SBC5	SBC6	SBC7	SBC8
2	525	38,6	3,7	**3,6**	7,6	50,5	8,9	23,3	67,1	201
2	500	226	9,2	**9,0**	26,2	75,7	27,8	48,7	87,3	110
2	475	363	33,9	**33,0**	202	259	126	231	398	679
2	450	1722	511	**509**	756	936	591	954	1332	2116
2	425	2422	**972**	1149	1609	1991	1425	1722	1825	2991
2	400	3562	2473	**2253**	2599	2772	2433	2899	2905	3493
2	375	3509	**3009**	3105	3273	3328	3012	3276	3272	3600
3	525	7,4	1,6	1,8	**1,6**	6,7	3,5	6,2	46,3	182
3	500	5,3	**1,7**	2,5	2,0	4,5	2,2	2,4	43,2	31,2
3	475	9,1	2,5	3,2	**1,6**	14,3	4,7	4,8	73,2	117
3	450	62,7	**3,1**	4,6	3,9	28,1	13,3	36,5	310	459
3	425	700	34,4	**16,3**	79,7	379	255	249	735	1250
3	400	901	**58,8**	289	405	702	401	370	1083	1924
3	375	2008	430	**341**	862	1189	704	959	1552	2713
3	350	2911	779	**637**	1807	1962	1827	1707	2152	3424
3	325	3528	**2321**	2566	3033	2900	2712	2903	3213	3450
3	300	3600	3080	3083	3477	3517	**3015**	3387	3438	3600
3	275	3600	**3120**	3371	3600	3600	3450	3507	3600	3600
3	250	3600	3600	3600	3600	3600	3600	3600	3600	3600
4	525	**0,3**	1,3	1,6	0,3	13,0	5,7	1,8	51,0	198
4	500	**0,4**	2,6	2,1	0,4	4,2	3,2	3,6	185	208
4	475	**0,5**	2,3	3,5	0,6	5,9	3,9	3,2	196	26,5
4	450	3,1	3,0	3,7	**1,6**	18,0	10,0	19,8	298	417
4	425	187	8,0	10,4	**4,9**	115	128	379	681	1004
4	400	385	**31,9**	51,1	55,5	418	374	302	621	1596
4	375	287	146	104	**57,8**	433	322	497	1000	1350
4	350	1253	**415**	523	684	1751	718	1268	1541	2680
4	325	2024	**1193**	1301	1779	2247	1867	1868	2252	3600
4	300	3290	**2416**	2437	2776	3137	2651	2791	2973	3531

4	275	3426	**2731**	2798	3120	3427	3161	2928	3270	3600
4	250	3600	3600	3600	3600	3600	3600	3600	3600	3600
4	225	3600	3600	3600	3600	3600	3600	3600	3600	3600
5	525	1,3	2,6	2,8	**0,4**	14,4	5,9	34,9	42,3	206
5	500	**0,6**	3,4	5,4	1,0	24,7	7,7	21,8	426	189
5	475	25,8	3,8	7,3	**0,8**	24,7	6,5	5,3	204	241
5	450	4,0	4,9	6,5	**3,6**	188	39,0	25,5	452	500
5	425	11,0	13,2	17,8	**4,3**	530	109	299	1053	852
5	400	366	85	364	**32,1**	640	395	335	972	1375
5	375	223	330	565	**30,9**	939	448	484	1322	2009
5	350	859	**594**	852	637	1734	1039	1182	1752	2584
5	325	2114	**1370**	1687	1670	2364	1866	1902	2193	3600
5	300	2820	2166	**2117**	2536	3163	2743	2677	3023	3600
5	275	3432	2750	**2690**	2885	3462	2983	3134	3351	3600
5	250	3600	3600	3600	3600	3600	3600	3490	3600	3600
5	225	3600	3600	3600	3600	3600	3600	3600	3600	3600
5	200	3600	3600	3600	3600	3600	3600	3600	3600	3600

Table 2: Average CPU times in seconds for the F21-F40 set for different machine and capacity levels

M	CAP	SBC0	SBC1	SBC2	SBC3
2	700	63,6	22,7	4,9	**4,5**
2	675	48,6	**5,0**	6,8	5,7
2	650	224	13,0	**9,5**	27,4
2	625	169	16,8	**15,6**	55,9
2	600	257	116	**26,1**	60,1
2	575	1173	169	**106**	818
2	550	2933	1348	**974**	1862
2	525	3348	2309	**2024**	2467
2	500	3421	2890	**2732**	3077
2	475	3600	**3373**	3405	3600
2	450	3600	3600	3600	3600
3	700	**0,1**	0,4	0,5	0,2
3	675	**0,2**	0,5	0,5	0,2
3	650	2,0	0,8	1,2	**0,8**
3	625	1,8	**1,2**	1,8	1,6
3	600	9,7	**1,3**	1,3	4,2
3	575	252	**10,8**	22,5	57,5
3	550	521	8,9	**7,3**	19,5
3	525	696	36,7	**29,6**	264
3	500	1107	**31,8**	66,7	231
3	475	2330	**352**	433	1377
3	450	2482	**951**	965	1755
3	425	3373	1504	**1492**	2721
3	400	3600	2534	**2526**	3282
3	375	3600	3232	**3186**	3600
3	350	3600	**3462**	3600	3600
4	700	**0,1**	0,9	0,9	0,2
4	675	**0,1**	0,9	1,0	0,2
4	650	**0,2**	1,1	1,1	0,3
4	625	0,3	1,1	1,1	**0,2**
4	600	0,3	1,3	1,7	**0,3**
4	575	29,6	**4,7**	6,6	15,6
4	550	22,1	**3,1**	6,6	35,3
4	525	87,1	14,1	20,9	**3,0**
4	500	522	**10,6**	15,0	64,1
4	475	771	**217**	239	443
4	450	1330	406	**390**	1282
4	425	1640	**580**	748	1387
4	400	2076	**1246**	1285	1529
4	375	3255	**1826**	2035	2838
4	350	3522	2965	**2903**	3315
4	325	3600	3600	3600	3600
5	700	**0,2**	1,5	1,5	0,2
5	675	**0,2**	1,5	1,7	0,2
5	650	**0,2**	1,5	2,2	0,3
5	625	0,3	1,9	2,7	**0,3**
5	600	**0,3**	2,1	2,9	0,4
5	575	2,0	3,3	7,2	**1,0**
5	550	2,1	13,5	8,6	**1,1**
5	525	2,6	53,9	36,0	**1,8**
5	500	133	**36,7**	41,5	182
5	475	**234**	248	503	268
5	450	663	**473**	518	564
5	425	933	**546**	609	774
5	400	1296	**1089**	1242	1091
5	375	2645	2182	**2050**	2325
5	350	2958	**2579**	2688	2914
5	325	3426	**3411**	3416	3423
5	300	3526	3422	**3422**	3423
5	275	3600	3600	3600	3600
5	250	3600	3600	3600	3600

In order to obtain a better overall view, we summarize these numbers in Tables 3 and 4 by grouping machine-capacity levels according to their average CPU times for SBC0 and then taking the average. It is not possible to group the various machine-capacity levels according to their capacity tightness. In the presence of setup times, measuring the a priori tightness of the capacity constraint is not possible, as this would require a decision concerning the production and number of setups. When setup times are present, even the feasibility problem becomes NP-complete (Trigeiro et al. 1989).

Table 3: F1-F20 CPU time averages, grouped according to average time for SBC0

	Time								
	SBC0	**SBC1**	**SBC2**	**SBC3**	**SBC4**	**SBC5**	**SBC6**	**SBC7**	**SBC8**
0-1 s	**0,43**	2,4	3,2	0,58	12,0	5,1	7,6	215	155
1-10 s	5,0	2,7	3,4	**1,8**	41,0	10,9	15,6	159	242
10-100 s	34,6	5,9	8,3	**4,2**	158	34,5	91,0	409	438
100-1000 s	450	**133**	229	153	570	352	408	865	1388
1000-3600 s	2716	**1681**	1722	2099	2473	2050	2214	2490	3286
3600 s	3600	**3475**	3507	3585	3590	3508	3548	3580	3600

Table 4: F21-F40 CPU time averages, grouped according to average time for SBC0

	Time			
	SBC0	**SBC1**	**SBC2**	**SBC3**
0-1 s	**0,21**	1,2	1,5	0,24
1-10 s	3,4	12,3	9,4	**1,8**
10-100 s	50,2	9,9	**9,2**	12,8
100-1000 s	448	**144**	170	232
1000-3600 s	2547	1626	**1610**	2102
3600 s	3600	**3375**	3390	3560

From Tables 3 and 4, we see that for easy problems, i.e. the ones that take less than 1 second on average using model SBC0, it is better not to add any symmetry breaking constraints, as this only increases the formulation size and slows down the optimization process. For these easy problems, adding the few constraints in SBC3 only increases the CPU slightly, but adding other symmetry breaking constraints increases the times substantially. These easy problems correspond to the problems where a lot of capacity is available. Next we look at the problems that take on average between 1 and 10 seconds without any symmetry breaking constraints. For these problems, adding a small amount of symmetry breaking constraints (SBC3) improves the SBC substantially. For these problems, adding SBC1 and SBC2 reduces the time for data set F1-F20, but increases it for F21-F40. For all other problems, i.e. the ones taking more than 10 seconds on average with SBC0, adding SBC1, 2 or 3 results in large decreases in CPU time.

From Table 3, it is also clear that SBC4 to 8 are not effective in speeding up the CPU time. For the easy problems which take up to 10 seconds on average in the base case, adding these symmetry breaking constraints actually substantially increases the CPU time. Especially SBC 7 and 8 show an enormous increase in CPU time for the easier problems, compared to SBC0. These two symmetry breaking constraints almost always result in much larger CPU times for the complete data set. These two symmetry breaking constraints are the only ones from the proposed set that combine binary and continuous variables. Apparently, this leads to extra difficulties in the branch-and-bound, rather than speeding it up. For the more difficult problems, which take over 10 seconds on average in the base case, SBC5, and to a lesser extent SBC6 as well, seem to help to reduce the CPU time. As we concluded that the symmetry breaking constraints SBC4 to SBC8 are clearly inferior to SBC1 to SBC3, we decided not to include them in the experiments for data sets F21-F40.

Table 5: F1-F20 node averages, grouped according to average time for SBC0

	Nodes								
	SBC0	**SBC1**	**SBC2**	**SBC3**	**SBC4**	**SBC5**	**SBC6**	**SBC7**	**SBC8**
0-1 s	**81**	365	627	117	3015	1209	1576	57128	44352
1-10 s	3377	**523**	803	881	10171	2902	3768	41838	73748
10-100 s	25557	**1450**	2226	2468	34501	7686	18670	104949	165006
100-1000 s	279190	**27701**	52603	85546	117502	75691	82287	206309	425895
1000-3600 s	1468208	**353855**	389211	902469	492944	415743	435490	575332	1047993
3600 s	1212347	360813	382936	947468	349636	**340480**	340528	349403	631663

Table 6: F21-F40 node averages, grouped according to average time for SBC0

	Nodes			
	SBC0	**SBC1**	**SBC2**	**SBC3**
0-1 s	8	74	138	**7**
1-10 s	2191	2609	2141	**927**
10-100 s	41173	4924	**3485**	7988
100-1000 s	268890	**30269**	34997	114857
1000-3600 s	1318845	**307035**	308861	872390
3600 s	1284799	**447127**	494424	1015179

The symmetry leads to duplication of nodes in the branch-and-bound algorithm. In Tables 5 and 6, we provide the data on the average number of nodes. Just as in Table 3 and 4, the machine-capacity levels are grouped according to the average CPU time for SBC0. We observe that SBC3 is effective in reducing the number of nodes for some of the smaller problems, while SBC1 and SBC2 are effective in reducing the number of nodes in the more difficult problems. Clearly there is a trade-off in using symmetry breaking constraints: adding symmetry breaking constraints might result in fewer nodes, but the models grow larger and this might lead to a longer CPU time per node. Therefore, the total CPU times presented in Tables 3 and 4 provide the best overall indication of the effectiveness of the symmetry breaking constraints.

6. Conclusion

In this paper, we combine two important issues which need to be taken into consideration when modeling and solving Mixed Integer Programming problems. We specifically look at the lotsizing problem with identical parallel machines. The first issue relates to finding tight formulations for the problem. This issue is well discussed in the OR literature for various problems. For the lotsizing problem, a tighter formulation has been proposed by Eppen and Martin (1986) representing the lotsizing decisions in a network formulation. We extend this formulation to the case of multiple machines. Using a tighter formulation allows us to solve lotsizing problems from a standard test set, which otherwise would have been too time-consuming to solve with the regular formulation. The second issue we consider for the lotsizing problem on identical parallel machines is the issue of symmetry. This issue is generally not well covered in the discussion on good MIP modeling. However, we demonstrate that removing symmetry from the formulation is very helpful in speeding up computational times for our problem. This research is in line with previous research on symmetry in MIP formulations (Sherali and Smith 2001). We conclude that the lexicographic ordering constraints are quite effective and lead on average to a substantial decrease in computational time. Other symmetry breaking constraints that impose an ordering which is not necessarily unique are less effective and often even lead to higher CPU times due to the increase in model size. Symmetry breaking constraints that include both binary and continuous decision variables lead on average to a substantial increase in computational time.

Interesting areas for future research include the exploration of symmetry issues in other applications such as parallel machine job scheduling. For small bucket lotsizing models with identical machines, it would also be interesting to compare formulations with general integer variables versus binary assignment variables complemented with symmetry breaking constraints. More generally, it is worth investigating whether symmetry can be automatically detected and consequently whether these types of symmetry breaking constraints can be automatically generated within a branch-and-bound algorithm. Such symmetry breaking constraints should be used in addition to other reformulation and cutting planes.

References

Ahmadi, R.H., Dasu, S., Tang, C.S., 1992. The dynamic line allocation problem. *Management Science*, 38 (9), 1341-1353.

Belvaux, G., Wolsey, L.A., 2001. Modelling Practical Lot-Sizing Problems as Mixed-Integer Programs. *Management Science,* 47 (7), 993-1007.

Clark, A.R., Clark, S.J., 2000. Rolling-horizon lot-sizing when set-up times are sequence-dependent. *International Journal of Production Research,* 38 (10), 2287-2307.

Clark, A.R., 2003. Optimization approximation for capacity constrained material requirements planning. *International Journal of Production Economics,* 84, 115-131.

Dastidar, S.G., Nagi, R., 2005. Scheduling injection molding operations with multiple resource constraints and sequence dependent setup times and costs. *Computers & Operations Research*, 32, 2987-3005.

Degraeve, Z., Gochet, W., Jans, R., 2002. Alternative Formulations for a Layout Problem in the Fashion Industry. *European Journal of Operational Research*, Vol. 143 (1), 80-93.

De Matta, R., Guignard, M., 1994^{a}. Dynamic production scheduling for a process industry. *Operations Research,* 42 (3), 492-503.

De Matta, R., Guignard, M., 1994^{b}. Studying the effects of production loss due to setup in dynamic production scheduling. *European Journal of Operational Research,* 72, 62-73.

De Matta, R., Guignard, M., 1995. The performance of rolling production schedules in a process industry. *IIE Transactions,* 27, 564-573.

Dillenberger, C., Escudero, L.F., Wollensak, A., Zhang, W., 1994. On practical resource allocation for production planning and scheduling with period overlapping setups. *European Journal of Operational Research*, 75, 275-286.

Dos Santos-Meza, E., Dos Santos, M.O., Arenales, M.N., 2002. A lot-sizing problem in an automated foundry. *European Journal of Operational Research*, 139, 490-500.

Dumoulin, A., Vercellis, C., 2000. Tactical models for hierarchical capacitated lot-sizing problems with set-ups and changeovers. *International Journal of Production Research*, 38 (1), 51-67.

Eppen, G.D., Martin, R.K., 1987. Solving Multi-Item Capacitated Lot-Sizing Problems Using Variable Redefinition. *Operations Research* 35 (6), 832-848.

Hung, Y.F., Cheng, G.J., 2002. Hybrid capacity modeling for alternative machine types in linear programming production planning. *IIE Transactions*, 34, 157-165.

Jans, R. Degraeve, Z., 2004. Improved lower bounds for the capacitated lot sizing problem with setup times. *Operations Research Letters*, 32, 185-195.

Jans, R., Degraeve, Z., 2004. An Industrial Extension of the Discrete Lot Sizing and Scheduling Problem. *IIE Transactions* 36 (1), 47-58.

Jans, R., Degraeve, Z., 2007^{a}. Modeling Industrial Lot Sizing Problems: A Review. accepted for publication in *International Journal of Production Research.*

Jans, R., Degraeve, Z., 2007^{b}. Meta-heuristics for lot sizing problems: review and comparison of solution approaches. accepted for publication in the *European Journal of Operational Research.*

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.

Kimms, A., Drexl, A., 1998. Proportional Lot Sizing and Scheduling: Some Extenstions. *Networks*, 32, 85-101.

Leachman, R.C., Carmon, T.F., 1992. On capacity modeling for production planning with alternative machine types. *IIE Transactions*, 24 (4), 62-72.

Madan, M.S., Gilbert, K.C., 1992. An exact solution algorithm for a class of production planning and scheduling problems. *Journal of the Operational Research Society*, 43 (10), 961-970.

Meyr, H. 2002. Simultaneous lotsizing and scheduling on parallel machines. *European Journal of Operational Research*, 139, 277-292.

Özdamar, L., Barbarosoğlu, G., 1999. Hybrid heuristics for the multi-stage capacitated lot sizing and loading problem. *Journal of the Operational Research Society,* 50, 810-825.

Özdamar, L., Birbil, S.I., 1998. Hybrid heuristics for the capacitated lot sizing and loading problem with setup times and overtime decisions. *European Journal of Operational Research,* 110, 525-547.

Pochet, Y., Wolsey, L.A., 2006. *Production Planning by Mixed Integer Programming*. Springer, New York, 499 pages.

Salomon, M., Kroon, L.G., Kuik, R., Van Wassenhove, L.N., 1991. Some Extensions of the Discrete Lotsizing and Scheduling Problem. *Management Science*, Vol. 37 (7), 801-812.

Sherali, H.F., Smith, J.C., 2001. Improving Discrete Model Representations via Symmetry Considerations. *Management Science*, Vol. 47 (10), 1396-1407.

Stadtler, H., 2003. Multilevel lot sizing with setup times and multiple constrained resources: internally rolling schedules with lot-sizing windows. *Operations Research*, 51 (3), 487-502.

Trigeiro, W., L.J. Thomas, McClain, J.O., 1989. Capacitated Lot Sizing with Set-Up Times. *Management Science*, 35 (3), 353-366.

Vanderbeck, F., 1998. Lot-Sizing with Start-Up Times. *Management Science*, 1998, 44 (10), 1409-1425.

Wagner, H.M., Whitin, T.M., 1958. Dynamic version of the economic lot size model. *Management Science*, 5 (1), 89-96.

Wolsey, L.A., 2002. Solving multi-item lot-sizing problems with an MIP solver using classification and reformulation. *Management Science*, 48 (12), 1587-1602.

Publications in the Report Series Research* in Management

ERIM Research Program: “Business Processes, Logistics and Information Systems”

2006

Smart Business Networks Design and Business Genetics
L-F Pau
ERS-2006-002-LIS
http://hdl.handle.net/1765/7319

Designing and Evaluating Sustainable Logistics Networks
J. Quariguasi Frota Neto, J.M. Bloemhof-Ruwaard, J.A.E.E. van Nunen and H.W.G.M. van Heck
ERS-2006-003-LIS
http://hdl.handle.net/1765/7320

Design and Control of Warehouse Order Picking: a literature review
René de Koster, Tho Le-Duc and Kees Jan Roodbergen
ERS-2006-005-LIS
http://hdl.handle.net/1765/7322

A Theoretical Analysis of Cooperative Behavior in Multi-Agent Q-learning
Ludo Waltman and Uzay Kaymak
ERS-2006-006-LIS
http://hdl.handle.net/1765/7323

Supply-Chain Culture Clashes in Europe. Pitfalls in Japanese Service Operations
M.B.M. de Koster and M. Shinohara
ERS-2006-007-LIS
http://hdl.handle.net/1765/7330

From Discrete-Time Models to Continuous-Time, Asynchronous Models of Financial Markets
Katalin Boer, Uzay Kaymak and Jaap Spiering
ERS-2006-009-LIS
http://hdl.handle.net/1765/7546

Mobile Payments in the Netherlands: Adoption Bottlenecks and Opportunities, or… Throw Out Your Wallets
Farhat Shaista Waris, Fatma Maqsoom Mubarik and L-F Pau
ERS-2006-012-LIS
http://hdl.handle.net/1765/7593

Hybrid Meta-Heuristics for Robust Scheduling
M. Surico, U. Kaymak, D. Naso and R. Dekker
ERS-2006-018-LIS
http://hdl.handle.net/1765/7644

VOS: A New Method for Visualizing Similarities between Objects
Nees Jan van Eck and Ludo Waltman
ERS-2006-020-LIS
http://hdl.handle.net/1765/7654

On Noncooperative Games, Minimax Theorems and Equilibrium Problems
J.B.G. Frenk and G. Kassay
ERS-2006-022-LIS
http://hdl.handle.net/1765/7809

* A complete overview of the ERIM Report Series Research in Management:
https://ep.eur.nl/handle/1765/1

ERIM Research Programs:

LIS	Business Processes, Logistics and Information Systems
ORG	Organizing for Performance
MKT	Marketing
F&A	Finance and Accounting
STR	Strategy and Entrepreneurship