A dynamic programming algorithm to plan the number of lots to be manufactured

S. Hernández-González[1]*, I. Flores-de-la-Mota[2], M.D. Hernández-Ripalda[1], E. Segura-Pérez[3]

[1]Instituto Tecnológico de Celaya,
Antonio García Cubas s/n,
Celaya, Gto., C.P. 38010
*salvador.hernandez@itcelaya.edu.mx
[2] Posgrado de Ingeniería, UNAM
Ciudad Universitaria, México D.F.
C.P. 04510
[3] Facultad de Ingeniería, UNAM
Ciudad Universitaria, México D.F.
CP 04510

ABSTRACT
Wepresent a mathematicalmodelforcalculatingtheoptimum number of batches of a set of products to be manufactured in equipment with limited capacity. Said optimization model is integer nonlinear, and although it has been proposed since 1989, the relevant literature contains very few solution methods and these use Lagrangean relaxation and branch and bound to obtain the optimum. This paper presents the way to get the optimum solution by employing dynamic programming. An example is solved and the results of computational tests are given, making comparisons with a branch and bound algorithm. The results show that the runtime for the dynamic programming method is competitive, in particular for instances of 20 or more products. This method allows managers and planners to make an analysis of the decision and its effects on the use of the equipment's capacity, particularly those involved in chemical industry.

Keywords: production, management, dynamic programming, optimization.

1. Introduction

The people in charge of scheduling the production and the inventory control are responsible for taking the necessary decisions to answer three fundamental questions about their activity: what to manufacture?, how much to manufacture? And, when to manufacture?(Hadley and Within 1963; Sipper and Bulfin 1997).
As a tool for answering these questions, there is a model known as Economic Order Quantity (EOQ). The original model assumes that the demand is deterministic and constant throughout the period and that there are no constraints in the system; however all the systems are limited in terms of available resources: money available for investment, warehouse space or the capacity of the equipment. In these cases variants of the EOQ model with constraints have been developed.
One of these is the next one: companies from a variety of fields have processes based on standardized units, known as batches. For example, the chemical industry often schedules the manufacture of n products in equipment with limited capacity. Every material requires a certain time to be processed, for example, to achieve the desired concentration of the main material, moreover, there is a certain amount of time available in the reactor that can, for example, be expressed in hours a month.
The people responsible for scheduling production must decide how many batches to manufacture, based on demand and inventory and production costs, subject to the equipment's capacity. This problem is known as the Economic Number of Batches Problem (ENBP) (Sundararaghavan and Ahmed, 1989).
Using the EOQ model directly to solve this problem is the equivalent of separately scheduling each product, which does not ensure a solution that satisfies the capacity constraint, let alone an optimum solution. The calculation methods for getting the optimum solution for instances of the problem, taking into account the capacity constraint are based on lagrangean relaxation.

Unlike other papers, this one shows how to solve instances of the problem using dynamic programming and thus determine the economic number of batches to be manufactured from several products in equipment with limited capacity. This strategy makes it possible to solve the problem while directly considering the integer variables. Even though, in 1963, Hadley and Within in their now classic book had already proposed the use of dynamic programming for constrained inventory systems, to date no paper has specifically dealt with scheduling the number of batches.

We cite below the literature on this problem, followed by the solution strategy using dynamic programming for the ENBP model, solving some examples and presenting the results of the tests that were done.

This article is developed as follows: section 2 covers the history of inventory models and their solution methods; section 3 is devoted to the optimization model; an application example is presented in section 4; our conclusions are given section 5 and, finally, section 6 presents the references used in this paper.

2. Background

In order to solve inventory models with a constraint, the textbooks recommend the method of Lagrange multipliers to obtain the order quantity (Sipper and Bulfin 1997). The papers of Ziegler (1981), Rosenblat (1981), Ventura and Klein (1988), Rosenblat and Rothblum (1990) and Maloney and Klein (1993) reported improvements to the method, mainly in respect of calculating bounds for the Lagrange multipliers.

It is worth mentioning that variants have been proposed for both the models and the solution techniques, so this is a good moment some briefly talk about lines of action:

1. Planning of n product with common order cycles or ranges. These types of models are in keeping, for example, with systems where the purchase of the products is only carried out and where the demands for each product and, in consequence, the quantities Q to be ordered are very different from one another. Page and Paul (1976) observed that under these assumptions, the multipliers method does not ensure that the optimum will be obtained: if there is a large difference between the quantities Q of products then the inventory will reach maximum levels where the constraint may even be violated, so they proposed an improvement consisting of grouping the products into common order cycles in order only to order the quantities of those products that are considered in a given cycle. It is also worth mentioning the papers of Goyal (1978), Lee (1994) and, more recently, Boctor (2010).

2. Planning of n products with individual cycles that are multiples of a basic cycle. In this approach, the quantity to be ordered depends on the individual cycle of the product and this is expressed as a multiple of a basic cycle: $T_i = k_i T$. Literature on this line of action abounds. We recommend the papers of Elmagharaby (1978), Khouja, and Goyal (2008) and Hernández, Flores and Vazquez (2012) for reviews of the most recent proposals. Several exact methods based on enumeration and global optimization techniques have been developed, while heuristic methods have also been proposed and meta-heuristic techniques implemented.

3. Planning of n products where the quantities are independent of the individual cycle. This paper falls within this line and below we shall discuss its contributions.

In 1989, Sundararaghavan proposes the ENBP model with a single capacity constraint, that can be applied to batch production systems, as are commonly used in companies in the chemicals industry: pigments, resins, paints are some examples. The proposal consists of a heuristic algorithm to obtain the number of batches that minimizes the cost. Said algorithm uses Lagrangean relaxation to determine the lower bound of the decision variables and later heuristically rounds up or down to obtain the number of batches.

In 1994, Brettahuer, Sidharta and Sethi propose a general model for production planning and inventory control. Among the problems with said general structure that are presented is Sundararaghavan's, but they also show others with formulations that are specific cases of the general model, such as economic quantity programming or the stratified sampling problem. They propose the use of Lagrangean relaxation as a solution method for cases where the decision variables are continuous, while for integer variables they propose a heuristic method to obtain a good solution for the problem, starting from the relaxed

solution and then using branch and bound to obtain the optimum solution.
They do the tests by solving an instance that is similar to the one presented by Sundararaghavan and with the order quantity (OQ) model. It is worth mentioning the fact that the article establishes the possibility of applying the procedure to models of the second line of action mentioned above, however they do not solve any instance of said class or specify the way to do it.
In Bretthauer, Shetty, Syam and Vokurka (2006) the 1994 results are extended to problems with several constraints, establishing the optimality conditions for the relaxed problem and using branch and bound to solve instances of the problem, as well as doing computational tests.
In Haksever and Moussourakis (2005) propose a mixed nonlinear optimization model in which, apart from the capacity constraints, they incorporate constraints for the number of orders, the planning approach to be employed (fixed time cycles for all the products or else independent cycles) as well as the batch size.
Chen & Chen (2010) proposed a procedure to calculate the required number of machines with serial and batch processing characteristics, respectively. They proposed heuristic algorithms to obtain lower and upper bounds and a near-optimal of the number of machines with capability constraint.
It is worth mentioning that we did not, in our review of the literature, find any other research on the model covered here, at least, no specifically. Whereas, a large number of papers have been devoted to the second line of action, even though the techniques require a greater computational effort (Rosenblatt, 1981).
We end this section mentioning that in our case demand is deterministic, but there are other approaches where demand is considered stochastic; examples are papers by Yang, Yuan, Wuec&, Zhoud (2014) where they considered a single-item periodic-review batch ordering inventory system with the consideration of the setup cost and the capacity constraint for each order over a finite planning horizon; in Lin, Chen & Chu (2014) authors developed a stochastic dynamic programming (SDP) model with an embedded linear programming (LP) to generate a capacity planning policy as the demand in each period is revealed and updated.

3. Optimization model

There is a list of products $i = 1,2,...,K$ which should be scheduled in a piece of equipment that has limited capacity *T*. The batch of each product requires a manufacturing time t_i, there is also a preparation cost a_i and a warehouse (commonly named inventory or carrying) cost h_i. The time that elapses between the preparation of the machine for two products is zero.
The number of batches of a product is the quotient: where Q_i is the quantity of orders and D_i is the average demand and constant of product *i*. The cost of preparing the equipment for the production of a batch is given by:

$$C_{P,i}(n_i) = \frac{a_i D_i}{Q_i} = n_i a_i \tag{1}$$

The inventory cost is given by:

$$C_{A,i}(n_i) = \frac{h_i Q_i}{2} = \frac{D_i h_i}{2n_i} \tag{2}$$

The total individual cost is the sum of the inventory and preparation cost:

$$TC_i(n_i) = C_{P,i} + C_{A,i} = n_i a_i + \frac{D_i h_i}{2n_i} \tag{3}$$

For a system with $i = 1,2,...,k$ products, the total cost of the production run shall be:

$$TC = \sum_{i=1}^{K} TC_i(n_i) = \left[n_1 a_1 + \frac{D_1 h_1}{2n_1} \right] + \left[n_2 a_2 + \frac{D_2 h_2}{2n_2} \right] + \ldots + \left[n_K a_K + \frac{D_K h_K}{2n_K} \right] \tag{4}$$

The total equipment hours consumed is the sum of the time consumed in the preparation of each batch:

$$\sum_{i=1}^{K} \left[t_i \frac{D_i}{Q_i} \right] = \left[t_1 n_1 + t_2 n_2 + \ldots + t_K n_K \right] \le T \tag{5}$$

The person in charge of production planning must schedule at least one batch of each product, therefore $n_{i,\inf} = 1$, therefore the range for the number of batches shall be $1 \le n_i$ for the $i = 1,2,...,K$ products, and the batches will also be supposed to be integer quantities. The problem of scheduling the number of batches with a capacity constraint can be stated as follows:

Minimize:

$$TC = \left[n_1 a_1 + \frac{D_1 h_1}{2n_1}\right] + \left[n_2 a_2 + \frac{D_2 h_2}{2n_2}\right] + \dots + \left[n_K a_K + \frac{D_K h_K}{2n_K}\right] \quad (6)$$

Subject to

$$\left[t_1 n_1 + t_2 n_2 + \dots + t_K n_K\right] \le T \quad (7)$$

$1 \le n_i \le n_{i,\sup}$ and integer $\forall i = 1,2,...,K$ (8)

This is a nonlinear integer model where (6) is the cost function, which must be minimized, (7) corresponds to the equipment capacity constraint (hours) and, finally, (8) constrains the number of batches that can be manufactured and is an integer variable. Nonlinear integer models belong to the set of NP- hard problems (Martello &Toth, 1990, Li & Sun, 2006).

3.1 Solution employing dynamic programming.

The dynamic programming method was proposed in the 1950s by Richard Bellman and is based on the principle of optimality: the property of an optimal policy is that irrespective of the initial state and the decisions of the subsequent states shall constitute an optimal policy in respect of the state resulting from the first decision (Bellman 1957).

In order to apply the dynamic programming (DP) method, we must define the following essential aspects: the main problem must be decomposed into smaller sub-problems, which are known as stages, associated with each sub-problem or stage there will be a set of states and a set of feasible decisions, a state shows the consequences of some action or decision.

The recursive optimization process, in other words, the steps that must be executed in order to solve each stage or sub-problem in an optimum manner, also has to be defined. Said process can be backwards (when starting the analysis in the last stage) or forwards (when started in the first stage) (Bellman 1957, Bradley, Hax and Magnanti, 1977). It is possible to implement this recursion process by defining the recursive equations and the boundary condition.

In order to schedule the manufacture of a set of products, it is necessary to decide the number of batches to be manufactured in one piece of equipment subject to the capacity constraint given for example in hours. We shall assume the products are scheduled in the same order and the decision process is sequential. Each product *i* shall be a stage in the solution process.

The scheduling process shall start with the last product *K*, then product $K-1$ is scheduled, followed by the product $K-2$ until product 1 is reached. When scheduling product *i*, the manufacture of products $i+1, i+2, \dots, K$ has been decided and there shall be a remnant of *u* equipment resource units which is an integer value, as the manufactured batches are integer units (Denardo, 2003).

The available equipment hours shall be found in the range $0 \le u \le T$ and a cost shall be incurred in the scheduling that can be expressed as follows:

$f_i(i,u) =$ Manufacturingcost(inventory and preparation)of the products $i+1.i+2, \dots, K$ using the *u* hours available on the equipment.(9)

Only a set of production programs is feasible because they satisfy the capacity constraint posed for the optimization model, this constraint is posed as follows:

$$t_i n_i \le u \quad (10)$$

Starting the schedule with product *K*, we have:

$$f_K(K,u) = \left(n_K a_K + \frac{D_K h_K}{2n_K}\right) \quad (11)$$

Given that we want to find the minimum, then we have:

$$f_K(K,u) = \min\left(n_K a_K + \frac{D_K h_K}{2n_K}\right) \quad (12)$$

Subject to:

$$1 \le n_K \le n_{i,\max} = \left\lfloor \frac{T}{t_i} \right\rfloor \quad (13)$$

$$t_K n_K \le u \quad (14)$$

Continuing the backward programming until the product $i(i < K)$, there are at least two products that should be programmed for their manufacture. When there are u available hours, the cost of manufacturing n_i batches shall consume $u - t_i n_i$ hours, this is expressed in the following equation:

$$\left(n_i a_i + \frac{D_i h_i}{2n_i} \right) + f_{i+1}(i+1, u - t_i n_i) \qquad (15)$$

The first term is the cost of manufacturing product *i*, the second is the cost of manufacturing product $i+1$ when there are $u - t_i n_i$ available hours. Given that we are looking for the production schedule that minimizes the cost, we have:

$$f_i(i,u) = \min \left[\left(n_i a_i + \frac{D_i h_i}{2n_i} \right) + f_{i+1}(i+1, u - t_i n_i) \right] \qquad (16)$$

Subject to:

$$1 \le n_K \le n_{i,\max} = \left\lfloor \frac{T}{t_i} \right\rfloor \qquad (17)$$

$$t_K n_K \le u \qquad (18)$$

Where, (16) is known as the recursive equation, and through these the stages *i* of the problem are linked and, in each stage the possible states $u = 0,1,2,\ldots,T$ of the available hours and each feasible value of n_i are analyzed. The total cost of each stage is calculated as follows: We select the decision that gives the minimum cost in each stage *i* of the sum of the manufacturing cost for n_i batches of product *i* plus the cost of manufacturing product $i+1$ when there are $u - t_i n_i$ available hours, taking into account that the $n_{i,\max} = \lfloor T/t_i \rfloor$ ratio establishes the maximum number of batches that can be manufactured, where $\lfloor T/t_i \rfloor$ is the maximum integer less than $\frac{T}{t_i}$.

4. Example of application (Sundararaghavan, 1989)

Suppose we wish to schedule the manufacture of three products in a piece of equipment with limited capacity for the number of available hours. The data for the demand D_i, the cost of inventory h_i, the preparation cost a_i, the lower bound of the number of batches n_{min}, the resource capacity T, and the required hours T_i for the production of each batch is given in the table 1.

D	3000	5000	8000
h	20	30	15
a	800	500	500
T_i	20	18	10
n_{min}	1	1	1
T	300		

Table 1.The problem's data (Sundararaghavan, 1989)

Each article represents a stage for which the model consists of three stages, then for $i = 4$ we have the boundary condition given by $f(4,u) = 0$. Backward calculations for the analysis for the product 3 ($i = 3$) gives us the following table:

(a)

(*u*)	$f_3(n_3) + f_4(4, u - t_3 n_3)$	*n**
10	60500	1
20	31000	2
30	21500	3
40	17000	4
50	14500	5
60	13000	6
70	12071.42	7
80	11500	8
90	11166.66	9
100	11000	10
110	10954.54	11
120	11000	12
130	11115.38	13
140	11285.71	14

(b)

(*u*)	$f_3(n_3)+f_4(4,u-t_3n_3)$	*n**
150	11500	15
160	11750	16
170	12029.41	17
180	12333.33	18
190	12657.89	19
200	13000	20
210	13357.14	21
220	13727.27	22
230	14108.69	23
240	14500	24
250	14900	25
260	15307.69	26
270	15722.22	27
280	16142.85	28
290	16568.96	29
300	17000	30

Table 2. Calculations for $i=3$. (a) *u*=(10-140);

(b) *u*=(150 – 300)

For questions of space, we only show states $u=10,20,\ldots 300$ (multiples of t_3) to visualize part of the calculation. We can observe that the minimum capacity for scheduling a batch of product 3 is 10 hours, therefore for states $u=0,1,2,\ldots 9$ there are no feasible solutions.

Tables3 and 4 show the calculations corresponding to products 2 and 1. One of the advantages of a dynamic programming model in the form of a table, is that its analysis facilitates the visualization of a decision for the production programmers, as well as having optimum solutions in each stage, for which the table corresponding to product 2 shall be analyzed.

It must once again be pointed out that, by way of illustration, the states are shown in multiples of t_i so as to be able to visualize part of the calculations. In the case of product 2 (Table 3), when 18 hours are available at least one product can be manufactured, however this consumes all the available capacity so it would not be possible to schedule a batch of product 1, therefore this entry corresponds to a solution that is not feasible.

Likewise, for the case where there are 18 available hours, then the manufacture of a batch of product 2 can be scheduled, dedicating the rest (18 hours) to the manufacture of a batch of product 3, however it is not feasible to schedule two batches of product 2 since once again all the available capacity would be consumed.

Finally the maximum number of batches that could be manufactured is 15, since a schedule of 16 batches consumes all the capacity and we must schedule yet another product: number 3. In the case of product 1 (Table 4), a feasible schedule can contain, as a maximum, 13 batches, as two other products must be scheduled: 2 and 3.Table 5 shows the solution obtained by dynamic programming compared to the one reported in Sundararaghavan (1989).

The solution reported by Sundararaghavan in 1989 using his heuristic method is $36414.28, using the dynamic programming method we get a cost solution of $36080.90, which is 0.9155% less (Table 5). The number of batches differs in product 3, the solution obtained using the heuristic procedure is 8 batches, and with dynamic programming, the solution points to producing 9 batches.

Finally, the solution obtained using the heuristic procedure requires 286 hours or 95.33% of the total capacity, whereas the solution obtained with dynamic programming requires 98.66% of the total capacity. The difference between both solutions is 3.378%.

	$f_2(n_2)+f_3(3,u-t_2n_2)$																
u	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	$f(n*)$	*n**
18																	
36	136000															136000	1
54	97000	99000														97000	2
72	90000	60000	87000													60000	3
90	87571	53000	43500	81250												43500	3
108	86667	50571	41000	42250	78000											41000	3
126	86500	49667	38571	35250	39000	76000										35250	4
144	86500	49500	37667	32821	32000	37000	74714									32000	5
162	86786	49500	37500	31917	30500	30000	35714	73875								30000	6
180	87250	49786	37500	31750	28667	26500	28714	30375	73333							26500	6
198	87833	50250	37786	31750	28500	26667	26286	27875	34333	73000						26286	7
216	88158	50833	38250	32036	28500	26500	25381	25446	27333	34000	72818					25381	7
234	88857	51158	38833	32500	28786	26500	25214	24542	24905	27000	33818	72750				24542	8
252	89609	51857	39158	33083	29250	26786	25214	24375	24000	24571	26818	21500	72769			21500	12
270	90400	52609	39857	33408	29833	27250	25500	24375	23833	23667	24390	26750	33769	72857		23667	10
288	91222	53400	40609	34107	30158	27833	25964	24661	23833	23500	23485	24321	26769	33857	73000	23485	11

Table 3. Calculationsfor $i=2$

	$f_1(n_1)+f_2(2,u-t_1n_1)$														
u	1	2	3	4	5	6	7	8	9	10	11	12	13	$f(n*)$	*n**
300	54466.6	38100	36941.67	36080.9	36285.7	36300	41885.7	45400	51533.3	54500	71527.2	109100	148707.7	36080.9	4

Table 4. Calculationsfor $i=1$

	*n**				
Product	1	2	3	*Capacity (hrs)*	*Cost ($)*
DP	4	7	9	296	36080.9
Sundararaghavan (1989)	4	7	8	286	36414.2

Table5. Solutions obtained with DP and Sundararaghavan's heuristic

5. Computational tests

A series of computational tests were done to empirically study the behavior of the algorithm, mainly in respect of the size of the instance to be solved. A variety of problems were solved in this case, where the demand parameters (*D*), inventory cost (*h*), preparation cost (*a*), hours per batch (t_i) and capacity of the equipment (*T*) were randomly generated using a uniform distribution function (*U*). Only 5 problems were generated for each value of the size of products (*n*). Table 6 shows the ranges used for the generation and following the guidelines given in other papers (Bretthauer, Shetty, Siam and White 1994, Sundararaghavan and Ahmed 1989).

Parameter	*Interval*
Number of products(*n*)	3,5,10,20,30, 50
Demand (D_i)	*U* (4000- 9000)
Inventorycost (h_i)	*U* (5 – 25)
Preparationcost (a_i)	*U* (2000 – 5000)
Resource consumption per batch (t_i)	*U* (1 – 5)
Availablecapacity (*T*)	*U* (300 – 800)

Table 6. Ranges for randomly generating tests parameters

The dynamic programming algorithm was encoded in Fortran 94 language, the optimization model was built in LINGO (equations 6 to 8) for the sake of comparisons and the branch and bound algorithm was employed to solve the instances. The runs were solved on a computer with a 2.5GHz Intel Core i5 processor and 4 G of RAM.

The runtime required for the DP algorithm was recorded and, in the case of branch and bound method, the time reported by the LINGO package was recorded. It is worth mentioning the following: in some prior tests with the optimization model we observed that the branch and bound algorithm does not return any solution for instances of 50 products after iterating for 60 minutes, therefore it was decided, in the case of the formal experiments, to stop running the algorithm at a maximum time of 60 minutes.Tables 7, 8 and 9 show the results obtained for the instances with 3, 5, 10, 20, 30 and 50 products.

In the case of instances with *n*=3 and 5 products (Table 7), both the dynamic programming model and the branch and bound method reached the optimum solution within the established time limit. Dynamic programming takes under 1 second for instances with *n* =3 products and about 0.01 seconds when *n* =5 products and the branch and bound method takes under 1 second.

	3 products			
Test	*DP*	*Exec. time (sec.)*	*B-B algorithm*	*Exec. time (sec.)*
1	135288	<0.01	135288	<0.01
2	140721	<0.01	140721	<0.01
3	132968	<0.01	132968	<0.01
4	137897	<0.01	137897	<0.01
5	119285	<0.01	119285	<0.01
	5 products			
1	238101	0.01	238101	<0.01
2	215974	<0.01	215974	<0.01
3	215243	0.01	215243	<0.01
4	229473	0.01	229473	<0.01
5	200997	0.01	200997	<0.01

Table 7. Tests results with 3 and 5 products

In the case of problems with *n*=10 (Table 8), the dynamic programming model requires at most 0.03

seconds, the branch and bound method takes under one second.
When *n* =20 (Table 8), the dynamic programming model finds the optimum solution in average of 0.0325 seconds, whereas we start to observe that the branch and bound method takes more time to find the solution, employing up to 4 seconds to get the optimum solution.
When *n* =30 and 50 products (Table 9), the dynamic programming algorithm takes an average of 0.070 and 0.246 seconds respectively. The branch and bound method takes up to 202 seconds when *n* =30. The case of *n* =50 was particularly interesting, as the runtime was used up in every case without the branch and bound algorithm achieving the optimum solution.
Figure 1 shows the speed in growth of the runtime for the dynamic programming algorithm in respect of the number of products (*n*); we can see that said growth is fast, however, the experiments showed that, in all the tests (including the 50-product tests), the optimum for the problem was found within the established time, unlike the branch and bound algorithm.

	10 products			
Test	*DP*	*Exec. time (sec.)*	*B-B algorithm*	*Exec.time (sec)*
1	431925	0.01	431925	<0.01
2	400276	0.01	400276	<0.01
3	446447	0.01	446447	<0.01
4	524708	0.01	524708	<0.01
5	426975	0.03	426975	<0.01
	20 products			
Test	*DP*	*Exec. time (sec.)*	*B-B algorithm*	*Exec.time (sec)*
1	968149	0.03	968149	3
2	900653	<0.01	900653	3
3	849028	0.03	849028	3
4	948993	0.06	948993	<0.01
5	943513	0.01	943513	4

Table 8. Test results with 10 and 20 products

	30 products			
Test	*DP*	*Exec.time (sec.)*	*B-B algorithm*	*Exec. Time (sec.)*
1	1340785	0.05	1340785	74
2	1351117	0.06	1351117	54
3	1215526	0.15	1215526	202
4	1790637	0.02	1790637	14
5	1362124	0.07	1362124	32

	50 products			
Test	*DP*	*Exec.time (sec.)*	*B-B algorithm*	*Exec.time (sec.)*
1	2361754	0.18	No Sol.	> 3600
2	3220948	0.05	No sol.	> 3600
3	2328213	0.28	No sol.	> 3600
4	2517709	0.19	No sol.	> 3600
5	2243616	0.53	No sol.	> 3600

Table 9. Test results with 30 and 50 products

Figure 1. Execution time vs. number of products(n)

6. Conclusions

A variant of the production planning problems consists of determining the quantity of batches of a set of products that have to be manufactured on equipment with limited capacity.
The mathematical model is of the non-linear integer type as the decision variables are entities that are generally indivisible. Although several solution proposals have been developed, these need the problem to be relaxed and then, by applying some technique, the integer solution of the problem is obtained. In this paper shows the way to find the optimum solution for the problem of scheduling the production for the number of batches in equipment with limited capacity, using dynamic programming.
The computational tests show that the DP method requires a similar runtime to the branch and bound method for instances of 3, 5 and 10 products,

whereas for 20 and 30 products the dynamic programming algorithm finds the solution faster and in the case of 50 only dynamic programming finds a solution within the maximum established runtime for the experiments (60 minutes).
When put in the form of a table, the dynamic programming algorithm facilitates the visualization, in this case, of the effects of the production-scheduling decisions, specifically in the consumption of capacity, as well as the feasible production schedules and getting the optimum in each stage.
It is a well-known fact that one of problems with the DP method is that when the number of constraints increases, the complexity and the number of required calculations also increase (dimensionality), however this analysis strategy is a tool that can be used during the first solution phases in these particular types of problems. The behavior remains to be studied of the dynamic programming algorithm using the bounds of the decision variables that are obtained when the relaxed problem is solved.

References

Bellman R.,1957.Dynamic Programming, Princeton University Press.

Boctor F.F., 2010. Offsetting inventory replenishment cycles to minimize storage space. Eur. J.Oper. Res., 203(2), 321–325.

Bradley, S.P., Hax, A.C., Magnanti, T.L., 1977.Applied Mathematical Programming, Addison- Wesley.

Bretthauer, K.M., Shetty, B., Syam, S., White, S., 1994. A Model for Resource Constrained Production and Inventory Management, DecisionSci., 25(4), 561 – 577.

Bretthauer, K.M., Shetty, B., Syam, S., Vokurka, R.J., 2006.Production and inventory management under multiple resource constraints.Math.Comput.Model., 44(1-2), 85-95.

Chen, J. & Chen, C., 2010 Capacity planning of serial and batch machines with capability constraints for wafer fabrication plants. Intl.J. Prod. Res., 48(11), 3207 – 3223.

Denardo, E., 2003. Dynamic programming: models and applications. Dover Publications Minneola, New York.

Elmaghraby, S.E., 1978. The Economic Lot Scheduling Problem (ELSP): Review and Extensions. Manage. Sci., 24(6), 587-598.

Goyal, S.K., 1978.A Note on "Multi-Product Inventory Situations with One Restriction".J.Oper. Res. Soc., 29(3), 269-271.

Hadley, G., Within, T.M., 1963. Analysis of inventory systems.Prentice-Hall. New Jersey.

Haksever, C., Moussourakis, J. 2005.bA model for optimizing multi-product inventory systems with multiple constraints. Int. J. Prod. Econ., 97(1), 18–30.

Hernández, S., Flores, I., Vázquez, J.A., 2012. Improved golden section algorithm for the multi-item replenishment problem. JART, 10(3), 388 - 397.

Khouja, M., Goyal, S., 2008.A review of the joint replenishment problem literature: 1989-2005. Eur. J.Oper. Res., 186(1), 1-16.

Lee, W.J. 1994. Optimal order quantities and prices with storage space and inventory investment limitations.Comput. Ind. Eng., 26(3), 481–488.

Li, D., Sun, X., 2006.Nonlinear integer programming.Springer, New York.

Lin, J.T., Chen, T.L., Chu, H.C., 2014.A stochastic dynamic programming approach for multi-site capacity planning in TFT-LCD manufacturing under demand uncertainty.Int. J. Prod. Econ., 148(2), 21–36.

Maloney, B., Klein, C., 1993. Constrained multi-item inventory systems: an implicit approach.Comput.Oper. Res.,20,.639–649.

Martello, S., Toth, P., 1990.Knapsack problems.Wiley, West Sussex.

Yang, Y., Yuan, Q., Xue, W., Zhoud, Y., 2014. Analysis of batch ordering inventory models with setup cost and capacity constraint. Int. J. Prod. Econ.,155(9), 340–350.

Page, E.,Paul, R.J., 1976.Multi-Product Inventory Situations with One Restriction.Oper. Res.Qart., 27(4), 815-834.

Rosenblatt, M., 1981.Multi-item system with budgetary constraint: a comparison between the lagrangian and the fixed cycle approach. Int. J. Prod. Res., 19(4), 331–339.

Rosenblatt, M., Rothblum, U., 1990.On the single resource capacity problem for multi-item inventory systems.Oper. Res., 38(4), 686–693.

Sipper, D., Bulfin, R.L., 1997. Production: planning, control and integration. McGraw-Hill. New York.

Sundararaghavan, P.S., Ahmed, M.U., 1989.Inventory control and scheduling of multiple products in a common facility, Decision Sci., 20(3), 476 – 491.

Ventura, J., Klein, C., 1988.A note on multi-item inventory systems with limitedcapacity.Oper. Res. Lett., 7(2), 71–75.

Ziegler, H., 1982.Solving certain singly constrained convex optimization problems in production planning.Oper. Res. Lett.1(6), 246–252.