Cátedra Nissan

-PROTHIUS-

Heuristics and exact algorithms for solving the Monden problem

Joaquín Bautista Valhondo, Ramón Companys Pascual y Albert Corominas Subias

WP-02/2010
(Rec. DIT 93/13- BCC - 1993)

Departamento de Organización de Empresas

Universidad Politécnica de Cataluña

Publica:

Universitat Politècnica de Catalunya
www.upc.edu

Edita:

Cátedra Nissan
www.nissanchair.com
director@nissanchair.com

TITOL:

HEURISTICS AND EXACT ALGORITHMS FOR SOLVING THE MONDEM PROBLEM

Autors: J. Bautista
R. Companys
A. Corominas

Document Intern de Treball
(D.I.T.) 93/13

Barcelona, maig 1993

Departament d'Organització d'Empreses
Universitat Politècnica de Catalunya

Trabajo realizado dentro del Proyecto de Investigación
PB-0504 (DGICYT)

HEURISTICS AND EXACT ALGORITHMS FOR SOLVING THE MONDEN PROBLEM

J. Bautista, R. Companys & A. Corominas
Department of Management
Universitat Politècnica de Catalunya

ABSTRACT: Sequencing units on an assembly line in order to obtain a regular consumption of resources is a problem that can be modelized in different ways. One of the most popular is known as MONDEN problem, and the heuristic proposed to obtain a "satisfactory" solution is called "goal chasing" method. In the paper is shown the myopic behaviour of this heuristic, and some improvements are proposed. An exact procedure, based in BDP, is also proposed. Relaxing the assumptions the BDP procedure becomes a new powerful heuristic. A sample of computational results is included.

KEY WORDS: SEQUENCING UNITS ON ASSEMBLY LINES, JIT, BOUNDED DYNAMIC PROGRAMMING

1. INTRODUCTION.

The sequencing of units flowing through an assembly line in order to regulate the consumption of necessary resources (work at the assembly stations or components) is a problem that has received attention in the literature for many years. Recent works have been published by COMPANYS (1989), MILTEMBURG (1989), SUMICHRAST and RUSSELL (1990), KUBIAK and SETHI (1991), YANO and RACHAMADUGU (1991), etc.

A variant of this problem was presented by MONDEN (1983) together with a heuristic to solve it. Our experimentation with this heuristic has shown that in many cases the results were far away from the desired objective. We have therefore experimented with other heuristics and a procedure based on "bounded dynamic programming" (BDP) which may lead to optimal solutions.

This paper is structured as follows: we explain the problem and show that it is equivalent to seeking a minimum path in an associated graph; we then describe several heuristics and the results obtained in a sample problem; we introduce the procedure based on BDP and its behaviour in the chosen problem; finally, we draw some conclusions. In annexes we describe two auxiliary algorithms and a sample of computational experience.

2. APPROACH TO THE PROBLEM

U homogeneous, but not totally identical, units must proceed through an assembly line at a constant rate for a given period (one day or one shift). There are P variants or distinguishable

products and u(i) is the number of identical units of the variant i (i=1,2,...,P) that form part of the U, such that

$$\sum_{i=1}^{P} u(i) = U \qquad (1)$$

A traditional problem consists in looking for the "most regular" sequence of units, i.e. that in which the variants are "best" distributed. Some variants may require more work than others in some stations of the line (because they incorporate special options, for example). If the line has been balanced bearing in mind the average load represented by the mix of variants, when a "rich" variant passes this station it will represent a work overload, compensated by the smaller load of the "poorer" units. However, in the short term if several "rich" units with relation to a station accumulate in a part of the sequence, in practice this compensation cannot be made and some units may abandon the station without all the programmed operations being performed in them. These units will have to be completed later, off-line or at special stations, with the extra tangible and intangible cost that this involves.

Although the problem is clear, the formalization of what is understood as the "regularity" of a sequence involves difficulties due to its ambiguity, and different approaches have been taken. We will focus here on that adopted by MONDEN (1983). The units incorporate critical components along the line in variable quantities that depend on the variant. The total number of types of critical components is C, and n(j,i) represents the number of components j that are incorporated in a unit of the variant i (j=1,2,..,C; i=1,2,..,P). Let us suppose initially that n(j,i) is a non-negative integer, and the matrix N of components n(j,i) is that which Vaszonyi calls the GOZINTO matrix.

It is easy to define the regular consumption of components. The total number of components j necessary in the period considered is:

$$T(j) = \sum_{i=1}^{P} n(j,i)*u(i) \qquad (2)$$

and the rate of consumption per unit entering the line is:

$$r(j) = T(j)/U \qquad (j=1,2,..,C) \qquad (3)$$

A sequence whose consumption of the component j was regular would mean that the first k (k=1,2,....,U) units of the sequence consume:

$$k*r(j) \qquad (4)$$

units of the component. Let us suppose that among the first k units of the sequence there are

x(1,k) of variant 1, x(2,k) of variant 2, ..., x(P,k) of variant P, x(i,k) being a non-negative integer no greater than u(i) such that:

$$\sum_{i=1}^{P} x(i,k) = k \qquad (5)$$

The number of units y(j,k) of the component j consumed will be:

$$y(j,k) = \sum_{i=1}^{P} n(j,i)*x(i,k) \qquad (6)$$

expression (2) is the adaptation of (6) to k=U:

$$T(j) = y(j,U) \qquad (7)$$

since obligatorily x(i,U)=u(i). If we define the column vector (P x 1) X(k) of components x(i,k) and the column vector (C x 1) Y(k) of components y(j,k), (6) is equivalent to the array-type relation of Vaszonyi:

$$Y(k) = N*X(k) \qquad (8)$$

The value y(j,k) does not have to be equal to k.r(j), and in some cases it cannot be for any sequence (when k.r(j) is a rational non-integer). We define the vectors (C x 1) R(k) of components r(j,k) (j=1,2,...,C). It is natural to define the regularity of the sequence in position k by the distance between Y(k) y R(k), and the regularity of the sequence by the distance between the matrices:

[Y(1) Y(2) .... Y(U)] and [R(1) R(2) ..... R(U)]

Though it is not the only possibility, a convenient way of measuring the distance is that based on quadratic differences:

$$SDQ = \sum_{k=1}^{U} \sum_{j=1}^{C} (y(j,k) - k*r(j))^2 \qquad (9)$$

although MONDEN uses the variant

$$SDE = \sum_{k=1}^{U} SQR \left[\sum_{j=1}^{C} (y(j,k) - k*r(j))^2 \right] \qquad (10)$$

Apart from some numerical simplifications that (9) will allow and (10) will not, in the search

for the optimal expression both expressions (and many others) permit the approach and the algorithms that we will develop below. In particular it is also interesting to measure the distance by the expression:

$$SDR = \sum_{k=1}^{U} \left[\sum_{j=1}^{C} ABS\ (y(j,k) - k*r(j)) \right] \qquad (11)$$

Our problem is reduced to seeking a sequence that minimizes (9).

2.1. Regularity in position k

If we were only interested in the regularity in a position k, the problem could be formulated as an integer quadratic program (12):

$$z(k) = [MIN] \sum_{j=1}^{C} (y(j,k) - k*r(j))^2$$

s.t.

$$y(j,k) = \sum_{i=1}^{P} n(j,i)*x(i,k) \quad j=1,2,..,C \qquad (12)$$

$$\sum_{i=1}^{P} x(i,k) = k$$

$$0 \le x(i,k) \le u(i) \qquad i=1,2,...,P$$

$$x(i,k) \text{ integer}$$

and the sum of the optimum values z(k) would be a lower bound of SDQ:

$$SDQ \ge z(1) + z(2) + ... + z(U) = BOUND0 \qquad (13)$$

The solving of integer quadratic problems (12) for all values of k (k=1,..,U-1, (since for k=U the solution is trivial and z(u)=0) would give us the optimum sequence if the vectors X(k) defined a sequence, fulfilling the condition:

$$X(k+1) \ge X(k) \qquad (14)$$

which is equivalent, in this case, to:

$$X(k+1) - X(k) = I(i) \qquad (15)$$

where all the components of vector I(i) are null except one, i, equal to 1. In general, condition

(15) will not be satisfied for all values of k by the X(k) solutions of (12).

3. GRAPH ASSOCIATED TO THE PROBLEM

Let us build a connected graph without loops or directed cycles with U+1 levels. The vertices at level k (k=0,1,2,...,U) are defined by the vectors X(k) such that:

$$\sum_{i=1}^{P} x(i,k) = k$$

$$0 \le x(i,k) \le u(i) \qquad i=1,2,\ldots,P \qquad (16)$$

$$x(i,k) \text{ integer}$$

At level 0 there will be a single vertex corresponding to X(0) = (0,0,..,0)'. This is also the case at level U, in which the vertex will correspond to X(U)=(u(1).....u(P))'. The number of vertices in the remaining levels NV(k) will be variable, reaching its maximum in the environment of (U+1)/2; there is an obvious symmetry, the number of vertices at level k being the same as at level U-k. In any case this number of vertices may be very high (see the recursive equations for the calculation of the number of vertices at the different levels in ANNEX I). We will give the name X(k,t) to one of the vertices of level k.

Between vertex X(k,t1) of level k and vertex X(k+1,t2) of level k+1 there will be an arc if the X vectors satisfy the condition of sequence (14) (or the equivalent (15)):

$$X(k+1,t2) \ge X(k,t1) \qquad (17)$$

To each vertex we associate a value:

$$A(k,t) = (Y(k,t) - R(k))'*(Y(k,t) - R(k)) = \| (Y(k,t) - R(k)) \| \qquad (18)$$

with

$$Y(k,t) = N*X(k,t) \qquad (19)$$

The definitions of level 0 and level U mean that:

$$A(0)=A(U)=0 \qquad (20)$$

Given the structure of the graph and of expression SDQ there would be no difference in associating this value to all the arcs directed toward to the vertex instead of to the vertex itself. Finding the sequence which minimizes SDQ is equivalent to finding the minimum path from X(0) to X(U) in the defined graph.

Therefore, any algorithm for determining extreme paths in a graph (e.g. that of

BELLMAN-KALABA) would lead us to the solution of the problem. An important obstacle is caused by the number of vertices of the graph, which as we have said may be very large according to U and P (and the u(i)); in most industrial cases this dimension makes it impossible to consider any of these algorithms in its pure form as an efficient procedure, which forces us to bear in mind all vertices of the graph explicitly.

4. HEURISTICS.

MONDEN (1983) proposes a very simple "greedy" heuristic, which he calls "goal chasing", used by TOYOTA. In our previous formulation this consists in progressively constructing a path in the graph, adding at each step an extension arc to the already built segment of path. This arc is chosen among the possible arcs (maximum P) according to their contribution to the value of the path. For example, at a given moment of application of the algorithm the segment of path constructed joins X(0) with X(k,t). The possible continuations are the arcs which emerge from X(k,t) and which join this vertex to X(k+1,t1), X(k+1,t2), ..., X(k+1,tP). (There will exist P possible following ones if in X(k,t) no variant has been exhausted, i.e. all the values x(i,k) are lower than u(i); otherwise the number of following ones will be lower: one per non-exhausted variant). The contribution of these arcs is A(k+1,t1), A(k+1,t2), ..., A(k+1,tP). The latter is chosen such that the contribution is lower:

$$A(k+1,th) = \min \{ A(k+1,t1),.....,A(k+1,tP) \} \quad (21)$$

and from (k+1,th) we proceed in the same way until X(U) is reached. Initially the constructed segment is void and the starting point is X(0). It can easily be seen that the number of values A(k,t) to be calculated in this procedure is limited (maximum P*(U - (P+1)/2)).

This heuristic is very fast, especially if recursive procedures of calculation of the A(k+1,th) are used starting from A(k,t). However, it has an undesirable behaviour in some cases in which it rapidly exhausts "comfortable" units by building a very good initial segment, but leaves the less appetizing units for sequencing at the end (the remainder). The final segment of the path is therefore very bad, which means that taken as a whole the path constructed is far away from the optimum one.

4.1. Recursive calculation of A(k,t)

Let us consider the calculation of A(k+1,ti) according to expression (18), assuming that X(k+1,ti) is that which follows X(k,t), by adding one more unit of variant i:

$$\begin{aligned} A(k+1,ti) &= \| (Y(k+1,ti) - R(k+1)) \| = \\ &= \| (Y(k,t) + N(i) - R(k) - R) \| = \\ &= A(k,t) + 2*(Y(k,t) - R(k))'*(N(i) - R) + \| (N(i) - R) \| \end{aligned} \quad (22)$$

where N(i) is the column corresponding to variant i in N. We observe that:

A(k,t) is the contribution in vertex X(k,t) that has already been calculated, and is independent of variant i chosen hereafter,

Y(k,t) - R(k) is the vector that measures the separation between the real consumption of components and the ideal or regular consumption in the vertex X(k,t), whose norm is A(k,t), and which is available in this vertex,

N(i) - R is the difference between the real consumption of components of variant i and the average consumption, and depends only on variant i,

$\| (N(i) - R) \|$ is the norm of the previous vector, and depends identically only on i,

Therefore, expression (22) represents a convenient recursive form for calculating A(k+1,ti) from A(k,t). Also, considering that the first term appears independently of the variant i chosen, we can replace expression (21) in order to determine the following arc of the path by:

$$\min \{ 2*(Y(k,t) - R(k))'*(N(i) - R) + \| (N(i) - R) \| \} \quad (23)$$

Since $\| (N(i) - R(i)) \|$ is inevitably incorporated into the contributions if there exists a variant i to be situated in the sequence, a reasonable heuristic consists in choosing the i that minimizes only the first term:

$$\min_i \{ (Y(k,t) - R(k))'*(N(i) - R) \} \quad (24)$$

unless A(k,t) is very small, which means that all components of (Y(k,t) - R(k)) are also small, and thus (24) does not discriminate.

We will use the name "revised" MONDEN heuristic for the new procedure defined above.

4.2. Improvements to the MONDEN heuristic

We have experimented with three types of modifications to improve the sequence obtained through the "greedy" procedure:

1) symmetry: using the symmetry of the graph and building the path by the concatenation of two segments, one starting at X(0) and one at X(U). The addition of arcs in the first segment is performed in accordance with what we have seen in 4, in the second segment, "backward" construction, the new arc is the first of the segment, that of minimum input being chosen. Alternatively arcs are added to Segments 1 and 2, bearing in mind the remaining units that have not been sequenced either in Segment 1 or 2 to enumerate the exhaustions of variants and therefore the possible arcs.

2) horizon: the "short-sightedness" of the MONDEN heuristic lies in the fact that it only takes into account short-term consequences: one step. If we build a segment starting on the left (X(0)), but on adding an arc we bear in mind two steps (i.e. the contributions of k+1 and k+2), the myopia is reduced. To this end, at each level except U-1 we consider all the possible prolongations of two concatenated arcs and we choose the one whose potential contributions A(k+1,th)+A(k+2,tl) is lowest. Only the first arc, the one that links A(k,t) with A(k+1,th), is added to the segment, and with the new segment we proceed likewise. We obtain better solutions, sometimes drastically better than with one step, by compensating for the increase in volume of the calculations that is acceptable. It is possible to generalize the procedure to three steps or more, though the improvement in the procedure is not appreciable compared with the increase in calculations.

3) rate preserving: since the main cause of the undesirable behavior of the MONDEN heuristic occurs when the rest of the units to be sequenced has undergone a impoverishment and its composition differs considerably from the initial one, one way consists in having a mechanism which eliminates from the possible prolongations of a segment those which will lead to compositions of the rest which are too distant from the initial composition.

Obviously, several of the improvements can be included at the same time.

We have also experimented with procedures of improvement based on a given sequence, permuting among themselves the positions of two units of different variants and analyzing the new value of the SDQ obtained. Within these procedures we may include the experimentation with simulated annealing and tabu search. The time needed to obtain good solutions with these procedures makes them unfeasible from a practical point of view.

4.3. Numerical example.

We will illustrate the above by a simple example with P=5, C=4, U=48, and the following values of n(j,i) y u(i):

	variants (i)				
components	1	2	3	4	5
1	3	2	1	0	3
2	2	2	4	2	3
3	0	3	2	5	0
4	4	2	2	2	3
quantity u(i)	12	8	6	15	7

The rates r(j) are: r(1)=1.646; r(2)=2.396; r(3)=2.312; r(4)=2.646

The choice of example is justified by the fact that in it the heuristics show their defects clearly. By applying five simple heuristics, we obtain different sequences with the following values of SDQ:

heuristic	SDQ
MONDEN	226.541
MONDEN revised	212.042
symmetrical	198.042
2-step	153.042
rate-preserving	176.208
revised/symmetrical	165.542
symmetrical/rate-pres.	161.042

For the rate-preserving heuristics we have adopted the criterion of considering the values of A(k,t) that differ by less than 5 % as tied, and deciding in favor of the arc that represents the incorporation to the sequence of the product whose subtraction from the remainder makes its composition even less distant from the initial one.

It is curious to observe that by using the revised procedure we improve the value of SDQ by 6 %, though part of the improvement may be attributed to the greater number of ties and the means of resolving them. The use of a control of the impoverishment of the remainder to resolve ties gives substantial improvements. The best procedure for this case is, however, the 2-step, though as we will see, the best solution found is still 30 % above the optimum value.

Extensive experimental work is underway on the behaviour of the heuristics. The first results can be found in BAUTISTA (1993, see a sample in ANNEX III).

5. BOUNDING THE VALUE OF THE SEQUENCES

We will first study the bounding of SQD, and then the bounding of the rest of the remaining path (complement) when the first segment has been constructed.

5.1. Global bounds of SQD

Let us consider the minimum possible contribution of each variant i to the values A(k,t) and A(k+1,ti) of two vertices united by an arc. For each component j this input will be composed (see Figure 1) of the square of the difference in A(k-1,t), which we call Θ, plus the square of the difference in A(k,ti) which will be $(\Theta + n(j,i) - r(j))$. The minimum of this sum of squares is easy to determine:

$$\min_{\Theta} \{ \Theta^2 + (\Theta + n(j,i) - r(j))^2 \} = \tfrac{1}{2}*(n(j,i) - r(j))^2 \quad (25)$$

which is obtained when $\Theta = \frac{1}{2}*(n(j,i) - r(j))$. As in any path the number of variants i that appear is u(i) and also A(0)=A(U)=0, which also means that the first and last unit cannot be situated in the optimum form indicated above

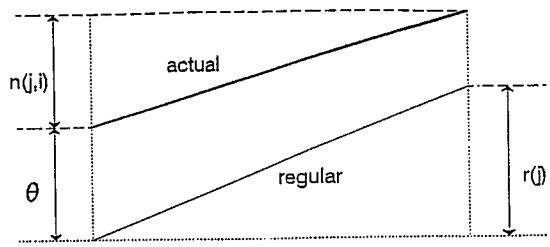

Fig. 1

$$2*\sum_{k=0}^{U} A(k,tk) \geq \frac{1}{2}* \sum_{j=1}^{C} \sum_{i=1}^{P} u(i)*(n(j,i) - r(j))^2 + 2*\text{supplement} \qquad (26)$$

Therefore, the first evaluation of the lower bound, leaving out the supplement, is:

$$\text{BOUND1} = \frac{1}{4} * \sum_{j=1}^{C} \sum_{i=1}^{P} u(i)*(n(j,i) - r(j))^2 = \frac{1}{4} * U * \sum_{j=1}^{C} \sigma^2(j) \qquad (27)$$

where $\sigma(j)$ is the standard deviation from the distribution of components, just as r(j) is the average.

We can establish the lower bound of the supplement, using the definition of its "raison d'etre", though in general the correction will not be very significant:

$$\text{BOUND2} = \frac{1}{4} * U * \sum_{j=1}^{C} \sigma^2(j) + \sum_{j=1}^{C} \text{MIN}_i \{ (n(j,i) - r(j))^2 \} \qquad (28)$$

With the data in the example we have:

BOUND1 = 90.432

BOUND2 = 90.937

We can make two important observations to improve the bound:

1) we have considered n(j,i) to be integers,

2) the values r(j) are thus rational.

Therefore the possible Θ values are defined in finite number and are also a function of k (or of k*r(j) if we prefer)

Let us give the name q(k,i) to the minimum contribution of the variant i situated at position k of the sequence, or likewise in the arc that links level k-1 with k (contribution to A(k-1,t) and to A(k,ti)).

$$A(k,t) + A(k+1,ti) \geq q(k,i) \qquad (29)$$

The algorithm indicated in ANNEX II allows us to calculate the values of q(k,i) for any k or i. By means of:

$$qmin(i) = \min_{k} \{ q(k,i) \} \qquad (30)$$

we can obtain:

$$BOUND3 = \tfrac{1}{2} * \sum_{i=1}^{P} u(i)*qmin(i) \qquad (31)$$

though it would be more efficient to order the U values of q(k,i) for fixed i in increasing order and, giving the name qo(l,i) to the value that occupies position l in this order, to bound with:

$$BOUND4 = \tfrac{1}{2} * \sum_{i=1}^{P} \sum_{l=1}^{u(i)} qo(l,i) \qquad (32)$$

A more refined procedure would consist in reducing the original problem to one of transport (33), though the greater volume of calculations needed to resolve it may make the procedure inadvisable:

$$BOUND5 = \tfrac{1}{2} * [MIN] \sum_{k=1}^{U} \sum_{i=1}^{P} s(k,i)*q(k,i)$$

s.t.

$$\sum_{i=1}^{P} s(k,i) = 1 \qquad k=1,2,\ldots,U$$

$$\sum_{k=1}^{U} s(k,i) = u(i) \qquad i=1,2,\ldots,P \qquad (33)$$

$$s(k,i) \in [\,0\,,\,1\,] \quad k=1,2,..,U\ ;\ i=1,2,\ldots,P$$

In our example the values obtained are:

BOUND3 = 92.438

BOUND4 = 94.969

BOUND5 = 99.000

5.2. Bound of the complement of a given segment

Let us suppose that we have built a path from X(0) to X(k,t) and we have obtained its value f(k,t) by adding the contributions of the vertices through which it passes. Giving the name x(i) to the component i of X(k,t) (making a simplification of the nomenclature) the number of units of the variant i that remain to be placed in the sequence is $u(i) - x(i) = s(i) \geq 0$. To complete the path we need U-k arcs, and the associated values will give us a complement which, when added to f(k,t), will give us SDQ:

$$COMP = A(k+1,t1) + A(k+2,t2) + \ldots\ldots + A(U) \qquad (34)$$

where A(U), as we have said, has a value of 0. Giving the name [l] to the variant that we will place in position l, according to (29), we can write:

$$A(k,t) + 2 * COMP \geq \sum_{l=k+1}^{U} q(l,[l]) \qquad (35)$$

which allows us to give lower bounds to the value COMP, since A(k,t) is known and we can apply to the sum of the second member any of the procedures that we have used to define BOUND3, BOUND4 or BOUND5. We will call these adaptations BOUND3(k,t), BOUND4(k,t) and BOUND5(k,t):

$$qmin(i,k) = \min_{l > k} \{ q(l,i) \} \qquad (36)$$

$$BOUND3(k,t) = \tfrac{1}{2} * \sum_{i=1}^{P} s(i) * qmin(i,k) - \tfrac{1}{2} * A(k,t)$$

If we order the U-k values q(l,i) for l>k for each i in increasing order and call the value that occupies position 1 in this order qo(l,i):

$$BOUND4(k,t) = \tfrac{1}{2} * \sum_{i=1}^{P} \sum_{l=1}^{s(i)} qo(l,i) - \tfrac{1}{2} * A(k,t) \qquad (37)$$

$$BOUND5(k,t) + \tfrac{1}{2} * A(k,t) = \tfrac{1}{2} * [MIN] \sum_{l=k+1}^{U} \sum_{i=1}^{P} s(l,i)*q(l,i)$$

s.t.

$$\sum_{i=1}^{P} s(l,i) = 1 \qquad l=k+1,k+2,\ldots,U$$

$$\sum_{l=k+1}^{U} s(l,i) = s(i) \qquad i=1,2,\ldots,P \qquad (38)$$

$$s(l,i) \in [\,0\,,\,1\,] \quad l=k+1,k+2,..,U \;;\; i=1,2,\ldots,P$$

In all cases the subtrahend ½*A(k,t) can be replaced by:

$$\tfrac{1}{2} * \min \{ A(k,t) , \min_{i} \{ q(k+1,i) \} \} \qquad (39)$$

since the purpose of this subtrahend is to correct the first contribution, and it would be illogical for it to exceed it.

In the search for a balance between the volume of work and the quality of the bound, we have generally opted for BOUND4.

6. THE USE OF BOUNDED DYNAMIC PROGRAMMING

The general principles of Bounded Dynamic Programming (BDP) have been described in BAUTISTA, COMPANYS, COROMINAS (1991), which contains more details. Previous work

on this had been done by MORIN & MARSTEN (1976) and MARSTEN & MORIN (1978), extended in CARRAWAY & SCHMIDT (1991).

Our approach consists essentially in moving a window of width H on the graph (only a maximum of H vertices will be taken into consideration at each level) in the calculation of the path with the minimum value. These H retained vertices will be the best or most promising of those liable to lead to a minimum value.

Initially, we have a value z0 corresponding to SDQ of a sequence obtained through a heuristic procedure. Each vertex retained X(k,t) is associated to the value f(k,t) of the best path found between X(0) and X(k,t) and to a lower bound of the completion of the path (obtained according to 5) BOUND4(k,t). A vertex may be eliminated from those considered if:

$$f(k,t) + BOUND4(k,t) \geq z0 \qquad (40)$$

Since in these conditions it cannot give us a sequence with a lower value than the sequence already found by the heuristic procedure.

The procedure consists in tentatively generating one by one the vertices following each vertex in the list of vertices at level k-1, L(k-1). Let X(k-1,t) be a vertex of L(k-1):

1) A following X(k,th) is generated. If all the following ones have been generated we go on to the other vertex of L(k-1). If all the vertices of L(k-1) have been generated we go on to level k.

2) If $f(k,th)+BOUND4(k,th) \geq z0$ the vertex is eliminated and we return to (1)

3) We search in L(k) to see if X(k,th) has already been generated. If so, we analyze the value of f(k,th) registered in L(k):

3.1) If this value is less than or equal to that corresponding to the new path, we return to (1),

3.2) If this value is greater than that corresponding to the new path, it is replaced by the f(k,th) recently found and X(k-1,t) is registered as the one previous to the vertex X(k,th). We return to step (1).

4) If X(k,th) has not been generated, we analyze the number of vertices already existing in the list, H(k):

4.1) If $H(k) < H$, we add to the list the entry (X(k,th),f(k,t),BOUND4(k,t),X(k-1,t)) and we perform $H(k)=H(k)+1$. We return to (1),

4.2) If $H(k)=H$ we determine the worst vertex of the list, X(k,tl), to which the greater value of the sum:

$$f(k,tl) + BOUND4(k,tl)$$

corresponds.

4.2.1) If

$$f(k,tl) + BOUND4(k,tl) \leq f(k,th) + BOUND4(k,th)$$

we leave out the vertex X(k,th), although the optimum path could potentially pass through it.

4.2.2) If

$$f(k,tl) + BOUND4(k,tl) > f(k,th) + BOUND4(k,th)$$

we replace in the list the entry of X(k,tl) with that of X(k,th). Now the vertex which is left out is X(k,tl), though the optimum path could potentially pass through it.

In both cases we return to (1)

The procedure is started by writing in the list the L(0) the entry (X(0),0,BOUND4,-) and performing k-1=0. When we have obtained the list L(U), if it is empty it means that we have not been able to find a sequence with a value lower than z0. Otherwise, the entry will be (X(U),f(U),0,X(U-1,t)) with $f(U) < z0$, and by reconstructing the path we will obtain a better sequence than the initial one. If the final values of H(k) are such that:

$$\max \{ H(1), H(2), \ldots , H(U-1) \} < H \qquad (41)$$

the value of this sequence will be the optimum one. The criterion may be tuned since it is possible that the first member is equal to H without having left out any potentially interesting vertex due to the width of the window. However, rather than saving information on whether or not any vertices have been left out, it is better to conserve the best value f(k,t) + BOUND4(k,t) of the vertices that have been left out. If this value is greater than (or equal to) the value f(U), none of these vertices could have led us to a sequence with a better value than that found, and this one is therefore the best. Otherwise we can only consider the solution as being close to the optimum one, and the procedure as just an heuristic.

The number of solutions retained at any level depends on the quality of z0 and on the bounding procedure. As z0 decreases, H(k) is non-increasing. Therefore, if the procedure ends without guaranteeing the optimum one, but giving us a better solution than the one initially available, it will be useful to reiterate it with the value of the new solution as z0.

In the case of the numerical example mentioned above, with z0=153.042 y H=150 we have obtained as the value of best solution 120.8745, followed by others (at level 47) with the values 122.8745, 123.2078, 124.0412 and 124.2079. Since max { H(k) } = 150, we cannot conclude directly that we have determined the optimum one until we check that all the vertices left out

fulfill the relation:

$$f(k,t) + BOUND4(k,t) \geq 126.8828$$

We can therefore guarantee that the optimum sequences have a value of 120.8745

By reiterating the procedure with z0 = 125 y H = 150 we obtain the same five solutions with max { H(k) } = H(24) = 142 < 150, which confirms that 120.8745 corresponds to the minimum.

With z0=153.042 y H=5 (equal to P) we would also have found the best five solutions mentioned above, but in this case the evaluation of the best vertex left out is of the order of 91.7812, which would have not allowed us to guarantee the optimality of the best solution found. However, the speed of the procedure makes it an very efficient heuristic.

7. CONCLUSIONS

We have presented different heuristics and an exact method for solving a MONDEN problem. We have used a quadratic criterion, but most of the procedures are adaptable to other types of criteria, e.g. the square root of the sum of squares of the differences, SDE of expression (19) or the sum of absolute values of the differences, SDR of expression (11). In particular, the BDP only specifies the calculation adapted to the criterion of bounds q(k,i).

In this explanation we have not used the properties inherent to the symmetry of the graph. In particular, at the level INT((U+1)/2) we have associated to each vertex another, at the same level or the previous level, with the units of each product that completes the total of units available u(i). Therefore, by connecting both vertices we can now evaluate the complete paths from X(0) to X(U).

8. BIBLIOGRAPHY

BAUTISTA, J., COMPANYS, R. and COROMINAS, A. (1991), "Introducción al BDP", Document Intern de treball, DOE, ETSEIB-UPC

BAUTISTA, J. (1993), "Procedimientos heurísticos y exactos para la secuenciación en sistemas productivos de unidades homogéneas (contexto J.I.T.)", Tesis Doctoral, DOE, ETSEIB-UPC

CARRAWAY, R. L. & SCHMIDT, R. L. (1991), "An improved discrete Dynamic Programming Algorithm for allocating resources among interdependent projects", Management Sci., vol. 37, n. 9, pp. 1195-1200

COMPANYS, R. (1989), "Secuenciación de productos en el monteje para lograr la regularidad en el consumo de recursos", CIM, n. 10,

KUBIAK, W. and SETHI, S. (1991), "A note on 'Level Schedules for Mixed-Model Assembly Lines in Just-in-Time Production Systemas'", Management Sci., vol. 37, n. 1

MARSTEN, R. E. & MORIN, Th. L. (1978), "A hybrid approach to discrete Mathematical programming", Mathematical Programming, vol. 14, pp. 21-40

MORIN, Th. L. & MARSTEN, R. E. (1976), "Branch-and-bound strategies for Dynamic Programming", Operations Research, vol. 24, n. 4, pp. 611-627

MILTEMBURG, J. (1989), "Level Schedules for Mixed-Model Assembly Lines in Just-in-Time Production Systems", Management Sci., vol. 35, n, 2

MONDEN, Y. (1983), Toyota Production System, Institute of Industrial Engineering Press

SUMICHAST, R. T. and RUSSELL, R. S. (1990), "Evaluating Mixed- model Assembly Line Sequencing Heuristics for Just-in-Time Production Systems", Journal of Operations Management, vol. 9, n. 3

YANO, C. A. and RACHAMADUGU, R. (1991), "Sequencing to minimize Work Overload in Assembly Lines with Product Options", Management Sci., vol. 37, n. 5

ANNEX I - NUMBER OF VERTICES AT LEVEL k

We can perform the calculation iteratively.

```
PHASE 1 - g(0)=1
        FOR k=1 TO U : g(k)=0 : NEXT k
        k1=0 : i=1
PHASE 2 - WHILE i ≤ P
                k1=k1+u(i)
            k=k1
PHASE 3 -     WHILE k > 0
                  SUM=0 : k2=0
PHASE 4 -       WHILE k2 ≤ u(i)
                k3=k - k2
                IF k3 < 0 THEN x=0 ELSE x=g(k3)
                SUM=SUM+x
                k2=k2+1
              WEND
PHASE 5 -       g(k)=SUM
            k=k-1
          WEND
PHASE 6 -     i=i+1
        WEND
PHASE 7 - END : 'g(k) indicates the number of vertices at level k.
```

The application to the example described gives at level 24, g(24) = 5574, which is the maximum of vertices at a level. The total number of vertices is 104832, which we could have obtained more easily through the product:

$$\text{total number of vertices} = \prod_{i=1}^{P} (u(i) + 1)$$

With a window of width H=100, the total number of vertices retained is 4152 (less than 4 % of the total) and with H=150, 6054 (less than 6 % of the total).

ANNEX II - ALGORITHM TO DETERMINE q(k,i)

PHASE 1 - For each component $j=1,2,..,C$ determine d(j):

1.1. Determine $\alpha = k*r(j) - INT(k*r(j))$
1.2. Calculate $\beta = \alpha + r(j) - n(j,i)$
1.3. WHILE $\alpha + \beta > 1$ perform $\alpha=\alpha-1$ and $\beta=\beta-1$
1.4. WHILE $\alpha + \beta < -1$ perform $\alpha=\alpha+1$ and $\beta=\beta+1$
1.5. $d(j) = \alpha^2 + \beta^2$

PHASE 2 - $q(k,i) = \sum_{j=1}^{C} d(j)$

ANNEX III - COMPUTATIONAL EXPERIENCE

We experimented with five structures (see Tables A.1 to A.5) and 45 productions programmes -vector of values of u(i)-, 225 problems in all.

In Table A.6 we present the results (value of SDQ) obtained applying four heuristics and the BDP method in the convenient form to drive at the optimum to structure 3. The heuristics are:

A-2 : Monden revised
A-2e: Monden revised/ rate-preserving
A-3 : 2-step
A-3b: 2-step/ rate-preserving

Tables A.7 to A.9 present a summary of results (average, maximun and minimun of % difference from optimum) for the five structures and the four heuristics.

$$\% \text{ difference} = \frac{\text{Heuristic} - \text{BDP}}{\text{BDP}} * 100$$

Table A.1

S-01	C1	C2	C3	C4
A1	1	4	1	1
A2	5	0	1	1
A3	2	3	0	2
A4	0	5	2	0

Structure-1

Table A.2

S-02	C1	C2	C3	C4	C5	C6	C7	C8
A1	2	4	1	1	1	1	4	1
A2	6	0	1	1	1	1	0	5
A3	4	2	0	2	0	2	3	2
A4	0	6	2	0	2	0	5	0

Structure-2

Table A.3

S-03	C1	C2	C3	C4
A1	0	3	3	4
A2	0	1	3	6
A3	0	0	5	5
A4	5	5	0	0

Structure-3

Table A.4

S-04	C1	C2	C3	C4
A1	1	1	0	1
A2	0	2	0	1
A3	0	0	2	1
A4	1	1	1	0

Structure-4

Table A.5

S-05	C1	C2	C3	C4	C5
A1	3	0	1	3	1
A2	6	1	2	6	0
A3	9	2	3	9	0
A4	12	3	4	12	0

Structure-5

Table A.6

#	Programme u(i)	HEURISTIC A-2	A-2e	A-3	A-3b	Optim. BDP
1	17,1,1,1	204.10	95.10	93.70	93.70	72.70
2	1,17,1,1	135.30	135.30	134.30	134.30	133.70
3	1,1,17,1	163.30	163.30	163.30	163.30	150.70
4	1,1,1,17	165.10	165.10	165.10	165.10	155.30
5	9,9,1,1	186.70	92.10	117.50	117.50	90.30
6	9,1,9,1	225.30	108.70	151.10	151.10	99.10
7	9,1,1,9	158.70	149.10	114.70	114.70	114.70
8	1,9,9,1	146.30	146.30	141.30	141.30	141.30
9	1,9,1,9	172.90	167.90	166.90	166.90	149.70
10	1,1,9,9	202.70	202.70	177.10	177.10	177.10
11	6,6,4,4	292.80	156.00	157.60	157.60	130.40
12	6,4,6,4	332.20	161.00	160.60	160.60	129.40
13	6,4,4,6	231.00	163.00	159.80	159.80	148.60
14	4,6,6,4	301.40	169.40	157.40	157.40	157.40
15	4,6,4,6	247.40	177.40	155.80	155.80	146.20
16	4,4,6,6	310.00	184.80	165.20	165.20	155.60
17	5,5,5,5	405.50	142.50	137.50	137.50	137.50
18	6,6,6,2	192.00	134.80	155.20	155.20	122.80
19	6,6,2,6	221.20	153.20	146.00	146.00	135.60
20	6,2,6,6	245.60	164.40	158.00	158.00	147.60
21	2,6,6,6	237.60	169.20	162.00	162.00	149.20
22	2,4,6,8	212.20	174.20	156.20	156.20	153.40
23	2,4,8,6	232.60	166.60	158.60	158.60	147.00
24	2,6,4,8	222.80	182.40	158.40	158.40	144.80
25	2,6,8,4	209.60	167.20	178.00	178.00	144.80
26	2,8,4,6	224.20	158.60	152.60	152.60	139.80
27	2,8,6,4	215.40	174.60	176.20	176.20	148.60
28	4,2,6,8	178.20	177.00	165.00	165.00	162.20
29	4,2,8,6	321.00	185.40	161.40	161.40	158.20
30	4,6,2,8	169.00	159.40	159.00	159.00	157.40
31	4,6,8,2	185.00	159.80	183.80	183.80	129.40
32	4,8,2,6	234.80	169.60	157.20	157.20	146.80
33	4,8,6,2	260.00	152.80	180.80	180.80	130.00
34	6,2,4,8	176.80	150.40	125.60	125.60	125.60
35	6,2,8,4	190.80	140.80	164.00	156.40	125.60
36	6,4,2,8	168.20	135.80	129.40	129.40	129.40
37	6,4,8,2	199.00	132.60	158.20	158.20	118.20
38	6,8,2,4	291.80	147.00	150.60	150.60	128.60
39	6,8,4,2	187.00	129.00	155.00	155.00	117.40
40	8,2,4,6	168.20	152.60	141.40	141.40	141.40
41	8,2,6,4	282.60	142.20	142.20	142.20	116.60
42	8,4,2,6	146.00	146.40	138.80	138.80	137.20
43	8,4,6,2	162.80	120.80	122.00	122.00	118.80
44	8,6,2,4	346.60	111.80	130.60	130.60	111.80
45	8,6,4,2	191.00	112.60	157.40	139.40	112.60

Results with Structure.3

Table A.7

Average % difference	HEURISTIC A-2	A-2e	A-3	A-3b	Number of Items
S-01	57.65	19.59	9.89	9.07	45
S-02	55.44	20.07	14.36	14.73	45
S-03	67.21	12.58	13.59	13.10	45
S-04	11.62	6.58	1.72	1.21	45
S-05	37.69	23.80	25.31	25.31	45
Global	45.92	16.52	12.97	12.68	225

Table A.8

Maximun % difference	HEURISTIC A-2	A-2e	A-3	A-3b	Number of Items
S-01	129.27	57.32	37.29	37.29	45
S-02	153.95	77.82	70.85	70.85	45
S-03	210.02	30.81	52.47	52.47	45
S-04	48.78	26.67	26.67	26.67	45
S-05	88.89	77.04	57.77	57.77	45
Global	210.02	77.82	70.85	70.85	225

Table A.9

Minimum % difference	HEURISTIC A-2	A-2e	A-3	A-3b	Number of Items
S-01	9.38	1.43	0.00	0.00	45
S-02	14.63	0.00	0.00	0.00	45
S-03	1.20	0.00	0.00	0.00	45
S-04	0.00	0.00	0.00	0.00	45
S-05	2.46	0.12	0.00	0.00	45
Global	0.00	0.00	0.00	0.00	225

example, at a given moment of application of the algorithm the segment of path constructed joins X(0) to X(k,t). The possible continuations are the arcs which emerge from X(k,t) and which join this vertex to $X(k+1,t_1)$, $X(k+1,t_2)$, ..., $X(k+1,t_P)$. (There will exist P possible following ones if in X(k,t) no product has been exhausted, i.e. all values $x_{i,k}$ are lower than u_i; otherwise the number of following ones will be lower: one per non-exhausted product). The contribution of these arcs is $A(k+1,t_1)$, $A(k+1,t_2)$, ..., $A(k+1,t_P)$. The latter is chosen such that the contribution is lower:

$$A(k+1,t_i) = \min \{ A(k+1,t_1),.....,A(k+1,t_P) \} \quad (17)$$

and from $X(k+1,t_i)$ we proceed in the same way until X(U) is reached. Initially the constructed segment is void and the starting point is X(0). It can easily be seen that the number of values A(k,t) to be calculated in this procedure is limited (at the most $P\cdot(U - (P+1)/2)$).

This heuristic is very fast, especially if recursive procedures are used to calculate $A(k+1,t_i)$ starting from A(k,t). However, it displays undesirable behaviour in some cases, rapidly exhausting "comfortable" units by building a very good initial segment but leaving the less appetizing units for sequencing at the end (the remainder). The final segment of the path is therefore very bad, which means that taken as a whole the path constructed is far removed from the optimum one.

4.1. Recursive calculation of A(k,t)

Expression (14) can be written in a more compact form. $\mathbf{R_k}$ is:

$$\mathbf{R_k} = k*\mathbf{R} = \mathbf{R}*(\mathbf{O}'*\mathbf{X_{k,t}}) \quad (18)$$

where $\mathbf{O} = [\ 1\ 1\ 1\ \ldots\ 1\ ]'$ is a (P x 1) vector, and $\mathbf{O}'*\mathbf{X_{k,t}}=k$. Then:

$$A(k,t) = \| (\mathbf{N} - \mathbf{R}*\mathbf{O}')*\mathbf{X_{k,t}} \| = \mathbf{X_{k,t}}'*\mathbf{A}*\mathbf{X_{k,t}} \quad (19)$$

where:

$$\mathbf{A} = (\mathbf{N} - \mathbf{R}*\mathbf{O}')'*(\mathbf{N} - \mathbf{R}*\mathbf{O}') \quad (20)$$

Let us consider the calculation of $A(k+1,t_i)$ according to expression (19), assuming that $X(k+1,t_i)$ is that which follows X(k,t), by adding one more unit of product i:

$$A(k+1,t_i) = \mathbf{X_{k+1,ti}}'*\mathbf{A}*\mathbf{X_{k+1,ti}} = (\mathbf{X_{k,t}} + \mathbf{I_i})'*\mathbf{A}*(\mathbf{X_{k,t}} + \mathbf{I_i}) = A(k,t) + 2\cdot\mathbf{A_i}'*\mathbf{X_{k,t}} + a_{i,i} \quad (21)$$