Journal of Industrial Engineering and Management

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

Optimal priority ordering in PHP production of multiple part-types in a failure-prone machine

Ana Sánchez, Albert Corominas, Rafael Pastor

Universitat Politècnica de Catalunya (SPAIN)

ana.sanchez@upc.edu; albert.corominas@upc.edu; rafael.pastor@upc.edu

Received March 2009
Accepted September 2009

Abstract: This note deals with the problem of minimising the expected sum of quadratic holding and shortage inventory costs when a single, failure-prone machine produces multiple part-types. Shu and Perkins (2001) introduce the problem and, by restricting the set of control policies to the class of prioritised hedging point (PHP) policies, establish simple, analytical expressions for the optimal hedging points provided that the priority ordering of the part-types is given. However, the determination of an optimal priority ordering is left by the authors as an open question. This leaves an embedded sequencing problem which we focus on in this note. We define a lower bound for the problem, introduce a test bed for future developments, and propose three dynamic programming approaches (with or without the lower bound) for determining the optimal priority orderings for the instances of the test bed. This is an initial step in a research project aimed at solving the optimal priority ordering problem, which will allow evaluating the performance of future heuristic and metaheuristic procedures.

Keywords: scheduling, cumulative resources, failure-prone machines, prioritised hedging point control, production control.

Journal of Industrial Engineering and Management
doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

1 Introduction and problem statement

Shu and Perkins (2001) introduce the problem of optimising the costs of a production system that consists in a single, failure-prone machine that is able to produce up to n part-types simultaneously with constant demand rates $\left(d_j; j=1,...,n\right)$. The machine alternately adopts an "up" state, in which it is fully functional, and a "down" state, in which it is not able to produce anything. The time that the machine spends in each state before switching to the other is distributed exponentially with parameters equal to $1/q_d$ and $1/q_u$ for the "down" and "up" states respectively. The vector of the production rates to be determined for each part-type in each instant of time t is $x(t)=\left[x_1(t), x_2(t),. . x_n(t)\right]$. Therefore, the first derivative, $s_j'(t)$, of the inventory level of part-type j at time t, $s_j(t)$, satisfies the equation $s_j'(t)=x_j(t)-d_j$. Backlog is unavoidable and therefore admitted. Without loss of generality, it is assumed that the maximum production rate for any part-type is equal to μ, which is the capacity of the machine; therefore, the production rates, $\forall t \geq 0$, must fulfil the condition $\sum_{j=1}^{n} x_j(t) \leq \mu$. It is also assumed that $\frac{q_u}{q_d+q_u}\mu-\sum_{j=1}^{n} d_j>0$, i.e., that the machine has enough capacity to meet demand. The instantaneous cost function of the system is assumed to be equal to $\sum_{j=1}^{n} c_j \cdot s_j^2(t)$, where c_j $\left(j=1, 2,..., n\right)$ are nonnegative constant costs; as it is pointed out in Shu and Perkins (2001), the quadratic instantaneous cost function is a useful cost approximation for systems in with productions are perishable or may become obsolete, as well as systems with storage-space competition. The objective, then, is to minimise the expected long-term average cost: $J=\lim_{T\to\infty}\frac{1}{T}E\left[\int_0^T\left(\sum_{j=1}^{n} c_j s_j^2(t)\right)dt\right]$.

In real industrial systems, there are several kinds of failure-prone resources that can share their capacity among various activities. A specific example known by the authors is the case of a painting plant in which a pump (which has a limited maximum flow, μ, and may break down) is able to simultaneously feed the flow

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

necessities, d_j, of n canning stations $(j=1,...,n)$, each of which has a particular, controllable rate of flow, $x_j(t)$. This plant has different pumps, each of which feeds the flow necessities of $n=4$ canning stations simultaneously. Many other resources, such as ovens, autoclaves and treatment bath containers can also share their capacity. Actually, an entire manufacturing system can play the role of the "failure-prone machine" in this model and face the problem of how to share its capacity among multiple products (Ketzenberg et al., 2006, discuss this in relation to the problems of a pencil manufacturer that produces different pencil types for the highly seasonal back-to-school market).

In prioritised hedging point (PHP) policies a priority ordering, $\{p_1, p_2, ..., p_n\}$, is established for the part-types $(j=1,...,n)$ and the machine attempts to drive the inventory level of each part-type j to its hedging point, z_{p_j}, and to keep it at this level. When the machine is "up" (when it is "down" the machine cannot work at all) its production capacity is assigned as follows. If the difference $z_{p_1}-s_{p_1}(t)$ for part-type p_1, is positive, zero or negative (this can only happen in a transient state) the production rate $x_{p_1}(t)$ is respectively equal to μ, d_{p_1} or 0. The production rate of any other part-type p_j $(j>1)$ is equal to zero unless the inventory levels of all the part-types with greater priority are above or equal to their corresponding hedging points. When this condition is fulfilled, if the difference $z_{p_j}-s_{p_j}(t)$ is positive, zero or negative (this can also only happen in a transient state) the production rate $x_{p_j}(t)$ is respectively equal to μ minus the capacity assigned to the part-types with greater priority, d_{p_j} or 0. Figure 1 shows the evolution of the inventory level of three part-types (*A*, *B* and *C*), taking into account the priority ordering $\{A,B,C\}$ and without an initial inventory $(s_A(0)=s_B(0)=s_C(0)=0)$. For $0\le t<t_1$, when $s_A(t_1)=z_A$, $x_A(t)=\mu$ and $x_B(t)=x_C(t)=0$; for $t_1\le t<t_2$, when $s_B(t_2)=z_B$, $x_A(t)=d_A$, $x_B(t)=\mu-d_A$ and $x_C(t)=0$; for $t_2\le t<t_3$, when $s_C(t_3)=z_C$, $x_A(t)=d_A$, $x_B(t)=d_B$ and $x_C(t)=\mu-d_A-d_B$; for $t_3\le t<t_4$, $x_A(t)=d_A$,

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

$x_B(t) = d_B$ and $x_C(t) = d_C$. At instant t_4 the machine adopts a "down" state in which it is not able to produce anything. In instant t_5 the machine adopts an "up" state and begins producing part-type *A* again with a production rate equal to μ.

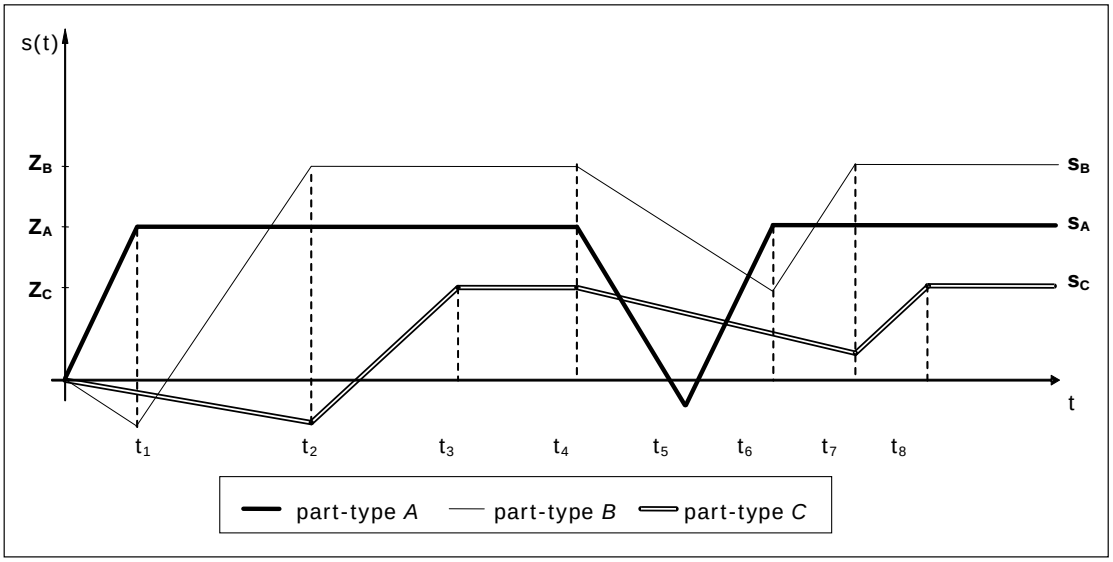

Figure 1. "Evolution of the inventory level of three part-types".

The optimal general solution of the problem is not known. However, Shu and Perkins (2001) establish simple closed-form expressions for the optimal hedging points and the corresponding cost, assuming that the prioritised hedging point policy is applied and a priority order of the part-types is given. For part-type 1

$$z_1^* = \frac{1-\gamma_1}{\lambda_1} \quad \text{and} \quad J_1^* = c_1 \cdot \frac{1-\gamma_1^2}{\lambda_1^2}$$

and for part-type $j > 1$

$$z_j^* = \frac{1-\gamma_j}{\lambda_j} - \frac{1-\gamma_{j-1}}{\lambda_{j-1}} \quad \text{and} \quad J_j^* = C_j - c_j \left(z_j^*\right)^2$$

where $D_j = \sum_{k=1}^{j} d_k$, $\lambda_j = \frac{q_u}{D_j} - \frac{q_d}{\mu - D_j}$, $\gamma_j = \frac{\mu q_u - (q_d + q_u) D_j}{(\mu - D_j)(q_d + q_u)}$,

$C_j = 2c_j z_j^* \left[\frac{1}{\lambda_j} - \frac{(1-\gamma_{j-1})\gamma_j}{(1-\gamma_j)\gamma_{j-1}\lambda_{j-1}} \right]$ and the total cost is $J^* = \sum_{j=1}^{n} J_j^*$.

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

Nevertheless, only initial results of the optimal priority ordering, summarised here in Section 3.1, are presented in Shu and Perkins (2001) and its authors point out that this is an area for future research.

This problem raises at least three questions: (i) What structure do optimal policies have? (ii) Given a priority ordering, how can the optimal values of the hedging points be calculated for hedging point policies? (iii) How can optimal orderings be found for prioritised policies?

As for many inventory management problems (Sethi & Thompson, 2000), it is very natural to approach the first two questions with the help of control theory. However, the third question corresponds to a family of scheduling problems in which the elements are characterised by the distribution of the times the machine spends in the “up” and “down” states and the cost function. The first objective of this note is to place the problem in the field of scheduling and relate it to other similar scheduling problems.

The distinction between disjunctive scheduling (a resource can execute one activity at a time at most) and cumulative scheduling (some resources can execute several activities in parallel, in our case several part-types, provided that their capacity is not exceeded) is well known. When a cumulative resource intervenes, elastic scheduling, i.e., the possibility of varying the amount of the resource assigned to any activity over time (Baptiste et al., 1999, 2001) may sometimes be applied. Therefore, the problems of the abovementioned family belong to the class of elastic, cumulative scheduling problems, which have mainly been dealt with in the field of project scheduling.

From another point of view, this problem falls into the category of scheduling problems related to failure-prone manufacturing systems (i.e., the system switches between an “operational” state and a “repair” state). Perkins and Srikant (1997) focus on the two part-types case with a failure-prone production system, and in Shu and Perkins (2001) the system spends an exponentially distributed interval of time in a state before switching to the other; while Perkins (2004) assumes exponentially distributed operational times and constant repair times, Bai and Gershwin (1994) focus on multiple part-types; in contrast, Hu and Xiang (1995) assume constant operational times and exponentially distributed repair times; finally, in Corominas and Pastor (2009) the system spends a pre-known

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

operational time in a state, before switching to the other. As is pointed out in Perkins (2004) the failure-prone problems are similar (or "dual") to the problems in which a variable demand stream is assumed (Perkins & Sikrant, 2001; Tan, 2001). Obviously, they are also similar to some problems of production systems with random yields (see, for instance, Ben-Zvi & Grosfeld-Nir, 2007).

The rest of this note is organised as follows: Section 2 defines a lower bound for the problem. Section 3 and Section 4 introduce a test bed and present a computational experiment in which the test bed is used to explain three dynamic programming approaches for determining optimal priority orderings. This test bed, the lower bound defined, the procedures developed and the optimal priority orderings obtained for the instances of the test bed will allow evaluating the performance of future heuristic and metaheuristic procedures. Section 5 outlines brief conclusions.

2 Calculating a lower bound

In Sánchez (2007) it is proved that the sign of the first derivative of the expected sum of quadratic holding and storage costs corresponding to a part-type j, J_j^*, according to D_{j-1}, $\frac{dJ_j^*}{dD_{j-1}}$, is always positive (see an sketch of the proof in the Annex). Therefore, function J_j^* is strictly increasing according to D_{j-1}. This means that the real quadratic holding and storage costs of a part-type j in the ith position of the priority ordering is not less than the cost of this product assuming that the $(i-1)$ preceding products are those of minor demand rates (excluding j).

Therefore, a lower bound for the problem can be calculated as follows: We obtain the quadratic holding and shortage inventory cost to sequence each part-type j $(j=1,...,n)$ in the ith position of the priority ordering $(i=1,...,n)$, taking into account that the $(i-1)$ preceding products are those of minor demand rates (excluding j). Then, a square matrix of costs Q of dimension n is obtained. The lower bound is calculated by resolving the assignment problem with Q. In this note we use the Jonker and Volgenant algorithm (Jonker & Volgenant, 1987), whose code can be found at http://www.magiclogic.com/assignment.html).

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

3 Determining the optimal priority orderings

3.1 Introduction

Shu and Perkins (2001) show that the expected cost corresponding to a part-type does not depend on the order of the part-types with greater priority, but only on the sum of all their demand rates. They also show that placing i directly before j when a part-type i *dominates* a part-type j yields a cost that is no greater than the cost corresponding to placing j directly before i (i.e., in order to find an optimal priority ordering there is no need to take into account orderings in which j is placed immediately before i). The definition of the dominance relations that we use in the present note is as follows, i *dominates* j if and only if one of the three following conditions is fulfilled: (i) $c_i > c_j$ and $c_i \cdot d_i \geq c_j \cdot d_j$; (ii) $c_i = c_j$ and $c_i \cdot d_i > c_j \cdot d_j$; or (iii) $c_i = c_j$, $d_i = d_j$ and $i < j$ (we have added this last condition in order to avoid reciprocity in the dominance relation).

These two properties allow the enumerative effort involved in finding an optimal priority ordering to be reduced. If there are no domination relations between pairs of part-types, or if these relations are not used, the computational complexity of the calculations is proportional to $\sum_{k=1}^{n} n \cdot \binom{n-1}{k-1} = n \cdot 2^{n-1}$. Of course, the domination relations, insofar as they reduce the number of orderings that need to be taken into account, help to reduce the computational effort.

3.2 Three dynamic programming approaches

The first property exposed in the previous subsection leads to approach the optimisation problem with dynamic programming. We have adopted the following decisions in our implementation of a dynamic programming scheme:

- The solutions are coded employing a vector $\{p_1, p_2, ..., p_n\}$, where p_j denotes the part-type sequenced in the jth position of the priority ordering. A state q in the kth stage of the dynamic programming procedure is defined by the set of the k already prioritised part-types (remember that the expected cost corresponding to a part-type does not

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

depend on the order of the part-types with greater priority, but only on the sum of all their demand rates). This is represented by a partial sequence in which prioritised part-types are ordered from lowest to highest according to their product index. If dominance relations are used, the characterisation of a state must also include an indication of which element is the last in the set, p_k. In order to increase the efficiency of the procedure, a bijective correspondence between the code of each state q of the k th stage and its position in the table that contains the states of the k th stage has been established.

- In each state q of the k th stage, the next part-type in the priority ordering, p_{k+1}, needs to be determined. This part-type must not yet be prioritised and, if the dominance relations defined in Section 3.1 are used, this part-type should not be dominated by the last part-type in the partial preceding priority ordering.

- Let Γ_k be the set of states belonging to the k th stage (where $\Gamma_1 = \{\{1\},...,\{n\}\}$); $\chi^*_{k,q}$ and $\chi^*_{k-1,s}$, respectively, the optimal costs corresponding to the state q of k th stage, and to the state s of $k-1$th stage; and $\kappa^*_{(k-1,s),p_k}$, the cost of adding p_k as the k th part-type in the partial priority ordering that defines the state s of $k-1$th stage.

  The recursive function is as follows:

$$\chi^*_{k,q} = \min_{\forall s \in \Gamma_{k-1}} \left(\chi^*_{k-1,s} + \kappa^*_{(k-1,s),p_k} \right) \quad \forall q \in \Gamma_k;\ k = 2,...,n$$

  The costs $\kappa^*_{(k-1,s),p_k} \left(k = 2,...,n\right)$ are computed as indicated for the J^*_j in Section 1

Two dynamic programming approaches were designed: one that takes dominance relations into account, and one that does not. We will denominate these two procedures $DP-Yes\_DR$ and $DP-No\_DR$ respectively.

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

Of course, the limitations of these procedures stem from the fact that the number of states increases exponentially as n increases. It is straightforward that for $n>1$ part-types the maximum number of states is reached at stage $k=\left\lceil \frac{n+1}{2} \right\rceil$ (and also, when $\frac{n+1}{2}$ is an integer, at $k=\left\lceil \frac{n+1}{2} \right\rceil -1$); for instance, the number of states for $n=23$ at stages 11 and 12 is equal to 1,352,078. Finally, we also designed a third dynamic programming approach that uses the lower bound defined in Section 2 but not the dominance relations introduced in Section 3.1. We will denominate this procedure $DP-LB$. The introduction of bounds in the dynamic programming scheme allows reducing the number of states considered in the search process. The operation is as follows:

- First, a feasible solution $\overline{X}$ (and the value of the objective function $\overline{Z}$) is calculated with a heuristic procedure. The heuristic used in this note consisted in determining the ordering of the part-types according to the non-increasing values of the product $d_j \cdot c_j$ (which is the best heuristic developed in Sánchez, 2007).

- A lower bound is calculated in each state q from each kth stage. If the lower bound value is lower than $\overline{Z}$, the next part-type in the priority ordering, p_{k+1}, is determined. If it is not lower than $\overline{Z}$, the state q is rejected.

- If the dynamic programming does not provide a solution, $\overline{X}$ is an optimal solution.

As shown in Section 4, procedure $DP-LB$ yields the worst results. Thus, we decided not to define a dynamic programming approach that used jointly the lower bound and the dominance relations.

In this new scheduling problem, it is necessary to have a procedure for finding optimal solutions to the problem, even for a limited number of part-types, since the optimal priority orderings allow evaluating the performance of future heuristic and metaheuristic procedures.

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

4 Computational experiment

First, we introduce a test bed for the problem. Then we report the results of a computational experiment in which optimal priority orderings for the instances of the test bed were determined using the three dynamic programming approaches defined in Section 3.2.

4.1 A test bed for the problem

To the authors' knowledge, there is no standard test bed for the problem. Therefore, the following test bed was generated: the set consisted of 1,400 instances, 100 instances for each value of n in $[10,23]$; the values of d_j and c_j were generated at random using uniform discrete distributions in $[1,100]$ and $[1,20]$ respectively.

In order to guarantee that the machine had enough capacity to meet demand, the value of μ was set equal to $\alpha \cdot \frac{q_d + q_u}{q_u} \cdot \sum_{j=1}^{n} d_j$, where $\alpha > 1$. Given the statement of the problem, the influence of the values of α and the ratio $\frac{q_d + q_u}{q_u}$ on the computing times was assumed to be insignificant. This assumption was confirmed by performing a short, initial experiment using 25 instances in which $n = 18$ with the values $\alpha = 1.05, 1.10, 1.25, 1.50, 2.00$ and $\frac{q_d + q_u}{q_u} = 3.00, 2.00, 1.50, 1.20, 1.10$. For the different combinations of the parameter values, the computing times for solving each instance turned out to be almost identical. Therefore, the parameter values α and $\frac{q_d + q_u}{q_u}$ could be fixed (specifically, we established that $\alpha = 1.10$ and $\frac{q_d + q_u}{q_u} = 1.20$, and therefore that $\mu = 1.32 \cdot \sum_{j=1}^{n} d_j$).

Presenting the design of the test bed for this new scheduling problem was another purpose of the present technical note. It can be obtained from http://www.ioc.upc.edu/EOLI/ for future developments in order to compare the results of new algorithms or solvers.

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

4.2 Results of a computational experiment

The following three dynamic programming approaches were used in the computational experiment: an approach that only takes dominance relations $(DP-Yes\_DR)$ into account, one that does not $(DP-No\_DR)$, and one that takes into account the lower bound but not the dominance relations $(DP-LB)$. These three procedures have different degrees of calculation difficulty and different numbers of generated states; hence, it was not possible to determine a priori which would be the most efficient.

These three approaches were applied to the set of instances generated in subsection 4.1. The experiment was performed on a PC Pentium IV, at 3.2 GHz, with 1 Gb RAM. Table 1 shows the minimum $(t_{\min})$, average (t_{ave}) and maximum $(t_{\max})$ computing times in seconds, which correspond to applying $DP-No\_DR$.

n	10	11	12	13	14	15	16	17	18	19	20	21	22	23
$t_{\min}$	0.00	0.00	0.02	0.03	0.08	0.44	0.98	2.13	4.36	9.56	21.03	45.27	96.70	207.28
t_{ave}	0.00	0.01	0.02	0.04	0.09	0.47	1.08	2.43	4.79	10.53	22.91	50.70	110.44	231.14
$t_{\max}$	0.02	0.02	0.03	0.06	0.14	0.56	1.20	3.02	5.88	12.91	25.83	55.48	126.80	267.98

Table 1. "Minimum, average and maximum computing times (in seconds) with $DP-No\_DR$".

One can appreciate the low dispersion of the computing times for a given value of n, and furthermore that the ratio between the average computing times corresponding to n and $n-1$ is approximately equal to $2\cdot\frac{n}{n-1}$, as could be expected from the analysis of the complexity of the algorithm. Figure 2 shows the average computing time as a function of the number of part-types, n.

The computing times using $DP-Yes\_DR$ depend on the density of these relations, which is defined as the ratio between the number of actual dominance relations corresponding to the instance and its maximum possible value (which is

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

equal to $n\cdot(n-1)/2$). Table 2 shows the minimum $(\%_{\min})$, average $(\%_{ave})$ and maximum $(\%_{\max})$ density of dominance relations corresponding to the set of instances used in the experiment, expressed as percentages. Table 3 shows the minimum $(t_{\min})$, average (t_{ave}) and maximum $(t_{\max})$ computing times in seconds using $DP-Yes\_DR$.

n	10	11	12	13	14	15	16	17	18	19	20	21	22	23
$\%_{\min}$	35.6	47.3	48.5	57.7	56.0	56.2	50.0	54.4	57.5	59.1	59.5	58.1	61.0	50.2
$\%_{ave}$	76.2	76.0	78.4	75.8	75.6	77.8	77.2	76.5	75.5	75.9	76.7	77.6	75.9	75.5
$\%_{\max}$	97.8	98.2	97.0	92.3	93.4	95.2	94.2	91.2	88.9	90.6	88.4	94.8	87.0	88.9

Table 2. “Minimum, average and maximum density of dominance relations (in %)”.

n	10	11	12	13	14	15	16	17	18	19	20	21	22	23
$t_{\min}$	0.00	0.00	0.02	0.03	0.08	0.19	0.41	0.84	1.91	3.48	7.39	15.42	30.97	72.78
t_{ave}	0.01	0.01	0.03	0.07	0.13	0.27	0.62	1.35	2.90	5.95	11.44	24.79	52.34	108.21
$t_{\max}$	0.02	0.03	0.06	0.24	0.22	0.63	1.38	2.55	5.08	12.83	21.72	44.66	100.58	187.5

Table 3. “Minimum, average and maximum computing times (in seconds) with $DP-Yes\_DR$”.

In general, the computing times are shorter when dominance relations are used than when they are not used. However, the dispersion for a given value of n is greater, as was expected. The average computing times increase exponentially with the value of n, as occurs when dominances are not taken into account (Figure 2).

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

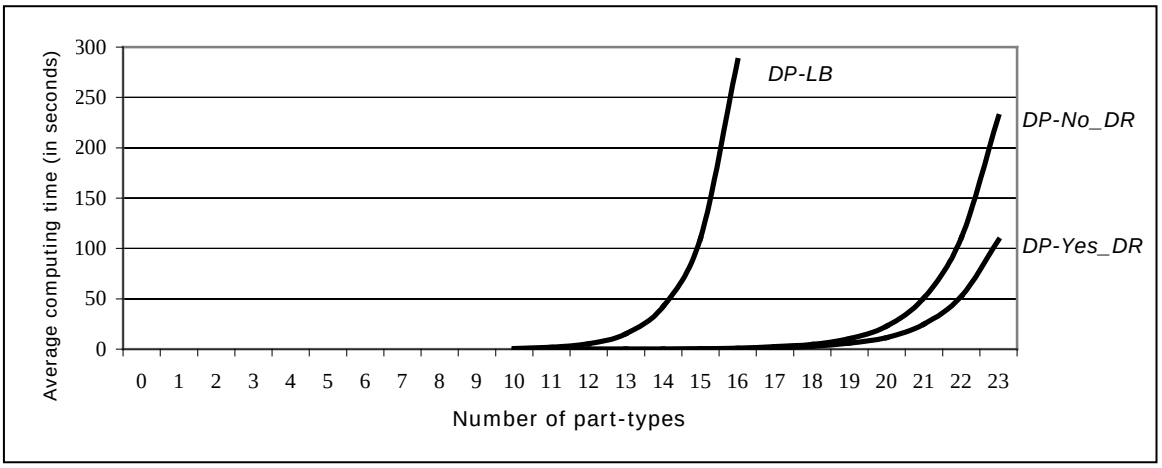

Figure 2. "Average computing time versus number of part-types".

Table 4 allows comparing, for $10 \leq n \leq 16$ the average computing times in seconds that correspond to applying $DP-LB$ and $DP-No\_DR$ ($DP-LB$ is similar to $DP-No\_DR$, but it uses the lower bound).

n	10	11	12	13	14	15	16
$DP-LB$	0.71	1.98	5.38	15.29	42.51	109.94	286.50
$DP-No\_DR$	0.00	0.01	0.02	0.04	0.09	0.47	1.08

Table 4. "Average computing times (in seconds) with $DP-LB$ and $DP-No\_DR$".

With the application of the lower bound, the computing times not only do not decrease but in fact they increase considerably (see also Figure 2). Using the lower bound in the dynamic programming procedure involves a decrease in the number of states compared to not using it, as can be seen in Table 5. The reason why the computing times corresponding to $DP-LB$ are higher than those corresponding to $DP-No\_DR$ may be that, for $DP-LB$, the lower bound has to be calculated in every state and this takes more time than when it is not calculated (when certain rejected states are not considered).

To sum up, the computational experiment shows that it is worthwhile using dominance relations and that instances with up to approximately 25 part-types can be solved in relatively short computing times with the procedure $DP-Yes\_DR$.

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

The values of the optimal solutions of the instances of the test bed can be obtained from http://www.ioc.upc.edu/EOLI/.

n	10	11	12	13	14	15	16
Number of states for $DP-No\_DR$	1023	2047	4095	8191	16383	32767	65535
Average of states for $DP-LB$	632	1265	2745	5876	11992	24810	51445
% Decrease	38.25	38.22	32.97	28.26	26.80	24.28	21.50

Table 5. "Number of states with $DP-No\_DR$ and average of states for $DP-LB$".

5 Conclusions and research prospects

This work establishes optimal priority orderings for prioritised hedging point (PHP) control policies in the problem of minimising the expected sum of quadratic holding and shortage inventory costs when a single, failure-prone machine produces multiple part-types.

In this note, the problem is placed in the field of scheduling and a lower bound for the problem is proposed. Three dynamic programming approaches for determining optimal priority orderings are explained and a test bed is introduced. Finally, a computational experiment in which the algorithms are applied to the test bed is presented. This test bed, the lower bound proposed, the procedures developed and the optimal priority orderings obtained for the instances of the test bed allow the performance of future developments to be evaluated.

The computational experiment shows that it is worthwhile using dominance relations and that instances with up to approximately 25 part-types can be solved in relatively short computing times. Moreover, using the lower bound in a dynamic programming scheme increases the computing time needed. However, as the memory required and the computing time increase exponentially with n, in order to solve larger instances future research must be based on:

a) Using heuristic and metaheuristic procedures.

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

b) Using the lower bound only in some states of the multistage graph of dynamic programming—for example, using the lower bound only in states with a high value of the partial solution built so far, which are more likely to be eliminated by taking into account the lower bound.

Annex

In this Annex it is proved that the sign of the first derivative of the expected sum of quadratic holding and storage costs corresponding to a part-type j, J_j^*, respect to D_{j-1}, $\frac{dJ_j^*}{dD_{j-1}}$, is always positive. Maple © and Derive 5 © commercial software have been used to help in this proof.

According to the terminology presented in Section 1 ($C_j = 2c_j z_j^* \left[\frac{1}{\lambda_j} - \frac{(1-\gamma_{j-1})\gamma_j}{(1-\gamma_j)\gamma_{j-1}\lambda_{j-1}} \right]$, $z_j^* = \frac{1-\gamma_j}{\lambda_j} - \frac{1-\gamma_{j-1}}{\lambda_{j-1}}$, $\lambda_j = \frac{q_u}{D_j} - \frac{q_d}{\mu - D_j}$ and $\gamma_j = \frac{\mu q_u - (q_d + q_u) D_j}{(\mu - D_j)(q_d + q_u)}$), the expected sum of quadratic holding and storage costs corresponding to a part-type j, $J_j^* = C_j - c_j \left(z_j^*\right)^2$, can be expressed as follows (1):

$$J_j^* = [c_j \cdot \mu^3 \cdot q_d \cdot q_u \cdot d_j^2 \cdot (2D_{j-1}^2 \cdot (q_d - q_u) \cdot (q_d + q_u)^2 + D_{j-1} \cdot (q_d + q_u) \cdot (2d_j \cdot (q_d + q_u) \cdot (q_d - q_u) + \mu \cdot q_u \cdot (4q_u - q_d)) + \mu \cdot q_u^2 \cdot (2d_j \cdot (q_d + q_u) - \mu \cdot (q_d + 2q_u)))] / [(q_d + q_u)^2 \cdot (D_{j-1} \cdot (q_d + q_u) - \mu \cdot q_u)^3 \cdot (D_{j-1} \cdot (q_d + q_u) + d_j \cdot (q_d + q_u) - \mu \cdot q_u)^2] \quad (1)$$

Then, we calculate the first derivative of J_j^* respect to D_{j-1}:

$$\frac{dJ_j^*}{dD_{j-1}} = -[2c_j \cdot d_j^2 \cdot \mu^3 \cdot q_d \cdot q_u \cdot (3D_{j-1}^3 \cdot (q_d - q_u) \cdot (q_d + q_u)^2 + D_{j-1}^2 \cdot (q_d + q_u) \cdot (5d_j \cdot (q_d + q_u) \cdot (q_d - q_u) + 3\mu \cdot q_u \cdot (3q_u - q_d)) + D_{j-1} \cdot (2d_j^2 \cdot (q_d - q_u) \cdot (q_d + q_u)^2 + 2d_j \cdot \mu \cdot q_u \cdot (q_d + q_u) \cdot (5q_u - q_d) - 3\mu^2 \cdot q_u^2 \cdot (q_d + 3q_u)) + \mu \cdot q_u \cdot (d_j^2 \cdot (q_d + q_u) \cdot (q_d + 2q_u) - d_j \cdot \mu \cdot q_u \cdot (3q_d + 5q_u) + 3\mu^2 \cdot q_u^2))] / [(D_{j-1} \cdot (q_u + q_d) + d_j \cdot (q_u + q_d) - \mu \cdot q_u)^3 \cdot (D_{j-1} \cdot (q_u + q_d) - \mu \cdot q_u)^4] \quad (2)$$

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

Taking into account that $c_j > 0$, $d_j > 0$, $\mu > 0$, $q_u > 0$, $q_d > 0$, $D_j = \sum_{k=1}^{j} d_k$, $(D_{j-1} \cdot (q_u + q_d) - \mu \cdot q_u)^4 > 0$ and $(D_{j-1} \cdot (q_u + q_d) + d_j \cdot (q_u + q_d) - \mu \cdot q_u)^3 < 0$ (because $(q_u \cdot \mu)/(q_d + q_u) > \sum_{j=1}^{n} d_j$), the sign of (3) has to be studied (it must be positive for the proof):

$$\begin{aligned}&(3D_{j-1}^3 \cdot (q_d - q_u) \cdot (q_d + q_u)^2 + D_{j-1}^2 \cdot (q_d + q_u) \cdot (5d_j \cdot (q_d + q_u) \cdot (q_d - q_u) + 3\mu \cdot q_u \cdot (3q_u - q_d))\\ &+D_{j-1} \cdot (2d_j^2 \cdot (q_d - q_u) \cdot (q_d + q_u)^2 + 2d_j \cdot \mu \cdot q_u \cdot (q_d + q_u) \cdot (5q_u - q_d) - 3\mu^2 \cdot q_u^2 \cdot (q_d + 3q_u))\\ &+\mu \cdot q_u \cdot (d_j^2 \cdot (q_d + q_u) \cdot (q_d + 2q_u) - d_j \cdot \mu \cdot q_u \cdot (3q_d + 5q_u) + 3\mu^2 \cdot q_u^2)) \end{aligned} \quad (3)$$

Given that the units used to express the demand rate may be chosen arbitrarily, one can take, without loss of generality, $d_j = 1$ and replace D_{j-1}/d_j with X. This way, expression (3) is reduced to expression (4):

$$\begin{aligned}&(3X^3 \cdot (q_d - q_u) \cdot (q_d + q_u)^2 + X^2 \cdot (q_d + q_u) \cdot (5 \cdot (q_d + q_u) \cdot (q_d - q_u) + 3\mu \cdot q_u \cdot (3q_u - q_d))\\ &+X \cdot (2 \cdot (q_d - q_u) \cdot (q_d + q_u)^2 + 2\mu \cdot q_u \cdot (q_d + q_u) \cdot (5q_u - q_d) - 3\mu^2 \cdot q_u^2 \cdot (q_d + 3q_u))\\ &+\mu \cdot q_u \cdot ((q_d + q_u) \cdot (q_d + 2q_u) - \mu \cdot q_u \cdot (3q_d + 5q_u) + 3\mu^2 \cdot q_u^2)) \end{aligned} \quad (4)$$

Replacing $(q_u + q_d)$ with C and $\mu \cdot q_u$ with B, expression (4) is reduced to expression (5):

$$\begin{aligned}&3X^3 \cdot (C - 2q_u) \cdot C^2 + X^2 \cdot C \cdot (5C \cdot (C - 2q_u) - 3B \cdot (C - 4q_u))\\ &-X \cdot (3B^2 \cdot (C + 2q_u) + 2B \cdot C \cdot (C - 6q_u) - 2\ C(-2 \cdot q_u) \cdot C^2)\\ &+B \cdot (3B^2 - B \cdot (3C + 2q_u) + C \cdot (C + q_u)) \end{aligned} \quad (5)$$

The expression $(q_u \cdot \mu)/(q_u + q_d) > D_{j-1} + d_j$ is equivalent to expression $B/C > X + 1$ and $C > 0$. Therefore $B > C \cdot (X + 1)$ and we can set $B = A \cdot C \cdot (X + 1)$, where $A > 1$ (see expression (6)):

$$\begin{aligned}&3A^3 \cdot C^3 \cdot X^3 - 3A^2 \cdot C^3 \cdot X^3 - 6A^2 \cdot C^2 \cdot X^3 \cdot q_u - 3A \cdot C^3 \cdot X^3 + 12A \cdot C^2 \cdot X^3 \cdot q_u + 3C^3 \cdot X^3 - 6C^2 \cdot X^3 \cdot q_u\\ &+9A^3 \cdot C^3 \cdot X^2 - 9A^2 \cdot C^3 \cdot X^2 - 14A^2 \cdot C^2 \cdot X^2 \cdot q_u - 5A \cdot C^3 \cdot X^2 + 24A \cdot C^2 \cdot X^2 \cdot q_u + 5C^3 \cdot X^2 - 10C^2 \cdot X^2 \cdot q_u\\ &+9A^3 \cdot C^3 \cdot X - 9A^2 \cdot C^3 \cdot X - 10A^2 \cdot C^2 \cdot X \cdot q_u - A \cdot C^3 \cdot X + 13A \cdot C^2 \cdot X \cdot q_u + 2C^3 \cdot X - 4C^2 \cdot X \cdot q_u\\ &+3A^3 \cdot C^3 - 3A^2 \cdot C^3 - 2A^2 \cdot C^2 \cdot q_u + A \cdot C^3 + A \cdot C^2 \cdot q_u \end{aligned} \quad (6)$$

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

The sign of the coefficients of X^3, X^2, X^1 and X^0 can be studied separately, in order to show that they are always positive and that, therefore, taking into account that $X \geq 0$, the whole expression (2) is positive.

As an example, we prove that the sign of the coefficient of X^3 is positive. Expression (7) shows, taken from (6) the expression that gives the value of this coefficient:

$$3A^3 \cdot C^3 - 3A^2 \cdot C^3 - 6A^2 \cdot C^2 \cdot q_u - 3A \cdot C^3 + 12A \cdot C^2 \cdot q_u + 3C^3 - 6C^2 \cdot q_u \quad (7)$$

Whose sign is the same than that (8):

$$A^3 \cdot C - A^2 \cdot C - 2A^2 \cdot q_u - A \cdot C + 4A \cdot q_u + C - 2q_u = C \cdot (A^3 - A^2 - A + 1) + q_u \cdot (-2A^2 + 4A - 2) \quad (8)$$

Remember that $A > 1$. Then, the coefficient of C is positive $(A^3 - A^2 - A + 1 > 0)$. However, the coefficient of q_u is negative $(-2A^2 + 4A - 2 = -(A-1)^2)$.

As $C = (q_u + q_d)$, to prove that $C \cdot (A^3 - A^2 - A + 1) + q_u \cdot (-2A^2 + 4A - 2) > 0$ it suffices to show that $C \cdot (A^3 - A^2 - A + 1) + C \cdot (-2A^2 + 4A - 2) > 0$, which is equivalent to show that $(A^3 - A^2 - A + 1) + (-2A^2 + 4A - 2) = A^3 - 3A^2 + 3A - 1 > 0$. And this last proposition is straightforward, since $A > 1$ and the expression is equal to 0 for $A = 1$ and its derivative $\left(3A^2 - 6A + 3\right)$ is > 0 for $A > 1$.

References

Bai, S. X., & Gershwin S. B. (1994). Scheduling manufacturing systems with work-in-process inventory control: multiple-part-type systems. *International Journal of Production Research,* 32, 365-385.

Baptiste, Ph., Le Pape, C., & Nuijten, W. (1999). Satisfiability tests and time_bound adjustments for cumulative scheduling problems. *Annals of Operations Research,* 92, 305-333.

Baptiste, Ph., Le Pape, C., & Nuijten, W. (2001). *Constraint-based scheduling. Applying Constraint Programming to Scheduling Problems*. Kluwer.

doi:10.3926/jiem.2009.v2n3.p418-436 © © JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953

Ben-Zvi, T., & Grosfeld-Nir, A. (2007). Serial production systems with random yields and rigid demand: A heuristic. *Operations Research Letters,* 35, 235-244.

Corominas, A., & Pastor, R. (2009). Scheduling production of multiple part-types in a system with pre-known demands and deterministic inactive time intervals. *European Journal of Operational Research,* 193, 639-643.

Hu, J. Q., & Xiang D. (1995). Optimal control for systems with deterministic production cycles. *IEEE Transactions on Automatic Control,* 40, 782-786.

Jonker, R., & Volgenant, A. (1987). A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing,* 38, 325-340.

Ketzenberg, M., Metters, R., & Semple, J. (2006). A heuristic for multi-item production with seasonal demand. *IIE Transactions,* 38, 201-211.

Perkins, J. R. (2004). Optimal control of failure-prone manufacturing systems with constant repair-times. *Annals of Operations Research,* 125, 233-261.

Perkins, J. R., & Srikant, R. (1997). Scheduling multiple partt-types in an unreliable single machine manufacturing system. *IEEE Transactions on Automatic Control,* 42, 364-377.

Perkins J. R., & Srikant, R. (2001). Failure-prone production systems with uncertain demand. *IEEE Transactions on Automatic Control,* 46, 441-449.

Sánchez, A. (2007). Determinación de secuencias en una máquina multiproducto sujeta a fallos y con costes cuadráticos. *Doctoral Thesis*, Universitat Politècnica de Catalunya, Barcelona.

Shu, C., & Perkins, J. R. (2001). Optimal PHP production of multiple part-types on a failure-prone machine with quadratic buffer costs. *IEEE Transactions on Automatic Control,* 46, 541-549.

Sethi S. P., & Thompson, G. L. (2000). *Optimal control theory: Applications to management science and Economics, 2nd edition.* Kluwer.

Tan, B. (2001). Production control of a pull system with production and demand uncertainty. *Working paper*, Graduate School of Business, Koç University.

Journal of Industrial Engineering and Management

doi:10.3926/jiem.2009.v2n3.p418-436 ©© JIEM, 2009 – 2(3): 418-436 - ISSN: 2013-0953