Experimentos con dos estrategias de búsqueda implementadas en el algoritmo recocido simulado para un problema de inventario

Tests with two heuristic search strategies in the simulated annealing algorithm for an inventory problem

Salvador Hernández González[*]

Departamento de Ingeniería Industrial. Instituto Tecnológico de Celaya. Antonio García Cubas s/n, C.P. 38010. Celaya, México

(Recibido el 1 de mayo de 2011. Aceptado el 10 de febrero de 2012)

Resumen

El principal problema en los resultados obtenidos con metaheurísticas implementadas para resolver instancias del problema de reaprovisionamiento multiproducto es el deterioro de la calidad de la solución. Se ha observado que las implementaciones se han concentrado en los parámetros del algoritmo prestando poca atención a la estrategia para acceder a la solución vecina. En este trabajo se estudia experimentalmente una implementación del algoritmo Recocido Simulado explorando los parámetros del algoritmo, además se estudian dos esquemas de obtención de la solución vecina realizando las comparaciones con el algoritmo RAND. El estudio se realiza mediante un diseño 2 factorial sobre 2.000 instancias generadas aleatoriamente, los resultados muestran que bajo las mismas combinaciones de parámetros del algoritmo recocido simulado, el esquema de perturbar una variable a la vez proporciona resultados muy pobres ya que devuelve la solución óptima con menor frecuencia, en cambio al acceder a la solución vecina tomando en cuenta grupos de productos, se obtienen mejores resultados y el algoritmo Recocido Simulado se comporta de manera robusta frente al incremento en el tamaño del problema.

----- *Palabras clave:* Metaheuristicas, Recocido simulado, inventarios, diseño experimental

Abstract

The problem in the results when implemented metaheuristics techniques to solve instances of the problem of multi-product replenishment is

* Autor de correspondencia: teléfono: + 01 + 461 + 611 75 75, correo electrónico: salvador.hernandez@itcelaya.edu.mx (S. Hernández)

the deterioration of the quality of the solution. It was noted that the implementations have focused on the parameters of the algorithm paying little attention to the strategy to access the neighboring solution. In this paper, we study experimentally an implementation of Simulated Annealing algorithm exploring several combinations of parameters and also, two schemes for obtaining the neighbor solution by comparison with the RAND algorithm. A 2-factorial experimental design was constructed and the study was conducted over 2.000 randomly generated instances, the results show that under the same combinations of parameters, perturbing a variable at a time provides poor results, however, to access the neighbor solution taking into account families of products provides better results and Simulated Annealing algorithm behaves robust against the increase in the size of the problem.

----- *Keywords:* metaheuristics, simulated annealing, inventory, experimental design

Introducción

El inventario se emplea en la mayoría de las empresas de manufactura, servicios, y distribución, debido a que es un factor básico en la medición del desempeño de la rentabilidad de una empresa. Sin embargo, frecuentemente las necesidades de controlar el inventario van más allá de un producto o de un solo proveedor. Al problema de determinar la frecuencia de producción o pedido de varios productos se le conoce como problema de reaprovisionamiento conjunto (Joint replenishment problem, JRP). Este problema es muy importante en control de inventarios y ha sido estudiado profusamente en los últimos 30 años, sin embargo persisten algunos problemas concernientes a los métodos de solución propuestos y que han motivado el diseño de nuevas técnicas [1].

Antecedentes

Goyal en 1974 [1] reporta el primer algoritmo conocido para encontrar el óptimo; sin embargo al ser de enumeración exhaustiva el tiempo de ejecución puede volverse prohibitivo, si bien existen modificaciones del algoritmo, estas se han enfocado en encontrar nuevas cotas pero continúan empleando la misma estrategia de búsqueda exhaustiva generando la necesidad de continuar desarrollando procedimientos heurísticos para resolver instancias del problema.

Algunos métodos heurísticos reportados se pueden consultar en [2-4], las propuestas más recientes se encuentran en [5, 6]. En lo que respecta a metaheurísticas, se han realizado algunas implementaciones para el problema de reaprovisionamiento multiproducto: se han reportado Algoritmos Genéticos (AG) en [7, 8] donde comparan contra el algoritmo RAND, y en [9] se implementa un híbrido Recocido Simulado – Algoritmo Genético.

En los resultados con las metaheurísticas implementadas, la calidad de la solución se deteriora muy rápido a medida que se incrementa el tamaño de la instancia, en comparación el algoritmo RAND es más robusto [4, 7, 8]. Hay que recordar que el desempeño de una metaheurística depende de dos aspectos: los parámetros seleccionados para su funcionamiento y en segundo lugar (y no menos importante), la estrategia de búsqueda implementada; es aquí donde se mantiene abierta la cuestión ya que se presta poca atención a estudiar el esquema para acceder a la solución vecina y en los último años, no se ha propuesto otra estrategia de exploración del entorno (N) de soluciones vecinas, existe aún esta laguna.

En este trabajo se implementa el algoritmo Recocido Simulado explorando distintas combinaciones de parámetros, y a diferencia de trabajos anteriores, aquí se comparan dos estrategias para acceder a la solución vecina. Ya que se trata de un estudio exploratorio, las pruebas se realizan mediante un diseño experimental factorial, para comparar los resultados se utiliza el algoritmo RAND, el cual es una heurística que ha sido ampliamente utilizada como referencia para comparar los distintos métodos diseñados para el problema de reaprovisionamiento multiproducto[3, 6-8].

En el siguiente apartado se presenta el modelo de costo, se continúa con una breve descripción del algoritmo RAND, posteriormente se describe el algoritmo recocido simulado, enseguida se muestra la implementación del algoritmo, después se presentan los experimentos con los resultados obtenidos y su discusión, se finaliza con las conclusiones y las referencias.

Modelo de costo

Primero se definen los parámetros del modelo: S es el costo de pedido o de activación de la orden de compra y es independiente del número de productos incluidos en dicha orden, s_i es el costo en que se incurre por pedir el producto i, D_i es la demanda del producto i, h_i es el costo de guardar o mantener inventario del producto i y n es el número de productos.

El costo total (CT) es la suma del costo de pedido y el costo de guardar o mantener el inventario. Los responsables deben calcular cada cuánto tiempo debe incluirse el producto i en la orden de compra, dado que es posible operar con costos más bajos que si se solicita cada producto por separado.

Como restricción adicional, dicho lapso se restringe a un múltiplo entero de un ciclo base de tiempo (o periodo base) expresado como $T_i = k_i T$, donde T es el tiempo que transcurre entre dos pedidos o activaciones consecutivas de la orden general de compra, y k_i es la frecuencia con la que se solicita específicamente el producto i. El modelo de optimización es el siguiente:

$$\min \quad CT = \frac{1}{T}\left(S + \sum_{i=1}^{n} \frac{s_i}{k_i}\right) + \frac{1}{2} T \sum_{i=1}^{n} k_i D_i h_i \qquad (1)$$

$$\text{sujeto a :} \quad T > 0 \qquad (2)$$

$$k_i \geq 1, \text{ enteros}, \forall\ i = 1,2,...,n. \qquad (3)$$

En el modelo, la ecuación (1) es la función de costo y debe minimizarse, la ecuación (2) restringe la variable T a valores mayores a cero, la frecuencia del producto i se restringe a valores enteros con la ecuación (3).

En el problema de reaprovisionamiento de productos múltiples se debe calcular la frecuencia óptima de pedido k_i^* (un valor entero) de cada producto y el ciclo base de tiempo T^*. Cabe señalar que para cada combinación de frecuencias existe un valor T que minimiza el costo localmente (figura 1) [1].

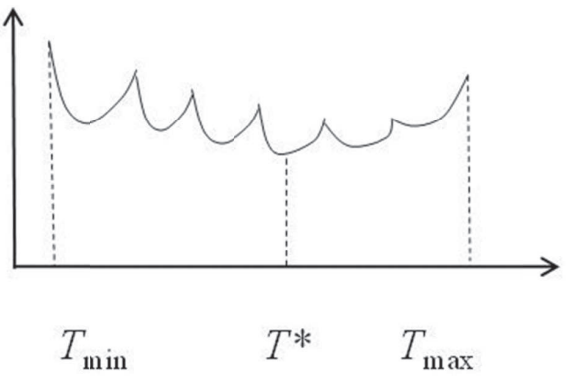


Figura 1 Forma aproximada de la función objetivo. Existen varios mínimos locales

La estrategia de la heurística RAND

La heurística RAND es un método desarrollado por Kaspi y Rossenblat [3, 4, 7], su funcionamiento básicamente consiste en: calcular el intervalo $[T_{\min}, T_{\max}]$ y dividirlo en m segmentos de igual tamaño, a continuación realiza una búsqueda local dentro de cada segmento r, a un valor dado de T se calculan las frecuencias óptimas de pedido de

cada producto con la ecuación (4) redondeándose a continuación al valor entero aplicando la ecuación (5), con estos valores de frecuencia calculados se obtiene el nuevo valor de *T*, los pasos se repiten y la solución va convergiendo (se sugiere consultar [4,7] para ver la deducción completa de las ecuaciones y la descripción del algoritmo RAND). Los autores sugieren un valor de *m*=10, ya que argumentan que valores mayores no presentan mejoras significativas en el costo y en cambio se incrementa el esfuerzo computacional [4, 7].

$$k_i = \frac{2s_i / D_i h_i}{T^2} \tag{4}$$

$$\sqrt{k_i(k_i - 1)} \le k_i < \sqrt{k_i(k_i + 1)} \tag{5}$$

Recocido simulado

Las técnicas metaheurísticas son un conjunto de algoritmos generales para optimización combinatoria; una de sus características es que no están diseñados para un problema concreto, todo lo contrario, son procedimientos muy flexibles y esto permite que sea posible aplicarlos a la gran mayoría de los problemas combinatorios. Las técnicas más populares son los Algoritmos Genéticos, la Búsqueda Tabú y el Recocido Simulado [10].

El Recocido Simulado (RS) es un procedimiento introducido por Kirkpatrick, Gelatt y Vecci [11] y ha sido muy empleado para resolver problemas combinatorios. Partiendo de una solución *x*, se selecciona a continuación una solución vecina *x*′ dentro de cierta vecindad o región *N*, posteriormente se evalúa *f*(*x*) y *f*(*x*′). Si la solución vecina mejora el valor de la función objetivo se acepta con probabilidad 1, en caso contrario se calcula la probabilidad para aceptarla aplicando el criterio de Metrópolis (figura 2).

Al valor *c* se le conoce como el parámetro de control; el valor inicial c_0 debe ser tal que al inicio las soluciones de mala calidad tengan una probabilidad alta de ser seleccionadas, a medida que *c* disminuye la probabilidad de aceptar soluciones de pobre calidad disminuye también, el algoritmo se detiene por ejemplo cuando el valor de *c* rebasa un umbral ε, finalmente el número de soluciones vecinas *n* evaluadas para cierto valor de *c* debe asegurar una buena exploración del entorno [10, 11].

El algoritmo:
Leer x, c_0, ε
Repetir
  Repetir (para cierto c)
    Seleccionar aleatoriamente x' en vecindad $N(x)$.
    Sea $\Delta = f(x') - f(x)$
      Si $\Delta < 0$ entonces $x = x'$
      En otro caso generar aleatoriamente $\lambda \in U(0,1)$
        Si $\lambda < \exp(\Delta/c)$ entonces $x = x'$
  Hasta llegar a equilibrio.
Actualizar temperatura c.
Hasta condición de parada ε

Figura 2 Algoritmo de recocido simulado

Implementación

A continuación se muestra la manera en que se implementó el algoritmo RS.

Función de costo

Fijando los valores de la frecuencia y aplicando las condiciones de 1er orden en la ecuación (1) se obtiene la función de costo que depende únicamente de los valores de frecuencia k_i:

$$CT = \sqrt{\left[\sum_{i=1}^{n} D_i h_i k_i\right]\left[2\left(S + \sum_{i=1}^{n} \frac{s_i}{k_i}\right)\right]} \tag{6}$$

$$T_{\min} = \min T_i^{ind} = \min_i \left[\left(\frac{2s_i}{D_i h_i}\right)^{1/2}\right] \tag{7}$$

$$T_{\max} = \left[\frac{2\left(S + \sum_{i=1}^{n} s_i\right)}{\sum_{i=1}^{n} D_i h_i}\right] \tag{8}$$

Ciclo base de tiempo

El valor óptimo del ciclo base de tiempo puede acotarse a un intervalo $T_{\min} \le T^* \le T_{\max}$ [1, 4, 7]. La cota inferior se calcula determinando primero el ciclo óptimo individual T_i^{ind} de cada producto *i* y posteriormente se selecciona el menor aplicando la ecuación (7). Por otro lado con la ecuación (8) se calcula la cota superior del ciclo base de tiempo $T_{\max}$.

Espacio de soluciones para valores de k

En el problema de reaprovisionamiento, también el valor del ciclo de tiempo óptimo de cada producto *i* se encuentra en el intervalo $T_{\min} \le T_i \le T_i^{\max}$, donde $T_i^{\max} = T_i^{ind}$ y es específico de cada producto.

Como ya se mencionó T_i se restringe a un múltiplo entero del ciclo base; por lo que el espacio de búsqueda de cada producto puede reescribirse en función de las frecuencias como $1 \le k_i \le k_i^{\max}$, donde el máximo valor de frecuencia del producto *i* puede obtenerse con la ecuación (9) y el mínimo valor posible de la frecuencia de pedido es obviamente k_i=1. Esto acota en gran medida el espacio de búsqueda para los valores de frecuencia, a diferencia de otras implementaciones donde las combinaciones posibles son mucho mayores [7].

$$k_i^{\max} = \frac{T_i^{ind}}{T_{\min}} \qquad (9)$$

Estructura de vecindades y solución vecina

Para acceder a la solución vecina es necesario efectuar dos pasos: seleccionar el índice *i* y aplicar un paradigma de búsqueda (perturbación o construcción). En esta implementación y una vez que se han calculado los valores de $k_i^{\max}$, los productos se arreglan en orden ascendente en base a su $k_i^{\max}$ de tal manera que se tendrán los productos agrupados de acuerdo a su frecuencia: frecuencia 1, frecuencia 2, y así sucesivamente.

$$\Pr(\text{seleccionar } i) = \frac{k_i^{\max}}{\sum_{i \in k_i^{\max} > 1} k_i^{\max}} \qquad (10)$$

La selección del índice *i* es al azar, pero para mantener un cierto control se procedió como sigue: una vez ordenados los productos de acuerdo al valor $k_i^{\max}$, con la ecuación (10) se determina la probabilidad de selección de cada índice que, como se observa, favorece la selección de aquellos que tienen un espacio de soluciones más amplio. Aquellos índices con $k_i^{\max} = 1$ no se contabilizan. El segundo paso consiste en acceder a la solución vecina; suponga que se tiene una solución con el siguiente arreglo:

$$1,2,2,3,3,3,3,4,4$$

Dicho arreglo corresponde a una solución donde las frecuencias de pedido de los productos son $k_1 = 1, k_2 = k_3 = 2$, $k_4 = k_5 = k_6 = k_7 = 3$ y $k_8 = k_9 = 4$, de costo *CT*. Se describen a continuación los dos esquemas estudiados.

a. Enfoque de perturbación individual. En este caso, el vector vecino de frecuencias *K'* se obtiene modificando la frecuencia de un solo producto en una unidad. Las únicas operaciones posibles son: $k_i + 1$ o bien $k_i - 1$ con probabilidad 0,5 cada una. La selección es como sigue: se genera un número aleatorio *a* utilizando una distribución uniforme en el intervalo *U*(0,1), si *a* < 0,5 entonces se incrementa la frecuencia en una unidad, en otro caso la frecuencia se disminuye en una unidad, como se muestra en la ecuación (11).

$$\Pr(\text{seleccionar operación}) = \begin{cases} k_i + 1 & \text{Si } a < 0{,}5 \\ k_i - 1 & \text{En otro caso} \end{cases} \qquad (11)$$

Una vez que se obtiene la nueva frecuencia, se evalúa el costo *CT'* de la nueva solución y se compara siguiendo lo especificado para el algoritmo RS.

b. Enfoque de construcción de familias. En este caso la solución vecina *K'* se obtiene al cambiar la frecuencia de pedido k_i por familias de productos (a lo largo del documento nos referiremos también a este enfoque con el nombre de esquema en grupo). Para el ejemplo anterior, suponga que se selecciona el producto 5 y se incrementa su

frecuencia de pedido en una unidad empleando la ecuación (11), el arreglo quedará:

$$1,2,2,3,4,3,3,4,4$$

El producto 5 tiene ahora una frecuencia de pedido $k_5 = 4$, sin embargo los productos 6 y 7 no pertenecen a la misma familia (tienen frecuencia 3), la idea consiste en que si se aumenta la frecuencia de *i* entonces todos los productos desde $i+1, i+2, ..., l$ deberán ahora incorporarse al siguiente grupo o familia de frecuencia $k_i + 1$, en el ejemplo *l* = 7:

$$1,2,2,3,4,4,4,4,4$$

Esta será una nueva solución con ciclo de tiempo costo *CT'*. A continuación se aplica el criterio de aceptación de RS. Considere ahora el caso contrario, suponga que empleando la ecuación (11) se disminuye la frecuencia de pedido del producto 5 en una unidad, el arreglo queda como sigue:

$$1,2,2,3,2,3,3,4,4$$

El producto 5 tiene ahora una frecuencia de pedido $k_5 = 2$, sin embargo ahora el producto 4 debe incorporarse al grupo de productos con frecuencia 2:

$$1,2,2,2,2,3,3,4,4$$

Que corresponde a otra solución de costo *CT'*. En otras palabras, si se disminuye la frecuencia de *i*, todos los productos $i-1, i-2, ..., l$ deberán ahora incorporarse al grupo de frecuencia $k_i - 1$, en el ejemplo *l* = 4. Como se observa, en este esquema se busca la mejor agrupación por frecuencias de los productos.

En lo concerniente a los valores de los parámetros de RS, estos se exploraron a través de un diseño experimental, a continuación se describen de forma general algunos aspectos de los mismos.

Esquema de enfriamiento

Se aplicó el esquema geométrico de enfriamiento para actualizar el valor del parámetro de control *c*, dado por la siguiente ecuación:

$$c = \alpha * c \qquad (12)$$

Este esquema se utiliza en las primeras implementaciones de RS, el valor de la rapidez de enfriamiento α se selecciona dentro del intervalo [0,8 - 0,9], un valor muy bajo implica un enfriamiento rápido y una probabilidad mayor de converger a un mínimo local, en caso contrario, un valor más alto implica una mayor probabilidad de converger al mínimo global pero a costa de un tiempo de ejecución mayor [10, 11].

Criterio de paro

El factor de control desciende hasta que se considera que el sistema está "frío", para el algoritmo se fija un valor tal que $c < \varepsilon$, donde $\varepsilon = 0,1$.

Puntos explorados

Se deben seleccionar de manera aleatoria un cierto número de puntos o soluciones, suficientes para asegurar que la región se explora de manera adecuada. En ésta implementación, ya que se desea estudiar por el momento el efecto del esquema de perturbación, dicho valor se fija igual al número de productos *n*.

Experimentos, resultados y discusión

Los algoritmos de RAND y RS se programaron en FORTRAN 94 y las corridas se realizaron en una PC con procesador AMD Dual Core a 2,3 Ghz y 4 Gb de memoria RAM. Para el algoritmo RAND se empleó un valor *m*=10 recomendado por los autores. Al no existir librería de instancias estas se generan de forma aleatoria siguiendo lineamientos fijados en la literatura para el costo de activación mayor *S* y el número de productos *n* y se muestran en la tabla 1 [6, 7, 8]. Para cada combinación de

S y *n* se construyeron 100 instancias generando los valores de demanda (*D*), costo de mantener inventario (*h*) y costo de pedido individual (*s*) de manera aleatoria empleando una función de distribución uniforme (*U*), se generaron en total 2.000 problemas (tabla 1).

Tabla 1 Intervalos de generación de los parámetros de demanda, y costos de inventario y pedido individual

Parámetro	*Intervalo*
Número de productos (*n*)	10, 20, 30, 50
Costo mayor de activación (*S*)	5, 10, 15, 20, 30
Demanda (*D*)	*U* (100 - 100.000)
Costo de mantener inventario (*h*)	*U* (0,5 - 5)
Costo de pedido individual (*s*)	*U*(2 - 3)

Cuando se implementan metaheurísticas, los diseños experimentales permiten construir un conjunto de pruebas controladas para examinar de forma empírica su desempeño. En este orden de ideas se aplicó un diseño experimental factorial de dos niveles para las pruebas con RS, en dicho diseño cada factor se evalúa en 2 puntos (nivel alto y bajo), los factores tomados en cuenta y sus niveles se muestran en la tabla 2; en cuanto al esquema de obtención de la solución vecina se trata de una variable categórica [12].

Tabla 2 Diseño experimental

Factor	*Nivel*	
	Alto	*Bajo*
c_0 (80-50 % acept. sol de mala calidad)	50	1
α (Recomendados en literatura)	0,95	0,9
Puntos explorados(soluciones vecinas exploradas)	*n*	*n*
	Categórica	
Perturbación	Individual	Familias

De cada combinación de parámetros de RS mostrados en la tabla 2 se realizaron 2 réplicas y se agregaron 3 puntos centrales, siendo 2 variables categóricas a cada una le corresponden 4 combinaciones x 2 réplicas + 3 puntos centrales=11 corridas por cada variable categórica. Para medir el comportamiento de los algoritmos RS y RAND se calculó el % de problemas en los que cada algoritmo devuelve la solución óptima. La solución óptima de cada problema se calculó con el procedimiento de Goyal propuesto en 1974 [1].

Los resultados de cada variable categórica se muestran separadas sólo a manera de ilustración en las tablas 3 y 4, las cuales se describen a continuación: en la primera columna se tiene identificado el experimento, las columnas 2 y 3 identifican la combinación de valores de los parámetros de RS, la columna 4 el tipo de estrategia y la columna 5 muestra la proporción de problemas en los que RS devuelve la solución óptima. Las corridas con asterisco corresponden a los puntos centrales.

Tabla 3 Resultados experimentales del esquema individual, % de óptimos obtenidos

Experimento	c_0	α	*Estrategia*	*% óptimos*
2	1	0,95	individual	91,8
4*	25,5	0,925	Individual	96
5	1	0,95	Individual	91,75
6*	25,5	0,925	Individual	95,9
7	50	0,95	Individual	96,85
9	50	0,95	Individual	96,8
12	1	0,9	Individual	77,5
16*	25,5	0,925	Individual	95,9
17	1	0,9	individual	76,76
19	50	0,9	individual	95
22	50	0,9	individual	91,8
RAND (*m*=10)				98,05 (1.961)

Tabla 4 Resultados experimentales del esquema por familias, % de óptimos obtenidos

Experimento	c_0	α	*Estrategia*	*% óptimos*
1	50	0,95	Familias	99,45
3	50	0,95	Familias	99,7
8	1	0,9	Familias	96,25
10	1	0,95	Familias	98
*11	25,5	0,925	Familias	99,2
13	50	0,9	Familias	98,85
14	1	0,9	Familias	97,36
*15	25,5	0,925	Familias	99,15
*18	25,5	0,925	Familias	99
20	50	0,9	Familias	98
21	1	0,95	Familias	97,2
RAND (m=10)				98,05 (1.961)

En una primera inspección visual de los resultados, el esquema individual de perturbación da resultados muy pobres ya que el % de óptimos de que devuelve RS está por debajo de la estrategia en grupo y del algoritmo RAND (tabla 3), sin embargo en el caso del esquema en familias (tabla 4) los resultados obtenidos son más alentadores. Con la combinación $c_0 = 1/\alpha = 0{,}9$ el algoritmo RS en promedio devuelve el óptimo en 96,805% de los problemas resueltos, con la combinación $c_0 = 1/\alpha = 0{,}95$ devuelve el 97,6%, es decir, incrementar el valor de α mejora el comportamiento del algoritmo. Las réplicas con $c_0 = 50/\alpha = 0{,}9$, en promedio devuelve el óptimo en 98,425% de las instancias.

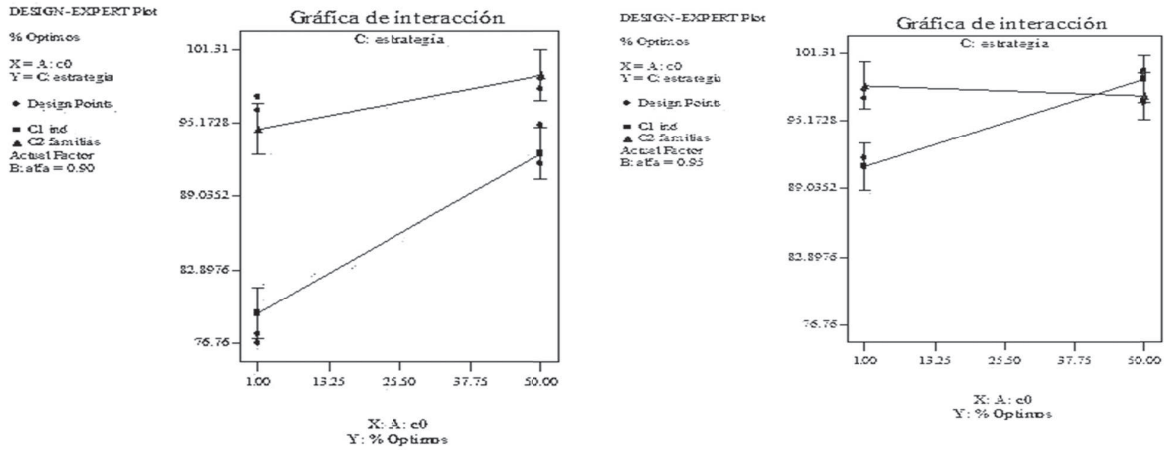
En general bajo las mismas combinaciones de parámetros, la estrategia de búsqueda en familias obtiene con mayor frecuencia la solución óptima, y los resultados son muy similares a los que se obtienen con el algoritmo RAND. La tabla 5 muestra el ANOVA del experimento y los efectos significativos: en primer lugar la estrategia de búsqueda, a continuación el valor inicial del parámetro de control, el parámetro de enfriamiento, seguido de las interacciones c_0-estrategia de búsqueda, α-estrategia de búsqueda y c_0-α.

El resultado de la falta de correlación indica la presencia de curvatura en la región y la conveniencia de emplear un modelo cuadrático, sin embargo y por cuestiones de espacio el análisis se limita a explorar el efecto de la estrategia de búsqueda utilizada.

Tabla 5 ANOVA, variables codificadas

Fuente	*Suma de* Cuadrados	*Cuadrado* medio	*F*	Prob > F	
Modelo	691,64	115,27	23,38	< 0,0001	Significativo
A (c_0)	155,19	155,19	31,48	< 0,0001	
B (α)	100,15	100,15	20,31	0,0005	
C (estrategia)	263,24	263,24	53,4	< 0,0001	
AB (c_0-α)	29,51	29,51	5,99	0,0282	
AC (c_0-estrategia)	78,54	78,54	15,93	0,0013	
BC (α-estrategia)	65	65	13,19	0,0027	
Curvatura	56,02	56,02	11,36	0,0046	Significativo
Falta de correlación	62,27	31,13	55,32	< 0,0001	significativo

La figura 3 (a) muestra que con α = 0,90 es conveniente utilizar un valor c_0 alto, sin embargo los resultados obtenidos con la estrategia en familias dominan a la estrategia individual. En la figura 3(b) se observa que con α = 0,95 y la estrategia individual es conveniente utilizar un valor c_0 alto; en el caso de la estrategia en grupo se observa que el efecto por modificar el valor c_0 es menor, aún así domina (aunque en menor medida) a la estrategia individual.



(a) (b)

Figura 3 Interacción c_0-estrategia de búsqueda: a). α =0,90 b). α =0,95

En la tabla 6 se muestra la forma en que se distribuyeron las proporciones de acuerdo al tamaño del problema en cada experimento con RS en los niveles altos de c_0, α y la estrategia en familias (experimentos 1 y 3), y para el algoritmo RAND con m=10. A partir de n= 20 productos el algoritmo RAND empieza a mostrar cierta dificultad para devolver la solución óptima; en el caso del algoritmo RS esto se empieza a presentar a partir de n =30 productos.

Tabla 6 % de óptimos obtenidos de acuerdo al valor de n y costos de penalización (entre paréntesis)

n	*S*	*RS1*	*RS3*	*RAND 10*
10	5	100	100	100
	10	100	100	100
	15	100	100	100
	20	100	100	100
	30	100	100	100
20	5	100	100	99 (0,85%)
	10	100	100	97 (1,92%)
	15	100	100	100
	20	100	100	100
	30	100	100	100

n	*S*	*RS1*	*RS3*	*RAND 10*
30	5	99 (0,27%)	99 (0,27%)	96 (1,17%)
	10	99 (0,2%)	100	95 (0,85%)
	15	100	100	100
	20	100	100	98 (0,835%)
	30	100	100	98 (0,4%)
50	5	92(0,45%)	97 (0,36%)	91 (1,94%)
	10	99 (0.00047)	99 (0.00047%)	95 (0,78%)
	15	100	100	96 (2,07%)
	20	100	99	97 (1,43%)
	30	100	100	99 (0,77%)
	Total	1.989	1.994	1.961
	%	99,45	99,7	98,05

Se observa que la dificultad se presenta para valores bajos de *S* y *n*, que son la clase de problemas más difíciles de resolver [6, 7, 8]. Entre paréntesis se muestra el % de diferencia promedio con respecto a la solución óptima (costo de penalización como lo han llamado algunos autores, ver por ejemplo [6, 7, 8]) . Se observa que existen costos de penalización a partir de 20 productos cuando se resuelven las instancias con el algoritmo RAND, en cambio con RS y la estrategia en familias la penalización se presenta a partir de 30 productos y en menor proporción.

Los tiempos de ejecución se muestran en la figura 4. RS tiene un tiempo de ejecución que crece más rápido. Una desventaja reconocida del RS es el consumo de tiempo, que en este caso es evidente a partir de 30 productos, en comparación el algoritmo RAND requiere en promedio 2 milisegundos como máximo de tiempo de ejecución. Cabe señalar que en una investigación posterior se pueden explorar nuevas combinaciones de parámetros de RS que permitieran reducir el tiempo de ejecución requerido.

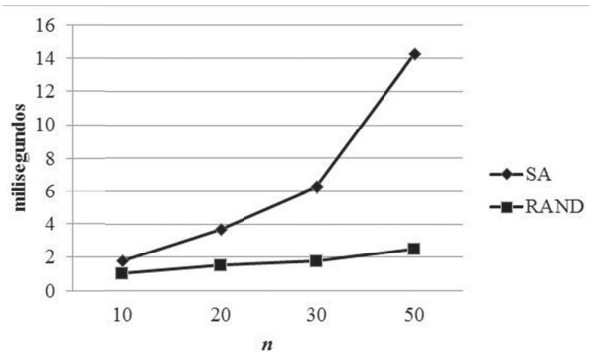


Figura4. Tiempo de ejecución promedio (milisegundos) de RAND y RS con perturbación por familias de productos (α =0,95 y c_0 = 50)

Conclusiones

El problema de reaprovisionamiento multiproducto ha sido estudiado por varias décadas; se han desarrollado varios procedimientos heurísticos para obtener la frecuencia de planeación de cada producto y el ciclo base, entre ellos se encuentra el procedimiento RAND que devuelve soluciones de muy buena calidad. De las metaheurísticas, se han implementado algoritmos genéticos y recocido simulado con resultados diversos ya que

la calidad de la solución se ve muy afectada a medida que se incrementa el tamaño el problema.

En este artículo se implementa el algoritmo recocido simulado donde, además de explorar las combinaciones de parámetros, se estudian dos estrategias de búsqueda: la perturbación de una variable a la vez y la agrupación por familias. Los resultados muestran que bajo las mismas combinaciones de parámetros, la estrategia de agrupación en familias de productos da mejores resultados ya que RS devuelve con mayor frecuencia la solución óptima y además es robusta. Hay interacción de los factores c_0 y α así como evidencia de curvatura en la región de experimentación. El esquema individual da resultados inferiores y no resulta robusto en comparación con lo que obtiene el algoritmo RAND.

Esta primera exploración muestra que la estrategia de búsqueda familias de productos da buenos resultados y da una pauta para realizar mejoras al algoritmo de RS realizando una nueva exploración empleando un diseño de superficie de respuesta.

Agradecimientos

El autor agradece a los árbitros sus valiosos comentarios para mejorar este trabajo.

Referencias

1. S. Goyal. "Determination of optimum packaging frequency for items jointly replenished". *Management Science.* Vol. 21. 1974. pp. 436-443.

2. E. Silver. "A simple method of determining order quantities in joint replenishments under deterministic demand". *Management Science*. Vol. 22. 1976. pp. 1351-1361.

3. M. Khouja, S. Goyal. "A review of the joint replenishment problem literature: 1989–2005". *European Journal of Operational Research*. Vol. 186. 2008. pp. 1- 16.

4. M. Kaspi, M. Rosenblatt. "An improvement of Silver's algorithm for the joint replenishment problem". *IIE Transactions.* Vol. 15. 1983. pp. 264-267.

5. A. Nilsson, A. Segersted, E. van der Sluis. "A new iterative heuristic to solve the joint replenishment problem using a spreadsheet technique". *International Journal of Production Economics.* Vol. 108. 2007. pp. 399 - 405.

6. A. Nisson, E. Silver. "A simple improvement on Silver's heuristic for the joint replenishment problem". *Journal of the Operational Research Society*. Vol. 59. 2008. pp. 1415-1421.

7. M. Khouja, Z. Michalewicz, S. Satoskar. "A comparison between genetic algorithms and the RAND method for solving the joint replenishment problem". *Production, planning and control.* Vol. 11. 2000. pp. 556-564.

8. A. Olsen. "An evolutionary algorithm to solve the joint replenishment problem using direct grouping". *Computers and Industrial Engineering*. Vol. 48. 2005. pp. 223-235.

9. M. Yoo, L. Lin. "A new solution method for the joint replenishment problem". *International Journal of Manufacturing Technology and Management.* Vol. 16. 2009. pp. 166 - 175.

10. H. Hoos, T. Stützle. *Stochastic local search: foundations and applications*. 1ª ed. Ed. Springer-Verlag. San Francisco, CA. 2005. pp. 61 - 74.

11. S. Kirkpatrick, C. Gellat, M. Vecci. "Optimization by simulated annealing". *Science*. Vol. 220. 1983. pp. 671-680.

12. R. Myers, D. Montgomery. *Response surface methodology: process and product optimization using designed experiments*. Ed. John Wiley & Sons. New Jersey. 2002. pp. 73 - 113.