

PROBLEMA DE ASIGNACION CUADRATICA MULTIOBJETIVO

FELIPE ORTEGA, A.
Dpto. Estadística e I.O.
Facultad de Ciencias Matemáticas
Universidad Complutense de Madrid

RESUMEN

Se define la versión multiobjetivo del Problema de Asignación Cuadrática. Se muestran los inconvenientes de la técnica de ponderación de objetivos y se desarrollan algoritmos locales bajo las metodologías de soluciones eficientes, lexicográficas y equilibradas mediante la generalización de los procedimientos r -óptimos al caso multidimensional. Se recogen resultados computacionales sobre los algoritmos propuestos.

Palabras Clave: Programación Multiobjetivo, Problema de Asignación Cuadrática, Algoritmos r -óptimos.

Clasificación AMS: 90C31-90C99.

SUMMARY

We define the Multiobjective Quadratic Assignment Problem. Because of the difficulties of the weighted objectives method we develop local algorithms which are based in the methodologies of efficient, lexicographic and balanced solutions. We generalize the r -optimum procedures to multidimensional problems and we show computational results of this algorithms.

Key words: Multiobjective Programming, Quadratic Assignment Problem, r -optimum procedures.

AMS Classification: 90C31-90C99.

Recibido, septiembre 1987.
Aceptado, noviembre 1988.

1. INTRODUCCION

El objeto de este trabajo es el estudio de la extensión multiobjetivo de un problema clásico en la Optimización Combinatoria: el Problema de Asignación Cuadrática (QAP).

El QAP es NP-completo (Sahni y González (1976)). Su resolución exacta cuando el número de facilidades es moderadamente alto (mayor que 15) requiere excesivo tiempo computacional como puede constatar-se en Burkard y Derigs (1980), que resuelven óptimamente problemas tests de 12 y 15 facilidades en 46,7 y 2947,3 segundos, respectivamente, en un ordenador CDC CYBER 76.

Entre las principales técnicas heurísticas aplicadas al QAP destacan los algoritmos r -óptimos utilizados aisladamente o en combinación con otros métodos (Nugent y otros (1968), Burkard y Derigs (1980), Burkard y Bönniger (1983), Bruijs (1984)).

En este trabajo se extienden los algoritmos r -óptimos al QAP multiobjetivo (MOQAP), dotados de las reglas necesarias para garantizar la convergencia. En la sección 2 se recogen los conceptos de Programación Multiobjetivo utilizados en secciones sucesivas. En la sección 3 se expone la formulación del QAP, se introduce la del MOQAP y se analiza el método de ponderación de objetivos en el MOQAP. En las secciones 4, 5 y 6 se desarrollan algoritmos para el MOQAP a partir de las metodologías de soluciones eficientes, lexicográficas y equilibradas, respectivamente. En la sección 7 se recogen resultados computacionales sobre estos algoritmos.

2. CONCEPTOS BASICOS DE PROGRAMACION MULTI OBJETIVO

Exponemos a continuación los conceptos de Programación Multiobjetivo que servirán de base para el estudio del MOQAP.

Definición 1

Llamamos problema multiobjetivo a:

$$\overrightarrow{\min}_{x \in X} z(x) = (z_1(x), \dots, z_m(x)) \quad (1)$$

donde :

X es el conjunto factible (decisiones admisibles)

z_1, \dots, z_m son los objetivos; $z_l: X \rightarrow \mathbb{R}, \forall l \in I_m = \{1, \dots, m\}$

$S \subset X$ es el conjunto cuyos elementos se consideran soluciones de (1) y depende de la metodología utilizada para resolverlo. ■

Entre los métodos usuales para tratar problemas multiobjetivo consideramos:

2.1. PONDERACION DE OBJETIVOS

Fijados unos pesos $w_l \geq 0 \forall l \in I_m$ se plantea el problema uniobjetivo

$$\min \phi(x) = \sum_l w_l z_l(x) \quad (2)$$

2.2. SOLUCIONES EFICIENTES

Utilizamos la notación habitual para comparar vectores: si $u = (u_1, \dots, u_m), v = (v_1, \dots, v_m)$, denotamos $u \leq v \Leftrightarrow u_i \leq v_i \forall i \in I_m$ y $\exists j$ tal que $u_j < v_j$.

La notación $u \not\leq v$ significa que $u = v$ o $\exists j$ tal que $u_j > v_j$.

Definición 2

Una solución factible $x \in X$ se llama eficiente o no dominada si $\forall x' \in X$ se verifica $z(x') \not\leq z(x)$. ■

Proposición 1

Toda solución óptima de (2) con pesos w_l estrictamente positivos es eficiente.

Demostración.—Inmediata. ■

2.3. SOLUCIONES PROPIAMENTE EFICIENTES

Definición 3

Una solución factible $x \in X$ se llama propiamente eficiente (Geoffrion (1968)) si es eficiente y $\exists T > 0$ tal que $\forall j \in I_m$ y $\forall x' \in X$ tales que $z_j(x') < z_j(x)$, $\exists i \in I_m$ tal que $z_i(x) < z_i(x')$ y $\frac{z_j(x) - z_j(x')}{z_i(x') - z_i(x)} \leq T$ ■

Proposición 2

Toda solución óptima de (2) con pesos w_i estrictamente positivos es propiamente eficiente.

Demostración.—Geoffrion (1968). ■

Si X es finito, entonces x es eficiente $\Leftrightarrow x$ es propiamente eficiente. Dado que en el MOQAP X es finito, modificamos este concepto fijando el valor de T .

Definición 4

Una solución factible $x \in X$ es propiamente eficiente de cota $T > 0$ si es eficiente y $\forall j \in I_m$ y $\forall x' \in X$ tales que $z_j(x') < z_j(x)$, $\exists i \in I_m$ tal que $z_i(x) < z_i(x')$ y $\frac{z_j(x) - z_j(x')}{z_i(x') - z_i(x)} \leq T$ ■

2.4. OPTIMIZACION LEXICOGRAFICA

Definición 5

Sea $z_1 \succcurlyeq, \dots, \succcurlyeq z_m$ un orden de preferencias establecido entre los objetivos y sea la sucesión de problemas:

$$(P_1) \min_{x \in X} z_1(x), \quad (P_2) \min_{x \in O_1} z_2(x), \dots, (P_m) \min_{x \in O_{m-1}} z_m(x)$$

donde O_l es el conjunto de soluciones óptimas de (P_l) , $\forall l \in I_m$.

Una solución factible $x \in X$ se llama óptima en sentido lexicográfico si $x \in O_m$. ■

2.5. SOLUCIONES EQUILIBRADAS

Definición 6

Sea z_{l-} (z_{l+}) cota inferior (superior) finita del objetivo l en X , $\forall l \in I_m$. Una solución factible $x \in X$ se llama equilibrada respecto a las cotas dadas, si resuelve el problema minimax

$$\min_{x \in X} \max_{l \in I_m} \frac{z_l(x) - z_{l-}}{z_{l+} - z_{l-}} \quad \blacksquare$$

Otra forma equivalente de definir una solución equilibrada, dados unos pesos w_l para los objetivos, es:

Definición 7

$x \in X$ es equilibrada respecto a los pesos $w_l > 0, \forall l \in I_m$ si resuelve el problema minimax $\min_{x \in X} \max_{l \in I_m} w_l z_l(x)$ ■

3. FORMULACION DE QAP Y MOQAP. PONDERACION DE OBJETIVOS EN EL MOQAP

El QAP, definido por primera vez por Koopmans y Beckmann (1957), consiste en asignar n facilidades a n lugares de forma que se minimice el coste total de transporte de cierto artículo entre las facilidades.

Definición 8

Si a_{ij} es el flujo que ha de transportarse de la facilidad i a la j , b_{pq} es el coste unitario de transporte del lugar p al q y x_{ip} la variable de decisión, que vale 1 si la facilidad i se asigna al lugar p y 0 en otro caso, la formulación algebraica del QAP es:

$$\min \sum_{i,j,p,q} a_{ij} b_{pq} x_{ip} x_{jq}$$

sujeto a

$$\begin{aligned} \sum_i x_{ip} &= 1, \quad \forall p \in I_n \\ \sum_p x_{ip} &= 1, \quad \forall i \in I_n \\ x_{ip} &\in \{0, 1\}, \quad \forall i, p \in I_n \end{aligned} \quad (3) \quad \blacksquare$$

Definición 9

Si ρ denota una permutación de los índices de I_n , la formulación combinatoria del QAP es:

$$\min_{\rho \in P_n} \sum_{i,j} a_{ij} b_{\rho(i)\rho(j)} \quad (4)$$

donde P_n es el grupo de permutaciones de los índices de I_n . ■

Se observa que las formulaciones (3) y (4) son equivalentes.

Dado que las soluciones factibles del QAP son permutaciones se habla preferentemente de permutaciones (eficientes, lexicográficas, ...) en lugar de soluciones (eficientes, lexicográficas, ...).

Definición 10

Se define el QAP general (QAPG) como

$$\min_{\rho \in P_n} \sum_{i,j} d_{ij\rho(i)\rho(j)} \quad (5)$$

■

Entre las aplicaciones prácticas del QAP se pueden citar:

- localización de plantas industriales (Koopmans y Beckmann (1957))
- localización de las componentes electrónicas de un circuito integrado (Steinberg (1961))
- localización de departamentos en un edificio (Armour y Buffa (1963), Pegels (1966)), en un hospital (Elshafei (1977)).

El planteamiento de la versión multiobjetivo del QAP es obligado en situaciones tales como:

- (1) el transporte de una unidad del artículo considerado se evalúa mediante un vector $(b_{pq}^1, \dots, b_{pq}^k)$ cuyas componentes son de diferente naturaleza.
- (2) existen h artículos a transportar con flujos y costes a_{ij}^l, b_{pq}^l , para $l = 1, \dots, h$
- (3) hay varios artículos a transportar y diferentes tipos de costes.

Definición 11

Se define el MOQAP como:

$$\overrightarrow{\min}_{\rho \in P_n} z(\rho) = (z_1(\rho), \dots, z_m(\rho))$$

donde:

$$z_l(\rho) = \sum_{i,j} a_{ij}^l b_{\rho(i)\rho(j)}^l, \quad \forall l \in I_m \quad (6)$$

y a_{ij}^l, b_{pq}^l son los flujos y costes unitarios asociados a los artículos y sus características. ■

Cuando la naturaleza de los objetivos lo permite, la asignación de pesos a los mismos transforma el MOQAP en el QAPG

$$\min_{\rho \in P_n} \phi(\rho) = \sum_{i,j} \sum_l w_l a_{ij}^l b_{\rho(i)\rho(j)}^l \quad (7)$$

Es obvio que este enfoque del problema no es aceptable en algunos casos. Por ejemplo, si cada artículo lleva asociadas características tales como coste económico, matices políticos, índice de contaminación, etc., las metodologías de soluciones eficientes, lexicográficas y equilibradas son más realistas.

En virtud de la proposición 1 toda permutación óptima de (7) con pesos estrictamente positivos es eficiente. Sin embargo, se observan las siguientes limitaciones:

- (1) es necesario resolver (7) exactamente. Las permutaciones casi-óptimas de (7) (esto es, aquellas en las que el valor de la función objetivo está próximo al valor óptimo de ésta) pueden no ser eficientes, pues aunque $\phi(\rho)$ varíe poco de una permutación casi-óptima a una óptima, los sumandos $w_l z_l(\rho)$ pueden ser notablemente diferentes.
- (2) la solución exacta de (7) sólo proporciona una permutación eficiente. Para obtener otras se requieren nuevos pesos w_l , lo que hace necesarias fórmulas de cambio de pesos. Puede suceder que dos conjuntos de pesos diferentes den la misma permutación eficiente.
- (3) El recíproco de la proposición 1 es falso. Para alguna permutación eficiente pueden no existir pesos $w_l > 0$ tales que ρ sea óptima en (7), como prueba el ejemplo 1, recogido en el Apéndice.

4. ALGORITMOS LOCALMENTE EFICIENTES

El problema de obtener todas las permutaciones eficientes de un MOQAP es NP-completo ya que puede suceder que las $n!$ factibles lo sean. Cuando n es moderadamente grande los métodos heurísticos se hacen imprescindibles. La generalización del concepto de permutación r -óptima en el caso uniobjetivo va a ser la base de los algoritmos que diseñaremos para el MOQAP.

4.1. ALGORITMOS R -OPTIMOS

Definición 12

Una permutación $\rho \in P_n$ se llama r -óptima si $\forall v \in N_r(\rho)$ se verifica $z(\rho) \leq z(v)$, donde $N_r(\rho) = \{v \in P_n / |\{i/v(i) \neq \rho(i)\}| = r\}$ es el r -entorno de ρ que consta de las permutaciones que difieren de ρ en exactamente r asignaciones. ■

Se verifica que $|N_r(\rho)| = \binom{n}{r} r! \sum_{k=2}^r \frac{(-1)^k}{k!}$.

Los r -entornos más utilizados en el QAP corresponden a $r = 2$ y $r = 3$, para los que $|N_2(\rho)| = \binom{n}{2}$, $|N_3(\rho)| = 2\binom{n}{3}$.

Para r pequeño los sucesivos cardinales aumentan en un factor $O(n)$. Por tanto, cabe esperar una mejor calidad, en media, de las permutaciones r -óptimas respecto de las s -óptimas si $s < r$ y r es pequeño frente a n .

Sin embargo, el incremento del coste computacional obliga a considerar pequeños valores de r . De ahí que la casi totalidad de autores que aplican algoritmos r -óptimos al QAP se limitan a $r = 2$ y $r = 3$.

La forma de uso más frecuente de los algoritmos r -óptimos consiste en, dada una permutación inicial ρ , buscar en su r -entorno una permutación mejor ρ' , sustituir ρ por ρ' (esto se llama r -intercambio) y repetir hasta que una permutación de la secuencia obtenida sea r -óptima. Este procedimiento se repite con un conjunto de permutaciones iniciales ρ_1, \dots, ρ_b , que pueden ser generadas aleatoriamente. Existen numerosas variantes: realizar el primer r -intercambio que mejora la permutación actual, el que más la mejora, sorteo entre los que proporcionan mejora (e incluso, se consideran los que la empeoran (Burkard y Rendl (1984)!), etcétera.

Proposición 3 (Fórmula del 2-intercambio)

Sean el QAP simétrico (A, B) simétricas con diagonales principales nulas), $\rho \in P_n$ y dos índices i_1, i_2 distintos de I_n . Sea $v \in N_2(\rho)$ tal que $v(i_1) = \rho(i_2)$, $v(i_2) = \rho(i_1)$, $v(i) = \rho(i) \forall i \neq i_1, i_2$. Entonces $z(v) = z(\rho) + 2 \sum_{j \neq i_1, i_2} (a_{i_1 j} - a_{i_2 j})(b_{\rho(i_2)\rho(j)} - b_{\rho(i_1)\rho(j)})$ ■

La evaluación de $z(v)$ a partir de $z(\rho)$ requiere $n - 1$ productos y $3n - 6$ sumas.

Proposición 4 (Fórmula del 3-intercambio)

Sean el QAP simétrico, $\rho \in P_n$ y tres índices i_1, i_2, i_3 distintos de I_n . Sea $v \in N_3(\rho)$ tal que $v(i_1) = \rho(i_2)$, $v(i_2) = \rho(i_3)$, $v(i_3) = \rho(i_1)$, $v(i) = \rho(i) \forall i \neq i_1, i_2, i_3$. Entonces

$$\begin{aligned} z(v) = z(\rho) &+ 2\{a_{i_1 i_2}(b_{\rho(i_2)} - b_{\rho(i_1)\rho(i_2)}) + a_{i_1 i_3}(b_{\rho(i_1)\rho(i_2)} - b_{\rho(i_1)\rho(i_3)} + \\ &+ a_{i_2 i_3}(b_{\rho(i_1)\rho(i_3)} - b_{\rho(i_2)\rho(i_3)}))\} + 2 \sum_{j \neq i_1, i_2, i_3} \{a_{i_1 j}(b_{\rho(i_2)\rho(j)} - b_{\rho(i_1)\rho(j)}) + \\ &+ a_{i_2 j}(b_{\rho(i_3)\rho(j)} - b_{\rho(i_2)\rho(j)}) + a_{i_3 j}(b_{\rho(i_1)\rho(j)} - b_{\rho(i_3)\rho(j)})\} \quad \blacksquare \end{aligned}$$

La evaluación de $z(v)$ a partir de $z(\rho)$ requiere $3n - 5$ productos y $6n - 12$ sumas.

La demostración de éstas y otras propiedades de los algoritmos puede verse en Felipe (1986), Apéndice B.

4.2. ALGORITMOS R -EFICIENTES PARA EL MOQAP

Son extensiones inmediatas de los algoritmos r -óptimos al MOQAP.

Definición 13

Una permutación $\rho \in P_n$ se llama r -eficiente si $\forall v \in N_r(\rho)$ se verifica $z(v) \not\leq z(\rho)$ ■

Algoritmo 1 (r -eficiente)

Paso 0: Elegir $\rho \in P_n$. Ir al paso 1.

Paso 1: Calcular $V(\rho) = \{v \in N_r(\rho) / z(v) \leq z(\rho)\}$. Ir al paso 2.

Paso 2: Si $V(\rho) = \emptyset$, ρ es r -eficiente: parar.

Si no, seleccionar $v \in V(\rho)$, poner $\rho = v$. Ir al paso 1. ■

Algoritmo 2 (r -eficiente, r -óptimo en l_o)

Paso 0: Seleccionar un objetivo $l_o \in I_m$. Elegir $\rho \in P_n$. Ir al paso 1.

Paso 1: Calcular $V(\rho) = \{v \in N_r(\rho) / z_{l_o}(v) < z_{l_o}(\rho) \text{ ó } [z_{l_o}(v) = z_{l_o}(\rho) \text{ y } z(v) \leq z(\rho)]\}$. Ir al paso 2.

Paso 2: Si $V(\rho) = \emptyset$, ρ es r -eficiente y r -óptima en l_o ; parar.

Si no, seleccionar $v \in V(\rho)$, poner $\rho = v$. Ir al paso 1. ■

Es inmediato comprobar que ambos algoritmos terminan en un número finito de iteraciones.

4.3. ALGORITMOS (R, S)-EFICIENTES

Sería deseable relajar la rigurosidad del algoritmo 1 permitiendo intercambios que mejoren la mayoría de los objetivos aunque empeoren alguno. Sin embargo, esta relajación puede provocar ciclos, incluso aunque la permutación inicial no se repita. Además puede suceder que una permutación de la sucesión generada esté dominada por otra anterior a ella.

Dada una permutación ρ denotamos por

$$\begin{aligned} O_{\rho}^{+}(v) &= |\{l \in I_m / z_l(v) < z_l(\rho)\}| \\ O_{\rho}^{-}(v) &= |\{l \in I_m / z_l(v) > z_l(\rho)\}| \end{aligned}$$

Es razonable permitir el cambio de ρ por v cuando $z(v) \leq z(\rho)$ ó $O_{\rho}^{+}(v) - O_{\rho}^{-}(v) \geq s$, donde $s \in I_m$, pero pueden ocurrir las anomalías anteriormente citadas. La proposición 5 da una regla para impedir estas circunstancias.

Proposición 5

Sean k objetivos fijos l_1, \dots, l_k , $k \leq s$. Si se acepta el cambio de ρ por v sólo si

$$z(v) \leq z(\rho) \tag{8a}$$

ó

$$O_{\rho}^{+}(v) - O_{\rho}^{-}(v) \geq s \text{ y } (z_{l_1}(v), \dots, z_{l_k}(v)) \leq (z_{l_1}(\rho), \dots, z_{l_k}(\rho)) \tag{8b}$$

entonces se evita el ciclo y no se pueden obtener permutaciones dominadas por otra previamente generada.

Demostración

En efecto, al hacer el cambio de ρ por v si $z(v) \leq z(\rho)$ las componentes de $(z_{l_1}(\rho), \dots, z_{l_k}(\rho))$ no aumentan y algún objetivo $z_l(\rho)$ disminuye estrictamente, y esto sólo puede ocurrir un número finito de veces consecutivas. Por otra parte, si se toma v verificando (8) el subvector $(z_{l_1}(\rho), \dots, z_{l_k}(\rho))$ disminuye al menos en una componente, lo cual puede ocurrir sólo un número finito de veces. Como consecuencia, ninguna permutación generada se repite y no puede obtenerse una permutación dominada por otra anterior. ■

Definición 14

Una permutación ρ se llama (r, s) -eficiente en l_1, \dots, l_k ($k \leq s$) si $\forall v \in N_r(\rho)$ se verifica $z(v) \not\leq z(\rho)$ y $[O_p^+(v) - O_p^-(v) < s = \text{ó } (z_{l_1}(v), \dots, z_{l_k}(v)) \not\leq (z_{l_1}(\rho), \dots, z_{l_k}(\rho))]$. ■

Algoritmo 3 $((r, s)$ -eficiente en l_1, \dots, l_k)

Paso 0: Seleccionar el subconjunto de objetivos $\{l_1, \dots, l_k\} \subset I_m$ ($k \leq s$).

Elegir $\rho \in P_n$. Ir al paso 1.

Paso 1: Calcular $V(\rho) = \{v \in N_r(\rho) / v \text{ verifica (8a) ó (8b)}\}$. Ir al paso 2.

Paso 2: Si $V(\rho) = \emptyset$, ρ es (r, s) -eficiente en l_1, \dots, l_k ; parar.

Si no, seleccionar $v \in V(\rho)$, poner $\rho = v$. Ir al paso 1 ■

La convergencia finita del algoritmo 3 se deriva de la proposición 5.

4.4. ALGORITMOS PROPIAMENTE R-EFICIENTES

Tratan de obtener soluciones propiamente eficientes locales. Permitiremos intercambios que no mejoren todos los objetivos por lo que adaptaremos la regla (8b) a esta situación para evitar las anomalías descritas.

Definición 15

Una permutación ρ se llama propiamente r -eficiente (p. r -ef) de cota $T > 0$ en l_1, \dots, l_k si es r -eficiente y $\forall v \in N_r(\rho)$ se verifica $\max_{j \in I_m} \{z_j(\rho) - z_j(v)\} \leq T \max_{i \in I_m} \{z_i(v) - z_i(\rho)\}$ ó $(z_{l_1}(v), \dots, z_{l_k}(v)) \not\leq (z_{l_1}(\rho), \dots, z_{l_k}(\rho))$. ■

Se observa que las condiciones

- (i) $\max_{j \in I_m} \{z_j(\rho) - z_j(v)\} > T \max_{i \in I_m} \{z_i(v) - z_i(\rho)\}$
- (ii) $\exists j \in I_m$ tal que $z_j(v) < z_j(\rho)$ y $z_j(\rho) - z_j(v) > T(z_i(v) - z_i(\rho)) \forall i \in I_m$ tal que $z_i(\rho) < z_i(v)$.

son equivalentes, de ahí la definición 15.

Algoritmo 4 (p. r -ef de cota T en l_1, \dots, l_k)

Paso 0: Fijar $T > 0$. Seleccionar $\{l_1, \dots, l_k\} \subset I_m$. Elegir $\rho \in P_n$. Ir al paso 1.

Paso 1: Calcular $V(\rho) = \{v \in N_r(\rho) / z(v) \leq z(\rho) \text{ ó } [\max_{j \in I_m} \{z_j(\rho) - z_j(v)\} > T \max_{i \in I_m} \{z_i(v) - z_i(\rho)\} \text{ y } (z_{l_1}(v), \dots, z_{l_k}(v)) \leq (z_{l_1}(\rho), \dots, z_{l_k}(\rho))]\}$. Ir al paso 2.

Paso 2: Si $V(\rho) = \emptyset$, ρ es p. r-ef de cota T en l_1, \dots, l_k ; parar.
Si no, seleccionar $v \in V(\rho)$, poner $\rho = v$. Ir al paso 1. ■

La convergencia del algoritmo 4 se deriva de la proposición 5.

Se observa que los algoritmos 1, 2, 3 y 4 no garantizan la obtención de permutaciones eficientes sino localmente eficientes, según la diferente definición de optimalidad local.

5. ALGORITMOS LOCALMENTE LEXICOGRÁFICOS

El empleo de métodos lexicográficos está justificado por el hecho de que el número de permutaciones óptimas y casi-óptimas en un objetivo concreto puede ser muy elevado y se pueden establecer claras diferencias entre ellas al considerar los otros objetivos. Por ejemplo, si en el objetivo 1 se verifica: $0 \leq L_a^l \leq a_{ij}^l \leq U_a^l$, $0 \leq L_b^l \leq b_{ij}^l \leq U_b^l$, $\forall i, j \in I_n$ y los a_{ij}^l , b_{ij}^l son enteros, los $n!$ valores de $z_l(\rho)$ se reparten en el intervalo de enteros $[n^2 L_a^l L_b^l, n^2 U_a^l U_b^l] \cap \mathbb{Z}$ y el número medio de empates es mayor o igual que $n! / (n^2 (U_a^l U_b^l - L_a^l L_b^l))$, notablemente grande, si bien cabe esperar mayor número de empates en valores centrales del intervalo que en los próximos a los extremos.

Dados los vectores $u = (u_1, \dots, u_m)$, $v = (v_1, \dots, v_m)$, denotamos

$$u \leq^l v \Leftrightarrow \exists l \in I_m \text{ tal que } u_i = v_i \ \forall i \in I_{l-1} \text{ y } u_l < v_l$$

$$u \not\leq^l v \Leftrightarrow u = v \text{ ó } \exists l \in I_m \text{ tal que } u_i = v_i \ \forall i \in I_{l-1} \text{ y } u_l > v_l$$

Supondremos un orden de preferencias $z_1 \succcurlyeq, \dots, \succcurlyeq z_m$ entre los objetivos.

Definición 16

Una permutación ρ se llama r -óptima en sentido lexicográfico (r -opt-lex) si $\forall v \in N_r(\rho)$ se verifica $z(v) \not\leq^l z(\rho)$. ■

Algoritmo 5 (r -óptimo-lexicográfico)

Paso 0: Ordenar $z_1 \succcurlyeq, \dots, \succcurlyeq z_m$. Elegir $\rho \in P_n$. Ir al paso 1.

Paso 1: Calcular $V(\rho) = \{v \in N_r(\rho) / z(v) \leq^l z(\rho)\}$. Ir al paso 2.

Paso 2: Si $V(\rho) = \emptyset$, ρ es r -opt-lex; parar.

Si no, seleccionar $v \in V(\rho)$, poner $\rho = v$. Ir al paso 1. ■

Es inmediato que el algoritmo 5 converge en un número finito de iteraciones, ya que la relación \leq^l impide la formación de ciclos.

Toda permutación r -opt-lex es r -eficiente. El recíproco obviamente no se cumple. Si llamamos MOQAP reducido al MOQAP obtenido al suprimir el objetivo de mayor prioridad z_1 , una permutación r -opt-lex del MOQAP completo puede no serlo en el reducido; además, éste puede tener permutaciones r -eficientes notablemente mejores en los objetivos z_2, \dots, z_m y sólo ligeramente peores en z_1 . Esta observación sugiere la elaboración de un algoritmo que relaje la rigurosidad lexicográfica y considere el concepto de eficiencia, lo cual haremos en el algoritmo 6 a partir de los siguientes elementos:

- (1) un orden de preferencias establecido $z_1 \succcurlyeq, \dots, \succcurlyeq z_m$;
- (2) un vector dado de niveles de significación $d = (d_2, \dots, d_m)$, con $d_l > 0$, cuyo fin es no aceptar cambios que mejoran el objetivo z_l en menos de d_l ;
- (3) un vector de niveles de tolerancia $e = (e_1, \dots, e_m)$, con $e_l > 0$ a determinar en el transcurso del algoritmo que garantiza la convergencia y permite cambios que mejoran los objetivos z_1, \dots, z_l o los empeoran en menos de e_1, \dots, e_l .

Algoritmo 6 (r -eficiente-lexicográfico)

Paso 0: Ordenar $z_1 \succcurlyeq, \dots, \succcurlyeq z_m$. Fijar $d = (d_2, \dots, d_m)$. Elegir $\rho_o \in P_n$. Ir al paso 1.

Paso 1: A partir de ρ_o obtener ρ_1 , r -óptima en z_1 . Si $\rho_o, \dots, \bar{\rho}, \rho_1$ han sido las permutaciones aceptadas por el algoritmo r -óptimo poner $y_1 = z_1(\rho_1)$, $e_1 = z_1(\bar{\rho}) - y_1$, $l = 1$. Ir al paso 2.

Paso 2: Poner $i = 1$, $\pi_1^l = \rho_l$. Ir al paso 3.

Paso 3: Calcular $V(\pi_i^l) = \{v \in N_r(\pi_i^l) / z_j(v) < y_j + e_j \quad \forall j \in I_l, z_{l+1}(v) \leq z_{l+1}(\pi_i^l) - d_{l+1}\}$. Ir al paso 4.

Paso 4: Si $V(\pi_i^l) = \phi$ poner $t_l = i$, $\rho_{l+1} = \pi_{t_l}^l$, $y_{l+1} = z_{l+1}(\rho_{l+1})$

$$e_{l+1} = \begin{cases} 0 & , \text{ si } t_l = 1 \\ z_{l+1}(\pi_{t_l-1}^l - y_{l+1}) & , \text{ si } t_l > 1 \end{cases}$$

$l = l + 1$. Ir al paso 5

Si no, seleccionar $v \in V(\pi_i^l)$, poner $\pi_{i+1}^l = v$, $i = i + 1$. Ir al paso 3.

Paso 5: Si $l < m$ ir al paso 2.

Si $l = m$ parar. ■

Proposición 6

El algoritmo 6 termina en un número finito de iteraciones.

Demostración

Para cada $l \in I_{m-1}$ la sucesión de permutaciones $\pi_1^l, \dots, \pi_{t_l}^l$ es finita y sus términos son todos diferentes porque $z_{l+1}(\pi_{i+1}^l) \leq z_{l+1}(\pi_i^l) - d_{l+1}$ y $d_{l+1} > 0$.

Por construcción de e_l y exigir $z_j(\pi_i^l) < z_j(\rho_j) + e_j \forall j \in I_l$ se cumple $\pi_i^l \neq \pi_{i'}^l \forall i \neq i'$ y $\forall l \neq l'$; por tanto, todas las permutaciones aceptadas son distintas y se llega a una iteración en la que $V(\pi_i) = \phi$ y $l = m$. La permutación final ρ_m verifica $z_l(\rho_m) < z_l(\rho_l) + e_l \forall l \in I_{m-1}$ y puede mejorarse, en general, aplicando el algoritmo 1 o el 2 con $l_0 = l$. ■

6. ALGORITMOS LOCALMENTE EQUILIBRADOS

Obtienen soluciones óptimas locales a partir del concepto de solución equilibrada.

Definición 17

Una permutación ρ se llama r -equilibrada (respecto a las cotas z_{l-} , z_{l+} en cada objetivo l) si $\forall v \in N_r(\rho)$ se verifica

$$\max_{l \in I_m} \frac{z_l(\rho) - z_{l-}}{z_{l+} - z_{l-}} \leq \max_{l \in I_m} \frac{z_l(v) - z_{l-}}{z_{l+} - z_{l-}} \quad \blacksquare$$

Algoritmo 7 (r -equilibrado)

Paso 0: Elegir $\rho \in P_n$. Ir al paso 1.

Paso 1: Calcular $V(\rho) = \left\{ v \in N_r(\rho) / \max_{l \in I_m} \frac{z_l(v) - z_{l-}}{z_{l+} - z_{l-}} < \max_{l \in I_m} \frac{z_l(\rho) - z_{l-}}{z_{l+} - z_{l-}} \right\}$. Ir al paso 2.

Paso 2: Si $V(\rho) = \phi$, ρ es r -equilibrada, parar.

Si no, seleccionar $v \in V(\rho)$, poner $\rho = v$. Ir al paso 1. ■

La convergencia del algoritmo 7 se deriva de la desigualdad estricta en la definición de $V(\rho)$.

7. COMENTARIOS GENERALES

La selección de la permutación $v \in V(\rho)$ en cada uno de los algoritmos 1 al 5 y 7 ($v \in V(\pi_i^t)$ en el 6) puede realizarse de alguna de las siguientes formas, sugeridas del caso uniobjetivo:

- 1) tomar $v \in V(\rho)$ aleatoriamente (no necesariamente con distribución uniforme)
- 2) tomar la primera permutación $v \in N_r(\rho)$ para la que se verifica $v \in V(\rho)$. Esta selección tiene la ventaja de que no es necesario calcular todo el conjunto $V(\rho)$ y es la que hemos considerado en la implementación de los algoritmos presentados.
- 3) asignar pesos w_l a los objetivos y tomar $v \in V(\rho)$ que minimice $\sum_l w_l(z_l(\rho) - z_l(v))$.

Selección de objetivos

Los algoritmos 2, 3 y 4, basados en el concepto de solución eficiente, el 5 y el 6, en el orden lexicográfico y el 7, en el de soluciones equilibradas favorecen la optimización particular de algunos objetivos. La elección de los parámetros l_o , s , k , l_1 , ..., l_k , T , z_{l+} , z_{l-} y el orden $z_1 \succ \dots \succ z_m$ es una cuestión ajena a la descripción de estos algoritmos y depende del problema real a resolver y de la información e interés del decisor sobre cada objetivo (por ejemplo, si el objetivo l_o es el más importante es aconsejable utilizar el algoritmo 2 con l_o , o colocar l_o como primer objetivo en algoritmos lexicográficos, etc.). Dada la flexibilidad de estos algoritmos puede ser útil en un caso concreto utilizar diversos valores de estos parámetros.

8. RESULTADOS COMPUTACIONALES

Implementamos los algoritmos 1 a 7, cada uno con $r = 2$ y $r = 3$ y los aplicamos a varios MOQAP variando el número de objetivos m y el de facilidades n .

La tabla 1 recoge los resultados obtenidos para un MOQAP simétrico con $m = 5$ y $n = 10$, cuyos flujos y costes fueron generados aleatoriamente, según una distribución uniforme discreta en los enteros 0, 1, ..., 10. El ordenador utilizado es un CYBER 180/810.

Generamos $n_i = 100$ permutaciones iniciales aleatorias comunes a cada uno de los 14 algoritmos. Para cada algoritmo calculamos la media, desviación típica, mínimo y máximo de cada objetivo en las permutaciones finales, número medio de intercambios examinados y tiempo computacional relativo.

Los parámetros $s, k, T, d = (d_2, \dots, d_m)$ y los objetivos l_0, l_1, \dots, l_k se generaron también aleatoriamente para cada permutación inicial. Las cotas z_{l-} y z_{l+} para el algoritmo 7 se obtienen mediante reducción de las matrices de flujos y costes de cada objetivo a partir de los autovalores de las matrices reducidas (Rendl (1985)).

Una información útil para comparar los algoritmos expuestos es lo que llamamos matriz de dominancias (Tabla 2) cuya celda (i, j) denota el número de veces que la permutación final dada por el algoritmo i domina a la dada por el j , obtenidas ambas a partir de la misma permutación inicial.

Análogamente a las tablas 1 y 2, las tablas 3 y 4 muestran los resultados para un MOQAP con $n = 20$, $m = 2$ y $n_i = 20$.

Entre otras consecuencias que se deducen de las tablas 1 a 4 destacamos:

- los algoritmos 1, 2 y 5 son los más rápidos;
- el algoritmo 5 favorece claramente el objetivo de mayor prioridad; el algoritmo 6 aminora la calidad en el objetivo 1 en favor de los demás;
- en problemas con muchos objetivos el algoritmo 7 es el que menos los discrimina, de ahí el nombre de «equilibrado». También es el que presenta menor varianza de los objetivos en las permutaciones finales. Es relativamente lento;
- en problemas con pocos objetivos se observa una clara ganancia de

Tabla 1

Resultados para un MOQAP con $n = 10$, $m = 5$, $n_i = 100$

Algoritmo	Medias * Desv. Típicas	Mínimos Máximos	N° r- inter- camb.	Tiempo comp. relat.
A.1 r=2	28,5 36,1 25,7 23,9 28,9 136 125 102 131 118	2198 1460 1750 2152 1978 2778 2034 2238 2814 2614	63	1
A.1 r=3	26,0 31,9 22,9 21,7 25,7 129 116 112 129 109	2192 1426 1728 2166 1906 2766 2034 2238 2744 2448	344	6,4
A.2 r=2	26,6 35,8 24,5 23,7 26,6 228 236 193 237 223	1954 1272 1596 2000 1736 2870 2144 2374 2966 2622	163	1,7
A.2 r=3	26,6 36,6 25,5 23,4 26,4 226 224 205 244 225	1954 1280 1606 2016 1738 2812 2082 2390 2904 2552	572	7,9
A.3 r=2	25,9 32,3 23,2 22,4 26,0 159 139 136 144 155	2064 1374 1682 2060 1868 2778 2038 2268 2800 2638	90	3,0
A.3 r=3	25,0 30,3 21,7 21,3 24,5 169 158 144 139 136	1958 1340 1600 2026 1844 2824 2034 2262 2732 2498	415	24,6
A.4 r=2	25,8 33,3 21,9 21,6 25,6 160 133 125 133 135	2092 1380 1600 2138 1812 2868 1974 2268 2748 2482	82	2,8
A.4 r=3	22,5 29,0 20,6 19,5 24,3 138 129 123 120 140	1972 1364 1668 2102 1756 2702 1936 2292 2630 2452	420	24,8
A.5 r=2	5,0 44,4 25,5 27,6 30,0 55 116 113 101 126	1950 1592 1744 2364 2044 2202 2178 2272 2876 2520	177	1,7
A.5 r=3	5,1 44,9 26,5 27,2 30,0 42 106 101 109 137	1950 1592 1774 2338 1952 2166 2120 2272 2876 2536	649	8,6
A.6 r=2	5,8 39,3 25,4 27,7 30,2 58 115 120 107 122	1950 1472 1744 2324 2044 2192 2098 2340 2856 2564	379	2,4
A.6 r=3	6,4 36,9 25,6 27,0 31,0 53 140 110 113 146	1958 1342 1744 2280 1840 2220 2120 2236 2876 2536	1800	20,2
A.7 r=2	24,6 30,4 21,1 20,3 24,4 75 73 54 70 59	2156 1446 1768 2160 1996 2586 1794 2034 2558 2334	116	4,3
A.7 r=3	23,5 28,0 19,1 18,5 23,4 63 51 49 72 46	2122 1416 1742 2152 1996 2528 1742 1984 2512 2238	519	40,3

* $100(\bar{z}_i - \min z_i)/\min z_i$; donde \bar{z}_i es la media de z_i en las n_i permutaciones finales y $\min z_i$ es el mejor valor obtenido para z_i entre las permutaciones finales calculadas.

Tabla 2

Matriz de dominancias para un MOQAP con $n = 10$, $m = 5$, $n_i = 100$

i \ j		Alg j r=2							Alg j r=3						
		1	2	3	4	5	6	7	1	2	3	4	5	6	7
Alg i r=2	1	-	0	0	2	0	0	1	4	0	2	0	0	0	0
	2	1	-	1	0	0	0	0	0	6	0	0	1	0	0
	3	7	0	-	1	0	0	0	4	0	2	1	0	1	0
	4	13	0	6	-	0	0	4	3	0	2	0	0	0	0
	5	1	0	1	0	-	0	0	0	1	0	0	5	3	0
	6	1	1	1	0	2	-	0	0	0	0	0	4	2	1
	7	20	0	7	3	0	0	-	2	0	1	1	0	0	5
Alg i r=3	1	35	0	9	10	0	0	4	-	0	0	2	0	0	2
	2	0	1	0	0	0	0	0	0	-	0	0	1	0	0
	3	12	0	10	5	0	0	2	1	0	-	1	0	0	1
	4	21	0	12	7	0	0	3	4	0	2	-	0	0	0
	5	1	0	0	1	2	1	0	0	0	0	0	-	0	0
	6	2	0	1	1	2	0	2	0	0	0	0	3	-	0
	7	26	0	9	8	0	0	21	6	0	4	0	0	0	-

calidad al pasar de $r = 2$ a $r = 3$ en cada algoritmo, mientras que cuando el número de objetivos aumenta esto sólo sucede en los algoritmos 1, 3, 4 y 7;

- la razón del número medio de 3-intercambios al de 2-intercambios es similar en todos los algoritmos para tamaño fijo de n . Se observa un crecimiento de esta razón en función de n .

Agradecimientos.—A los «referees» de este trabajo cuyas oportunas indicaciones lo han mejorado notablemente.

Apéndice

Soluciones eficientes no óptimas para ninguna ponderación de objetivos.

Tabla 3

Resultados para un MOQAP con $n = 20$, $m = 2$, $n_i = 20$

Algoritmo	Medias * Desv. Típicas	Mínimos Máximos	N° r- inter camb.	Tiempo comp. relat.
A.1 r=2	11,4 13,2 261 218	9018 7808 10156 8758	823	1,0
A.1 r=3	10,5 12,3 207 223	8934 7836 9762 8690	6611	14,8
A.2 r=2	15,6 16,7 867 933	8614 7370 10636 9872	1018	1,0
A.2 r=3	14,9 14,4 929 970	8508 7208 10860 9752	8443	15,8
A.3 r=2	11,4 13,2 261 218	9018 7808 10156 8758	823	1,9
A.3 r=3	10,5 12,3 207 223	8934 7836 9762 8690	6611	28,7
A.4 r=2	11,7 13,2 339 335	8766 7714 10068 9070	779	1,8
A.4 r=3	10,6 10,8 343 305	8470 7454 9780 8532	7315	28,0
A.5 r=2	3,6 29,3 125 279	8374 8846 8900 9872	1348	1
A.5 r=3	2,9 28,1 108 275	8388 8466 8826 9752	8835	15,7
A.6 r=2	4,3 26,2 148 376	8438 8172 9026 9832	1583	1,1
A.6 r=3	3,0 25,5 127 293	8388 8466 8828 9606	12792	25,6
A.7 r=2	5,7 22,0 129 150	8592 8482 9142 9090	1068	2,1
A.7 r=3	4,6 20,8 101 109	8516 8466 8920 8952	8287	15,2

* Igual significado que en la tabla 1.

Tabla 4

Matriz de dominancias para un MOQAP con $n = 20$, $m = 2$, $n_i = 20$

i \ j		Alg j r=2							Alg j r=3						
		1	2	3	4	5	6	7	1	2	3	4	5	6	7
Alg i r=2	1	-	0	0	6	0	0	0	2	0	2	1	0	0	0
	2	0	-	0	0	0	0	0	0	0	0	0	0	0	0
	3	0	0	-	6	0	0	0	2	0	2	1	0	0	0
	4	4	0	4	-	1	1	2	2	0	2	0	0	1	1
	5	0	0	0	0	-	2	1	0	0	0	0	3	4	0
	6	1	0	1	1	0	-	3	0	1	0	0	4	2	1
	7	0	1	0	0	1	2	-	0	1	0	0	3	3	5
Alg i r=3	1	8	0	8	5	0	0	1	-	0	0	0	2	0	0
	2	0	10	0	0	3	3	1	0	-	0	0	0	0	0
	3	8	0	8	5	0	0	1	0	0	-	0	2	0	0
	4	8	1	8	9	2	1	2	3	1	3	-	2	1	2
	5	0	3	0	0	6	6	2	0	0	0	0	-	2	1
	6	0	7	0	0	9	7	5	0	3	0	0	5	-	1
	7	0	5	0	1	6	8	13	0	1	0	0	2	2	-

Ejemplo 1

Sea el MOQAP con $m = 2$, $n = 3$, definido por las matrices:

$$A^1 = \begin{pmatrix} 0 & 5 & 4 \\ 5 & 0 & 2 \\ 4 & 2 & 0 \end{pmatrix} \quad B^1 = \begin{pmatrix} 0 & 4 & 5 \\ 4 & 0 & 6 \\ 5 & 6 & 0 \end{pmatrix}$$

$$A^2 = \begin{pmatrix} 0 & 7 & 6 \\ 7 & 0 & 1 \\ 6 & 1 & 0 \end{pmatrix} \quad B^2 = \begin{pmatrix} 0 & 5 & 3 \\ 5 & 0 & 2 \\ 3 & 2 & 0 \end{pmatrix}$$

Sean $\rho^1 = (1, 2, 3)$, $\rho^2 = (1, 3, 2)$, $\rho^3 = (2, 1, 3)$, $\rho^4 = (2, 3, 1)$, $\rho^5 = (3, 1, 2)$, $\rho^6 = (3, 2, 1)$.

Se verifica $z(\rho^1) = (104, 110)$, $z(\rho^2) = (106, 106)$, $z(\rho^3) = (108, 100)$, $z(\rho^4) = (112, 94)$, $z(\rho^5) = (114, 76)$, $z(\rho^6) = (116, 74)$.

Como $z_1(\rho^1) < \dots < z_1(\rho^6)$ y $z_2(\rho^1) > \dots > z_2(\rho^6)$ las seis permutaciones son eficientes.

Según los pesos w_1, w_2 las soluciones óptimas de (7) son

ρ^6 si $0 \leq w_1 < w_2$ ρ^1 si $0 \leq 3,4w_2 < w_1$ ρ^5 si $w^2 < w_1 < 3,4w_2$ ρ^5, ρ^6 si $w_1 = w_2$ ρ^1, ρ^5 si $w_1 = 3,4w_2$

esto es, hay 3 permutaciones eficientes ρ^2, ρ^3, ρ^4 (el 50 %) que no pueden obtenerse como solución óptima de (7).

La única solución equilibrada respecto a las cotas (104,116) en z_1 y (74,110) en z_2 es ρ^4 .

REFERENCIAS

- ARMOUR, G. C., y BUFFA, E. S. (1963): «A Heuristic Algorithm and Simulation Approach to the Relative Location of Facilities», *Man. Sci.*, 9, 294-309.
- BRUIJS, P. A. (1984): «On the Quality of Heuristic Solutions to a 19×19 Quadratic Assignment Problem», *E.J.O.R.*, 17, 21-30.
- BURKARD, R. E., y BÖNNIGER, T. (1983): «A Heuristic for Quadratic Boolean Programs with Applications to Quadratic Assignment Problems», *E.J.O.R.*, 13, 374-386.
- BURKARD, R. E., y DERIGS, U. (1980): «Assignment and Matching Problems: Solution Methods with FORTRAN Programs», *Lecture Notes in Economics and Mathematical Systems*, 184, Springer, Berlin.
- BURKARD, R. E., y RENDL, F. (1984): «A Thermodynamically Motivated Simulation Procedure for Combinatorial Optimization Problems», *E.J.O.R.*, 17, 169-174.
- ELSHAFEI, A. N. (1977): «Hospital Layout as a Quadratic Assignment Problem», *Op. Res. Quart.*, 28, 167-179.
- FELIPE, A. (1986): «Problema de Asignación Cuadrática. Extensiones», Tesis Doctoral, Universidad Complutense de Madrid.
- GEOFFRION, A. M. (1968): «Proper Efficiency and the Theory of Vector Maximization», *J. Math. Analysis and Applic.*, 22, 618-630.
- KOOPMANS, T. C., y BECKMANN, M. J. (1957): «Assignment Problems and The Location of Economic Activities», *Econometrica*, 25, 53-76.
- NUGENT, C. E.; VOLLMANN, T. E., y RUMML, J. (1968): «An Experimental Comparison of Techniques of the Assignment of Facilities to Locations», *Op. Res.*, 16, 150-173.
- PEGELS, C. (1966): «Plant Layout and Discrete Optimizing», *International J. of Prod. Res.*, 5, 81-92.

- RENDL, F. (1985): «Ranking Scalar Products to Improve Bounds for the Quadratic Assignment Problem», *E.J.O.R.*, 20, 363-372.
- SAHNI, S., y GONZALEZ, T. (1976): «P-complete Approximation Problem», *J.A.C.M.*, 23, 555-565.
- STEINBERG, L. (1961): «The Blackboard Wiring Problem: A Placement Algorithm», *SIAM Rev.*, 3, 37-50.