QÜESTIIÓ, Vol 13, nº 1,2,3 pp. 173-191, 1989

# ALGORITMOS HEURÍSTICOS DETERMINISTAS Y ALEATORIOS EN SECUENCIACIÓN DE PROYECTOS CON RECURSOS LIMITADOS

**RAMÓN ALVAREZ-VALDÉS OLAGUÍBEL**
**Y JOSÉ MANUEL TAMARIT GOERLICH**
Universitat de València

*En este trabajo se estudia la eficiencia relativa de un conjunto de algoritmos heurísticos, deterministas y aleatorizados, para el problema de la secuenciación de proyectos con limitación de recursos. Se presentan los resultados de un extenso estudio computacional y se aplican tests no paramétricos para contrastar estadísticamente las conclusiones obtenidas.*

**Deterministic and sampling heuristic algorithms for project scheduling with resource constraints**

**Keywords:** Project Scheduling, Heuristics, Non parametric tests

## 1. INTRODUCCIÓN

El problema considerado en este trabajo es el de secuenciación de un proyecto con limitación de recursos y sin posibilidades de interrupción del proceso de las actividades que lo componen una vez iniciado. También suponemos que las duraciones de las actividades, los requerimientos de recursos y la disponibilidad de cada tipo de recursos son cantidades enteras no negativas, conocidas y constantes a lo largo del tiempo en el que se realiza el proyecto. El objetivo es minimizar el tiempo total de ejecución.

Un proyecto puede ser definido como un conjunto de actividades $\{1, 2, \ldots, n\}$. Cada actividad, $i$, tiene una duración (tiempo necesario para su ejecución) $d_i$ y requiere para su proceso la cantidad $r_{ik}$ de cada recurso, $k = 1, 2, \ldots, K$,

-Departament d'Estadística i Investigació Operativa. Facultat de Matemàtiques. Universitat de València.
-Article rebut l'octubre de 1989.

siendo $b_k$ la cantidad total disponible de cada uno de ellos. Las relaciones de precedencia entre las actividades están descritas por un conjunto $H$ de arcos $(i, j)$, significando que la actividad $j$ no puede iniciarse hasta que se haya completado la actividad $i$. Se trata de determinar un conjunto de tiempos $t_1, t_2, \ldots, t_n$ en los que debe comenzar cada una de las actividades de forma que se respeten las relaciones de precedencia, que la cantidad de cada recurso requerida en cada momento no supere el máximo disponible y tal que el tiempo total de ejecución del proyecto sea mínimo.

El problema así definido puede formularse de la forma siguiente:

$$\text{Min } t_n$$

$$\text{sujeto a} \qquad t_j - t_i \geq d_i, \qquad (i,j) \in H$$

$$\sum_{S(t)} r_{ik} \leq b_k, \; t = 1, \ldots, T; \; k = 1, \ldots, K$$

donde $S(t)$ es el conjunto de actividades en proceso en el tiempo $t$.

Definimos el *grafo de precedencia* $G = (V, H)$ asociado con el problema, donde el conjunto de vértices $V$ es el conjunto de actividades del proyecto, incluyendo dos actividades ficticias, la actividad 1, que marca el inicio del proyecto y la actividad $n$, que marca el final del mismo. A estas dos actividades les asignamos duración y necesidades de recursos nulas. El conjunto de arcos $H$ tiene un arco $(i, j)$ si existe una relación de precedencia entre las actividades $i$ y $j$, así como un arco desde la actividad 1 a cualquier actividad sin predecesores y un arco desde cualquier actividad sin sucesores hasta la actividad $n$. El coste de cada arco es la duración de su vértice inicial. El grafo $G$ es dirigido y acíclico con costes no negativos y puede ser usado para calcular el camino más largo entre cualquier par de vértices.

## 2. ALGORITMOS HEURÍSTICOS

### 2.1 ESQUEMAS ALGORÍTMICOS

Kelley y Walker (1959) distinguen dos formas básicas de generar una secuencia posible,que denomina *en serie* y *en paralelo*. En ambos procedimientos, una vez una actividad ha sido introducida en la secuencia, nunca es resecuenciada.

La *secuenciación en serie* comienza numerando los vértices de forma que para cada arco su vértice inicial tiene un número menor que su vértice final. Este esquema de numeración tiene la propiedad de que si se secuencian las actividades en el orden indicado por su número, entonces ninguna actividad

aparecerá antes que ninguna de sus predecesoras. Se puede construir una secuencia posible considerando las actividades en este orden y secuenciando cada una de ellas tan pronto como las relaciones de precedencia y las restricciones sobre los recursos lo permitan. El procedimiento de numeración no es, en general, único. En algunos momentos puede existir un conjunto de actividades de forma que las predecesoras de todas ellas ya estén numeradas y todas pueden recibir el siguiente número. El orden entre las mismas puede obtenerse de forma aleatoria o mediante una función de prioridad tal como el requerimiento de recursos, duración de las actividades, etc.. Cada regla de prioridad define un algoritmo en serie diferente.

En la *secuenciación en paralelo* se construye una secuencia posible procediendo hacia delante en el tiempo. En cada momento durante la construcción, se determina el conjunto de actividades que pueden ser secuenciadas de acuerdo con las restricciones de precedencia y de recursos. Este conjunto se ordena mediante una regla de prioridad y las actividades se secuencian en ese orden mientras no se rebase la capacidad de recursos, y así sucesivamente. Cada regla de prioridad para ordenar las actividades que pueden ser secuenciadas define un algoritmo heurístico distinto. Si las actividades reciben la prioridad independientemente de la secuencia ya existente, el algoritmo se denomina *estático*. En caso contrario se denomina *dinámico*. Algunas reglas, por ejemplo aquellas basadas en las características de las actividades, son claramente estáticas. Algunas otras, como las basadas en medidas sobre el camino crítico, pueden ser estáticas si estas medidas se efectúan una sola vez y son mantenidas durante la construcción de la secuencia, o dinámicas, si se actualizan teniendo en cuenta la secuencia parcial ya obtenida.

La mayor parte de los algoritmos propuestos en la literatura y usados en la práctica son algoritmos en paralelo ya que ofrecen mejores resultados que los algoritmos en serie. Todos ellos siguen el siguiente esquema:

Etapa 1. *Inicialización.*

Hacer SP (*secuencia parcial*) = CAN (*conjunto de candidatos,* conjunto de actividades para las cuales ha finalizado la ejecución de todos sus predecesores) = POS (*conjunto posible,* conjunto de actividades que pueden ser secuenciadas) = ACT (conjunto de actividades en proceso) = $\emptyset$.

Hacer $t = 0$; $RU_k$ (recursos utilizados) = 0, para todo $k$.

Etapa 2. *Construcción de* CAN.

Hacer CAN = $\{j / t_i + d_i \leq t, \forall (i,j) \in H\}$.

Si CAN = $\emptyset$, ir a la etapa 7, en otro caso:

Etapa 3. *Construcción de* POS.

Hacer POS = $\{j \in \text{CAN}/r_{jk} \leq b_k - \text{RU}_k,\ k = 1, \ldots, K\}$

Si POS = $\emptyset$, ir a la etapa 6. En otro caso:

Etapa 4. *Regla de prioridad.*

Ordenar POS de acuerdo con la regla de prioridad elegida.

Etapa 5. *Construcción de la secuencia.*

5.1. Tomar el siguiente $j \in$ POS. Si $r_{jk} \leq \text{RU}_k$, ir a 5.2. En otro caso ir a la etapa 6.

5.2. Hacer ACT = ACT $\cup\{j\}$; $t_j = t$; SP = SP $\cup\{t_j\}$; $\text{RU}_k = \text{RU}_k + r_{jk}\ \forall k$.

Ir a la etapa 5.1.

Etapa 6. *Determinación de un nuevo tiempo t.*

Encontrar un $j$ tal que $t_j + d_j = \min\{t_i + d_i, i \in \text{ACT}\}$.

Hacer $t = t_j + d_j$ ; ACT = ACT - $\{j/t_j + d_j = t\}$ ; $\text{RU}_k = \text{RU}_k - r_{jk}$, $\forall\, j/t_j + d_j = t, \forall k$.

Ir a la etapa 2.

Etapa 7. *FIN.*

Todas las actividades han sido secuenciadas. Parar.

Una forma completamente diferente de obtener una secuencia posible es el *método de muestreo*. Los métodos de muestreo forman *un conjunto de secuencias posibles* usando técnicas de aleatorización y eligen la mejor secuencia obtenida. La construcción de cada una de las secuncias posibles sigue el mismo esquema que la secuenciación en paralelo, pero en la etapa 4 la ordenación se basa en una selección aleatorizada, usando la prioridad que se asigna a cada actividad como un factor de peso; la probabilidad de que la actividad $j$ sea la elegida para ser introducida en la secuencia viene dada por su valor de prioridad dividido por la suma de los valores de prioridad del conjunto POS. Para cada regla de prioridad utilizada en la aleatorización obtenemos un algoritmo de muestreo diferente.

## 2.2 REGLAS HEURÍSTICAS DE PRIORIDAD

A lo largo de los años se han propuesto una gran cantidad de reglas para la construcción de soluciones heurísticas de los problemas de secuenciación. Algunas de ellas se diseñaron para los problemas de secuenciación en máquinas,

en particular para el problema del job shop, y se adaptaron al problema que nos ocupa. Otras se basan en los métodos PERT/CPM y fueron diseñadas específicamente para el problema de secuenciación de un proyecto con restricciones sobre los recursos. Estas reglas han sido descritas, clasificadas y comparadas en trabajos como los de Davis y Patterson (1975), Cooper (1976) y Lawrence (1984). En un estudio anterior, Alvarez-Valdés y Tamarit (1989), acerca de la eficiencia relativa de los principales algoritmos concidos, hemos obtenido una clasificación en la que en los primeros lugares aparecían un grupo, sustancialmente homógeneo, de seis reglas que pueden considerarse la más eficientes. Estas reglas tienen en común el usar de forma correcta información global acerca del proyecto y difieren en el tipo de información utlizado acerca de la estructura del grafo y de los recursos. Los algoritmos definidos por estas seis reglas heurísticas, que se describen a continuación, y sus versiones aleatorizadas constituyen el objeto del presente trabajo, en el que se estudian en detalle cada una de ellas y se las compara en términos de su eficiencia relativa respecto al óptimo.

**1.MTS (Most Total Succesors).**

Este heurístico elige primero aquella actividad con mayor número de sucesores, inmediatos o no, ya que el retraso de dicha actividad los retrasa a todos ellos.

Si $S^*(j) = \{i/\text{existe un camino de } j \text{ a } i \text{ en } G\}$ tenemos:

$\text{Max}|S^*(j)|$; $j \in \text{POS} = \{\text{actividades que pueden comenzar su proceso}\}$

**2.GRPW (Greatest Rank Positional Weight).**

Esta regla selecciona las actividades de acuerdo con su *peso posicional*, obtenido sumando a la duración de la actividad la duración de todos sus sucesores:

$$\text{Max}\{d_j + \sum_{i \in S^*(j)} d_j\};\ j \in \text{POS}.$$

**3.LST (Latest Start Time)**

Esta regla secuencia en primer lugar aquella actividad que tiene un menor tiempo máximo de inicio (LST), obtenido mediante el método del camino crítico. El LST da una medida de la urgencia de empezar la actividad, ya que si ésta se secuencia en un tiempo posterior a su LST se producirá un retraso en el tiempo total de ejecución del proyecto respecto a la solución dada por el método del camino más largo. La máxima prioridad se dará a la actividad que produce más retraso:

$$\text{Min } \text{LST}_j;\ j \in \text{POS}.$$

**4.LFT (Late Finish Time)**

Es similar a la anterior, pero tiene en cuenta la duración de la actividad:

$$\text{Min } \text{LFT}_j = \text{Min}(\text{LST}_j + d_j);\ j \in \text{POS}.$$

**5.RSM (Resource Scheduling Method)**

Esta regla la desarrollaron Brand, Meyer y Shaffer (1964) y se encontró bastante efectiva en una serie de pruebas dirigidas a proyectos de la industria de la construcción.

La prioridad de la actividad se calcula de la forma siguiente: dar precedencia a la actividad con menor $d_{ij}$, donde $d_{ij}$ es el incremento en la duración del proyecto cuando se realiza la actividad $j$ después que la actividad $i$, esto es, $d_{ij} = \text{Max}(0, \text{EFT}_i - \text{LST}_j)$ y la comparación se realiza entre todos los pares de actividades que pueden ser secuenciadas en ese momento.

**6.CUMRED (Cumulative Resource Equivalent Duration)**

La duración equivalente por recursos de la actividad $j$, $\text{RED}_j$, se define como

$$\text{RED}_j = \sum_{k=1}^{K} \frac{r_{jk}}{b_k} \frac{\text{RUD}_k}{\text{RUDMAX}} d_j$$

donde $\text{RUD}_k$, la duración por el uso de recursos correspondiente al recurso $k$, es una estimación del tiempo mínimo en el que se puede completar el proyecto considerando únicamente la disponibilidad y requerimientos del recurso $k$. Esta estimación se calcula de la forma siguiente: sea $E_k$ la cantidad total en la que los requerimientos de recursos superan la disponibilidad de los mismos cuando todas las actividades están colocadas en sus tiempos mínimos de inicio (EST), obtenidos mediante el método del camino crítico, y sea $L_k$ el último periodo de tiempo en el que, con las actividades así colocadas, hay demanda del recurso $k$. Entonces $\text{RUD}_k = L_k + E_k/b_k$. RUDMAX es el valor máximo de $\text{RUD}_k$ sobre el conujunto de recursos.

CUMRED suma las duraciones equivalentes por recursos de una actividad y la de todos sus sucesores, dando prioridad a la actividad con mayor suma:

$$\text{Max } \text{CUMRED}_j = \text{RED}_j + \sum_{i \in S^*(j)} \text{RED}_i;\ j \in \text{POS}$$

## 3. DISEÑO EXPERIMENTAL

Para comparar la eficiencia de los algoritmos descritos y sus versiones aleatorizadas (denotadas por la terminación -AL), hemos diseñado un experimento generando un conjunto de proyectos con diferentes características. Siguiendo básicamente el esquema de Cooper (1976), cada proyecto es descrito mediante parámetros relativos al tamaño y la estructura del grafo del proyecto y las propiedades relativas al tiempo y a los recursos.

*El Grafo del Proyecto*

El tamaño del proyecto es un parámetro al que denotaremos NACT. Hemos usado proyectos de 103 actividades, para extender los resultados de trabajos anteriores a problemas de tamaño más realista. En todos los casos, las duraciones de las actividades se asignaron aleatoriamente en el intervalo [1,10].

Para caracterizar la estructura del grafo del proyecto, Cooper usa el *nivel de ordenación, OS* definido como OS = T/U, donde T es el número total de relaciones de precedencia en el proyecto y U = N(N - 1)/2 es el número máximo de relaciones de precedencia que pueden existir entre las N actividades del proyecto. Davis y Patterson (1975) proponen un parámetro que indica el *número medio de arcos por actividad, NAR*, que parece más intuitivo y fácil de usar. Nosotros nos hemos inclinado por usar el parámetro NAR con valores 1,2 y 3.

*Tiempo*

El parámetro de interés relativo al tiempo se refiere a la holgura libre de las actividades. La *holgura libre*, $ff_i$, de la actividad $i$ es la holgura asociada con $i$ cuando todas las actividades empiezan lo antes posible, y es una medida de la posibilidad de desplazar una actividad en el tiempo sin afectar a ninguna otra. La holgura libre del proyecto se mide mediante un parámetro denominado *densidad* $D = \sum d_i / \sum (d_i + ff_i)$. Obviamente, $0 < D \leq 1$, donde $D = 1$ refleja la situación en la que todas las actividades son críticas. Hemos usado valores de $D$ de 0.50 y 0.75.

*Recursos*

El número $K$ de tipos de recursos en cada proyecto es 6. El número de tipos de recursos utilizado por cada actividad se mide mediante el parámetro *factor de utilización del recurso, RF*, definido como el cociente del número medio de los diferentes tipos de recursos utilizados por cada actividad y el número total de tipos de recursos $K$. Hemos usado valores de RF de 0.5 y 1.

La cantidad de cada recurso $k$ requerido por las actividades del proyecto, en relación a la cantidad disponible $b_k$ se mide mediante el parámetro *nivel de requerimiento del recuros, RS*, definido como $\mathrm{RS} = b_k N / \sum r_{ik}$. Se usaron los valores 2,3,4,5. Los valores de $r_{ik}$ se asignaron aleatoriamente en el intervalo [1,10], mientras que los de $b_k$ se obtuvieron de la ecuación anterior.

*Número de proyectos*

Se generó un proyecto para cada combinación de valores de los parámetros, lo que supuso un total de 48 proyectos (1 NACT por 3 NAR por 2 D por 2 RF por 4 RS).

## 4. RESULTADOS COMPUTACIONALES

Los 48 problemas generados han sido resueltos con los seis pares de algoritmos heurísticos, deterministas y aleatorizados, definidos por cada regla de prioridad. Con cada uno de los algoritmos aleatorizados se obtuvieron 100 soluciones para cada problema, tomándose como resultado en cada caso la mejor solución obtenida. Asimismo, se intentaron resolver mediante uno de los algorimos exactos de los que disponemos: Christofides et al. (1987) y Stinson et al. (1978). Sólo algunos de ellos se resolvieron óptimamente. Para el resto de los problemas, bien el tiempo, bien la memoria disponible, impidieron obtener la solución óptima. En estos casos, en lugar de usar el óptimo como cota superior se usó la mejor solución obtenida. En alguna ocasión esta solución fue mejorada por algún algoritmo heurístico y entonces la nueva solución fue la empleada en los resultados de las tablas. Por ello, los resultados obtenidos no nos permiten ser categóricos en cuanto a la eficiencia *absoluta*(es decir, en relación con las soluciones óptimas) de los algoritmos, pero sí pueden utilizarse para estudiar la eficiencia *relativa* de unos algoritmos respecto de los otros.

Los resultados obtenidos para los dos tipos de algoritmos aparecen en las Tablas 4.1 y 4.2 de distancias medias al óptimo (o mejor solución conocida) en tanto por ciento, desglosados por tipos de proyectos. La comparación global de las dos versiones de cada algoritmo, determinista y aleatorizada, aparece en la Tabla 4.3, en la que junto a las distancias medias respecto al óptimo ($\overline{X}$) aparecen las desviaciones típicas de dichas distancias ($S_x$) y la distancia máxima obtenida por cada algoritmo en los 48 problemas (DMAX).

La comparación numérica puede completarse de forma gráfica. La calidad relativa de cada par de algotimos para cada uno de los 48 problemas no sigue una pauta uniforme, sino que presenta grandes variaciones. Como ejemplo, las figuras 4.1(a) y (b) muestran la solución determinista (marcada por la flecha vertical) en relación con las soluciones obtenidas por el algoritmo aleatorizado (agru-

padas en forma de histograma) en dos de los problemas test. Mientras que en el primer caso el algoritmo aleatorizado apenas mejora la solución obtenida por el determinista, en el segundo la mejora es sustancial. La comparación de las 48 soluciones obtenidas por las parejas de algoritmos CUMRED/CUMREDAL y LFT/LFTAL aparece en las figuras 4.2 y 4.3, respectivamente. En ellas se aprecia que la versión aleatorizada no garantiza una solución mejor que la determinista en todos los casos, aunque sí en la gran mayoría y en ocasiones con mejoras sustanciales. Una última comparación gráfica global de las parejas de algoritmos, mediante la comparación de los respectivos histogramas se refleja en las figuras 4.4/4.5 para el par CUMRED/CUMREDAL y en las figuras 4.6/4.7 para el par LFT/LFTAL. En ambos casos parece existir una mejora global en la calidad de las soluciones obtenidas mediante los algoritmos aleatorizados y lo mismo ocurre para las restantes parejas de algoritmos.

TABLA 4.1

Distancias medias al óptimo de los algoritmos deterministas.

| | | Num. problem | CUMRED | GRPW | LFT | LST | MTS | RSM |
|---|---|---|---|---|---|---|---|---|
| GLOBAL | | 48 | 3.78 | 3.23 | 3.45 | 3.98 | 3.52 | 4.22 |
| NAR | 1 | 16 | 6.08 | 5.18 | 4.80 | 5.27 | 5.32 | 6.04 |
| | 2 | 16 | 3.72 | 3.34 | 3.21 | 4.14 | 3.21 | 3.66 |
| | 3 | 16 | 1.54 | 1.15 | 2.33 | 2.52 | 2.04 | 2.96 |
| DENS | .75 | 24 | 4.36 | 3.39 | 4.12 | 4.53 | 4.04 | 5.23 |
| | .50 | 24 | 3.21 | 3.06 | 2.78 | 3.42 | 3.01 | 3.22 |
| RF | 1.0 | 24 | 2.19 | 1.95 | 2.65 | 2.54 | 2.86 | 3.32 |
| | 0.5 | 24 | 5.38 | 4.50 | 4.25 | 5.42 | 4.19 | 5.12 |
| RS | 2 | 12 | 4.10 | 3.22 | 3.11 | 4.39 | 2.34 | 3.92 |
| | 3 | 12 | 5.16 | 4.66 | 4.83 | 5.54 | 5.55 | 5.62 |
| | 4 | 12 | 3.64 | 3.18 | 3.94 | 3.73 | 4.06 | 4.26 |
| | 5 | 12 | 2.22 | 1.84 | 1.90 | 2.24 | 2.14 | 3.08 |

TABLA 4.2

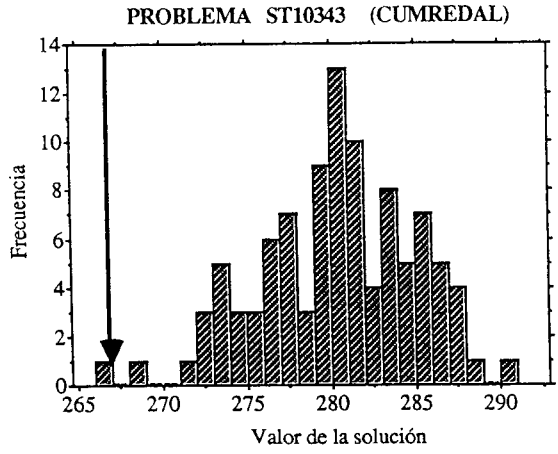
Distancias medias al óptimo de los algoritmos aleatorizados.

| | | Num. problem | CUMREDAL | GRPWAL | LFTAL | LSTAL | MTSAL | RSMAL |
|---|---|---|---|---|---|---|---|---|
| GLOBAL | | 48 | 2.13 | 2.31 | 1.65 | 2.38 | 1.97 | 2.97 |
| NAR | 1 | 16 | 3.66 | 4.17 | 2.35 | 4.17 | 2.77 | 6.38 |
| | 2 | 16 | 1.88 | 1.86 | 1.66 | 1.96 | 2.10 | 1.61 |
| | 3 | 16 | 0.86 | 0.91 | 0.94 | 1.01 | 1.02 | 0.93 |
| DENS | .75 | 24 | 2.42 | 2.23 | 1.88 | 2.74 | 2.36 | 3.43 |
| | .50 | 24 | 1.84 | 2.39 | 1.42 | 2.03 | 1.57 | 2.52 |
| RF | 1.0 | 24 | 1.21 | 1.50 | 1.18 | 1.56 | 1.31 | 1.73 |
| | 0.5 | 24 | 3.06 | 3.12 | 2.12 | 3.21 | 2.62 | 4.22 |
| RS | 2 | 12 | 1.66 | 1.93 | 1.04 | 1.96 | 1.28 | 2.91 |
| | 3 | 12 | 3.27 | 3.58 | 2.26 | 3.70 | 2.78 | 4.09 |
| | 4 | 12 | 2.28 | 2.35 | 1.61 | 2.25 | 1.88 | 3.01 |
| | 5 | 12 | 1.37 | 1.39 | 1.69 | 1.62 | 1.93 | 1.89 |

TABLA 4.3

Comparación de algoritmos deterministas y aleatorizados.

| ALGORITMO | $\bar{X}$ | $S_{\bar{X}}$ | DMAX |
|---|---|---|---|
| CUMRED | 3.79 | 3.42 | 18.31 |
| CUMREDAL | 2.13 | 2.27 | 11.05 |
| GRPW | 3.23 | 3.22 | 16.61 |
| GRPWAL | 2.31 | 2.46 | 12.15 |
| LFT | 3.45 | 2.87 | 12.71 |
| LFTAL | 1.65 | 1.55 | 7.25 |
| LST | 3.98 | 2.95 | 13.56 |
| LSTAL | 2.38 | 2.29 | 10.17 |
| MTS | 3.53 | 3.02 | 14.92 |
| MTSAL | 1.97 | 1.87 | 10.14 |
| RSM | 4.23 | 2.89 | 13.26 |
| RSMAL | 2.97 | 3.66 | 14.36 |



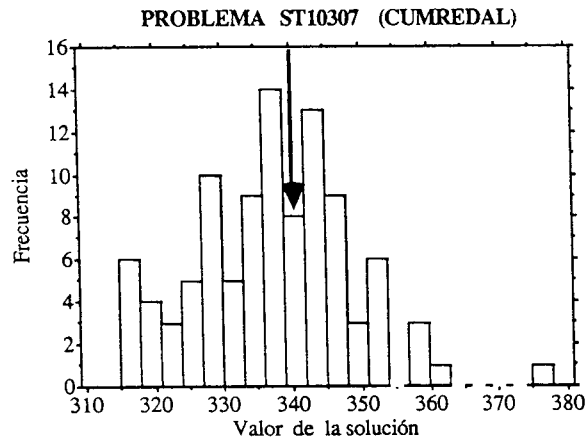




FIGURA 4.1 (a) y (b)

Soluciones deterministas y aleatorizadas para dos problemas test.

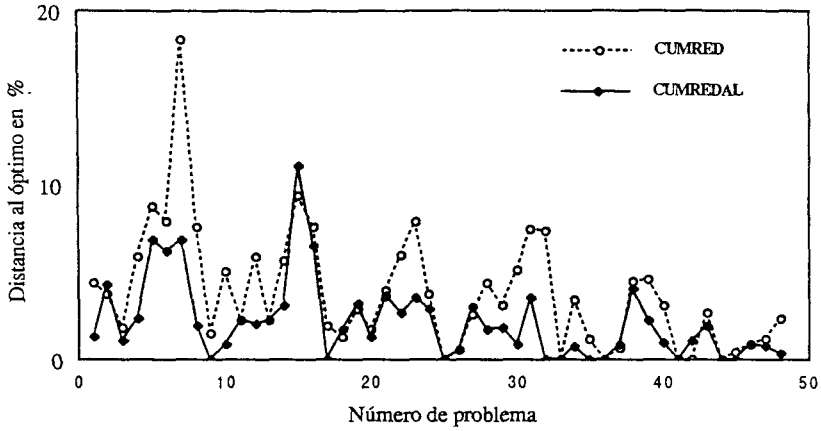



FIGURA 4.2

Soluciones de los algoritmos CUMRED y CUMREDAL en los 48 problemas test.

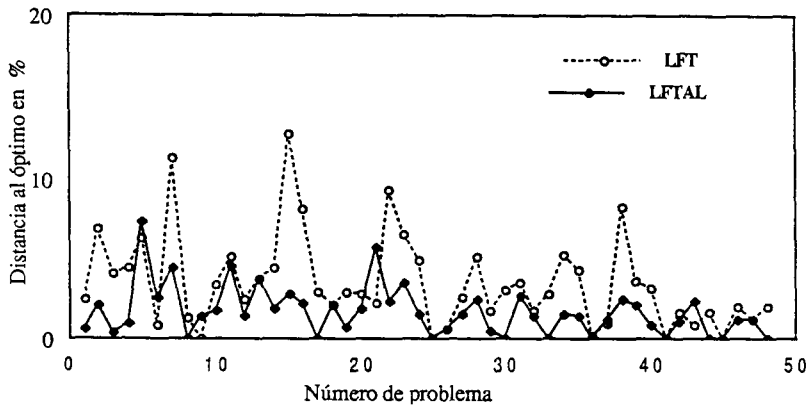



FIGURA 4.3

Soluciones de los algoritmos LFT y LFTAL en los 48 problemas test.

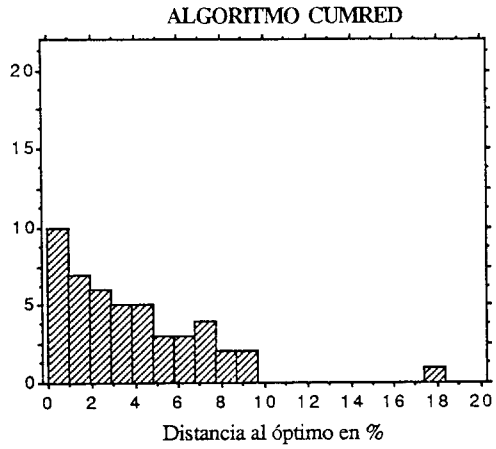



FIGURA 4.4

Resultados obtenidos por el algoritmo CUMRED.

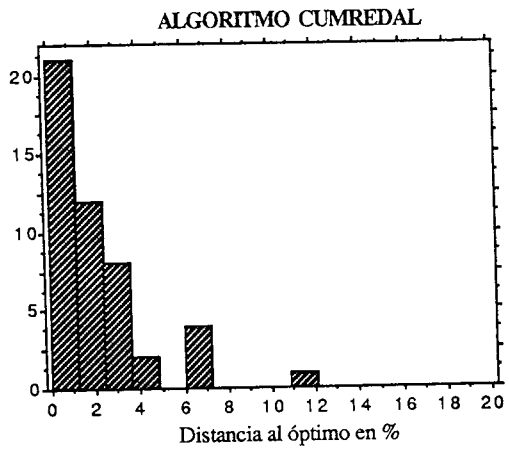



FIGURA 4.5

Resultados obtenidos por el algoritmo CUMREDAL.

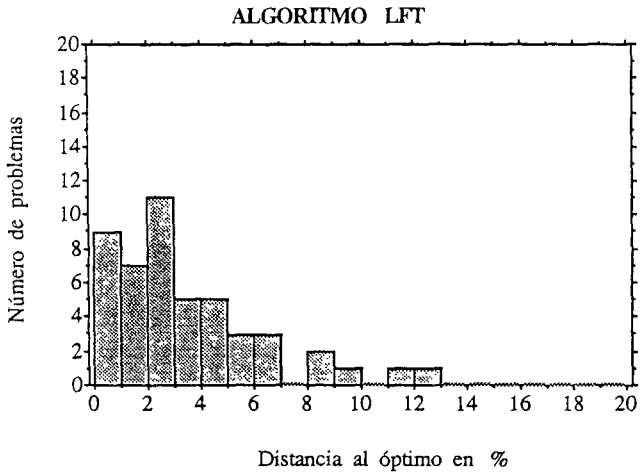



FIGURA 4.6

Resultados obtenidos por el algoritmo LFT.

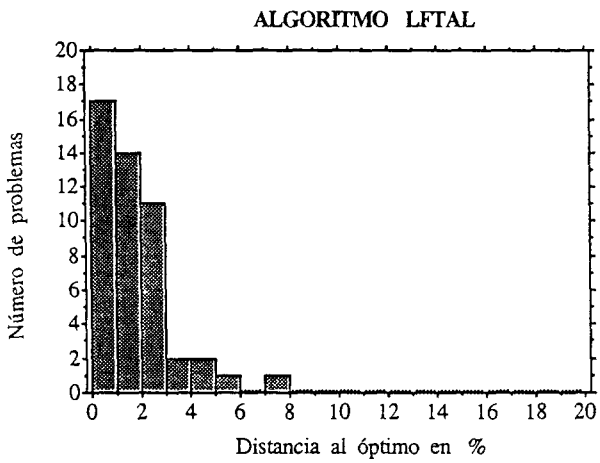



FIGURA 4.7

Resultados obtenidos por el algoritmo LFTAL.

## 5. ANÁLISIS ESTADÍSTICO

Para poder obtener conclusiones a partir de estos datos es necesario realizar el análisis estadístico adecuado. De las figuras 4.4 - 4.7, referidas a la variable de interés, distancia al óptimo (o mejor solución conocida), se tiene la impresión de que los datos no siguen una distribución normal, y lo mismo ocurre con los algoritmos no representados.

Esta impresión puede contrastarse estadísticamente mediante tests de normalidad. La Tabla 5.1 muestra los resultados del test de Kolmogorov-Smirnov y del test de asimetría para los doce algoritmos estudiados. Aunque el test de Kolmogorov-Smirnov no rechaza la hipótesis de normalidad, quizá debido al tamaño muestral (48), el test de asimetría de la distribución lleva a rechazar la normalidad en todos los casos. Los valores críticos son 0.558 al nivel de significación 0.05 y 0.825 al nivel 0.01. Todos los algoritmos estudiados superan el primero de ellos y, salvo LST y RSM, los restantes superan ampliamente el segundo, lo que confirma la impresión visual obtenida. Por ello, la comparación entre algoritmos se ha llevado a cabo mediante tests no paramétricos, lo que permite establecer conclusiones más generales no ligadas a la normalidad de la variable.

El test de Friedman para los seis algoritmos deterministas da una significatividad de 0.0123. Existen diferencias entre ellos, pero la hipótesis de igualdad no puede rechazarse completamente. La comparación por parejas mediante el test de Wilcoxon aparece en la Tabla 5.2 y muestra diferencias significativas entre LFT y RSM y, en menor medida, en los pares RSM/GRPW, RSM/MTS y CUMRED/GRPW. Podemos concluir que estos seis algoritmos forman un grupo relativamente homogéneo, con distancias medias al óptimo en torno al 3-4%, aunque RSM parece ser ligeramente menos eficiente.

Al aplicarlo al grupo de los seis algoritmos aleatorizados el test de Friedman obtiene una significatividad de 0.0594. El grupo es más homogéneo que en el caso anterior, pero el test de Wilcoxon muestra algunas diferencias muy significativas: LSTAL/LFTAL y RSMAL/LFTAL. Nuevamente la conclusión global sería la de que forman un grupo homogéneo, con distancias medias al óptimo en torno al 2%, con las diferencias ya señaladas.

La Tabla 5.2 puede usarse también para comparar cada pareja de algoritmos, determinista y aleatorizado. En los seis casos hay diferencias significativas entre ellos, muy acusadas salvo en el par GRPW/GRPWAL. La impresión numérica y visual producida por los resultados de la sección anterior queda, pues, contrastada estadísticamente. Podemos concluir, por tanto, que la aleatorización de las reglas de prioridad produce nuevos algoritmos heurísticos probadamente más eficientes y su uso debe ser considerado siempre que se disponga de un tiempo de computación suficiente.

TABLA 5.2

Resultados de los tests de normalidad para los doce algoritmos heurísticos.

| ALGORITMO | TEST DE KOLMOGOROV - SMIRNOV | | | COEFICIENTE DE SIMETRIA |
|---|---|---|---|---|
| | Dif. máxima | Estadístico | Valor P | |
| CUMRED | 0.1344 | 0.932 | 0.351 | 1.6983 |
| CUMREDAL | 0.1737 | 1.203 | 0.110 | 1.7567 |
| GRPW | 0.1579 | 1.094 | 0.182 | 1.7307 |
| GRPWAL | 0.1757 | 1.217 | 0.103 | 1.7575 |
| LFT | 0.1351 | 0.936 | 0.345 | 1.2953 |
| LFTAL | 0.1433 | 0.993 | 0.278 | 1.4521 |
| LST | 0.1054 | 0.730 | 0.660 | 0.6963 |
| LSTAL | 0.1602 | 1.111 | 0.170 | 1.4942 |
| MTS | 0.1215 | 0.842 | 0.477 | 1.2504 |
| MTSAL | 0.1468 | 1.017 | 0.252 | 1.8793 |
| RSM | 0.0907 | 0.629 | 0.824 | 0.8034 |
| RSMAL | 0.2284 | 1.583 | 0.013 | 1.6729 |

| | CUMRED | GRPW | LFT | LST | MTS | RSM | CUMREDAL | GRPWAL | LFTAL | LSTAL | MTSAL | RSMAL |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CUMRED | 1.000 | | | | | | | | | | | |
| GRPW | .0201 | 1.000 | | | | | | | | | | |
| LFT | .6455 | .3946 | 1.000 | | | | | | | | | |
| LST | .3644 | .0147 | .1073 | 1.000 | | | | | | | | |
| MTS | .5360 | .3494 | .9651 | .4838 | 1.000 | | | | | | | |
| RSM | .0951 | .0118 | .0030 | .7720 | .0247 | 1.000 | | | | | | |
| CUMREDAL | .0000 | .0024 | .0000 | .0000 | .0001 | .0000 | 1.000 | | | | | |
| GRPWAL | .0002 | .0191 | .0008 | .0001 | .0011 | .0000 | .2319 | 1.000 | | | | |
| LFTAL | .0000 | .0006 | .0000 | .0000 | .0000 | .0000 | .1352 | .0270 | 1.000 | | | |
| LSTAL | .0000 | .0098 | .0003 | .0000 | .0005 | .0000 | .0778 | .8811 | .0019 | 1.000 | | |
| MTSAL | .0001 | .0032 | .0002 | .0000 | .0001 | .0000 | .6567 | .2640 | .0814 | .0483 | 1.000 | |
| RSMAL | .0069 | .1031 | .0305 | .0078 | .0736 | .0004 | .0219 | .0374 | .0099 | .1717 | .0503 | 1.000 |

Tabla 5.2

Resultados del test de Wilcoxon para los doce algoritmos estudiados

## 6. REFERENCIAS


[1] **Alvarez-Valdés, R.** and **Tamarit, J.M.** (1989). "Heuristic algorithms for resource-constrained project scheduling: a review and an empirical analysis". Advances in Project Scheduling (R. Slowinski y J. Weglarz, eds.) Elsevier, pp. 113-134.

[2] **Brand, J.D.**, **Meyer, W.L.** and **Schaffer, L.R.** (1964). "The Resource Scheduling problem in Construction". Civil Engineering Studies, Report No.5, Dept. of Civil Engineering University of Illinois, Urbana.

[3] **Christofides, N.**, **Alvarez-Valdés, R.** and **Tamarit, J.M.** (1987). "Project scheduling with resource constraints: A branch and bound approach". European Journal of Operational Research 29 (3), 262-273.

[4] **Cooper, D.F.** (1976). "Heuristics for scheduling resource-constrained projects:an experimental investigation". Management Science 22 (11), 1186-1194.

[5] **Davis, E.W.** (1973). "Project Scheduling under resource constraints: Historical review and categorization of procedures". AIIE Transactions 5 (4), 297-313.

[6] **Davis, E.W.** and **Patterson, J.H.** (1975). "A comparison of heuristic and optimal solutions in resource-constrained project scheduling". Management Science 21 (8), 944-955.

[7] **Kelley, J.** and **Walker, M.** (1959). "Critical Path Planning and Scheduling". Proceedings of the Eastern Joint Computer Conference.

[8] **Lawrence, S.R.** (1984). "An experimental investigation of heuristic scheduling techniques". GSIA, Carnegie-Mellon, Pittsburgh.

[9] **Stinson, J.P.**, **Davis, E.W.** and **Khumawala, B.H.** (1978). " Multiple resource-constrained scheduling using branch and bound". AIIE Transactions 10, 252-259.