# Aplicación de la heuristica de CDS en la secuenciación de n tareas en m máquinas: un caso de estudio

## Application CDS´ heuristic for sequencing n jobs trough m machines: a study case

Jorge Hernán Restrepo Correa, María Elena Bernal Loaiza, Germán Cock Sarmiento
*Ingeniería Industrial, Universidad Tecnológica de Pereira, Pereira, Colombia*
jhrestrepoco@utp.edu.com
bernal@utp.edu.co
cook20038@gmail.com

***Resumen*— Este documento presenta como es aplicada la heurística de CDS para resolver el problema de programación de n tareas a través de m maquinas. El documento hace de forma breve una introducción al problema del Flow Shop, el modelo matemático, los pasos para resolver el problema, la solución con base en una medida de desempeño y las conclusiones.**

***Palabras clave*— Heurística, programación de tareas.**

***Abstract*— this paper shows how is applied the CDS´ heuristic to solve a programming problem of n jobs through m machines. The document explains a short introduction of Flow shop problem, the mathematical model, and the steps to solve the problem, problem solution and conclusions.**

***Key Word* —Heuristic, jobs programming**

## I. INTRODUCCIÓN

En 1950[1] aparece la teoría de Scheduling como una rama independiente de la investigación de operaciones y se plantea el interrogante de determinar la mejor programación. Por lo tanto, La secuenciación[2] adecuada de pedidos constituye un importante problema que se plantea dentro de la Dirección de Operaciones a corto plazo y el orden en que los pedidos serán atendidos o procesados, o en general el orden en el que cualquier tipo de tareas serán realizadas, no resulta indiferente, sino que determinará algún parámetro de interés cuyos valores convendrá optimizar en la medida de lo posible. Así podrá verse afectado el costo total de ejecución de las tareas, el tiempo necesario para concluirlas o el stock de productos en curso que será generado. Esto conduce de forma directa al problema de determinar cuál será el orden más adecuado para llevar a cabo las tareas con vistas a optimizar alguno de los anteriores parámetros u otros similares. Se trata de un problema de secuenciación o scheduling que se presenta de forma habitual en la programación de operaciones a corto plazo en entornos industriales o manufactureros y que puede adoptar una enorme variedad de formulaciones.

La gran dificultad para resolver el problema determinando una secuencia óptima, o al menos admisible, junto con la importancia de conseguirlo, han hecho proliferar reglas, más o menos complejas, muchas de ellas heurísticas, y algunas incluso empíricas, que proporcionan soluciones rápidas y fáciles de calcular destinadas a su uso en situaciones de trabajo reales.

Los modelos [3] de secuenciales tienen aplicaciones principalmente en un taller de tareas, donde un conjunto de máquinas, de propósito general, ejecutan una serie de operaciones sobre órdenes de trabajos o tareas de producción. Las tareas son a menudo únicas y ordenadas por un determinado cliente. El proceso en el taller de tareas es un modelo fundamental para un considerable número de sistemas operacionales, tales como las actividades de mantenimiento, asignación de aulas de clase a un grupo de materias, la programación de llegada y salida de buses, la programación de n tareas en m máquinas en un taller de fabricación intermitente, etc.

La programación de un taller de tareas consiste en determinar el orden o la secuencia de las tareas en las máquinas para optimizar alguna medida de ejecución.

Fecha de Recepción: 18 de Mayo de 2012
Fecha de Aceptación: 30 de Diciembre de 2012

Existen cuatro factores que describen y clasifican un problema específico de programación de un taller de tareas de acuerdo a:

1. El patrón de llegada de los trabajos: si n tareas llegan simultáneamente al taller y quedan disponibles para iniciar su proceso tendremos un problema d e programación estática. Si las tareas llegan intermitentemente, posiblemente de acuerdo a un proceso estocástico, el problema de programación es dinámico.

2. El número de máquinas que integran el taller. Existe un problema de secuenciación cuando n trabajos son programados en m maquinas.

3. El flujo de producción: el flujo de proceso de las tareas a través de las máquinas debe ser especificado, si todas las tareas siguen la misma ruta el flujo de producción es continuo o en serie. En el extremo opuesto, donde no existe una ruta preconcebida de procesos se tiene un taller cuyo flujo de producción es aleatorio. Los trabajos pueden ser independientes unos de otros, o bien interdependientes. Cuando se mezclan los diferentes tipos de flujos de producción, los de serie con los aleatorios, existen rutas generales de proceso.

4. El objetivo que se desea optimizar: la medida de desempeño que frecuentemente se utiliza es la optimización del tiempo total de proceso de todas las tareas o trabajos en todas las máquinas, pero se puede pensar también en la tardanza máxima, tardanza promedio o mínimo número de trabajos tardíos entre otras.

## II. REGLAS Y NOTACIÓN[4]

En todos los problemas de programación considerados en numero de tareas y maquinas son finitos. En número de trabajos es denotado por n y el número de máquinas por m. normalmente j se refiere a la tarea e i para la máquina. Si una tarea requiere un número de pasos de proceso u operaciones, entonces el par (i,j) significa el paso del proceso del trabajo j en la máquina i. las siguientes piezas de datos son asociadas con el trabajo j.

1. Tiempo de proceso $p_{i,j}$ : representa el tiempo de proceso del trabajo j en la máquina i.
2. Fecha de llegada $r_j$ : esto significa cuando el trabajo llega del trabajo j al sistema.
3. Fecha de entrega $d_j$ : Fecha prometida para la entrega del trabajo j.
4. Importancia o peso $w_j$ : Es básicamente el factor de prioridad, denotando la importancia del trabajo j con relación a los otros trabajos

Un problema de programación es descrito por la tripleta $\alpha/\beta/\gamma$. Donde:

$\alpha$: Este campo contiene el ambiente de la máquina

$\beta$ : En este campo contiene las características del proceso y las restricciones.

$\gamma$ : En este campo contiene el objetivo a ser minimizado.

## III. EL PROBLEMA DEL FLOW SHOP (*Fm*)[5]

Hay m máquinas en serie. Cada trabajo tiene que ser procesado en cada una de las m máquinas. Cada trabajo tiene que seguir la misma ruta. Por ejemplo, primero en la máquina 1, luego en la máquina 2 y así sucesivamente. Después de la terminación en una máquina, un trabajo es unido a la cola de la siguiente máquina. Usualmente todas las colas son asumidas con la disciplina de primero en llegar primero en ser atendido FIFO. Si la disciplina FIFO esta en el efecto del FlowShop, es referida como una permutación Flow Shop y el campo B incluye la entrada ***prmu***.

El Flowshop[6] ha sido una investigación muy activa y enriquecedora desde el documento fundamental de Johnson en 1954. El problema del Flowshop (FSP) un conjunto N de n tareas no relacionadas están para ser procesadas en un conjunto M de m maquinas. Estas maquinas están dispuestas en serie y cada tarea tiene que visitar a todas ellas en el mismo orden. Este orden puede ser asumido como 1, ….., m. Por lo tanto cada tarea j, j pertenece a N que contiene n tareas. Los tiempos de procesamiento de cada tarea en las maquinas son conocidos, no negativos y determinísticos. Ellos se pueden expresar por Pij, j pertenece a N, i pertenece a J. Se asume lo siguiente:

- Todos los trabajos son independientes y disponibles para ser procesados.
- Las maquinas están continuamente disponibles
- Cada maquina solo puede procesar una tarea a la vez
- Cada trabajo puede ser procesado solo en una maquina a la vez
- Una vez un el procesamiento de un trabajo dado ha iniciado en una maquina dada, no puede ser interrumpido y el proceso continua hasta ser completado
- Los tiempos de preparación (setup) son independientes de la secuencia y son incluidos en el tiempo de proceso, o ignorados

El objetivo es obtener una secuencia de producción de los N trabajos en las M maquinas así que un criterio dado es optimizado. Una secuencia de producción es normalmente definida por cada máquina. Desde que hay n! posibles permutaciones por tarea por maquina, el numero total de secuencias posibles es $(n!)^m$. Sin embargo, una simplificación en la literatura`es asumir que la misma permutación de la tarea es mantenida a lo largo de todas las maquinas. Esto efectivamente prohíbe a la tarea pase entre maquinas y reduce el espacio de soluciones a n!. Con esta simplificación, el problema es referido como el problema de programación Flowshop de permutaciones o PFSP.

## IV. HEURÍSTICA DE CDS.

En las pasadas tres décadas, extensivas investigaciones han sido hechas sobre el problema del Flow Shop, pero no hay algoritmos que provean una fácil solución óptima. Las técnicas de programación entera y el branch and bound pueden ser usadas para encontrar una óptima solución. Sin embargo, ellos no son efectivos en problemas grandes o igual en problemas medianos. El problema del Flow Shop ha sido presentado verdaderamente como un problema NP completo. Por esta razón, muchas heurísticas han sido desarrolladas para entregar una solución muy buena y de forma rápida.

Campbeel, Dudek, y Smith[7] usan una múltiple aplicación del algoritmo de Johnson para dos máquinas para tratar de obtener una buena programación en el problema del Flowshop. Esencialmente, ellos crean (m-1) problemas de programación, los cuales resuelven con el algoritmo de Johnson. Para ser precisos los k problemas (k=1,2,..(m-1)) son formados como sigue. Los tiempos de proceso para la primera maquina para el i-ésimo trabajo es:

$$a_i^{(k)} = \sum_{j=1}^{k} P_{ij}$$

Esto es la suma de los tiempos de proceso para la i-ésima tarea en el primer k de maquinas actuales. De la misma manera, se construye el tiempo para la segunda maquina que es la suma de los tiempos de proceso para la i-ésima tarea en la última k de las maquinas actuales.

$$b_i^{(k)} = \sum_{j=m-k+1}^{m} P_{ij}$$

Note que si m=3 k=2 y esto corresponde al caso especial del algoritmo de Johnson. Por lo tanto la heurística CDS, puede ser vista como una generalización intuitiva de este problema.

A continuación presentamos el algoritmo de Johnson[8]:

*Paso1*: hacer k=1, *l*=n

*Paso 2*: elabore una lista de los trabajos no programados = { $j_1$, $j_2 \cdots j_n$}.

*Paso 3*: encuentre el menor valor de los tiempos $a_i$ y $b_i$ par alas tareas no programadas.

*Paso 4*: si el menor tiempo es para $j_i$ sobre la primera máquina o sea que $a_i$ es el menor de todos, entonces

i. Programe j en la k-esima posición de la secuencia de procesamiento.
ii. Elimine $j_i$ de la lista de tareas no programados.
iii. Incremente k a k+1
iv. Vaya al paso 6.

*Paso 5*: si el menor tiempo es para $j_i$ sobre la segunda máquina o sea que $b_i$ es el menor de todos, entonces

v. Programe j en la l-esima posición de la secuencia de procesamiento.
vi. Elimine $j_i$ de la lista de tareas no programados.
vii. redusca *l* a (*l*-1)
viii. Vaya al paso 6.

*Paso 6*: si hay algún trabajo sin programar, vaya al paso 3. De lo contrario, pare.

## V. PROBLEMA

Para mostrar el funcionamiento de la heurística de CDS, planteamos un problema de 10 tareas en 5 máquinas. El objetivo es minimizar el tiempo de terminación o Cmáx.

El problema a tratar se describe a continuación:

| | MÁQUINA | | | | |
|---|---|---|---|---|---|
| TAREA | 1 | 2 | 3 | 4 | 5 |
| 1 | 3 | 7 | 3 | 3 | 2 |
| 2 | 10 | 4 | 9 | 9 | 8 |
| 3 | 7 | 6 | 3 | 1 | 10 |
| 4 | 2 | 3 | 1 | 7 | 1 |
| 5 | 3 | 2 | 4 | 2 | 4 |
| 6 | 10 | 8 | 7 | 10 | 8 |
| 7 | 9 | 1 | 10 | 4 | 4 |
| 8 | 10 | 5 | 8 | 1 | 5 |
| 9 | 8 | 2 | 9 | 4 | 1 |
| 10 | 6 | 1 | 7 | 4 | 4 |

Con m= 5 y k=1 y aplicando el CDC se tiene el primer problema a resolver con Johnson.

| | Maquina | |
|---|---|---|
| TAREA | $a_i^{(k)} = \sum_{j=1}^{k} P_{ij}$ | $b_i^{(k)} = \sum_{j=m-k+1}^{m} P_{ij}$ |
| 1 | 3 | 2 |
| 2 | 10 | 8 |
| 3 | 7 | 10 |
| 4 | 2 | 1 |
| 5 | 3 | 4 |
| 6 | 10 | 8 |
| 7 | 9 | 4 |
| 8 | 10 | 5 |
| 9 | 8 | 1 |
| 10 | 6 | 4 |

Se obtiene la siguiente secuencia:

| TAREA | **1** | **3** | **2** | **6** | **8** | **5** | **7** | **10** | **4** | **9** |
|---|---|---|---|---|---|---|---|---|---|---|

Y al computar los $P_{ij}$ en cada máquina se tiene:

| | MÁQUINA | | | | |
|---|---|---|---|---|---|
| TAREA | 1 | 2 | 3 | 4 | 5 |
| 1 | 3 | 10 | 13 | 16 | 18 |
| 3 | 10 | 16 | 19 | 20 | 30 |
| 2 | 20 | 24 | 33 | 42 | 50 |
| 6 | 30 | 38 | 45 | 55 | 63 |
| 8 | 40 | 45 | 53 | 56 | 68 |
| 5 | 43 | 47 | 57 | 59 | 72 |
| 7 | 52 | 53 | 67 | 71 | 76 |
| 10 | 58 | 59 | 74 | 78 | 82 |
| 4 | 60 | 63 | 75 | 85 | 86 |
| 9 | 68 | 70 | 84 | 89 | 90 |

Y un Cmáx= 90

Con m=5 y k=2, se tiene el segundo problema a resolver con Johnson.

| | Maquina | |
|---|---|---|
| TAREA | $a_i^{(k)} = \sum_{j=1}^{k} P_{ij}$ | $b_i^{(k)} = \sum_{j=m-k+1}^{m} P_{ij}$ |
| 1 | 10 | 5 |
| 2 | 14 | 17 |
| 3 | 13 | 11 |
| 4 | 5 | 8 |
| 5 | 5 | 6 |
| 6 | 18 | 18 |
| 7 | 10 | 8 |
| 8 | 15 | 6 |
| 9 | 10 | 5 |
| 10 | 7 | 8 |

Se obtiene la siguiente secuencia:

| TAREA | **4** | **5** | **6** | **2** | **3** | **7** | **10** | **8** | **1** | **9** |
|---|---|---|---|---|---|---|---|---|---|---|

Y al computar los $P_{ij}$ en cada máquina se tiene:

| | MÁQUINA | | | | |
|---|---|---|---|---|---|
| TAREA | 1 | 2 | 3 | 4 | 5 |
| 4 | 2 | 5 | 6 | 13 | 14 |
| 5 | 5 | 7 | 11 | 15 | 19 |
| 6 | 15 | 23 | 30 | 40 | 48 |
| 2 | 25 | 29 | 39 | 49 | 57 |
| 3 | 32 | 38 | 42 | 50 | 67 |
| 7 | 41 | 42 | 52 | 56 | 71 |
| 10 | 47 | 48 | 59 | 63 | 75 |
| 8 | 57 | 62 | 70 | 71 | 80 |
| 1 | 60 | 69 | 73 | 76 | 82 |
| 9 | 68 | 71 | 82 | 86 | 87 |

Y un Cmáx= 87

Con m=5 y k=3, se tiene el segundo problema a resolver con Johnson.

| | Maquina | |
|---|---|---|
| TAREA | $a_i^{(k)} = \sum_{j=1}^{k} P_{ij}$ | $b_i^{(k)} = \sum_{j=m-k+1}^{m} P_{ij}$ |
| 1 | 13 | 8 |
| 2 | 23 | 26 |
| 3 | 16 | 14 |
| 4 | 6 | 9 |
| 5 | 9 | 10 |
| 6 | 25 | 25 |
| 7 | 20 | 18 |
| 8 | 23 | 14 |
| 9 | 19 | 14 |
| 10 | 14 | 15 |

Se obtiene la siguiente secuencia:

| TAREA | 4 | 5 | 10 | 2 | 6 | 7 | 3 | 8 | 9 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|

Y al computar los $P_{ij}$ en cada máquina se tiene:

| | MÁQUINA | | | | |
|---|---|---|---|---|---|
| TAREA | 1 | 2 | 3 | 4 | 5 |
| 4 | 2 | 5 | 6 | 13 | 14 |
| 5 | 5 | 7 | 11 | 15 | 19 |
| 10 | 11 | 12 | 19 | 23 | 27 |
| 2 | 21 | 25 | 34 | 43 | 51 |
| 6 | 31 | 39 | 46 | 56 | 64 |
| 7 | 40 | 41 | 56 | 60 | 68 |
| 3 | 47 | 53 | 59 | 61 | 78 |
| 8 | 57 | 62 | 70 | 71 | 83 |
| 9 | 65 | 67 | 79 | 83 | 84 |
| 1 | 68 | 75 | 82 | 86 | 88 |

Y un Cmáx= 88

Con m=5 y k=4, se tiene el segundo problema a resolver con Johnson.

| Maquina |
|---|

| TAREA | $a_i^{(k)} = \sum_{j=1}^{k} P_{ij}$ | $b_i^{(k)} = \sum_{j=m-k+1}^{m} P_{ij}$ |
|---|---|---|
| 1 | 16 | 15 |
| 2 | 32 | 30 |
| 3 | 17 | 20 |
| 4 | 13 | 12 |
| 5 | 11 | 12 |
| 6 | 35 | 33 |
| 7 | 24 | 19 |
| 8 | 24 | 19 |
| 9 | 23 | 16 |
| 10 | 18 | 16 |

Se obtiene la siguiente secuencia

| TAREA | 5 | 3 | 6 | 2 | 7 | 8 | 9 | 10 | 1 | 4 |
|---|---|---|---|---|---|---|---|---|---|---|

Y al computar los $P_{ij}$ en cada máquina se tiene:

| | MÁQUINA | | | | |
|---|---|---|---|---|---|
| TAREA | 1 | 2 | 3 | 4 | 5 |
| 5 | 3 | 5 | 9 | 11 | 15 |
| 3 | 10 | 16 | 19 | 20 | 30 |
| 6 | 20 | 28 | 35 | 45 | 53 |
| 2 | 30 | 34 | 44 | 54 | 62 |
| 7 | 39 | 40 | 54 | 58 | 66 |
| 8 | 49 | 54 | 62 | 63 | 71 |
| 9 | 57 | 59 | 71 | 75 | 76 |
| 10 | 94 | 145 | 145 | 145 | 145 |
| 1 | 97 | 152 | 155 | 158 | 160 |
| 4 | 99 | 155 | 156 | 165 | 166 |

Y un Cmáx= 166

## VI. CONCLUSIONES

- La mejor secuencia encontrada es:

| TAREA | 4 | 5 | 6 | 2 | 3 | 7 | 10 | 8 | 1 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|

Con Cmax de 87 unidades de tiempo.

- El CDS muestra la facilidad en su operatividad para encontrar la secuencia al hacerlo igual que el algoritmo de Johnson y el tamaño de su búsqueda lo expresa en la resolución de (m-1) problemas de n tareas en 2 máquinas.

## RECOMENDACIONES

Se recomienda seguir resolviendo este problema con otros algoritmos o heurísticas con el principal propósito de mostrar su operatividad y la calidad de los resultados, entre ellos esta probar cl el algoritmo NEH.

## REFERENCIAS

[1]. Muños P Jose L,The problema of task scheduling and sequencing in the sawmill drying process, Theoria vol 16(1) p páginas 15-22, 2007.

[2]. Díaz Zuleyka,Fernández José, Martínez Paloma, Secuenciación de tareas en el ámbito de la producción: una aplicación del algoritmo de recocido simulado, Biblioteca Universidad Complutence,2004

[3]. http://eprints.ucm.es/6832/1/04008.pdf

[4]. Ospina Bolaños Dagoberto, Sistemas Administrativos de Producción y Operaciones, Programación Secuencial, Editorial UTP 1996, página 231.

[5]. M.L Pinedo, Schedulling: theory,algorithms and systems. Editorial Springer, tercera edición, Página 13.

[6]. M.L Pinedo, Schedulling: theory, algorithms and systems. Editorial Springer, tercera edición, Página 15.

[7]. Naderi, B.; Ruiz, Rubén, The distributed permutation flowshop scheduling problem , Computers & Operations Research Volume: 37 - 2010, Páginas 754-768

[8]. French Simon, Sequencing and scheduling: An introduction to the Mathematics of the Job-Shop, Ed John Willey & Son – 1982, página 171.

[9]. French Simon, Sequencing and scheduling: An introduction to the Mathematics of the Job-Shop, Ed John Willey & Son – 1982, página 70.