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

Application gupta´s heuristic for sequencing n jobs trough m machines: a study case

Jorge Hernán Restrepo C.
Ingeniería Industrial, Universidad Tecnológica de Pereira, Pereira, Colombia
jhrestrepoco@utp.edu.co

***Resumen*— Este documento presenta como es aplicada la heurística de Gupta para resolver un 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 Gupta´s 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, the steps to solve the problem, problem solution and conclusions.**

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

I. INTRODUCCIÓN

Los modelos 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.

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 de 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

En todos los problemas de programación considerados en número 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.

Fecha de Recepción: 25 de Enero de 2011
Fecha de Aceptación: 11 de Agosto de 2011

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:

α: Este campo contiene el ambiente de la máquina

β : En este campo contiene las características del proceso y las restricciones.

γ : En este campo contiene el objetivo a ser minimizado.

III. EL PROBLEMA DEL FLOW SHOP (*Fm*)

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 Flow Shop, es referida como una permutación Flow Shop y el campo B incluye la entrada ***prmu***.

IV. HEURÍSTICA DE GUPTA.

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.

Gupta diseño un método el cual es similar al de Palmer, excepto que el define los índices en un manera diferente, tomando dentro de la cuenta unos interesantes hechos de la optimización de la regla de Johnson para el problema de tres máquinas

El índice S_j para el trabajo j se calcula así:

$$S_j = \frac{e_j}{min_{t\le k\le m-1}\{t_{k,j} + t_{k+1,j}\}}$$

$$e_j = \begin{cases} -1 \; si \; t_{1,j} < t_{m,j} \\ +1 \; si \; t_{1,j} \geq t_{m,j} \end{cases}$$

Después de esto, los trabajos son secuenciados de acuerdo al índice, de menor a mayor. Si hay un empate se programa primero la tarea con menor suma total de tiempos de proceso.

V. PROBLEMA.

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

	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

Calculando el índice, se obtiene:

e_j	$t_{k,j} + t_{k+1,j}$				S_j	j
1	10	10	6	5	0,2	1
1	14	13	18	17	0,076923077	2
-1	13	9	4	11	-0,25	3
1	5	4	8	8	0,25	4
-1	5	6	6	6	-0,2	5
1	18	15	17	18	0,066666667	6
1	10	11	14	8	0,125	7
1	15	13	9	6	0,166666667	8
1	10	11	13	5	0,2	9
1	7	8	11	8	0,142857143	10

Ordenando las tareas de menor a mayor por índice se obtiene:

S_j	j
-0,25	3
-0,2	5
0,06666667	6
0,07692308	2
0,125	7
0,14285714	10
0,16666667	8
0,2	1
0,2	9
0,25	4

Calculando el Cmáx **:**

	MÁQUINA				
TAREA	**1**	**2**	**3**	**4**	**5**
3	7	13	16	17	27
5	10	15	20	22	31
6	20	28	35	45	53
2	30	34	44	54	62
7	39	40	54	58	66
10	45	46	61	65	70
8	55	60	69	70	75
1	58	67	72	75	77
9	66	69	81	85	86
4	68	72	82	92	93

Cmax: 93

VI. CONCLUSIONES Y RECOMENDACIONES

- La heurística es una herramienta amigable, que permite encontrar rápidamente soluciones sin realizar exhaustivos cómputos.
- En la práctica se requieren de herramientas que sean fáciles de entender y manipular. Día a día se requieren hacer programaciones y programaciones trabajos, donde el cliente requiere oportunamente una promesa de entrega

REFERENCIAS

[1]. Ospina Bolaños Dagoberto, Sistemas Administrativos de Producción y Operaciones, Programación Secuencial, Editorial UTP 1996, página 231.
[2]. M.L Pinedo, Schedulling: theory,algorithms and systems. Editorial Springer, tercera edición, Página 13.
[3]. M.L Pinedo, Schedulling: theory,algorithms and systems. Editorial Springer, tercera edición, Página 15.
[4]. Mitsuo Gen, Runwei Cheng, Genetic algorithms and engineering design, Página 176.
[5]. Ospina Bolaños Dagoberto, Sistemas Administrativos de Producción y Operaciones, Programación Secuencial, Editorial UTP 1996, página 248