PROGRAMACIÓN DISYUNTIVA PARA n TAREAS EN m MÁQUINAS EN PRODUCCIÓN POR PEDIDOS.

PEDRO PABLO BALLESTEROS SILVA
Ingeniero Industrial,
Ms. en Investigación de Operaciones y Estadística con énfasis en producción.
Profesor Asociado.
Universidad Tecnológica de Pereira.
ppbs@utp.edu.co

DIANA PAOLA BALLESTEROS RIVEROS
Estudiante de VIII semestre de Ingeniería Industrial
Universidad Tecnológica de Pereira
dianap@utp.edu.co

MARÍA ELENA BERNAL LOAIZA
Ingeniera de Sistemas
UNAD
Profesora Catedrática
Universidad Tecnológica de Pereira.
mbernal@utp.edu.co

RESUMEN
Este articulo trata de la producción por pedidos, aplicada en organizaciones donde cada pedido del cliente es único. Inicialmente, se describen brevemente algunos métodos que facilitan el uso adecuado de los recursos disponibles y la efectiva toma de decisiones, luego se presenta la producción por pedidos en una sola estación de trabajo con una máquina o con un conjunto de máquinas en paralelo, describiendo en forma breve algunos modelos que consideran distintas funciones objetivo.

Al final se destaca la importancia de la programación matemática y su aplicación en un caso especifico a la producción por pedidos, el cual es solucionado mediante la utilización de un novedoso software llamado LEKIN.

PALABRAS CLAVES: Programación por pedidos, algoritmos, heurísticas, modelos, máquinas en paralelo, gráficas Gantt.

ABSTRACT
This article is about the production for orders, applied in organizations where each order of the client is only. Initially, some methods that facilitate the appropriate use of the available resources and the effective taking of decisions are described, then the production by orders is presented in a single working station with a machine or with a group of machines in parallel, describing in brief form some models that consider different functions objective.

At the end is stand out the importance of the mathematical programming and their application in a case specify to the job shop, which is solved with a new software called LEKIN

KEYWORDS: *Job shop, algorithms, heuristics, models, machines in parallel, Gantt chart, LEKIN*

1. INTRODUCCIÓN

En todas las organizaciones existe una diversidad de problemas económicos y financieros que no están aislados de los relacionados con unidades productivas, de la gestión y organización de la producción.

El objetivo de este artículo es precisamente abordar parte de estas situaciones, mediante la consideración de algunos métodos, procedimientos y técnicas que permitan llegar a una solución. Para ello, se realizó un análisis de las técnicas de organización de la producción, de la secuencia de actividades, de algunos métodos y modelos de investigación de operaciones (simulación, heurísticos y de optimización), de la teoría de la programación, entre otros.

De acuerdo con lo anterior, cuando hay *n* trabajos y cada trabajo visita un número *m* de máquinas o centros de trabajo, es posible seguir una ruta predeterminada[1]. En algunos modelos un trabajo puede visitar una máquina dada al menos una vez y en otros modelos un trabajo visita cada máquina más de una vez. El último caso afirma que la producción por pedidos está sujeta a la recirculación. Una generalización de la producción básica por pedido es la llamada producción flexible por pedidos, la cual consiste en una colección de centros de trabajo y cada centro de trabajo está formado por un número de máquinas similares dispuestas en paralelo. Cuando un trabajo visita un centro de trabajo, puede procesarse en cualquiera de las máquinas de ese centro de trabajo.

La producción por pedidos es adecuada para industrias, donde, cada pedido del cliente es único y tiene sus propios parámetros. Un ejemplo clásico de producción por pedidos es un hospital. Los pacientes son los trabajos. Cada paciente tiene que seguir una ruta dada y tiene que ser tratado en varias estaciones diferentes mientras él o ella pasan por el sistema.

Los problemas producción por pedidos son NP-duros (no polinomiales) y no pueden formularse como programas

[1] Para *n* trabajos el número posible de secuencias es $n!$ y si existen *n* tareas que se pueden procesar en *m* máquinas, el número de programas alternativos para esta serie de tareas es $(n!)^m$.

Fecha de Recepción: 30 Enero de 2006
Fecha de Aceptación: 4 Abril de 2006

lineales. Sin embargo, es factible que se formulen con programación entera o con programas disyuntivos.

En este artículo se explicará la producción por pedidos con una sola estación de trabajo; que es básicamente equivalente a una máquina o un número determinado de máquinas en paralelo. Se considerarán varias funciones objetivo. Así mismo, se presentará una formulación de la programación matemática para la producción por pedido con una sola máquina en cada estación de trabajo y el objetivo es el tiempo máximo de terminación de todos los trabajos (makespan), C_{max}. Dicha programación se realizará utilizando restricciones disyuntivas. Este tipo de restricción permite especificar que cada máquina sólo procese una tarea a la vez, es decir, dadas dos tareas j, g para una misma máquina, debe cumplirse la siguiente disyunción: o bien j precede a g o bien g precede a j. En estas restricciones se tienen en cuenta los tiempos de preparación para pasar de un pedido a otro dentro de cada máquina.

Seguidamente se hará una aproximación completamente diferente para la producción por pedido con el objetivo minimizar C_{max},, conociendo la programación aproximada de restricciones. Esta aproximación ha ganado la aceptación durante la última década gracias a la planeación de problemas del mundo real.

2. MÁQUINA ÚNICA Y LOS MODELOS DE LAS MÁQUINA EN PARALELO.

Una única máquina es el caso más simple de la producción por pedidos y un ambiente de máquinas en paralelo es equivalente a la producción flexible por pedidos que está compuesta por un solo centro de trabajo. Las técnicas de descomposición de las más complejas producciones por pedidos a menudo tienen que considerar una sola máquina y problemas derivados de las máquinas en paralelo dentro de su estructura.

A continuación se hace la descripción de una sola máquina y n trabajos: El trabajo j tiene un tiempo de proceso p_j, una fecha de liberación r_j, y una fecha de entrega d_j. Si el $r_j = 0$ y $d_j = \infty$, entonces el proceso de trabajo j está básicamente sin restricción. Para varios objetivos ciertas reglas de prioridad generan el programa óptimo. Si el objetivo es minimizar el total de los tiempos ponderados, es decir, $\Sigma w_j C_j$, y el procesamiento de los trabajos no está restringido, entonces la regla del tiempo ponderado de procesamiento más corto (WSPT), que fija la programación de los trabajos en el orden decreciente de w_j / p_j, es óptima. Si el objetivo es el retraso máximo L_{max} y los trabajos son liberados en el tiempo 0, entonces la regla de la fecha de entrega más temprana (EDD), que fija las órdenes de trabajos según d_j, es óptima. La regla de WSPT y regla de EDD son ejemplos de reglas de prioridad estáticas. Una regla que se relaciona un poco con la regla de EDD es la llamada regla Mínima Actividad Primero (MS) que selecciona, cuando la máquina está disponible en el momento t, el trabajo con la menor inactividad; la inactividad del trabajo j en momento t se define como el máx $(d_j - p_j - t, 0)$. Esta regla no opera exactamente de la misma manera como la regla de EDD, pero producirá programaciones que son similares a la programación de EDD. Sin embargo, la regla de MS es un ejemplo de una regla de prioridad dinámica en la cual, la prioridad de cada trabajo es una función de tiempo.

Otros objetivos, como la tardanza total $\Sigma\ \ _j$ y la tardanza ponderada total $\Sigma w_j T_j$, son más difíciles para optimizar que el tiempo completo ponderado o el retraso máximo. Una heurística que se aplica en este caso es la regla del primer costo aparente de retraso o tardanza (ATC). Si la máquina se libera en momento t, la regla de ATC selecciona entre los trabajos restantes el trabajo con el valor más alto de la expresión mostrada en la fórmula (1)

$$I_j(t) = \frac{w_j}{p_j} \exp\left(-\frac{\max(d_j - p_j - t, 0)}{K\overline{p}}\right), \qquad (1)$$

Donde K es un parámetro llamado escalar y $\overline{p}$ es el promedio de los tiempos de proceso de los trabajos que permanecen programados. La regla de prioridad ATC realmente es una mezcla de los pesos de WSPT y la regla de prioridad de MS expresada anteriormente. Ajustando el parámetro de escalar K la regla opera la regla WSPT o la regla MS.

Cuando los trabajos en un problema de una sola máquina tienen diferentes fechas de liberación r_j, los problemas son más complicados. Un problema que ha sido ampliamente estudiado es el de programación para una sola máquina no preventiva (no prioritaria) en el que los trabajos tienen fechas diferentes de liberación y el retraso máximo L_{max} tiene que ser minimizado. Este problema es no polinomial y desafortunadamente, no existe algoritmo en tiempo polinomial eficiente para solucionarlo. Sin embargo, el problema puede resolverse por ramificación y acotamiento o por la programación dinámica.

Cuando hay m máquinas en paralelo el makespan C_{max} es la programación dependiente. El objetivo de makespan juega un papel importante, ya que, cuando las cargas de varias máquinas tienen que ser balanceadas, se da lugar a otra regla de prioridad: la regla "primero los trabajos *de tiempo de Proceso más Largo (LPT)*". Según esta regla, siempre que se libere una de las máquinas, el trabajo más largo es seleccionado para ir al inicio. La intuición detrás de esta regla es clara: A uno le gustaría procesar los trabajos más pequeños hacia el final de la programación desde que eso sea más fácil para equilibrar las máquinas. Si los trabajos más pequeños van en último lugar, entonces los trabajos más largos tienen que ser fijados más al principio. La regla *LPT* desafortunadamente no garantiza una solución óptima.

Las reglas SPT y WSPT también son importantes en el escenario de las máquinas en paralelo. Si todos los *n* trabajos están disponibles en $t = 0$, entonces la regla no preventiva SPT minimiza el tiempo total de la realización ΣCj: la regla no preventiva SPT permanece óptima aún cuando se permiten las prioridades. Desafortunadamente, minimizando el tiempo total ponderado $\Sigma w_j C_j$ de realización en un escenario de máquina paralela cuando todos los trabajos están disponibles en $t = 0$ son no polinomiales, por lo que la regla de WSPT no minimiza el $\Sigma w_j C_j$ en este caso. Sin embargo, el WSPT todavía puede usarse como un heurístico y se garantiza generar una solución que está dentro de 22% de optimalidad. [2]

El problema más general de las máquinas en paralelo con el objetivo de la tardanza ponderada total ($\Sigma w_j T_j$) es aún más complejo. La regla de ATC descrita antes también es aplicable al escenario de las máquinas paralelas; sin embargo, la calidad de las soluciones deja algunas veces algo que desear.

3. PRODUCCIÓN POR PEDIDOS Y LA PROGRAMACIÓN MATEMÁTICA.

Al estudiar la producción por pedidos para *n* trabajos y *m* máquinas, se debe tener en cuenta que: cada trabajo tiene que ser procesado por varias máquinas en un orden dado y no hay ninguna recirculación. El proceso de trabajo *j* en la máquina *i* está referido como la operación *(i,j)* y su duración es el p_{ij}. El objetivo también es minimizar C_{max}.

El problema de minimizar C_{max} en la producción por pedidos sin que se presente recirculación se representa por medio de un gráfico disyuntivo. Se puede realizar un gráfico G con nodos fijos *N* y dos conjuntos de arcos *A* y *B*. Los nodos *N* corresponden a todas las operaciones *(i, j)* permitirán la realización de *n* trabajos. Los llamados arcos conjuntivos *A* representan las rutas de los trabajos. Si el arco $(i,j) \rightarrow (h,j)$ es parte de *A*, entonces el trabajo *j* tiene que ser procesado en la máquina *i* antes de que se procese en la máquina *h*, es decir, la operación *(i,j)* antecede a la operación *(h,j).* Dos operaciones que pertenecen a dos trabajos diferentes y que tienen que ser procesados en la misma máquina se conectan entre si por dos arcos llamados arcos disyuntivos que van en direcciones opuestas. Los arcos disyuntivos *B* forman las *m* cliques de arcos dobles, un clique para cada máquina. (Un clique es un término que en teoría de grafos se refiere a un gráfico en que dos nodos cualquiera se conectan entre si a algún otro; en este caso, cada conexión dentro de un clique es un par de arcos disyuntivos.) Todas las operaciones (nodos) en el mismo clique han sido realizadas en la misma máquina. Todos los arcos que salen de un nodo, conjuntivo o disyuntivo, tienen como longitud el tiempo de proceso de la operación que es representada por el nodo. Además, hay un origen U y un destino V que son los nodos mudos. El nodo del origen U tiene *n* arcos conjuntivos que se dirigen a las primeras operaciones de los *n* trabajos y el nodo destino V tiene *n* arcos conjuntivos que entran de todas las últimas operaciones. Los arcos que emanan desde el origen tienen longitud cero. (Ver figura 1, donde se denota este gráfico por *G = (N, A, B).)*

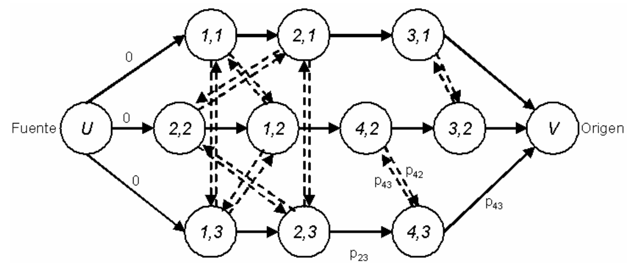


Figura 1. Gráfica dirigida para producción por pedidos con el tiempo máximo de terminación de todos los trabajos como objetivo.

Una programación factible corresponde a una selección de un arco disyuntivo de cada par tal que el resultado del gráfico es acíclico. Tal selección determina la secuencia en que las operaciones serán realizadas en esa máquina. Que una selección desde un clique ha sido acíclica se explica así: si hubo un ciclo dentro de un clique, una posible secuencia de las operaciones en las correspondientes máquinas no es factible. No puede ser inmediatamente obvio por qué no debe haber cualquier ciclo formado por los arcos conjuntivos y los arcos disyuntivos desde los diferentes cliques. Tal ciclo también correspondería a una situación que no es factible. Por ejemplo, si *(h,j)* y *(i,j)* denotan dos operaciones consecutivas que pertenecen al trabajo *j* y *(i,k)* y *(h,k)* representan dos operaciones consecutivas que pertenecen al trabajo *k.* Si bajo una programación de una operación dada *(i,j)* que precede a una operación *(i,k)* en la máquina *i* y la operación *(h,k)* que precede la operación *(h,j)* en la máquina *h*, entonces el gráfico contiene un ciclo con cuatro arcos, dos arcos conjuntivos y dos arcos disyuntivos desde los diferentes cliques. Tal programación es físicamente imposible. En resumen, si *D* denota el subconjunto de los arcos disyuntivos seleccionados y el gráfico *G (D)* está definido por el conjunto de arcos conjuntivos y el subconjunto *D*, entonces *D* corresponde a una posible programación si y solo si *G (D)* no contiene ningún ciclo dirigido.

El tiempo máximo de terminación de todos los trabajos C_{max}, (makespan) de una programación factible está determinado por el camino más largo en G (D) desde el origen (fuente) U al destino V. Este camino más largo consiste en un conjunto de operaciones donde la primera comienza en el tiempo 0 y la última finaliza en C_{max}. Cada operación en este camino es seguida

---

[2] PINEDO, Michael. Planning and scheduling in manufacturing and services. SPRINGER. USA, Standford University. Page 84

inmediatamente por la próxima operación en la misma máquina o por la próxima operación en el mismo trabajo en otra máquina. El problema de minimizar C_{max} (makespan) se reduce a encontrar una selección de arcos disyuntivos que minimizan la longitud del camino más largo (es decir, el camino crítico).

Hay varias formulaciones matemáticas para la programación por pedido, que incluyen un número de modelos en programación entera. Sin embargo, la formulación más utilizada es la llamada la programación disyuntiva. Ésta se relaciona estrechamente con la representación del gráfico disyuntivo de la programación por pedido.

Para presentar la programación disyuntiva, se considera que la variable y_{ij} denota el tiempo de iniciación de la operación *(i, j).* Se debe tener en cuenta que el conjunto N denota el conjunto de todas las operaciones *(i, j)* y el conjunto *A* el conjunto de todas las restricciones asignadas $(i, j)\rightarrow(h, j)$ que requiere el trabajo *j* para ser procesado en la máquina *i* antes de que se procese en la máquina *h,* el programa matemático siguiente minimiza el tiempo máximo de terminación de todos los trabajos C_{max} (makespan):

Minimizar C_{max}
Con las condiciones:
$y_{hj} - y_{ij} \geq p_{ij}$………… para todo $(i, j)\rightarrow(h, j) \in A$
$C_{max} - y_{ij} \geq p_{ij}$…………para todo $(i, j) \in N$
$y_{ij} - y_{ik} \geq p_{ik}$ o $y_{ik} - y_{ij} \geq p_{ij}$… para todo (i, k) y (i, j), $i=1,\ldots,n$
$y_{ij} \geq 0$……………… para todo $(i, j) \in N$

En este modelo, el primer conjunto de restricciones establecen que esa operación *(h, j)* no puede empezar antes de que la operación *(i, j)* esté terminada. El tercer conjunto de restricciones son llamadas restricciones disyuntivas; ellas establecen que algún ordenamiento existe entre las operaciones de trabajos diferentes que tienen que ser procesados en la misma máquina. Debido a estas restricciones la información está referida la formulación de la programación disyuntiva.

Lo anterior se ilustra con el siguiente ejemplo de Programación Disyuntiva, con cuatro máquinas y tres trabajos. La ruta, es decir, la secuencia de la máquina, así como los tiempos de proceso se presentan en la Tabla 1.

El objetivo consiste en determinar la variable C_{max}. El primer conjunto de restricciones tiene siete restricciones: dos para el trabajo 1, tres para el trabajo 2 y dos para el trabajo 3. Por ejemplo, una de estas es la observada en la formula 2:

$$y_{21} - y_{11} \geq 10 (= p_{11}) \quad (2)$$

Trabajos	Secuencia de máquina	Tiempo de proceso
1	1, 2, 3	$p_{11} = 10$, $p_{21} = 8$, $p_{31} = 4$
2	2, 1, 4, 3	$p_{22} = 8$, $p_{12} = 3$, $p_{42} = 5$, $p_{32} = 6$
3	1, 2, 4	$p_{13} = 4$, $p_{23} = 7$, $p_{43} = 3$

Tabla 1. La secuencia de la máquina y los tiempos de proceso.

El segundo conjunto está integrado por diez restricciones, una para cada operación. Un ejemplo se muestra en la fórmula (3)

$$C_{max} - y_{11} \geq 10 (= p_{11}) \quad (3)$$

El conjunto de restricciones disyuntivas contiene ocho restricciones: tres para la máquina 1, tres para la máquina 2, una para la máquina 3 y una para la máquina 4. Un ejemplo de restricción disyuntiva se observa en las formulas (4) y (5):

$$y_{11} - y_{12} \geq 3 (= p_{12}) \text{ o} \quad (4)$$
$$y_{12} - y_{11} \geq 10 (= p_{11}) \quad (5)$$

El último conjunto incluye diez restricciones no negativas, una para cada tiempo de iniciación.

Un problema de programación puede ser formulado como un programa disyuntivo, pero ello no significa que exista un procedimiento disponible para la solución estándar que siempre trabajará satisfactoriamente. Minimizar el tiempo máximo de terminación de todos los trabajos (makespan) en una programación por pedidos es un problema complejo y los procedimientos de las soluciones están basados en la enumeración o en heurísticas. Para encontrar las soluciones óptimas también se emplea el método de ramificación y acotamiento. (Branch and bound)

3.1 Modelo del ejemplo

Minimizar C_{max}
Sujeto a:
Primer conjunto de restricciones:

1 $y_{21} - y_{11} \geq 10$
2 $y_{31} - y_{21} \geq 8$
3 $y_{12} - y_{22} \geq 8$
4 $y_{42} - y_{12} \geq 3$
5 $y_{32} - y_{42} \geq 5$
6 $y_{23} - y_{13} \geq 4$
7 $y_{43} - y_{23} \geq 7$

Segundo conjunto de restricciones

1 $C_{max} - y_{11} \geq 10$
2 $C_{max} - y_{21} \geq 8$
3 $C_{max} - y_{31} \geq 4$
4 $C_{max} - y_{22} \geq 8$
5 $C_{max} - y_{12} \geq 3$
6 $C_{max} - y_{42} \geq 5$
7 $C_{max} - y_{32} \geq 6$
8 $C_{max} - y_{13} \geq 4$
9 $C_{max} - y_{23} \geq 7$
10 $C_{max} - y_{43} \geq 3$

Tercer conjunto de restricciones:

1	$Y_{12} - y_{11} \geq$	10	5	$y_{23} - y_{22} \geq$	8
2	$Y_{13} - y_{12} \geq$	3	6	$y_{23} - y_{21} \geq$	8
3	$Y_{13} - y_{11} \geq$	10	7	$y_{32} - y_{31} \geq$	4
4	$Y_{22} - y_{21} \geq$	8	8	$y_{43} - y_{42} \geq$	5

Restricción de no negatividad:

$y_{11}, y_{21}, y_{31}, y_{22}, y_{12}, y_{42}, y_{32}, y_{13}, y_{23}, y_{43} \geq 0$

3.2 Solución y análisis del ejercicio

Este problema se resuelve aplicando métodos heurísticos o métodos exactos. En este caso, se utiliza el sistema Lekin, que es uno de los muchos sistemas que han sido desarrollados para atender el complejo medio ambiente de las máquinas. Este sistema contiene un número de algoritmos y heurísticas de programación y se diseñó para permitir al usuario enlazar y probar sus propias heurísticas y comparar su desempeño con las heurísticas y algoritmos incluidos en el sistema Lekin.

El sistema puede trabajar en diferentes ambientes de máquinas como: máquinas simples, maquinas paralelas, flujo continuo, flujo flexible continuo, flujo intermitente y flujo flexible intermitente. Además, es capaz de distribuir las secuencias dependientes con los tiempos de iniciación en los ambientes antes citados.

Al ingresar toda la información del problema al software LEKIN, éste muestra inicialmente 4 ventanas, que son: gráfica Gantt, conjunto de trabajos, secuencia y cuadro de máquinas, las permiten visualizar diferentes aspectos del proceso. Luego, cuando se define el algoritmo a utilizar entre los 9 que ofrece LEKIN (ATCS, EDD, LPT, MS, SPT, WSPT, (descritos en el numeral 2), CR (Regla de proporción crítica. Los trabajos se organizan según la proporción del tiempo de salida hasta la fecha del vencimiento y el tiempo del proceso restante), FCFS (primero en entrar - primero en salir. Organiza los trabajos en el orden de sus fechas del descargo.), y WSPU), aparecen dos ventanas más (un libro de navegación y grafica de los objetivos). La información que brindan estas 6 ventanas es explicada en seguida:

- Gráfica Gantt (Ver figura 2.): En este cuadro es posible visualizar los tiempos de proceso de cada uno de los trabajos y el nivel de utilización de las máquinas. Aplicando al problema propuesto se puede observar que la máquina 1 es utilizada con una eficiencia óptima (no tiene tiempos ociosos), en la máquina 2 existe un tiempo ocioso menor debido a la espera por la máquina 1.

Los casos mas preocupantes lo registran las máquinas 3 y 4 cuyos tiempos ociosos son mucho más de la mitad del tiempo de proceso total.

- Conjunto de trabajos (Ver figura 3.): Contiene toda la información concerniente a cada trabajo (peso, tiempos de inicio y proceso, entre otros). En el ejercicio planteado se observa que los tres trabajos tienen igual prioridad. Así mismo, es de anotar que los trabajos 1 y 2 tienen un tiempo promedio de proceso más largo que el trabajo 3, pero a pesar de ello el tiempo de finalización del trabajo 2 y 3 fue algo mayor que el del trabajo 1.

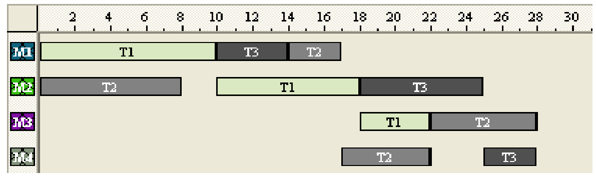


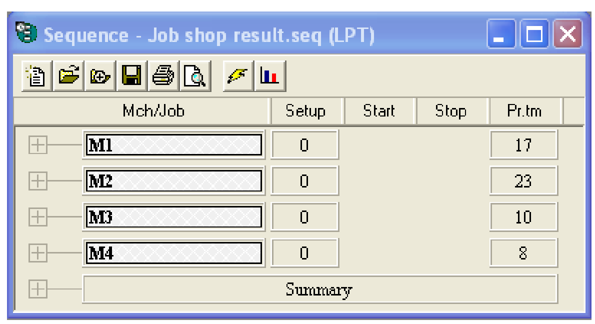
Figura 2. Gráfica Gantt para los resultados de la programación por pedidos.
Fuente: Resultados del software LEKIN.

Job Pool - Job shop result.job (Job Shop)

ID	Wght	Rls	Due	Pr.tm.	Stat.	Bgn	End	T	wT
T1	1	0	0	22		0	22	22	22
T2	1	0	0	22		0	28	28	28
T3	1	0	0	14		10	28	28	28

Figura 3. Ventana informativa del Job Shop.
Fuente: Resultados del software LEKIN.

- Secuencia (Ver figura 4.): Muestra la lista de trabajos en el orden en el que son procesados en cada una de las máquinas y el tiempo promedio de utilización de las mismas en el proceso (sin contar el tiempo ocioso). En el ejercicio se nota que la máquina 2 tiene el mayor tiempo promedio de utilización, lo que nos llevaría a estudiar la posibilidad de comprar otra máquina más moderna o que cumpla las mismas funciones, con el fin de disminuir los tiempos de entrega, así como, los tiempos ociosos de las maquinas 3 y 4.



Sequence - Job shop result.seq (LPT)

Mch/Job	Setup	Start	Stop	Pr.tm
M1	0			17
M2	0			23
M3	0			10
M4	0			8
Summary				

Figura 4. Tiempos totales de procesos de los trabajos por máquinas.
Fuente: Resultados del software LEKIN.

- Cuadro de máquinas: (Figura 5) Despliega toda la información con respecto a los centros de trabajo y a las máquinas. En la ventana se muestran primero los centros de trabajo y si el usuario desea ver que maquinas pertenecen a cada estación de trabajo, solo tiene que realizar un clic en el centro de trabajo deseado.

- Libro de navegación: (Figura 6) Muestra los resultados obtenidos en la aplicación de 7 de los 9 algoritmos que ofrece el software.

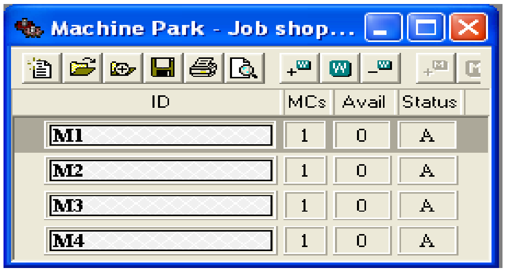


Figura 5. Vista del cuadro de máquinas
Fuente: Resultados del software LEKIN.

En la figura 6 se aprecia que el resultado de $C_{máx}$ al aplicar los algoritmos CR, EDD, FCFS, LPT y MS es mucho menor que si se utilizan los algoritmos SPT, WSPT y WSPU. Lo que nos lleva a concluir que los primeros logran la optimización y los segundos no modelan adecuadamente el problema.

Schedule	Time	C_{max}	T_{max}	$\sum U_j$	$\sum C_j$	$\sum T_j$	$\sum w_j C_j$	$\sum w_j T_j$
CR	1	28	28	3	78	78	78	78
FCFS	1	28	28	3	78	78	78	78
LPT	1	28	28	3	78	78	78	78
MS	1	28	28	3	78	78	78	78
SPT	1	44	44	3	96	96	96	96
WSPT	1	44	44	3	96	96	96	96
WSPU	1	44	44	3	96	96	96	96

Figura 6. Solución del problema empleando siete reglas de secuenciación.
Fuente: Resultados del software LEKIN.

- Gráfica de los objetivos: (figura 7) se observa gráficamente los resultados del objetivo minimizar $C_{máx}$ con los diferentes algoritmos que posee LEKIN.

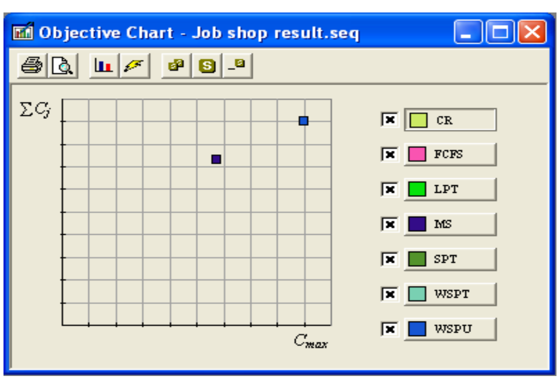


Figura 7. Gráfica de la solución del modelo.
Fuente: Resultados del software LEKIN.

4. CONCLUSIONES Y RECOMENDACIONES

Se pretende despertar o motivar a los empresarios a orientar la gestión de la producción hacia la utilización de procesos de sistematización, empleando técnicas y procedimientos de análisis que han sido desarrollados en las últimas dos décadas.

Presentar alternativas de solución de problemas que se observan en el área de producción en las pequeñas y medianas empresas.

La complejidad de los problemas de programación en si misma, debido a su explosión combinatorial, explica o muestra porque no es fácil llegar rápidamente a una solución y de ahí lo interesante que se torne su estudio, invitando a los lectores a considerar adecuadamente un criterio de elección entre la simulación, los procedimientos heurísticos o metaheurísticos o la búsqueda de una solución óptima.

Es claro que se necesita sólida habilidad y conocimiento detallado de las organizaciones y sus restricciones a la hora de abordar, establecer o diseñar el modelo de secuenciación.

En los problemas de secuenciación son muchos los objetivos que se pueden alcanzar, por ejemplo minimizar el tiempo total transcurrido, maximizar el nivel de producción, minimizar el retraso total, minimizar el retraso máximo, minimizar el costo de producción y en ocasiones al lograr un objetivo puede resultar la consecución de otros por estar estrechamente relacionados (por ejemplo, minimizar el tiempo de producción puede provocar una reducción del costo de producción).

El objetivo principal de los problemas de programación por lo general depende de la disponibilidad e interés empresariales.

BIBLIOGRAFÍA

[1] CHASE, Richard; AQUILANO, Nicholas; JACOBS, Robert: Administración de Producción y de Operaciones, 885 páginas, Mc Graw-Hill, Bogotá., 2001.
[2] PINEDO Michael. Planning and Scheduling in Manufacturing and Services. 506 pages, Springer Series in Operations Research, New York, USA, 2005.
[3] SCHROEDER, Roger.: Administración de Operaciones, 856 páginas, Mc Graw-Hill, Bogotá., 1997.
[4] SIPPER, Daniel; BULFIN, Robert: Planeación y control de la producción. 657 páginas Mc Graw-Hill, México.1998.