CAPÍTULO 1

PROGRAMACIÓN DE ACTIVIDADES (Scheduling)

La *programación de actividades* en un sistema productivo se refiere al proceso de organizar, elegir y dar tiempos al uso de recursos para llevar a cabo todas las actividades necesarias, para producir las salidas deseadas en los tiempos deseados, satisfaciendo a la vez un gran número de restricciones de tiempo y relaciones entre las actividades y los recursos." (Morton y Pentico (1993). Un programa especifica el tiempo en el que comienza y termina cada trabajo en cada máquina. Una secuencia es un orden simple de trabajos, 3-1-2 indica que el trabajo 3 se hace primero, el 1 es el segundo y el 2 es el último. De esta forma la secuencia determina los tiempos de inicio y terminación y, por lo tanto, determina la programación.

Los datos que se utilizan para el estudio de la *programación de actividades*, son datos conocidos con certeza y con antelación, o lo que es igual se manejan datos determinísticos. Un trabajo consiste en varias etapas de proceso, llevadas a cabo en diferentes máquinas o centros de procesamiento, determinados por los requerimientos de cada etapa. Las decisiones en la *programación de actividades* deben resolver el *cuando,* y el *cual* en orden que las medidas de desempeño tales como la tardanza, inventario en proceso, y tardanza máxima (*makespan*) sean minimizados.

En la programación de actividades se identifican algunos tipos de problemas diferenciados entre sí por la naturaleza del flujo de materiales a través de los recursos disponibles (máquinas), estos se describen a continuación:

- **Problemas con una sola máquina**

Para los ***problemas en una sola máquina***, deben procesarse en ella todos los trabajos. La máquina puede procesar a lo más un trabajo a la vez. En este tipo de problema solo se busca una secuencia que dicta en qué orden se introducirán los trabajos a dicha máquina.

- **Problemas con Máquinas Paralelas**

Varias máquinas que pueden realizar el mismo tipo de procesamiento se llaman ***máquinas paralelas***. Un trabajo se puede procesar en cualquiera de las máquinas, y una vez procesado por cualquiera de ellas, queda terminado. A menos que se diga lo contrario, se supone que todas las máquinas son idénticas. El tiempo para procesar un trabajo en una de varias máquinas idénticas es independiente de qué máquina lo haga.

- **Problemas con un Sistema de Producción continua**
  (Flow Shop Problems)

Un taller de producción continua contiene máquinas diferentes. Cada trabajo debe procesarse en cada máquina exactamente una vez. Más aún, todos los trabajos siguen la misma ruta; esto es, deben visitar las máquinas en el mismo orden.

- **Problemas con un Sistema de Producción intermitente**
  (Job Shop Problems)

Un taller de producción intermitente es más general que el de producción continua:
Cada trabajo o pieza a realizar está formado por un conjunto de operaciones , cada una de las cuales deben ser procesadas en una determinada máquina durante un tiempo preestablecido de antemano. Cada trabajo tiene su propio flujo, es decir la secuencia en que visitan cada una de las máquinas no tiene que ser igual al de los otros.

- **Problemas con un sistema de Producción intermitente flexible**
  (Flexible Job Shop Problems)

Este tipo de problemas son una variante de los problemas para un sistema de producción intermitente. La diferencia se basa en que una actividad tiene asignado un grupo de máquinas en las cuales podrá ser procesada y no solo una máquina en específico. Por lo tanto, además de una secuenciación se tiene que realizar un proceso de asignación.

Este tipo de problema se ajusta de manera adecuada a los problemas reales de casi todos los tipos de manufactura y en todos los sectores, incluyendo el que concierne a este trabajo.

1.1. El problema de la programación de actividades para un sistema de producción intermitente flexible (FJSSP)

El problema de secuenciación de FJSS presenta una serie de variantes dependiendo de la naturaleza y el comportamiento; tanto de las operaciones como de las máquinas. Una de las variantes más difíciles de plantear, debido a su alta complejidad computacional, es aquella en donde las tareas son dependientes y las máquinas son diferentes. En esta variante cada trabajo presenta una lista de operaciones que lo preceden y para ser ejecutado debe de esperar el procesamiento de dicha lista en su totalidad. A esta situación hay que agregarle la característica de heterogeneidad de las máquinas: cada tarea demora tiempos distintos de ejecución en cada máquina. El objetivo más extensamente utilizado en la planeación de la producción es minimizar el tiempo acumulado de ejecución de las máquinas conocido en la literatura como *makespan*.

En un principio, existen un número infinito de programas posibles para un FJSSP debido a que un tiempo ocioso cualquiera puede ser insertado entre dos operaciones. Alternativamente se pueden acomodar las operaciones a la izquierda (en el eje del tiempo), para hacer el programa lo más compacto posible.

El problema de FJSS suele ser definido por las siguientes condiciones [3]:

- Hay *n* trabajos con subíndice *i*, y estos trabajos son independientes entre ellos.
- Cada trabajo *i* tiene una secuencia de operación, denotada por J_i
- Cada trabajo consiste de una o más operaciones $O_{i,j}$
- Cada secuencia de operación esta ordenada por un juego de operaciones $O_{i,j}$
- Hay *m* máquinas con subíndice *K* (La k-ésima máquina es denotada por m_k)
- Para cada operación $O_{i,j}$ hay un juego de máquinas capaces de cumplir con la función objetivo. Este juego de máquina es denotado por $U_{i,j}$
- El tiempo de procesamiento P_i de una operación $O_{i,j}$ en una máquina K es predefinido y mayor que cero.

Restricciones Generales.

- Cada operación no puede ser interrumpida durante el cumplimiento de la ejecución de la misma
- Todas las máquinas están disponibles en el tiempo t=0
- Cada máquina K, no puede procesar más de una operación simultáneamente.

1.2. ***Técnicas utilizadas para la resolución del FJSSP***

Las técnicas para resolver un problema de programación de actividades para un sistema de producción intermitente flexible (FJSSP) pueden ser clasificadas en dos categorías: soluciones con planteamientos jerárquicos y soluciones con planteamientos integrados.

a) *Planteamiento jerárquico* trata de resolver el problema por descomposición de una secuencia en sub-problemas, y con ello reducir la dificultad. Esta descomposición consiste en realizar la asignación de cada actividad a la máquina en la cual se procesará y posteriormente realizar la secuenciación, como si se tratara de un ejercicio de JSS. Como ejemplo se puede realizar la asignación utilizando reglas de despacho, y luego resolver el resultado de JSSP usando algún algoritmo como la búsqueda Tabú.
b) Por otra parte el *planteamiento integrado*, consiste en realizar de forma integrada la asignación y la secuenciación, lo cual implica una mayor dificultad para su resolución, pero que también arroja mejores resultados.

La parte que corresponde a la asignación de las actividades para el FJSSP se puede atacar como se dijo antes con reglas de despacho, o con cualquier técnica propia de un problema con máquinas paralelas. En cuanto a la secuenciación, las técnicas no son muy diferentes de la forma en que se atacan los problemas de JSS, y las más comunes se describen a continuación.

Algoritmo Heurístico de Despacho

El enfoque más común para la producción intermitente es usar reglas de despacho con prioridades. La idea básica es programar una operación de un trabajo tan pronto como se pueda; si hay más de un trabajo que espera ser procesado por la misma máquina, se programa el que tiene la mayor prioridad.

La prioridad, con frecuencia expresada en forma numérica, se utiliza para determinar la secuencia en que deben procesarse las órdenes. Las reglas descritas a continuación son tal vez las más comunes, aunque hay muchas variaciones y combinaciones de éstos métodos.

La lista que aparece en el cuadro 1.1 proporciona una buena visión general de las reglas básicas y sus objetivos.

Existen otras prioridades y en la literatura se encuentran algunas pequeñas variaciones en las definiciones. La prioridad usada depende de lo que se quiere de la medida del programa.

Cuadro 1.1 Reglas de despacho tradicionales y su descripción

REGLA	OBJETIVO
PEPS (Primeras entradas primeras salidas)	Producir órdenes en la secuencia en la que llegan al centro de trabajo. Esta regla de "justicia" es especialmente apropiada en las organizaciones de servicio donde la mayor parte de los clientes con frecuencia necesita o desea la terminación del servicio tan pronto como sea posible.
MTP-MTO (El menor tiempo de proceso)	Producir órdenes en razón inversa al tiempo requerido para procesarlas en el departamento (primero el tiempo más breve). Esta regla da como resultado menor producción en proceso, menor promedio para terminación del trabajo y menor retraso promedio. A menos que esta regla se combine con la fecha de entrega o la regla de tiempo ocioso, los trabajos (ordenes) con tiempos largos de procesamiento pueden estar listos muy tarde.
MTPT, tiempo remanente del tiempo de procesamiento total mas corto.	Producir órdenes en razón inversa al tiempo remanente del procesamiento total; la lógica de esta regla es similar a la precedente. Cumple con objetivos similares cuando la mayor parte de los trabajos siguen un proceso común.
FEMP, fecha de entrega mas próxima	Producir órdenes con la primera fecha de entrega más inmediata. Esta regla trabaja bien cuando los tiempos de procesamiento son aproximadamente los mismos.
MO, Menos operaciones	Producir primero las órdenes con menos operaciones remanentes. La lógica de esta regla es que menos operaciones abarcan menor tiempo de cola y, como resultado, la regla reduce la producción promedio en proceso, el tiempo de obtención de la producción y el retraso promedio. No obstante los trabajos con un número relativamente grande de operaciones pueden abarcar tiempos excesivamente largos.
ST, Tiempo de holgura	Producir primero la orden con menor tiempo de holgura y continuar la secuencia en orden ascendente de tiempos de holgura. El tiempo de holgura es igual a la fecha de entrega menos el tiempo restante de procesamiento. Esta regla favorece el objetivo de fecha de entrega. El tiempo de holgura restante por operación es una variante de esta regla.
CR, Tasa Crítica	Para órdenes todavía no retrasadas (vencidas) producir primero las órdenes con menor tasa crítica. La tasa crítica es igual a la fecha de entrega menos la fecha actual, dividida entre el tiempo restante de conducción de la producción.

Algoritmo heurístico de Enfriamiento Simulado (Simulated annealing)

El *enfriamiento simulado* (Kirkpatrick, Gelatt y Vecchi, 1983; Cerny, 1985) se basa en una analogía con el proceso físico de recocido, en el cual se forma una estructura reticular en un sólido calentándolo por encima de su temperatura crítica seguido de un enfriamiento lento hasta llegar a un estado de baja energía.

Desde el punto de vista de la optimización combinatoria, el *enfriamiento simulado* es un algoritmo aleatorio de búsqueda local. Las soluciones vecinas de mejor costo son siempre aceptadas e incluso se aceptan las soluciones de peor costo, aunque con una probabilidad que decrece gradualmente durante el transcurso de la ejecución del algoritmo. Esta reducción de la tolerancia o probabilidad de aceptación es controlada por un conjunto de parámetros cuyos valores son determinados por un esquema de enfriamiento prescrito.

El *enfriamiento simulado* ha sido aplicado con considerable éxito. Su naturaleza aleatoria permite convergencia asintótica a la solución óptima bajo condiciones moderadas. Desafortunadamente, la convergencia requiere típicamente de tiempo exponencial, convirtiendo el *enfriamiento simulado* en impráctico como instrumento para procurarse soluciones óptimas.

Algoritmo Heurístico Búsqueda Tabú (Tabu Search)

La *Búsqueda Tabú* (Glover, 1990) combina un algoritmo determinístico de avance iterativo con la posibilidad de aceptar soluciones que incrementen el costo. De esta manera, la búsqueda es dirigida fuera de óptimos locales y otras partes del espacio de soluciones pueden ser exploradas. Cuando es visitada una nueva solución, es considerada un vecino legítimo de la solución actual aún cuando empeore el costo.

El conjunto de vecinos legítimos queda representado por una lista tabú, cuyo objetivo es restringir la elección de soluciones para prevenir el regreso a puntos recientemente visitados. La lista tabú es actualizada dinámicamente durante la ejecución del algoritmo y define soluciones que no son aceptables en unas pocas siguientes iteraciones. Sin embargo, una solución de la lista tabú puede ser aceptada si su calidad, en algún sentido, elevada lo justifica. En este caso se establece que se ha alcanzado cierto nivel de aspiración *(aspiration level).*

La *Búsqueda tabú* ha sido aplicada en una gran variedad de problemas con considerable éxito ya que presenta un esquema de gran adaptabilidad que se ajusta a los detalles del problema considerado. Por otro lado, existen vagos conocimientos teóricos que guíen ese proceso de ajuste y cada usuario debe recurrir a la información práctica disponible y a su propia experiencia.

OPT/TOC

Un enfoque para atacar el JSSP está dado por una visión integral de la Planeación y Control de la Producción, la cual contiene una componente técnica y un concepto administrativo. La componente técnica de este enfoque es un programador de recursos restricciones **(cuello de botella)** conocido como *optimized production technology* (OPT) o *tecnología de producción optimizada*. El concepto administrativo se llama *teoría de restricciones* (TOC), y abarca no solo la programación de los trabajos en un taller, sino objetivos como lo son aumentar la producción, reducir el inventario y disminuir los gastos operativos.

El OPT es en esencia un sistema de retroalimentación en el que una vez identificados los cuellos de botella se realiza la programación de los trabajos asignando una secuencia para este recurso, en seguida se programan las operaciones hacia atrás, desde el recurso en

cuestión hasta el punto de despacho de la materia prima y, después, hacia adelante hasta el envío.

El problema para obtener la secuencia de actividades escogida para ser procesada en la máquina cuello de botella es un problema de optimización combinatoria del tipo NP-Complejo, en el cual se puede utilizar un heurístico de despacho con diferentes reglas de prioridades o algún otro tipo de algoritmo heurístico.

Algoritmos genéticos

La técnica de los Algoritmos genéticos, inicialmente desarrollada por John Holland en el año de 1975 cobró fuerza para la resolución de problemas del tipo NP-Complicados. La técnica solo fue utilizada para la resolución de problemas de programación de actividades hasta finales de la década de los ochentas y principios de los noventas, *Davis*, *L*. (*1985*). *Job shop scheduling with* genetic algorithms*,* Falkenauer E. y S Bouffouix (1991), *A genetic Algorithm For Job shop*. El buen desempeño de los algoritmos genéticos para este tipo de problemas permitió que se incursionara en un problema más complejo, que es una particularidad del problema de JSS. El problema para la programación de actividades para un sistema de producción intermitente flexible (FJSSP) ha sido tratado con distintas variantes de los algoritmos genéticos. Algunas de las más recientes la desarrollada por Ho, N. B. y Tay, J. C., 2008. *"Evolving dispatching rules using genetic programming for solving multiobjective flexible jobshop problems."*