8

# PROGRAMACIÓN DE LA PRODUCCIÓN EN UNA EMPRESA ALIMENTARIA, UTILIZANDO LA PARALELIZACIÓN EN LOS CÁLCULOS

JOAQUÍN BAUTISTA
RAMÓN COMPANYS
ALBERT COROMINAS
MANUEL MATEO
RAFAEL PASTOR
EUGENI ROURES
UNIVERSITAT POLITECNICA DE CATALUNYA. BARCELONA

**Resumen:**

Una empresa del sector cárnico, que produce bajo pedido, disponía de un algoritmo para programar la producción. La complejidad y variedad en los recursos y los procesos, además del carácter perecedero del material, dificultaba alcanzar respuestas satisfactorias en tiempos aceptables.
Con el fin de reducir los tiempos de cálculo, se consideró el uso de la paralelización. El algoritmo diseñado calcula en paralelo la asignación de operaciones a los diferentes recursos productivos, y se ha desarrollado en el marco de los proyectos PACOS *(Parallel Computing for Spain)*.

Palabras clave: programación de la producción, cálculo paralelo, PVM.

## 1. Introducción

En este artículo se expone una aplicación práctica en el campo de la planificación de la producción en una planta de Hesperia de Alimentación S.A., empresa del sector cárnico, que produce toda clase de productos derivados de la carne. El proyecto ha reunido a cuatro socios: el usuario final (Hesperia de Alimentación S.A.),una empresa dedicada a software (Neosystems Informática S.A.) y dos departamentos de la Universitat Politècnica de Catalunya,el Departamento de Estadística e Investigación Operativa y el Departamento de Organización de Empresas. Esta colaboración se ha llevado a cabo en el marco de un proyecto ESPRIT, llamado Sirius, impulsado por la Comunidad Europea e incluido en el programa PACOS (*Parallel Computing for Spain*).

El objetivo principal de este proyecto se centraba en la reducción de los tiempos de ejecución del entonces vigente algoritmo de carga de máquinas, para lo cual se propuso adaptar el algoritmo secuencial a una nueva versión que realizase cálculos en paralelo. Además, se incluían otros objetivos como: establecer unas reglas de prioridad en el lanzamiento y en la secuenciación de las operaciones, independizar el algoritmo de la plataforma donde debía ejecutarse o facilitar la interacción entre usuario y algoritmo mediante nuevas técnicas de representación gráfica. Estos objetivos respondían a la cuestión semanal a que se enfrentaban los planificadores de esta empresa: cómo fabricar los pedidos requeridos a mínimo coste, para lo cual debían determinar qué tareas realizar en cada uno de los recursos productivos de su fábrica y cuándo.
La situación descrita puede enmarcarse en el ámbito de los problemas de "job-shop", aunque con diversos matices que incrementan su dificultad de tratamiento respecto al caso general. Basándose en anteriores trabajos realizados por algunos de los autores en el campo de la programación y secuen-

ciación de órdenes de producción, y adaptando a este caso los procedimientos de resolución de los problemas "job-shop"[2], se ha comprobado cómo los problemas de programación de operaciones en este tipo de industrias se pueden abordar con resultados satisfactorios utilizando procedimientos heurísticos.

El artículo está organizado como se indica a continuación. En la sección 2 se describen las características del sistema productivo. En la sección 3 se presenta el algoritmo utilizado en la programación de la producción y en la siguiente sección se explica la incorporación del paralelismo en el algoritmo y sus consecuencias. Ya en la sección 5, se exponen algunos resultados al aplicar el nuevo algoritmo sobre datos reales. Finalmente, en la sección 6 se expresan algunas conclusiones obtenidas a partir de este proyecto.

## 2. El sistema de producción

En primer lugar, se ofrece una descripción del sistema de producción sobre el cual debe realizarse la programación, para comprender correctamente las particularidades tratadas por el algoritmo implementado.

La fábrica, ubicada en Cataluña, produce una gran variedad de productos derivados de la carne:jamones, salchichas, fiambres, etc. (figura 1). Su planta de producción se compone de diversas líneas especializadas, aunque se utilizan también recursos comunes, lo que obliga a considerarlos todos a la vez en el proceso de programación.

Los recursos productivos actualmente tratados son del orden de 130, sólo máquinas, cuya disponibilidad se controla mediante "calendarios" que indican sus períodos activos de fabricación. De todas formas, el algoritmo se ha desarrollado para optimizar no tan sólo máquinas, sino también el personal requerido.

Los recursos productivos que realizan las operaciones de transformación de la carne y los demás ingredientes en productos finales pueden agruparse básicamente en tres tipos:

- los recursos propios de la planta y capaces de realizar una única actividad;
- llos recursos "compartidos", también propios, y capaces de realizar más de una de actividad simultáneamente;
- ly finalmente, los recursos productivos externos, que se hallan físicamente fuera de la planta.

Cabe destacar que la asignación de actividades simultáneas a un recurso compartido requiere una total compatibilidad entre todas ellas.

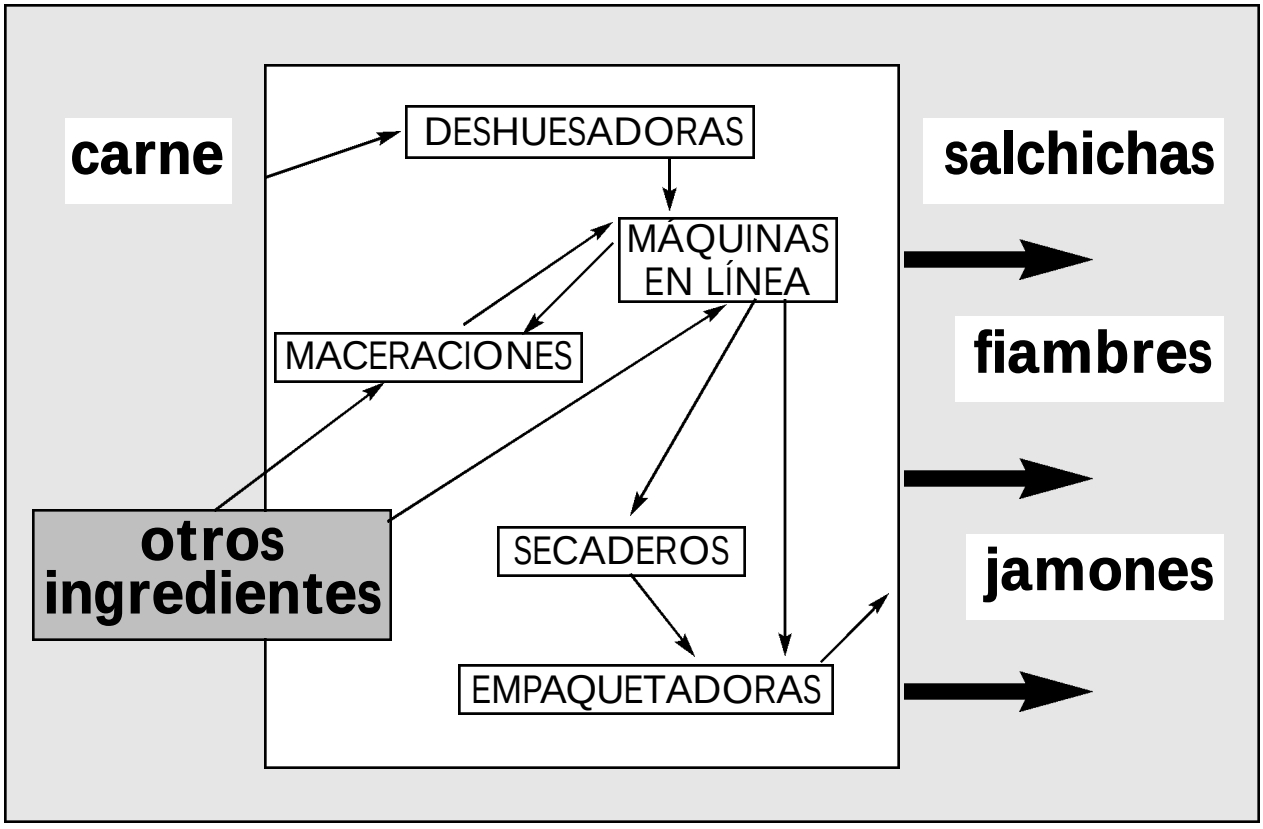



Figura 1. Proceso productivo en la planta

La elaboración de cada producto sigue uno de los procesos productivos definidos (en este momento, unos 30). Los procesos de producción presentan gran diversidad en su número de actividades (desde cuatro o cinco hasta unas veinte) por la amplia variedad de productos. Las líneas de producción en esta fábrica, como en otras, introducen relaciones de precedencia entre actividades, parte indivisible del proceso productivo; además, al existir recursos con capacidad muy limitada y actividades muy largas, éstas deben realizarse simultáneamente en varios recursos. En esta planta se denomina modularidad a la división de la cantidad total de un producto en diversas partes, para que se pueda realizar una actividad a la vez en diversos recursos.El reparto de una cantidad en dichos módulos es una tarea compleja; véase el ejemplo de la cocción de 2.000 Kg. al final de este punto.

Presentados los recursos disponibles y el proceso de transformación de la carne y las demás materias primas en los productos finales, a continuación se tratan las órdenes de producción, cuyo número es actualmente de unas 80 semanales, aunque se espera incrementarlo por el uso eficiente de los recursos, al aplicar la nueva manera de programar.

Las órdenes de fabricación se descomponen en actividades, sus componentes básicos,que pueden dividirse en "dinámicas" y "estáticas"o modulares,términos usados en la fábrica para referirse a la obra en curso. En las actividades dinámicas, la duración de la actividad es directamente proporcional a la cantidad a fabricar; en cambio, en las actividades estáticas o modulares, realizables en varios recursos llamados módulos, la duración depende básicamente de la capacidad física y de la cantidad aún disponible para fabricación en cada recurso.

Como una actividad puede asignarse habitualmente a varios recursos de la planta, todos con una misma función en el sistema productivo, el algoritmo trabaja con grupos productivos, listas de recursos agrupados de acuerdo con las funciones que puedan desempeñar. Cada una de las operaciones se relaciona con uno de estos grupos, de entre los cuales se debe escoger un recurso para realizar dicha actividad.

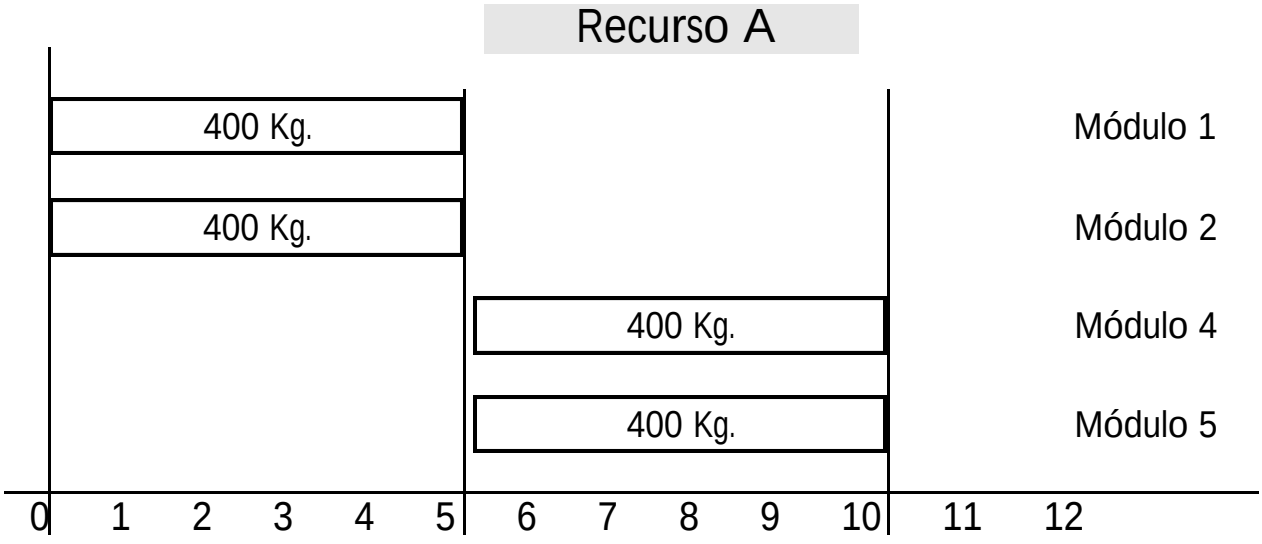
El aspecto clave del algoritmo de programación es la elección del recurso del grupo productivo al cual asignar una actividad. El nexo de unión entre actividades y recursos productivos son aspectos tecnológicos, como pueden ser:

- Las capacidades totales y de cada compartimento de los recursos.
- Los límites de temperatura en algunas actividades.
- Las dimensiones de los recursos.
- Los tiempos para efectuar una actividad con un recurso.

Los aspectos tecnológicos son condiciones para comprobar la total compatibilidad entre las características tecnológicas de un recurso y las requeridas por una actividad que se le asigna, o bien la compatibilidad entre dos o más actividades que compartan el mismo recurso durante un cierto período de tiempo.

Imagínese, por ejemplo, que se quieren cocer 2.000 Kg. de un producto, cuyo tiempo de cocción en condiciones normales son 5 horas y que se dispone de dos recursos (A y B) para realizar esta actividad:

- Recurso A: horno con capacidad para 800 Kg. de producto, separado en dos compartimentos de 400 Kg. cada uno.
- Recurso B: horno con una sola cavidad con capacidad para 150 Kg.de producto, aunque la cocción dura solamente 4 horas.



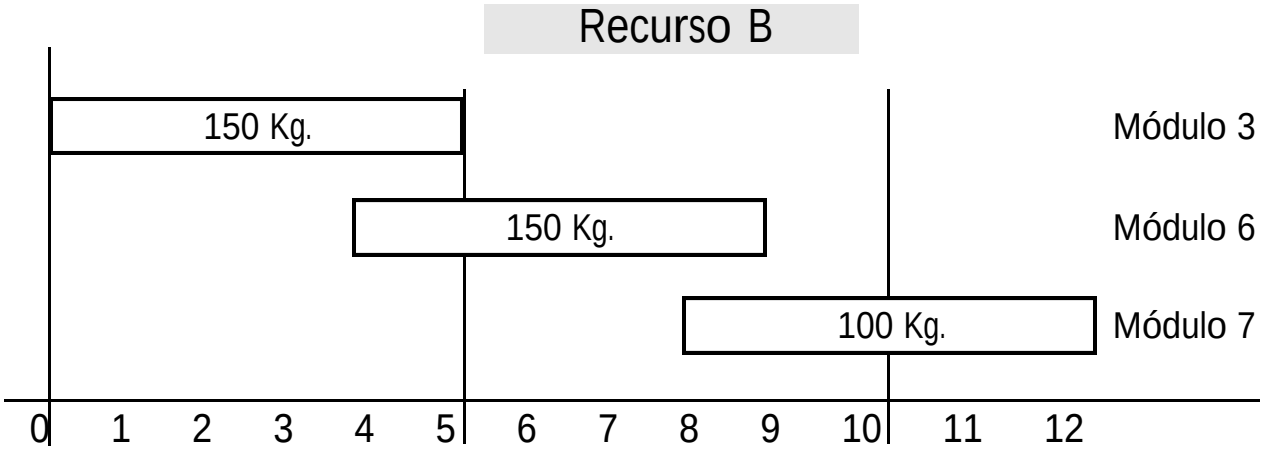



Figura 2. Ejemplo de actividad modular

Obsérvese (figura 2) como en este caso, los 2.000 Kg.a tratar se reparten en módulos de 400, 150 y 100 Kg. respectivamente, hasta completar la cantidad total.

### 3. El algoritmo de carga de máquinas

El objetivo principal del algoritmo de carga de máquinas es la programación de las órdenes de fabricación, aunque no menos importante, es lograrlo en un tiempo lo más corto posible.
Algunas características que la programación en los recursos de la fábrica considera son:

- la asignación de una actividad o un módulo de una actividad dentro de una estructura de producción de capacidad limitada,
- las órdenes de fabricación tratadas globalmente, siguiendo unas prioridades específicas en una primera etapa de la programación,
- las actividades de una orden asignadas en secuencia a los recursos disponibles para cada caso,
- posibles incompatibilidades tecnológicas entre las características de los recursos y de las actividades,
- las dependencias entre la fabricación de productos intermedios y finales.

El algoritmo de carga de máquinas tiene en cuenta los elementos presentados en la sección anterior y proporciona los resultados, la programación de la producción, impresos en forma de boletines de trabajo. El personal de la planta los utiliza para controlar la producción y anotar las incidencias, que más tarde serán introducidas para una próxima ejecución del algoritmo. Los datos se manejan en memoria durante toda la programación para optimizar tiempo de cálculo.

PRINCIPALES ETAPAS EN EL ALGORITMO:

El algoritmo de carga de máquinas original consta de cinco partes:

1. Preparar los datos para los cálculos de programación de las órdenes de fabricación. Obtener las precedencias entre actividades, para cada orden de producción a incluir en el programa.
2. Actualizar la estructura productiva para iniciar la programación: determinar qué recursos están ocupados por actividades ya programadas y cuándo estarán disponibles dichos recursos.
3. Determinación de la secuencia de carga de las actividades de las órdenes de fabricación: establer una secuencia de carga entre órdenes y entre actividades en una misma orden, para realizar la programación en la siguiente fase.
4. Programación de las actividades incluidas en las órdenes de producción a planificar: determinar en qué recurso se realizará cada actividad.El proceso consiste en evaluar cada actividad o módulo en todos los recursos productivos disponibles, y según los respectivos resultados, escoger el recurso más apropiado.
5. Actualizar los datos de la carga de máquinas programada.

El algoritmo propuesto es del tipo *"dispatching"* [2], en que el conjunto de operaciones elegibles en cada momento está constituido por el subconjunto de las operaciones con sus precedentes ya programadas, es decir, conocidos los instantes de inicio y de finalización. Esto implica que no es necesario

tomar decisiones sobre las asignaciones en el mismo orden en que serán ejecutadas posteriormente.
El algoritmo de programación incorpora intervalos de carga, siendo un intervalo de carga una parte del horizonte temporal de programación (véase figura 3). En alguna ocasión podría considerarse un único intervalo, si se toma globalmente todo el horizonte.
El intervalo de carga usado en este método de *"dispatching"* sirve para sincronizar las actividades de diferentes órdenes de producción y equilibrar el tiempo total de fabricación entre ellas. Si las órdenes de producción se cargasen de principio a fin, las primeras podrían ocupar algunos recursos principales y las últimas, sufrir demoras considerables, con lo que el alimento podría deteriorarse. Esto, por supuesto, comportaría abandonar la programación en curso y empezar de nuevo la carga de todas las actividades. Por tanto, esta actitud prudente en el diseño del algoritmo intenta evitar al máximo esta situación.
Todas las órdenes implicadas en un programa de fabricación se van cargando progresivamente en los recursos. Así, todas pueden retrasarse debido a tiempos de espera entre actividades. Si se equilibran los diversos retrasos, será mucho más difícil que alguna orden se demore hasta el punto de sobrepasar su fecha de deterioro como alimento.

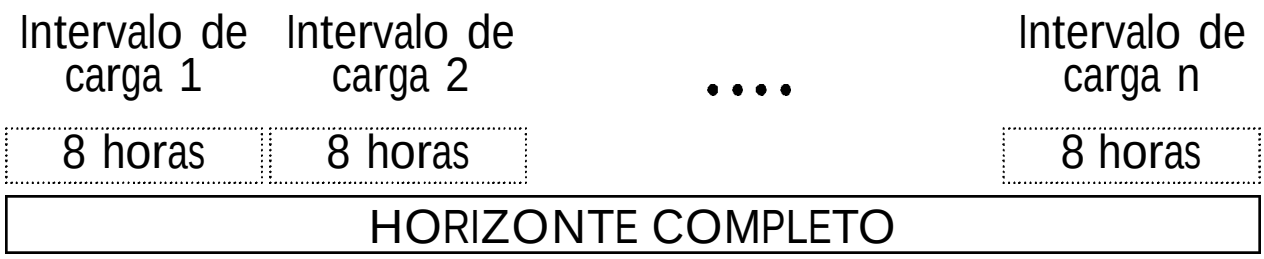



Figura 3. Intervalos de carga

Para reducir el tiempo total de cálculo se ha trabajado en dos direcciones: por un lado, en la estructuración adecuada de los datos necesarios para los cálculos; y por otro, en una redefinición del algoritmo previo aprovechando las prestaciones del cálculo en paralelo.

## 4. El nuevo algoritmo en paralelo

De las partes que componen el algoritmo, solamente la cuarta incorpora el cálculo en paralelo, ya que consumía más tiempo en el algoritmo secuencial de partida. Esta parte consiste en la selección de una actividad de una orden de fabricación y la evaluación numérica de su posible asignación a cada uno de los recursos donde se pueda realizar. Estas evaluaciones son susceptibles de ejecutarse en paralelo porque un programa de fabricación está formado por múltiples órdenes,a la vez integradas por numerosas actividades.
Como resultado de la evaluación de una actividad en algún recurso en que se pueda programar, se asocia una nota a dicho recurso. Esta nota depende de factores como la ocupación del recurso, el tiempo para realizar la actividad o la cantidad de producto a fabricar. Una vez evaluados todos los recursos, se escoge aquél con una mejor nota.
Para lanzar en paralelo estas tareas, lo primero es seleccionar el software que permita llevarlo a cabo [5]. En este caso, se eligió PVM (*Parallel Virtual Machine*) [1] para desarrollar el algoritmo paralelo, porque permite trabajar con la llamada máquina virtual, que puede consistir en una red de procesadores UNIX heterogéneos o en una única máquina con varios procesadores paralelos (de memoria distribuida). PVM es un software de libre distribución, accesible desde Internet: <http://www.epm.ornl.gov/pvm/pvm_home.html>.
Un algoritmo puede lograr resultados similares tanto en un sistema multi-CPU como en un sistema de múltiples máquinas (cluster).Mientras en una situación el tiempo de transmisión de datos es menor, el número de procesadores está limitado a la disponibilidad de la máquina; en la otra, en cambio, se permite trabajar con cualquier número de procesadores. PVM también tiene estipuladas las acciones con aquellos tipos de mensajes intercambiados más a menudo [1].
Uno de los indicadores de la eficiencia de un algoritmo paralelo es el speed-up, que indica la mejora en tiempos de ejecución al pasar de una máquina que realiza cálculos en serie a otra con varios procesadores [6]. El "speed-up" es el cociente entre el tiempo de ejecución del algoritmo en su versión secuencial y el tiempo de ejecución del algoritmo en su versión paralela, con **n** procesadores. Este valor debería aproximarse idealmente a **n**. Como conclusión, un algoritmo será tanto mejor cuanto más próximos sean los valores de speed-up al número

de procesadores en la máquina virtual.
Otro aspecto a considerar al construir aplicaciones en PVM es el modelo de la estructura de cálculos. Hay dos estructuras típicas [5] en aplicaciones donde interviene el paralelismo: el modelo SPMD (Single Process Multiple Data), en que todos los procesos en paralelo son idénticos, aunque cada uno trabaja sobre un conjunto de datos diferentes; y el modelo Master/Slave (maestro y esclavos), en el cual un conjunto de procesos esclavos trabaja en una tarea para uno o varios maestros.
En el presente proyecto, se optó por el modelo Master/Slave, con un conjunto de procesos esclavos llevan a cabo tareas para un solo maestro, que las distribuye [4].Así pues, antes de iniciar el desarrollo del nuevo algoritmo, debían definirse las funciones del proceso maestro y de los procesos esclavos.
Las funciones del proceso maestro básicamente son:

- Identificar y controlar los procesadores que pertenecen a la máquina virtual.
- Repartir entre los procesadores de la máquina virtual todas aquellas tareas de evaluación necesarias.
- Recibir desde los procesos esclavos los resultados de las evaluaciones.
- Escoger, de acuerdo con las evaluaciones recibidas, el recurso productivo mejor considerado para realizar cada actividad o módulo.

Las funciones de un proceso esclavo, por su parte, son:

- Recibir información sobre una actividad y un recurso productivo para evaluar su posible asignación.
- Hallar un período de tiempo de disponibilidad en el recurso productivo para cargar la actividad.
- Obtener el resultado de una evaluación propuesta para compararlo posteriormente con otras opciones.
- Enviar al proceso maestro el resultado calculado.

Al introducir el paralelismo en este algoritmo, algunos condicionantes evitan que cualquier conjunto de cálculos pueda realizarse en paralelo. Estos aspectos plantean que la estructura de los cálculos se base en los llamados niveles de paralelismo, asociados a las siguientes circunstancias:

1. Primer nivel: paralelismo en las evaluaciones de actividades en cada uno de los recursos. Las actividades en una cierta orden de fabricación deben cargarse secuencialmente. Por lo tanto, los cálculos referidos a una actividad o módulo no pueden comenzar si antes no ha sido cargado el módulo o actividad anterior.
2. Segundo nivel:paralelismo en las evaluaciones de actividades y de órdenes en cada uno de los recursos. Las actividades de diferentes órdenes pueden ocasionar conflictos si los cálculos se refieren al mismo recurso. Esta posibilidad comporta asegurar que la información está actualizada cuando se asigna una actividad.

PRIMER NIVEL DE PARALELISMO

Si sólo se considera el primer nivel de paralelismo, las órdenes se cargan según una secuencia previamente fijada. Por otro lado, y debido a las relaciones de precedencia, las actividades de cada orden se cargan ordenadamente. Así pues, en el curso de la ejecución del algoritmo los procesos esclavos evalúan, paralelamente, los recursos candidatos a ser ocupados por una actividad de la orden en curso.
El proceso maestro, pues, debe esperar los resultados de todas las evaluaciones de una actividad sobre cada posible recurso. Una vez recogidos, se selecciona el recurso más apropiado, la actividad se carga y pueden enviarse las evaluaciones involucradas en la siguiente actividad a los procesos esclavos.
Analizando los valores del speed-up, se observó que eran relativamente bajos: la introducción del cálculo en paralelo era positiva para algunas actividades que podían asignarse a diversos recursos, pero en caso de un número de recursos escaso el algoritmo perdía eficiencia. Por lo tanto, el grado de paralelismo en los cálculos debía complementarse, y como consecuencia, incrementarse el speed-up.

SEGUNDO NIVEL DE PARALELISMO

Las actividades dentro de cada orden de fabricación están ordenadas. Las órdenes siguen una secuencia, pero no es necesario finalizar las anteriores antes de empezar a cargar actividades de una orden.Así pues, en el curso de ejecución del algoritmo los procesos esclavos pueden evaluar, en paralelo, recursos candidatos para actividades de varias órdenes.
El proceso maestro no debe esperar los resultados de todas las evaluaciones de una actividad antes de enviar nuevas evaluaciones de otras actividades a los procesos esclavos,a menos que esta actividad se deba programar ya en el siguiente intervalo de carga.
Este nuevo modelo de evaluación de órdenes en paralelo aumentaba la eficiencia porque las diversas actividades implicaban mayores posibilidades de cálculo simultáneo. Y esta eficiencia podía incremen-

tarse incluso más al estar programando actividades en varios puntos de la estructura de producción, y evitar posibles repeticiones en los cálculos,como se explica seguidamente.

### CONFLICTOS ENTRE ÓRDENES DE PRODUCCIÓN

En la aplicación del algoritmo pueden surgir conflictos entre órdenes de producción en paralelo cuando algunas de sus actividades compitan por el mismo recurso. Cuando una actividad vaya a asignarse a un recurso, deberá comprobarse que ninguna actividad se haya cargado en dicho recurso desde su evaluación hasta ese momento. Si el contexto que afecta al recurso ha cambiado, la evaluación de carga podría no ser correcta.

Ante este problema, se pueden tomar diversas alternativas [3], entre las cuales se consideraron dos: la primera,basada en un sistema de bloqueos de los recursos; la segunda,basada en un sistema de recálculo de las evaluaciones. Finalmente, se optó por esta última opción porque la relación de los tiempos de cálculo (speed-up) no mejoraba sustancialmente respecto la utilización de bloqueos y además se corría el riesgo de un bloqueo general del sistema, posible al ejecutar acciones paralelas dependientes entre sí [3].

### SISTEMA DE RECÁLCULOS

El sistema de recálculos, causado por la existencia de los dos niveles de paralelismo, permite evaluar varias actividades, pertenecientes a más de una orden de fabricación, sobre un mismo recurso. Cualquier evaluación puede ser calculada en algún momento, aunque siempre con la posibilidad de tener que recalcularse más tarde. El recálculo tendrá lugar si un recurso es el más adecuado para dos actividades diferentes que se están programando.

Analizados los resultados, se observó como el speed-up había mejorado respecto al único nivel de paralelismo, aunque la presencia de intervalos de carga quizá no permitía el máximo grado de paralelismo. El número de recálculos no era excesivo y, además, era inferior normalmente al número de bloqueos.

## 5.Resultados

El algoritmo secuencial original ha tenido que adaptarse para ser eficiente desde el punto de vista del paralelismo, es decir, que la cantidad de cálculos simultáneos repercuta en una reducción significativa del tiempo de ejecución. Ello obliga a permitir evaluaciones simultáneas de diversas actividades sobre un mismo recurso, lo que conduce a veces a contratiempos, recálculos en las evaluaciones. Esta cantidad depende del número y del tipo de procesadores en que se ejecute el algoritmo. Si se analizan los tiempos, se observa que esta incidencia apenas perjudica la duración total.

El código del nuevo algoritmo se ha escrito en lenguaje de programación C, y para el cálculo en paralelo se utilizan aquellas rutinas de paralelización propias del software PVM (*Parallel Virtual Machine*).

### DURACIONES DE LA EJECUCIÓN DEL NUEVO ALGORITMO

Reducir la duración de la ejecución del Algoritmo de Carga de Máquinas de varias horas hasta no más de 2 horas, y así responder ágilmente a incidencias en la planta, se ha superado ampliamente. Introducido el cálculo en paralelo para la programación de la producción, las duraciones del algoritmo han pasado a minutos, debido a varios factores.

Para empezar, se depuró la gestión de datos reduciendo sustancialmente el tiempo en operaciones no directamente vinculadas a la programación de la producción,y se estructuró la memoria,a fin de disponer de la información estrictamente necesaria para su funcionamiento en paralelo. Como último paso, quedaba la mejora fundamental en el algoritmo: la utilización del cálculo paralelo para obtener más rápidamente la programación. Estos tres elementos adecuadamente equilibrados han conducido a resultados satisfactorios, como los seguidamente detallados.

Los resultados facilitados se basan en la ejecución del nuevo algoritmo sobre datos referidos a cinco programaciones semanales. Esta experiencia computacional se realizó en una Silicon Graphics (con un procesador MIPS de 64 bits) utilizando hasta 8 procesadores actuando conjuntamente. En las pruebas, se evaluó la duración del algoritmo con 1, 2, 4 y 8 procesadores, respectivamente. El proceso maestro y el primer proceso esclavo comparten procesador ya que los períodos de actividad de ambos apenas se solapan.

El proceso maestro consta básicamente de dos partes: en la primera, se capta la información necesaria para la programación de la producción de los datos almacenados en un fichero previamente generado y las tareas se preparan para ser tratadas (en estas acciones, la paralelización no interviene); en la otra parte, se gestiona el algoritmo en paralelo usando el modelo de maestro y esclavos. Los tiempos correspondientes a la primera parte no se consideran a efectos de "speed-up"porque no afectan a la ejecución en paralelo, pero sí, en cambio, al comparar las duraciones totales de las versiones nueva y anterior.

| Nº procesadores | 1 | 2 | 4 | 8 |
|---|---|---|---|---|
| Tiempo parte inicial | 3560 | 3955 | 4790 | 5429 |
| Tiempo parte en paralelo | 92808 | 63881 | 49828 | 34795 |
| Tiempo total | 96368 | 67836 | 54618 | 40224 |

Tabla 1 (en ms)

| Nº procesadores | 1 | 2 | 4 | 8 |
|---|---|---|---|---|
| Tiempo total | 96368 | 67836 | 54618 | 40224 |
| Tiempo actividad maestro | 1021 | 1091 | 1408 | 2196 |
| Tiempo medio actividad esclavos | 87292 | 59155 | 38832 | 17786 |
| Tiempo actividad esclavo 1 | 87292 | 61888 | 47928 | 33365 |
| Tiempo actividad esclavo 2 | | 56421 | 40347 | 26778 |
| Tiempo actividad esclavo 3 | | | 34867 | 21160 |
| Tiempo actividad esclavo 4 | | | 32185 | 16498 |
| Tiempo actividad esclavo 5 | | | | 12904 |
| Tiempo actividad esclavo 6 | | | | 11346 |
| Tiempo actividad esclavo 7 | | | | 10378 |
| Tiempo actividad esclavo 8 | | | | 9862 |

Tabla 2 (en ms)

| Nº procesadores | 1 | 2 | 4 | 8 |
|---|---|---|---|---|
| Tiempo parte inicial | 3560 | 3955 | 4790 | 5429 |
| Tiempo del proceso maestro (ms) | 92808 | 63881 | 49828 | 34795 |
| Speed-up | 1,000 | 1,453 | 1,863 | 2,667 |
| Utilización media del procesador | 1 | 0,73 | 0,47 | 0,33 |

Tabla 3

Los tiempos de la tabla 1 son representativos del tiempo de actividad de cada procesador, puesto que se ha incorporado una sentencia interna que recoge los tiempos de puesta en marcha y parada de cada procesador. Seguidamente, se desglosa la duración de actividad del proceso maestro y de los procesos esclavos para cada uno de los cuatro casos:

La utilización media de los procesos (% del cociente entre el tiempo de utilización de los procesos respecto al tiempo total de ejecución) se sitúa entre el 1% y el 6% en el proceso maestro, mientras que en los esclavos ronda el 90%. Estos datos indican que, por un lado, es sensato que el proceso maestro y un proceso esclavo compartan procesador, y por otro, el algoritmo paralelo tiene una estructura adecuada para usar los procesadores eficientemente.

El "speed-up" real, cuando el número de procesadores (procesos esclavos) aumenta, con la actual cantidad de órdenes y de factores en la evaluación de actividades, se aleja pronto del "speed-up" ideal. A la vista de los resultados, la aplicación no podría considerarse eficiente desde el punto de vista del paralelismo, es decir, de la utilización casi al 100% de los procesadores implicados.En la siguiente tabla,se observa que al aumentar el número de procesadores, se incrementa el "speed-up", si bien con una tendencia a reducir la tasa de incremento:

No obstante, la empresa usuaria del algoritmo no dispone de tecnología tan potente como la utilizada para obtener los anteriores resultados, sino de un servidor, con la arquitectura y el software necesario para esta aplicación,con múltiples procesadores, dentro de una red con varios PCs también equipados convenientemente. Esto implica que las duraciones de ejecución se incrementen sensiblemente.

## VALIDACIÓN DE LA PROGRAMACIÓN OBTENIDA

Para valorar cualitativamente el algoritmo, se considera, por un lado, la programación de actividades resultante de aplicar el nuevo algoritmo y, por otro, aquellas estadísticas surgidas de su utilización.

## PROGRAMACIÓN DE LOS RECURSOS PRODUCTIVOS

Los recursos productivos, definidos según las necesidades del usuario final, siguen unos procesos muy heterogéneos; esto implica gran variedad de aspectos. Estudiando la ocupación media de los recursos, fijado un número de turnos y de paradas por descanso del personal, se puede optimizar no sólo el uso de máquinas, sino también del personal. Al analizar la carga de trabajo en diferentes recursos de producción, la asignación de actividades a los recursos tiene perfectamente en cuenta los criterios de capacidad disponible total, la capacidad por módulo (en el supuesto de recursos modulares) y el número de módulos del recurso.

Según el número de órdenes de producción,la cantidad de actividades por orden y el tipo de proceso utilizado, la duración de la carga de máquinas será más corta o más larga. Sin embargo, el tiempo total se ha reducido suficientemente para un uso frecuente del algoritmo si la fábrica requiere cambios en la programación de las órdenes de fabricación.

El grado de paralelismo en el algoritmo también queda patente al programar órdenes con la carga en curso, con actividades que deben reprogramarse transcurrido un cierto tiempo de trabajo. Normalmente se trata de actividades de órdenes regeneradas a partir de antiguos archivos de programaciones. Las asignaciones de actividades a recursos en el instante inicial de la nueva carga de máquinas se actualizan y quedan pendientes de programar el resto de actividades de cada orden.

Los resultados de la programación obtenidos en dos ejecuciones del algoritmo no han de ser idénticos, porque las evaluaciones y asignaciones dependen de la distribución de cálculos entre los procesadores, que varía en cada puesta en marcha del algoritmo.

Si se tienen en cuenta los intervalos de carga,su número suele ser bastante alto (en la fábrica es habitual tomar 6 horas por intervalo), debido a unas pocas órdenes que duran muchos días. Estas programaciones tan prolongadas están ocasionadas por actividades realmente largas (de unos 30 días, en el caso de los secaderos) en comparación con las demás. Esta circunstancia resulta adversa al objetivo de máximo grado de paralelismo.

## ESTADÍSTICAS DEL ALGORITMO PARALELO

Finalizada la ejecución del algoritmo pueden extraerse algunos datos estadísticos para su análisis:el número mínimo de cálculos a repartir entre los procesos esclavos, la cantidad de cálculos que se han repetido, etc.

El número de actividades por carga de máquinas se sitúa alrededor de unas 800, subdivididas en unos 5.000 módulos o entregas, valor que es variable según el número de órdenes por semana en la planta. Como el número medio de recursos se sitúa entre 3 y 4, el número mínimo de cálculos a distribuir estará entre 14.000 y 20.000.

Por otro lado, hay una cantidad de cálculos repetidos porque la evaluación obtenida para una actividad no sea representativa en el momento de la asignación definitiva. En este caso, la incertidumbre en el reparto de tareas entre los esclavos no permite controlar exhaustivamente este factor. No obstante, este número de repeticiones no ha logrado reducirse a pesar de algunos intentos encaminados a minimizarlo.

Otro parámetro estadístico es el número de situaciones de espera durante la ejecución, cuando algún proceso esclavo está libre y sin cálculos pendientes de evaluación. Esta cantidad aumenta más que linealmente cuantos más procesos esclavos se utilizan, si queda una única orden al final de un intervalo de carga. Sin embargo, esta cantidad de situaciones de espera se mantiene muy similar en caso de ampliar la duración del intervalo de carga.

Como el usuario no toma fechas de entrega como referencia para los lanzamientos de órdenes en su estructura productiva,es necesario referirse a la duración mínima de una orden de fabricación.Es decir,

| Duración mínima de una orden | Duración programada de una orden |
|---|---|
| Hasta 24 horas | Máximo: 36 horas |
| Hasta 48 horas | Máximo: 88 horas |
| Hasta 72 horas | Máximo: 101 horas |
| Hasta 90 horas | Máximo: 174 horas |
| Hasta 120 horas | Máximo: 180 horas |

Tabla 4

se utiliza una desviación entre la "duración ideal" (sin considerar superposiciones entre actividades ni con otras órdenes) y la "duración programada" (lograda con el algoritmo).
Los resultados obtenidos con la nueva aplicación del algoritmo tienen una desviación en promedio del 33%.Analizados estos valores de manera absoluta, se puede elaborar la siguiente tabla, donde las órdenes se clasifican según su duración mínima:
En las programaciones elaboradas, se constata la existencia de dos grandes bloques de recursos: por un lado, aquellos recursos comunes a todas las líneas productivas, que marcan la mayor o menor duración de una orden; y por otro lado, otros recursos más específicos de cada línea de productos.

## 6. Conclusiones

La revisión del anterior algoritmo de carga de máquinas y la introducción de la paralelización en los cálculos ha permitido reducir los tiempos globales de ejecución. No obstante, la resolución de otras deficiencias detectadas ha conducido a lograr duraciones de ejecución aún más competitivas.
La primera reducción de tiempo se logró manejando adecuadamente los datos necesarios para la programación de la producción. Superada esta etapa, se preparó la estructura del algoritmo siguiendo el modelo de proceso maestro y procesos esclavos, con bajos rendimientos en paralelismo si se utilizaba un solo nivel. A lo cual, se respondió añadiendo un segundo nivel, que permitía acercarse más a los "speed-up" usuales en aplicaciones donde el paralelismo se demuestra válido.
Los tiempos de ejecución, incluso en ordenadores con procesadores estándares,son más que satisfactorios y plantean la posibilidad de incorporar progresivamente prestaciones complementarias para aplicar este algoritmo en tiempo real.
No obstante, los algoritmos tipo "dispatching" que procuran avanzar sincronizadamente en diferentes frentes plantean ciertos límites difíciles de superar al aplicar paralelización en los cálculos. Los intervalos de carga y/o los recálculos no propician mejores resultados en el speed-up [6].

## Bibliografía:


[1] Breshears, Clay."A beginner's guide to PVM- Parallel Virtual Machine". URL, <http://csep1.phy.oml.gov/CSEP/PVM/PVM.html>. University of Tennessee, 1995.

[2] Companys, Ramón; Corominas, Albert. Organización de la producción II. Dirección de operaciones 4. Edicions UPC, 1996.

[3] Feitelson, Dror; Rudolph, Larry. Job Scheduling Strategies for Parallel Processing. Springer, 1995.

[4] Jájá, Joseph. An Introduction to Parallel Algorithms.Addison-Wesley Publishing Company, 1992.

[5] Labarta, Jesús; Valero, Mateo; Gregoris, Luis."Introduction to parallel computing & PVM", CEPBA-UPC Working Paper, 1995.

[6] Zenios, Stavros A."Parallel and Supercomputing in the Practice of Management Science". Interfaces 24: 5 set-oct, pp. 122-140, 1994.