4 Análisis de los Trabajos Programados de una etapa en un Sistema Fijo

A continuación analizaremos diversos enfoques del problema de programación de los descargos como un problema FSP, es decir, considerando que el tiempo de inicio y finalización de un trabajo es conocido de antemano y no sufrirá variaciones. Además trataremos el problema como trabajos compuestos por una única etapa, dejando el estudio del problema considerando los trabajos formados por varias etapas (maniobras del descargo, trabajo en sí y maniobras de reposición) para capítulos posteriores. Comenzaremos analizando el problema cuando únicamente consideramos una clase de recurso, planteando su modelo desde dos objetivos distintos, el objetivo táctico que busca una gestión de los recursos para minimizar los costes realizando todos los trabajos planificados, y el objetivo operacional, que busca maximizar el beneficio obtenido por realizar el máximo número de trabajos que te permiten realizar los recursos de que dispones. Una vez visto el modelo para una única clase de recurso, procederemos a realizar el mismo estudio considerando varias clases de recursos, situación que es más interesante para su estudio, al igual que más compleja. Nuevamente se plantearan los distintos modelos para los dos objetivos que estudiaremos, el objetivo táctico y el objetivo operacional. Destacamos de nuevo la importancia de la existencia de distancias físicas entre los trabajos a realizar, y que por tanto están modeladas e introducidas en el modelo. Es interesante destacar este punto, ya que en la literatura actual no aparece nada sobre este tipo de problema.

4.1 Una Única Clase de Recurso

Este podría ser el escenario en que todos los trabajadores de la empresa estén cualificados para maniobrar, por lo que no habría lugar para la distinción de recursos (trabajadores en nuestro caso). Otro caso, quizás el más habitual dentro de las empresas eléctricas, es que sólo se subcontrate los trabajos de reparación, mantenimiento, conexión, siendo las maniobras en la red, realizadas por personal propio (de la empresa distribuidora). Esta situación encajaría dentro de este apartado para cualquiera de las

dos empresas, tanto la distribuidora como la subcontratada, ya que cada una de ellas únicamente realizaría un tipo de trabajo, que requeriría un único tipo de trabajador, o como en este caso una única clase de recurso.

En el problema FSP, en su versión más simple, sin considerar clases de trabajos ni recursos, se disponen de n trabajos con fecha de inicio y tiempo de procesado dados. Se define para un trabajo J_i, s_i como el instante de comienzo, f_i el instante de finalización y $t_i = s_i - f_i$ la duración del mismo. Se asocia también con cada trabajo un valor o peso del trabajo w_i, que representa el beneficio asociado a la realización del mismo. Los trabajos deben ser procesados sobre un conjunto de m recursos (trabajadores) paralelos (P_m). FSP se reduce al problema de determinar el número de recursos para procesar todos los trabajos (planificación táctica) o al de establecer una asignación de recursos que maximice el peso de los trabajos procesados (planificación operacional).

La figura 4 muestra los datos de un trabajo.

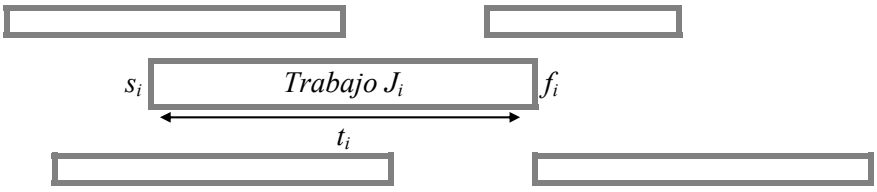

Figura 4. Datos de un problema FSP

Sin embargo, dada la casuística de nuestro problema, tenemos que considerar las distancias existentes entre los distintos trabajos, ya que, como comentamos con anterioridad, una de las características de este problema era la distinta ubicación geográfica de los trabajos a realizar. Estas distancias son conocidas de antemano, y por tanto son datos del problema.

4.1.1 Objetivo Táctico

Si nos referimos a la clasificación de los problemas de Interval Scheduling Problem (ISP en adelante), que hicimos en el capítulo 3.2, estamos en el caso de un problema FSP (fijo en invariable en el tiempo) con una planificación táctica (vamos a ralizar todos los trabajos con el mínimo coste) y considerando una única clase de trabajadores o recursos. A continuación se representa el esquema clasificatorio en el que se han marcado en amarillo la situación del problema que estamos estudiando en este capítulo.

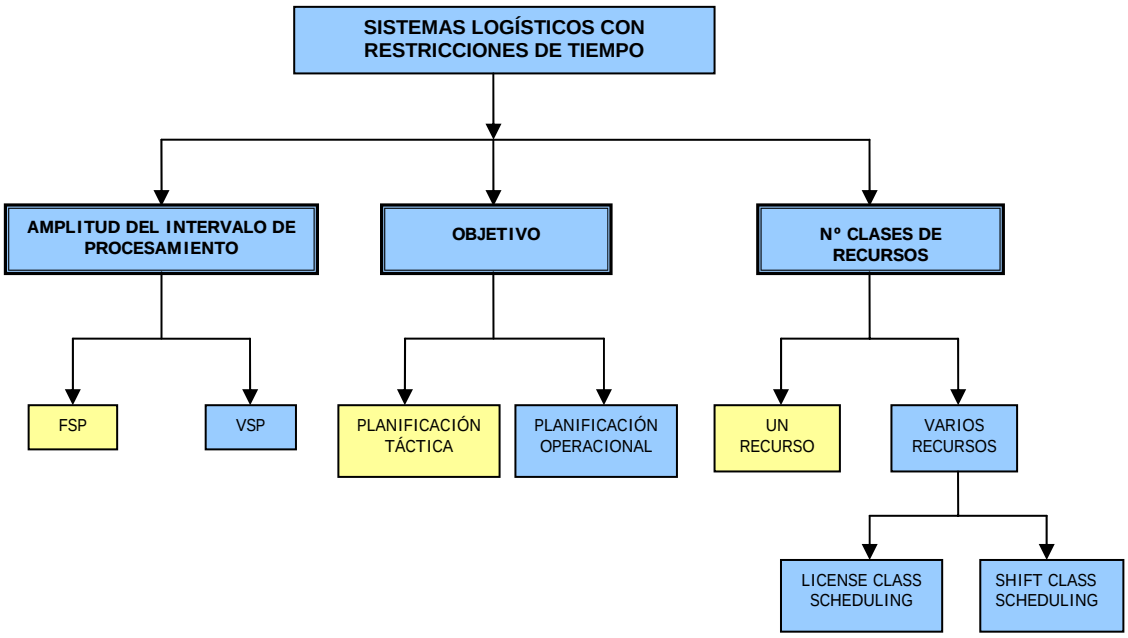

Figura 5. Clasificación de nuestro caso dentro de los problemas en sistemas logísticos

La formulación del problema es la siguiente:

Variables:

x_{ij} Variable binaria $\{0,1\}$, que toma el valor de la unidad si el trabajo J_i es asignado al recurso j

y_j Variable binaria $\{0,1\}$, que toma el valor de la unidad si utilizo el recurso j

Datos:

c_j	Coste de uso del recurso j
s_i	Instante de comienzo de J_i
f_i	Instante de finalización de J_i
D_{ik}	Distancias entre el trabajo J_i y el trabajo J_k (se refleja en unidades de tiempo)

Restricciones:

$$\sum x_{ij} = 1 \quad \forall i \tag{1.1}$$

Estas restricciones obligan a la realización de cada uno de los trabajos.

$$x_{ij} + x_{kj} \leq 1 \qquad i = 1 \ldots n\text{-}1;\ j = 1 \ldots m\ ;\ \{k > i;\ s_k < f_i\} \tag{1.2}$$

Estas restricciones controlan que dos trabajos solapados no sean procesados por el mismo recurso.

$$x_{ij} + x_{kj} \leq 1 \quad i = 1 \ldots n\text{-}1;\ j = 1 \ldots m\ ;\ \{k > i;\ f_i + D_{ik} > s_k\} \tag{1.3}$$

Restricción para el cumplimiento del tiempo de desplazamiento entre un trabajo y su predecesor.

Función Objetivo:

$$Min \sum_{j=1}^{m} c_j y_j \tag{1.4}$$

La función objetivo minimiza los costes totales de uso de cada recurso utilizado.

Modelo:

$$Min \sum_{j=1}^{m} c_j y_j$$

$$\sum x_{ij} = 1 \quad \forall i$$

$$x_{ij} + x_{kj} \leq 1 \qquad i = 1...n\text{-}1;\ j = 1...m\ ;\ \{k > i;\ s_k < f_i\}$$

$$x_{ij} + x_{kj} \leq 1 \qquad i = 1...n\text{-}1;\ j = 1...m\ ;\ \{k > i;\ f_i + D_{ik} > s_k\}$$

La solución del problema clásico FSP con una única clase de recurso, se obtiene como una consecuencia directa del teorema de Dilworth (Dilworth, 1950) sobre la descomposición de conjuntos parcialmente ordenados, donde se establece que en cualquier conjunto parcialmente ordenado el número mínimo de cadenas disjuntas que albergan todos los elementos es igual al número mayor de elementos mutuamente no relacionados en el conjunto. En este enfoque, la resolución del problema FSP se limita al cálculo del grado de simultaneidad de los trabajos en el tiempo, según el siguiente lema propuesto en Kroon et al, (1995):

Lema: *Una programación admisible para todos los trabajos existe si y sólo si el máximo grado de simultaneidad de los trabajos es menor o igual al número de recursos m*.

Sin embargo, y como hemos comentado en la introducción de los trabajos eléctricos, éstos tienen la peculiaridad de no encontrarse ubicados en el mismo lugar geográfico, lo que supone un desplazamiento entre los trabajos, y por tanto un tiempo de demora que debemos de formular e introducir en el modelo. En consecuencia, se define una matriz $\boldsymbol{D_{nxn}}$, en la que se refleja el tiempo de desplazamiento entre cada uno de los trabajos. Así pues, para este caso no es aplicable el teorema de Dilworth, ni por tanto el lema de Kroon.

El problema puede resolverse mediante un algoritmo de asignación avaricioso que, a pesar de asumir un número de recursos ilimitado, es equivalente considerar un número de recursos lo suficientemente grande de forma que siempre existan recursos disponibles. Una cota superior para el número de recursos podría ser el número de

trabajos (cada recurso procesaría un trabajo). Este algoritmo no garantiza la solución óptima, ya que cuando dispones de varios recursos libres (pongamos por ejemplo dos recursos M1 y M2), a la hora de asignar el trabajo a uno de ellos no podemos garantizar que en su asignación se esté eligiendo el óptimo, ya que puede que cojamos el recurso M1 para realizar un trabajo J_i en lugar del recurso M2 por estar éste más alejado. Se puede dar el caso, que una vez hecha esta asignación, tengamos un nuevo trabajo J_{i+1} a asignar, trabajo que es incompatible con M2 pero compatible con M1 (si éste no hubiera sido asignado a J_i). Al ser incompatible J_{i+1} y M2, tenemos que recurrir a un nuevo recurso M3, recurso que no tendría que haberse activado si se hubieran asignado correctamente los recursos M1 y M2 (J_i a M2 y J_{i+1} a M1). Así pues, estamos ante un problema de mayor complejidad, ya que estamos introduciendo un problema de asignación dentro del problema de gestión de los trabajos programados que estamos analizando. Para reducir en la medida de lo posible esta problemática, cogeremos un algoritmo que asigne al trabajo la máquina compatible que haya terminado su trabajo más tarde, ya que así estamos dando más tiempo al resto de recursos para realizar sus desplazamientos, o lo que es lo mismo, estamos reduciendo las incompatibilidades por desplazamiento del resto de recursos.

El algoritmo comienza creando dos conjuntos vacíos (M_A) y (M_L). El conjunto que denominaremos **Recursos Activados (M_A)** será el conjunto de recursos que hayan sido empleados para la ejecución de algún trabajo. Por eso, inicialmente se crea el conjunto sin ningún valor, ya que aún no se ha asignado ningún trabajo a ningún recurso. El conjunto que denominaremos **Recursos Libres (M_L)**, será el conjunto de recursos que, habiendo sido asignados al menos una vez a un trabajo, es decir que pertenecen al conjunto (M_A), estén libres o desocupados por haber finalizado el trabajo que se le asignó en su momento, es decir el conjunto (M_L) será un subconjunto de (M_A) en el que no se encuentran aquellos recursos que en el instante t están realizando algún trabajo. La estrategia que sigue el algoritmo consiste en recorrer el conjunto de trabajos, ordenados por fecha de inicio s_i, e ir asignando un recurso libre a cada trabajo. Inicialmente, al no haber ningún recurso en el conjunto (M_A), ni por consiguiente en el conjunto (M_L), asigna un recurso (M_a) cualquiera a ese trabajo. La primera asignación no tiene en cuenta ningún tipo de distancias ya que suponemos que todos los recursos se

encuentran en el mismo punto de partida. Una vez asignado el recurso, actualizamos el conjunto (M_A), introduciendo el recurso (M_a) a dicho conjunto, y calculamos las distancias existentes entre el trabajo asignado y el resto de trabajos pendientes de asignar (L_a).

Una vez iniciado el algoritmo, para la asignación de nuevos trabajos a los recursos disponibles, se exploran primero los recursos que ya han realizado algún trabajo (es decir los pertenecientes al conjunto recursos libres). De entre ellos, el trabajo es asignado al recurso que haya finalizado su trabajo predecesor más tarde (como ya hemos comentado), siempre que cumpla con las restricciones de compatibilidad en desplazamientos, es decir que una vez finalizado el trabajo precedente en f_i, al sumarle el tiempo de desplazamiento necesario para ir desde el trabajo J_i al J_k, el instante resultante sea previo a s_k, instante de inicio de la tarea J_k. Para ello, creamos un conjunto que llamaremos **Recursos Libres utilizables (M_S),** el cual coincide en primera instancia con M_L, cogemos el recurso que haya quedado libre más tarde y comprobamos si cumple con las restricciones de compatibilidad. Si así fuera, el trabajo es asignado a este recurso, calculando las nuevas distancias de este recurso al resto de trabajos pendientes, y asignando el tiempo de finalización del trabajo en cuestión como tiempo de liberación del recurso (a partir de ese instante pasa a pertenecer al conjunto M_L). En el caso de que no cumpliera la restricción, sacaríamos este recurso del conjunto de Recursos Libres utilizables (M_S), y volveríamos a proceder de manera análoga, con el recurso perteneciente a M_S que hubiese sido liberado más tarde. Si finalmente el conjunto (M_S), quedase vacío sin que ningún recurso fuera compatible, se procedería a activar un nuevo recurso como ya se vio con anterioridad.

El pseudocódigo para el algoritmo, denominado *GREEDY*, se muestra a continuación en la figura 6 así como la definición de las variables usadas.

Variables:

- **Recursos Activados (M_A)** conjunto de recursos que hayan sido empleados para la ejecución de algún trabajo

- **Recursos Libres (M_L)**, conjunto de recursos que pertenecen a (M_A) y no están realizando ningún trabajo en el instante t
- **Recursos Libres utilizables (M_S)** conjunto de recursos libres a los que se les ha de comprobar su compatibilidad por desplazamiento.
- ***a*** devuelve el número de recursos utilizados
- L_a Vector en el que se refleja las distancias existentes entre el recurso Ma y el resto de trabajos pendientes de realizar
- l_a Instante en el que el recurso M_a queda liberado

Procedimiento *GREEDY*

$M_A = \varnothing$
$M_L = \varnothing$
$a = 0$

Para cada $J_i \in J$
*Actualizar_Conjunto_M_L (s_i)

Si $M_L = \varnothing$ **entonces**
$a = a + 1$
$M_A = M_A + \{M_a\}$
Asigna J_i a M_a
$L_a=(D_{a\,i+1},\ldots..,\ D_{ak},\ldots..,\ D_{an})$
Calcula distancias entre *Ma* y el resto de trabajos.
$l_a = f_i$
Asigno el tiempo de finalización del trabajo a la liberación del recurso

si no
$M_S = M_L$
*Búsqueda_Mejor_Compatible_ M_S

Fin Si

Fin Para
Fin

Figura 6. Pseudo-código del algoritmo de asignación

El procedimiento "Actualizar_Conjunto_M_L (s_i)" introduce en M_L aquellos recursos de M_A que hayan quedado liberadas para el instante s_i. En cuanto al procedimiento "Búsqueda_Mejor_Compatible_ M_S" selecciona del conjunto M_S el recurso que hay sido liberado más tarde y comprueba su compatibilidad. Si es compatible le asigna al trabajo, si no es así, elimina este recurso del conjunto M_S , y vuelve a proceder de la misma manera hasta que el conjunto M_S esté vacío, en cuyo caso activa un nuevo recurso al conjunto M_A. La variable a devuelve el número de recursos utilizados. La variable L_a calcula las distancias existentes entre el trabajo asignado a esa máquina y el resto de trabajos por asignar. En este caso estudiado, se asigna a cada recurso activado los tiempos que emplearía en desplazarse para atender cada uno de los trabajos. Así podemos comprobar si el recurso que esté realizando el trabajo J_i, es capaz de atender en el instante s_k el trabajo J_k, o si por el contrario, debemos de disponer de otro recurso.
Veamos como se resolvería a modo de ejemplo el problema planteado en el apartado 3.1 adaptado a las condiciones de este capítulo, siguiendo este algoritmo. Se disponen de 10 trabajos, con 10 recursos disponibles (equivalente a suponer capacidad de recursos ilimitada). En la tabla 3 se presentan las principales características de los trabajos a realizar. Asimismo, en la tabla 4 se representan los tiempos que se emplearían en ir de un trabajo a otro. Consideraremos que todos los recursos son análogos, tanto en coste como en tiempo de ejecución.

Tabla 3. Características de los trabajos

J_i	a_i	b_i
1	0	4
2	0	4
3	1	8
4	2	11
5	5	12
6	10	15
7	11	15
8	12	18
9	15	18
10	16	20

Tabla 4. Distancias entre trabajos

	1	2	3	4	5	6	7	8	9	10
1	0	0	2	5	6	3	1	5	6	2
2	0	0	1	4	7	4	2	6	4	2
3	2	1	0	3	2	6	0	7	8	2
4	5	4	3	0	3	1	1	9	3	1
5	6	7	2	3	0	2	4	12	2	1
6	3	4	6	1	2	0	5	8	1	6
7	1	2	0	1	4	5	0	8	11	6
8	5	6	7	9	12	8	8	0	9	4
9	6	4	8	3	2	1	11	9	0	8
10	2	2	2	1	1	6	6	4	8	0

En la figura 7 se representa la distribución de los trabajos en el tiempo

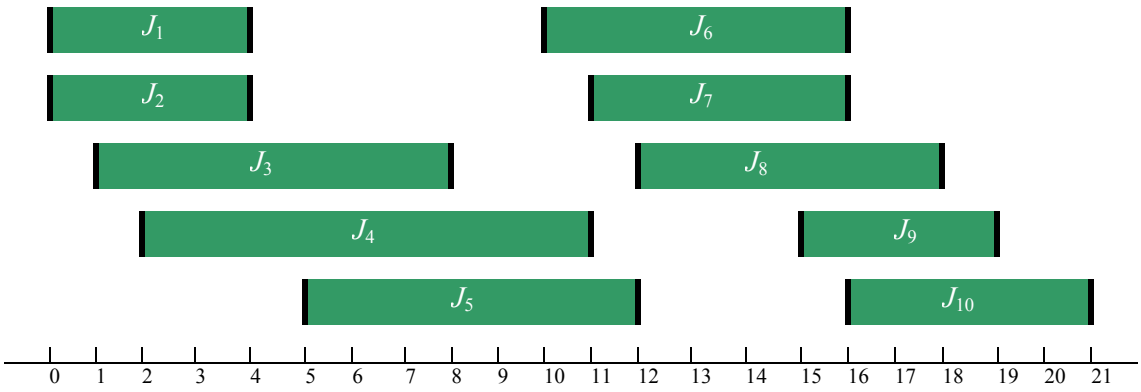

Figura 7. Representación del problema

Veamos ahora paso a paso como va evolucionando el algoritmo. En primer lugar, se crea el conjunto M_A o conjunto de recursos asignados que inicialmente está vacío. Asimismo, creamos un segundo conjunto M_L o conjunto de recursos libres, que al igual que el anterior, inicialmente está libre. Comienza el algoritmo con la llegada del primer trabajo J_1, actualizamos en primer lugar los recursos libres, que en este caso sigue siendo nulo, por lo que actualizamos el valor de a, que ahora tiene el valor de la unidad, incluimos en M_A al recurso M_1 y calculo las distancias con el resto de trabajos pendientes de asignar (datos reflejados en la tabla 5). Asimismo, le asignamos el tiempo de liberación de la máquina el del final del trabajo, es decir $\boldsymbol{l_1=4}$

Tabla 5. Distancias del recurso 1 al resto de trabajos

2	0
3	2
4	5
5	6
6	3
7	1
8	5
9	6
10	2

Una vez asignado recurso al trabajo J_1, procedemos a asignar de la misma forma el trabajo J_2, actualizando los recursos libres, asignándole un nuevo recurso M_2, que junto a M_1 formarán el nuevo conjunto M_A, siendo el nuevo valor de a 2 y se calculan las nuevas distancias del recurso utilizado con el resto de trabajos así como el instante de liberación del recurso M_2, l_2=4. Se opera de la misma forma hasta la llegada del trabajo J_5. En la figura 8 se representa el estado de nuestro problema justo en el instante de la llegada del trabajo J_5.

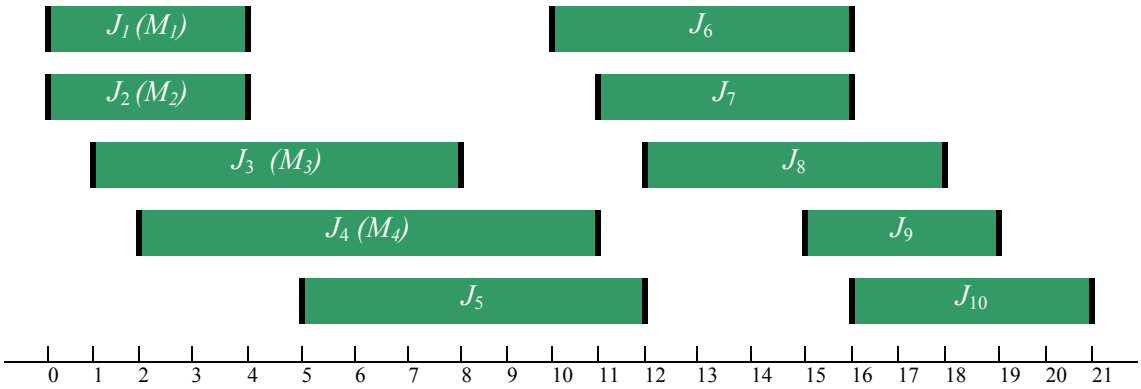

Figura 8. Representación del problema en t=5

En el instante de llegada del trabajo J_5, al actualizar el conjunto de recursos libres, se añade un nuevo recurso (el primero en nuestro problema) ya que hasta el momento estaba el conjunto vacío. En este instante se añaden los recursos M_1 y M_2 al conjunto de recursos libres ya que en este instante ya habían terminado los trabajos a los que habían

sido asignados. Asimismo, creamos el conjunto M_S, conjunto de recursos libres utilizables en el que están incluidos ambos recursos Así pues ordenamos los recursos por orden de finalización de los trabajos predecesores, que en este caso ambos coinciden, por lo que elegimos el recurso M_1. Una vez elegido el recurso, comprobamos si es capaz de llegar al inicio del trabajo, para ello vemos en la tabla 5 el tiempo que tardaría en el desplazamiento que en nuestro caso es de 6 unidades, lo que sumado al tiempo de finalización del primer trabajo asignado daría una disponibilidad para realizar el trabajo J_5 en t=10, lo cual no es viable. Así pues, procedemos ahora a quitar del conjunto M_S el citado recurso M_1, quedando el conjunto M_S reducido a M_2. Llegados a este punto, volvemos a comprobar la compatibilidad del siguiente recurso perteneciente a M_S, que en este caso en M_2 comprobando al igual que en el caso anterior su incompatibilidad con el trabajo, por lo que recurrimos a introducir un nuevo recurso M_5. Con la llegada del trabajo J_6 volvemos a repetir el proceso, de nuevo al actualizar los recursos libres, se incorpora el recurso M_3, ya que en este instante ya ha finalizado el trabajo al que fue asignado. Nuevamente creamos el conjunto M_S y elegimos el recurso que ha sido liberado más tarde, M_3 en nuestro caso. Elegido el recurso, procedemos a comprobar la compatibilidad, para ello, al instante de liberación l_3 (8 en este caso) le sumamos el desplazamiento hasta el trabajo J_6, L_{36} (6 en este caso), por lo que el recurso podría atender el trabajo J_6 a partir de t=14, siendo por tanto incompatible. De nuevo procedemos a eliminar M_3 del conjunto M_S y volvemos a aplicar la casuística para el siguiente recurso, que en este caso es M_1. Calculemos pues su compatibilidad, l_1=4 y su desplazamiento hasta el trabajo M_1 que vienen reflejados en la tabla 5 L_{16} es de 3 unidades, por lo que desde el instante t=7, podríamos utilizar el recurso M_1 para este trabajo, instante posterior a su inicio, por lo que definitivamente asignamos el recurso M_1 al trabajo J_6. A continuación eliminamos este recurso del conjunto de recursos libres y volvemos a calcular las distancias con el resto de trabajos por asignar así como el nuevo instante de liberación del recurso.

Repitiendo el proceso hasta el último trabajo, la planificación quedaría como se representa en la figura 9.

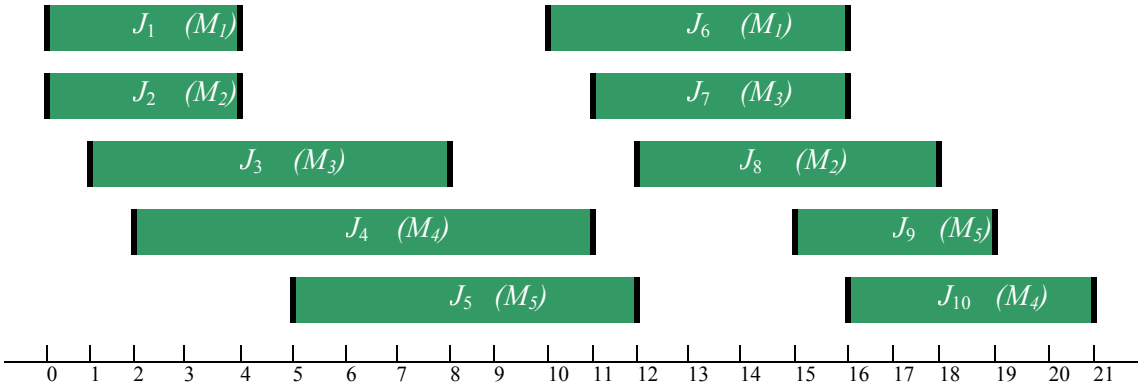

Figura 9. Representación de la solución del problema

En la asignación realizada se procesan los 10 trabajos, utilizando 5 de los 10 recursos, con el consiguiente ahorro de coste.

4.1.2 Objetivo Operacional

Refiriéndonos de nuevo a la clasificación de los problemas de ISP que hicimos en el capítulo 3.2, estamos en el caso de un problema FSP (fijo en invariable en el tiempo) con una planificación operacional (vamos a maximizar los beneficios con los recursos disponibles) y considerando una única clase de trabajadores o recursos. A continuación se representa el esquema clasificatorio en el que se han marcado en amarillo la situación del problema que estamos estudiando en este capítulo.

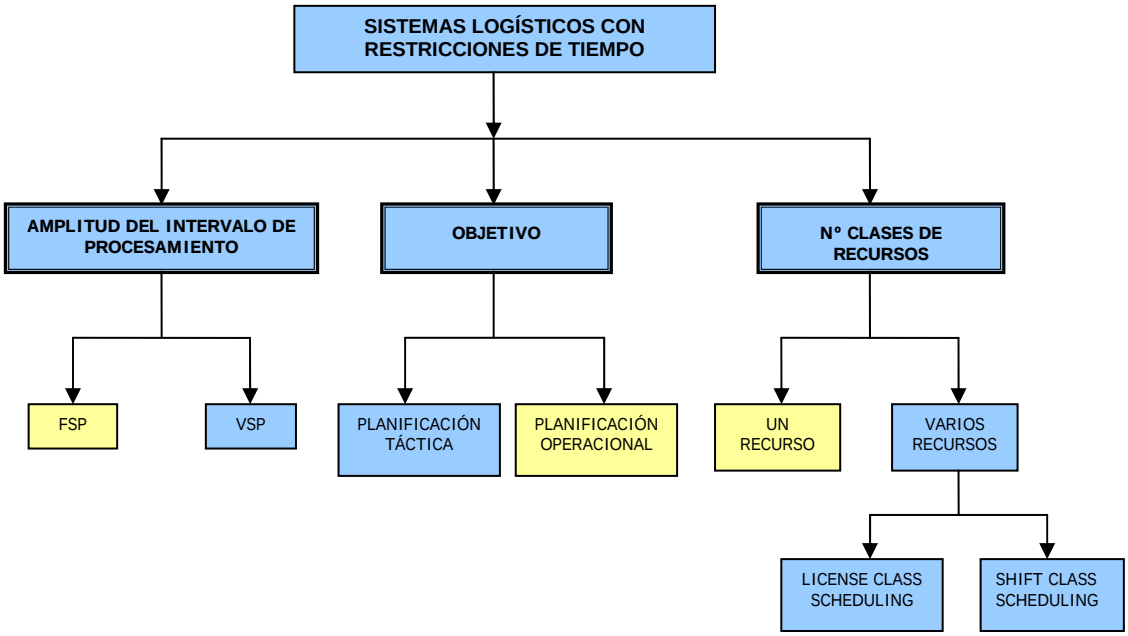

Figura 10. Clasificación de nuestro caso dentro de los problemas en sistemas logísticos

Cuando el número de recursos es menor que el mayor grado de simultaneidad de los trabajos, teniendo en cuenta en esta simultaneidad el tiempo de desplazamiento para atender con el mismo recurso los distintos trabajos, el problema FSP llega a ser más interesante. En este caso el objetivo del problema pasa a ser el de encontrar el subconjunto de trabajos a ser procesados. Para la selección de trabajos se atiende al criterio del máximo valor total del conjunto de trabajos seleccionados. Cuando suponemos el mismo valor w_i para todos los trabajos, el objetivo se convierte en maximizar el número de trabajos procesados. Encontramos por tanto un problema de secuenciación en el que no se realizan todos los trabajos, sino solamente un subconjunto de ellos, elegido en base a los pesos de los mismos. La formulación del problema es la siguiente:

Variables:

x_i Variable binaria {0,1}, que toma el valor de la unidad si se realiza el trabajo J_i.

x_{ij} Variable binaria {0,1}, que toma el valor de la unidad si el trabajo J_i es asignado al recurso j

Datos:

w_i	Peso asociado al trabajo J_i
s_i	Instante de comienzo de J_i
f_i	Instante de finalización de J_i
D_{ik}	Distancias entre el trabajo J_i y el trabajo J_k (se refleja en unidades de tiempo)

Restricciones:

$$\sum_{j=1}^{m} x_{ij} \le x_i \quad \forall i \tag{2.1}$$

Estas restricciones activa como procesado un trabajo que ha sido asignado a una recurso.

$$x_{ij} + x_{kj} \le 1 \qquad i = 1 \ldots n-1;\ j = 1 \ldots m;\ \{k > i : s_k < f_j\} \tag{2.2}$$

Estas restricciones controlan que dos trabajos solapados no sean procesados por el mismo recurso.

$$x_{ij} + x_{kj} \le 1 \quad i = 1 \ldots n\text{-}1;\ j = 1 \ldots m\ ;\ \{k > i; f_i + D_{ik} > s_k\} \tag{2.4}$$

Restricción para el cumplimiento del tiempo de desplazamiento entre un trabajo y su predecesor.

Función Objetivo:

$$Max \sum_{i=1}^{n} w_i x_i \tag{2.5}$$

La función objetivo maximiza el peso total de los trabajos procesados.

Modelo:

$$Max \sum_{i=1}^{n} w_i x_i$$

$$\sum_{j=1}^{m} x_{ij} \le x_i \quad \forall i$$

$$x_{ij} + x_{kj} \le 1 \qquad i = 1 \ldots n-1;\ j = 1 \ldots m;\ \{k > i : s_k < f_j\}$$

$$\sum_{\{i / s_i \le t \le f_i\}} x_i \le m \quad t \in T$$

$$x_{ij} + x_{kj} \le 1 \quad i = 1 \ldots n\text{-}1;\ j = 1 \ldots m\ ;\ \{k > i;\ f_i + D_{ik} > s_k\}$$

La resolución de este modelo no se puede acometer mediante el uso de grafos, ya que al introducir el concepto de desplazamiento entre trabajos, no es posible su aplicación. Por ello, para resolver este modelo, recurriremos a la utilización de librerías de optimización XA, resolución que al aplicar un método que no hemos tratado hasta ahora, vamos a dedicar un nuevo apartado (apartado 5) en el que se explicará con más detalle tanto el concepto de las librerías de optimización como su resolución.

4.2 Varias Clases de Recursos

Sería éste el caso más complejo del problema FSP. Ahora un trabajo no puede ser procesado por cualquier recurso. El caso de varias clases de recursos y trabajos integra las dos variantes señaladas en la clasificación realizada en la sección 2 (incompatibilidad por motivos técnicos o por disponibilidad temporal de las recursos). Ambas variantes presentan las mismas características respecto a su complejidad. A pesar de ello, han sido analizados en la bibliografía de forma independiente. Básicamente, puede concluirse que el problema en su objetivo táctico y operacional es de carácter NP a partir de 3 clases diferentes de recursos (Kolen y Kroon, 1991, 1992, 1993, 1994). Sin embargo, no encontramos en la literatura, el estudio del problema cuando existen distancias entre los trabajos a realizar.

Como ya hemos comentado al describir los trabajos a realizar en una empresa distribuidora eléctrica, éste es el caso que más se acerque a la realidad, ya que tenemos dos tipos de recursos, aquellos trabajadores que no están capacitados para realizar las maniobras, y aquellos que están capacitados y autorizados para realizar las maniobras y consecuentemente todos los tipos de trabajos.

A continuación se presentan los modelos matemáticos asociados con cada objetivo.

4.1.3 Objetivo Táctico

Nuevamente hacemos referencia a la clasificación de los problemas de ISP que hicimos en el capítulo 3.2, estamos en el caso de un problema FSP (fijo en invariable en el tiempo) con una planificación táctica (vamos a minimizar costes en la realización de los trabajos programados) y considerando varias clases de trabajadores o recursos, concretamente y como ya se ha visto a lo largo de este trabajo, los recursos habilitados para maniobrar y los que no. A continuación se representa el esquema clasificatorio en el que se han marcado en amarillo la situación del problema que estamos estudiando en este capítulo.

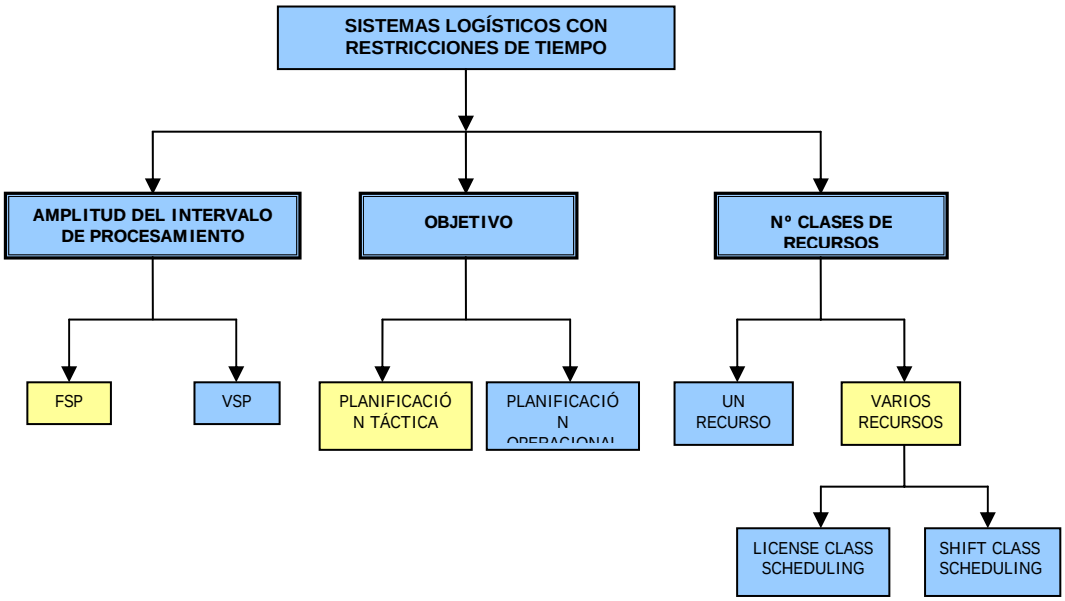

Figura 11. Clasificación de nuestro caso dentro de los problemas en sistemas logísticos

El modelo matemático que expresa el objetivo táctico del problema con varias clases es el siguiente:

Variables:

x_{ij}	Variable binaria {0,1}, que toma el valor de la unidad si el trabajo J_i es asignado a la recurso j
y_j	Variable binaria {0,1}, que toma el valor de la unidad si utilizo la recurso j

Datos:

c_j	Coste de uso de la recurso j
s_i	Instante de comienzo de J_i
f_i	Instante de finalización de J_i
D_{ik}	Distancias entre el trabajo J_i y el trabajo J_k (se refleja en unidades de tiempo)

Parámetros:

$L_{D\times C}$	Matriz de compatibilidad trabajo-recurso

Restricciones:

$$\sum_{j/L(d_i,c_j)=1} x_{ij} = 1 \quad \forall i \tag{3.1}$$

Estas restricciones obligan a la realización de cada trabajo, mediante alguna de las recursos con compatibilidad.

$$x_{ij} + x_{kj} \le 1 \qquad i = 1\ldots n-1;\ j = 1\ldots m;\ \{k > i : s_k < f_j\} \tag{3.2}$$

Estas restricciones controlan que dos trabajos solapados no sean procesados por el mismo recurso.

$$x_{ij} + x_{kj} \le 1 \quad i = 1\ldots n\text{-}1;\ j = 1\ldots m\ ;\ \{k > i;\ f_i + D_{ik} > s_k\} \tag{3.3}$$

Restricción para el cumplimiento del tiempo de desplazamiento entre un trabajo y su predecesor.

Función Objetivo:

$$Min \sum_{j=1}^{m} c_j y_j \qquad (3.4)$$

La función objetivo minimiza los costes totales de uso de cada recurso utilizada.

Modelo:

$$Min \sum_{j=1}^{m} c_j y_j$$

$$\sum_{j/L(d_i,c_j)=1} x_{ij} = 1 \quad \forall i$$

$$x_{ij} + x_{kj} \le 1 \qquad i = 1 \ldots n-1;\ j = 1 \ldots m;\ \{k > i : s_k < f_j\}$$

$$x_{ij} + x_{kj} \le 1 \qquad i = 1 \ldots n\text{-}1;\ j = 1 \ldots m\ ;\ \{k > i;\ f_i + D_{ik} > s_k\}$$

4.1.4 Objetivo Operacional

De la clasificación de los problemas de ISP que hicimos en el capítulo 3.2, estamos en el caso de un problema FSP (fijo en invariable en el tiempo) con una planificación operacional (vamos a maximizar los beneficios con los recursos disponibles) y considerando una varias clase de trabajadores o recursos como ya hemos explicado. En el esquema adjunto se marcan en amarillo el lugar en el que ubicamos nuestro problema.

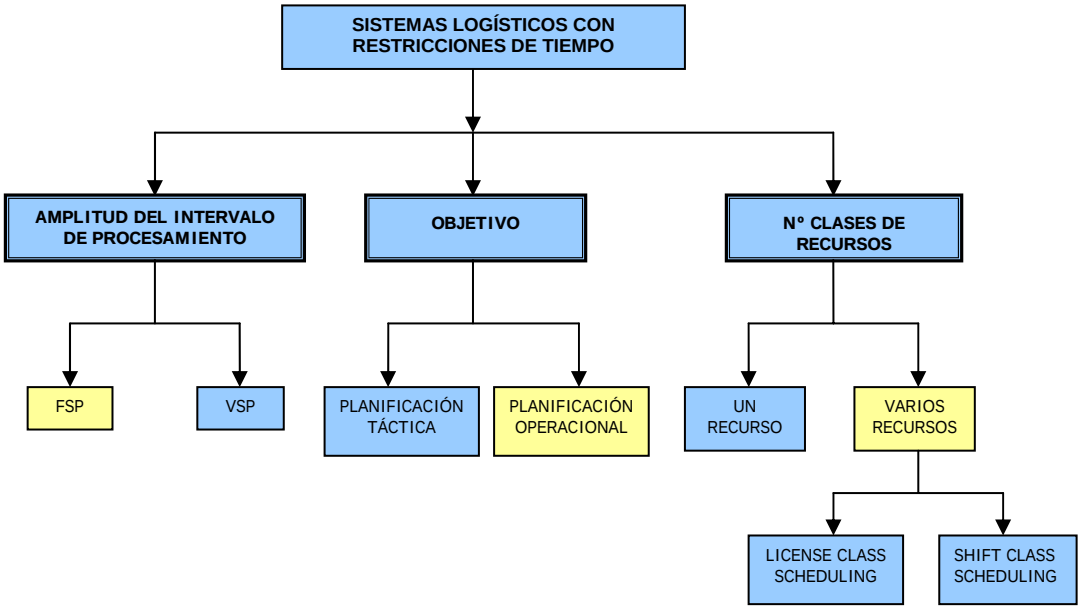

Figura 12. Clasificación de nuestro caso dentro de los problemas en sistemas logísticos

El modelo que expresa el objetivo operacional es el siguiente:

Variables:

x_{ij}	Variable binaria {0,1}, que toma el valor de la unidad si el trabajo J_i es asignado a la recurso j
x_i	Variable binaria {0,1}, que toma el valor de la unidad si el trabajo J_i es procesado

Datos:

w_i	Peso del trabajo J_i
s_i	Instante de comienzo de J_i
f_i	Instante de finalización de J_i
D_{ik}	Distancias entre el trabajo J_i y el trabajo J_k (se refleja en unidades de tiempo)
M	Número de recursos

Parámetros:

$L_{D\times C}$ Matriz de compatibilidad trabajo-recurso

Restricciones:

$$\sum_{j/L(d_i,c_j)=1} x_{ij} = 1 \quad \forall i \tag{4.1}$$

Estas restricciones controlan el procesamiento de cada trabajo por una clase de recurso compatible.

$$x_{ij} + x_{kj} \le 1 \qquad i = 1\ldots n-1;\ j = 1\ldots m;\ \{k > i : s_k < f_j\} \tag{4.2}$$

Estas restricciones controlan que dos trabajos solapados no sean procesados por el mismo recurso.

$$x_{ij} + x_{kj} \le 1 \quad i = 1\ldots n\text{-}1;\ j = 1\ldots m\ ;\ \{k > i; f_i + D_{ik} > s_k\} \tag{4.4}$$

Restricción para el cumplimiento del tiempo de desplazamiento entre un trabajo y su predecesor.

Función Objetivo:

$$Max \sum_{i=1}^{n} w_i x_i \tag{4.5}$$

La función objetivo maximiza el peso total de los trabajos procesados.

Modelo:

$$Max \sum_{i=1}^{n} w_i x_i$$

$$\sum_{j/L(d_i,c_j)=1} x_{ij} = 1 \quad \forall i$$

$$\sum_{\{i/s_i \le t \le f_i\}} x_i \le m \quad t \in T$$

$$x_{ij} + x_{kj} \le 1 \qquad i = 1 \ldots n-1;\ j = 1 \ldots m;\ \{k > i : s_k < f_j\}$$

$$x_{ij} + x_{kj} \le 1 \quad i = 1 \ldots n\text{-}1;\ j = 1 \ldots m\ ;\ \{k > i; f_i + D_{ik} > s_k\}$$

Cuando el peso de los trabajos es el mismo o, simplemente, la unidad, y el objetivo es maximizar el número de trabajos realizados, el problema FSP con objetivo operacional e intervalos de disponibilidad para recursos aparece también en la literatura como *k-Track Assignment Problem* (TAP), donde *k* define el número de recursos.

TAP es considerado por Brucker y Nordmann (1994), quienes proponen un método exacto de resolución de orden $O(n^{k-1}k!k^{k+1})$. El método construye un grafo que recoge todos los posibles programas admisibles del problema. La ruta máxima en el grafo obtiene la solución óptima del problema. El método se muestra muy ineficiente con un número de recursos superior a 4.

Para un caso particular del problema presenta un algoritmo de orden $O(n^{m-1})$ basado también en el cálculo de la ruta máxima. Considera también el caso en el que únicamente existen dos intervalos distintos de trabajo para los recursos, proponiendo un algoritmo de orden $O(n)$. Este caso es contemplado también por Hsu y Tsai (1989). Faigle and Nawijin (1995) y Faigle *et al* (1999) que estudian también el problema TAP tanto en su forma normal como en una versión online.

El problema con únicamente dos clases recursos, problema que se ajusta al que estamos aquí tratando, ha sido estudiado por Dondeti y Emmons (1992), mostrando que puede ser resuelto en tiempo polinomial.

Para el caso genérico del problema, permitiendo pesos en los trabajos, Arkin y Silverberg(1987) proponen un método exacto con una filosofía parecida a la descrita por Brucker y Nordman, aunque presenta un comportamiento mejor, puesto que el orden del número de nodos y arcos del problema es, respectivamente, $O(n^m)$ y $O(n^{m+1})$.

Aproximaciones heurísticas para este problema han sido consideradas en Gabrel (1995) y Kroon *et al* (1995). En el primer caso, el problema FSP es modelado como un problema de máximo conjunto de nodos independientes (Problema MIS). Se analizan tres procedimientos heurísticos para su resolución. Siempre que no se consideren pesos, los problemas FSP y VSP con objetivo operacional pueden ser modelados como un problema MIS.

El problema FSP permitiendo *preemption* es considerado en Dondeti y Emmons (1993) probando que puede ser resuelto en tiempo polinomial transformando el problema en un grafo de transporte.

Sin embargo, como ya hemos comentado, al aplicar el concepto de desplazamientos entre trabajos, no es posible aplicar grafos para su resolución, por lo que no podemos aplicar los ejemplos anteriores a nuestro caso.