3. ESCENARIO I: PROGRAMACIÓN DE TRABAJOS FIJOS INCORPORANDO DISTANCIA ENTRE TRABAJOS

3. ESCENARIO I: PROGRAMACIÓN DE TRABAJOS FIJOS INCORPORANDO DISTANCIA ENTRE TRABAJOS

3.1 INTRODUCCIÓN

Este escenario está conformado por un problema FSP con una clase de recurso (o máquina)[1], al cual se incorporan desplazamientos entre trabajos, que se traducen en tiempos de traslado, por no encontrarse éstos en igual lugar geográfico. A cada uno de los lugares geográficos donde puede realizarse un trabajo se los llamará, de ahora en adelante y por simplicidad, *destino*. El teorema de Dilworth y el lema de Kroon dejan de ser aplicables, y se incorpora la definición de una matriz D_{nxn} para reflejar estos tiempos de traslado entre destinos.

La incorporación de la distancia es interesante cuando las tareas a realizar están dispersas geográficamente, de manera que el recurso que las realiza pueda tener que desplazarse, entre la finalización de una tarea y el comienzo de la siguiente, una cierta distancia. De esta forma, a la hora de determinar qué recurso procesará cada tarea y cuándo lo hará, se puede tomar como objetivo el de minimizar estos tiempos de traslado, o simplemente tenerlos en cuenta a la hora de minimizar el tiempo total de ejecución de las tareas, al aumentar el peso total de las tareas realizadas, o cualquier otra combinación que resulte interesante para el problema en cuestión.

Un ejemplo del marco de aplicación de este modelo es el que se presenta en las compañías de autobuses. Habitualmente, éstas cuentan con más de un punto de relevo de conductor, de forma que estos terminan un viaje en un punto y deben comenzar el siguiente en otro, debiendo trasladarse en consecuencia. Otro campo de aplicación es el de trabajadores que deben realizar tareas en más de un punto durante su jornada, como puede ser el de los empleados de compañías eléctricas que acuden a reparar averías. Queda a cargo del lector pensar en más ejemplos.

---

[1] Para más información sobre este tipo de problema, ver Capítulo 2.

3.2 EL MODELO

3.2.1 OBSERVACIONES

Si bien el campo de aplicación más habitual para el modelo es probablemente el de los recursos humanos, en el desarrollo del mismo se hablará de trabajos y máquinas, para mantener coherencia con el resto del documento. Entiéndanse como trabajos las tareas a realizar, y como máquinas el recurso que las realice, ya sea máquina o trabajador.

Se modela un objetivo operacional, en el cual se pretende maximizar la suma de los pesos de los trabajos realizados, y un objetivo táctico, en el cual se pretende minimizar el número total de recursos necesarios.

3.2.2 OBJETIVO OPERACIONAL

Variables:

x_{ij}	Variable binaria {0,1}, que toma el valor de la unidad si el trabajo *i* lo realiza la máquina *j.*
x_i	Variable binaria {0,1}, que toma el valor de la unidad si se realiza el trabajo *i.*

Datos:

s_i	Instante de comienzo del trabajo *i.*
f_i	Instante de finalización del trabajo *i.*
w_i	Peso asociado al trabajo *i.*
D_{ik}	Distancias entre el trabajo i y el trabajo k, reflejadas en unidades de tiempo.

Restricciones:

$$\sum_{j} x_{ij} \leq x_i \ \ \forall i \tag{3.1}$$

Activa como procesado un trabajo que ha sido asignado a una máquina.

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

Controla que dos trabajos solapados no los realice una misma máquina.

$$x_{ij} + x_{kj} \leq 1 \ \ i = 1, \dots n-1; j = 1, \dots m; \{k > i; f_i + D_{ik} > s_k\} \tag{3.3}$$

Garantiza el cumplimiento del tiempo de desplazamiento entre un trabajo y otro.

Función Objetivo:

$$Max \sum_{i=1}^{n} w_i . x_i \tag{3.4}$$

Maximiza el peso total de los trabajos procesados.

Modelo:

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

$$\sum_{j} x_{ij} \leq x_i \ \ \forall i$$

$$\sum_{i/s_i \leq t \leq f_i} x_i \leq M \ \ \ t \in T$$

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

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

3.2.3 OBJETIVO TÁCTICO

Variables:

x_{ij}	Variable binaria {0,1}, que toma el valor de la unidad si el trabajo *i* lo realiza el recurso *j.*
y_j	Variable binaria {0,1}, que toma el valor de la unidad si se utiliza el recurso *j.*

Parámetros:

N	Cota superior de trabajos que puede realizar *j*

Datos:

s_i	Instante de comienzo del trabajo *i.*
f_i	Instante de finalización del trabajo *i.*
D_{ik}	Distancia entre el trabajo *i* y el trabajo *k*, reflejada en unidades de tiempo.

Restricciones:

$$\sum_j x_{ij} = 1 \;\; \forall i \tag{3.5}$$

Obliga a la realización de todos los trabajos por una máquina.

$$x_{ij} + x_{kj} \leq 1 \;\; i = 1 \dots n-1; j = 1 \dots m; \; \{k > i \colon s_k < f_j\} \tag{3.6}$$

Controla que dos trabajos solapados no los realice un mismo recurso.

$$x_{ij} + x_{kj} \leq 1 \;\; i = 1, \dots n-1; j = 1, \dots m; \{k > i; f_i + D_{ik} > s_k\} \tag{3.7}$$

Garantiza el cumplimiento del tiempo de desplazamiento entre un trabajo y otro.

$$\sum_i x_{ij} \leq N. y_j \quad t \in T \tag{3.8}$$

Condición de activación de la máquina *j*.

Función Objetivo:

$$Min \sum_{j=1}^{m} y_j \tag{3.9}$$

Minimiza el número de máquinas utilizadas.

Modelo:

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

$$\sum_{j} x_{ij} = 1 \ \ \forall i$$

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

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

$$\sum_{i} x_{ij} \leq N.y_j \ \ \ t \in T$$

3.3 RESOLUCIÓN

3.3.1 OBSERVACIONES

La resolución del problema se hace basándose en la teoría de grafos, resolución que, con pequeñas adaptaciones, proporciona solución tanto al objetivo operacional como al táctico.

En el apartado siguiente se explica la creación del grafo, proceso idéntico para ambos objetivos. La variación se produce en la inyección de flujo, la cual se especifica para cada objetivo en su correspondiente apartado.

3.3.2 OBJETIVO OPERACIONAL

En primer lugar se construye el grafo, para la formación del cual se define un conjunto de instantes temporales ordenado cronológicamente y formado por los instantes de comienzo y finalización de los *n* trabajos (s_i,f_i) y por un nodo de inicio general (I) y otro de finalización general (F). Cada uno de estos instantes constituye un nodo del grafo.

Los arcos serán los siguientes: unión del nodo de comienzo de cada trabajo con el nodo de finalización del mismo trabajo (es decir, s_i con f_i); unión del nodo de finalización de cada trabajo en su correspondiente destino, con los posibles nodos de inicio de trabajo en otros destinos, siempre y cuando sea posible acceder a éstos teniendo en cuenta los tiempos de traslado (es decir, f_i con s_j $\{f_i+d_{ij}\leq s_j\}$); unión del nodo de inicio general con todos los nodos de comienzo de trabajo (I con si); unión de todos los nodos de fin de trabajo con el nodo de finalización general (f_i con F).

El manejo del flujo que se inyectará más adelante sí es propio del objetivo que se esté tratando, y se consigue imponiendo costes y capacidades. Los arcos del primer tipo (s_i con f_i) tendrán capacidad unitaria, y un coste negativo ($-w_i$) igual al peso de dicho trabajo. El resto de arcos (f_i con s_j $\{f_i+d_{ij}\leq s_j\}$, I con si, fi con F) tendrán capacidad ilimitada y coste nulo. De esta forma, al resolver un problema de *Flujo a Coste Mínimo*[2], será prioritario realizar un trabajo que simplemente trasladarse.

Para el siguiente ejemplo de trabajos (Figura 9):

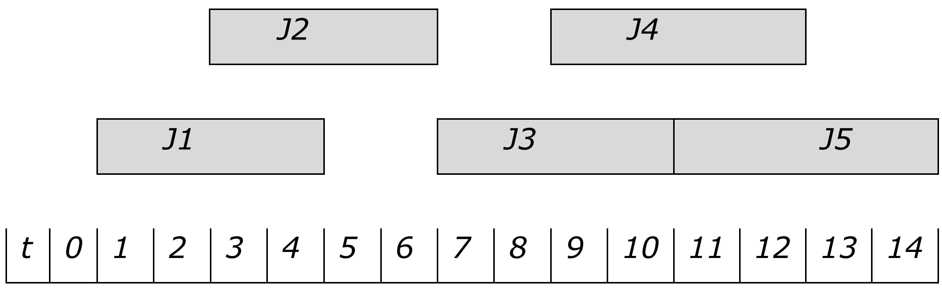

Figura 9

Con la matriz de tiempos (Figura 10):

$$D_{ij} = \begin{vmatrix} 0 & 1 & 1 & 4 & 10 \\ 1 & 0 & 5 & 1 & 3 \\ 1 & 5 & 0 & 2 & 3 \\ 4 & 1 & 2 & 0 & 4 \\ 10 & 3 & 3 & 4 & 0 \end{vmatrix}$$

Figura 10

Dibujando a distintas alturas cada centro de trabajo por aportar claridad a la representación, el grafo sería (Figura 11):

[2] Para más información sobre problemas de Flujo a Coste Mínimo, ver Anexo 2.

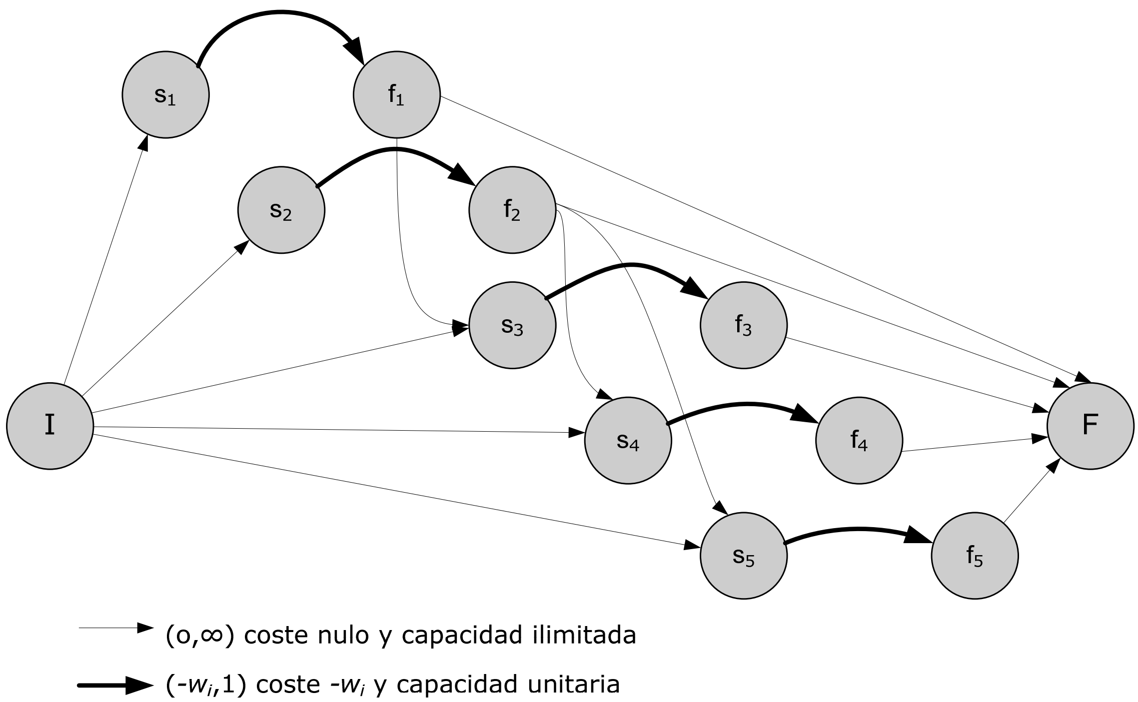

Figura 11

En este problema, por ser de carácter operacional, se debe conocer de antemano el flujo ofertado y demandado por cada uno de los nodos al grafo. Partiendo de esos valores, el flujo es enviado por aquellos arcos de menor coste desde los nodos que ofertan flujo hasta los nodos que demandan flujo, conservándose éste (flujo ofertado total = flujo demandado total). Una vez que todo el flujo inyectado ha sido absorbido, se observan los arcos, y aquellos por los que haya circulado flujo son los de las tareas que sí se realizarán. Esto es los que se conoce como problema de *Flujo a Coste Mínimo*.

En el ejemplo que estamos tratando, se ve a simple vista que con 3 máquinas podrían realizarse todos los trabajos. Pero cuáles trabajos se realizarían si solo dispusiéramos de 2 máquinas?

Supongamos los siguientes pesos (Tabla 3):

Nº de Trabajo	***1***	***2***	***3***	***4***	***5***
Peso (-w_i)	*-3*	*-1*	*-7*	*-1*	*-2*

Tabla 3

Se inyecta un flujo igual al número de máquinas, en este caso 2, en el nodo de inicio I, que será absorbido por el nodo de finalización F. Este flujo recorre los caminos (I, s_1, f_1, s_3, f_3, F) y (I, s_2, f_2, s_5, f_5, F), por ser estos los de coste mínimo, tal y como se muestra en la Figura 12:

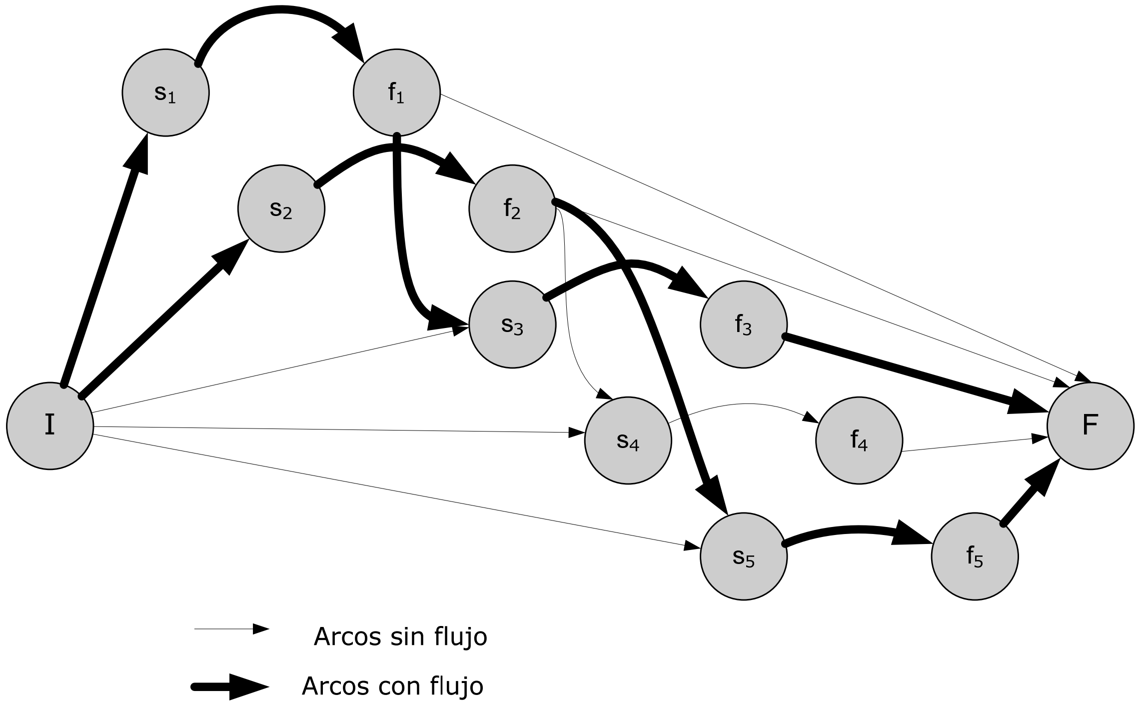

Figura 12

Para probar esta técnica, se plantean casos de hasta 200 trabajos. Los tiempos de resolución obtenidos, usando las librerías RELAXIV[3], son del orden de segundos.

Como complemento a la resolución planteada, se recurre al uso de librerías XA[4], herramienta que, atendiendo a las restricciones y objetivos impuestos en la búsqueda, da la solución óptima. Esta herramienta es de utilidad a la hora de ver el grado de complejidad del problema que se está tratando.

Se plantean casos de hasta 25 trabajos a realizar con cuatro máquinas disponibles, y con un índice de incompatibilidad entre trabajos (por desplazamientos) de 4,86. Esto es decir que cada trabajo es incompatible, por término medio, con otros 4,86 trabajos. Los resultados obtenidos se reflejan en la siguiente tabla, donde se hace referencia al grado de

[3] Para más información sobre librerías de Flujo a Coste Mínimo RELAX IV, ver Anexo 2.

[4] Para más información sobre librerías XA, ver Anexo 1.

solapamiento medio entre trabajos así como al grado de solapamiento máximo entre trabajos en un instante determinado, los trabajos realizados con la solución óptima y el tiempo que tarda en resolverse (Tabla 4).

Grado Medio de Solapamiento	***Solapamiento Máximo***	***Trabajos Realizados***	***Tiempo de Resolución (s)***
2,77	*8*	*17*	*72*
2,87	*10*	*14*	*51*
2,70	*7*	*15*	*308*
2,83	*9*	*14*	*451*
3,50	*10*	*15*	*170*
2,66	*6*	*17*	*196*
2,89	*9*	*14*	*2217*
2,58	*9*	*13*	*2111*
3,19	*8*	*12*	*2729*
3,25	*12*	*11*	*920*

Tabla 4

Cuando la complejidad se aumenta a 50 trabajos a realizar con cuatro recursos, el tiempo de resolución en XA es 24 horas. Tras una serie de pruebas, se concluye que el problema puede resolverse de forma óptima con librerías XA en un tiempo razonable, si tenemos un máximo de 25 trabajos y 4 máquinas.

No obstante, con el uso de técnica de grafos, tal y como se comentó, se obtienen resultados con una velocidad del orden de segundos para problemas más extensos.

3.3.3 OBJETIVO TÁCTICO

Con el mismo grafo utilizado para el objetivo operacional, modificando los costes asociados a los arcos, y sin saber ahora el número m de máquinas necesarias para la realización de todas las tareas, se resuelve el objetivo táctico.

Los arcos del primer tipo (s_i con f_i) tendrán capacidad unitaria, y un coste negativo $-N$ (número arbitrario) asociado al paso de una unidad de flujo por el mismo. El resto de arcos (f_i con s_j $\{f_i+d_{ij}\leq s_j\}$, I con si, fi con F) tendrán capacidad ilimitada y coste nulo. De esta forma, al resolver un problema de

Flujo a Coste Mínimo, será prioritario realizar un trabajo que simplemente trasladarse.

El pseudocódigo de resolución sería (Figura 13):

```
Procedimiento OBJETIVO TÁCTICO
        Do
           m=m+1
           Resuelve problema Flujo a Coste Mínimo
        Until
           Todos los trabajos hayan sido seleccionados
        Fin
```

Figura 13

Para resolver el problema de flujo a coste mínimo, se inyectan *m* unidades en I, las cuales son absorbidas por F. Este procedimiento iterativo termina de ejecutarse cuando hay flujo pasando por todos los arcos que representan la realización de un trabajo.

Para el ejemplo usado en el apartado anterior, el flujo recorrería los caminos $(I, s_1, f_1, s_3, f_3, F)$, $(I, s_2, f_2, s_4, f_4, F)$ y (I, s_5, f_5, F), quedando todos los trabajos realizados con m=3 máquinas (Figura 14).

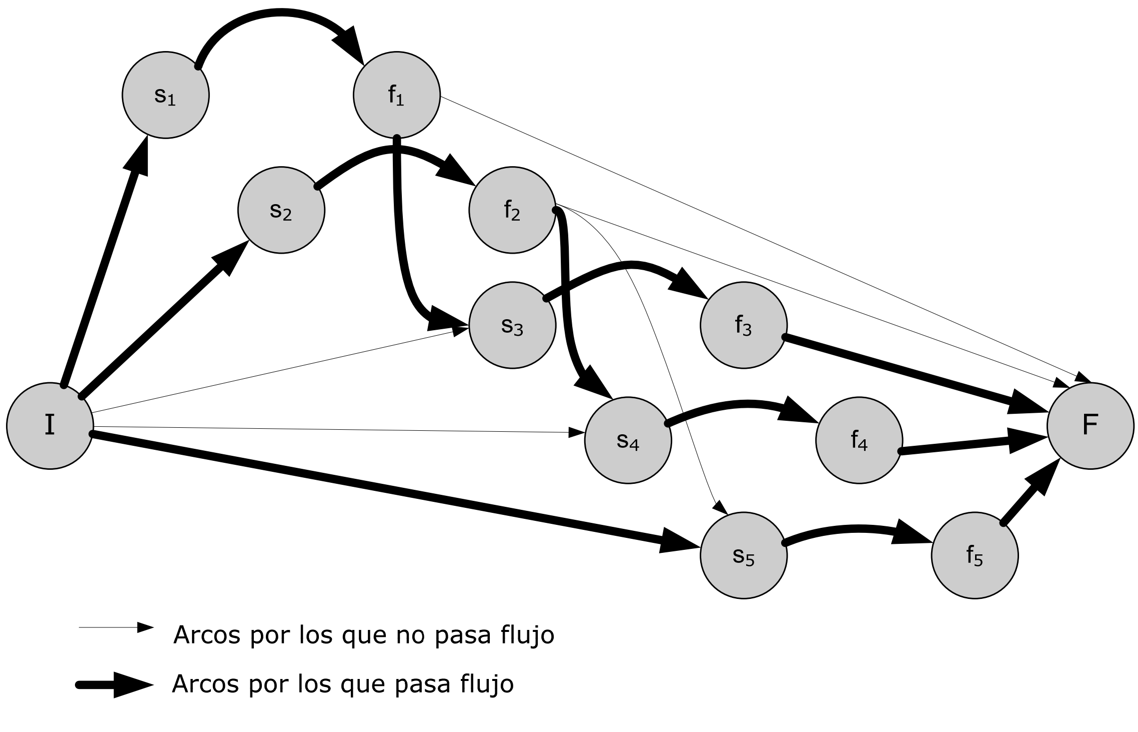

Figura 14

En cuanto a los resultados obtenidos para este objetivo, la teoría de grafos vuelve a ser mejor herramienta que las librerías de optimización, ya que estas últimas arrojan tiempos de resolución notablemente elevados para el objetivo táctico, incluso mayores que para el operacional.