Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería en Organización Industrial

Problemas de Asignación Generalizada: modelización, aplicaciones lógicas y métodos de solución

Autor:
Sesma Gutiérrez, Clara

Tutor: Mata Crespo, Raquel

DPTO: Estadística e Investigación Operativa

Valladolid,

Septiembre 2019.

AGRADECIMIENTOS

Quisiera agradecer en primer lugar a Raquel Mata, mi tutora, pues sin su constante ayuda este trabajo no habría sido posible. Raquel me ha servido de guía durante estos últimos meses, siempre ha estado disponible ya sea mediante tutorías o correos. También considero importante mencionar al profesor Jesús Sáez, cuyas indicaciones contribuyeron a sentar las bases de este trabajo, además de las facilidades que me aportó para realizarlo.

Otro tipo de aportación a este trabajo que considero de igual importancia es la ofrecida por mi familia y amigos, que, durante este trabajo y también durante todo el curso, me han dado su apoyo y consejos.

RESUMEN

A lo largo de esta memoria se analizarán problemas de asignación generalizada (GAP) y sus diferentes variantes (embotellamiento o minimax, MGAP, EGAP, MRGAP, GMAP, GQAP, etc.) partiendo de un conjunto de máquinas o agentes para realizar un conjunto de tareas. Cada tarea debe ser asignada a una máquina o agente y existe la posibilidad de que una máquina realice más de un trabajo (tarea) sin sobrepasar la capacidad máxima disponible en cada máquina que no es necesariamente la misma para todas. Se tendrá en cuenta la productividad de la máquina al asignarle un trabajo o tarea. El problema de optimización consistirá en realizar una buena asignación de los trabajos a los recursos existentes, con objeto de maximizar la producción, teniendo en cuenta las distintas restricciones tanto de los trabajos como de las máquinas.

Este modelo, y sus generalizaciones, se puede asociar a diversas circunstancias en múltiples contextos teniendo en cuenta su aplicación en Ingeniería de Organización para resolver situaciones de perfil muy amplio, por ejemplo, la asignación de personal a máquinas, herramientas a puestos de trabajos, candidatos a vacantes laborales, vendedores a zonas territoriales etc.

En este trabajo, se ha centrado la atención en las aplicaciones logísticas del GAP a diferentes entornos como scheduling, transporte, planificación de la producción y telecomunicaciones, entre otros problemas de optimización combinatorial.

Se abordarán métodos de solución exactos y heurísticos, presentándose los resultados experimentales obtenidos con Xpress Mosel.

PALABRAS CLAVE

GAP, heurísticas, asignación, optimización, búsqueda local.

ÍNDICE

Lista de Figuras

El índice de las imágenes empleadas en el trabajo es el siguiente:

Lista de Tablas

Se enumeran a continuación las tablas presentes en este Trabajo Fin de Grado:

1. INTRODUCCIÓN Y OBJETIVOS

1.1. Motivación

La asignación de recursos es un problema complejo que posee gran número de aplicaciones y en campos de trabajo distintos. El denominado Problema de Asignación Generalizado (GAP), y sus distintas variantes, proporciona modelos de optimización necesarios para la resolución de este tipo de problemas tan presentes en la vida cotidiana. Desde mi óptica personal, me empezaron a interesar las cuestiones sobre programación entera y optimización combinatoria en la asignatura de *Métodos Cuantitativos en Ingeniería de Organización I*, impartida en tercer curso. También utilicé ese mismo curso el entorno de programación de Xpress-Mosel, con el que trabajé durante la citada asignatura resolviendo problemas de optimización. Por otra parte, este curso, en *Métodos Cuantitativos en Ingeniería de Organización II*, profundizamos en la teoría de los métodos heurísticos de optimización. Estos métodos presentan, como es sabido, la posibilidad de que, aun no hallando la solución óptima del problema, se encuentre una alternativa muy interesante a los métodos exactos.

Además, la asignatura de *Estadística* de primer curso me proporcionó las herramientas necesarias para el análisis de datos, estadísticos y representaciones gráficas del trabajo.

Dado el gran número de aplicaciones que en la actualidad tiene el problema de asignación, la resolución de este problema es una importante herramienta para muchas más actividades prácticas relacionadas con este asunto. Dada la complejidad, no obstante, que pueden alcanzar este tipo de problemas, los métodos heurísticos son una herramienta ideal para su resolución, como decíamos anteriormente, puesto que pueden hallar una solución lo suficientemente buena de forma rápida, sencilla y probablemente la más barata.

1.2. Introducción del modelo

La programación lineal es una herramienta de optimización (maximización y minimización) de gran relevancia para la toma de decisiones. Esta área de las matemáticas se emplea actualmente como apoyo imprescindible dentro del desarrollo empresarial. Tiene gran interés como herramienta financiera, y se utiliza también para la optimización de sistemas de producción, transporte, telecomunicación, servicios públicos etc. En resumen, la aplicación de la programación lineal como herramienta de optimización puede adaptarse a un gran número de campos de trabajo.

El Problema de Asignación Generalizada (GAP) es un problema de programación entera cuya finalidad es buscar la asignación óptima de una serie de tareas a un conjunto de recursos de capacidad limitada.

Todo problema de optimización incluye tres elementos que son: las variables de decisión, las restricciones que deben cumplir dichas variables y la función objetivo cuyo valor se quiere maximizar o minimizar.

La formulación GAP es la siguiente:

m: Número de agentes (índice i)

n: Número de tareas (índice j)

x_{ij}: Variables de decisión binarias (1= se asigna la tarea j al agente i, 0=no se asigna la tarea j al agente i)

b_j: Capacidad del recurso j

r_{ij}: Necesidad resuelta si el agente i es asignado a la tarea j

c_{ij}: Coste de asignar la tarea j al agente i

$$Min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \qquad (1)$$

s.a.

$$\sum_{j=1}^{n} r_{ij} x_{ij} \leq b_i \qquad i = 1, \ldots, m \qquad (2)$$

$$\sum_{i=1}^{m} x_{ij} = 1 \qquad j = 1, \ldots, n \qquad (3)$$

$$x_{ij} \in \{0,1\} \qquad \forall \text{ i, j} \qquad (4)$$

La función (1) es la función objetivo a minimizar y representa el coste total tras la asignación. En otros casos, la función objetivo es de maximización y, por lo tanto, los coeficientes c_{ij} indican los rendimientos. En cualquier caso, la función objetivo está sujeta a las restricciones (2), (3) y (4).

La restricción (2) es una inecuación que limita la capacidad de asignar un agente a una determinada tarea. Por otro lado, las restricciones (3) y (4) establecen que toda tarea debe ser ejecutada por un único agente.

La resolución del GAP puede alcanzarse mediante algoritmos exactos, obteniendo la solución óptima. Otra opción es el empleo de métodos heurísticos que, de forma más rápida y sencilla, son capaces de hallar una solución que sea suficientemente buena. Existen métodos heurísticos de búsqueda como el Greedy o algoritmo voraz y el algoritmo Martello - Toth. También se emplean heurísticas de mejora como la búsqueda local que, partiendo de una solución factible como puede haberse obtenido tras el empleo de un método de búsqueda como Greedy, mejoran el resultado acercándose al óptimo. Para la aplicación de la búsqueda es necesario definir estructuras de entornos para el problema, donde sobresalen los entornos shift y swap.

Otros métodos heurísticos muy empleados son el recocido simulado, basado en la mecánica estadística o los algoritmos genéticos, que recrean la idea de la selección natural para hallar una buena solución.

Otra técnica muy empleada en la actualidad debido a su potencia computacional es el empleo de redes neuronales, cuyo objetivo es resolver problemas de igual manera que lo hace el cerebro humano.

1.3. Objetivos

- Formular el Problema de Asignación Generalizada (GAP) y la definición de los componentes que lo conforman.

- Analizar las distintas variantes y extensiones del GAP.

- Enumerar las muchas aplicaciones del GAP dentro de diversos campos.

- Desarrollar métodos heurísticos para la resolución del GAP de forma aproximada.

- Manejar el software Xpress-Mosel para la obtención de resultados e implementación de las heurísticas del GAP.

- Obtener soluciones a varios GAP bien conocidos.

- Análisis estadístico de los resultados obtenidos con los métodos heurísticos en cuanto a tiempo computacional y calidad de la solución con Statgraphics.

- Demostrar la eficacia de la heurística de mejora para hallar una solución al GAP muy próxima a la óptima de forma más rápida y sencilla en base al análisis estadístico de los datos obtenidos.

2. DESARROLLO

2.1. Formulación del modelo

Existen un gran número de problemas con una estructura similar al GAP. Este hecho ha derivado en la creación de una serie de estudios que tratan de clasificarlos. En este trabajo se ha considerado al GAP como un caso especial del WAP (*Weight Problem Assignament*). El WAP encuentra la asignación óptima cuando cada tarea es asignada a un único agente. Además, cada tarea debe ser realizada en un determinado nivel de ejecución; introduciéndose la variable $x_{ijk,}$ cuyo valor es igual a 1, solo si la tarea *j* es completada por el agente *i* al nivel de ejecución *k*, y 0 en caso contrario.

La formulación del WAP es la siguiente:

$$Min \quad f(x) = \sum_{j=1}^{m} \sum_{i=1}^{n} \sum_{k \epsilon k_{ij}} c_{ijk} x_{ijk} \qquad (5)$$

s.a.

$$\sum_{i=1}^{m} \sum_{k \in kij} x_{ijk} = 1 \qquad j = 1, \dots, n \qquad (6)$$

$$a_i \leq \sum_{i=1}^{n} \sum_{k \in kij} r x_{ijk} x_{ijk} \leq b_i \qquad i = 1, \dots, m \qquad (7)$$

$$\sum_{j=1}^{m} \sum_{k \in kj} s_{ijk}\, x_{ijk} \begin{bmatrix} \leq \\ \geq \\ = \end{bmatrix} e_j \qquad j = 1, \dots, n \qquad (8)$$

$$x_{ijk} \in \{0,1\} \qquad \forall \text{ i, j, k.} \qquad (9)$$

La restricción (6) asegura que cada tarea sea realizada por un agente y con un cierto nivel. La restricción (7) marca los límites superior e inferior de consumo de recursos de cada agente mientras que la restricción (8) muestra el nivel de la tarea j una vez completado. Estos límites, como muestra la fórmula, pueden ser mayor que, menor que o igual. Cuando no existe un límite inferior de consumo, es decir, a_i es igual a 0, no existen niveles de actuación, de forma que el problema WAP se transforma en el problema GAP.

2.2. Extensiones y variantes del GAP

El empleo del GAP como herramienta de optimización para todo tipo de aplicaciones ha conducido multitud de variantes del modelo inicial.

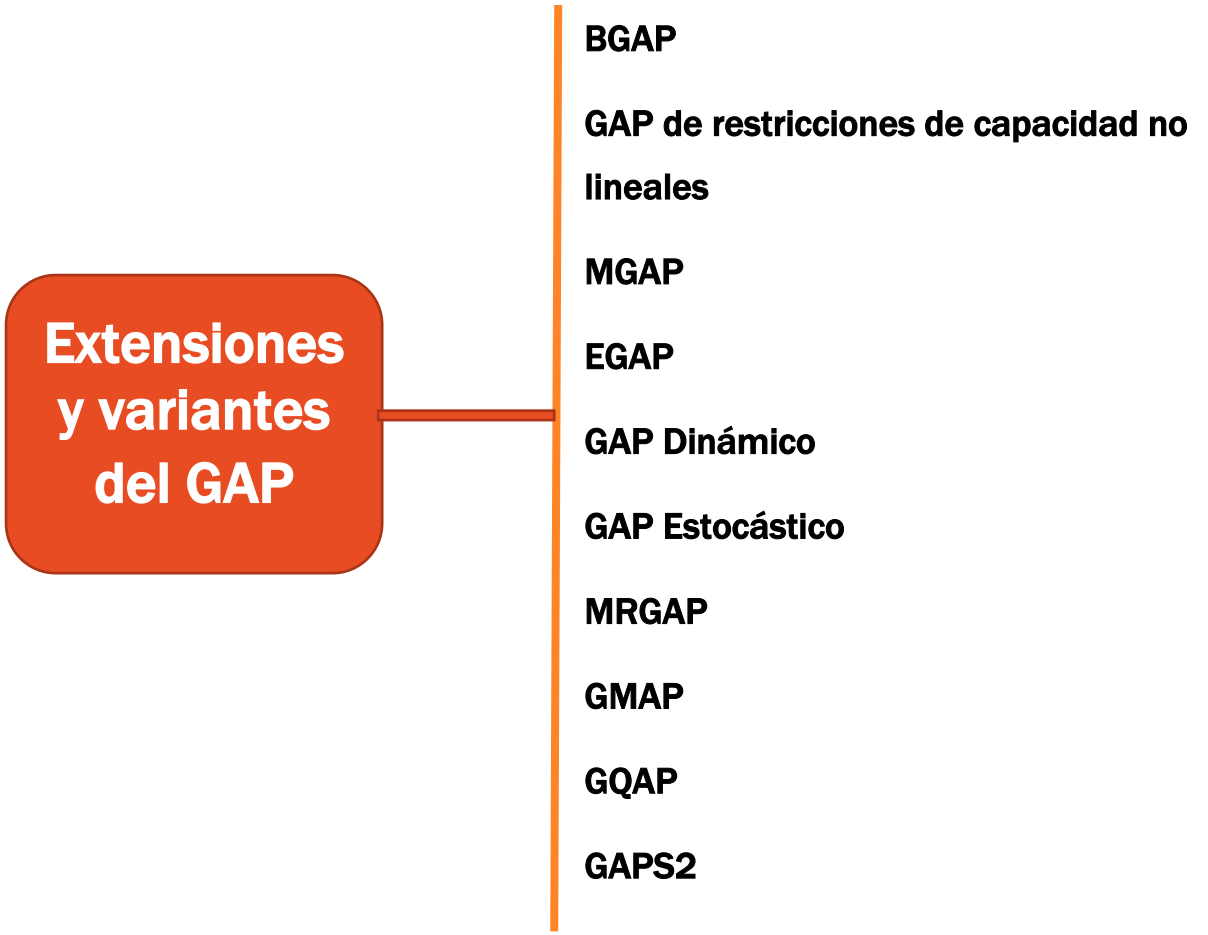

**Imagen 1: Variantes y extensiones del GAP*

2.2.1. BGAP

El BGAP (Bottleneck GAP) se trata de un GAP cuyo objetivo es la minimización de la máxima penalización de una asignación, es decir, un GAP cuya función objetivo es min-max.

$$\max_{i = 1,\dots,m; j = 1,\dots,n} c_{ij}x_{ij} \qquad (10)$$

2.2.2. GAP de restricciones de capacidad no lineales

La diferencia con respecto a la formulación del GAP clásico se encuentra en la restricción (2), que es sustituida por la siguiente restricción de capacidad no lineal:

$$f_i(x_{i1}, \dots, x_{in}) \leq b_i \qquad i = 1, \dots, m \qquad (11)$$

Esta condición de no linealidad puede dar lugar a un intercambio de tareas a lo largo del tiempo.

2.2.3. GAP Multinivel

Este tipo de GAP (MGAP) se creó sobre la base de un problema de asignación de tareas y máquinas a gran escala. Busca el mínimo coste de asignación de

tareas a distintos niveles de eficiencia. La diferencia clave entre el GAP clásico y el MGAP es la existencia de más de un único nivel de eficiencia. El MGAP se considera una variante del WAP donde $a_i=0$. Este problema se compone de las restricciones (5), (6), (11) y añade además la siguiente:

$$\sum_{j=1}^{n} \sum_{k=k_{ij}} r_{ijk} x_{ijk} \leq b_i \qquad i = 1, \ldots, m \qquad (12)$$

2.2.4. GAP Elástico

El EGAP permite a los agentes utilizar más recursos de los que marca su capacidad con la penalización de acarrear un coste adicional. Este problema añade dos nuevas variables u_j y v_j que hacen referencia, respectivamente, a los recursos no utilizados por los agentes y los recursos adicionales que son asignados a los agentes. La función objetivo de este problema es la siguiente:

$$min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} + \sum_{i=1}^{m} (d_i u_i + e_i v_i) \qquad (13)$$

En la formulación del EGAP, las restricciones de capacidad (2) del GAP se sustituyen por:

$$\sum_{i=1}^{n} r_{ij} x_{ij} + u_i - v_i = b_i \qquad i = 1, \ldots, m \qquad (14)$$

y también pueden añadirse los límites de las variables uj u vj:

$$0 \leq u_i \leq g_i \qquad i = 1, \dots, m$$

$$0 \leq v_i \leq h_i \qquad i = 1, \dots, m \qquad (15)$$

2.2.5. GAP Dinámico

El GAP dinámico se diferencia del GAP clásico al considerar las secuencias de trabajo de los agentes. Se tienen en cuenta la jornada laboral de los agentes, la escasez y los costes de inventario. La función objetivo del GAP dinámico es minimizar el inventario, la escasez y los costes de asignación.

2.2.6. GAP Estocástico

Dentro de este GAP se distinguen dos tipos distintos según la aleatoriedad. En el primer tipo, la capacidad de los agentes y la cantidad de recursos necesitados en el proceso no se conocen a priori. En ese caso, la capacidad de los agentes y la cantidad de recursos son modelados mediante variables continuas aleatorias. En el segundo tipo de aleatoriedad la presencia y ausencia de los trabajos son inciertos y los recursos necesarios para cada trabajo se pueden modelar como una variable aleatoria con la distribución de probabilidad de Bernoulli (éxito/fracaso). Como ejemplos ilustrativos, el primer tipo de aleatoriedad correspondería a la asignación de trabajo a programadores, donde el tiempo de creación de un programa no se puede

determinar, ni tampoco el tiempo de ejecución de un programa del ordenador. El segundo tipo de aleatoriedad correspondería, por ejemplo, con la asignación de tareas y recursos llevada a cabo por los servicios de emergencias.

En relación con el segundo tipo de aleatoriedad, puede considerarse aleatoria la ejecución de las tareas una vez que ya han sido asignadas a los agentes. De esta forma, una vez que la demanda es conocida, un trabajo puede reasignarse a otro agente con un coste añadido de reasignación, si el anterior ha excedido su capacidad. Puede ocurrir que la cantidad de recursos totales necesarios sean mayores que la suma de las capacidades de todos los agentes, en ese caso debe penalizarse el coste añadido correspondiente. La función objetivo trata de encontrar el coste mínimo de la asignación y las penalizaciones por reasignación de tareas y por no finalizar un trabajo.

2.2.7. GAP Multi-Recursos

El GAP Multi-Recursos (MRGAP) es otra extensión del GAP donde hay diferentes restricciones de recursos asociados para cada agente. En el MRGAP las restricciones de capacidad (2) del GAP clásico se sustituyen por la siguiente restricción:

$$\sum_{j=1}^{n} r_{ijq} x_{ij} \leq b_{iq} \qquad i = 1, \dots, m;\ q = 1, \dots, Q \qquad (16)$$

Donde cada agente se encuentra restringido por un conjunto de recursos $q=1, \dots, Q$ finitos. Además, para cada agente i y para cada recurso q están

disponibles las unidades de recurso b_{iq} y la tarea j requiere r_{ijq} unidades del recurso q usadas por el agente i. Una aplicación típica del MRGAP surge resolviendo el conocido Problema de Enrutamiento de Vehículos o VRP (Vehicle Routing Problem), donde la capacidad del vehículo se define por volumen y peso. Otras aplicaciones del MRGAP aparecen en sistemas de computación distribuida, programación "job shop", diseño de red de telecomunicaciones o diseño de carga y almacén. Se ha estudiado el MRGAP dinámico, donde la demanda de tareas cambia a lo largo del tiempo y la capacidad de asignación es dinámica. Una variante sería el GAP de capacidad conjunta (CCGAP), en el que las diferentes restricciones de recursos están conjuntamente asociadas con todos los agentes en lugar de con cada agente de forma individual. Una aplicación de este GAP en la vida real se encuentra en el Problema de Asignación de Recursos donde el presupuesto y el equipo están conjuntamente restringidos para todos los agentes.

Existe una extensión del MRGAP: el MRGAP con instalaciones (MRGAPS). La diferencia con el MRGAP reside en que el MRGAPS permite la división de lotes (tareas) entre las diferentes máquinas (agentes). La función objetivo de esta ampliación añade el efecto de la implantación de tiempos y costes. El MRGAPS tiene interés en ambientes de fabricación repetitivos.

Otra aplicación del MRGAP es el problema de asignación de eliminación de nieve o SDAP (Snow Disponsal Assignment Problem). El problema intenta encontrar la óptima asignación de lo lugares donde remover la nieve y los lugares donde almacenarla. Los recursos necesarios para este problema son la capacidad de nieve anual y la tasa máxima de recepción.

2.2.8. Problema de Multiasignación Generalizado

El Problema de Multiasignación Generalizado o GMAP (Generalized Multi-assignment Problem) es una generalización del GAP. En el GMAP, las restricciones de asignación (2) del GAP son sustituidas por:

$$\sum_{i=1}^{m} x_{ij} \geq t_j \qquad j = 1, \ldots, n \qquad (17)$$

Donde t_j es un parámetro tal que $t_j \leq n$ para $j=1, \ldots, n$. El Problema de multiasignación, por lo tanto, cuando $tj = 1$, se convierte en el GAP clásico. Para la resolución de este tipo de GAP se propone una Lagrangiana doble basada en el algoritmo de ramificación y poda. Este problema permite la asignación de un ítem a diferentes mochilas. Una aplicación real de este problema es un sistema de bases de datos distribuida donde cada carpeta es almacenada en múltiples lugares por motivos de seguridad y con rápida velocidad de respuesta. Otra aplicación desarrollada paralelamente sería el software de procesamiento de una instrucción múltiple, flujo de datos múltiple o MIMD (Multiple Instruction Multiple Data Stream). Se trata de un sistema informático donde múltiples copias de programas son ejecutadas desde distintos procesadores conectados en red.

2.2.9. Problema de Asignación Cuadrático Generalizado

El Problema de Asignación Cuadrático Generalizado o GQAP (Generalized Quadratic Assignment Problem), trata la asignación de un conjunto de

instalaciones $j=1, \ldots, n$ a un conjunto de destinos $i=1, \ldots, m$ de modo que se minimiza el coste de asignación y de transporte y el peso total de todas las instalaciones asignadas al mismo destino, no pudiendo exceder su capacidad. En el GQAP la función objetivo (1) del GAP clásico es sustituido por la siguiente función:

$$min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} + \gamma \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{o=1}^{m} \sum_{p=1}^{n} \alpha_{io} \beta_{jp} x_{ij} x_{op} \quad (18)$$

Donde α_{io} es la distancia entre situaciones o e i, β_{jp} es la intensidad de tráfico entre instalaciones p y j y γ es la unidad de coste de trasporte. Una aplicación real del GQAP se encuentra en la determinación de la localización de los equipos dentro de una industria, lo que marca el recorrido de las piezas y por lo tanto el coste de transporte: la función objetivo a minimizar. Otra aplicación del GQAP sería la gestión del patio de contenedores donde el problema es la localización del grupo de contenedores en el área de almacenamiento, de modo que se minimicen las maniobras de movimiento.

2.2.10. GAP con Conjuntos Ordenados Especiales TIPO II

Un conjunto de variables $(x_1, x_2, \ldots, x_n)$ es un conjunto especial ordenado de tipo II si $x_i x_j = 0$ siempre que $|i-j| \geq 2$. El GAP con Conjuntos Ordenados Especiales de Tipo II, o GAPS2 (GAP Special Ordered Sets of Type II) trata de la asignación de tareas a periodos de tiempo. La variable de decisión x_{ij} indica

la fracción de tareas j asignadas a un periodo de tiempo i. El GAPS2 se obtiene reemplazando las restricciones (4) del GAP clásico por una de las siguientes funciones:

$$\{x_{i-1j} = 0 \; and \; x_{i+1j} = 1 - x_{ij}\}$$

$$or$$

$$\{x_{i+1j} = 0 \; and \; x_{i-1j} = 1 - x_{ij}\} \qquad (19)$$

Es importante resaltar que, en el GAPS2, cualquier tarea puede ser ejecutada dentro de un periodo de tiempo, pero también podría dividirse entre dos periodos de tiempo consecutivos.

Una aplicación del GAPS2 sería la programación de la producción de cables de fibra óptica donde se permite compartir los trabajos entre periodos de tiempo adyacentes. Una propuesta de método de solución exacto para este problema se basa en aproximaciones poliédricas, con las cuales se han resuelto pequeños ejemplos de optimización. También se ha propuesto un sencillo método heurístico de programación lineal para la resolución del GAPS2.

Una aplicación del GAPS2 podría surgir de la acumulación y/o distribución de la carga de vehículos donde está permitido dividir la carga entre dos localizaciones vecinas, pero no está permitido realizar sucesivas operaciones de carga/descarga en un viaje determinado. Este problema podría surgir debido a las restricciones del tiempo de operaciones de carga/descarga de bienes perecederos.

2.2.11. GAP Biobjetivo

Para la resolución del GAP Biobjetivo o BiGAP (Biobjetive GAP), existe un método heurístico de programación lineal. La función objetivo de este problema es la siguiente:

$$max \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}^{l} x_{ij} \qquad l = 1,2. \qquad (20)$$

Una aplicación del BiGAP se encuentra en la planificación del sistema de producción, donde - c^1_{ij} y $-c^2_{ij}$ representan respectivamente el coste y tiempo cuando el trabajo j es asignado a la máquina i. Otra aplicación sería en la planificación de las zonas de los servicios de emergencia, donde - c^1_{ij} y $-c^2_{ij}$ representan respectivamente las ganancias y los costes de los transportes debidos a la asignación de la zona j (cliente) a la unidad de servicio i (vehículo).

2.3. Aplicaciones relacionadas con el GAP en la vida real

Se han mencionado anteriormente varias aplicaciones del GAP. A continuación, se tratarán algunos de los problemas ya mencionados y otras aplicaciones que forman parte de problemas más densos.

2.3.1. Aplicaciones en la programación

Muchas de las aplicaciones del GAP tienen lugar en los problemas de programación. Programación de empleados, máquinas, tareas de multiprocesadores, planificación de la mano de obra, aulas, lotes etc.

Un problema de programación que incluye al GAP como un subproblema se desarrolla dentro de un proyecto de redes. Por ejemplo, en el Problema de Asignación del Trabajo con restricciones no preventivas de recursos. El objetivo es encontrar el coste mínimo de la asignación de un conjunto de recursos (trabajadores) a un periodo limitado de tiempo. Trabajos y recursos están considerados respectivamente como ítems y como mochilas.

El Problema de Equilibrio de Carga es otra extensión del GAP. El problema es encontrar la asignación óptima de un conjunto de trabajos a un conjunto de máquinas. Pueden usarse diferentes tipos de función objetivo para este problema: la minimización del makespan, la minimización de la media del tiempo de flujo, o la maximización de la equidad en el trabajo asignado. Una propiedad importante de este problema es que cada trabajo se asume que tiene una unidad de tiempo de procesamiento.

2.3.2. Aplicaciones en el Transporte y creación de rutas

A principios del siglo XX se empleó el GAP para el transporte de pacientes entre hospitales militares en los Estados Unidos. Con el GAP se determinó la asignación de pacientes con vuelos. La función objetivo buscaba minimizar molestias a los pacientes, sabiendo el número de días que los pacientes pasarían la noche en el hospital. El segundo objetivo para minimizar era el tiempo de vuelo o longitud de la ruta.

El GAP aparece como subproblema en el conocido VPR (Problema de Enrutamiento de Vehículos), mediante la asignación de ciudades a posiciones preseleccionadas. También para el problema TSP (problema del vendedor viajero).

Como aplicación dentro de la política de una empresa se propone el TP, un caso especial del problema de transporte y, por lo tanto, una variante del GAP. El TP se basa en que cada punto de demanda (cliente) requiere satisfacer sus necesidades (proveedor) de un único recurso, sin exceder la cantidad máxima de recursos que el proveedor puede suministrar. El objetivo es minimizar el coste total asociado al material transportado de los recursos asignados a los puntos de demanda. Para este caso, la restricción de que cada cliente debe ser asignado a un único recurso se cumple con la fórmula (3). Sin embargo, particularidad de este problema radica en que $r_j=r_{ij}$ para todo $i=1, ..., m$ y $j=1, ..., n$.

En 1997 surgió una variación del GAP a causa de las actividades de una industria en Nueva Zelanda: el Problema de Asignación Cubierto o CAP (Covering Assignment Problem). El problema se desarrolla por la demanda diaria de leche que las compañías suministraban de diferentes granjas. El CAP determinó la localización granjas (proveedores) a industrias (clientes), con el mínimo coste de transporte, de manera que cada granja fuera asignada a una única industria y la demanda de cada industria pudiera satisfacerse. En el CAP la restricción de los parámetros $r_j=r_{ij}$ para todo $i=1, ..., m$ y $j=1, ..., n$ se mantiene y la restricción (2) pasa a ser de tipo "$\geq$".

2.3.3. Aplicaciones en Telecomunicaciones

La aplicación del GAP en telecomunicaciones tuvo lugar por primera vez en 2003 para optimizar el protocolo de vuelta de enlace de frontera o BGP (Border Gateway Protocol) y su uso en el enrutamiento de dominios minimizando el coste de enrutamiento del tráfico. El BGP juega un papel importante en el control del flujo de tráfico entre clientes y proveedores

Otra aplicación del GAP en telecomunicaciones, creada como una extensión del GAP, es la máxima cobertura de multiplexación de código de acceso a red de telecomunicaciones con restricciones de capacidad (potencia y flujo). El problema trata de encontrar la asignación óptima de terminales a estaciones base.

2.3.4. Aplicaciones en la Planificación de la Producción

Hay muchas aplicaciones del GAP dentro de la Planificación de la Producción. El problema NP-Complejo de Carga y Programación de Lotes, puede dividirse en dos problemas anidados: carga de lotes y programación de lotes. El Problema de Cargas por Lotes o BLP (Batch Loading Problem) con una secuencia de lotes determinada, halla la asignación óptima entre trabajos y lotes. El Problema de Programación de Lotes es el problema de secuenciación de lotes. El mencionado BLP es un caso especial del GAP con $a_{ij} = r_{ij}$ donde a_j es el volumen de trabajo j y $b = b_j$ donde b es del procesador, $x_{ij} = 1$ solo si el trabajo j es asignado al lote i. Sin embargo, la función del coste del BLP es mucho más complicada y difícil de manejar que la función de coste del GAP.

El GAP también tiene aplicaciones dentro de la Tecnología de Grupo o GT (Group Technology). Relacionado con la de asignación de máquinas y los problemas de formación de células. En el Problema de Asignación de Máquinas o MAP (Machine Assignment Problem) el objetivo es minimizar el coste de utilización del emplazamiento de las máquinas y el coste de movimiento entre células. La primera restricción del MAP sería que al menos un tipo de máquina debe estar ubicada en cada célula. La segunda restricción impondría que el tiempo total de operación del conjunto de máquinas asociadas a una máquina sea menor que el tiempo disponible de operación de la célula. Una de las soluciones factibles para el MAP se basa en la estructura del GAP, mediante la resolución de un problema mochila para cada célula. El objetivo de la función es cuadrático y minimiza el coste de los movimientos entre células. Las restricciones son exactamente las mismas que el GAP (2)-(4) con $r_{ij} = 1$.

Una gran aplicación del GAP surge en GT donde la división eficiente de la fabricación de piezas en familias, llamado Problema de Formación de Grupos o GFP (Group Formation Problem), el cual juega un papel importante en todos los sistemas de evaluación. Este problema es una generalización del Problema de Generalizado de Formación de Grupos o GGFP, equivalente al GAP. Ambos, GFP y GGFP, intentan asignar cada pieza a una única familia de piezas, maximizando la suma total de similitudes entre piezas asignadas a una familia.

Una variante del GAP encuentra aplicación en los sistemas de recuperación y almacenamiento automatizados (AS/AR). Suponiendo como datos conocidos al conjunto de tipos ítems pedidos y la frecuencia de procesamiento de pedidos durante el horizonte de planificación del conjunto de ítems, cada pedido se almacena independientemente a un tiempo y conjunto de movimientos determinados de la máquina de recuperación y almacenamiento. La máquina de recuperación y almacenamiento es capaz de realizar un movimiento simultáneo vertical y horizontal y cada localización de almacenamiento es uniforme y puede contener solo un tipo de ítem.

El Problema de Distribución de Almacenamiento consiste en optimizar la determinación de las localizaciones de los pedidos durante el horizonte de planificación, de modo que cada localización de almacenamiento contenga un único ítem y cada ítem i sea asignado a un número fijo de localizaciones. La función objetivo busca minimizar el total de tiempo de recogida de pedidos por periodo de la máquina de recuperación y almacenamiento.

El Problema de Selección de Orden Multicriterio en Sistemas de Manufacturación Flexible (FMSs) trata de encontrar la asignación de un conjunto de pedidos a un conjunto de periodos minimizando costes debidos tanto al tiempo como a la subcontratación. Los costes por anticipación y por tardanza se calculan empleando la fecha de vencimiento. El coste de subcontratación se añade cuando el pedido no es asignado durante el horizonte de planificación (el conjunto de los periodos) por las restricciones de capacidad. Las restricciones de capacidad se deben a la capacidad de la máquina y del almacén de herramientas. Para adoptar técnicas de relajación Lagrangiana eficientes se ha propuesto una formulación de programación binaria del problema original, considerando la subcontratación añadiendo un periodo virtual al horizonte de planificación y redefiniendo por consiguiente los parámetros de la función objetivo.

El Problema de Programación de Lote Económico para múltiples productos en procesadores paralelos idénticos o MELSP (Economic Lot Scheduling Problem) es otro problema donde el GAP actúa como un subproblema durante el proceso de solución. El MELSP determina el óptimo tamaño de lotes y programa para un número de productos, varios procesadores paralelos idénticos. Esto convierte al MELPS en un GAP no lineal. Para la resolución de este problema se propone un algoritmo heurístico, el cual resuelve el GAP clásico en cada iteración.

2.3.5. Aplicaciones de Localización

Existen problemas para determinar la ubicación de instalaciones públicas y privadas, que pueden ser modelados como un GAP: Problema p-mediana, problema p-mediana capacitado, problema de localización de instalaciones no capacitado, problema de localización de instalaciones capacitado y problema de localización de instalaciones con elección del tipo de instalación. También puede formularse como un GAP el problema de máxima cobertura de ubicaciones.

Uno de los problemas de localización en el que el GAP se desarrolla como un subproblema es en la localización de p-centro capacitado, instalaciones múltiples bajo la capacidad basada en economías de escala, localización capacitado de una sola fuente, p-mediana capacitado y el problema de localización de concentradores.

El problema de localización de centros públicos multicriterio es otro problema donde los parámetros del GAP se resuelven como un subproblema. El problema se basa en seleccionar un subconjunto de m sitios para establecer las instalaciones públicas, de modo que cada uno de los n clientes son atendidos por una única instalación. La función objetivo busca minimizar los costes, efectos peligrosos, tiempos de respuesta y la distancia de recorrido.

2.3.6. Aplicaciones de Logísticas de cadena de suministro

Una aplicación del GAP se encuentra en las decisiones de suministro de caña de azúcar para maximizar los ingresos netos de una región de fábricas de azúcar en el horizonte de planificación. El problema es encontrar el óptimo intervalo de tiempo de cosecha de un conjunto de granjas dentro del horizonte

de planificación. A parte de las restricciones de capacidad no lineales relacionadas con las toneladas máximas a transportar y cosechar, también se definen un conjunto de restricciones con respecto a la gestión del suministro. Además, el modelo posee una función objetivo no lineal.

Otra aplicación del GAP tiene lugar en el problema de partición de la demanda o DPP (Demand Partitioning Problem). El DPP busca encontrar el mínimo coste de asignar una serie de productos a un conjunto proveedores donde la demanda de cada producto solo puede satisfacerse por un único proveedor. La función objetivo del DPP la cual es más compleja que la función perteneciente al GAP trata de minimizar el coste de mantenimiento del inventario de un producto i y el coste de pedidos pendientes de un producto i por periodo de tiempo y los costes de producción y transporte de un producto i por suministrado por un proveedor j. Cuando los costes de pedidos pendientes y de mantenimiento son cero en el problema DPP, la función objetivo es la misma que en el GAP.

El MPSSP (Multi-Period Single Sourcing Problem) intenta optimizar la asignación de horas extra, de un conjunto de clientes a almacenes de tal modo que cada cliente sea asignado un único almacén en cada periodo. La función objetivo busca minimizar los costes de transporte y de inventario.

El TPRP (Third Party Routing Problem), el cual es una variación del VRP, puede formularse como el GAP con restricciones añadidas.

Otros problemas que guardan íntima relación con el GAP son: el conocido como “Market Clearing Problem”, el problema de la mochila múltiple (Multiple Knapsack) y el BPP (Bin Packing Problem), problema de empaquetado de objetos.

2.3.7. Otras Aplicaciones

El problema de división de una base de datos central y asignación de esas divisiones entre procesadores de un sistema de control distribuido. En un sistema de control distribuido la información se transmite mediante procesadores interconectados. Sin embargo, cuando surge un interrogante en uno de los procesadores, este es respondido por el procesador local, si es posible. A parte de eso, la cuestión es reenviada a otros procesadores. El problema es optimizar la localización de las divisiones de la base de datos de tal modo que el coste total de comunicación sea mínimo, y que no se exceda la capacidad de procesamiento, almacenamiento y de comunicación del procesador. Este problema surge en las redes bancarias y su formulación es un caso especial del GAP llamado TP. Una propuesta para resolver este problema es mediante la aproximación al problema de la mochila de múltiples restricciones, un caso especial del GAP.

La programación de las tareas durante una misión de medio año del telescopio espacial internacional ROSAT fue resuelto mediante el GAP. El problema trata de asignar objetos para ser observados a intervalos de tiempo. El objetivo es maximizar el tiempo de observación para todos los objetos.

El LAP (Land-use Allocation) es otro caso especial del GAP. Dadas n parcelas de tierra y m actividades sobre el uso de la tierra, el LAP consiste en encontrar la asignación óptima de cada parcela de tierra a una actividad como máximo, de tal modo que se satisfaga el nivel de demanda de cada actividad. El objetivo es minimizar la distancia total disponible (coste) de la asignación de parcelas de tierra hasta su valor ideal.

Problema de asignación de trabajadores polivalentes a entornos de servicios multidepartamentales como una variante del GAP no lineal. La función objetivo es maximizar la función de la utilidad cóncava asociada con la asignación entre trabajadores y departamentos. La programación de los sistemas sanitarios, especialmente la programación de los enfermeros es una importante aplicación real para este problema.

Un problema de gestión de potencia el cual busca encontrar la asignación de tareas a niveles de voltaje de tal modo que se minimice la energía disipada en el sistema. El problema se plantea en forma de GAP y existe un método heurístico para hallar su solución.

En la industria nuclear, existe un problema de gestión de existencias que es una extensión del GAP que se basa en la asignación de p lotes de plutonio almacenados a n pedidos. Para cada lote las restricciones (2) son referidas a diferentes características (masa, composición isotópica y valor energético). Una diferencia entre este problema y el GAP clásico es la existencia de un límite mínimo requerido para una de las restricciones (valor energético).

2.3.8. Tabla resumen

La siguiente tabla recoge los campos ya vistos donde puede aplicarse el GAP y algunos problemas particulares, a modo de resumen.

Campo de aplicación	Problema
Programación	Equilibrio de Carga
Transporte	Problema de Asignación Cubierto
Telecomunicaciones	Optimización del BGP
Planificación de la producción	Problema de Asignación de Máquinas
Localización	Problema de Localización de centros públicos
Logística	Problema de partición de la demanda
Otras aplicaciones	Programación del telescopio ROSAT

Tabla 1: Aplicaciones del GAP

2.4. Procedimientos de resolución del GAP

El GAP se trata de un problema NP-hard puesto que no existe ningún algoritmo de tiempo polinomial que pueda encontrar una solución factible. La aproximación a una solución exacta solo puede obtenerse para problemas sencillos. Para superar las limitaciones que conllevan los métodos exactos, se han creado tanto aproximaciones heurísticas como metaheurísticas. El método de búsqueda local, utilizado por la metaheurística, explora el espacio solución sin perfeccionar necesariamente la función objetivo y permitiendo en ocasiones movimientos no factibles.

Han sido aplicadas muchas aproximaciones provenientes de las últimas metaheurísticas a variables de búsqueda local y de procedimientos para hallar una solución exacta para el conocido problema de optimización combinatorio.

Para hallar la solución óptima en el GAP existen alternativas como la utilización de programas con licencia, tales como el Xpress-Mosel. Estos programas son capaces de hallar una solución óptima y entera a problemas grandes y complejos de optimización.

2.4.1. Relajación lineal

Un problema relajado es aquel en el que las variables binarias se sustituyen por variables continuas. La técnica de la relajación lineal se puede describir de manera sencilla como una cota capaz de aportar gran cantidad de información. Este método aplicado al problema de asignación sustituye la restricción (3) del GAP clásico por una restricción de no negatividad en las variables continuas ($0 \leq x_{ij} \leq 1$), convirtiendo el problema en la relajación lineal llamada LGAP.

Para resolver este problema se puede recurrir a aproximaciones o redondeos o construir métodos más elaborados añadiendo restricciones que eliminen las

soluciones fraccionales obtenidas. Esta segunda opción lleva a la generación de un ciclo.

Para hallar soluciones más precisas en la relajación lineal se utiliza el método de Ramificación y Acotamiento (Branch & Bound). Este algoritmo genera dos subproblemas a partir de la solución no factible (fraccionaria) mediante la adición de restricciones. El algoritmo continuará creando y descartando la solución de los subproblemas hasta dar con una solución óptima factible.

En general la relajación lineal, al ser un problema de programación lineal, es mucho más fácil de resolver que el problema original de programación entera. En Xpress-Mosel, si en vez del óptimo entero estamos interesados en el óptimo lineal (o valor de la relajación lineal), para maximizar un objetivo con nombre ganancia_total basta sustituir el comando maximize(ganancia_total) por un comando como: maximize(XPRS LIN,ganancia_total).

2.4.2. Heurística de redondeo

Esta heurística trata simplemente de buscar la solución menor entera al problema. Se trata de un algoritmo muy sencillo que no suele proporcionar una buena solución, de hecho, por lo general se suele encontrar una solución bastante alejada del óptimo.

El pseudocódigo implementado dentro del programa es el siguiente:

```
forall (i in maquinas, j in tareas) xp(i,j) := integer (floor (x(i,j).sol))

zp: = sum (i in maquinas, j in  tareas) p(i,j) * xp(i,j)

writeln ("\n\tzp = ",zp)
```

Siendo floor una función propia de la librería de Xpress que toma la parte entera de la solución al problema (.sol).

2.4.3. Heurística MT de penalizaciones

El método de penalizaciones se conoce también como el algoritmo Martello - Toth. Es una heurística voráz o *greedy*, la cual en cada paso escoge la solución óptima.

La heurística *greedy* avanza sin tener en cuenta las consecuencias posteriores, únicamente toma la solución óptima en cada momento, por lo que se dice que es un algoritmo miope.

Este método utiliza las siguientes notaciones:

Conjunto de tareas: $N = \{1, \ldots, n\}$

Conjunto de agentes o máquinas: $M = \{1, \ldots, m\}$

Conjunto de tareas sin asignar: $\mathrm{U} \subseteq \mathrm{N}$

Número de tareas asignadas: na

Capacidad del agente i: $\mathrm{b_i}$

Agente asignado a la tarea j: $\mathrm{y_j}$

Valor objetivo de la solución heurística: z

Variable binaria que indica si una solución es factible (1) o no (0): feas

El algoritmo se desarrolla en tres etapas:

1. Inicialización. El algoritmo comienza con $\mathrm{na} = 0$, $\mathrm{z} = 0$ y $\mathrm{feas} = 1$.

2. **Finalización. Si** $\text{feas} = 1$ **y** $\text{na} = \text{n}$ **el método da la solución final. Si** $\text{feas} = 0$ **el algoritmo termina sin haber hallado una solución factible. La tercera posibilidad se da cuando** $\text{feas} = 1$ **y** $\text{na} < \text{n}$ **el algoritmo continúa con la siguiente etapa.**

3. **Penalizaciones. Si no hay ningún agente con la capacidad suficiente para realizar la tarea** j $(\text{nk}_\text{j} = 0)$ **entonces** $\text{feas} = 0$ **y el algoritmo termina sin haber encontrado una solución factible. Si** $\text{nk}_\text{j} = 1$ **la penalización** p_j **toma el valor** ∞**. Si** $\text{nk}_\text{j} \geq 2$, p_j **se define como la diferencia entre el segundo menor y el menor coeficiente** c_{ij}**. La tarea** j^* **con la penalización** p_j **máxima se asigna entonces a la máquina** i^* **con menor coeficiente** c_{ij}**.**

2.4.4. Métodos de búsqueda local

Los procedimientos de mejora basados en estrategias de búsqueda local son los más usados. Este tipo de métodos se encargan de buscar la mejora de una solución inicial factible, como la que puede obtenerse tras aplicar, por ejemplo, un método greedy como el anterior. El método se basa en la exploración de un entorno mediante movimientos. Estos movimientos son operaciones que el algoritmo emplea sobre la solución para hallar soluciones de su entorno.

Para un problema de optimización que busca la minimización de una función $\text{f}(\text{x})$, se considera que el x pertenece al conjunto X que representa el conjunto de todas las soluciones factibles. Un mínimo global es una solución $\text{x}^* \in \text{X}$ cumpliendo que $\text{f}(\text{x}*) \leq \text{f}(\text{x}), \forall \text{x} \in \text{X}$.

Bases de la búsqueda local:

1. Definición de entorno: Cada solución $x \in X$ es un mínimo local respecto del sistema de entornos $N(x) \subseteq X$, que denominaremos entorno de x.

2. Definición de mínimo local: Una solución x* Є X es un mínimo local respecto del sistema de entornos N(x), si se verifica que f(x*) ≤ f(x) ∀x Є N(x*).

3. Definición de movimiento: Dada una solución x Є X, cada solución en su entorno x' Є N(x) puede obtenerse directamente a partir de x mediante la operación llamada movimiento.

Un procedimiento de búsqueda local parte de una solución inicial x_0, examina su entorno $N(x_o)$ y escoge una nueva solución x_1 Є $N(x_0)$, es decir, realiza un movimiento. Este proceso puede aplicarse de forma reiterada, obteniendo una trayectoria:

$$x_1 \rightarrow x_2 \rightarrow \ldots \rightarrow x_n$$

Si un método de búsqueda por entornos solo permite movimientos en el entorno de la solución inicial para hallar el óptimo, se denomina un método de descenso. El método de descenso procede de la siguiente forma:

1. Se escoge x Є X para iniciar el proceso.

2. Se busca x' Є N(x) tal que f(x') < f(x).

3. Si no se puede encontrar un x' Є N(x) tal que f(x') < f(x), se termina, pues x es un óptimo local.

4. En otro caso, se sustituye x por x' y se vuelve a la etapa 2.

Un método de descenso finaliza en un óptimo local, que es mejor o igual que todas las soluciones de su entorno. El defecto de este método es que no garantiza la obtención de un óptimo global. Pueden existir un gran número de óptimos locales y el método de descenso no permite determinar si a su vez también se tratan de un óptimo global.

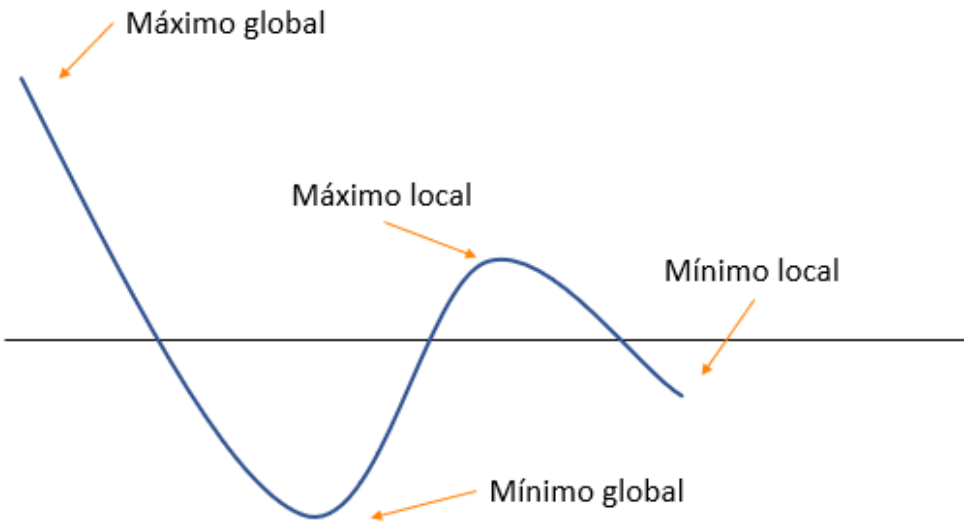

Imagen 2: Máximo global y local

Existen distintas versiones del método de descenso. Una de estas versiones se trata del método de mayor descenso (*steepest descent o best improvement*). El método del mayor descenso requiere examinar completamente el entorno de la solución x para obtener el menor $f(x')$ sobre $x' \in X$. Si N(x) contiene muchos elementos, puede ser preferible un método

llamado *first improvement*, que consiste en seleccionar el primer movimiento que produce una mejora de la solución actual. En cualquiera de sus versiones este método termina en un óptimo local.

Para realizar un método de búsqueda local es necesario la definición previa de una o varias estructuras de entornos. Los entornos que se utilizan son:

1. Entorno Shift o de cambio. Este movimiento consiste en, dada solución, cambiar la asignación de una tarea a otro agente diferente.

2. Entorno Swap o de intercambio. Este movimiento consiste en el intercambio de las asignaciones a dos tareas.

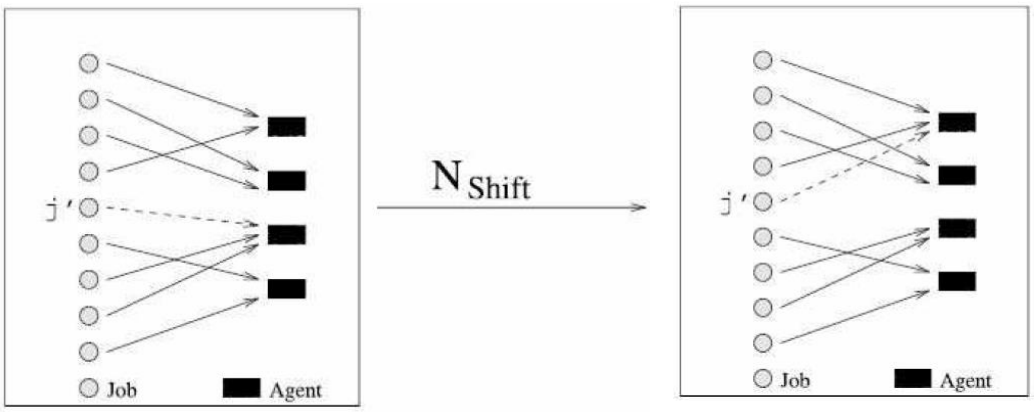

**Imagen 3: Explicación gráfica del entorno Shift*

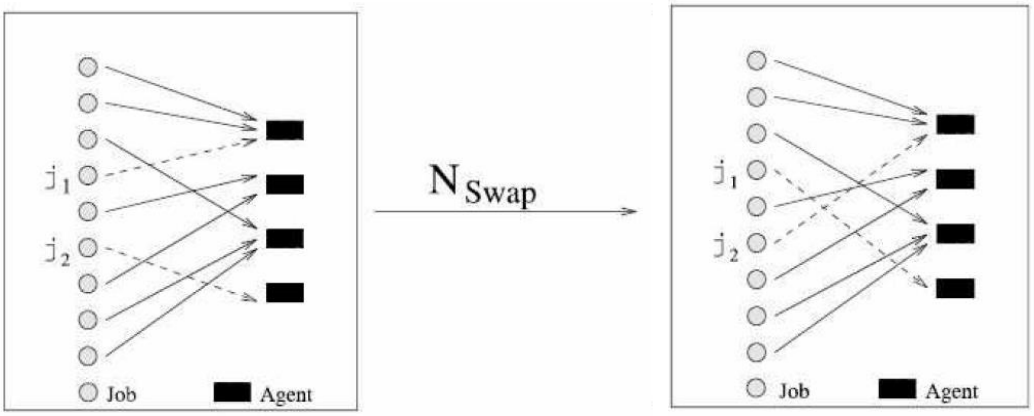

**Imagen 4: Explicación gráfica del entorno Swap*

La utilización de ambos entornos permite diseñar algoritmos heurísticos para el GAP, comenzando con la solución obtenida de la heurística de Martello-Toth y aplicando posteriormente la búsqueda local.

En algunas ocasiones es posible garantizar que un óptimo local se trata a su vez de un óptimo global mediante un método de descenso. Si la función objetivo $f(x)$ es convexa y el conjunto factibles X es convexo, todo mínimo local es un mínimo global y un método de descenso converge al mínimo global.

Un método de búsqueda local necesita, además de la definición la estructura de los entornos, un criterio de selección de una solución dentro del entorno.

La definición de entorno o movimiento depende de la estructura del problema a resolver y de como sea la función objetivo.

El pseudocódigo del procedimiento de búsqueda local con intercambios y best-improvement en lenguaje Mosel es el siguiente:

```
final:=0
while(final = 0)do
    mejora_max:=-M (siendo M un número suficientemente grande)
    forall(j, k in 1..n|x(j)=1 and x(k)=0)do
        x1(j):=0
        x1(k):=1
        forall(i in 1..n|i<>j and i<>k x1(i):=x(i)      !se calcula f(x1)
        mejora:= f(x) - f(x1)
        if(mejora > mejora_max)then
            mejora_max:= mejora
            jmax:=j
            kmax:=k
        end-if
```

end-do

if(mejora_max <=0)then

final:=1

else ! se hace el intercambio:

x(jmax):=0

x(kmax):=1

end-if

end-do

Donde $\mathrm{f}(\mathrm{x})$ es la función objetivo a minimizar y la solución está dada mediante un vector x Є $\{0,1\}^n$.

En el caso de aplicar los dos entornos, pueden aplicarse de forma secuencial o con un método de descenso VNS o Búsqueda por entornos variables (*Variable Neightbour Search*). El método VNS se basa en:

1. Un mínimo local para una estructura de entornos no lo es necesariamente para otra.

2. Un mínimo global tiene que ser un mínimo local para todas las posibles estructuras de entornos.

El método llamado "Búsqueda por Entornos Variables Descendientes" se trata de encontrar un mínimo local respecto a todas las estructuras de entorno. Se inicia con una solución x y se toma $\mathrm{k}=1$. Repetir hasta que $\mathrm{k}=\mathrm{p}$ los pasos:

1. Exploración del entorno. Encontrar la mejor solución x' del k-ésimo entorno $N_K(x)$.

2. Decisión de moverse o no. Si la solución obtenida x' es mejor que x, se toma $x = x'$ y $k = 1$. En otro caso, se toma $k = k + 1$.

Para este método se suelen emplear $p=2$ o $p=3$ entornos. Existen otras variantes como VNS Reducida, VNS Básica, VNS General o VNS Anidada. Sin embargo, de ninguna forma, este método garantiza que el óptimo local sea también el óptimo global del problema.

El problema que presentan los métodos de búsqueda local es la posibilidad que quedarse atrapado en un óptimo local que se distancie enormemente del óptimo global del problema. Este hecho hace imprescindible añadir un mecanismo que impida la creación de ciclos, en el cual aparezcan soluciones ya obtenidas. También es necesario establecer un criterio de parada, pues el procedimiento podría iterar indefinidamente.

Los procedimientos metaheurísticos solucionan los problemas anteriores y además son capaces de conducir la búsqueda de forma inteligente.

Estos métodos suelen ser muy rápidos y proporcionan soluciones relativamente cerca del óptimo global. Además, se suelen emplear métodos de búsqueda local para inicializar otras metaheurísticas más complejas como la Búsqueda Tabú o el Recocido Simulado, que se considerarán en las futuras extensiones de esta memoria.

3. ESTUDIO COMPUTACIONAL

3.1. Implementación en Xpress-Mosel

El Xpress-MP es un entorno de modelización matemática y optimización con la capacidad de resolver problemas de programación lineal, entera y cuadrática. Este entorno posee además un gran número de herramientas como la generación de gráficos o la posibilidad de conectarse a una base de datos y además cuenta con diversas librerías (C, Java, etc.). La interfaz que se utilizará del programa será Xpress-IVE.

Imagen 5: Icono Xpress-Mosel

La elección de este programa para la elaboración del estudio computacional se justifica por dos razones principales:

1. Es una herramienta potente por su capacidad de indexación de las variables y ecuaciones, que permite cambiar sin dificultad las dimensiones del modelo

2. El programa permite una formulación sencilla, además de poder detectar errores de consistencia en la definición y verificación del modelo, incluso en modelos grandes y complejos

3.2. Introducción de datos al programa

Los datos que se utilizarán para el estudio computacional proceden de la página web https://www.gap-system.org. Esta página contiene una gran biblioteca de funciones y datos para el GAP. El acceso a los datos de la página es abierto.

Se introducirán los datos al programa Xpress mediante los ficheros en formato .txt descargados directamente de la página.

El formato de los datos que contienen los ficheros .txt es el siguiente:

n = Número de tareas (índice j)

m = Número de máquinas (índice i)

w_{ij} = Carga que aporta la tarea j en la máquina i

p_{ij} = Peso que aporta la tarea j en la máquina i

cap_i = Capacidad de la máquina i

Se trabajará con los ficheros que almacenan los datos en formato de texto y corresponden a 5 grupos de datos. Cada grupo de distinto tamaño $n \times m$.

Estos datos se leerán en orden desde el programa introducido en Xpress de la siguiente forma:

fopen(archivo_datos,F_INPUT)

readln(m,n)

declarations

```
    tareas=1..n
    maquinas=1..m
    p, w:array(maquinas,tareas)of integer
    f:array(maquinas,tareas)of real
    criterio,y:array(tareas)of integer
    xp:array(maquinas,tareas)of integer
    cap, capres:array(maquinas)of integer
    feas, nasign:integer

    x:array(maquinas,tareas)of mpvar

end-declarations

forall(i in maquinas,j in tareas)read(p(i,j))
forall(i in maquinas,j in tareas)read(w(i,j))
forall(i in maquinas)read(cap(i))
fclose(F_INPUT)
```

3.3. Resultados

Los resultados presentados a continuación en forma de tabla corresponden a problemas de maximización del GAP ya que los datos procedentes directamente de la página mencionada en el apartado anterior vienen dados en ese formato. Se ha implementado la función objetivo junto con las restricciones para este tipo de problemas en el software de optimización Xpress-Mosel.

Este estudio computacional considera casos variados (5 grupos de datos). Cada grupo de distinto tamaño $n \times m$.

La primera columna de las tablas recoge las situaciones consideradas, la segunda columna detalla la solución óptima entera y el tiempo (en segundos) en alcanzarla. En la tercera columna aparece la relajación lineal del problema.

En la columna cuarta y quinta las heurísticas consideradas de redondeo y la adaptación de MT basada en penalizaciones. La última columna recoge la heurística de mejora con estrategias de búsqueda local.

Por otra parte, en cada grupo de datos se muestra el promedio de las soluciones ya que es la zona o punto central en torno al cual se aglutinan los valores de las soluciones.

- **GAP1** (5 x 15)

	Método	Solución óptima entera	Relajación lineal	Heurística de redondeo	Heurística MT de penalizaciones	Mejora Búsqueda local
GAP1_1	Solución Tiempo	336 0,031	343,587 0,016	287 0	303 0	327 0
GAP1_2	Solución Tiempo	327 0,022	339,377 0,023	327 0	314 0	324 0
GAP1_3	Solución Tiempo	339 0,031	349,683 0,015	155 0	312 0	326 0
GAP1_4	Solución Tiempo	341 0,015	350,4 0,016	341 0	322 0	322 0
GAP1_5	Solución Tiempo	326 0,092	335,764 0,025	70 0	296 0	319 0
Media	Solución Tiempo	333,8 0,0382	343,7622 0,019	236 0	309,4 0	323,6 0

**Tabla 2: Resultados del GAP1*

- **GAP2** (5 x 20)

	Método	Solución óptima entera	Relajación lineal	Heurística de redondeo	Heurística MT de penalizaciones	Mejora Búsqueda local
GAP2_1	Solución Tiempo	434 0,015	444,03 0,022	414 0	393 0	413 0
GAP2_2	Solución Tiempo	436 0,038	446,916 0,015	374 0	399 0	424 0
GAP2_3	Solución Tiempo	420 0,031	425,133 0,016	384 0	373 0	422 0
GAP2_4	Solución Tiempo	419 0,019	428,329 0,035	399 0	376 0	414 0
GAP2_5	Solución Tiempo	428 0,019	431,828 0,018	428 0	402 0,001	418 0
Media	Solución Tiempo	427 0,0244	435,2472 0,0212	399,8 0	388,6 0,0002	418,2 0

**Tabla 3: Resultados del GAP2*

- **GAP10** (10 x 40)

	Método	Solución óptima entera	Relajación lineal	Heurística de redondeo	Heurística MT de penalizaciones	Mejora Búsqueda local
GAP10_1	Solución Tiempo	958 0,053	962,743 0,015	958 0	879 0,003	949 0,004
GAP10_2	Solución Tiempo	963 0,292	973,219 0,015	938 0	836 0,003	954 0,008
GAP10_3	Solución Tiempo	960 0,063	967,403 0	936 0	884 0,015	945 0
GAP10_4	Solución Tiempo	947 0,063	950,689 0,016	901 0	833 0	937 0,015
GAP10_5	Solución Tiempo	947 0,217	955,951 0,022	600 0,015	847 0	941 0
Media	Solución Tiempo	955 0,1376	962,001 0,0136	866,6 0,003	855,8 0,0042	945,2 0,0054

Tabla 4: Resultados del GAP10

- **GAP11** (10 x 50)

	Método	Solución óptima entera	Relajación lineal	Heurística de redondeo	Heurística MT de penalizaciones	Mejora Búsqueda local
GAP11_1	Solución Tiempo	1139 0,047	1145,03 0,015	1069 0	1004 0,016	1120 0
GAP11_2	Solución Tiempo	1178 0,047	1183,87 0,015	1132 0	1028 0,022	1164 0,011
GAP11_3	Solución Tiempo	1195 0,037	1197,45 0,016	1171 0	1085 0,004	1186 0,01
GAP11_4	Solución Tiempo	1171 0,173	1179,68 0,019	1037 0	1011 0,006	1140 0,009
GAP11_5	Solución Tiempo	1171 0,205	1176,46 0,018	1147 0,001	1014 0,004	1151 0,012
Media	Solución Tiempo	1170,8 0,1018	1176,498 0,0166	1111,2 0,0002	1028,4 0,0104	1152,2 0,0084

**Tabla 5: Resultados del GAP11*

- **GAP12** (10 x 50)

	Método	Solución óptima entera	Relajación lineal	Heurística de redondeo	Heurística MT de penalizaciones	Mejora Búsqueda local
GAP12_1	Solución Tiempo	1451 0,044	1454,07 0,019	1451 0	1320 0,006	1431 0,021
GAP12_2	Solución Tiempo	1449 0,111	1453,84 0,023	1449 0	1308 0,006	1435 0,022
GAP12_3	Solución Tiempo	1433 0,079	1436,83 0,021	1316 0,001	1266 0,006	1412 0,017
GAP12_4	Solución Tiempo	1447 0,055	1450,06 0,021	1374 0,001	1301 0,007	1419 0,017
GAP12_5	Solución Tiempo	1446 0,126	1451,91 0,02	1377 0	1293 0,007	1424 0,015
Media	Solución Tiempo	1445,2 0,083	1449,342 0,0208	1393,4 0,0004	1297,6 0,0064	1424,2 0,0184

**Tabla 6: Resultados del GAP12*

3.4. Calidad de los resultados

Para poder comparar la proximidad de los resultados es necesario contrastarlos con las soluciones óptimas reales. Las soluciones óptimas a los problemas considerados también se encuentran en la página web y se comprueba que coinciden con los resultados alcanzados con Xpress Mosel. Estas soluciones se resumen a continuación para cada grupo de problemas del estudio computacional:

GAP1

Problema	Solución óptima
GAP1_1	336
GAP1_2	327
GAP1_3	339
GAP1_4	341
GAP1_5	326

**Tabla 7: Solución óptima a GAP1*

GAP2

Problema	Solución óptima
GAP2_1	434
GAP2_2	436
GAP2_3	420
GAP2_4	419
GAP2_5	428

**Tabla 8: Solución óptima a GAP2*

GAP10

Problema	Solución óptima
GAP10_1	958
GAP10_2	963
GAP10_3	960
GAP10_4	947
GAP10_5	947

**Tabla 9: Solución óptima a GAP10*

GAP11

Problema	Solución óptima
GAP11_1	1139
GAP11_2	1178
GAP11_3	1195
GAP11_4	1171
GAP11_5	1171

**Tabla 10: Solución óptima a GAP11*

GAP12

Problema	Solución óptima
GAP12_1	1451
GAP12_2	1449
GAP12_3	1433
GAP12_4	1447
GAP12_5	1446

Tabla 11: Solución óptima a GAP12

Los errores calculados para cada GAP se muestran en las tablas que aparecen a continuación, en las que se considera un porcentaje indicador del hueco de la solución, definido para las heurísticas como:

$$\text{Hueco}= \frac{Solución\ óptima - Solución\ heurística}{Solución\ óptima} x100$$

Y para la relajación como:

$$\text{Hueco}= \frac{Solución\ relajación - Solución\ óptima}{Solución\ óptima} x100$$

Por lo tanto, el resultado obtenido será el porcentaje en el que varía la solución obtenida mediante la heurística con la solución óptima siendo un 0% una coincidencia exacta.

- **GAP1**

	GAP1_1	GAP1_2	GAP1_3	GAP1_4	GAP1_5	Media
Relajación lineal	2,26	3,79	3,15	2,76	3,00	**2,99**
Heurística de redondeo	14,58	0,00	54,28	0,00	78,53	**29,48**
Heurística MT de penalizaciones	9,82	3,98	7,96	5,57	9,20	**7,31**
Mejora. Búsqueda local	2,68	0,92	3,83	5,57	2,15	**3,03**

**Tabla 12: Media del error calculado en GAP1*

- **GAP2**

	GAP2_1	GAP2_2	GAP2_3	GAP2_4	GAP2_5	Media
Relajación lineal	2,31	2,50	1,22	2,23	0,89	**1,83**
Heurística de redondeo	4,61	14,22	8,57	4,77	0,00	**6,43**
Heurística MT de penalizaciones	9,45	8,49	11,19	10,26	6,07	**9,09**
Mejora. Búsqueda local	4,84	2,75	-0,48	1,19	2,34	**2,13**

**Tabla 13: Media del error calculado en GAP2*

- **GAP10**

	GAP10_1	GAP10_2	GAP10_3	GAP10_4	GAP10_5	Media
Relajación lineal	0,50	1,06	0,77	0,39	0,95	**0,73**
Heurística de redondeo	0,00	2,60	2,50	4,86	36,64	**9,32**
Heurística MT de penalizaciones	8,25	13,19	7,92	12,04	10,56	**10,39**
Mejora. Búsqueda local	0,94	0,93	1,56	1,06	0,63	**1,03**

**Tabla 14: Media del error calculado en GAP10*

- **GAP11**

	GAP11_1	GAP11_2	GAP11_3	GAP11_4	GAP11_5	Media
Relajación lineal	0,53	0,50	0,21	0,74	0,47	**0,49**
Heurística de redondeo	6,15	3,90	2,01	11,44	2,05	**5,11**
Heurística MT de penalizaciones	11,85	12,73	9,21	13,66	13,41	**12,17**
Mejora. Búsqueda local	1,67	1,19	0,75	2,65	1,71	**1,59**

**Tabla 15: Media del error calculado en GAP11*

- **GAP12**

	GAP12_1	GAP12_2	GAP12_3	GAP12_4	GAP12_5	Media
Relajación lineal	0,21	0,33	0,27	0,21	0,41	**0,29**
Heurística de redondeo	0,00	0,00	8,16	5,04	4,77	**3,60**
Heurística MT de penalizaciones	9,03	9,73	11,65	10,09	10,58	**10,22**
Mejora. Búsqueda local	1,38	0,97	1,47	1,94	1,52	**1,45**

**Tabla 16: Media del error calculado en GAP12*

4. CONCLUSIONES

4.1. Resumen de promedios

Se calculan los huecos promedio para cada uno de los métodos teniendo en cuenta los resultados obtenidos en las situaciones descritas en el estudio computacional:

Método	Hueco medio
Relajación lineal	1,27%
Heurística de redondeo	10,79%
Heurística MT de penalizaciones	9,84%
Mejora. Búsqueda local	1,85%

**Tabla 17: Hueco medio obtenido para cada método*

También se calculó el promedio para cada uno de los tiempos computacionales del estudio (en segundos) y se recoge en la siguiente tabla:

Método	Tiempo medio
Relajación lineal	0,02
Heurística de redondeo	0,0007
Heurística MT de penalizaciones	0,004
Mejora. Búsqueda local	0,006

**Tabla 18: Tiempo medio obtenido para cada método*

4.2. Análisis de los resultados

Una vez obtenidos los resultados y calculados los promedios en los huecos de las soluciones y tiempos computacionales encontramos una gran diferencia entre los diferentes métodos.

Para llevar a cabo un análisis de los resultados más profundo se ha manejado Statgraphics, introduciendo los datos de los estadísticos obtenidos en dos columnas (Tiempo y Hueco) con variables de tipo numérico y una tercera columna (Método) de tipo carácter que distingue las tres heurísticas estudiadas en este trabajo, junto con la relajación lineal. De esta forma podremos comparar a la vez gráficamente la calidad de los métodos.

Se representa gráficamente la variable cuantitativa bidimensional (Tiempo,Hueco) con un diagrama de dispersión, representando con un triángulo azul la heurística de mejora con búsqueda local, una cruz naranja la heurística de MT con penalizaciones, un círculo rosa la heurística de redondeo y un positivo verde la relajación lineal.

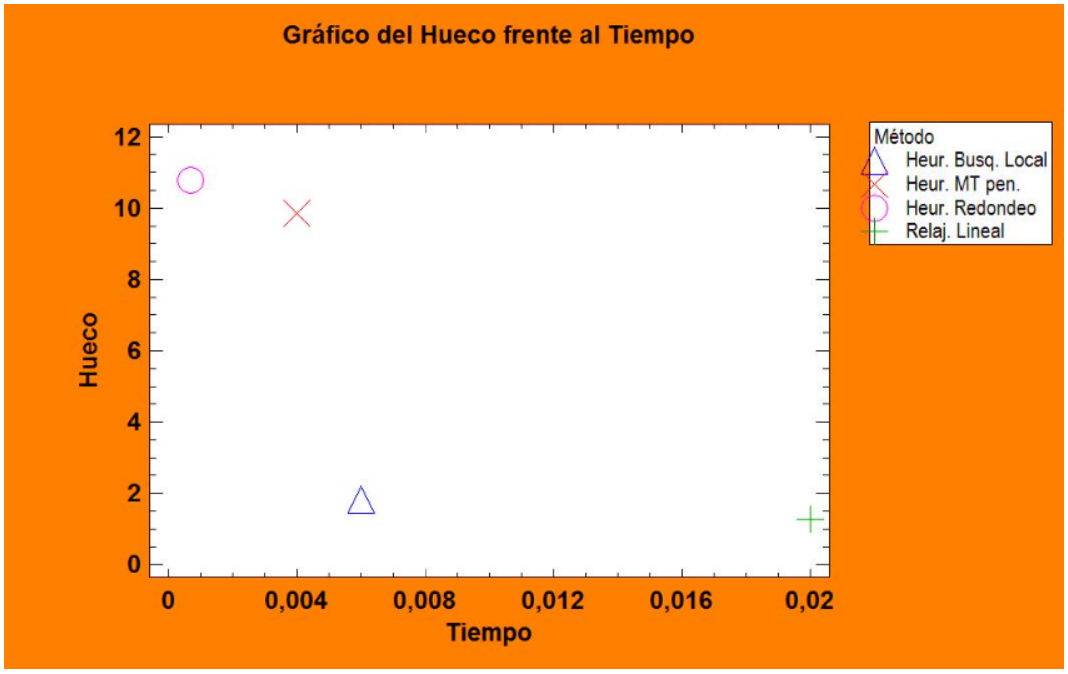

Imagen 6: Diagrama de dispersión bidimensional (Hueco-Tiempo)

- En la relajación lineal, los resultados se aproximan muy bien al óptimo, a pesar de ser un método de resolución muy sencillo. Es una cota superior de la solución del problema, puesto que estamos en un problema de maximización, pero al igual que la solución exacta, necesita más tiempo computacional para ejecutarse por el método de optimización.

- Se observa que la heurística de redondeo obtiene el peor resultado del estudio computacional, aunque también se trata del método más rápido de resolución.

- La heurística MT de penalizaciones resulta similar a la heurística de redondeo en cuanto a la calidad de las soluciones. Se trata de un método rápido para encontrar la solución, pero también encuentra una solución bastante alejada del óptimo.

- Se demuestra que la heurística de mejora con estrategias de búsqueda local proporciona una excelente calidad de la solución disminuyendo considerablemente el tiempo computacional. Teniendo en cuenta la rapidez en la obtención de las soluciones, se comprueba la validez de este método en la resolución de problemas de asignación generalizada considerados en este trabajo para hallar una solución muy próxima a la óptima.

4.3. Conclusiones

La heurística de mejora con estrategias de búsqueda local propuesta en este trabajo ha demostrado ser un método muy eficaz y rápido para la obtención de una solución muy próxima a la óptima en el problema del GAP. Si bien a veces no se alcanza la solución óptima al problema, puede ser una opción ventajosa cuando:

- No exista un método exacto de resolución o éste requiere mucho tiempo de cálculo o memoria.
- No se disponga de un solver de optimización para hallar directamente la solución óptima.
- Los datos sean poco fiables.
- Haya limitaciones de recursos (tiempo y/o espacio).
- Los algoritmos sean complejos y requieran etapas intermedias o soluciones iniciales de partida para ser mejoradas posteriormente.

La implementación en Xpress-Mosel permite la modificación del programa con facilidad, de hecho, resulta sencillo adaptar esta programación a las diferentes variantes del GAP, por ejemplo, en el estudio computacional se ha tratado el caso de maximización, pero si fuera necesario minimizar bastaría con cambiar la acción y los datos en el input.

5. FUTURAS EXTENSIONES

5.1. Soluciones aproximadas al GAP

5.1.1. Algoritmos de aproximación de tiempo polinomial

El primer algoritmo de aproximación conocido para el GAP fue propuesto en 1993. El GAP se definió como un problema de programación de una máquina en paralelo donde el objetivo es encontrar el mínimo coste al asignar cada trabajo j a una única máquina i, en un tiempo de procesamiento r_{ij} y un coste de procesamiento c_{ij}, de tal modo que la restricción de capacidad (tiempo disponible) de la máquina i, b_i, no sea excedido. Dado un valor C, se ideó un plan de aproximación de tiempo polinomial el cual o prueba la no existencia de una asignación factible (programa) con coste C o la generación de una asignación (programa) de coste máximo C donde la capacidad de uso de cada máquina i sea como máximo de $2b_i$ unidades.

Otra variante del GAP trata de, dado un conjunto de mochilas con diferentes capacidades y un conjunto de ítems de pesos diferentes y valores diferentes para cada mochila, encontrar un subconjunto de ítems que pueda asignarse a la mochila y que maximice el valor. Para su solución se propone un algoritmo basado en programación lineal con una aproximación $e/e - 1 + \varepsilon$ donde ε es una constante positiva y es la base de la función logarítmica natural.

Otro caso especial del GAP similar al anterior sería el máximo beneficio del GAP con beneficio fijo. En este caso, cada ítem j tiene un beneficio fijo p_j cada mochila i. Por lo tanto, la función objetivo es la siguiente:

$$max \sum_{i=1}^{m} \sum_{j=1}^{n} p_j x_{ij} \qquad (20)$$

El algoritmo de aproximación propuesto para este problema es $(1 - 1/e)$.

5.1.2. Algoritmo heurístico tipo Greedy

Este método surge de considerar una medida de conveniencia de maximizar el problema (f_{ij}) de asignar una tarea j a un agente i. El algoritmo heurístico considera de forma iterativa todas las tareas no asignadas y escoge una tarea j que tenga la máxima diferencia entre los dos f_{ij} menores. Esta tarea será asignada a un agente buscando el menor f_{ij}. Tras la aplicación de este algoritmo heurístico, se emplea un plan de mejora mediante cambios locales de la solución actual. Se propone emplear 4 tipos de medidas distintas para obtener la mejor solución posible.

Otra propuesta para resolver el GAP sería mediante la aplicación de un algoritmo dual. Primeramente, resolver una versión más sencilla del problema ignorando las restricciones de capacidad (2). Una vez asignados agentes y tareas, localizar las tareas que exceden las restricciones de capacidad para asignarlas a otros agentes cumpliendo así las restricciones y minimizando tanto como sea posible el coste total de asignación

5.1.3. Algoritmos heurísticos de partición de conjuntos

Un problema de partición de conjuntos está basado en las técnicas de generación de columnas con límites superiores e inferiores propuestos. El GAP formulado como un problema de partición de conjuntos (SPP) donde cada columna corresponde con una asignación factible de un subconjunto de tareas a un agente. Mediante la generación de columnas, se halla una solución del problema de la mochila para cada agente.

5.1.4. Algoritmos heurísticos de relajación lagrangiana

Existen seis propuestas de relajaciones utilizando métodos heurísticos mediante la lagrangiana y mediante un sustituto de relajación de la formulación del GAP. Ambos surgen de la relajación de las restricciones de capacidad (2). Para resolver ambos, se ha empleado el método del subgradiente.

Un nuevo método de relajación heurística surge con la utilización de múltiples relajaciones y métodos heurísticos constrictivos. Este método se conoce como relajación lagrangiana/sustituto. Se basa en relajar las restricciones de capacidad (2), las restricciones de semi-asignación (3) y las restricciones de descomposición lagrangiana. La relajación lagrangiana/sustituto puede dar mejores resultados que los métodos heurísticos de relajación lagrangiana usando el método de optimización subgradiente con relajaciones lagrangianas tradicionales.

La descomposición de la relajación lagrangiana basada en métodos heurísticos se realiza sustituyendo $y_{ij} = r_{ij}x_{ij}$ y reduciendo las restricciones del GAP original.

Otro método heurístico lagrangiano que encuentra y mejora las soluciones factibles al GAP se basa en utilizar la restricción de capacidad (2) como una dualidad y aplicar el procedimiento de optimización del método subgradiente. En cada iteración del método heurístico lagrangiano, previamente se emplea un método heurístico de aproximación para encontrar una asignación inicial.

Los problemas de relajación lagrangiana basados en métodos heurísticos de búsqueda espacial combinan la habilidad de realizar búsquedas iterativas del método de optimización subgradiente y el plan de búsqueda del espacio del problema de perturbación. Se lleva a cabo tomando las restricciones de capacidad (2) como una dualidad y aplicando el procedimiento de optimización subgradiente. Para hallar una solución factible se proponen tres métodos heurísticos de restauración viables. Para incrementar las

posibilidades de obtener una buena solución de los métodos heurísticos se adopta la técnica metaheurística de búsqueda en espacios (PSS). El PSS genera problemas artificiales próximos que perturban temporalmente los datos del problema.

5.1.5. Relajación mediante programación lineal basada en métodos heurísticos

Mediante la exploración de la estructura clásica, este método heurístico iterativamente elimina las variables redundantes (las variables x_{ij} que corresponden a r_{ij} con $r_{ij} > b_i$), resuelve la relajación con las variables restantes, fija las variables con $x_{ij} = 1$ y actualiza las capacidades y elimina los trabajos asignados. Este procedimiento se repite hasta haber eliminado todas las variables redundantes. Al final de este método se aplica una operación de intercambio basada en un plan de mejora.

5.1.6. Otros métodos de obtención de una solución aproximada

Para la resolución de un GAP de gran escala se propone el uso de una solución aproximada agregada/desagregada. Este método se realiza agregando el problema inicial y desagregando la solución óptima obtenida para obtener una solución factible del problema original.

Existe una metodología basada en técnicas de búsqueda por entornos orientada a objetos para resolver problemas de asignación (ATP) incluyendo al GAP (método descendente, búsqueda tabú, proceso de intercambio, recocido simulado). En otras palabras, se trata de software base que puede ser empleado para cualquier problema de asignación. De acuerdo con su definición, los problemas de asignación consisten en restricciones de semi-asignación (3) y en otras restricciones. La restricción de semi-asignación asegura la asignación de un ítem a una única mochila.

5.2. Algunas metaheurísticas

5.2.1. Búsqueda tabú

La búsqueda tabú (TS) es un plan de búsqueda local o por vecindades el cual comienza con una solución inicial que se traslada hasta una nueva solución seleccionada entre un conjunto de soluciones vecinas que no necesariamente mejoran la función objetivo. Para prevenir un movimiento indeseado de la solución se emplean estructuras de memoria. Los algoritmos TS conservan durante un corto periodo de memoria de los atributos de algunos movimientos en la lista tabú. Los atributos que permanecen en la lista tabú son utilizados para prohibir algunas soluciones que ya fueron aceptadas en un cierto número de iteraciones. Estos atributos se denominan “tabú-activos” mientras que las soluciones que los contengan se denominan “tabú”. Dicho de otra forma, las restricciones TS se definen en los atributos que se encuentran en la lista tabú.

La estrategia denominada “cadenas de eyección” ha dado buenos resultados para la realización de la búsqueda tabú. Adoptando la aproximación “path relinking” en el algoritmo de “cadenas de eyección” mejoran los resultados obtenidos. Se proponen dos fases para realizar el algoritmo “path relinking”. En la primera fase el conjunto de soluciones se obtiene mediante la programación lineal de la formulación del GAP, reduciendo el problema y aplicando la búsqueda local. La segunda fase aplica el algoritmo “path relinking” el cual es una generalización de “scatter search” o búsqueda dispersa que opera con una combinación muy pequeña de soluciones.

5.2.2. Recocido simulado

El recocido simulado (SA) está basado en la analogía entre el recocido de sólidos y la resolución de problemas de optimización combinatorios. SA es una estrategia de búsqueda local el cual evita evita la salida de un óptimo local usando estrategias de selección y aceptación aleatorias. Mediante la aceptación aleatoria las soluciones más desfavorables son aceptadas con cierta probabilidad, la cual es controlada por un parámetro relacionado con la temperatura. La actualización de este parámetro proviene de un proceso de enfriamiento.

5.2.3. Algoritmo genético

El algoritmo genético (GA) es un algoritmo de búsqueda probabilístico que simula el proceso de la evolución. Un GA toma un conjunto de soluciones y reproduce nuevas soluciones utilizando operadores genéticos. Las soluciones reproducidas son evaluadas y las mejores soluciones son seleccionadas para su reproducción. El algoritmo se ejecuta hasta que no sea posible encontrar una solución mejor o hasta que se haya excedido un límite de tiempo. GA trata con un conjunto de soluciones descritos por unos parámetros conocidos como genes. El conjunto de estos parámetros forma cadenas de valores conocidos como cromosomas. La representación de los cromosomas es imprescindible para el desarrollo del GA.

5.2.4. Redes neuronales

Las redes neuronales (NN) son principalmente procesos de aprendizaje adaptativo que actualizan constantemente algunos pesos hasta llegar a un punto aceptable, es decir, cuando se alcanza una solución casi factible o factible. Durante las últimas décadas un gran número de investigaciones han estudiado el uso de las NN para resolver problemas de optimización combinatoria. El primer intento de resolver el GAP usando una competición basado en redes neuronales consistía en una matriz m×n donde cada posición ocupada por una neurona que compite por volverse activa. Otra aplicación de una red neuronal intentó cuatro métodos diferentes para estructurar la función de energía de la red neuronal: método de función de penalización exterior, método de lagrangiano aumentado, método dual lagrangiano y método de función de penalización interior. El método de lagrangiano aumentado proporciona mejores soluciones que otros métodos con respecto a la integridad de la medida manteniendo la viabilidad y medida de estabilidad.

5.2.5. Colonia de hormigas y GRASP

Esta metaheurística está basada en una aproximación híbrida. Una de las metaheurísticas propuestas podría ser la colonia de hormigas MAX-MIN la cual es una mejora de la colonia de hormigas. La colonia de hormigas muestra un procedimiento adaptativo, que tiene en cuenta la experiencia adquirida de iteraciones anteriores. También se podría proponer un método GRASP (Greedy Ramdomized Adaptative Search Procedure) cuyo funcionamiento es similar al algoritmo de búsqueda por vecindades que emplea un plan de mejora local en varios tiempos, cada uno con diferentes soluciones factibles al comienzo. Las soluciones iniciales que emplea el algoritmo son generadas mediante un procedimiento aleatorio.

6. BIBLIOGRAFÍA

En este apartado se enumeran las fuentes (documentos, libros, artículos, webs, etc.) consultadas para la realización de este Trabajo Fin de Grado.

- ✓ Di Vita, A. Programación Lineal – Métodos Cuantitativos. [online] Escuela de organización industrial. Available at: https://www.eoi.es [Accesed 11 Mar. 2019]

- ✓ Guéret, C., Prins, C., & Sevaux, M. (1999). Applications of optimization with Xpress-MP. contract, 00034.

- ✓ Öncan, T. A Survey of the Generalized Assingnment Problem and Its Applications (Estudio del Problema de Asignación Generalizado y sus Aplicaciones). INFOR Journal, Vol 45, No.3, Agosto 2007, pp. 123-141. ISSN 0315-5986. EISSN 1916-0615.

- ✓ Lourenço, H.R. (2002), Heurísticas adaptativas para el problema de asignación generalizada (Adaptive heuristics for the generalized assignment problem). In Proceedings of The First Spanish Congress in Evolutive and Bioinspired Algorithms, Merida, Spain, February 6-7, pp. 267-275. ISBN 84-607-3913-9.

- ✓ Martello S., Toth P.(1992). Generalized assignment problems. In: Ibaraki T., Inagaki Y., Iwama K., Nishizeki T., Yamashita M. (eds) Algorithms and Computation. ISAAC 1992. Lecture Notes in Computer Science, vol 650. Springer, Berlin, Heidelberg.

- ✓ Martı, R. (2003). Procedimientos metaheurısticos en optimización combinatoria. Matemátiques, Universidad de Valencia, 1(1), 3-62.

- ✓ Salazar López, B. (2016). Problemas de Asignación. [online] Ingeniería industrial online. Available at: https://www.ingenieriaindustrialonline.com [Accessed 15 Mar. 2019].

- ✓ Taha, Hamdy A. Operation Research (Investigación de Operaciones). Rodolfo Navarro, Rodolfo (trad.). 9ª ed. México: Pearson educación, 2012. ISBN: 978-607-32-0796-6.

- ✓ Trick, M.A. A linear relaxation heuristic for the generalized assignment problem (Heurística de relajación lineal para el problema de asignación generalizado). Naval Researh Logistic (NRL), Vil 30, Ed 2, Marzo 1992, pp. 137-151.

- ✓ https://www.fico.com/es/products/fico-xpress-optimization

- ✓ https://www.gap-system.org

- ✓ https://statgraphics.net