Resolución de un problema real de planificación de horarios escolares mediante métodos matheurísticos

Isabel Méndez[1][0000−0001−5717−4157], Rubén Gordaliza[1], Sergio Pérez-Peló[2][0000−0002−1915−4160], Alfredo G. Hernández-Díaz[3][0000−0002−2392−3786], and Jesús Sánchez-Oro[2][0000−0003−1702−4941]

[1] OGA S.L., Judería 2, Planta 1, Módulo 3A, Parque Empresarial Vega del Rey, 41900, Camas – Sevilla
{isabel.mendez, ruben.gordaliza}@oga.ai
https://www.oga.ai

[2] GRAFO, Dept. Computer Science and Statistics, Universidad Rey Juan Carlos, C/Tulipán S/N
{sergio.perez.pelo, jesus.sanchezoro}@urjc.es
https://grafo.etsii.urjc.es/en/

[3] Department of Economic, Quantitative Method and Economic History, Universidad Pablo de Olavide, Ctra. de Utrera, 1, 41013 Sevilla
agarher@upo.es
www.upo.es

Resumen En este trabajo se propone un algoritmo basado en Iterated Greedy para resolver un problema de planificación de horarios real derivado de centros docentes públicos de la Junta de Andalucía. La propuesta se compara con la herramienta de caja negra HorW, que lleva utilizándose numerosos años en el entorno docente de la región como estándar en la planificación horaria de los docentes del centro. Los resultados muestran como la propuesta ofrece resultados competitivos frente a la herramienta original. Además, se prueba como los modelos exactos existentes para este problema no presentan la escalabilidad necesaria para resolver instancias cada vez más exigentes, requiriendo de algoritmos aproximados como el propuesto en este trabajo.

Keywords: Iterated Greedy · horarios escolares · planificación · matheurísticos.

1. Introducción

En todas las instituciones educativas es necesario disponer de una asignación de recursos eficiente, entendiendo por recursos los materiales como las aulas, los personales como los profesores, y los temporales como los períodos de tiempo en los que se pueden impartir las clases. Esta asignación de recursos es una solución al problema de planificación de horarios, en el que proporcionar una

solución de alta calidad en un tiempo de cómputo razonable es muy importante tanto pedagógica como económicamente. La factibilidad de una solución derivada de un caso real, como el tratado en esta investigación, depende de numerosas restricciones de diferente naturaleza, que hacen que en muchas ocasiones sea difícil incluso generar una solución factible.

Generar soluciones de alta calidad en la planificación de horarios escolares transforma la gestión educativa al optimizar el uso de los recursos físicos y humanos, garantizando que tanto docentes como alumnos cuenten con jornadas equilibradas y pedagógicamente coherentes. Una solución de calidad minimiza las horas libres innecesarias, respeta las restricciones de disponibilidad del profesorado (lo que mejora el clima laboral y reduce el agotamiento) y asegura que los espacios especializados, como laboratorios o gimnasios, se utilicen de manera eficiente sin solapamientos. Al reducir el error humano y los conflictos logísticos mediante algoritmos avanzados, la institución no solo ahorra tiempo administrativo crítico, sino que también eleva la satisfacción de la comunidad escolar y el rendimiento académico al permitir que el enfoque principal regrese a la enseñanza y el aprendizaje. A pesar de la relevancia de este problema, a menudo se resuelve de forma manual, sobre todo en las instituciones más pequeñas.

El problema de la planificación de horarios surge formalmente en la década de los 60 [8], y desde entonces ha sido objeto de estudio por parte de la comunidad científica. La variante original requiere planificar un conjunto de eventos (generalmente clases) con un conjunto de profesores de manera que ningún profesor tenga asignada más de una clase a la vez. Esta variante se resuelve en tiempo polinómico con un algoritmo de mínimo coste de flujo en redes [17]. Sin embargo, cuando el problema refleja una situación real, aparecen nuevas restricciones, como la no disponibilidad de los profesores en ciertos momentos, o la necesidad de conciliación que limitan los horarios docentes, entre otras, que convierten el problema en un problema $\mathcal{NP}$-completo [6]. La diversidad de restricciones existentes han resultado en una familia de variantes del problema en la literatura, de manera que el conjunto de objetivos a optimizar y las restricciones dependen completamente del contexto de la aplicación, el centro escolar y el lugar donde se encuentra [5,11,15].

El problema original y sus variantes se han estudiado desde diferentes puntos de vista, como se refleja en el estudio [15], donde se presentan las principales variantes, así como su formulación y las propuestas de resolución principales. En su origen, no se consideraron métodos exactos para la resolución de este tipo de problemas, debido a su complejidad, ya que solo era posible resolver o bien instancias muy pequeñas, o bien versiones muy simplificadas que se alejaban del problema real. Sin embargo, las mejoras recientes en *solvers* de programación lineal entera mixta [10] han motivado el uso de técnicas exactas para abordar estos problemas [2]. Es por ello que la mayoría de investigaciones recientes se enfrentan al problema mediante el uso de técnicas híbridas que combinan el rigor de los métodos exactos con la eficiencia de los métodos heurísticos.

En este trabajo se propone un modelo de programación lineal entera mixta para una variante del problema de planificación de horarios derivado de un

entorno real. En concreto, el objetivo es proponer una solución eficiente para realizar la planificación escolar de centros de la Junta de Andalucía, para los que se disponen de datos reales. La Sección 2 presenta la definición del problema, incluyendo cuáles son las restricciones contempladas en esta variante, la Sección 3 presenta la propuesta algorítmica para resolver este problema, la Sección 4 analiza la eficiencia del modelo comparándolo con la solución actual implantada, y la sección 5 plantea las conclusiones derivadas de la investigación y las posibles futuras líneas de trabajo.

2. Definición del problema

En primer lugar, es necesario definir quiénes son los componentes principales que entran a formar parte de este problema. Al tratarse de un problema de planificación de horarios, es necesario definir la estructura temporal que necesitamos planificar. El *slot* mínimo al que se puede asignar una clase. Sin pérdida de generalidad, y para el problema concreto tratado, se considera que todas las clases tienen una duración de una hora y que existe un período de 30 minutos de descanso correspondiente al recreo. Cabe destacar que los centros pueden tener jornadas de mañana, de tarde, o partidas y a cada *slot* se le puede asignar cualquiera de las tareas disponibles: clases, guardias, etc. La mayoría de centros educativos abarcan además diferentes etapas educativas: ESO, Bachillerato (con diferentes itinerarios o modalidades), Ciclos de Formación Profesional, etc.

Los estudiantes se clasifican en grupos y subgrupos. Un grupo se define como una clase o sección que comparte la mayoría de sus asignaturas, y en ocasiones se forman subgrupos para realizar ciertas actividades, como pueden ser asignaturas optativas, diversificación curricular, etc. Cada materia impartida tiene un número fijo de horas semanales que se deben planificar en el horario de cada grupo que la cursa, y no es posible tener dos horas consecutivas de la misma materia. Además, algunas materias pueden requerir del uso de espacios particulares (taller, gimnasio, laboratorios, etc.) o tener preferencia de ciertas franjas horarias. Algunas asignaturas se agrupan en bloques, ya que requieren que se impartan a la vez que las otras del mismo bloque, como pueden ser optativas o desdobles, entre otros, añadiendo una restricción de sincronía en su planificación, y obligando a que tengan profesores diferentes ya que se imparten a la vez.

El centro educativo cuenta con una plantilla de docentes que están asignados a departamentos, y que tienen habilitación para impartir un subconjunto de materias. Cada uno de los profesores tiene un horario definido por sus cargas horarias y sus funciones, con una permanencia de 25 horas semanales en el centro, de las cuáles 18 son lectivas y 7 no lectivas. Algunas situaciones pueden reducir la carga lectiva: un tutor de grupo reduce en 2 horas su carga, el jefe de departamento en 3 horas, los mayores de 55 años reducen 2 horas, entre otras. Antes de realizar la planificación de horarios, la dirección asigna a los departamentos los grupos y horas que debe cubrir, y el departamento realiza la distribución de asignaturas entre sus profesores. Esta asignación previa se considera fija para la

variante tratada en este trabajo, y los profesores además participan en reuniones y realizan guardias que ocuparán huecos en su planificación.

Las guardias son periodos en los que ciertos docentes tienen asignada la vigilancia y atención de incidencias del centro. En general, se consideran guardias de pasillo, de recreo y de convivencia, aunque no tiene por qué limitarse a estos tipos concretos. La normativa del centro determinará el número de profesores de guardia necesarios de cada tipo y el número de guardias que debe tener asignadas un profesor de manera semanal.

El centro dispone de un conjunto de aulas y espacios educativos. Cada grupo tiene asignada una clase de referencia, pero los docentes pueden cambiar de clase según las necesidades de la materia impartida. Aunque por norma general un aula solo puede usarse para una única actividad a la vez, existen excepciones como el uso del gimnasio por varios grupos de forma simultánea.

En primer lugar, es necesario definir cuáles son los conjuntos que representan los elementos anteriormente definidos. En concreto:

- E: Conjunto de eventos, donde cada evento $e \in E$ es una combinación fija de grupo, docente, asignatura y aula.
- S: Conjunto de *slots* temporales que representan los períodos lectivos. Se divide en el conjunto D (días de la semana) y T (franjas horarias por día).
- P: Conjunto de profesores.
- G: Conjunto de grupos de alumnos.
- A: Conjunto de aulas/espacios.
- B: Conjunto de bloques (eventos que deben ser paralelos).
- M: Conjunto de reuniones colectivas.
- D: Conjunto de días de la semana.

Existen además una serie de parámetros que serán dependientes del centro:

- H_e: número de sesiones semanales requeridas para el evento e según el currículo.
- $MaxD_p$: máximo de horas lectivas diarias permitidas para el profesor p (prefijado en 5).
- $MinD_p$: mínimo de horas lectivas diarias permitidas para el profesor p (prefijado en 2).
- DG_s: demanda mínima de profesores de guardia necesaria en el slot s.
- Cap_a: capacidad máxima de grupos simultáneos en el aula a.

A partir de estos conjuntos, se definen las siguientes variables de decisión:

- $x_{e,s} \in \{0, 1\}$: variable binaria que toma el valor 1 si el evento e se asigna al slot s, y 0 en caso contrario.
- $g_{p,s} \in \{0, 1\}$: variable binaria que toma el valor 1 si el profesor p realiza una guardia en el slot s, y 0 en caso contrario.
- $y_{m,s} \in \{0, 1\}$: variable binaria que toma el valor 1 si la reunión m se programa en el slot s, y 0 en caso contrario.
- $h_{p,s} \in \{0, 1\}$: variable binaria auxiliar que toma el valor 1 si el profesor p tiene libre el slot s, y 0 en caso contrario.

El objetivo del problema es minimizar el número de huecos que tienen los profesores en sus horarios, para generar un horario compacto que minimice el tiempo que pasan los docentes en el centro de trabajo. Es decir,

$$\text{mín} \sum_{p \in P, s \in S} h_{p,s}$$

Para que una solución sea factible, debe satisfacer las siguientes restricciones:

1. **Cobertura de sesiones obligatorias:** Cada evento e (combinación de grupo-materia-docente) debe programarse exactamente el número de sesiones semanales H_e fijadas por el currículo.

$$\sum_{s \in S} x_{e,s} = H_e, \quad \forall e \in E \tag{1}$$

2. **No solapamiento de docente:** Un profesor p no puede realizar más de una actividad (clase, guardia o reunión) simultáneamente en el mismo slot s.

$$\sum_{e \in E_p} x_{e,s} + g_{p,s} + \sum_{m \in M_p} y_{m,s} \leq 1, \quad \forall p \in P, \forall s \in S \tag{2}$$

3. **Unicidad de grupo:** Un grupo de alumnos g solo puede tener asignado un bloque o actividad en cada periodo horario.

$$\sum_{e \in E_g} x_{e,s} \leq 1, \quad \forall g \in G, \forall s \in S \tag{3}$$

4. **Sincronización de bloques (paralelismos):** Los eventos que pertenecen a un bloque de paralelismo B_k (como optativas o desdobles) deben ocurrir en el mismo slot s.

$$x_{e_i,s} = x_{e_j,s}, \quad \forall e_i, e_j \in B_k, \forall s \in S \tag{4}$$

5. **Límite de carga diaria docente:** Ningún profesor puede impartir más del máximo de horas de clase en una misma jornada diaria d.

$$\sum_{t \in T} \sum_{e \in E_p} x_{e,(d,t)} \leq MaxD_p, \quad \forall p \in P, \forall d \in D \tag{5}$$

6. **Mínimo de horas diarias:** Se debe garantizar que un docente imparta al menos un número mínimo de sesiones (generalmente 2) cada día que asista al centro para evitar desplazamientos por clases aisladas.

$$\sum_{t \in T} \sum_{e \in E_p} x_{e,(d,t)} \geq MinD_p \vee \sum_{t \in T} \sum_{e \in E_p} x_{e,(d,t)} = 0, \quad \forall p \in P, \forall d \in D \tag{6}$$

7. **Capacidad de espacios:** El número de grupos en instalaciones compartidas no puede exceder su capacidad máxima Cap_a (ej. el gimnasio) .

$$\sum_{e \in E_a} x_{e,s} \leq Cap_a, \quad \forall a \in A, \forall s \in S \tag{7}$$

8. **Cobertura de vigilancia:** En cada periodo s debe haber un número mínimo de profesores disponibles para guardias de pasillo y recreo .

$$\sum_{p \in P} g_{p,s} \geq DG_s, \quad \forall s \in S \tag{8}$$

9. **Secuenciación académica (ESO/Bachillerato):** Se prohíbe asignar dos sesiones de la misma materia al mismo grupo en un solo día .

$$\sum_{e \in E_{g,asign}} x_{e,s} \leq 1, \quad \forall s \in d, \forall g \in G, \forall d \in D \tag{9}$$

Este conjunto de restricciones son aquellas necesarias para garantizar que una solución es factible, pero en el problema de planificación de horarios existe un conjunto de restricciones *soft* adicionales que pueden tener diferentes objetivos, entre los que destacan favorecer las preferencias de los docentes respecto a sus horarios y favorecer los criterios pedagógicos, de manera que, por ejemplo, no se asignen las materias más difíciles a última hora o no haya dos materias difíciles consecutivas. Estas restricciones son particulares de cada usuario y se incluyen mediante penalizaciones en la función objetivo.

3. Propuesta algorítmica

En este trabajo se propone un algoritmo basado en Iterated Greedy para resolver el problema definido en la sección anterior. Iterated Greedy (IG) es una metaheurística propuesta originalmente para resolver problemas de planificación de tareas [13], lo que lo hace especialmente adecuado para el problema tratado en esta investigación. Además, IG se encuentra en evolución continua, y ha sido adaptado para resolver diferentes problemas de optimización combinatoria, entre los que se pueden destacar: detección de comunidades en redes sociales [14], problemas de planificación de tareas para robots [9], o problemas de localización de regeneradores [12], entre otros. El éxito de IG reside en su simplicidad, ya que su idea principal se basa en aplicar de manera iterativa dos fases consecutivas: destruir la solución actual para posteriormente reconstruirla. La fase de destrucción o de reconstrucción suelen incluir cierta aleatoriedad, permitiendo así al algoritmo escapar de óptimos locales. El Algoritmo 1 detalla el pseudocódigo asociado al algoritmo propuesto.

IG requiere de tres parámetros: I, los datos de entrada para el problema a resolver; Δ, el número de iteraciones que se van a ejecutar; y β, el porcentaje de solución que será destruida en cada iteración con el procedimiento de destrucción que se detalla en la Sección 3.2.

En primer lugar, se genera una solución inicial con el procedimiento de generación de soluciones descrito en la Sección 3.1 (paso 1). A continuación, el procedimiento realiza el número de iteraciones indicado por el parámetro Δ (pasos 2-9). En cada iteración, se destruye parcialmente la solución con el procedimiento descrito en la Sección 3.2 (paso 3), para después reconstruirla con el

Algoritmo 1 $IG(I, \Delta, \beta)$

1: $S_b \leftarrow Construir(I)$
2: **for** $i \in 1 \ldots \Delta$ **do**
3: $\quad S' \leftarrow Destruir(S_b, \beta)$
4: $\quad S'' \leftarrow Reconstruir(S')$
5: $\quad S''' \leftarrow Mejorar(S'')$
6: $\quad$ **if** $f(S''') < f(S_b)$ **then**
7: $\quad\quad S_b \leftarrow S'''$
8: $\quad$ **end if**
9: **end for**
10: **return** S_b

procedimiento descrito en la Sección 3.3 (paso 4). La solución reconstruida no es necesariamente un óptimo local, por lo que se le aplica el procedimiento de mejora descrito en la Sección 3.4 (paso 5). Si la solución es mejor que la mejor solución encontrada hasta el momento, se actualiza (pasos 6-8). Finalmente, el algoritmo devuelve la mejor solución encontrada durante la búsqueda.

3.1. Solución inicial

La solución inicial de la que parte IG puede ser generada al azar o bien con un constructivo más elaborado que permita al algoritmo comenzar la búsqueda desde un punto de partida más prometedor, acelerando así su convergencia.

Los algoritmos exactos han demostrado su eficiencia resolviendo problemas de planificación de horarios en las últimas décadas [1,3,16], a menudo combinados con técnicas heurísticas para resolver problemas más complejos [4,7]. Debido a la dificultad de generar una solución factible partiendo desde cero, en este trabajo se propone el uso de un modelo exacto para generar una solución inicial factible y de alta calidad. Hay que tener en cuenta que la generación de esta solución tendrá un tiempo límite, de manera que el modelo exacto no podrá garantizar la optimalidad de la misma en la mayoría de ocasiones. La solución inicial se utilizará como un punto de partida prometedor para el algoritmo propuesto.

3.2. Destrucción

La fase de destrucción tiene como objetivo diversificar la búsqueda para escapar de óptimos locales dentro del marco de IG. Para ello, realiza una destrucción parcial de la solución, consistente en la eliminación de ciertos componentes de la misma, para posteriormente reconstruirla y encontrar nuevas regiones prometedoras del espacio de búsqueda. Esta fase requiere de un parámetro β que indicará el porcentaje de destrucción a realizar en cada iteración. Es necesario realizar un ajuste de este parámetro ya que una destrucción demasiado pequeña podría llevar a una solución prácticamente igual a la original, mientras que

una demasiado grande sería equivalente a comenzar la búsqueda desde un nuevo punto inicial.

La destrucción consiste en la eliminación de un subconjunto de componentes de la solución con el objetivo de que la posterior reconstrucción guíe la búsqueda a nuevos óptimos locales. Una de las primeras elecciones a realizar es el tipo de componente que se va a eliminar de la solución. El elemento natural a eliminar es un *slot* de tiempo, debido a que es la unidad mínima de tiempo que se considera en una solución. Sin embargo, la eliminación de slots individuales dejaría huecos en el horario que serían difíciles de cubrir con otros elementos diferentes a los ya asignados anteriormente, llevándonos a obtener la misma solución original. Por este motivo, se elige como unidad a eliminar un día completo. Esta elección permite reestructurar un día completo cada vez, dejando más libertad al método de reconstrucción pero sin alejarse demasiado de la solución original en el espacio de búsqueda, en línea con la filosofía de IG. El número de días eliminados en cada iteración dependerá del valor del parámetro β, que debe ser ajustado experimentalmente.

Para evitar repetir la eliminación del mismo día en iteraciones consecutivas, se mantiene un memoria tabú que evitará eliminar durante varios días consecutivos el mismo día. Esta memoria mantiene una estructura FIFO (*First-In First-Out*), de manera que, una vez se haya llegado al límite establecido, el elemento más antiguo vuelve a estar disponible para ser eliminado.

La elección de los días a eliminar se realiza de manera completamente aleatoria, con el objetivo de favorecer la diversificación de la búsqueda. Esto se debe a que la fase de destrucción es la única que tiene como objetivo diversificar dentro del algoritmo propuesto.

3.3. Reconstrucción

El objetivo principal de la fase de reconstrucción es generar una nueva solución factible a partir de la solución resultante de la fase de destrucción, que no es factible ya que hay días sin ninguna actividad planificada en ninguno de los *slots*. Es deseable que la solución reconstruida sea de alta calidad, para ofrecer a la posterior fase de mejora un punto de partida prometedor para alcanzar el nuevo óptimo local, acelerando su convergencia.

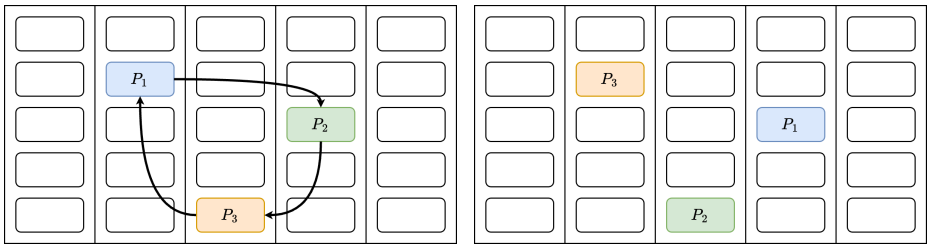
El método de reconstrucción más extendido dentro de IG es un procedimiento voraz simple que sea capaz de generar una solución en tiempos de cómputo reducidos. Además, el criterio voraz utilizado suele ser similar al utilizado en la fase de construcción en la mayoría de las investigaciones.

En el contexto de esta investigación, el modelo exacto utilizado en la fase de construcción es capaz de generar soluciones de alta calidad en tiempos de cómputo razonables, por lo que se propone como fase de reconstrucción utilizar el mismo modelo pero esta vez partiendo de una solución parcial, fijando las variables correspondientes a los *slots* ya asignados y dejando libertad al modelo para asignar los nuevos *slots* correspondientes a los días eliminados en el proceso de destrucción. La fase termina cuando el modelo ha generado una nueva solución factible.

3.4. Mejora

La solución obtenida tras la fase de reconstrucción no tiene por qué ser un óptimo local, incluso considerando el modelo exacto dentro de la fase de reconstrucción, ya que partirá de una solución parcial con muchas de las variables prefijadas. Con el objetivo de alcanzar un nuevo óptimo local a partir de la solución reconstruida se propone una búsqueda local basada en un movimiento de rotación de actividades entre profesores.

En concreto, el procedimiento de búsqueda explora iterativamente las actividades asociadas a cada profesor, analizando posibles intercambios con actividades de otros profesores. Esto forma un ciclo de intercambios de un tamaño predeterminado. El tamaño mínimo de ciclo es 2, lo que genera un intercambio clásico: el slot asignado al profesor 1 pasa al profesor 2 y el asignado al profesor 2 pasa al profesor 1. La Figura 1 muestra un posible movimiento de rotación que involucra a 3 profesores.



(a) Solución antes del movimiento. (b) Solución después del movimiento.

Figura 1: Ejemplo de rotación de actividades entre los profesores P_1, P_2 y P_3.

Los profesores involucrados son P_1, P_2 y P_3. En este caso concreto, el slot reservado para P_2 pasará a estar ocupado por P_1, mientras que P_2 ocupará el slot anteriormente ocupado por P_3. Finalmente, P_3 se moverá al hueco que ha dejado P_1 al moverse a su nuevo slot. Cabe destacar que la búsqueda local propuesta solo realizará aquellos movimientos que resulten en una solución factible, ignorando aquellos que incumplan alguna de las restricciones del problema.

Se considera que un movimiento produce una mejora siempre que reduzca los huecos disponibles en el horario o si mejora alguno de los criterios secundarios que puedan ser considerados. La búsqueda itera sobre todos los profesores con un tamaño de rotación prefijado como parámetro del algoritmo hasta no encontrar ningún movimiento que lleve a una solución mejor. El incremento en el tamaño de la rotación hace crecer de forma exponencial el número de movimientos posibles, por lo que se considerarán valores bajos para este parámetro.

4. Resultados experimentales

Dado que se trata de un problema real, el conjunto de instancias se han generado a partir de la plantilla de un centro público educativo que imparte tres niveles educativos principales: Educación Secundaria Obligatoria (ESO), con duración de 4 años; Bachillerato, con duración de 2 años; y Formación Profesional Básica o de Grado Medio (FP), con duración de 1 año. Se han generado un total de 5 instancias derivadas de los cursos 2021/2022 hasta 2025/2026 para poder realizar la comparativa frente a la operativa actual del centro.

La jornada escolar se organiza de lunes a viernes con 6 sesiones diarias de 60 minutos cada una, divididas en dos tramos: el primer tramo abarca el horario 08:15-11:15 (3 slots) y el segundo tramo el horario 11:45-14:45 (3 slots). Estos dos tramos están divididos por un periodo de descanso (recreo) de 30 minutos entre las 11:15 y las 11:45. Además, cada tramo horario tiene definida una demanda mínima de profesores para guardias de pasillo, recreo y convivencia, que deben ser cubiertas por el personal docente del centro.

El centro cuenta con 19 departamentos activos con un amplio catálogo de asignaturas, lo que supone un reto para cualquier algoritmo de optimización. La mayoría de asignaturas tienen entre 2 y 4 horas semanales, pero existen ámbitos específicos en programas de diversificación que pueden alcanzar hasta las 8 horas semanales. En total el centro dispone de 71 profesores, cada uno especializado en uno de los departamentos.

En total se dispone de 5 instancias, relativas a los últimos 5 cursos académicos en el centro: 2021/22, 2022/23, 2023/24, 2024/25 y 2025/26. Para analizar el comportamiento del algoritmo propuesto se lleva a cabo la comparativa con el software de caja negra HorW[4], que utiliza como entrada el mismo formato de instancia que el utilizado en esta investigación y genera el horario para el centro en ese curso académico. Cabe destacar que se trata del software que se está utilizando actualmente en el centro utilizado como referencia en este trabajo. Además, por requisitos del centro, se evalúan dos escenarios: uno con un tiempo de ejecución de 30 minutos y otro de 60 minutos (Tabla 1). Adicionalmente, se incluye en la comparativa el método exacto, para analizar la aportación de IG y sus búsquedas locales.

Las métricas utilizadas en todos los casos son: F.O., valor de la función objetivo (número de huecos restantes); y Pref (%), porcentaje de preferencias docentes satisfechas en la solución. Es importante reseñar que los tres algoritmos propuestos tratan de optimizar el número de huecos, mientras que las preferencias son un objetivo secundario. El hardware utilizado es un Intel Core i5 (2.60 GHz) con 32 GB de RAM.

Como se puede observar, el algoritmo muestra resultados competitivos, tanto en número de huecos libres (F.O.) como en porcentaje de preferencias satisfechas en las primeras instancias, mientras que en los años más recientes el comportamiento es justo el inverso. En todos los casos, se puede verificar que la propuesta de IG obtiene mejores resultados que el modelo exacto, que no es capaz de re-

[4] https://www.horw.es

		HorW		MILP		IG	
T(m.)	Instancia	F.O.	Pref. (%)	F.O.	Pref. (%)	F.O.	Pref. (%)
30	2122	210.50	58.33	192.00	55.56	166.00	62.96
	2223	172.00	55.84	190.50	51.95	187.50	59.74
	2324	136.00	72.14	183.50	58.57	168.50	61.43
	2425	154.00	64.24	172.00	55.76	166.50	57.58
	2526	185.50	64.18	236.50	55.22	205.50	55.22
60	2122	196.50	58.33	175.50	55.56	163.00	62.96
	2223	143.00	55.84	175.50	58.44	175.00	61.04
	2324	125.00	71.43	152.50	65.00	168.50	61.43
	2425	152.50	63.03	159.00	55.76	160.50	59.39
	2526	185.50	46.74	213.50	76.12	196.50	51.49

Tabla 1: Resultados obtenidos con un horizonte temporal de 30 y 60 minutos de ejecución por cada instancia.

solver ninguna de las instancias de forma óptima en el tiempo proporcionado debido a su complejidad. Cabe destacar que la mejora del modelo exacto cuando se permite un horizonte de tiempo más amplio es considerable, pero sigue estando aún lejos de IG tanto en calidad de la solución como en preferencias satisfechas. Analizando los resultados, IG emerge como una propuesta competitiva frente al software comercial utilizado hasta el momento, HorW, con la ventaja de ser un algoritmo completamente explicable en lugar de una propuesta de caja negra.

5. Conclusiones

En este trabajo se propone un algoritmo que hibrida un modelo de programación lineal entera mixta con un algoritmo basado en Iterated Greedy para generar soluciones de alta calidad para el problema de la planificación de horarios en centros escolares. La propuesta ofrece a los centros la posibilidad de automatizar la asignación de recursos en el centro escolar con el objetivo de minimizar los huecos en los horarios de cada profesor, respetando numerosas restricciones tanto estrictas como deseables. La propuesta ofrece mejores soluciones que las utilizadas actualmente por el centro, garantizando su aplicabilidad directa en escenarios reales.

Agradecimientos Esta investigación está parcialmente financiada mediante el apoyo de la Comunidad Autónoma de Madrid (ref. TEC-2024/COM-404), Ministerio de Economía y Competitividad (ref. PID2024-160226OB-C22 y PID2022-139543OB-C41), Ministerio para la Transformación Digital y de la Función Pública (Cátedra ENIA AI4DDS, ref. TSI-100930-2023-3), y por las Ayudas A5 del Plan Propio de la UPO en el marco del Programa FEDER Andalucía 2021-2027 (ref.: PPI2307).

Referencias

1. Andrade, P.R.d.L., Steiner, M.T.A., Góes, A.R.T.: Optimization in timetabling in schools using a mathematical model, local search and iterated local search procedures. Gestão & Produção **26**, e3421 (2019)
2. Birbas, T., Daskalaki, S., Housos, E.: School timetabling for quality student and teacher schedules. Journal of Scheduling **12**(2), 177–197 (2009)
3. Domenech, B., Lusa, A.: A milp model for the teacher assignment problem considering teachers' preferences. European Journal of Operational Research **249**(3), 1153–1160 (2016)
4. Dorneles, Á.P., De Araújo, O.C., Buriol, L.S.: A fix-and-optimize heuristic for the high school timetabling problem. Computers & Operations Research **52**, 29–38 (2014)
5. Drexl, A., Salewski, F.: Distribution requirements and compactness constraints in school timetabling. European Journal of Operational Research **102**(1), 193–214 (1997)
6. Even, S., Itai, A., Shamir, A.: On the complexity of time table and multi-commodity flow problems. In: 16th annual symposium on foundations of computer science (sfcs 1975). pp. 184–193. IEEE (1975)
7. Gestrelius, S., Aronsson, M., Peterson, A.: A milp-based heuristic for a commercial train timetabling problem. Transportation Research Procedia **27**, 569–576 (2017)
8. Gotlieb, C.: The construction of class-teacher timetables. In: IFIP congress. vol. 62, pp. 73–77 (1963)
9. Li, W., Chen, X., Li, J., Sang, H., Han, Y., Du, S.: An improved iterated greedy algorithm for distributed robotic flowshop scheduling with order constraints. Computers & Industrial Engineering **164**, 107907 (2022)
10. Lodi, A.: Mixed integer programming computation. In: 50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art, pp. 619–645. Springer (2009)
11. Post, G., Kingston, J.H., Ahmadi, S., Daskalaki, S., Gogos, C., Kyngas, J., Nurmi, C., Musliu, N., Pillay, N., Santos, H., et al.: Xhstt: an xml archive for high school timetabling problems in different countries. Annals of Operations Research **218**(1), 295–301 (2014)
12. Quintana, J.D., Martin-Santamaria, R., Sanchez-Oro, J., Duarte, A.: Solving the regenerator location problem with an iterated greedy approach. Applied Soft Computing **111**, 107659 (2021)
13. Ruiz, R., Stützle, T.: A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. European journal of operational research **177**(3), 2033–2049 (2007)
14. Sanchez-Oro, J., Duarte, A.: Iterated greedy algorithm for performing community detection in social networks. Future Generation Computer Systems **88**, 785–791 (2018)
15. Schaerf, A.: A survey of automated timetabling. Artificial intelligence review **13**(2), 87–127 (1999)
16. Soto-DeLaVega, M., Molina, C., Quiroz, A.T., Cera, J.C., Marrugo, J.C.L.: A mixed integer linear programming model for school timetable in cartagena. In: Workshop on Engineering Applications. pp. 290–300. Springer (2023)
17. Werra, D.: Construction of school timetables by flow methods. INFOR: Information Systems and Operational Research **9**(1), 12–22 (1971)