33

REPOSITORIO INTERACTIVO PARA FACILITAR EL APRENDIZAJE DE ALGORITMOS HEURÍSTICOS APLICADOS A PROBLEMAS DE OPTIMIZACIÓN

INTERACTIVE REPOSITORY TO EASE THE LEARNING OF HEURISTIC ALGORITHMS APPLIED TO OPTIMIZATION PROBLEMS

Israel López Plata
ilopezpl@ull.edu.es
Christopher Expósito Izquierdo
cexposit@ull.edu.es
María Belén Melián Batista
mbmelian@ull.edu.es
José Marcos Moreno Vega
jmmoreno@ull.edu.es

Universidad de La Laguna, España

RESUMEN

La algoritmia es una de las ramas de aprendizaje principales en la formación de un Graduado en Ingeniería Informática, ya que permite abordar la resolución de un gran número de problemas de forma automatizada. Las (meta)heurísticas son técnicas aproximadas destinadas a resolver problemas de optimización mediante una demanda de recursos computacionales reducida. Esto hace que puedan resolver problemas de gran complejidad.

En el presente trabajo se expone el repositorio creado con el fin de explicar el conjunto de algoritmos heurísticos de mayor utilización en la resolución de diferentes tipos de problemas, los cuales son impartidos frecuentemente en las asignaturas que componen el Grado de Ingeniería Informática.

Con la idea de complementar la comprensión de los algoritmos, el repositorio incluye un conjunto de problemas de optimización altamente estudiados en la literatura, de tal forma que el alumno pueda comprender el comportamiento de los algoritmos disponibles sobre problemas de distintas características.

Además de una explicación detallada de tanto los algoritmos heurísticos como de los problemas de optimización, se incluyen ejemplos interactivos de funcionamiento, sobre los cuales se puede realizar un análisis de resultados así como una comparativa entre los mismos. Con ello se pretende fomentar la interacción del alumno en el aprendizaje, mediante la ejecución paso a paso de los diferentes ejemplos de algoritmos a través de un software específico, consiguiendo que el alumno comprenda en profundidad los distintos algoritmos, sus características y sus posibles usos.

PALABRAS CLAVE: Heurísticas; Metaheurísticas; Optimización; Ejemplos interactivos; Educación 2.0.

ABSTRACT

Algorithmics is one of the main branches of learning in the studies of a Graduate on Computer Engineering, because it allows to address the resolution of a huge amount of problems on an automated way. (Meta)heuristics are approximate techniques aimed at solving optimization problems by means of a reduce number of computational resources. This allows they can solve high complex optimization problems.

The present work exposes the repository created with the goal to explain the set of heuristic algorithms with more use in the resolution of different types of problems, and that are frequently taugh in the subjects that compose the Grade on Computer Engineering.

With the idea of complement the understanding of the algorithms, the repository includes a set of optimization problems highly studied in the literature, in such a way that the student can learn the behaviour of the available algorithms in problems with different characteristics.

In addition to a detailed explanation of the heuristic algorithms as well the optimization problems, a set interactive examples of execution are included, on which an analysis of results can be carried out as well as a comparison between them. The idea is to encourage the interaction of the student in learning, through a step-by-step execution of different algorithm examples using a specific software, getting the student to understand in depth the different algorithms, their characteristics and their possible uses.

KEYWORDS: Heuristics; Metaheuristics; Optimization; Interactive examples; Education 2.0.

INRODUCCIÓN

El aprendizaje de cualquier tipo de disciplina se ha basado principalmente en lo que se conoce como el *paradigma clásico de educación* (Miller 1988Miller1988), implantado en todos los ámbitos en la mayoría de sistemas educativos occidentales. Este tipo de educación basa su funcionamiento en los siguientes puntos:

– La principal vía de información son las clases magistrales. Esto provoca una enseñanza donde la información fluye de manera unidireccional entre profesor y alumno, con pocas posibilidades de réplica por parte de este último.
– La evaluación se realiza por medio de exámenes escritos, lo que fomenta la memorización de los contenidos en lugar de su comprensión por parte del alumno. Esto se debe en su mayor parte por la incapacidad de proporcionar al alumno un entorno con el cual pueda experimentar y en consecuencia comprender los conceptos con una mayor facilidad.
– Los canales de información son limitados, utilizando habitualmente libros, apuntes o transparencias proporcionadas por el profesor.

En los últimos años se han planteado diferentes modelos ante la necesidad de cambiar el paradigma clásico de la educación (Tomozii and Topala 2014Tomozii and Topala2014). En estos nuevos modelos, se desea fomentar en el alumno el autodescubrimiento y la experimentación para el aprendizaje de los conceptos, proporcionándole los medios adecuados para ello.

Con el fin de adaptar la educación a los nuevos paradigmas y dada la gran mejora en la tecnología y las telecomunicaciones, se impone un nuevo tipo de educación denominada **educación 2.0** (Walks 2013Walks2013). Este tipo de educación parte de la posibilidad de tener un mayor número de recursos, normalmente a través de la web, por lo que se puede realizar un proceso de aprendizaje contínuo y a través de diversos medios, fomentando así el aprendizaje autónomo por parte del alumno. Entre las características de la educación 2.0 encontramos:

– Las clases magistrales se encuentran reforzadas con un mayor número de recursos didácticos, como vídeos, cuestionarios o ejemplos interactivos.
– Los recursos didácticos de las diferentes asignaturas se encuentran centralizados y en diferentes formatos, normalmente a través de una web específica para una determinada asignatura.
– Los canales de información no solo se limitan a los presenciales. Se incluyen cuestionarios, foros e incluso actividades supervisadas.

Con ello se consigue que la información fluya de manera multidireccional, provocando una mayor participación por parte del alumno y una mayor posibilidad de la existencia de feedback entre alumno y profesor.

– Se fomenta el aprendizaje colaborativo entre los alumnos, dado que se poseen herramientas para ello. Con este tipo de aprendizaje se fuerza al estudiante a deducir las soluciones a diferentes problemas, profundizando así en el aprendizaje de los conceptos.

Los entornos que permiten este tipo de aprendizaje se conocen como entornos de **Aprendizaje Mejorados por la Tecnología** (Technology-Enhanced Learning, TEL) (Balacheff et al. 2008Balacheff et al.2008). En estos nuevos entornos, las tecnologías de la comunicación proveen un amplio rango de oportunidades para su utilización como complemento de los sistemas de aprendizaje tradicionales.

Las herramientas software dedicadas se han consolidado como un instrumento de indudable valor pedagógico en las instituciones educacionales existentes (Castro et al. 2009Castro-Sánchez et al.2009). Existe un gran número de ejemplos de aplicaciones que ya se están aplicando en entornos pedagógicos de campos heterogéneos, como las redes de comunicaciones (Sanguino et al. 2013Mateo-Sanguino et al.2013), la arquitectura de computadores (Ozturk 2011Ozturk2011) o la ingeniería biomédica (Guerrero et al. 2007Guerrero et al.2007).

En el presente trabajo se propone la creación de un repositorio web que ayude en el aprendizaje de la resolución de problemas de optimización utilizando para ello algoritmos (meta)heurísticos. Siguiendo los principios de la educación 2.0, el repositorio presentado se encuentra centralizado y obtiene la información de diferentes fuentes. Además, a través de una serie de ejemplos interactivos, se fomenta la experimentación en el aprendizaje así como el uso colaborativo de la herramienta.

El presente trabajo continúa con una explicación general del repositorio propuesto para posteriormente hablar de las 3 partes que lo componen: repositorio de heurísticas, repositorio de problemas de optimización y ejemplos interactivos. Por último, se extraen una serie de conclusiones y se proponen un conjunto de líneas de trabajo futuras.

REPOSITORIO INTERACTIVO DE HEURÍSTICAS

A lo largo de los estudios de Graduado en Ingeniería Informática se imparten conceptos de diferentes ramas profesionales de la ingeniería, por lo que se trata de una enseñanza diversa y amplia. Sin embargo, existen una serie de conceptos que poseen un gran grado de transversalidad y que por lo tanto deben ser impartidos en varias

asignaturas de diferentes ramas profesionales a lo largo de la carrera. Éste es el caso de la algoritmia en general y de las (meta)heurísticas y los problemas de optimización en particular.

En el ámbito académico, las disciplinas de carácter transversal se suelen impartir únicamente centradas en cumplir con los objetivos fijados en la asignatura, por lo que es habitual que no se expliquen todos los conceptos que las componen o que la explicación no tenga la suficiente profundidad. Si se desea un aprendizaje completo de este tipo de disciplinas, el alumno necesita un sistema que albergue todos los conceptos que las componen explicados de la manera más detallada posible.

El Repositorio Interactivo de Heurísticas presentado en este trabajo es un entorno web donde se recopila información sobre el conjunto de algoritmos (meta)heurísticos de mayor difusión en el entorno académico. Esta información posee fines didácticos y se combina con diferentes ejemplos de ejecución, de tal forma que el alumno comprenda mejor cada uno de los conceptos.

Además de sobre algoritmos heurísticos, el repositorio incluye información sobre problemas de optimización bien conocidos en la literatura científica y con características heterogéneas, de tal forma que pueda comprobarse el comportamiento de las (meta)heurísticas explicadas en su aplicación en situaciones diversas.

Los objetivos que se desean cumplir con el Repositorio Interactivo de Heurísticas son los siguientes:

– **Centralización:** Albergar en un solo sitio web toda la información necesaria para el aprendizaje de técnicas de optimización heurísticas y su aplicación sobre problemas de optimización. Con ello se consigue poseer un entorno donde tanto el alumno como el profesor saben que pueden consultar dicha información, y utilizarla en consecuencia para cualquier asignatura.
– **Fiabilidad**: Garantizar que la información incluida en el repositorio es completa y fiable, ya que en su mayor parte se encuentra añadida y revisada por expertos en la materia.
– **Actualización**: Al tratarse de un repositorio web, se puede permitir el acceso remoto para editar su información, obteniendo así un sistema ampliable y actualizable tanto por el profesor como por el alumno. Esto se aplica tanto para las (meta)heurísticas como para los problemas de optimización o los ejemplos interactivos.

El repositorio presentado se organiza en 3 secciones claramente diferenciadas, las cuales se relacionan entre sí tal y como se especifica en el siguiente esquema:

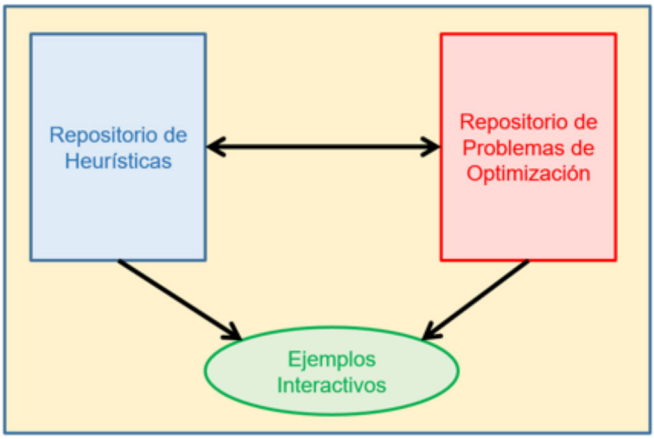

Figura 1. Esquema general del Repositorio Interactivo.

Las diferentes secciones representadas en el esquema anterior son las siguientes:

– **Repositorio de Heurísticas:** Recopila todas las (meta)heurísticas del repositorio, empezando por una descripción a nivel general de las mismas. Además, para cada (meta)heurística se indican sus particularidades, en que mejora la búsqueda de soluciones y para que tipos de problemas de optimización suele ser de mas utilidad. Los detalles de esta sección se explican en **Repositorio de Heurísticas**.
– **Repositorio de Problemas de Optimización:** Recoge un conjunto de problemas de optimización a los que aplicar las (meta)heurísticas incluidas en el repositorio. Estos problemas son altamente conocidos en la literatura y con un gran valor didáctico. Para cada problema se explica entre otras cosas en que consiste, por qué es necesaria la aplicación de un algoritmo heurístico para solucionarlo así como las aplicaciones del problema en entornos reales. También se indican que algoritmos heurísticos suelen tener un buen funcionamiento para la resolución de cada problema. El contenido de esta sección se explica en detalle en el apartado **Repositorio de Problemas de Optimización**.
– **Ejemplos Interactivos:** Conjunto de ejemplos que muestran la resolución de un problema de optimización con la utilización de una o varias heurísticas. Para ello se utiliza un software específico que permite diferentes modos de ejecución, entre los que se encuentra un paso a paso con el que el alumno puede comprender el funcionamiento de cada heurística sobre un problema en detalle. El contenido de los ejemplos interactivos se detalla en **Ejemplos Interactivos**.

Tal y como se observa en el esquema, cada una de las 3 partes se encuentran relacionadas entre sí. Los repositorios de técnicas de optimización heurísticas y de problemas de optimización poseen enlaces directos en su contenido, de una (meta)heurística a un problema y viceversa. Con ello se complementa en gran medida el aprendizaje de cada algoritmo o problema, consiguiendo que el alumno tenga una visión más amplia de los contenidos.

Los ejemplos interactivos poseen una relación directa con los 2 repositorios, ya que dichos ejemplos se centran en la ejecución de diferentes algoritmos heurísticos sobre un conjunto de problemas de optimización.

El repositorio se encuentra organizado en forma de blog en la dirección **https://optimizationproblemsblog.wordpress.com**, con el objetivo de conseguir una buena organización de sus contenidos, así como para facilitar su edición. Todos los apartados del blog poseen la misma estructura, permitiendo una navegación por sus contenidos rápida y sencilla.

REPOSITORIO DE HEURÍSTICAS

Las heurísticas son algoritmos que utilizan información de fácil acceso y poco aplicable para controlar la resolución de problemas de optimización (Martí et al. 2018Martí et al.2018). El término heurística es aplicado en un gran número de disciplinas, como pueden ser Matemáticas, Psicología, Investigación Operativa y Computación, entre otras, y significa el arte de inventar o descubrir hechos valiéndose de hipótesis o principios que, aun no siendo verdaderos, estimulan la investigación.

En el campo de la Investigación Operativa, una heurística se define como una técnica que encuentra soluciones de buena calidad con un coste computacional razonable, pero que no es capaz, en muchos casos, de garantizar ni la factibilidad, ni la optimalidad, ni establecer lo cerca de la optimalidad que está una solución factible particular. Es en este campo donde las heurísticas se han convertido en muchos casos en elementos imprescindibles para la resolución práctica de un problema.

A modo resumen, el uso de heurísticas para la resolución de un determinado problema posee una serie de ventajas en comparación con el uso de técnicas exactas:

– Obtiene soluciones de una calidad aceptable en un corto espacio de tiempo.
– Un conjunto de algoritmos genéricos o una combinación de los mismos permiten solucionar problemas de distintas características con una alta eficiencia.

Dado que un algoritmo heurístico no garantiza la obtención de una solución óptima, su utilización es más frecuente en las siguientes casuísticas:

– No se dispone de un procedimiento exacto para resolver el problema planteado, o si se dispone de este, es ineficiente.
– Se prefiere abordar, por medio de heurísticas, una representación mas ajustada del problema planteado que una versión menos realista de tal problema que pueda resolverse de forma exacta.
– No se poseen conocimientos específicos sobre el problema que permitan abordarlo de forma exacta.
– Se tiene que resolver repetidas veces un mismo problema, probablemente con datos distintos.
– Se quiere crear un sistema de ayuda a la decisión, donde se deben proporcionar diferentes soluciones de buena calidad al decisor de forma instantánea, de tal forma que el mismo pueda elegir la más adecuada.

El modo de operar de todo algoritmo heurístico es similar para la mayoría de los algoritmos existentes en la literatura. La heurística se inicia creando soluciones factibles de una forma rápida con el fin de poder devolver un resultado en un corto espacio de tiempo. Esta solución se suele obtener de forma aleatoria o mediante métodos constructivos. A partir de la solución inicial, se obtienen nuevas soluciones de cada vez mayor calidad. Es decisión del diseñador del algoritmo determinar el momento en el que la heurística deja de intentar mejorar la solución y, o termina su ejecución o vuelve a iniciar el proceso a partir de una nueva solución inicial.

Un buen diseño de una heurística para resolver un determinado problema requiere del conocimiento de los detalles de dicho problema. Sin embargo, existen un conjunto de factores configurables por cada heurísitca que determinan el comportamiento y la calidad de la misma. Son independientes del problema a resolver y no todos son necesarios para todos los algoritmos heurísticos. Estos factores se explican con detalle en el repositorio presentado para cada heurística y son los siguientes:

– **Criterio de aceptación:** Determina si una heurística acepta una nueva solución generada como la mejor solución. Cada heurística puede usar el criterio de aceptación que más le convenga para cumplir con sus objetivos, existiendo así una variedad ilimitada de criterios de aceptación. Algunos ejemplos pueden ser aceptar siempre las soluciones que mejoren en calidad o aquellas que se encuentre dentro de un rango de mejora con respecto a la mejor solución.
– **Criterio de parada:** Este tipo de criterios es muy usual en las heurísticas de búsqueda. Indican a la heurística cuando termina su ejecución y por lo tanto la mejor solución obtenida hasta el momento pasa a ser la solución devuelta por la heurística. Existen distintas variantes

de criterios de parada y es de gran importancia elegir el criterio adecuado para cada heurística en particular. Ejemplos de criterios de parada pueden ser un número determinado de iteraciones en un bucle o el alcance de un límite temporal de ejecución.

– **Otros parámetros:** Muchas heurísticas utilizan una serie de parámetros que puedan cambiar el comportamiento del algoritmo dependiendo de cómo se encuentren estos configurados. La inclusión de este tipo de parámetros consigue una gran interactividad, permitiendo variar en gran medida los resultados obtenidos en una misma heurística.

Además de heurísticas, el presente repositorio contiene una serie de algoritmos basados en **metaheurísticas** (Gendreau and Potvin 2010Gendreau and Potvin2010). Una metaheurística se puede definir como *una estrategia inteligente para diseñar o mejorar procedimientos heurísticos muy generales con un alto rendimiento* (Melián et al. 2013Melián-Batista et al.2003). Las metaheurísticas realizan búsquedas mas eficientes del espacio de soluciones, lo que implica la obtención de soluciones de mayor calidad a costa de un mayor tiempo de ejecución.

Un buen uso de algoritmos metaheurísticos depende en gran medida de las características del problema que se desea resolver (Talbi 2009Talbi2009), por lo que es importante realizar una buena elección de la metaheurística a aplicar, así como de los parámetros de la misma.

En el Repositorio Interactivo de Heurísticas se incluyen los algoritmos heurísticos y metaheurísticos de mayor difusión en la literatura y que poseen un valor didáctico, de tal forma que se integran en el plan de estudios de varias asignaturas dentro del Grado en Ingeniería Informática en la Universidad de La Laguna. Actualmente, los algoritmos descritos en el repositorio son los siguientes, aunque el mismo se encuentra preparado para ser ampliado con mas heurísticas o metaheurísticas:

– **Búsqueda Aleatoria (Rastrigin 1986**Rastrigin1986**):** Algoritmo que genera un número específico de soluciones completamente aleatorias dentro de la región de soluciones factibles del problema. La solución final del algoritmo es aquella solución de mayor calidad de entre todas las generadas de forma aleatoria.

– **Búsqueda Local (Aarts and Lenstra 1997**Aarts and Lenstra1997**):** Una búsqueda local es una heurística en la que, a partir de una solución inicial, se va buscando en el entorno de la solución actual (soluciones vecinas) y aceptando aquellos resultados que la mejoren. Esta búsqueda sigue hasta que la solución actual no se pueda mejorar con las soluciones que se encuentran en su entorno, y por lo tanto se obtiene lo que se conoce como Óptimo Local.

– **Búsqueda Local Guiada (Voudouris and Tsang 2003**Voudouris and Tsang2003**):** Algoritmo metaheurístico que mejora el comportamiento de la búsqueda local estándar. La búsqueda local

guiada añade una serie de penalizaciones al valor de la solución en el momento en el que se detecta que la búsqueda local ha alcanzado un óptimo local, con el objetivo de explorar más allá de dicho óptimo local.

– **Búsqueda Multiarranque (Martí et al. 2013**Martí et al.2013**):** Algoritmo heurístico que consiste en la ejecución de múltiples heurísticas cada una sobre su propia solución inicial, almacenando el mejor resultado que devuelto por dichas heurísticas. En cada arranque se puede cambiar tanto la generación inicial como las heurísticas ejecutadas y su configuración. Es habitual combinar búsquedas multiarranque con algoritmos que devuelven óptimos locales como la Búsqueda Local.

– **Recocido Simulado (Simulated Annealing) (Kirkpatrick 1983**Kirkpatrick et al.1983**):** Algoritmo heurístico inspirado en el enfriamiento controlado de un sólido después de ser calentado. En este caso, se maneja el ratio de aceptación de soluciones, el cual es cada vez mayor según avanza la ejecución del algoritmo.

– **Greedy Randomized Adaptative Search Procedure (GRASP) (Feo and Resende 1995**Feo and Resende1995**):** Un GRASP es un algoritmo metaheurístico en el que cada iteración se basa en 2 pasos: (i) una fase constructiva que genera soluciones a partir de construirlas desde cero añadiendo elementos a la misma y (ii) un postprocesamiento que mejora la solución construida en el paso anterior. En algunos casos, al inicio del algoritmo se realiza un Preprocesamiento cuyo objetivo es facilitar en la medida de lo posible las fases constructivas posteriores.

– **Variable Neighbourhood Search (VNS) (Hansen et al. 2010**Hansen et al.2010**):** Metaheurística que obtiene soluciones explorando el entorno de la solución actual, a través de un cambio sistemático de diferentes estructuras de entorno. Su ejecución se basa en 3 fases (i) generar una solución inicial sobre la que empezar a explorar y, para cada exploración se ejecutan (ii) una fase de perturbación que diversifica la búsqueda y (iii) una búsqueda local que obtiene el óptimo local dentro del entorno explorado.

– **Algoritmos Genéticos (Liao and Sun 2001**Liao and Sun2001**):** Conjunto de algoritmos metaheurísticos inspirados en la evolución biológica y la selección genética. Este tipo de algoritmos *evolucionan* a una población de soluciones mediante cambios aleatorios, para posteriormente realizar un proceso de selección con aquellas más prometedoras.

Con el fin de plasmar toda esta información de tal forma que sea comprensible para el alumno se crea el **Repositorio de Heurísticas**. Este repositorio se divide en dos secciones claramente diferenciadas, una de ellas centrada en la explicación de

conceptos generales y una segunda en la que se explican cada uno de los algoritmos incluidos en el repositorio.

En la sección de conceptos generales se explican aquellos que son necesarios para comprender la optimización a través de algoritmos heurísticos. Esto incluye:

– Definición de las heurísticas y metaheurísticas.
– El manejo de soluciones en este tipo de algoritmos.
– Configuración de los algoritmos hurísticos. Elementos dependientes del algoritmo.
– Consejos de cara a la elección de algoritmos y configuraciones para resolver distintos tipos de problemas.

Por otro lado, el repositorio incluye un apartado por cada algoritmo incluido, centralizando así toda la información del mismo. Cada uno de estos apartados posee siempre la misma estructura, lo que facilita en gran medida su usabilidad así como la navegación por cada uno de sus apartados. La explicación de cada una de las (meta)heurísticas se compone de los siguientes elementos:

1. **Descripción general de la (meta)heurística:** Descripción del algoritmo. Indica cómo funciona, en que se basa para su funcionamiento e incluso la historia detrás del origen del algoritmo.
2. **Paso a paso:** Explica en detalle cada uno de los pasos que sigue el algoritmo para la obtención de soluciones. Para ello se apoya en la utilización de diagramas y pseudocódigos.
3. **Justificación de resultados:** Se indica por qué el algoritmo devuelve buenos resultados en determinadas situaciones y, si fuera el caso, en que mejora otros algoritmos incluidos en el repositorio.
4. **Problemas comunes:** Sección que explica en qué tipo de problemas de los incluidos en el repositorio el algoritmo devuelve buenos resultados.
5. **Enlace a ejemplos interactivos:** Enlace a los ejemplos interactivos incluidos en el repositorio en los que se utilice la heurística o metaheurística explicada.

La interfaz de usuario utilizada para mostrar la información de cada algoritmo heurístico se muestra en la Figura 2.

Figura 2. Esquema general del Repositorio Interactivo.

La interfaz utilizada para mostrar los datos de cada heurística consta de los siguientes puntos:

1. **Cabecera del repositorio**. Cabecera con el título del repositorio de heurísticas, así como los logos de la Universidad de La Laguna.
2. **Menú principal**. Menú con el que acceder a las diferentes secciones del repositorio de forma directa.
3. **Contenido**. Alberga toda la información referente a la página en cuestión. Puede contener texto, imágenes, pseudocódigos, ficheros descargables así como enlaces a secciones del repositorio con las que tenga algún tipo de relación.
4. **Área de widgets**. Sección con un conjunto de widgets que pueden ser de utilidad a la hora de manejarse con el repositorio, como son las búsquedas, las últimas publicaciones e incluso publicaciones antiguas. Este área puede ser ampliada con más widgets en un futuro.

REPOSITORIO DE PROBLEMAS DE OPTIMIZACIÓN

La optimización es aquel proceso que implica la búsqueda de la mejor alternativa posible de cara al desarrollo de una determinada actividad (Zelinka et al. 2013Zelinka et al.2013). Esto se puede aplicar a diferentes ejemplos de la vida real, como encontrar la ruta más corta entre 2 puntos (Eiselt and Sandblom 2000Eiselt and Sandblom2000), encontrar la localización de una infraestructura de tal forma que cubra al mayor

número de usuarios (Church and ReVelle 1974Church and ReVelle1974) o determinar la posición de los objetos dentro de un contenedor de forma que se aproveche el espacio lo máximo posible (Bortfeldt and Wäscher 2013Bortfeldt and Wäscher2013).

La resolución de un problema de optimización implica encontrar la solución factible del problema que sea óptima según el valor de la función objetivo. La función objetivo permite asignar un valor cuantitativo a las soluciones del problema de tal forma que las mismas puedan compararse entre sí. Esta función es una parte inherente del problema y siempre debe ser definida con el mismo.

Los problemas de optimización se resuelven a través de los procesos algorítmicos conocidos como **métodos de optimización** (Cavazzuti 2012Cavazzuti2012), los cuales son una serie de operaciones finitas y bien definidas que permiten obtener una solución factible del problema de optimización.

El estudio de este tipo de problemas es un área de investigación de gran extensión en todas las universidades del mundo, en las cuales se imparte con gran frecuencia dentro de sus programas de estudios. La resolución de problemas de optimización requiere de un conjunto de fases a realizar (Eiselt and Sandblom 2010Eiselt and Sandblom2010), explicadas a continuación:

1. **Identificación del problema**: Consiste en captar todos los requisitos que requiere la realización del problema, obteniendo así toda la información necesaria por parte de la persona que desea su resolución.
2. **Formalización del problema**: En esta fase se formaliza de forma matemática el problema de optimización. Permite definir todas las restricciones del problema así como su función objetivo.
3. **Selección de los métodos de optimización**: Fase en la que se pretende diseñar los pasos a seguir por el algoritmo que debe resolver el problema. En este punto se puede escoger entre el gran número de técnicas existentes en la literatura u optar por el diseño de una técnica nueva.
4. **Configuración de los métodos de optimización**: Consiste en configurar todas las características que requieran los métodos de optimización para su ejecución.
5. **Ejecución de los métodos de optimización**: Fase en la que se ejecutan los métodos de optimización diseñados con el fin de generar soluciones factibles para el problema.
6. **Análisis de resultados**: Último punto del proceso de resolución, donde se analizan los resultados de la fase anterior y se obtienen conclusiones.

Dada la introducción de la tecnología en todos los ámbitos profesionales y la mayor competitividad que ello implica, cada vez se requiere con mayor frecuencia la optimización de los procesos relacionados con las diferentes actividades profesionales. Teniendo en cuenta estas circunstancias, las empresas demandan los servicios de un profesional que optimice en la medida de lo posible todos sus procesos. Las universidades se han adaptado a estos requerimientos, fomentando la formación de estos profesionales en sus diferentes titulaciones.

Los métodos de optimización utilizados para resolver los diferentes problemas son muy diversos y su elección depende de las características de los resultados que se desean conseguir. Entre los métodos existentes destacan los determinísticos, los cuales son una serie de pasos exactos en los que se garantiza que para la misma entrada, siempre se obtiene la misma salida. Esto contrasta con los métodos (meta)heurísticos, los cuales añaden variaciones probabilísticas y por lo tanto permiten obtener conjuntos de soluciones más diversos que en el caso de los determinísticos.

Los problemas de optimización incluidos en el Repositorio Interactivo de Heurísticas son aquellos que se consideran que tienen un valor didáctico alto y, al igual que ocurre con los algoritmos heurísticos del repositorio, se imparten a lo largo del Grado en Ingeniería Informática en la Universidad de La Laguna. Además, los problemas de optimización incluidos representan ejemplos sencillos de los diferentes tipos de problemas existentes en la literatura. En el repositorio de problemas de optimización se incluyen los siguientes problemas, los cuales pueden ser ampliados en un futuro:

– **Problema del viajante de comercio (Steiglitz and Papadimitriou 1982**Steiglitz and Papadimitriou1982**):** El problema del viajante de comercio, comunmente conocido por sus siglas en inglés TSP (Travelling Salesman Problem), es uno de los problemas más utilizados para la introducción al alumnado a la resolución de problemas NP-Duros. El problema parte de un grafo completo en el que cada uno de sus nodos representa una ciudad. Se busca la ruta mínima que, partiendo de una ciudad en concreto, recorra todas ciudades del grafo. Este problema es frecuentemente explicado en entornos didácticos, ya que permite una mejor comprensión de problemas complejos (NP-Duros) así como afianzar conceptos sobre la teoría de grafos.
– **Knapsack Problem (Karp 1972**Karp1972**):** El problema de la mochila (Knapsack Problem, KP) representa uno de los ejemplos más estudiados de problemas de optimización combinatoria. En el mismo, se pretende llenar una mochila con un conjunto de objetos en los que cada uno posee un valor y un peso. Se debe maximizar el valor de los objetos insertados en la mochila, teniendo en cuenta que esta posee un límite de peso. Este problema posee en la literatura un gran número de propuestas de resolución, desde algoritmos

determinísticos con algoritmos voraces o combinatorios hasta diferentes tipos de algoritmos heurísticos.

– **Problema de rutas de vehículos (Golden et al. 2008**Golden et al.2008**):** El problema de rutas de vehículos o VRP por sus siglas en inglés (Vehicle Routing Problem) representa un tipo de problemas de gran difusión en el ámbito científico. Consiste en, dado un grafo completo, gestionar las rutas de una flota de vehículos que parten de un nodo específico del grafo al que se considera almacén. Cada ruta sirve un conjunto de clientes representados por nodos del grafo, y empieza y finaliza en el almacén. Se debe conseguir las rutas mínimas que sirvan a todos los clientes del grafo. Dada su gran aplicación en entornos reales, el VRP posee un gran número de variantes que derivan en diferentes líneas de investigación, por lo que actualmente es uno de los problemas mas estudiados en la literatura.

– **Job Shop Schedulling Problem (Brucker and Knust 2012**Brucker and Knust2012**):** El problema de planificación de trabajos (Job Shop Schedulling Problem, JSP) es uno de los problemas más conocidos de optimización combinatoria. En el problema se poseen una serie de trabajos en los que cada uno de ellos tiene un tiempo de procesado. Para atender estos trabajos, se dispone de un conjunto de máquinas, cada una de ellas con diferente potencia de procesado. Se desea asignar los trabajos a las máquinas correspondientes de tal forma que se minimice el *makespan* de su planificación.

– **Árbol de Recubrimiento Mínimo** (**Cormen et al. 2001**Cormen et al.2001**):** El árbol de recubrimiento mínimo (Minimum Spanning Tree) es un concepto propio de la teoría de grafos. Representa al subconjunto de aristas de un grafo que conectan todos los nodos del mismo, de tal forma que la suma del peso de dichas aristas sea mínimo. La obtención de este árbol consiste en un problema NP-Duro dada su complejidad.

– **Problema de Asignación Cuadrática (Lawler 1963**Lawler1963**):** El problema de asignación cuadrática (Quadratic Assignment Problem, QAP) consiste en decidir dónde colocar un conjunto de instalaciones de entre un conjunto de lugares posibles, de tal forma que se minimice el coste de dicha asignación. Este coste depende de diversos factores como la localización, su distancia a las demás, el flujo existente entre las mismas y más.

La información relacionada con la resolución de problemas de optimización se incluye en el **Repositorio de Problemas de Optimización**. Al igual que en el repositorio explicado en la sección anterior, este se organiza en 2 partes. En una primera, se incluyen la información necesaria

para comprender la optimización, como son los tipos de problemas, sus parámetros configurables así como otros conceptos de carácter general.

La segunda sección del repositorio se centra en explicar cada uno de los problemas de optimización incluidos en el mismo. Los apartados que componen la explicación de los problemas de optimización son los siguientes:

1. **Descripción general del problema:** Descripción del problema, donde se indica lo que se desea conseguir con el mismo, los elementos que se encuentran implicados así como el origen de dicho problema.
2. **Objetivos y variantes:** Explica el objetivo principal del problema en detalle, así como las restricciones que impone. Muestra las variantes del problema de mayor difusión en la literatura e indica su diferencia con respecto a la versión genérica del mismo.
3. **Complejidad del problema:** Justifica de forma exacta por qué el problema es de difícil resolución y por lo tanto, requiere de la aplicación de algoritmos heurísticos para su resolución en un tiempo computacional razonable.
4. **Aplicaciones reales:** Muestra una serie de ejemplos reales donde se aplica el problema descrito. Con ello se pretende facilitar la comprensión del alumno sobre el problema, ya que impide que considere al mismo como un concepto abstracto e irreal.
5. **Algoritmos heurísticos:** Tipos de algoritmos heurísticos que, bajo la experiencia de los autores, suelen funcionar para resolver el problema descrito.
6. **Enlace a ejemplos interactivos:** Enlace a los ejemplos interactivos incluidos en el repositorio en los que se resuelve el problema de optimización.

La interfaz de usuario del repositorio de problemas de optimización sigue el mismo esquema que la vista en el repositorio de heurísticas, englobando así toda la información necesaria en una estructura que resulta común para todo el aplicativo.

EJEMPLOS INTERACTIVOS

El repositorio presentado en este artículo tiene como objetivo principal facilitar el aprendizaje de algoritmos heurísticos aplicados a problemas de optimización. Esto se consigue con una información completa, actualizada y perfectamente organizada, la cual el alumno pueda consultar siempre que lo considere necesario.

En el ámbito de la psicología se reconocen diferentes tipos de aprendizaje (Willingham et al. 2015Willingham et al.2015), los cuales son distintas formas de hacer que un alumno obtenga los conocimientos requeridos. Uno de estos tipos es el **aprendizaje por descubrimiento o heurístico** (Bruner 1961Bruner1961), en el cual el alumno obtiene los conceptos por sí mismo, a través de la experimentación y observación. Con el fin de reforzar este tipo de aprendizaje, en el repositorio se incluye un conjunto de **ejemplos de ejecución interactivos**, donde se detalla el funcionamiento de un determinado algoritmo heurístico para resolver diferentes problemas de optimización.

Para cada ejemplo interactivo se incluye un conjunto de datos de ayuda al alumno como son la explicación del objetivo del ejemplo, una guía de ejecución, los resultados esperados o los enlaces a los problemas o algoritmos implicados. Además, se permite la configuración del experimento, permitiendo elegir entre diferentes instancias del problema, así como entre distintas configuraciones tanto del algoritmo heurístico como del problema.

La ejecución de los ejemplos puede ser completa e incluso paso a paso, donde el alumno observe los cambios en la solución para cada una de las iteraciones que necesita el algoritmo para resolver el problema. Con esta última forma de ejecución se permite al alumno comprender en detalle mediante interacción tanto las (meta)heurísticas como los problemas de optimización a estudiar, lo que influye en gran medida su aprendizaje. También facilita que el alumno pueda realizar preguntas sobre una parte muy concreta de la ejecución, fomentando el *feedback* y el aprendizaje colaborativo.

Al finalizar la ejecución de un ejemplo interactivo se muestra un análisis de resultados así como una posible comparativa entre los diferentes algoritmos utilizados para resolver el mismo problema. Con ello el alumno obtiene una visión global del comportamiento de cada uno de los algoritmos. Por otro lado, permite deducir los posibles usos de los algoritmos heurísticos, así como sobre qué tipos de problemas ofrecen un mejor comportamiento.

Para la ejecución de los ejemplos interactivos se utiliza el software **Problem Metaheuristic Solver (PMHS)** (Expósito-Izquierdo et al. 2015Expósito-Izquierdo et al.2015), el cual es una herramienta centrada en la creación y ejecución de algoritmos heurísticos sobre diferentes tipos de problemas de optimización dentro de un ámbito académico. Para cada algoritmo, permite su creación y configuración desde cero, aunque ya posee integrados aquellos de uso mas común, facilitando así su uso y configuración. Es ampliable con cualquier tipo de algoritmos y problemas, lo que la convierte en la herramienta adecuada para su integración en el repositorio presentado.

La interfaz principal de Problem Metaheuristic Solver se muestra en la Figura 3:

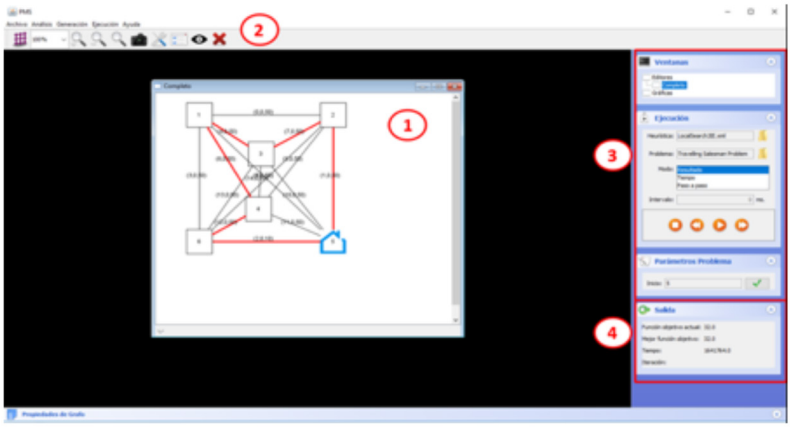

Figura 3: PMHS ejecutando una búsqueda local sobre el TSP.

En la interfaz se pueden observar las siguientes características de la herramienta:

1. **Instancia y solución del problema:** Panel que permite la edición y creación de la instancia del problema. En este caso, al tratarse del Travelling Salesman Problem, la instancia se trata de un grafo completo y la solución se muestra como una ruta marcada en rojo, la cual parte de un nodo seleccionado por el usuario y representado por una casa.
2. **Menú de la aplicación:** Menú que permite acceder a todas las acciones del software, que implican desde la carga y guardado de instancias, configuración y creación de algoritmos hasta diferentes opciones de análisis de las instancias cargadas.
3. **Menú de ejecución:** Menú con el que el usuario configura las diferentes ejecuciones de los ejemplos. Permite seleccionar el algoritmo heurístico ya configurado y el problema sobre el que ejecutarlo, el cual puede también ser configurado. En este menú se observan los diferentes modos de ejecución disponibles, además de una serie de botones a modo de reproductor multimedia que permiten la ejecución del ejemplo paso a paso. Como se puede ver en los botones, la ejecución puede ser pausada, finalizada e incluso avanzar o retroceder iteración a iteración. Los resultados de cada iteración se observan sobre la instancia del problema (punto 1) y en el panel de información (punto 4).
4. **Panel de información de salida:** Panel donde se pueden observar una serie de datos de la ejecución, como la iteración actual, el valor de la función objetivo y el tiempo de ejecución.

Al finalizar la ejecución de los ejemplos, la aplicación muestra datos estadísticos en forma de gráfica tal y como se observa en la Figura 4, así como genera un informe en formato PDF con la descripción de todos los pasos seguidos por el algoritmo para llegar a la solución final.

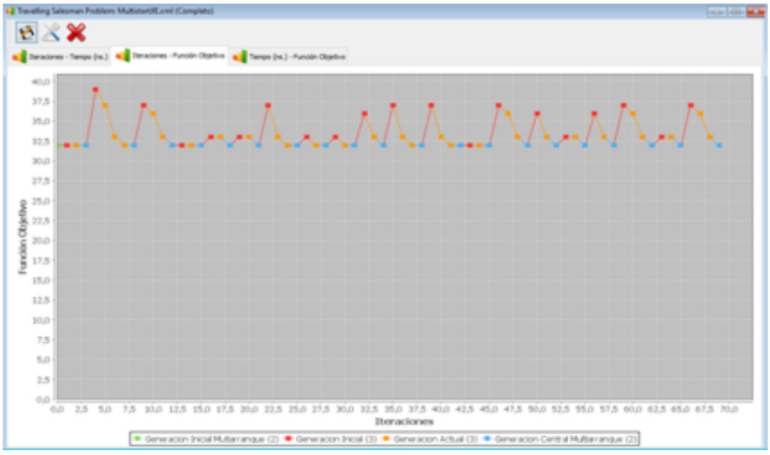

Figura 4: Gráfica con los resultados de ejecución de una búsqueda multiarranque sobre el TSP.

CONCLUSIONES Y LÍNEAS DE TRABAJO FUTURAS

Los nuevos paradigmas de educación buscan reforzar el aprendizaje a partir de la experimentación, con la que el alumno puede probar diferentes elementos de los conceptos aprendidos y sacar sus propias conclusiones. Con la mejora de las tecnologías y la llegada de Internet se ha facilitado este cambio de paradigma, creando lo que se conoce como educación 2.0, la cual se basa en ofrecer una mayor variedad de contenidos permanentemente accesibles, en permitir la experimentación mediante el uso de aplicaciones en la medida de lo posible y en fomentar el aprendizaje colaborativo.

En el presente artículo se describe el repositorio creado con el objetivo de reforzar el aprendizaje de la resolución de problemas de optimización haciendo uso de algoritmos heurísticos. Este repositorio ha sido creado por expertos en la materia y se divide en 3 secciones relacionadas entre sí: el repositorio de (meta)heurísticas, el repositorio de problemas de optimización y los ejemplos interactivos. Estos últimos permiten al alumno observar la ejecución paso a paso de un algoritmo heurístico determinado al resolver un problema de optimización, obteniendo estadísticas con el comportamiento general de dicha ejecución. Para ello se hace uso del software específico Problem Metaheuristic Solver.

Con el repositorio creado se consigue que el alumno comprenda los conceptos tanto de la optimización como de las (meta)heurísticas mediante herramientas propias de la educación 2.0. Se posee una información centralizada, actualizable y que fomenta el aprendizaje colaborativo así como el *feedback* entre alumno y profesor. Además, a través de los ejemplos interactivos, permite al alumno experimentar con diferentes ejecuciones y observar su comportamiento.

Para trabajos futuros se proponen las siguientes líneas de trabajo:

– Añadir más problemas de optimización y algoritmos heurísticos al repositorio. Ello implica no solo incluir sus correspondientes secciones, sino añadirlos a los ejemplos interactivos.
– Cambiar el repositorio del formato actual de blog y trasladarlo a una de las plataformas institucionales de la Universidad de La Laguna. Esto no solo permite tener el repositorio dentro del ecosistema de aplicaciones de la universidad, sino que también permite incluir características que en un blog no son posibles de realizar.
– Adaptar el software de ejecución de ejemplos interactivos para convertirlo en una herramienta online. Con ello se evita que el alumno deba descargar el software y se permite una actualización del mismo de forma directa.

REFERENCIAS BIBLIOGRÁFICAS

Aarts and Lenstra1997 Aarts, E. and Lenstra, J., editors (1997). *Local Search in Combinatorial Optimization*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition.

Balacheff et al.2008 Balacheff, N., Ludvigsen, M., de Jong, T., Lazonder, A., Barnes, S., and Montandon, L., editors (2008). *Technology Enhanced Learning: Principles and Products. Kaleidoscope Legacy Book*. Springer, Berlin.

Bortfeldt and Wäscher 2013 Bortfeldt, A. and Wäscher, G. (2013). Constraints in container loading - a state-of-the-art review. *European Journal of Operational Research*, 229(1):1 – 20.

Brucker and Knust 2012 Brucker, P. and Knust, S. (2012). Complex job-shop scheduling. In *Complex Scheduling*, GOR-Publications, pages 239–317. Springer Berlin Heidelberg.

Bruner1961 Bruner, J. (1961). The act of discovery. *Harvard Educational Review*, 31:21–32.

Castro-Sánchez et al.2009 Castro-Sánchez, J., del Castillo, E., Hortolano, J., and Rodríguez, A. (2009). Designing and using software tools for educational purposes: Flat, a case study. *IEEE Trans. Education*, 52(1):66–74.

Cavazzuti 2012 Cavazzuti, M. (2012). *Optimization Methods: From Theory to Scientific Design and Technological Aspects in Mechanics*. Springer.

Church and ReVelle 1974 Church, R. and ReVelle, C. (1974). The maximal covering location problem. *Papers of the Regional Science Association*, 32(1):101–118.

Cormen et al.2001 Cormen, T., Leiserson, C., Rivest, R., and Stein., C. (2001). *Introduction To Algorithms*, chapter 23: Minimum Spanning Trees, pages 561–579. McGraw-Hill, 2 edition.

Eiselt and Sandblom 2000 Eiselt, H. and Sandblom, C. (2000). Shortest path problems. In *Integer Programming and Network Models*, pages 283–313. Springer Berlin Heidelberg.

Eiselt and Sandblom 2010 Eiselt, H. and Sandblom, C. (2010). Introduction to operations research. In *Operations Research: A Model-Based Approach*, pages 1–12. Springer Berlin Heidelberg.

Expósito-Izquierdo et al.2015 Expósito-Izquierdo, C., López-Plata, I., and Moreno-Vega, J. (2015). Problem metaheuristic solver: An educational tool aimed at studying heuristic optimization methods. *Computer Applications in Engineering Education*, 23(6):897–909.

Feo and Resende 1995 Feo, T. and Resende, M. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6(2):109–133.

Gendreau and Potvin 2010 Gendreau, M. and Potvin, J. (2010). *Handbook of Metaheuristics*. Springer Publishing Company, Incorporated, 2nd edition.

Golden et al. 2008 Golden, B., Raghavan, S., and Wasil, E., editors (2008). *The vehicle routing problem : latest advances and new challenges*. Operations research/Computer science interfaces series, 43. Springer.

Guerrero et al. 2007Guerrero, J., Bataller, M., Soria, E., and Magdalena, R. (2007). Biolab: An educational tool for signal processing training in biomedical engineering. *Education, IEEE Transactions on*, 50(1):34–40.

Hansen et al.2010Hansen, P., Mladenovic, N., and Moreno Pérez, J. A. (2010). Variable neighbourhood search: methods and applications. *Annals of Operations Research*, 175(1):367–407.

Karp 1972 Karp, R. (1972). Reducibility among combinatorial problems. *Complexity of Computer Computations*, pages 85–103.

Kirkpatrick et al.1983 Kirkpatrick, S., Gelatt, C., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220:671–680.

Lawler1963 Lawler, E. (1963). The quadratic assignment problem. *Management Science*, 9(4):586–599.

Liao and Sun 2001Liao, Y. and Sun, C. (2001). An educational genetic algorithms learning tool. *IEEE Transactions in Education*, 44:20–28.

Martí et al.2018 Martí, R., Pardalos, P., and Resende, M. (2018). *Handbook of Heuristics*. Springer.

Martí et al.2013 Martí, R., Resende, M., and Ribeiro, C. (2013). Multi-start methods for combinatorial optimization. *European Journal of Operational Research*, 226(1):1–8.

Mateo-Sanguino et al. 2013 Mateo-Sanguino, T., Serrano-López, C., and Márquez-Hernández, F. (2013). Wifisim: An educational tool for the study and design of wireless networks. *Education, IEEE Transactions on*, 56(2):149–155.

Melián-Batista et al.2003 Melián-Batista, B., Moreno-Pérez, J. A., and Moreno-Vega, J. M. (2003). Metaheuristics: A global view. *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial*, 19:7–28.

Miller1988 Miller, G. (1988). *The Meaning of General Education: The Emergence of a Curriculum Paradigm*. Teachers College Press.

Ozturk 2011 Ozturk, O. (2011). Multicore education through simulation. *Education, IEEE Transactions on*, 54(2):203–209.

Rastrigin 1986 Rastrigin, L. (1986). Random search as a method for optimization and adaptation. In Arkin, V., Shiraev, A., and Wets, R., editors, *Stochastic Optimization*, volume 81 of *Lecture Notes in Control and Information Sciences*, pages 534–544. Springer Berlin Heidelberg.

Steiglitz and Papadimitriou1982Steiglitz, K. and Papadimitriou, C. H. (1982). Combinatorial optimization: Algorithms and complexity. *Prentice Hall, New Jersey., UV Vazirani (1984). On two geometric problems related to the travelling salesman problem. J. Algorithms*, 5:231–246.

Talbi 2009 Talbi, E. (2009). *Metaheuristics: From Design to Implementation*. Wiley Publishing.

Tomozii and Topala 2014 Tomozii, S. and Topala, I. (2014). Why do we need to change the educational paradigms? *Procedia - Social and Behavioral Sciences*, 142:586–591.

Voudouris and Tsang 2003 Voudouris, C. and Tsang, E. (2003). *Guided Local Search*, volume 57 of *International Series in Operations Research & Management Science*, chapter Handbook of Metaheuristics, pages 185–218. Springer.

Walks 2013 Walks, L. (2013). *Education 2.0: The Learningweb Revolution and the Transformation of the School*. Paradigm Publishers.

Willingham et al.2015 Willingham, D., Hughes, E., and Dobolyi, D. (2015). The scientific status of learning styles theories. *Teaching of Psychology*, 42(3):266–271.

Zelinka et al.2013 Zelinka, I., Snášel, V., and A., A., editors (2013). *Handbook of Optimization: From Classical to Modern Approach*, volume 38 of *Intelligent Systems Reference Library*. Springer, Berlin.