VNIVERSITAT DE VALÈNCIA Escola Tècnica Superior d'Enginyeria

VNIVERSITAT DE VALÈNCIA

Análisis y Comparación de GRASP y GRASP con Path Relinking en la Resolución del Maximum Diversity Problem

Proyecto asignatura Logística basada en Datos

Universidad de Valéncia

Autoras: Lidia Moreno Marín y Amparo Gálvez Vilar

Correos: momali@alumni.uv.es y galviam@alumni.uv.es

Profesor: Rafael Martí

Curso académico: 2024–2025

Fecha: 16-05-2025

ÍNDICE

INTRODUCCIÓN

La resolución de problemas de optimización combinatoria es una tarea central en numerosos contextos reales como la planificación logística, el diseño de redes o la diversificación de carteras. Este tipo de problemas, definidos sobre conjuntos discretos y generalmente asociados a estructuras como grafos o matrices de decisiones, se caracterizan por su alta complejidad computacional, especialmente cuando crecen en tamaño o en restricciones. Tal como argumentan Resende y Ribeiro (2016), en estos casos los algoritmos exactos dejan de ser prácticos, lo que justifica la adopción de técnicas aproximadas como las metaheurísticas, que ofrecen soluciones de alta calidad sin garantizar optimalidad, pero con una eficiencia notable en la práctica.

Entre estas técnicas, destaca GRASP (Greedy Randomized Adaptive Search Procedure), una metaheurística multiarranque basada en la combinación de una construcción aleatorizada y una búsqueda local intensiva. En cada iteración, GRASP genera una solución inicial mediante una Lista Restringida de Candidatos (RCL) y luego aplica una mejora sucesiva mediante movimientos en su vecindad, hasta alcanzar un óptimo local. GRASP es fácil de implementar y permite explorar distintas soluciones, por lo que se ha convertido en un método útil y eficaz en muchos tipos de problemas.

No obstante, al tratarse de un enfoque sin memoria, GRASP puede quedarse atrapado en soluciones subóptimas, especialmente cuando la búsqueda local converge rápidamente a óptimos locales poco representativos del espacio global. Para superar esta limitación, se propone su hibridación con Path Relinking (PR), una estrategia de intensificación que incorpora un mecanismo de memoria y aprovecha la información de soluciones previamente obtenidas. El objetivo principal de PR es generar nuevas soluciones explorando caminos que conectan una solución inicial (generalmente la obtenida tras la búsqueda local) con una solución guía de alta calidad, que suele pertenecer a un conjunto élite formado por las mejores soluciones encontradas hasta el momento.

Durante este proceso, se generan soluciones intermedias a lo largo de una trayectoria en el espacio de soluciones, aplicando cambios incrementales que transforman la solución inicial en la solución guía. Cada paso de esa transformación consiste en modificar parcialmente la solución actual, acercándola a la guía, y evaluando su calidad. De esta forma, se pueden descubrir soluciones no exploradas que combinan características de ambas, y que en muchos casos superan a las soluciones originales. Esta técnica no solo intensifica la búsqueda en regiones prometedoras, sino que también introduce una forma controlada de diversificación si se combinan soluciones suficientemente distintas.

Tal como se detalla en el capítulo 9 del libro de Resende y Ribeiro (2016), la integración de PR en GRASP permite aprovechar el conjunto élite como fuente de información estratégica, mejorando la calidad de los resultados sin que ello implique un aumento significativo del coste computacional. Además, su implementación es flexible y se puede adaptar con distintos criterios de selección de pares de soluciones, longitud mínima de trayectorias, o variantes como el relinking bidireccional o truncado (Resende & Ribeiro, 2016, cap. 9).

En este trabajo se aborda la resolución del Maximum Diversity Problem (MDP), un problema clásico de optimización combinatoria en el que se pretende seleccionar un subconjunto de m elementos a partir de un conjunto de n candidatos, de forma que se maximice la diversidad total, medida como la suma de las distancias por pares entre los elementos seleccionados. Este tipo de problema tiene aplicaciones en múltiples contextos donde se requiere diversidad, como el diseño de equipos multidisciplinares, la selección de carteras de inversión variadas, la planificación de productos diferenciados o la conservación genética.

METODOLOGÍA

En esta sección se describe el enfoque seguido para implementar y evaluar los algoritmos utilizados en la resolución del Maximum Diversity Problem (MDP). En primer lugar, se detallan las características de las instancias empleadas en el estudio. A continuación, se explica el funcionamiento del algoritmo GRASP puro y se justifica la elección del parámetro α que controla el grado de aleatoriedad en la fase constructiva. También se describe el procedimiento de búsqueda local aplicado sobre las soluciones generadas. Finalmente, se presenta la extensión del algoritmo mediante Path Relinking, justificando su incorporación y explicando el criterio seguido para seleccionar las soluciones guía y el funcionamiento general del procedimiento híbrido.

Conjunto de Datos

Para llevar a cabo la evaluación experimental de los algoritmos, se ha utilizado un conjunto de 15 instancias del Maximum Diversity Problem (MDP) proporcionadas a través del aula virtual de la asignatura. Cada instancia se define mediante una matriz de distancias simétrica, en la que cada entrada d_ij representa la distancia entre los elementos i y j. El objetivo consiste en seleccionar un subconjunto de m elementos a partir de un conjunto total de n, de forma que la suma de las distancias por pares entre los elementos seleccionados sea máxima.

Las instancias están divididas en dos grupos, con diferentes tamaños y niveles de dificultad:

- 9 instancias con tamaño reducido: $n = 100, m = 10$.
- 6 instancias de mayor tamaño: $n = 500, m = 25$.

Este conjunto mixto permite evaluar tanto la calidad de las soluciones como la escalabilidad de los algoritmos en función del tamaño del problema. Las matrices utilizadas contienen únicamente valores reales no negativos y cumplen la propiedad de simetría ($d\_ij = d\_ji$). Las instancias han sido utilizadas en todas las fases del estudio: la calibración de parámetros y la posterior comparación entre GRASP puro y GRASP con Path Relinking, asegurando en todos los casos que ambos métodos se ejecutan bajo un mismo límite de tiempo.

GRASP puro – Fase 0 del proyecto

La Fase 0 del proyecto consistió en la ejecución y análisis de una primera versión funcional del algoritmo GRASP puro (Greedy Randomized Adaptive Search Procedure) para resolver instancias del Maximum Diversity Problem (MDP). El código fue proporcionado como base por el profesor, en esta fase inicial el objetivo era comprender su funcionamiento, probar su correcta ejecución y empezar a observar resultados preliminares sobre distintas instancias del problema.

El algoritmo sigue la estructura clásica de GRASP: una fase de construcción aleatorizada, seguida de una búsqueda local que intenta mejorar la solución generada. Todo el proceso se repite varias veces (iteraciones), y se conserva la mejor solución obtenida. El script principal de ejecución es main.py, que carga una instancia y lanza el procedimiento definido en grasp.py.

Fase de construcción (cgrasp.py)

En esta fase, se genera una solución inicial factible seleccionando m elementos de un conjunto total de n, con el objetivo de maximizar la diversidad (suma de distancias entre los elementos seleccionados). El archivo cgrasp.py implementa una heurística semi-codiciosa, es decir, un procedimiento que no elige siempre el candidato con la mejor contribución posible, sino que introduce cierta aleatoriedad controlada para diversificar las soluciones generadas. Esta utiliza:

- Una Lista de Candidatos (CL): elementos aún no seleccionados.
- Una Lista Restringida de Candidatos (RCL): subconjunto de CL formado por los candidatos con mejor puntuación según una regla controlada por el parámetro α.

Para cada candidato se calcula su contribución marginal a la diversidad. Luego se identifican los valores máximo y mínimo (g_{max} y g_{min}) y se calcula un umbral dinámico:

$$threshold = g_{max} - \alpha \cdot (g_{max} - g_{min})$$

Todos los candidatos cuya contribución es igual o superior a este umbral se incluyen en la RCL, y de ahí se selecciona uno de forma aleatoria. Esto permite controlar el equilibrio entre determinismo y aleatoriedad mediante el valor de α, que se pasa como parámetro al algoritmo.

Fase de búsqueda local (lsbestimp.py)

Una vez generada la solución, se aplica una búsqueda local por mejora de intercambio. En concreto, se identifica el peor elemento de la solución (el que aporta menos diversidad) y se evalúan posibles sustituciones por elementos no incluidos. Si el intercambio mejora la solución, se realiza. Este proceso se repite hasta alcanzar un óptimo local, es decir, una solución que ya no puede mejorar con un solo intercambio.

Esta parte está implementada en el archivo lsbestimp.py, a través de las funciones improve() y tryImprove(), que se apoyan en las utilidades del módulo solution.py para calcular distancias y modificar soluciones.

Control general del proceso (grasp.py)

El archivo grasp.py coordina las iteraciones del algoritmo. En cada una de ellas:

- Se construye una solución con un valor dado de α.
- Se mejora mediante la búsqueda local.
- Se actualiza la mejor solución global si se obtiene una de mayor valor objetivo.

Cada iteración imprime por pantalla el valor de la solución antes y después de la búsqueda local, lo que permite evaluar rápidamente su efecto.

Primera ejecución (main.py)

En main.py se definió una ejecución sencilla para probar el comportamiento del algoritmo sobre una instancia concreta (en nuestro caso ejecutamos "MDG-a_1_100_m10.txt"). Esta fase sirvió para verificar que:

- Las instancias se leían correctamente.
- El algoritmo generaba soluciones viables.
- La búsqueda local realizaba intercambios válidos y mejoraba la solución en muchos casos.
- El código funcionaba correctamente de principio a fin.

Esta versión inicial no incluía calibración de parámetros ni técnicas avanzadas, pero se utilizó para familiarizarse con el funcionamiento interno del código proporcionado y sentar las bases para las siguientes fases del proyecto.

Elección del parámetro α – Fase 1 del proyecto

Una vez comprobado el correcto funcionamiento del algoritmo en la Fase 0, la Fase 1 del proyecto se centró en la calibración del parámetro α, fundamental en la fase de construcción de GRASP para controlar el equilibrio entre determinismo y aleatoriedad. Mientras que en la fase inicial se había utilizado un único valor de α (o se dejaba aleatorio con α = −1), en esta nueva etapa se implementó un bucle que permitiera evaluar el comportamiento del algoritmo para varios valores fijos de α sobre todas las instancias disponibles.

Para ello, se modificó el archivo main.py, reemplazando la ejecución de una sola instancia por un procedimiento automatizado que recorriera todos los archivos .txt de la carpeta instances. Por cada instancia, se ejecutó el algoritmo GRASP con distintos valores de α, concretamente:

$$\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$$

Inicialmente, se realizaron 5 ejecuciones por cada valor de alpha. Sin embargo, se observó que los tiempos obtenidos con α=0,1 eran algo más variables entre ejecuciones. Para tener una mejor referencia de su comportamiento, se decidió incrementar a 10 el número de ejecuciones solo para este valor, manteniéndose 5 para los demás.

Para cada combinación instancia – alpha se registraron dos métricas principales:

- El mejor valor obtenido entre todas las ejecuciones realizadas.
- El tiempo total de ejecución en completar dichas ejecuciones.

Los resultados se exportaron automáticamente a un archivo CSV llamado resultados_alpha_tiempo.csv. Este archivo incluye una fila por instancia, y para cada valor de α, se recogen dos columnas: una con el mejor valor objetivo alcanzado entre varias ejecuciones, y otra con el tiempo total de ejecución necesario para completarlas. Además, la tabla contiene una última columna que recoge el mejor valor global obtenido en esa instancia, sin importar el valor de α con el que se alcanzó.

Para evaluar de forma objetiva qué configuración de α ofrecía un comportamiento más robusto, se calculó la desviación absoluta entre el valor obtenido por cada α y el mejor valor global por

instancia. A continuación, se calculó la media de estas desviaciones para cada valor de α, permitiendo así comparar su rendimiento promedio a lo largo de todas las instancias.

Los resultados de este análisis se resumen en la siguiente tabla:

α	0.1	0.3	0.5	0.7	0.9
Promedio	0,43%	1,36%	0,64%	0,92%	1,13%

Tabla 1. Desviación porcentual media de cada valor de α respecto al mejor valor por instancia

Como se puede observar, la configuración con α=0.1 fue la que obtuvo una desviación media más baja, con un valor del 0.43%, superando al resto de configuraciones. Este resultado indica que α=0.1 proporciona soluciones más cercanas al óptimo en la mayoría de los casos, mostrando un comportamiento más estable y eficaz. En consecuencia, se eligió α=0.1 como configuración definitiva para las siguientes fases del proyecto.

GRASP con Path Relinking – Fase 2 del proyecto

Selección de soluciones

La primera etapa de la Fase 2 del proyecto consistió en generar un conjunto representativo de soluciones de partida utilizando el algoritmo GRASP puro. Para cada instancia del problema, se ejecutó GRASP 50 veces con el valor de α=0.1, seleccionado en la fase de calibración anterior. El objetivo era construir una muestra amplia de soluciones viables sobre la que aplicar posteriormente el procedimiento de Path Relinking.

Una vez generadas 50 soluciones GRASP para cada instancia del MDP, se procedió a seleccionar cuatro configuraciones específicas para aplicar el procedimiento de Path Relinking (PR). Esta selección se realizó con el objetivo de construir trayectorias entre soluciones de distinta calidad y explorar regiones prometedoras del espacio de soluciones.

Para cada instancia:

1. Se identificó la solución con el mayor valor objetivo entre las 50 obtenidas, es decir, a mejor solución. Esta solución se tomó como la solución inicial (s_0) para el Path Relinking.
2. Se calcularon las desviaciones absolutas de cada una de las 50 soluciones respecto al mejor valor hallado.
3. Se seleccionaron aquellas tres soluciones con mayor desviación, es decir, las más alejadas del óptimo. Estas tres configuraciones, etiquetadas como s_1, s_2 y s_3, actuaron como soluciones guía (guiding solutions) de las trayectorias de Path Relinking, lo cual puede conducir a la mejora de los resultados finales.

Estructura de la tabla de soluciones seleccionadas

La información asociada a estas soluciones se registró en una tabla que recoge, para cada instancia:

- El valor objetivo de la mejor solución.
- El conjunto de nodos que componen dicha solución.
- El valor y nodos de las tres soluciones guía seleccionadas por su desviación.

A modo de ejemplo, la siguiente tabla muestra el caso de la instancia MDG-a_10_100_m10.txt:

Instancia	Mejor valor	Nodos	Peor valor 1	Nodos	Peor valor 1	Nodos	Peor valor 1	Nodos
MDG-a_10_100_m10,txt	351,79	10 51 60 63 68 69 74 87 88 90	308,64	0 10 14 27 49 51 71 73 79 97	315,62	10 18 28 30 40 45 49 64 77 98	317,88	6 9 28 44 52 57 62 67 94 96

Tabla 2. Ejemplo de selección de la mejor solución y las tres más alejadas (por desviación absoluta)

Aplicación de Path Relinking

Una vez seleccionadas las cuatro soluciones base por instancia (la mejor solución GRASP y las tres con mayor desviación), se aplicó el procedimiento de Path Relinking (PR) para intensificar la búsqueda en torno a la solución de partida. El objetivo de esta fase era aprovechar trayectorias que conectan soluciones ya conocidas para generar nuevas configuraciones potencialmente mejores.

Funcionamiento del procedimiento

El Path Relinking se ejecuta desde la solución inicial s_0, la mejor encontrada por GRASP, hacia cada una de las soluciones guía (s_1, s_2, s_3), que representan configuraciones de menor calidad, pero estructuralmente distintas, es decir, soluciones que comparten pocos nodos en común con la solución inicial y, por tanto, permiten generar trayectorias más diversas. El propósito de esta elección es recorrer trayectorias que introduzcan cambios significativos en la solución, lo que permite explorar nuevas regiones del espacio de soluciones.

En cada relinking, se sigue el siguiente proceso:

1. Se identifican los nodos que diferencian s_0 y la solución guía. Esto genera un conjunto de posibles intercambios: nodos que deben entrar y salir de la solución actual para convertirla progresivamente en la guía.
2. En cada paso, se evalúan todos los intercambios posibles entre nodos del conjunto actual y nodos de la guía, manteniendo constante el tamaño m de la solución.
3. De entre todos los movimientos válidos, se selecciona aquel que mayor mejora produce en la función objetivo (suma de distancias).
4. La solución resultante pasa a ser la nueva solución actual, y el proceso se repite hasta:
    - alcanzar exactamente la solución guía, o
    - completar un máximo de 20 pasos (para controlar la duración de cada trayectoria).
5. Durante el recorrido, se guarda siempre la mejor solución encontrada, aunque no coincida con el extremo final de la trayectoria.

Este procedimiento se repite hasta tres veces por instancia (una por cada solución guía), siempre que no se haya agotado el tiempo máximo disponible, calculado previamente para garantizar una comparación justa con GRASP puro.

Control del tiempo y resultado final

El procedimiento de Path Relinking se ejecutó bajo un tiempo máximo específico para cada instancia, calculado como el producto del tiempo medio por iteración de GRASP (obtenido a partir de ejecuciones previas) multiplicado por 50, más un margen adicional fijo de 60 segundos. Esta fórmula se aplicó a cada instancia para asegurar que PR dispusiera de un presupuesto temporal comparable al que tendría GRASP al ejecutar el mismo número de iteraciones.

Esta decisión se tomó por dos razones. Por un lado, Path Relinking supone una mayor carga computacional que GRASP puro, ya que evalúa múltiples intercambios en cada paso. Por otro, como el objetivo era comparar ambos métodos en igualdad de condiciones, se estableció un límite de tiempo equivalente. Así se evita que una mejora en los resultados de PR se deba simplemente a disponer de más tiempo.

Durante la ejecución, si el tiempo asignado a una instancia se agotaba antes de completar las tres trayectorias de relinking, el procedimiento se detenía inmediatamente y se continuaba con las soluciones generadas hasta ese momento. Una vez ejecutados todos los Path Relinking posibles dentro del límite, se recogían las siguientes soluciones:

- La solución inicial (s_0).
- Hasta tres soluciones intermedias obtenidas desde PR hacia las guías (s_1, s_2, s_3).

La solución final se seleccionó como aquella con el mayor valor objetivo entre todas las anteriores. Esta solución representa el resultado definitivo del procedimiento GRASP con Path Relinking para esa instancia. Los datos se almacenaron en el archivo resultados_grasp_con_pr.csv, incluyendo el nombre de la instancia, el valor objetivo de la solución final, el conjunto de nodos seleccionado y el tiempo total de ejecución.

RESULTADOS

Resultados obtenidos con GRASP puro

La primera serie de experimentos consistió en aplicar el algoritmo GRASP puro sobre las 15 instancias del problema, ejecutando 50 iteraciones por instancia y utilizando α=0.1 como valor fijo, previamente calibrado. Para cada instancia se ejecutaron **50 iteraciones**, registrando tres métricas clave:

- El mejor valor objetivo alcanzado,
- El tiempo total de ejecución, y
- El tiempo medio por iteración.

A continuación, se muestra un resumen de los resultados obtenidos:

Instancia	Mejor valor	Tiempo total (s)	Tiempo medio por iteración (s)
MDG-a_10_100_m10.txt	354.96	0.10	0.0020
MDG-a_12_100_m10.txt	354.25	0.09	0.0018
MDG-a_13_n500_m50.txt	7732.14	3.69	0.0738
MDG-a_14_100_m10.txt	356.06	0.13	0.0026
MDG-a_16_n500_m50.txt	7734.17	3.88	0.0776

Tabla 3. Resultados de GRASP puro (selección de instancias)

Los resultados muestran que GRASP puro es eficiente en tiempo y ofrece soluciones de calidad, especialmente en instancias pequeñas. En las instancias más grandes, aunque el tiempo de ejecución aumenta, se mantiene dentro de márgenes razonables para un algoritmo heurístico.

La tabla completa con los resultados para las 15 instancias se encuentra disponible en el archivo adjunto Resultados_GRASP_puro.csv.

Resultados obtenidos con GRASP + Path Relinking

Tras ejecutar el procedimiento de Path Relinking sobre las soluciones seleccionadas de cada instancia, se obtuvieron los resultados finales correspondientes a la versión híbrida del algoritmo. En todos los casos, se respetó un límite de tiempo específico por instancia, calculado previamente en función del tiempo medio de ejecución de GRASP puro.

Para cada instancia se registraron los siguientes datos:

- El valor objetivo final, correspondiente a la mejor solución encontrada entre la solución inicial y las generadas durante el relinking.
- El tiempo total de ejecución empleado dentro del límite asignado.
- El conjunto de nodos que conforma la solución final.

Una muestra representativa de los resultados puede verse en la siguiente tabla:

Instancia	Valor final	Tiempo (s)
MDG-a_10_100_m10.txt	351.90	0.0038
MDG-a_12_100_m10.txt	354.25	0.0050
MDG-a_13_n500_m50.txt	7644.90	10.9620
MDG-a_14_100_m10.txt	349.15	0.0080
MDG-a_16_n500_m50.txt	7647.02	1.1200

Tabla 4. Resultados de GRASP con Path Relinking (selección de instancias)

En general, los valores alcanzados por GRASP + PR se situaron en un rango alto dentro de cada instancia, lo cual confirma la capacidad del procedimiento de intensificar la búsqueda en trayectorias prometedoras. Además, el tiempo de ejecución registrado se mantuvo dentro de

los márgenes definidos, lo que demuestra que la implementación respetó adecuadamente las restricciones temporales fijadas para asegurar la equidad experimental.

La totalidad de resultados obtenidos para las 15 instancias se encuentra recogida en el archivo adjunto resultados_grasp_con_pr.csv.

Comparativa entre GRASP puro y GRASP con Path Relinking

Una vez ejecutamos ambos métodos sobre las 15 instancias del problema, se procedió a comparar los resultados en tres dimensiones principales:

- Calidad de la solución (desviación respecto al mejor valor observado).
- Número de veces que obtuvo el mejor resultado.
- Tiempo medio de ejecución por instancia.

Para facilitar el análisis, las instancias se agruparon en función de su tamaño: n = 100 (problemas pequeños) y n = 500 (problemas grandes). Los resultados se resumen en la siguiente tabla:

		GRASP PURO	GRASP + PR
n = 100			
	Dev.	1,41%	2,28%
	Best.	6	1
	Time	0,117	0,007
n= 500			
	Dev.	0,5%	0,29%
	Best.	6	0
	Time	3,97	6,28

Tabla 5. Comparativa general entre GRASP puro con GRASP más Path Relinking

Los resultados muestran un comportamiento diferenciado según el tamaño del problema:

- En las instancias pequeñas (n=100), GRASP puro obtuvo mejores resultados en la mayoría de los casos, con menor desviación medio y mejor tiempo medio. Esto se debe en parte a que la estructura del espacio de soluciones es menos compleja, por lo que el comportamiento aleatorio del Path Relinking no ofrece ventaja suficiente y, en ocasiones, introduce ruido.
- En las instancias grandes (n=500), GRASP con Path Relinking mejora ligeramente la calidad media de las soluciones (menor desviación), aunque no logra superar a GRASP puro en ninguna instancia individual. Sin embargo, su tiempo medio de ejecución es más elevado debido al coste computacional adicional de Path Relinking.

Estos resultados reflejan que, aunque Path Relinking puede mejorar ligeramente la robustez de la solución en problemas grandes, su uso no garantiza una mejora sistemática frente a GRASP puro, especialmente si se evalúan soluciones bajo una misma restricción temporal.

CONCLUSIONES

Este trabajo ha abordado la resolución del Maximum Diversity Problem (MDP) mediante el uso de dos metaheurísticas: GRASP puro y una variante extendida mediante Path Relinking (PR). A lo largo del estudio se ha implementado, calibrado y evaluado ambos métodos sobre un conjunto representativo de instancias, con tamaños n= 100 y n= 500, bajo condiciones de tiempo controladas para garantizar una comparación justa.

Los principales resultados obtenidos permiten extraer las siguientes conclusiones:

- GRASP puro ha demostrado ser una técnica eficiente, especialmente en instancias pequeñas, donde alcanza rápidamente soluciones de alta calidad con un bajo coste computacional. Su simplicidad y rapidez lo convierten en una buena opción base para problemas de tamaño reducido.
- La incorporación de Path Relinking permite explorar trayectorias entre soluciones distantes en el espacio de búsqueda, con el objetivo de intensificar la exploración. Aunque este enfoque no siempre mejora la calidad final de forma directa, sí ha mostrado una menor desviación media en instancias grandes, lo que sugiere una mayor robustez en entornos más complejos.
- En términos generales, no se ha observado una mejora sistemática al incorporar PR, y el coste computacional adicional que implica puede limitar su utilidad en problemas de tamaño medio o cuando se dispone de poco tiempo de ejecución. Sin embargo, su potencial se hace más evidente en problemas de mayor escala, donde GRASP por sí solo podría resultar insuficiente para escapar de óptimos locales.

En conjunto, el trabajo ha permitido explorar las fortalezas y limitaciones de GRASP y PR aplicados al MDP, y plantea como líneas futuras de mejora la posibilidad de incorporar mecanismos adaptativos, relinking probabilístico, o estrategias híbridas con otras metaheurísticas para seguir mejorando la calidad y eficiencia de las soluciones.

BIBLIOGRAFÍA

Material docente de la asignatura de Logística basada en Datos Gr.A-T (36444), Universidad de Valencia (2024–2025).
Presentaciones teóricas sobre GRASP, Path Relinking y Metaheurísticas. Departamento de Estadística e Investigación Operativa.

Resende, M. G. C., & Ribeiro, C. C. (2016). *Optimization by GRASP: Greedy Randomized Adaptive Search Procedures* (1st ed.). Springer New York.

https://doi.org/10.1007/978-1-4939-6530-4