Procedimientos de búsqueda adaptativa aleatorios codiciosos: Adelantos y Prórrogas

Mauricio GC Resende y Celso C. Ribeiro

Resumen – Un procedimiento de búsqueda adaptativa aleatoria voraz (GRASP) es una metaheurística de inicio múltiple para problemas de optimización combinatoria, en la que cada iteración consta básicamente de dos fases: construcción y búsqueda local. La fase de construcción construye una solución factible cuya vecindad se investiga hasta que se encuentra un mínimo local durante la fase de búsqueda local. La mejor solución global se mantiene como resultado. En este capítulo, primero describimos los componentes básicos de GRASP. Las técnicas de implementación exitosas se discuten e ilustran mediante resultados numéricos obtenidos para diferentes aplicaciones. También se describen soluciones mejoradas o alternativas con mecanismos de construcción y técnicas para acelerar la búsqueda: esquemas alternativos de construcción codiciosos aleatorios, GRASP reactivo, costo por turbaciones, funciones de sesgo, memoria y aprendizaje, heurística constructiva de Lagrange y GRASP de Lagrange, búsqueda local en estructuras parcialmente construidas. soluciones, hashing y filtrado. También discutimos las estrategias de implementación de la intensificación basada en la memoria y las técnicas posteriores a la optimización utilizando la vinculación de rutas. También se revisan estrategias de reinicio para acelerar la búsqueda, hibridaciones con otras metaheurísticas y aplicaciones.

1. Introducción

Consideramos en este capítulo un problema de optimización combinatoria, definido por un conjunto base finito E = {1,...,n}, un conjunto de soluciones factibles F ÿ 2^E y una función objetivo f : 2^E ÿ R. En su versión de minimización , buscamos una solución óptima $S^ÿ$ ÿ F tal que f($S^ÿ$) ÿ f(S), ÿS ÿ F. El conjunto base E, la función de costo f , y

---

[illegible] Niteroi, RJ Brasil. [illegible] Seattle, WA [illegible] celso@ic.uff.br [illegible] @uw.edu

el conjunto de soluciones factibles F se definen para cada problema específico. Por ejemplo, en el caso del problema del viajante de comercio, el conjunto base E es el de todas las aristas conectando las ciudades a visitar, f(S) es la suma de los costos de todas las aristas en S, y F está formado por todos los subconjuntos de aristas que determinan el ciclo hamiltoniano.

GRASP (Greedy Randomized Adaptive Search Procedure) [90, 91] es una metaheurística iterativa o de inicio múltiple, en la que cada iteración consta de dos fases: construcción y búsqueda local. La fase de construcción construye una solución usando un codicioso algoritmo adaptativo aleatorizado. Si esta solución no es factible, entonces es necesario aplicar un procedimiento de reparación para lograr o hacer un nuevo intento de construir la viabilidad la solución factible. Una vez que se obtiene una solución factible, se invierte su vecindad hasta que se encuentra un mínimo local durante la fase de búsqueda local. El mejor en general la solución se mantiene como el resultado.

Amplias encuestas bibliográficas sobre procedimientos de búsqueda adaptativa aleatorios codiciosos se presentan en [100, 101, 102, 217, 218, 224]. El primer libro sobre GRASP fue publicado en 2016 por Resende y Ribeiro [220].

El pseudocódigo de la Figura 1 ilustra los bloques principales de un procedimiento GRASP para la minimización, en el que se realizan las iteraciones Max Iterations y Seed se utiliza como semilla inicial para el generador de números pseudoaleatorios.

```
procedimiento GRASP (Iteraciones máximas, Semilla)
1 Entrada de lectura ();
dos     para k = 1,...,Máximo de iteraciones
            Solución ÿ Construcción aleatoria codiciosa (semilla);
3           si la solución no es factible entonces
4               Solución ÿ Reparación(Solución);
5           fin;
6           Solución ÿ Búsqueda local (Solución);
            Solución de actualización (Solución, Mejor solución);
7 8 9 fin;
10 devuelven la mejor solución;
terminar AGARRE.
```

Higo. 1 Pseudocódigo de la metaheurística GRASP.

La figura 2 ilustra la fase de construcción con su pseudocódigo. En cada iteración de esta fase, permita que el conjunto de elementos candidatos esté formado por todos los elementos de el conjunto de suelo Y que se puede incorporar a la solución parcial que se está construyendo, con posibilidad de evitar la construcción de una solución parcial con el conjunto de suelo parcial restante elementos. La selección del siguiente elemento a incorporar está determinada por el evaluación de todos los elementos candidatos según una función de evaluación codiciosa. Este La función codiciosa generalmente representa el aumento incremental en la función de costo debido a la incorporación de este elemento a la solución en construcción. La evaluación de los elementos por esta función conduce a la creación de un candidato restringido lista (RCL) formada por los mejores elementos, es decir, aquellos cuya incorporación a la actual solución parcial da como resultado los costos incrementales más pequeños (este es el aspecto codicioso de el algoritmo). El elemento a incorporar en la solución parcial es aleatoriamente

seleccionados de aquellos en el RCL (este es el aspecto probabilístico de la heurística).
Una vez incorporado el elemento seleccionado a la solución parcial, la lista de candidatos
se actualiza y se reevalúan los costos incrementales (este es el aspecto adaptativo de
la heurística). Los pasos anteriores se repiten mientras exista al menos un elemento de fecha candidato.
Esta estrategia es similar a la heurística semi-codiciosa propuesta por Hart
y Shogan [121], que también es un enfoque de inicio múltiple basado en codiciosos aleatorios
construcciones, pero sin búsqueda local.

```
procedimiento Construcción aleatorizada codiciosa (semilla)
1 Solución ÿ /0;
dos     Inicializar el conjunto de elementos candidatos;
3       Evaluar los costos incrementales de los elementos candidatos;
4 mientras exista al menos un elemento candidato do
            Construir la lista de candidatos restringidos (RCL);
5 6         Seleccione un elemento s del RCL al azar;
7           Solución ÿ Soluciónÿ {s};
8           Actualizar el conjunto de elementos candidatos;
            Reevaluar los costos incrementales;
9 10 fin;
11 solución de retorno ;
Fin de la construcción aleatoria codiciosa.
```

Higo. 2 Pseudo-código de la fase de construcción.

Un procedimiento de construcción codicioso aleatorio no siempre es capaz de producir un
solución factible. Puede ser necesario aplicar un procedimiento de reparación a la solución para
lograr la viabilidad. Se pueden encontrar ejemplos de procedimientos de reparación en [81, 83, 169, 180].

Las soluciones generadas por una construcción aleatoria codiciosa no son necesariamente
óptimo, incluso con respecto a los vecindarios simples. La fase de búsqueda local suele mejorar la
solución construida. Un algoritmo de búsqueda local funciona de manera iteractiva reemplazando
sucesivamente la solución actual por una mejor solución en
su barrio. Termina cuando no se encuentra una solución mejor en el vecindario. El pseudocódigo de un
algoritmo básico de búsqueda local a partir de la solución.
Solución construida en la primera fase (y posiblemente factible por la reparación
heurística) y usando una vecindad N se da en la Figura 3.

```
procedimiento Búsqueda local (Solución)
        mientras que la solución no es localmente óptima hacer
1           encontrar s' ÿ N(Solución) con f(s') < f(Solución);
            Solución ÿ s';
2 3 4 fin;
5 solución de retorno ;
finalizar la búsqueda local.
```

Higo. 3 Pseudocódigo de la fase de búsqueda local.

La velocidad y eficacia de un procedimiento de búsqueda local depende de varios aspectos, como la estructura del vecindario, la técnica de búsqueda del vecindario, la estrategia utilizada para la evaluación del valor de la función de costo en los vecinos y la solución de partida en sí. La fase de construcción juega un papel muy importante con respecto a este último aspecto, construyendo soluciones de partida de alta calidad para la búsqueda local.
Por lo general, se utilizan vecindarios simples. La búsqueda de vecindad se puede implementar utilizando una estrategia de mejora óptima o de mejora inicial. En el caso de la estrategia de mejora óptima, se investigan todos los vecinos y la solución actual se reemplaza por la del mejor vecino. En el caso de una estrategia de primera mejora, la solución actual se traslada al primer vecino cuyo valor de función de costo es menor que el de la solución actual. En la práctica, observamos en muchas aplicaciones que muy a menudo ambas estrategias conducen a la misma solución final, pero en tiempos de cálculo más pequeños cuando se usa la estrategia de primera mejora. También observamos que es más probable que ocurra una convergencia prematura a un mínimo local malo con una estrategia de mejora óptima.

2 Construcción de la lista restringida de candidatos

Una característica especialmente atractiva de GRASP es la facilidad con la que se puede implementar. Es necesario configurar y ajustar pocos parámetros. Por lo tanto, el desarrollo puede enfocarse en implementar estructuras de datos apropiadas para una construcción eficiente y algoritmos de búsqueda locales. GRASP tiene dos parámetros principales: uno relacionado con el criterio de parada y el otro con la calidad de los elementos en la lista restringida de candidatos.

El criterio de parada utilizado en el pseudocódigo descrito en la Figura 1 está determinado por el número Max Iterations de iteraciones. Aunque la probabilidad de encontrar una nueva solución que mejore la actual (la mejor solución actual) disminuye con el número de iteraciones, la calidad de la actual solo puede mejorar con el número de iteraciones. Dado que el tiempo de cálculo no varía mucho de una iteración a otra, el tiempo total de cálculo es predecible y aumenta linealmente con el número de iteraciones. En consecuencia, cuanto mayor sea el número de iteraciones, mayor será el tiempo de cálculo y mejor será la solución encontrada.

Para la construcción del RCL utilizado en la primera fase consideramos, sin pérdida de generalidad, un problema de minimización como el formulado en la Sección 1. Denotamos por c(e) el costo incremental asociado con la incorporación del elemento e ÿ E en la solución en construcción. En cualquier iteración de GRASP, sean c y $c_{máximo}$ ser, respectivamente, el menor y el mayor costo incremental.

La lista restringida de candidatos RCL está formada por los elementos e ÿ E con los mejores (es decir, los menores) costes incrementales c(e). Esta lista puede estar limitada por el número de elementos (basada en la cardinalidad) o por su calidad (basada en el valor). En el primer caso, está formado por los p elementos con los mejores costes incrementales, donde p es un parámetro. En este capítulo, el RCL está asociado con un parámetro de umbral ÿ ÿ [0,1]. La lista restringida de candidatos está formada por todos los elementos y ÿ E que se pueden insertar factiblemente en la solución parcial en construcción y cuya calidad

es superior al valor umbral, es decir, c(e) ÿ [c ^{me} , c ^{mi} +ÿ(c max ÿ ÿcmín)]. El caso
= 0 corresponde a un algoritmo codicioso puro, mientras que ÿ = 1 es equivalente a un algoritmo aleatorio
construcción. El pseudocódigo en la Figura 4 es un refinamiento del codicioso aleatorio
pseudocódigo de construcción que se muestra en la Figura 2. Muestra que el parámetro ÿ controla
las cantidades de codicia y aleatoriedad en el algoritmo.

```
procedimiento Construcción aleatorizada codiciosa (ÿ, semilla)
 1 Solución ÿ /0;
dos     Inicialice el conjunto de candidatos: C ÿ E;
3       Evaluar el costo incremental c(e) para todo e ÿ C;
4       mientras que C 6= /0 hacer
5           c^mi ÿ min{c(e) | y ÿC};
6           c^máximo ÿ máx{c(e) | y ÿC};
            RCL ÿ {e ÿ C | c(e) ÿ c^me +ÿ(c máx . ÿc min)};
            Seleccione un elemento s del RCL al azar;
            Solución ÿ Soluciónÿ {s};
7           Actualizar el conjunto de candidatos C;
8           Vuelva a evaluar el costo incremental c(e) para todo e ÿ C;
9 10 11 12 fin;
13 solución de retorno ;
Fin de la construcción aleatoria codiciosa.
```

Higo. 4 Pseudocódigo refinado de la fase de construcción.

La construcción GRASP puede verse como una técnica de muestreo repetitivo. cada
iteración produce una solución de muestra de una distribución desconocida, cuya media
y la varianza son funciones de la naturaleza restrictiva de la RCL. Por ejemplo, si el
RCL está restringido a un solo elemento, entonces se producirá la misma solución en todos
iteraciones La varianza de la distribución será cero y la media será igual
al valor de la solución codiciosa. Si se permite que el RCL tenga más elementos,
entonces se producirán muchas soluciones diferentes, lo que implica una varianza mayor. Sincero
la codicia juega un papel menor en este caso, el valor promedio de la solución debe ser
peor que la de la solución codiciosa. Sin embargo, el valor de la mejor solución encontrada
supera el valor promedio y puede ser casi óptimo o incluso óptimo. Es inprobable
que GRASP encontrará una solución óptima si el valor promedio de la solución es alto, incluso
si hay una gran variación en los valores generales de la solución. Por otro lado, si hay
poca variación en los valores generales de la solución, también es poco probable que GRASP encuentre
una solución óptima, incluso si la solución promedio es baja. Lo que a menudo conduce al bien
Las soluciones son valores de solución promedio relativamente bajos en presencia de un relativamente
gran varianza, como es el caso de ÿ = 0,2.

Otra observación interesante es que las distancias entre las soluciones obtenidas en cada iteración
y la mejor solución encontrada aumentan a medida que avanza la fase de construcción.
se mueve de más codicioso a más aleatorio. Esto hace que el tiempo promedio que toma el
búsqueda local para aumentar. Muy a menudo, se pueden generar muchas soluciones GRASP en el
misma cantidad de tiempo requerida por el procedimiento de búsqueda local para converger desde un
solo inicio aleatorio. En estos casos, el tiempo ahorrado al iniciar la búsqueda local desde

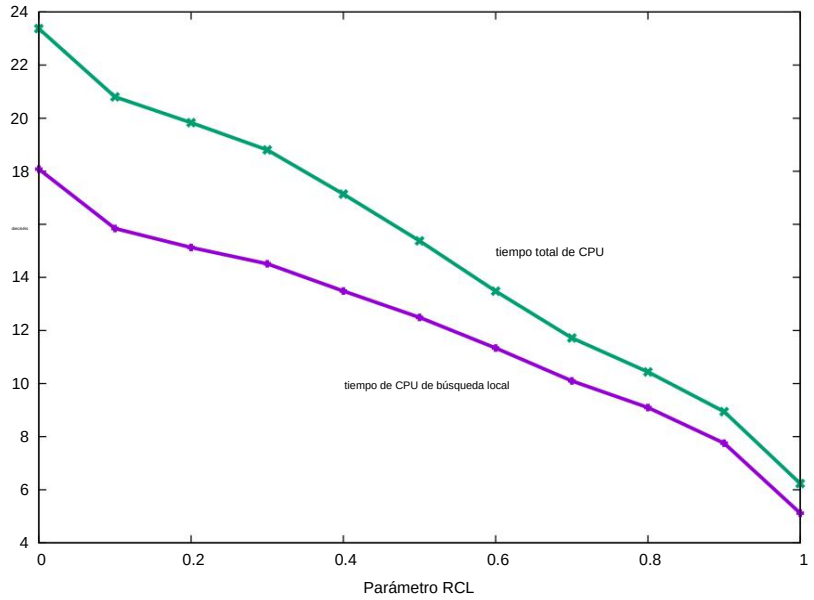
buenas soluciones iniciales se pueden utilizar para mejorar la calidad de la solución mediante la realización de más iteraciones GRASP.

Estos resultados se ilustran en la Tabla 1 y la Figura 5, para una instancia de la Problema de MAXSAT [213] donde se ejecutaron 1000 iteraciones. Para cada valor de ÿ que va de 0 (construcción puramente aleatoria para problemas de maximización) a 1 (construcción puramente aleatoria construcción codiciosa para problemas de maximización), damos en la Tabla 1 el promedio Distancia de Hamming entre cada solución construida durante la fase de construcción y el local óptimo correspondiente obtenido después de la búsqueda local, el número promedio de se mueve del primero al segundo, el tiempo de búsqueda local en segundos, y el total tiempo de procesamiento en segundos. La Figura 5 resume los valores para el total observado el tiempo de procesamiento y el tiempo de búsqueda local. Notamos que ambas medidas de tiempo disminuyen considerablemente a medida que ÿ tiende a 1, acercándose a la elección puramente codiciosa. en particular, observamos que el tiempo de búsqueda local promedio tomado por ÿ = 0 (puramente aleatorio) es aproximadamente 2,5 veces más que el tiempo empleado por ÿ = 0,9 (casi codicioso). en En este ejemplo, se pueden investigar de dos a tres soluciones construidas con avidez en el mismo tiempo necesario para aplicar la búsqueda local a una única solución construida aleatoriamente. La elección adecuada del valor del parámetro RCL es claramente crítica y relevante para lograr un buen equilibrio entre el tiempo de cálculo y la calidad de la solución.

Tabla 1 Número promedio de movimientos y tiempo de búsqueda local en función del parámetro RCL ÿ para el problema de la maximización.

ÿ	promedio	promedio	tiempo(s) de búsqueda	equipos totales
0,0	distancia		local 18	23,378
0,1	12.487		083 15	20,801
0,2	10.787		842 15	19,830
0,3	10.242		127 14	18,806
0,4	9.777		511 13	17,139
0,5	9.003		489 12	15,375
0,6	8.241		494 11	13,482
0,7	7.389		338 10	11,720
0,8	6.452		098 9	10,441
0,9	5.667		094 7	8,941
1,0	4.697 2.733	movimientos 12.373 10.709 10.166 9.727 58.957 18.189 7.341 6.436 56.235 4.691 2.733		6,234

Prais y Ribeiro [201] muestran que el uso de un único valor fijo para el parámetro ÿ de RCL a menudo dificulta encontrar una solución de alta calidad, que podría encontrarse si se utiliza otro valor. Proponen una ampliación del procedimiento GRASP básico, que denominan Reactive GRASP, en el que el parámetro ÿ se autoajusta y su valor se modifica periódicamente en función de la calidad de las soluciones obtenidas a lo largo de la búsqueda. En particular, los experimentos computacionales sobre el problema de la asignación de tráfico en satélites de comunicación [202] muestran que Reactive GRASP encuentra mejores soluciones que el algoritmo básico para muchas instancias de prueba. Estos resultados motivaron el estudio del comportamiento de GRASP con diferentes estrategias para la variación del valor del parámetro RCL ÿ:



Higo. 5 Tiempo total de CPU y tiempo de CPU de búsqueda local en función del parámetro RCL ÿ para un problema de maximización (1000 repeticiones para cada valor de ÿ).

A: ÿ autoajustado con un procedimiento Reactive GRASP;
E: ÿ elegido aleatoriamente de una distribución de probabilidad discreta uniforme;
H: ÿ elegido aleatoriamente de una distribución de probabilidad discreta no uniforme decreciente botón;
F: ÿ fijo, cercano al valor de elección puramente codicioso.

Resumimos los resultados obtenidos por los experimentos informados en [200, 201]. Estas cuatro estrategias se incorporan a los procedimientos GRASP desarrollados para cuatro problemas de optimización diferentes: (P-1) descomposición de matrices para asignación de tráfico en satélites de comunicaciones [202]; (P-2) cubierta del juego [90]; (P-3) MAX-SAT ponderado [213, 214]; y (P-4) planarización de grafos [215, 227]. Dejar

$$\ddot{y} = \{\ddot{y}1,...,\ddot{y}m\}$$

Sea el conjunto de valores posibles para el parámetro ÿ para las tres primeras estrategias. La estrategia para elegir y autoajustar el valor de ÿ en el caso del procedimiento GRASP Reactivo (R) se describe más adelante en la Sección 3. En el caso de la estrategia (E) basada en el uso de la distribución uniforme discreta, todas las probabilidades de elección son iguales a 1/m. El tercer caso correspondiente a la estrategia híbrida (H), en el que los autores consideraron p(ÿ = 0.1) = 0.5, p(ÿ = 0.2) = 0.25, p(ÿ = 0.3) = 0.125, p(ÿ = 0.4) = 0,03, p(ÿ = 0,5) = 0,03, p(ÿ = 0,6) = 0,03, p(ÿ = 0,7) = 0,01, p(ÿ = 0,8) = 0,01, p(ÿ = 0,9) = 0,01, y p(a = 1,0) = 0,005. Finalmente, en el último

estrategia (F), el valor de ÿ se fija como se recomienda en las referencias originales de problemas P-1 a P-4 citados anteriormente, donde se ajustó este parámetro para cada problema. Se consideró un subconjunto de las instancias de la literatura para cada clase de problemas de prueba. Los resultados informados en [201] se resumen en la Tabla 2. Para cada problema, primero enumere el número de instancias consideradas. A continuación, para cada estrategia, damos la número de veces que encontró la mejor solución (aciertos), así como el tiempo promedio de CPU (en segundos) en un IBM 9672 modelo R34. El número de iteraciones se fijó en 10.000.

Cuadro 2 Resultados computacionales para diferentes estrategias para la variación del parámetro ÿ.

	R		Y		H		F	
Instancias del problema	aciertos	tiempo	aciertos	tiempo	aciertos	tiempo	aciertos	tiempo
P-1	36	34 579,0	35	358,2	32	612,6	24	642,8
P-2	7	7 1346.8	6	1352,0	6	668,2	5	500,7
P-3	44	22 2463.7	23	2492.6	16	1740.9	11	1625.2
P-4	37	28 6363.1	21	7292.9	24	6326.5	19	5972.0
Total	124	91		85		78		59

La estrategia (F) presentó los tiempos de cálculo promedio más cortos para tres de los cuatro tipos de problemas. También fue el que presentó menor variabilidad en la construcción. soluciones y, en consecuencia, encontró la mejor solución el menor número de veces. el reactivo estrategia (R) es la que más a menudo encontró las mejores soluciones, sin embargo, en el costo de los tiempos de computación que son más largos que los de algunas de las otras estrategias. El elevado número de aciertos observado para la estrategia (E) también ilustra la eficacia de estrategias basadas en la variación del parámetro RCL.

3 Mecanismos de construcción alternativos

Una posible deficiencia del marco GRASP estándar es la independencia de sus iteraciones, es decir, el hecho de que no aprende del historial de búsqueda o de soluciones encontradas en iteraciones anteriores. Esto es así porque el algoritmo básico descarta información sobre cualquier solución encontrada previamente que no mejora el titular. La información recopilada de buenas soluciones se puede utilizar para implementar procedimientos basados en memoria para influir en la fase de construcción, modificando las probabilidades de selección asociadas con cada elemento de la RCL o haciendo cumplir elecciones específicas. Otra posible deficiencia de la construcción aleatoria codiciosa es su complejidad. En cada paso de la construcción, cada candidato aún no seleccionado elemento tiene que ser evaluado por la función codiciosa. En los casos en que la diferencia entre el número de elementos en el conjunto base y el número de elementos que aparece en una solución es grande, esto puede no ser muy eficiente.

En esta sección, consideramos mejoras y técnicas alternativas para la fase de construcción de GRASP. Incluyen aleatorio más codicioso, muestreado codicioso, Re-

GRASP activo, perturbaciones de costos, funciones de sesgo, memoria y aprendizaje, búsqueda local en soluciones parcialmente construidas y heurística GRASP lagrangiana.

3.1 Construcción aleatoria más codiciosa y codiciosa muestreada

En la Sección 2, describimos el esquema de construcción semi-codicioso utilizado para construir soluciones codiciosas aleatorias que sirven como puntos de partida para la búsqueda local. En [221] se propusieron otros dos enfoques codiciosos aleatorios, con complejidades más pequeñas en el peor de los casos que el algoritmo semi-codicioso.

En lugar de combinar la codicia y la aleatoriedad en cada paso del procedimiento de construcción, el esquema aleatorio más codicioso aplica la aleatoriedad durante los primeros p pasos de construcción para producir una solución parcial aleatoria. A continuación, el algoritmo completa la solución con uno o más pasos de construcción codiciosos puros. La solución resultante es codicioso aleatorio. Se puede controlar el equilibrio entre codicia y aleatoriedad en la construcción cambiando el valor del parámetro p. Los valores más grandes de par están asociados con soluciones que son más aleatorias, mientras que los valores más pequeños dan como resultado soluciones más codiciosas.

Similar al procedimiento aleatorio más codicioso, la construcción codiciosa muestreada también combina aleatoriedad y codicia pero de una manera diferente. Este procedimiento también está controlado por un parámetro p. En cada paso del proceso de construcción, el procedimiento crea una lista restringida de candidatos mediante un muestreo aleatorio de min{p,| C|} elementos del conjunto de candidatos C. Cada elemento de la RCL es evaluado por la función codiciosa. El elemento con el valor de función voraz más pequeño se agrega a la solución parcial.
Este proceso de dos pasos se repite hasta que no haya más elementos candidatos. La solución resultante también es ambiciosa al azar. El equilibrio entre codicia y aleatoriedad se puede controlar cambiando el valor del parámetro p, es decir, el número de elementos candidatos que se muestrean. Los tamaños de muestra pequeños conducen a soluciones más aleatorias, mientras que los tamaños de muestra grandes conducen a soluciones más codiciosas.

3.2 Reactivar GRASP

La primera estrategia para incorporar un mecanismo de aprendizaje en la fase de construcción sin memoria del GRASP básico fue el procedimiento GRASP reactivo presentado en la Sección 2. En este caso, el valor del parámetro RCL no es fijo, sino que se selecciona aleatoriamente en cada iteración de un conjunto discreto de valores posibles. Esta selección está guiada por los valores de solución encontrados a lo largo de las iteraciones anteriores. Una forma de lograr esto es usar la regla propuesta en [202]. Sea ÿ = {ÿ1,...,ÿm} un conjunto de valores posibles para ÿ. Las probabilidades asociadas con la elección de cada valor se igualan inicialmente a pi = 1/m, para i = 1,...,m. Además, sea z la solución principal y sea Ai el valor promedio de todas las soluciones encontradas usando ÿ = ÿi , para i = 1,...,m. Las probabilidades de selección se

"recalculan periódicamente y

ado tomando pi = qi/ÿ m con qi = z ÿ/Ai para i = 1, ..., m. El valor de qi, i=1 qi, será mayor para valores de y = yr dando lugar a las mejores soluciones en promedio. Los valores más grandes de qi corresponden a valores más adecuados para el parámetro ÿ. Las probabilidades asociadas con los valores más apropiados aumentarán luego cuando se reevalúen.

El enfoque reactivo conduce a mejoras sobre el GRASP básico en términos de solidez y calidad de la solución, debido a una mayor diversificación y una menor dependencia del ajuste de parámetros. Además de las aplicaciones en [200, 201, 202], este enfoque se ha utilizado en la planificación de redes de transmisión de sistemas eléctricos [48], programación de talleres [47], asignación de canales en redes de telefonía móvil [117], desarrollo de redes de caminos rurales [249], ubicación capacitada [70], strip-packing [16] y un problema combinado de producción y distribución [50].

3.3 Perturbaciones de costos

La idea de introducir algo de ruido en los costos originales es similar a la del llamado método de "ruido" de Charon y Hudry [57, 58]. Añade más flexibilidad al diseño del algoritmo y puede ser incluso más eficaz que la construcción aleatoria codiciosa del procedimiento GRASP básico en circunstancias en las que los algoritmos de construcción no son muy sensibles a la aleatorización. De hecho, este es el caso de la heurística de ruta más corta de Takahashi y Matsuyama [259], utilizada como uno de los principales bloques de construcción de la fase de construcción del procedimiento GRASP híbrido propuesto por Ribeiro et al. [234] para el problema de Steiner en gráficos. Otra situación en la que las perturbaciones de costos pueden ser muy efectivas aparece cuando no se dispone de un algoritmo codicioso capaz de realizar una aleatorización directa. Este es el caso del GRASP híbrido desarrollado por Canuto et al. [54] para el problema del árbol de Steiner de recolección de premios, que hace uso del algoritmo primal-dual de Goemans y Williamson [116] para construir soluciones iniciales usando costos alterados.

En el caso del GRASP para el problema del árbol de Steiner de recolección de premios descrito en [54], se construye una nueva solución en cada iteración utilizando premios de nodo actualizados por una función de perturbación, basada en la estructura de la solución actual. Se utilizaron dos esquemas de perturbación de premios diferentes. En la perturbación por eliminaciones, el algoritmo primal-dual utilizado en la fase de construcción es impulsado a construir una nueva solución sin algunos de los nodos que aparecieron en la solución construida en la iteración anterior. En la perturbación por cambios de precio, se introduce algo de ruido en los premios de nodo para cambiar la función objetivo, de manera similar a lo propuesto en [57, 58].

Los métodos de perturbación de costos utilizados en GRASP para el problema del árbol mínimo de Steiner descrito en [234] incorporan mecanismos de aprendizaje asociados con estrategias de intensificación y diversificación. Se aplicaron tres métodos distintos de aleatorización de peso. En una iteración GRASP dada, el peso modificado de cada borde se selecciona aleatoriamente de una distribución uniforme en un intervalo que depende del método de aleatorización de peso seleccionado aplicado en esa iteración. Los diferentes métodos de aleatorización de pesos usan información de frecuencia y pueden usarse para

hacer cumplir las estrategias de intensificación y diversificación. Los resultados experimentales informados en [234] muestran que la estrategia que combina estos tres métodos de perturbación es más robusta que cualquiera de ellos utilizados de forma aislada, lo que genera los mejores resultados generales en una combinación bastante amplia de instancias de prueba con diferentes características. La heurística GRASP que usa esta estrategia de perturbación de costos se encuentra entre las heurísticas más efectivas actualmente disponibles para el problema de Steiner en gráficos.

3.4 Funciones de sesgo

En el procedimiento de construcción del GRASP básico, el siguiente elemento a introducir en la solución se elige aleatoriamente entre los candidatos del RCL. A los elementos de la RCL se les asignan las mismas probabilidades de ser elegidos. Sin embargo, se puede usar cualquier distribución de probabilidad para sesgar la selección hacia algunos candidatos en particular. Otro mecanismo de construcción fue propuesto por Bresina [51], donde se introduce una familia de tales distribuciones de probabilidad. Se basan en el rango r(e) asignado a cada elemento candidato y ÿC, según su valor de función voraz. Se propusieron varias funciones de sesgo, tales como: • sesgo aleatorio: sesgo(r) = 1; • sesgo lineal: sesgo(r) = 1/r; • sesgo logarítmico: sesgo(r) = logÿ1 (r +1); • sesgo exponencial: bias(r) = e • sesgo polinomial de orden n: bias(r) = r Sea r(e) el rango del elemento e ÿ C y sea bias(r(e)) uno de los sesgos funciones definidas anteriormente. Una vez evaluados estos valores para todos los elementos del conjunto candidato C, la probabilidad ÿ(e) de seleccionar el elemento e ÿ C es

$$\ddot{y}(e) = \ddot{y}e\ 0\ddot{y}C\ \text{sesgo}(r(e\ 0)).\ \frac{\text{sesgo}(r(e))}{\ddot{y}_{n}} \qquad (1)$$

La evaluación de estas funciones de polarización puede estar restringida a los elementos de la RCL. El procedimiento de selección de Bresina restringido a elementos de la RCL se utilizó en [47]. El GRASP estándar utiliza una función de sesgo aleatorio.

3.5 Construcción inteligente: memoria y aprendizaje

Fleurent y Glover [105] observaron que el GRASP básico no utiliza una memoria a largo plazo (información recopilada en iteraciones anteriores) y propusieron un esquema de memoria a largo plazo para abordar este problema en la heurística de inicio múltiple. La memoria a largo plazo es uno de los fundamentos en los que se basa la búsqueda tabú.

Su esquema mantiene un grupo de soluciones de élite para ser utilizadas en la fase de construcción. Para convertirse en una solución de élite, una solución debe ser mejor que la mejor

miembro del grupo, o mejor que su peor miembro y suficientemente diferente de las otras soluciones del grupo. Por ejemplo, uno puede contar componentes de vectores de solución idénticos y establecer un umbral para el rechazo.

Una variable fuertemente determinada es aquella que no se puede cambiar sin erosionar el objetivo o cambiar significativamente otras variables. La variable consistente es aquella que recibe un valor particular en una gran parte del conjunto de soluciones de élite. Sea la función de intensidad I(e) una medida de las características de fuerte determinación y consistencia de un elemento de solución e ÿ E. Entonces, I(e) se vuelve más grande y aparece con mayor frecuencia en el conjunto de soluciones de élite. La función de intensidad se utiliza en la fase de construcción de la siguiente manera. Recuerde que c(e) es la función codiciosa, es decir, el costo incremental asociado con la incorporación del elemento y ÿ E en la solución en construcción. Sea K(e) = F(c(e),I(e)) una función de las funciones codiciosa y de intensificación. Por ejemplo, K(e) = ÿc(e)+I(e). El esquema de intensificación sesga la selección del RCL hacia aquellos elementos e ÿ E con un valor alto de K(e) estableciendo su probabilidad de selección en p(e) = K(e)/ÿsÿRCLK(s).

La función K(e) puede variar con el tiempo cambiando el valor de ÿ. Por ejemplo, ÿ se puede establecer en un valor grande que se reduce cuando se requiere diversificación. Los procedimientos para cambiar el valor de ÿ están dados por Fleurent y Glover [105] y Binato et al. [47].

3.6 POP en construcción

El Principio de Optimalidad Próxima (POP) se basa en la idea de que “es probable que se encuentren buenas soluciones en un nivel 'cerca de' buenas soluciones en un nivel adyacente” [114]. Fleurent y Glover [105] proporcionaron una interpretación GRASP de este principio. Ellos sugirieron que las imperfecciones introducidas durante los pasos de la construcción de GRASP se pueden “eliminar” aplicando la búsqueda local durante (y no solo al final de) la fase de construcción de GRASP.

Debido a consideraciones de eficiencia, la implementación práctica de POP a GRASP consiste en aplicar la búsqueda local varias veces durante la fase de construcción, pero no en cada iteración de construcción. La búsqueda local fue aplicada por Binato et al. [47] después de que se hayan realizado el 40% y el 80% de los movimientos de construcción, así como al final de la fase de construcción.

3.7 Heurísticas GRASP lagrangianas

La relajación lagrangiana [45, 104] es una técnica de programación matemática que se puede utilizar para proporcionar límites inferiores para problemas de minimización. Held y Karp [122, 123] fueron de los primeros en explorar el uso de multiplicadores duales producidos por la relajación lagrangiana para derivar cotas inferiores, aplicando esta idea en el contexto del problema del viajante de comercio. La heurística lagrangiana explora más a fondo el uso de

diferentes multiplicadores duales para generar soluciones factibles. Beasley [43, 44] describió una heurística lagrangiana para la cobertura de conjuntos.

3.7.1 Relajación lagrangiana y optimización de subgradiente

La relajación lagrangiana se puede utilizar para proporcionar límites inferiores para problemas de optimización de operaciones combinatorias. Sin embargo, las soluciones primarias producidas por los algoritmos utilizados para resolver el problema dual de Lagrange no son necesariamente factibles. La heurística lagrangiana aprovecha los multiplicadores duales para generar soluciones primarias factibles.

Dado un problema de programación matemática P formulado como

$$F^{\ddot{y}} = \min f(x) \quad (dos)$$

$$g_i(x) \ddot{y} 0, \; i = 1,...,m, \quad (3)$$

$$x \ddot{y} X, \quad (4)$$

su relajación lagrangiana se obtiene asociando multiplicadores duales $\ddot{y}i \ddot{y} R+$ a cada desigualdad (3), para i = 1,...,m. Esto da como resultado el siguiente problema de acción de relajación de Lagrange LRP(ÿ)

$$\text{mi } f^{0}(x) = f(x) + \sum_{yo=1}^{metro} \ddot{y}i \cdot g\ddot{i}(x) \quad (5)$$

$$x \ddot{y} X, \quad (4)$$

cuya solución óptima x(ÿ) da un límite inferior f al problema $^{0}(x(\ddot{y}))$ al valor óptimo de original P definido por (2) a (4). El mejor límite inferior (dual) viene dado por la solución del problema dual lagrangiano D

$$f_D = f^{0}(x(\ddot{y}^{\ddot{y}})) = \max_{\ddot{y}\ddot{y}Rm+} F^{0}(x(\ddot{y})). \quad (6)$$

La optimización del subgradiente se utiliza para resolver el problema dual D definido por (6). Los algoritmos de subgradiente parten de cualquier conjunto factible de multiplicadores duales, como $\ddot{y}i = 0$, para i = 1,...,m, y generan iterativamente multiplicadores actualizados.

En cualquier iteración q, sea $\ddot{y}\,q$ el vector actual de multiplicadores y sea $x(\ddot{y}\,q\,) (x(\ddot{y}\,q\,))$. una solución óptima al problema LRP($\ddot{y}\,q\,$), cuyo valor óptimo es f^{0} Además, sea $\bar{f}$ una cota superior conocida del valor óptimo del problema P. Además, sea $g\,q \ddot{y} R = gi(x(\ddot{y}\,q\,))$ para i = 1,...,m. Para actualizar los multiplicadores de Lagrange, el algoritmo utiliza un tamaño de paso

$$\text{cuando}_{-} = \frac{\ddot{y} \cdot (\bar{f}\ddot{y} f^{0}(x(\ddot{y}\,q\,)))}{\ddot{y}_{yo=1}^{metro} (\text{gramo}i\,q\ddot{i})^{dos}}, \quad (7)$$

) donde $\ddot{y} \ddot{y} (0,2)$. Luego, los multiplicadores se actualizan como

ÿ $_i^{q+1}$ = máx{0;ÿ $_i^{q}$ qq ÿd q · gramo yo }, i = 1,...,m, (8)

y el algoritmo de subgradiente procede a la iteración q+1.

3.7.2 Una plantilla para la heurística lagrangiana

Describimos a continuación una plantilla para la heurística lagrangiana que hace uso de los multiplicadores duales ÿ q y de la solución óptima x(ÿ q) para cada problema LRP(ÿ q) para construir soluciones factibles al problema original P definido por (2) para (4). A continuación, asumimos que la función objetivo y todas las restricciones son funciones lineales, es decir, f(x) = ÿ no j=1 cjxj y gi(x) = ÿ no j=1 dijxj ÿ ei , para i = 1,...,m.

Sea H una heurística primal que construye una solución factible x a P, comenzando desde = x(ÿ q) en cada iteración q del algoritmo del subgradiente.

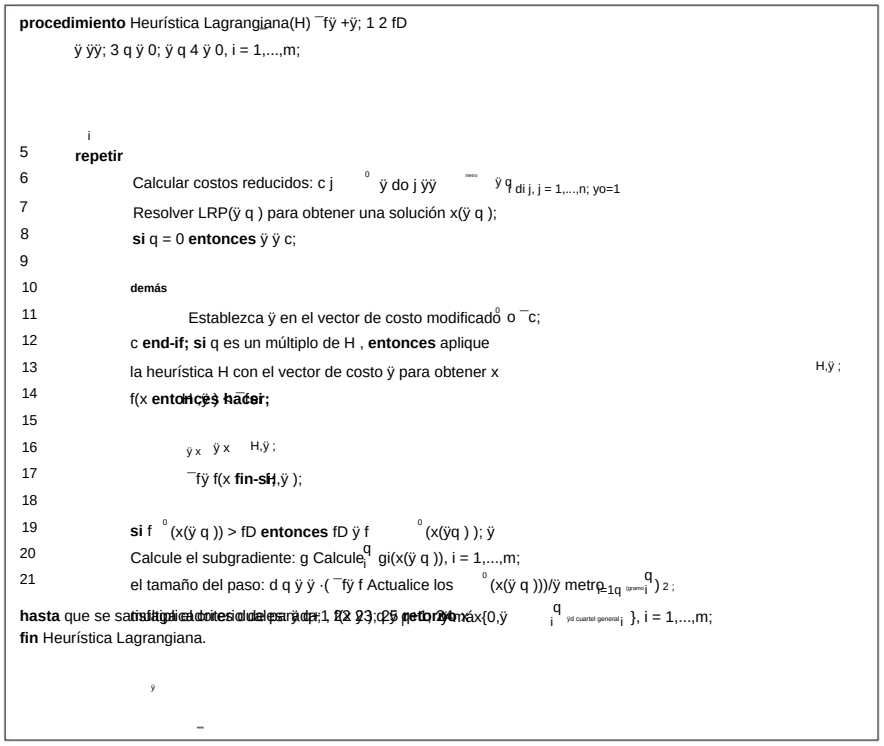
0 la solución inicial x

Primero se aplica la heurística H usando los costos originales c j , es decir, usando la función de costo f(x). En cualquier iteración subsiguiente q del algoritmo de subgradiente, H usa los costos complementarios ¯cj= j (1ÿxj(ÿ q))· c j do j ÿ ÿ metro yo=1 reducidos lagrangianos ÿ q c di j o los costos

Sea x Sea H ,ÿ la solución obtenida por la heurística H , usando un vector de costo genérico ÿ correspondiente a uno de los esquemas de costo modificados anteriores o al vector de costo original. Su costo puede usarse para actualizar el límite superior ¯fal valor óptimo del problema original. Este límite superior se puede mejorar aún más mediante la búsqueda local y se utiliza para ajustar el tamaño del paso definido por la ecuación (7).

La figura 6 muestra el pseudocódigo de una heurística lagrangiana. Las líneas 1 a 4 inicializan los límites superior e inferior, el contador de iteraciones y los multiplicadores duales. Las acciones de iteración del algoritmo subgradiente se realizan a lo largo del ciclo definido en las líneas 5 a 24. Los costos reducidos se calculan en la línea 6 y el problema de relajación de Lagrangian se resuelve en la línea 7. En la primera iteración de la heurística de Lagrangian, el costo original el vector se asigna a ÿ en la línea 9, mientras que en las iteraciones posteriores se asigna un vector de costo modificado a ÿ en la línea 11. La heurística H está en la línea 13 en la primera iteración y después de cada H iteraciones aplicadas a partir de entonces (es la iteración se aplica después H ,ÿ al problema de un múltiplo del parámetro de entrada H) para producir una solución factible x del contador es P. Si el costo de esta solución es menor que el límite superior actual, entonces la mejor (x(ÿ q)) solución y su costo se actualizan en las líneas 14 a 18. Si el límite inferior f es mayor 0 es que el límite inferior actual fD , entonces fD se actualiza en la línea 19. La línea 20 calcula un subgradiente en x(ÿ q) y la línea 21 calcula el Numero de pie. Los multiplicadores duales se actualizan en la línea 22 y el contador de iteraciones se incrementa en la línea 23. La mejor solución encontrada y su costo se devuelven en la línea 24.

La estrategia propuesta por Held, Wolfe y Crowder [124] se usa comúnmente en la implementación de la heurística lagrangiana para actualizar los multiplicadores duales de una iteración a la siguiente. Beasley [44] informó como computacionalmente útil el ajuste de los componentes de los subgradientes a cero siempre que no contribuyan efectivamente a la actualización de los multiplicadores, es decir, establecer arbitrariamente g = 0 siempre que g_i^q > 0 y ÿ q = 0, para i = 1,...,m.



Higo. 6 Pseudocódigo de una plantilla para una heurística lagrangiana.

Diferentes opciones para la solución inicial x 0, para los costos modificados ÿ, y para la heurística primal H misma conducen a diferentes variantes del algoritmo anterior. El parámetro entero H define la frecuencia en la que se aplica H. Cuanto menor sea el valor de H, mayor será el número de veces que se aplica H. Por lo tanto, el tiempo de cálculo aumenta a medida que disminuye el valor de H. En particular, se debe establecer H = 1 si se va a aplicar la heurística primaria H en cada iteración.

3.7.3 GRASP Lagrangiano

Pessoa, Resende y Ribeiro [195, 196] propusieron la hibridación de GRASP y la relajación lagrangiana que condujo a la heurística lagrangiana de GRASP que se describe a continuación. Las diferentes opciones para la heurística primaria H en la plantilla del algoritmo de la Figura 6 conducen a distintas heurísticas lagrangianas. Consideramos dos variantes: la primera hace uso de un algoritmo voraz con búsqueda local, mientras que en la segunda se usa un GRASP con reenlace de ruta (ver Sección 4).

Heurística codiciosa: Esta heurística repara codiciosamente la solución x(ÿ q) producida en la línea 7 de la heurística lagrangiana descrita en la Figura 6 para hacerla factible para

problema P. Hace uso de los costos modificados (c^0 o $\bar{c}$). Se puede aplicar la búsqueda local a la solución resultante, usando el vector de costo original c. Nos referimos a este enfoque como una heurística lagrangiana codiciosa (GLH).

Heurística GRASP: en lugar de simplemente realizar un paso de construcción seguido por búsqueda local, como lo hace GLH, esta variante se aplica a la heurística GRASP para reparar el solución $x(\psi q)$ producida en la línea 7 de la heurística lagrangiana para hacerla factible para problema P

Aunque la heurística GRASP produce mejores soluciones que la heurística codiciosa, la heurística codiciosa es mucho más rápida. Para abordar adecuadamente esta compensación, adapte la línea 10 de la Figura 6 para usar la heurística GRASP con probabilidad ψ y la heurística codiciosa con probabilidad $1\psi\psi$, donde ψ es un parámetro del algoritmo.

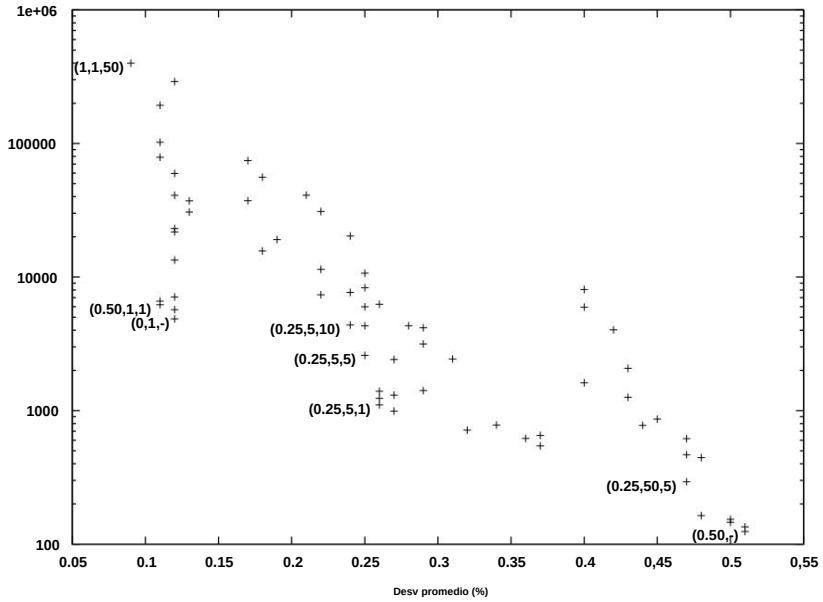
Notamos que esta estrategia involucra tres parámetros principales: el número H de acciones de iteración después de las cuales siempre se aplica la heurística básica, el número Q de iteraciones realizado por la heurística GRASP cuando se elige como la heurística principal, y la probabilidad ψ de elegir la heurística GRASP como H . Nos referiremos a la Heurística lagrangiana que utiliza esta estrategia híbrida como LAGRASP(ψ,H,Q).

A continuación resumimos los resultados computacionales obtenidos para 135 instancias del conjunto problema de k-recubrimiento. Estas instancias tienen hasta 400 restricciones y 4000 binarios variables El problema del k-covering set, o multi-covering set, es una extensión del problema clásico de cobertura de conjuntos, en el que se requiere cubrir cada elemento al menos k veces. El problema encuentra aplicaciones en el diseño de la comunicación. redes y en biología computacional.

El primer experimento con la heurística lagrangiana de GRASP estableció la relación entre los tiempos de ejecución y la calidad de la solución para diferentes configuraciones de parámetros. Parámetro ψ, la probabilidad de que GRASP se aplique como la heurística H 0, 0.25, 0.50, , se fijó en 0.75 y 1. Parámetro H, el número de iteraciones entre sucesivas llamadas a la heurística H , se fijó en 1, 5, 10 y 50. El parámetro Q, el número de iteraciones realizadas por la heurística GRASP, se fijó en 1, 5, 10 y 50. Combinando algunos de estos valores de parámetros, 68 variantes de el híbrido LAGRASP(ψ,H,Q) se crearon heurísticas. Cada variante se aplicó ocho veces a un subconjunto de 21 instancias, con diferentes semillas iniciales que se le dieron al generador de números aleatorios.

El gráfico de la Figura 7 resume los resultados de todas las variantes evaluadas, mostrando puntos cuyas coordenadas son los valores de la desviación promedio de la mejor valor de solución conocido y el tiempo total en segundos para procesar las ocho corridas en todas las instancias, para cada combinación de valores de parámetros. Se identifican ocho variantes de especial interés y se etiquetan con los correspondientes parámetros ψ, H y P, en este orden. Estas variantes corresponden a puntos de Pareto seleccionados en el gráfico en Figura 7. Ajuste ψ = 0 y H = 1 correspondiente a la heurística lagrangiana voraz (GLH) o, de forma equivalente, a LAGRASP(0,1,-), cuya desviación media (en porcentaje de edad) del mejor valor asciende al 0,12% en 4.859,16 segundos de funcionamiento total equipo. La Tabla 3 muestra la desviación promedio del valor de solución mejor conocido y el tiempo total para cada una de las ocho variantes seleccionadas.

En otro experimento, se consideraron las 135 instancias de prueba para la comparación. de las ocho variantes de LAGRASP seleccionadas anteriormente. La Tabla 4 resume los resultados



Higo. 7 Desviación promedio del mejor valor y tiempo de ejecución total para 68 variantes diferentes de LAGRASP sobre un conjunto reducido de 21 instancias del problema de cobertura de conjuntos k: cada punto representa un combinación única de parámetros ÿ, H y Q.

Tabla 3 Resumen de los resultados numéricos obtenidos con las variantes seleccionadas del GRASP Heurística lagrangiana sobre un conjunto reducido de 21 instancias del problema de cobertura de conjunto k. Estos valores corresponden a las coordenadas de las variantes seleccionadas en la Figura 7. El tiempo total se da en segundos.

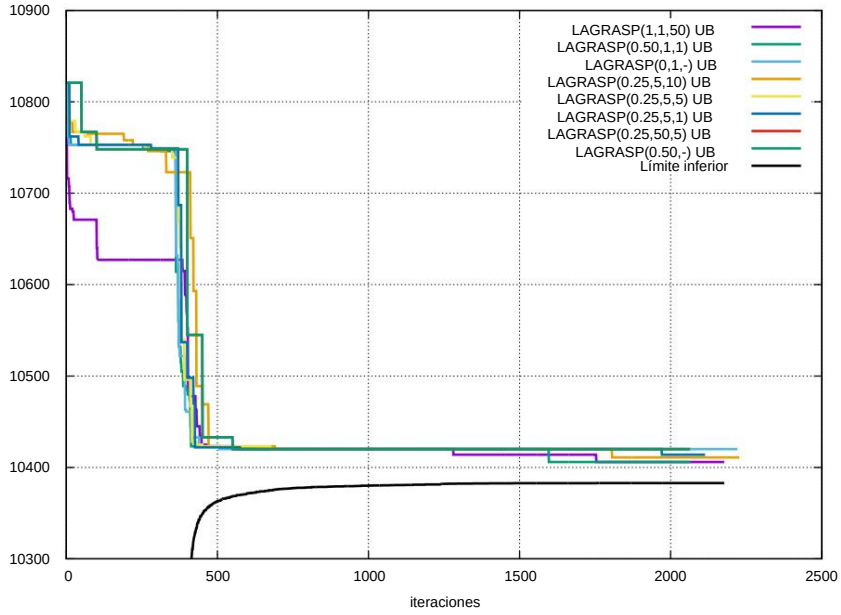
heurístico	Desviación media	Tiempo total (s)
LAGRASP(1,1,50)	0,09%	399.101,14
LAGRASP(0.50,1,1)	0,11%	6.198,46
LAGRASP(0,1,-)	0,12%	4.859,16
LAGRASP(0.25,5,10)	0,24%	4.373,56
LAGRASP(0.25,5,5)	0,25%	2.589,79
LAGRASP(0.25,5,1)	0,26%	1.101,64
LAGRASP(0.25,50,5)	0,47%	292,95
LAGRASP(0.50,-)	0,51%	124,26

obtenidos por las ocho variantes seleccionadas. Muestra que LAGRASP(1,1,50) encontró el mejores soluciones, con una desviación media de los mejores valores de 0,079%. También encontró las soluciones más conocidas en 365 ejecuciones (cada variante se ejecutó ocho veces en cada instancia), nuevamente con el mejor desempeño cuando las ocho variantes se evalúan una al lado de la otra, aunque sus tiempos de ejecución son los más largos. Por otro lado, el Se observaron tiempos de ejecución más pequeños para LAGRASP (0,50,-), que fue más de 3000 veces más rápido que LAGRASP(1,1,50) pero encontró las soluciones de peor calidad entre las ocho variantes consideradas.

Tabla 4 Resumen de los resultados numéricos obtenidos con las variantes seleccionadas del GRASP
Heurística lagrangiana sobre el conjunto completo de 135 instancias del problema de cobertura del conjunto k. el tiempo total se da en segundos.

LAGRASP	Desviación media	Aciertos	Tiempo(s) total(es)
heurístico(1,1,50)	0,079%	365	1.803.283,64
LAGRASP(0.50,1,1)	0,134%	242	30.489,17
LAGRASP(0,1,-)	0,135%	238	24.274,72
LAGRASP(0.25,5,10)	0,235%	168	22.475,54
LAGRASP(0.25,5,5)	0,247%	163	11.263,80
LAGRASP(0.25,5,1)	0,249%	164	5.347,78
LAGRASP(0.25,50,5)	0,442%	100	1.553,35
LAGRASP(0.50,-)	0,439%	97	569,30

La Figura 8 ilustra los méritos del enfoque propuesto para una de las instancias de prueba. Primero notamos que todas las variantes alcanzan los mismos límites inferiores, como se esperaba, ya que dependen exclusivamente del algoritmo de subgradiente común. Sin embargo, como el límite inferior parece estabilizarse, el límite superior obtenido por GLH
(LAGRASP(0,1,-) también parece congelarse. Por otro lado, las otras variantes continúan realizando mejoras al descubrir mejores límites superiores, ya que la construcción aleatoria de GRASP les ayuda a escapar de soluciones localmente óptimas y
encontrar nuevos límites superiores mejorados.



Higo. 8 Evolución de los límites inferior y superior sobre las iteraciones para diferentes variantes de LAGRASP.
El número de iteraciones que toma cada variante de LAGRASP depende del tamaño del paso, que a su vez depende de los límites superiores producidos por cada heurística.

Finalmente, proporcionamos una comparación entre GRASP con reenlace de ruta hacia atrás y las variantes LAGRASP en las 135 instancias de prueba cuando los mismos límites de tiempo se utilizan para detener todas las heurísticas. Se realizaron ocho ejecuciones para cada heurística y cada instancia, utilizando diferentes semillas iniciales para el generador de números aleatorios. cada la heurística se ejecutó un total de (8 × 135 =) 1080 veces. Los resultados en la Tabla 5 muestran que todas las variantes de LAGRASP superaron a GRASP con reenlace de ruta hacia atrás y fueron capaces de encontrar soluciones cuyos costos son muy cercanos o tan buenos como los mejores valores de solución conocidos, mientras que GRASP con reenlace de ruta hacia atrás encontró soluciones cuyos costos son en promedio 4.05% mayores que los valores de solución más conocidos.

Tabla 5 Resumen de resultados para las mejores variantes de LAGRASP y GRASP.

heurístico	Desviación media Aciertos
LAGRASP(1,1,50)	3.30% 0
LAGRASP(0.50,1,1)	0,35% 171
LAGRASP(0,1,-)	0,35% 173
LAGRASP(0.25,5,10)	0,45% 138
LAGRASP(0.25,5,5)	0,45% 143
LAGRASP(0.25,5,1)	0,46% 137
LAGRASP(0.25,50,5)	0,65% 97
LAGRASP(0.50,-)	0,65% 93
GRASP con reenlace de ruta hacia atrás	4.05% 0

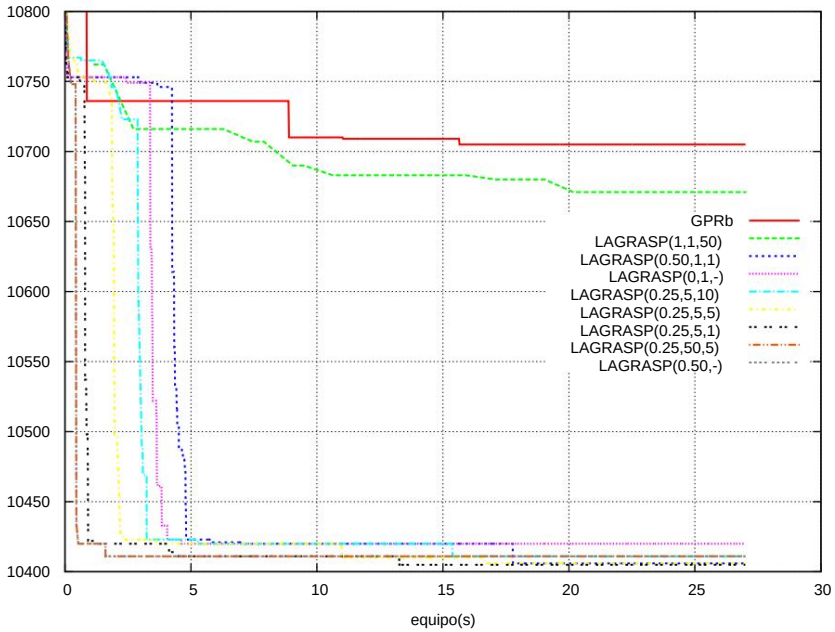
La Figura 9 muestra para una instancia de prueba el comportamiento típico de estas heurísticas. En a diferencia del GRASP con reenlace de rutas, las heurísticas lagrangianas son capaces de escapar de los óptimos locales por más tiempo y seguir mejorando las soluciones para obtener los mejores resultados

Observamos que una característica importante de las heurísticas lagrangianas es que proporcionan no solo una solución factible (que da un límite superior, en el caso de un problema de minimización), sino también un límite inferior que puede usarse para dar una estimación de la brecha de optimalidad que puede ser considerada como un criterio de parada.

4 Reenlace de rutas

La heurística LAGRASP presentada en la Sección 3.7.3 hizo uso de la vinculación de caminos. La vinculación de rutas es otra mejora del procedimiento GRASP básico, lo que lleva a mejoras significativas tanto en la calidad de la solución como en los tiempos de ejecución. Esta tecnica fue propuesta originalmente por Glover [111] como una estrategia de intensificación para explorar trayectorias que conectan soluciones de élite obtenidas mediante búsqueda tabú o búsqueda dispersa [112, 114, 115].

Consideramos el grafo no dirigido asociado al espacio solución G = (S,M), donde los nodos en S corresponden a soluciones factibles y las aristas en M corresponden a movimientos en la estructura de vecindad, es decir (i, j) ÿ M si y solo si i ÿ S, j ÿ S, j ÿ N(i) e i ÿ N(j), donde N(s) denota la vecindad de un nodo s ÿ S.



Higo. 9 Evolución de los costos de la solución con el tiempo para las mejores variantes de LAGRASP y GRASP con back-path-relinking (GPRb).

La reconexión de caminos generalmente se lleva a cabo entre dos soluciones: una se denomina solución inicial, mientras que la otra es la solución guía. Uno o más caminos en el gráfico de espacio de soluciones que conectan estas soluciones se exploran en la búsqueda de mejores soluciones. La búsqueda local se aplica a la mejor solución en cada uno de estos caminos, ya que no hay garantía de que la mejor solución sea localmente óptima.

Sea s ÿ S un nodo en el camino entre una solución inicial y una solución guía g ÿ S. No todas las soluciones en la vecindad N(s) son candidatas a seguir s en el camino de s a g. Restringimos la elección solo a aquellas soluciones que son más similares a g que a s. Esto se logra seleccionando movimientos de los atributos de introducción contenidos en la solución guía g. Por lo tanto, la vinculación de caminos puede verse como una estrategia que busca incorporar atributos de soluciones de alta calidad (es decir, las soluciones de élite guía), favoreciendo estos atributos en los movimientos seleccionados.

El uso de la reconexión de caminos dentro de un procedimiento GRASP, como una estrategia de intensificación aplicada a cada solución localmente óptima, fue propuesto por primera vez por Laguna y Mart´ÿ [147]. Le siguieron varias extensiones, mejoras y aplicaciones exitosas [6, 7, 22, 54, 97, 183, 206, 217, 221, 222, 229, 234, 249]. En [218] se puede encontrar un estudio de GRASP con reenlace de rutas.

Mejorar GRASP con la vinculación de rutas casi siempre mejora el rendimiento de la heurística. Como ilustración, la Figura 10 muestra gráficas de tiempo hasta el objetivo [9, 10, 230, 231, 232] para GRASP y GRASP con implementaciones de vinculación de rutas para cuatro aplicaciones diferentes. Estas gráficas de tiempo hasta el objetivo muestran el problema acumulativo empírico

distribuciones de capacidad de la variable aleatoria de tiempo hasta el objetivo cuando se usa GRASP puro y GRASP con reconexión de rutas, es decir, el tiempo necesario para encontrar una solución al menos tan buena como un valor objetivo preespecificado. Para todos los problemas, los gráficos muestran que GRASP con la vinculación de rutas puede encontrar soluciones objetivo más rápido que GRASP.

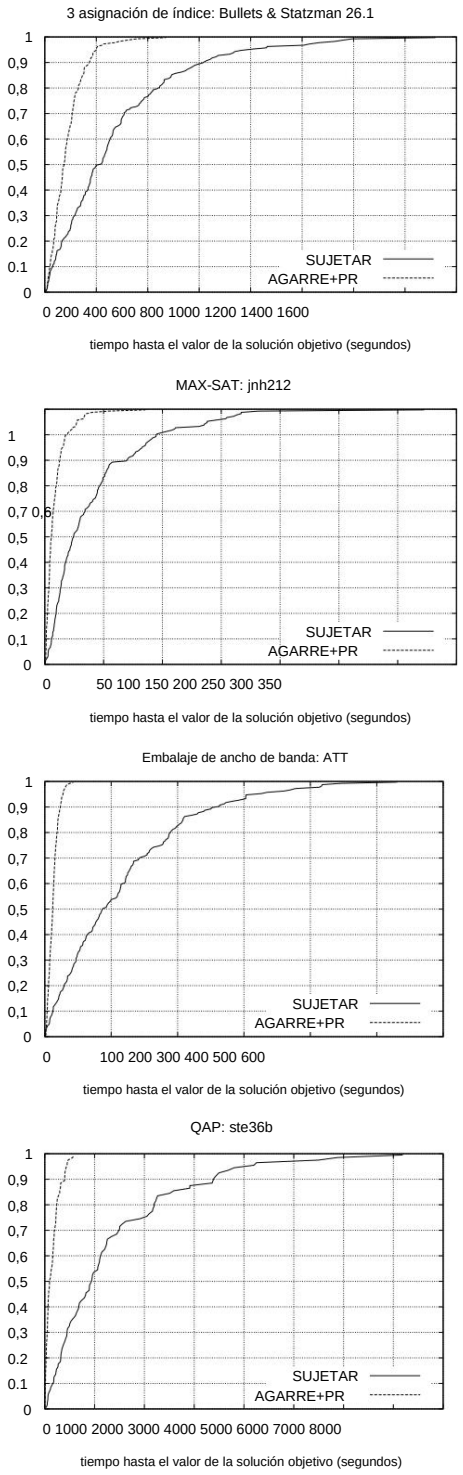
GRASP con vinculación de rutas hace uso de un conjunto de élite para recopilar un grupo diverso de soluciones de alta calidad encontradas durante la búsqueda. Este grupo tiene un tamaño limitado, es decir, puede tener como máximo soluciones Max Elite. Se han propuesto varios esquemas para la implementación de la reconexión de rutas, que se pueden aplicar como:

- una estrategia de intensificación, entre cada óptimo local obtenido después del local fase de búsqueda y una o más soluciones de élite; • el
paso posterior a la optimización, entre cada par de soluciones élite; • una estrategia
de intensificación, periódicamente (después de un número fijo de iteraciones GRASP desde la última fase de intensificación) sometiendo el conjunto de soluciones de élite a un proceso evolutivo (ver Subsección 4.7);
- la fase posterior a la optimización, sometiendo el conjunto de soluciones de élite a una evolución proceso ario; o
- cualquier otra combinación de los esquemas anteriores.

El grupo de soluciones de élite está inicialmente vacío. Cada solución localmente óptima obtenida por búsqueda local y cada solución resultante de la reconexión de rutas se considera candidata para ser insertada en el grupo. Si el grupo aún no está lleno, el candidato simplemente se agrega al grupo. De lo contrario, si el candidato es mejor que el titular (la mejor solución encontrada hasta el momento), reemplaza un elemento del grupo. En caso de que el candidato sea mejor que el peor elemento del grupo pero no mejor que el mejor elemento, entonces reemplaza algún elemento del grupo si es lo suficientemente diferente de cualquier otra solución actualmente en el grupo. Para equilibrar el impacto en la calidad y diversidad de la piscina, el elemento seleccionado para ser reemplazado es el que es más similar a la solución de entrada entre aquellas soluciones de élite de calidad no mejor que la solución de entrada [221].

Dado un óptimo local s1 producido al final de una iteración GRASP, necesitamos seleccionar al azar una solución s2 del grupo para aplicar la vinculación de rutas entre s1 y s2. En principio, se podría seleccionar cualquier solución de piscina. Sin embargo, es posible que deseemos evitar soluciones de agrupación que sean demasiado similares a s1, porque volver a vincular dos soluciones que son similares limita el alcance de la búsqueda de vinculación de rutas. Si las soluciones están representadas por vectores indicadores 0-1, deberíamos buscar pares de soluciones que estén lejos entre sí, en función de su distancia de Hamming (es decir, la cantidad de componentes que toman valores diferentes en cada solución). Una estrategia introducida en Resende y Werneck [221] es seleccionar un elemento de la piscina s2 al azar con una probabilidad proporcional a la distancia de Hamming entre el elemento de la piscina y el óptimo local s1. Dado que el número de caminos entre dos soluciones crece exponencialmente con su distancia de Hamming, esta estrategia favorece los elementos del grupo con un gran número de caminos que los conectan hacia y desde s1.

Después de determinar qué solución (s1 o s2) se designará como la solución inicial i y cuál será la solución guía g, el algoritmo comienza calculando el conjunto ÿ(i,g) de componentes en los que i y g difieren. Este conjunto corresponde a los movimientos



Higo. 10 Gráficas de tiempo para alcanzar el objetivo que comparan los tiempos de ejecución de GRASP puro y GRASP con la vinculación de rutas en cuatro instancias de distintos tipos de problemas: asignación de tres índices, satisfacción máxima, empaquetamiento de ancho de banda y asignación cuadrática.

que debe aplicarse a i para llegar a g. A partir de la solución inicial, la mejor
mover en ÿ(i,g) aún no realizado se aplica a la solución actual, hasta que se alcance la solución de
guía. Por mejor movimiento, nos referimos al que da como resultado el mayor
solución de calidad en el barrio restringido. La mejor solución encontrada a lo largo de este
es la trayectoria sometida a la búsqueda local y devuelta como la solución producida por el
Algoritmo de reenlace de caminos.

```
procedimiento GRASP+PR(Semilla);
1     Conjunto de soluciones de élite E ÿ ÿ;
dos   Establecer el mejor valor de solución f* ÿ ÿ;
3     mientras que el criterio de parada no se cumple
4         Solución ÿ Construcción aleatoria codiciosa (semilla);
5         si la solución no es factible entonces
6             Solución ÿ Reparación(Solución);
          terminara si;
          Solución ÿ Búsqueda local (Solución);
          si |E | > 0 entonces
7             Seleccione una solución de élite Solución' al azar de E ;
8             Solución ÿ PR(Solución,Solución0 );
9         terminara si;
10        si f(Solución) < f* entonces
11 12 13 14 Mejor solución ÿ Solución;
15            f* ÿ f(S);
16 final si;
17 Actualice el conjunto de soluciones de élite E con Solución;
18 fin-mientras;
19 devuelve la mejor solución;
Fin AGARRE+PR.
```

Higo. 11 Pseudo-código de una plantilla de GRASP con reenlace de ruta para un problema de minimización.

El pseudocódigo que se muestra en la Figura 11 resume los pasos de un GRASP con
Reconexión de caminos para un problema de minimización. El pseudocódigo sigue la estructura
del algoritmo GRASP básico en la Figura 1. Las líneas 1 y 2 inicializan el grupo de élite
soluciones y el mejor valor de solución, respectivamente. Path-relinking se realiza en
línea 11 entre la solución Solución obtenida al final de la búsqueda local
fase (línea 8) y una solución Solution0 seleccionada al azar del grupo de élite
soluciones E (línea 10). El procedimiento PR(Solución,Solución0) podría hacer uso, por
ejemplo, de cualquier variante de una estrategia de reconexión de caminos pura o combinada. El mejor
solución global encontrada La mejor solución se devuelve en la línea 19 después de detener
se cumple el criterio.

Se han considerado y combinado varias alternativas en implementaciones recientes de reconexión
de rutas. Estos incluyen adelante, atrás, atrás y adelante, mixto,
truncado, codicioso, aleatorizado, adaptativo, evolutivo y de reconexión de caminos externos.
Todas estas alternativas, que se describen a continuación, implican compromisos entre el tiempo de
cálculo y la calidad de la solución.

4.1 Reenlace de ruta hacia adelante

En la reconexión de caminos directos, el óptimo local de GRASP se designa como la solución inicial y la solución común se convierte en la solución guía. Este es el esquema original propuesto por Laguna y Martˊÿ [147].

4.2 Reenlace de ruta hacia atrás

En la reconexión de caminos hacia atrás, la solución del conjunto se designa como la solución inicial y el óptimo local GRASP se convierte en la guía. Este esquema fue propuesto originalmente en Aiex et al. [7] y Resende y Ribeiro [217]. La principal ventaja de este enfoque sobre el reenlace directo proviene del hecho de que, en general, hay más soluciones de alta calidad cerca de los elementos del grupo que cerca de los óptimos locales de GRASP. El reenlace de ruta hacia atrás explora más a fondo el vecindario alrededor de la solución de grupo, mientras que el reenlace de ruta hacia adelante explora más el vecindario alrededor del óptimo local GRASP. Los experimentos en [7, 217] han demostrado que la vinculación de rutas hacia atrás generalmente supera a la vinculación de rutas hacia adelante.

4.3 Reenlace de ruta hacia adelante y hacia atrás

La vinculación de rutas hacia adelante y hacia atrás combina la vinculación de rutas hacia adelante y hacia atrás. Como se muestra en [7, 217], encuentra soluciones al menos tan buenas como la vinculación de ruta hacia adelante o la vinculación de ruta hacia atrás, pero a expensas de tardar aproximadamente el doble en ejecutarse. La razón por la que la vinculación de caminos hacia atrás y hacia adelante a menudo encuentra soluciones de mejor calidad que la simple vinculación de caminos hacia atrás o hacia adelante proviene del hecho de que explora minuciosamente las vecindades de ambas soluciones s1 y s2.

4.4 Enlace mixto de rutas

La vinculación de rutas mixtas comparte los beneficios de la vinculación de rutas hacia adelante y hacia atrás, es decir, explora a fondo ambos vecindarios, pero lo hace aproximadamente al mismo tiempo que la vinculación de rutas hacia adelante o hacia atrás sola. Esto se logra intercambiando los roles de las soluciones inicial y guía en cada paso del procedimiento de reconexión de caminos. Por lo tanto, se generan dos caminos, uno que comienza en s1 y el otro en s2.
Los caminos evolucionan y finalmente se encuentran en alguna solución a mitad de camino entre s1 y s2. El camino unido vuelve a vincular estas dos soluciones. Glover [111] sugirió la reconexión de rutas mixtas y fue implementada y probada por primera vez por Ribeiro y Rosseti [229], donde se demostró que supera a la reconexión de rutas hacia adelante, hacia atrás y hacia adelante y hacia atrás. La Figura 12 muestra una comparación de GRASP puro y cuatro variantes de

Reconexión de caminos: adelante, atrás, adelante y atrás, y mixto. Los gráficos de tiempo hasta el objetivo muestran que GRASP con reconexión de rutas mixtas tiene el mejor perfil de tiempo de ejecución entre las variantes comparadas.

4.5 Reenlace de ruta truncada

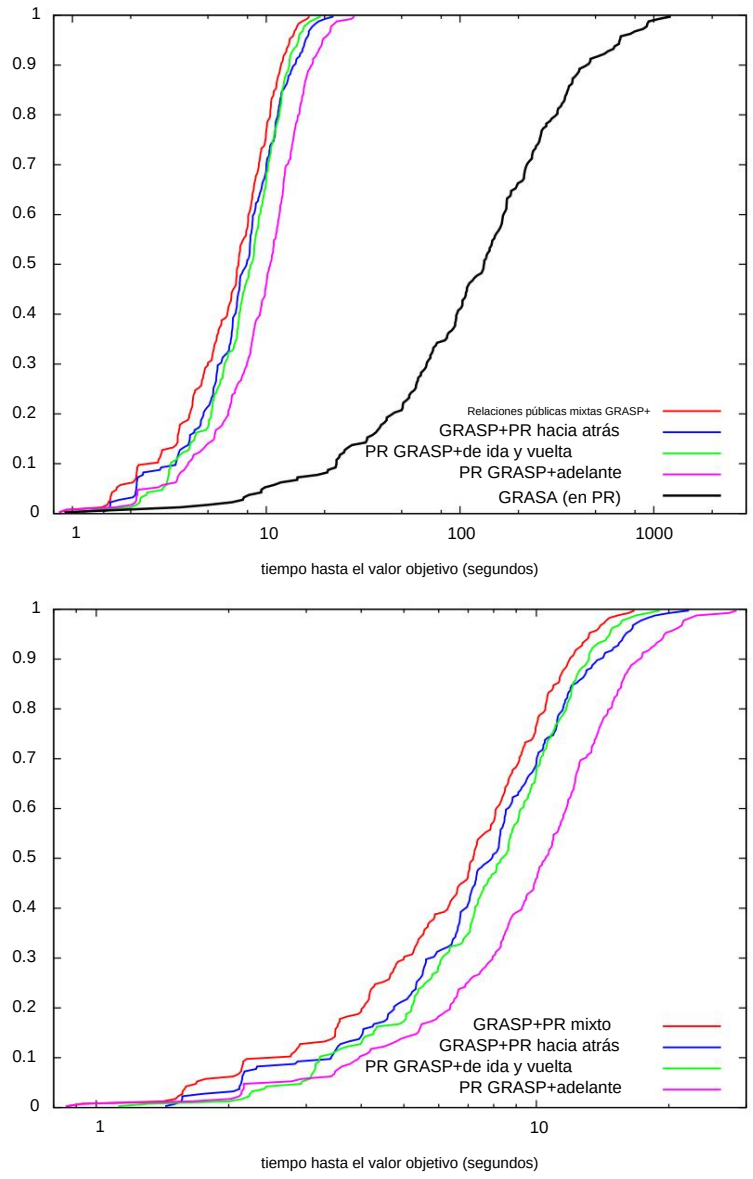
Dado que las soluciones de buena calidad tienden a estar cerca de otras soluciones de buena calidad, uno esperaría encontrar las mejores soluciones con reenlace de caminos cerca de la solución inicial o guía. De hecho, Resende et al. [211] mostró que este es el caso para instancias del problema de diversidad máximo-mínimo, como se muestra en la Figura 13. En ese experimento, se probó un esquema de reconexión de rutas hacia adelante y hacia atrás. La figura muestra el número promedio de las mejores soluciones encontradas mediante la vinculación de rutas tomadas en varias instancias y varias aplicaciones de la vinculación de rutas. El rango de 0 a 10 % en esta figura corresponde a subtrayectos cercanos a las soluciones iniciales para la fase de reconexión de trayectos directos, así como a la fase de retroceso, mientras que el rango de 90 a 100 % son subtrayectos cercanos a las soluciones guía. Como indica la figura, explorar los subtrayectos cerca de los extremos puede producir soluciones tan buenas como las que se encuentran explorando el trayecto completo. Hay una mayor concentración de mejores soluciones cerca de las soluciones iniciales exploradas por la vinculación de caminos.

La vinculación de ruta truncada se puede aplicar a la vinculación de ruta hacia adelante, hacia atrás, hacia atrás y hacia adelante, o mixta. En lugar de explorar la ruta completa, la vinculación de ruta truncada solo explora una fracción de la ruta y, en consecuencia, toma una fracción del tiempo para ejecutarse. La vinculación de rutas truncadas se ha aplicado en [22, 211].

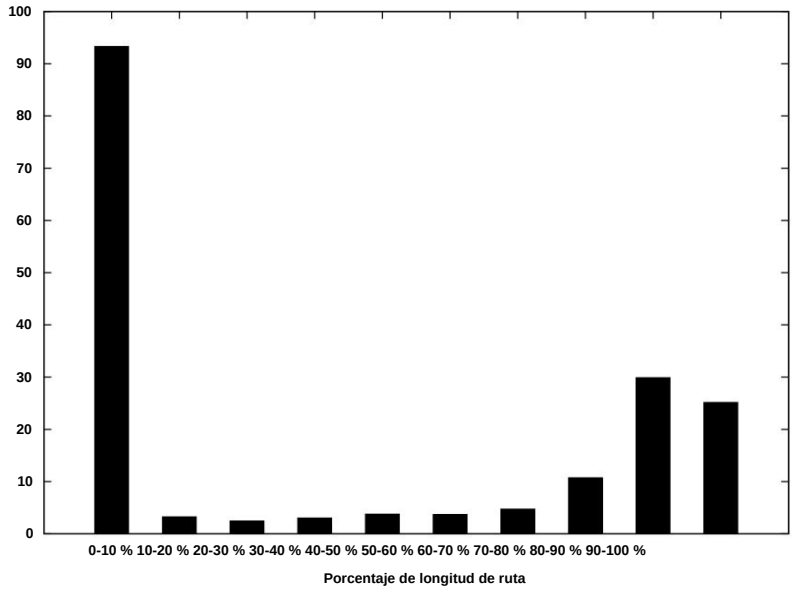
4.6 Reconexión de rutas adaptativas aleatorias codiciosos

En la vinculación de caminos, el movimiento mejor realizado hasta ahora en el conjunto ÿ(i,g) se aplica a la solución actual, hasta que se alcanza la solución guía. Si los empates se rompen de manera determinista, esta estrategia siempre producirá el mismo camino entre las soluciones inicial y guía. Dado que el número de caminos que conectan i y g es exponencial en |ÿ(i,g)|, explorar un solo camino puede ser algo limitante.

Reconexión de ruta adaptativa aleatoria codiciosa, introducida por Binato et al. [46], es una versión semi-codiciosa de la vinculación de rutas. En lugar de tomar la mejor jugada en ÿ(i,g) que aún no se ha realizado, se establece una lista de candidatas restringidas de buenas jugadas que aún no se han realizado y se aplica una jugada seleccionada aleatoriamente de esta última. Al aplicar esta estrategia en los tiempos entre las soluciones inicial y guía, se pueden explorar varios caminos. En [22, 85, 211].



Higo. 12 Gráficas de tiempo hasta el objetivo para GRASP puro y cuatro variantes de GRASP con vinculación de rutas (hacia delante, hacia atrás, hacia atrás y hacia adelante, y mixta) en una instancia del problema de diseño de red de 2 rutas.

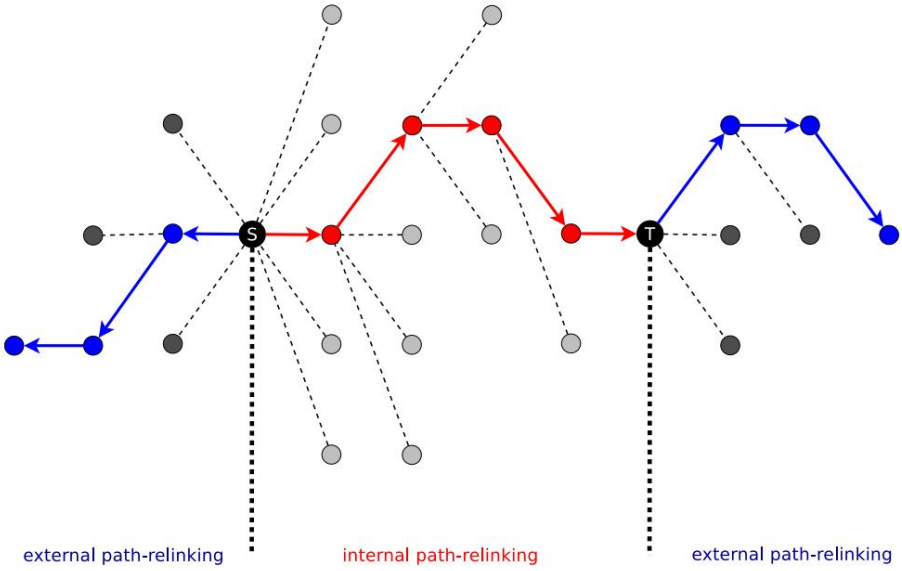


Higo. 13 Número promedio de mejores soluciones encontradas a diferentes profundidades del camino desde la solución inicial hasta la solución guía en instancias del problema de diversidad máximo-mínimo.

4.7 Reconexión de caminos evolutivos

GRASP con reconexión de rutas mantiene un grupo de soluciones de élite. La aplicación de la vinculación de rutas entre pares de soluciones de grupo puede dar como resultado un grupo de soluciones aún mejor. Aiex et al. [7] aplicó la vinculación de rutas entre todos los pares de soluciones de élite como un esquema de intensificación para mejorar la calidad del grupo y como un paso posterior a la optimización. La aplicación de la vinculación de caminos se repitió hasta que no fue posible ninguna mejora adicional.

Resende y Werneck [221, 222] describieron un esquema de reconexión de caminos evolutivos aplicado a pares de soluciones de élite y utilizado como un paso posterior a la optimización. El conjunto resultante de GRASP con iteraciones de reconexión de rutas se denomina población P0. En el paso k, todos los pares de soluciones de conjuntos de élite de la población Pk se vuelven a vincular y las soluciones resultantes se convierten en candidatas para su inclusión en la población Pk+1 de la próxima generación. Las mismas reglas para la aceptación en el grupo durante GRASP con reenlace de ruta se utilizan para la aceptación en Pk+1. Si la mejor solución en Pk+1 es mejor que la mejor en Pk , entonces se incrementa el número de generaciones. Resende et al. [211] describe otra forma de implementar la reconexión de caminos evolutiva, donde se mantiene una sola población. Cada par de soluciones de élite se vuelve a vincular y la solución resultante es candidata para ingresar al conjunto de élite. Si se acepta, reemplaza una solución de élite existente. El proceso continúa mientras aún quedan parejas de soluciones élite que aún no se han vuelto a vincular.



Higo. 14 Un camino interno (arcos rojos, nodos rojos) desde la solución S a la solución T y dos externos (arcos rojos, nodos rojos) arcos, nodos azules) caminos, uno que emana de la solución S y el otro de la solución T. Estos caminos son producidos por la reconexión de rutas internas y externas.

Andrade y Resende [21] utilizaron este esquema evolutivo como una intensificación procesar cada 100 iteraciones GRASP. Durante la fase de intensificación, cada solución del grupo se vuelve a vincular con las dos mejores. Dado que dos soluciones de élite pueden volver a vincularse más de una vez en distintas convocatorias al proceso de intensificación, codiciosos Se utilizó la reconexión de rutas adaptativas aleatorias.

Resende et al. [211] mostró que una variante de GRASP con reenlace de ruta evolutiva superó a varias otras heurísticas usando GRASP con reenlace de ruta, recocido simulado y búsqueda tabú para el problema de diversidad máximo-mínimo.

4.8 Reconexión y diversificación de rutas externas

Hasta ahora, en esta sección, hemos considerado variantes de vinculación de rutas en las que una ruta en el espacio de búsqueda el grafo conecta dos soluciones factibles introduciendo progresivamente en una de ellas (la solución inicial) atributos de la otra (la solución guía).
Dado que los atributos comunes a ambas soluciones no se modifican y todas las soluciones visitadas pertenecen a un camino entre las dos soluciones, también podemos referirnos a este tipo de solución. reenlace de ruta como reenlace interno de ruta.

La reconexión de caminos externos extiende cualquier camino que conecte dos soluciones factibles S y T más allá de sus extremos. Para extender tal camino más allá de S, los atributos no presentes en ya sea S o T se introducen en S. Simétricamente, para extenderlo más allá de T, los atributos

no presentes en S o T se introducen en T. En su variante voraz, se evalúan todos los movimientos y se elige como solución siguiente en el camino la de mejor coste o, en caso de que todas sean inviables, la de menor inviabilidad. En cualquier dirección, el procedimiento se detiene cuando se han probado todos los atributos que no aparecen en S o T para ampliar la ruta. Una vez completadas ambas rutas, se podrá aplicar la búsqueda local a la mejor solución en cada una de ellas. El mejor de los dos mínimos locales se devuelve como la solución producida por el procedimiento de reconexión de rutas externas.

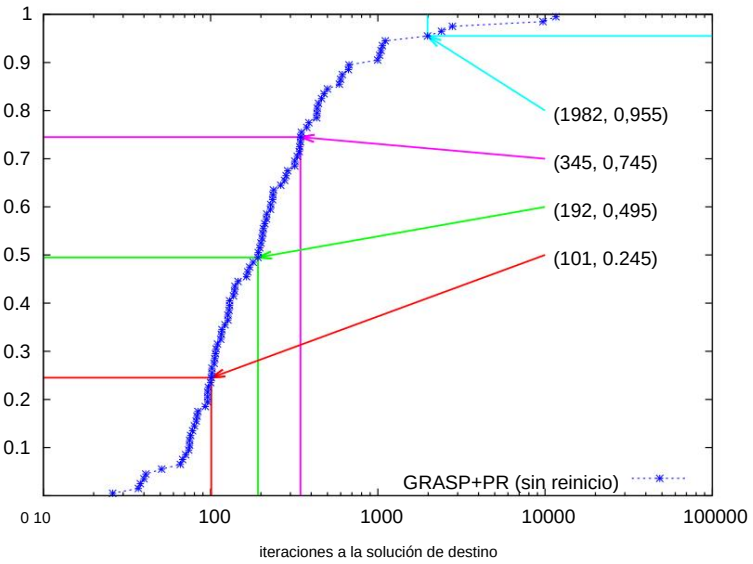
La Figura 14 ilustra la vinculación de rutas internas y externas. El camino con nodos y aristas en rojo es el resultado de la reconexión interna de caminos aplicada con S como solución inicial y T como solución guía. Observamos que la orientación que introducen los arcos en esta trayectoria se debe únicamente a la elección de las soluciones inicial y guía. Si se intercambiaran los roles de las soluciones S y T, podría haberse calculado y generado en la dirección inversa. La misma figura también ilustra dos caminos obtenidos por reconexión de caminos externos, uno que emana de S y el otro de T, ambos representados con nodos y bordes azules. Las orientaciones de los arcos en cada uno de estos caminos indican que necesariamente emanan de la solución S o T.

Para concluir, establecemos un paralelismo entre la reconexión de caminos internos y externos. Dado que la reconexión interna de caminos funciona fijando todos los atributos comunes a las soluciones inicial y guía y busca caminos entre ellos que satisfagan esta propiedad, es claramente una estrategia de intensificación. Por el contrario, la vinculación de caminos externos elimina progresivamente los atributos comunes y los reemplaza por otros que no aparecen ni en la solución inicial ni en la guía. Por tanto, puede verse como una estrategia de diversificación que produce soluciones cada vez más alejadas tanto de la solución inicial como de la guía. Por lo tanto, la vinculación de rutas externas se convierte en una herramienta para la diversificación de la búsqueda.

Glover [113] introdujo la reconexión de rutas externas y la aplicó por primera vez Duarte et al. [82] en una heurística para la minimización de la dispersión diferencial.

5 Estrategias de reinicio

La Figura 15 muestra una distribución típica de conteo de iteraciones para un GRASP con reenlace de ruta. Tenga en cuenta en este ejemplo que para la mayoría de las ejecuciones independientes cuyos recuentos de iteraciones componen la gráfica, el algoritmo encuentra una solución objetivo en relativamente pocas iteraciones: alrededor del 25% de las ejecuciones toman como máximo 101 iteraciones; alrededor del 50% toma como máximo 192 iteraciones; y alrededor del 75% toma como máximo 345. Sin embargo, algunas ejecuciones toman mucho más tiempo: el 10% toma más de 1000 iteraciones; 5% sobre 2000; y 2% sobre 9715 iteraciones. La ejecución más larga tomó 11607 iteraciones para encontrar la solución al menos tan buena como el objetivo. Estas largas colas contribuyen a un gran recuento de iteraciones promedio, así como a una alta desviación estándar. Esta sección propone estrategias para reducir la cola de la distribución y, en consecuencia, reducir el recuento de iteraciones promedio y su desviación estándar.



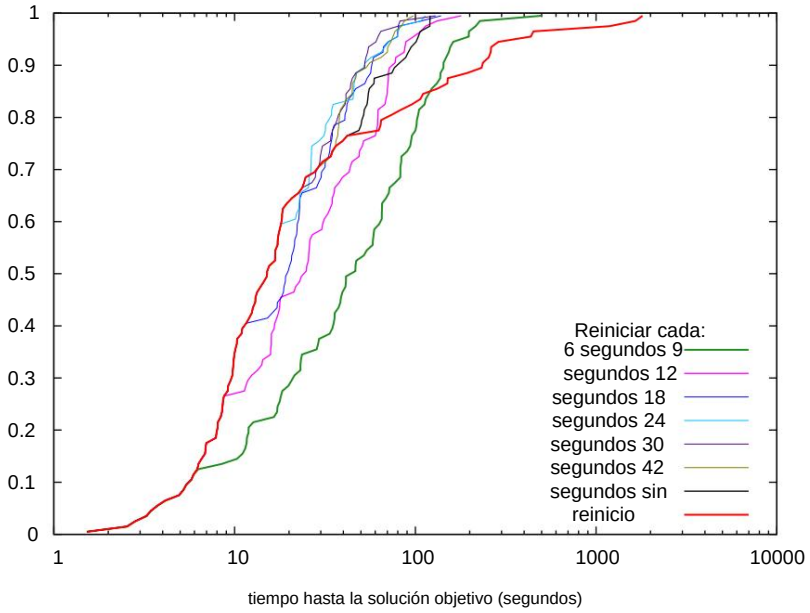
Higo. 15 Distribución típica del recuento de iteraciones de GRASP con reconexión de rutas.

Considere nuevamente la distribución en la Figura 15. La distribución muestra que cada ejecución tomará más de 345 iteraciones con una probabilidad de alrededor del 25%. Por lo tanto, cada vez que se reinicia el algoritmo, la probabilidad de que la nueva ejecución tome más de 345 iteraciones también es de alrededor del 25 %. Al reiniciar el algoritmo después de 345 iteraciones, la nueva ejecución requerirá más de 345 iteraciones con una probabilidad de alrededor del 25 %. Por lo tanto, la probabilidad de que el algoritmo se siga ejecutando después de 345+345 = 690 iteraciones es la probabilidad de que tome más de 345 iteraciones multiplicada por la probabilidad de que tome más de 690 iteraciones dado que tomó más de 345 iteraciones, es decir, aproximadamente $(1/4)\times(1/4)=(1/4)^2$. De igual manera, la probabilidad de que el algoritmo se siga ejecutando después de k períodos de 345 iteraciones es $1/(4^k)$. En este ejemplo, la probabilidad de que el algoritmo se ejecute después de 1725 iteraciones será de alrededor del 0,1 %, es decir, mucho menor que el 5 % de probabilidad de que el algoritmo realice más de 2000 iteraciones sin reiniciar.

Una estrategia de reinicio se define como una secuencia infinita de intervalos de tiempo ÿ1, ÿ2, ÿ3,... que define las épocas ÿ1, ÿ1 +ÿ2, ÿ1 +ÿ2 +ÿ3,... cuando el algoritmo se reinicia desde cero. Se puede demostrar que la estrategia óptima de reinicio usa ÿ1 = ÿ2 = ⋯ = ÿ donde ÿ, ÿ es una constante (desconocida). Luby, Sinclair y Zuckerman [156] propusieron por primera vez estrategias para acelerar los algoritmos de búsqueda estocástica local utilizando reinicios, donde demostraron la existencia de una estrategia de reinicio óptima. Las estrategias de reinicio en metaheurísticas se han abordado en [67, 138, 182, 187, 250]. Se pueden encontrar más trabajos sobre estrategias de reinicio en [251, 252].

Implementar la estrategia óptima puede ser difícil en la práctica porque requiere que los el valor constante ÿ · tiempos de ejecución varíen mucho para las diferentes combinaciones de algoritmo, instancia y calidad de la solución buscada. Dado que por lo general uno no tiene in-

información acerca de la distribución del tiempo de ejecución del algoritmo de búsqueda estocástica para el problema de optimización bajo consideración, se corre el riesgo de elegir ÿ ÿ un valor demasiado pequeño o demasiado grande. Por un lado, un valor demasiado pequeño puede hacer que la variante de reinicio del algoritmo tarde mucho más en converger que una variante sin reinicio. Por otro lado, un valor que es demasiado grande nunca puede conducir a un reinicio, lo que hace que la variante de reinicio del algoritmo tarde tanto en converger como la variante sin reinicio. La Figura 16 ilustra las estrategias de reinicio con gráficos de tiempo hasta el objetivo para la instancia de corte máximo G12 [125] en un gráfico de 800 nodos con una densidad de borde del 0,63 % con un valor de solución objetivo de 554 para ÿ = 6, 9, 12, 18, 24, 30 y 42 segundos. Para cada valor de ÿ, se realizaron 100 ejecuciones independientes de un GRASP con reenlace de ruta con reinicios. La variante con ÿ = ÿ corresponde a la heurística sin reinicio. La figura muestra que, para algunos valores de ÿ, la heurística resultante superó a su contraparte sin reinicio por un amplio margen.



Higo. 16 Gráfica de tiempo hasta el objetivo para el valor de la solución objetivo de 554 para un GRASP con enlace de ruta con reinicio en la instancia de corte máximo G12 usando diferentes valores de ÿ.

En GRASP con reconexión de rutas, el número de iteraciones entre las mejoras de la solución actual tiende a variar menos que los tiempos de ejecución para las diferentes combinaciones de calidad de instancia y solución buscadas. Si se tiene esto en cuenta, una estrategia de reinicio simple y efectiva para GRASP con reenlace de ruta es hacer un seguimiento de la última iteración cuando se mejoró la solución actual y reiniciar GRASP con reenlace de ruta si han pasado ÿ iteraciones sin mejorar. llamaremos

como reiniciar (ÿ). Un reinicio consiste en guardar el titular y vaciar
fuera del conjunto de élite.

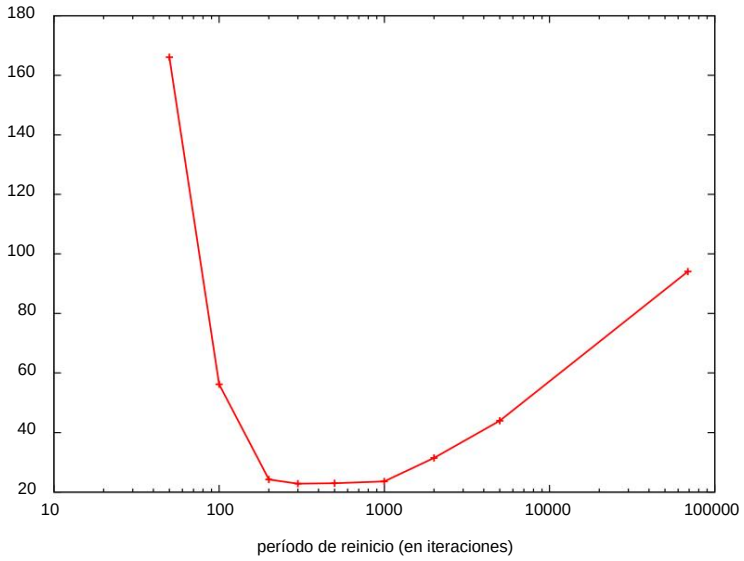
```
procedimiento GRASP+PR+Reinicio(Semilla);
1       Conjunto de soluciones de élite E ÿ ÿ;
dos     Establecer el mejor valor de solución f* ÿ ÿ;
3       ÚltimaImprov ÿ 0;
4 IterActual ÿ 0;
5       mientras que el criterio de parada no se cumple
6               IterActual ÿ IterActual+1;
7               Solución ÿ Construcción aleatoria codiciosa (semilla);
                si la solución no es factible entonces
                        Solución ÿ Reparación(Solución);
8               terminara si;
9               Solución ÿ Búsqueda local (Solución);
10              si |E | > 0 entonces
11                      Seleccione una solución de élite Solución' al azar de E ;
12                      Solución ÿ adelante-PR(Solución,Solución0 );
13              terminara si;
14              si f(Solución) < f*  entonces
15                      Mejor Solución ÿ Solución;
16                      F* ÿ f(S);
17                      ÚltimaImprov ÿ IteraciónActual;
18                      terminara si;
19              si CurrentIterÿLastImprov > ÿ entonces
20                      E ÿ ÿ;
21                      ÚltimaImprov ÿ IteraciónActual;
22              demás
23 24 25 Actualice el conjunto de soluciones de élite E con Solution;
26 final si;
27 fin-mientras;
28 devuelve la mejor solución;
Fin GRASP+PR+Reinicios.
```

Higo. 17 Pseudo-código de una plantilla de un GRASP con reenlace de ruta con reinicios para un problema de minimización.

El pseudocódigo que se muestra en la Figura 17 resume los pasos de un GRASP con Reconexión de rutas usando la estrategia de reinicio (ÿ) para un problema de minimización. El algoritmo realiza un seguimiento de la iteración actual (CurrentIter), así como de la última iteración cuando se encontró una solución de mejora (LastImprov). Si una mejora la solución se detecta en la línea 16, luego esta solución y su costo se guardan en las líneas 17 y 18, respectivamente, y la iteración de la última mejora se establece en la actual iteración en la línea 19. Si, en la línea 21, se determina que más de ÿ iteraciones han pasado desde la última mejora del titular, entonces se activa un reinicio, vaciando el conjunto élite en la línea 22 y restableciendo la iteración de la última mejora a la iteración actual en la línea 23. Si no se activa el reinicio, entonces en la línea 25 el la solución actual se prueba para su inclusión en el conjunto de élite y el conjunto se actualiza si es

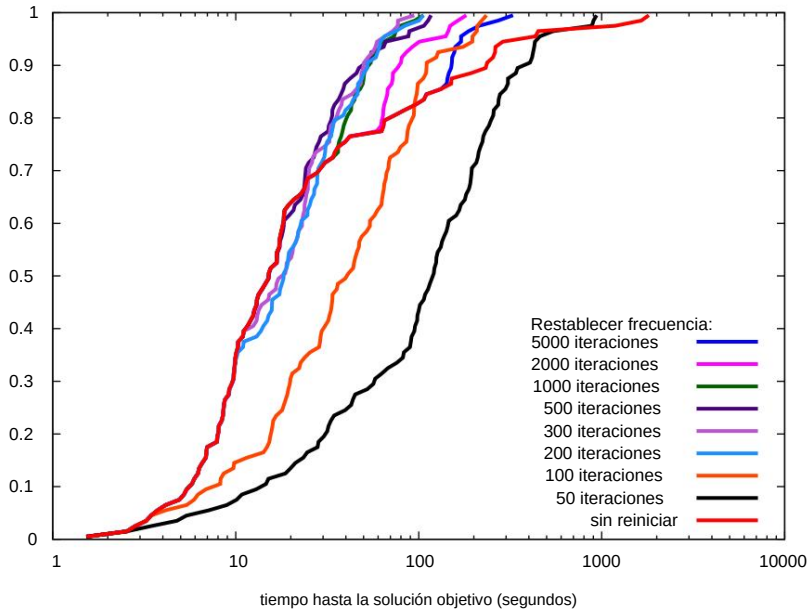
aceptado. La mejor solución general encontrada Best Solution se devuelve en la línea 28 después de que se satisface el criterio de parada.

Como ilustración del uso de la estrategia de reinicio (ÿ) dentro de un GRASP con enlace de ruta, considere la instancia de corte máximo G12. Para los valores ÿ = 50, 100, 200, 300, 500, 1000, 2000 y 5000, la heurística se ejecutó de forma independiente 100 veces y se detuvo cuando se encontró un corte de peso 554 o superior. También se implementó la estrategia sin reinicios. Las Figuras 18 y 19, así como la Tabla 6, resumen estas corridas, mostrando el tiempo promedio hasta la solución objetivo en función del valor de ÿ y las gráficas de tiempo hasta el objetivo para diferentes valores de ÿ. Estas cifras ilustran bien el efecto sobre el tiempo de ejecución de seleccionar un valor de ÿ que sea demasiado pequeño (ÿ = 50,100) o demasiado grande (ÿ = 2000,5000). Además, muestran que existe una amplia gama de valores de ÿ (ÿ = 200, 300, 500, 1000) que dan como resultado tiempos de ejecución más bajos en comparación con la estrategia sin reinicios.



Higo. 18 Tiempo promedio hasta la solución objetivo para la instancia de corte máximo G12 utilizando diferentes valores de ÿ. Todas las ejecuciones de todas las estrategias han encontrado una solución al menos tan buena como el valor objetivo de 554.

La Figura 20 ilustra adicionalmente el comportamiento de las estrategias de reinicio (100), reinicio (500) y reinicio (1000) para el ejemplo anterior, cuando se compara con la estrategia sin reinicios en la misma instancia de corte máximo G12. Sin embargo, en esta figura, para cada estrategia, trazamos el número de iteraciones al valor de la solución objetivo. Es interesante notar que, como se esperaba, cada estrategia de reinicio (ÿ) se comporta exactamente como la estrategia sin reinicios para las ÿ primeras iteraciones, para ÿ = 100,500,1000. Después de este punto, cada trayectoria se desvía de la estrategia sin reinicios. Entre estas estrategias, restart(500) es la de mejor rendimiento.



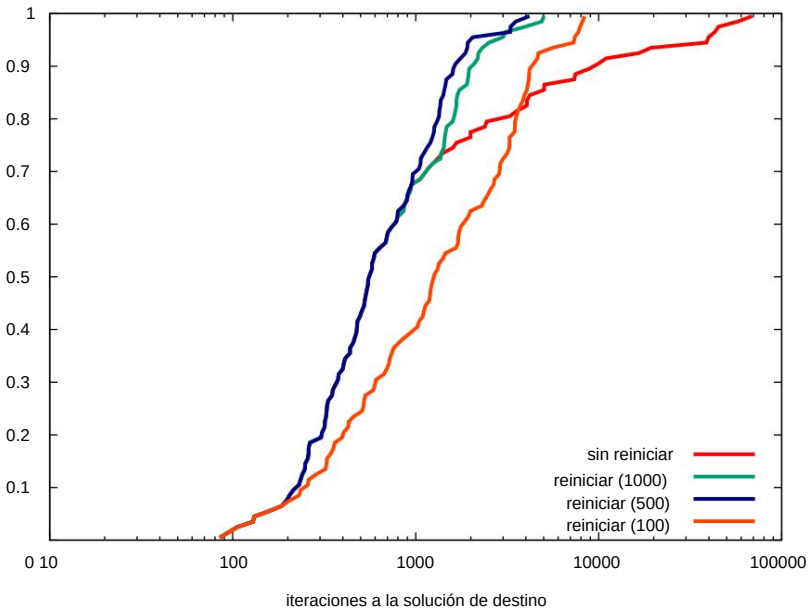
Higo. 19 Gráficas de tiempo hasta el objetivo para la instancia de corte máximo G12 usando diferentes valores de ÿ. La figura también muestra el gráfico de tiempo hasta el objetivo para la estrategia sin reinicios. Todas las ejecuciones de todas las estrategias encontradas la solución es al menos tan buena como el valor objetivo de 554.

Hacemos algunas observaciones finales sobre estos experimentos. El efecto de la Las estrategias de reinicio se pueden observar principalmente en la columna correspondiente a la cuarta cuartil de la Tabla 6. Las entradas en este cuartil corresponden a las de las colas pesadas de las distribuciones Las estrategias de reinicio en general no afectaron a los otros cuartiles. de las distribuciones, lo cual es una característica deseable. En comparación con el no reinicio estrategia, las estrategias de reinicio reiniciar (500) y reiniciar (1000) fueron capaces de reducir la número máximo de iteraciones, así como el promedio y la desviación estándar.

Tabla 6 Resumen de los resultados computacionales en la instancia de corte máximo G12 con cuatro estrategias. Para cada estrategia, se ejecutaron 100 ejecuciones independientes, cada una de las cuales se detuvo cuando una solución tan buena ya que se encontró el valor de la solución objetivo 554. Para cada estrategia, la tabla muestra la distribución de el número de iteraciones por cuartil. Para cada cuartil, la tabla da el número máximo de iteraciones tomadas por todas las ejecuciones en ese cuartil, es decir, el más lento del más rápido 25% (1°), 50% (2°), 75% (3º) y 100% (4º) de las carreras. El número medio de iteraciones sobre las 100 ejecuciones y el también se proporciona la desviación estándar (st.dev.) para cada estrategia.

	Iteraciones en cuartil					
Estrategia	1.º	2.º	3.º	4.º	Promedio	desv.
Sin reinicios	326	550	1596	68813	4525.1	11927.0
reiniciar (1000)	326	550	1423	5014	953.2	942.1
reiniciar(500)	326	550	1152	4178	835.0	746.1
reiniciar(100)	509	1243	3247	8382	2055.0	2005.9

La estrategia de reinicio (100) también lo hizo, pero no tanto como reiniciar (500) y reiniciar (1000). Las estrategias de reinicio restart(500) y restart(1000) fueron claramente las mejores estrategias de las probadas.



Higo. 20 Comparación de las gráficas de iteraciones al objetivo para la instancia de corte máximo G12 usando las estrategias de reinicio (100), reinicio (500) y reinicio (1000). La figura también muestra el gráfico de iteraciones hasta el objetivo para la estrategia sin reinicios. Todas las ejecuciones de todas las estrategias encontraron una solución al menos tan buena como el valor objetivo de 554.

La estrategia de reinicio (ÿ) para GRASP con reenlace de rutas discutida en esta sección fue propuesta originalmente por Resende y Ribeiro [219]. Además de los experimentos presentados en este capítulo para la instancia de corte máximo G12, ese documento también consideró otras cinco instancias de corte máximo, satisfacibilidad ponderada máxima y empaquetamiento de ancho de banda. Interian y Ribeiro [135] implementaron estrategias de reinicio para GRASP con reenlace de rutas para el problema del viajante de comercio de Steiner.

6 extensiones

En esta sección, comentamos algunas extensiones, estrategias de implementación e hibridaciones de GRASP.

Woodruff y Zemel [266] propusieron el uso de tablas hash para evitar los ciclos junto con la búsqueda tabú. Posteriormente, Ribeiro et al. exploraron un enfoque similar. [228] en su algoritmo de búsqueda tabú para la optimización de consultas en relación

bases de datos nacionales. En el contexto de las implementaciones de GRASP, las tablas hash fueron primero utilizado por Martins et al. [167] en su heurística de barrios múltiples para Steiner problema en gráficos, para evitar la aplicación de búsqueda local a soluciones ya visitadas en iteraciones anteriores.

Las estrategias de filtrado se utilizan para acelerar las iteraciones de GRASP, véase, por ejemplo, [92, 167, 202]. Con el filtrado, la búsqueda local no se aplica a todas las soluciones obtenidas en el final de la fase de construcción, pero solo a algunas soluciones no visitadas más prometedoras, definido por un umbral con respecto al titular.

Casi todo el esfuerzo de aleatorización en el algoritmo GRASP básico implica la fase de construcción. La búsqueda local se detiene en el primer óptimo local. Por otro lado, como VNS (Ⱳariable Neighborhood Search), propuesta por Hansen y estrategias Mladenovic [120, 172], se basan casi por completo en la aleatorización de la búsqueda local para escapar de los óptimos locales. Con respecto a la aleatorización, GRASP y variable estrategias de vecindad pueden considerarse complementarias y potencialmente capaces de conducir a métodos híbridos efectivos. El primer intento en esta dirección se hizo por Martins et al. [167] donde la fase de construcción de una heurística híbrida para el El problema de Steiner en gráficos sigue la estrategia aleatoria codiciosa de GRASP, mientras que la fase de búsqueda local hace uso de dos estructuras de vecindad diferentes, como la Procedimiento VND (descenso de vecindad variable) [120, 172]. Esa heurística fue posterior mejorado por Ribeiro et al. [234], donde uno de los componentes clave del nuevo algoritmo era otra estrategia para la exploración de diferentes barrios. arroyo y Souza [233] también combinaron GRASP con VND en una heurística híbrida para el Problema del árbol de expansión mínimo con restricciones de grado. Festa et al. [103] estudió diferentes variantes y combinaciones de GRASP y VNS para el problema de corte máximo, encontrando y mejorando las soluciones más conocidas para algunas instancias abiertas de la literatura.

GRASP también se ha utilizado junto con algoritmos genéticos. el codicioso Se aplica la estrategia aleatoria utilizada en la fase de construcción de una heurística GRASP. generar la población inicial para un algoritmo genético. Como ejemplo, considere el algoritmo genético de Ahuja et al. [5] para el problema de asignación cuadrática. eso hace uso de la heurística GRASP propuesta por Li et al. [150] para crear la inicial población de soluciones. Un enfoque similar fue utilizado por Armony et al. [31], con la población inicial compuesta tanto por soluciones generadas aleatoriamente como por aquellas construidas por la heurística GRASP.

La ƕibridación de GRASP con la búsqueda tabú fue estudiada por primera vez por Laguna y González-Velarde [146]. Delmaire et al. [70] consideró dos enfoques. En el primero, GRASP se aplica como una poderosa estrategia de diversificación en el contexto de un procedimiento de búsqueda tabú. El segundo enfoque es una implementación del algoritmo Reac had GRASP presentado en la Sección 3.2, en el que la fase de búsqueda local es fortalecido por la búsqueda tabú. Resultados informados para el problema de ubicación capacitada muestran que los enfoques híbridos funcionan mejor que los métodos aislados anteriormente utilizado. En [1] se proponen dos heurísticas de dos etapas para resolver el problema de diseño de instalaciones de varios pisos. GRASP/TS aplica GRASP para encontrar el diseño inicial y tabú buscar para refinarlo.

La búsqueda local iterada (ILS) construye iterativamente una secuencia de soluciones generadas por la aplicación repetida de la búsqueda local y la perturbación de los óptimos locales encontrados por la búsqueda local [42]. Lourenço et al. [154] señalan que ILS se ha redescubierto muchas veces y también se conoce como descenso iterado [40, 41], cadenas de Markov de gran paso [165], Lin-Kernighan iterado [136] y optimización local encadenada [164]. Se puede obtener un híbrido GRASP/ILS reemplazando la búsqueda local estándar de GRASP por ILS. La construcción GRASP produce una solución que se pasa al procedimiento ILS. Ribeiro y Urrutia [235] presentaron un GRASP híbrido con ILS para el problema del torneo itinerante en espejo, en el que las perturbaciones se logran generando aleatoriamente soluciones en el vecindario de la cadena de eyección de rotación del juego [109, 110].

7 Aplicaciones

La primera aplicación de GRASP se describió en la literatura en 1989 [90]. En ese documento, GRASP se aplicó a problemas de cobertura de conjuntos difíciles. Desde entonces, GRASP se ha aplicado a una amplia gama de problemas. Las principales áreas de aplicación se resumen a continuación con enlaces a referencias específicas:

• Problemas de asignación [5, 7, 89, 105, 150, 153, 155, 169, 177, 178, 183, 189, 170, 191, 198, 202, 204, 212, 241] •
Biología [23, 64, 68, 75, 96, 107, 236] • Visión
artificial [53, 131, 246, 247] • Recubrimiento,
empaquetado y partición [14] , 15, 16, 27, 30, 71, 76, 90, 108, 118, 192, 195, 196, 223, 239, 243, 245] •
Diversidad y dispersión [78, 82, 163, 211] • Finanzas
[19, 126] • Dibujo de gráficos y mapas [66, 95, 147,
159, 160, 162, 184, 215, 227] • Ubicación y disposición [1, 60, 66, 70, 119, 132,
140, 171, 181, 186, 253, 255, 260, 261]
• Lógica [74, 97, 190, 209, 213, 214] •
Árbol mínimo de Steiner [54, 166, 167, 168, 234] •
Optimización en grafos [2, 3, 4, 12, 32, 56, 72, 80 , 79, 92, 98, 99, 133, 145, 148, 157, 160, 161, 167, 179, 188, 193, 208, 210, 215, 227, 234, 248, 257] • Sistemas de alimentación [25, 48 , 49, 85, 203, 263] • Robótica [143, 240] • Enrutamiento [29, 33, 38, 52, 55, 63, 135, 142, 144, 152, 175, 181, 206, 262, 264, 265] • Ingeniería de software [158] • Deportes [26, 139, 225, 235] • Telecomunicaciones [2, 18, 17, 20, 22, 31, 61, 106, 140, 153, 174, 176, 194, 197, 199 , 202, 207, 208, 217, 226, 258] • Horarios, programación y fabricación [6, 11, 13, 20, 22, 24, 35, 36, 37, 39, 47, 50, 59, 62, 65 , 69, 73, 77, 84, 86, 87, 88, 93, 94, 137, 141, 146, 149, 151, 173, 180, 185, 205, 235, 237, 238, 242, 244, 267, 268]

• Transporte [29, 34, 86, 89, 256]
• Diseño VLSI [27, 28]

Se remite al lector a Festa y Resende [102] y al libro de Resende y Ribeiro [220] para bibliografías comentadas extendidas de aplicaciones GRASP.

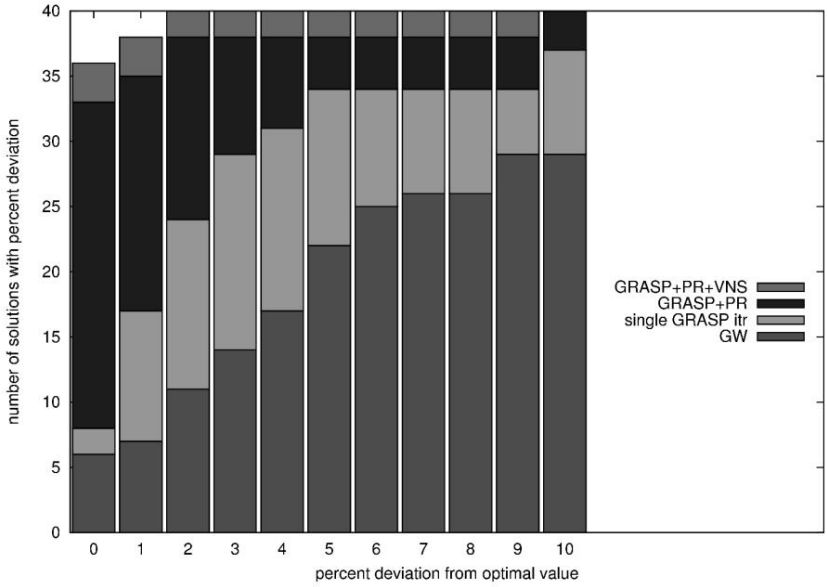
8 Observaciones finales

Los resultados descritos en este capítulo reflejan aplicaciones exitosas de GRASP a un gran número de problemas clásicos de optimización combinatoria, así como a problemas que surgen en situaciones del mundo real en diferentes áreas de los negocios, la ciencia y la tecnología.

Destacamos la sencillez de implementación de GRASP, que hace uso de bloques de construcción simples (procedimientos de construcción de soluciones y métodos de búsqueda local) que a menudo están fácilmente disponibles. Al contrario de lo que ocurre con la mayoría de las metaheurísticas, como la búsqueda tabú o los algoritmos genéticos, que hacen uso de un gran número de parámetros en sus implementaciones, la variante básica de GRASP requiere la justificación de un solo parámetro, es decir, el parámetro ÿ de la lista restringida de candidatos (RCL).

Desarrollos recientes, presentados en este capítulo, muestran que diferentes extensiones de el procedimiento básico permite mejoras adicionales en las soluciones encontradas por GRASP. Entre estos, destacamos GRASP reactivo, que automatiza el ajuste de el parámetro de lista restringida de candidatos; vecindades variables, lo que permite una búsqueda local acelerada e intensificada; reconexión de caminos, que más allá de permitir la implementación de estrategias de intensificación basadas en la memoria de soluciones de élite, abre el camino para el desarrollo de estrategias cooperativas paralelas muy efectivas [6, 7, 8, 229]; y reiniciar estrategias para agilizar la búsqueda.

Estas y otras extensiones conforman un conjunto de herramientas que se pueden agregar a heurísticas simples para encontrar soluciones de mejor calidad. Para ilustrar el efecto de extensiones sobre la calidad de la solución, la Figura 21 muestra algunos resultados obtenidos para el problema del árbol de Steiner de recolección de premios (PCSTP), según lo discutido por Canuto et al. en [54]. Él La figura muestra los resultados para once niveles diferentes de precisión de la solución (que varían de óptimo al diez por ciento del óptimo) en 40 instancias de PCSTP. Para cada nivel de precisión de la solución, la figura muestra el número de instancias para las cuales cada componente encontró soluciones dentro del nivel de precisión. Los componentes son el primal-dual algoritmo constructivo (GW) de Goemans y Williamson [116], GW seguido de búsqueda local (GW+LS), correspondiente a la primera iteración GRASP, 500 iteraciones de GRASP con path-relinking (GRASP+PR), y el algoritmo completo, utilizando búsqueda de vecindad variable como procedimiento de post-optimización (GRASP+PR+VNS). Observamos que el número de soluciones óptimas encontradas va de seis, utilizando sólo el algoritmo constructivo, hasta un total de 36, utilizando el algoritmo completo descrito en [54]. La mayor desviación relativa con respecto al valor óptimo disminuye del 36,4% en el primer caso, a sólo el 1,1% para el algoritmo completo. Es fácil observe el aporte realizado por cada prórroga adicional.



Higo. 21 Desempeño del algoritmo de aproximación GW, una sola iteración GRASP (GW seguido por búsqueda local), 500 iteraciones de GRASP con enlace de ruta y 500 iteraciones de GRASP con Reenlace de ruta seguido de VNS para problemas de árbol de Steiner de recolección de premios de la serie C.

La estructura de GRASP lo hace muy susceptible de ser directo y eficiente. implementaciones paralelas que se benefician de la arquitectura de la computadora. Las implementaciones paralelas de GRASP [6, 7, 8, 229] son bastante robustas y conducen a aceleraciones lineales tanto en estrategias independientes como cooperativas. Las estrategias cooperativas se basan en la colaboración entre procesadores a través de la reconexión de rutas y un grupo global de soluciones de élite. Esto permite el uso de más procesadores para encontrar mejores soluciones en menos tiempo de cómputo. Muchas implementaciones paralelas de GRASP han sido reportadas en la literatura, ver por ejemplo [167, 168, 178, 189, 190]. En muchos de estos documentos, el común se hizo la observación: las aceleraciones en los tiempos de ejecución medidos fueron proporcionales al número de procesadores. Esta observación se puede explicar si el azar la variable tiempo hasta el valor de la solución objetivo se distribuye exponencialmente. Aiex, Resende y Ribeiro [9] desarrollaron una metodología gráfica basada en distribuciones de tiempo de ejecución para mostrar empíricamente que los tiempos de ejecución de las heurísticas GRASP se ajustan a distribuciones exponenciales, como se resume a continuación.

Las distribuciones de tiempo de ejecución o los gráficos de tiempo hasta el objetivo muestran en el eje de ordenadas el probabilidad de que un algoritmo encuentre una solución al menos tan buena como un valor objetivo dado dentro de un tiempo de ejecución dado, que se muestra en el eje de abscisas. Ellos proporcionan el herramienta muy útil para caracterizar los tiempos de ejecución de algoritmos de optimización estotica para problemas combinatorios y para comparar diferentes algoritmos o estrategias para la solución de un problema dado. Las gráficas de tiempo hasta el objetivo fueron utilizadas por primera vez por Feo, Resende y Smith [92] y han sido ampliamente utilizados como una herramienta para el diseño de algoritmos y com-

parisón Las distribuciones de tiempo de ejecución también han sido defendidas por Hoos y Stutzle [134] como una forma de caracterizar los tiempos de ejecución de los algoritmos de búsqueda estocásticos locales para la optimización combinatoria. En particular, se han aplicado en gran medida para evaluar y comparar la eficiencia de diferentes estrategias de implementaciones secuenciales y paralelas de GRASP con (y sin) heurísticas de reconexión de rutas. Aiex, Resende y Ribeiro [9] usaron gráficos de tiempo hasta el objetivo para mostrar experimentalmente que los tiempos de ejecución de las heurísticas GRASP se ajustan a distribuciones exponenciales desplazadas (o de dos parámetros), informando resultados computacionales para 2400 ejecuciones de heurísticas GRASP para cada uno de cinco diferentes problemas: conjunto máximo estable, asignación cuadrática, planarización de grafos [215, 216, 227], máxima satisfacibilidad ponderada y máxima cobertura. Aiex, Resende y Ribeiro [10] desarrollaron un programa Perl para crear gráficas de tiempo hasta el objetivo para los tiempos medidos que se supone que se ajustan a una distribución exponencial desplazada, siguiendo de cerca el trabajo de [9]. Ribeiro, Rosseti y Vallejos [231] desarrollaron un resultado de forma cerrada para comparar dos algoritmos exponenciales y un procedimiento iterativo para comparar dos algoritmos siguiendo distribuciones de tiempo de ejecución genéricas. Este trabajo fue ampliado por Ribeiro, Rosseti y Vallejos [232] y también fue aplicado en la comparación de heurísticas paralelas. Ribeiro y Rosseti [230] desarrollaron un código para comparar distribuciones de tiempo de ejecución de algoritmos aleatorios.

Este capítulo proporciona al lector las herramientas para construir un GRASP básico para encontrar soluciones óptimas o casi óptimas a un problema de optimización combinatoria. El capítulo también proporciona los medios para agregar a este GRASP básico funciones más avanzadas, como estrategias de reinicio y vinculación de rutas, que permiten un mejor rendimiento, tanto con respecto a la calidad de la solución como al tiempo de ejecución de la solución. Queda fuera de este capítulo el uso de GRASP para resolver problemas de optimización continua. El lector interesado puede consultar [127, 128, 129, 130, 220, 254] para obtener una introducción a C-GRASP, o Continuous GRASP, así como a algunos programas y aplicaciones de C-GRASP.

Referencias

1. S. Abdinnour-Helm y SW Hadley. Heurística basada en la búsqueda tabú para el diseño de instalaciones de varios pisos. International J. of Production Research, 38:365-383, 2000.
2. J. Abello, PM Pardalos y MGC Resende. Sobre problemas de clic máximo en gráficos muy grandes. En J. Abello y J. Vitter, editores, External Memory Algorithms and Visualization, volumen 50 de DIMACS Series on Discrete Mathematics and Theoretical Computer Science, páginas 199–130. Sociedad Matemática Estadounidense, 1999.
3. J. Abello, MGC Resende y S. Sudarsky. Detección masiva de cuasi-clic. En S. Rajsbaum, editor, LATIN 2002: Theoretical Informatics, volumen 2286 de Lecture Notes in Computer Science, páginas 598–612. Springer-Verlag, 2002.
4. RK Ahuja, JB Orlin y D. Sharma. Estructuras de vecindad de intercambio múltiple para el problema del árbol de expansión mínimo capacitado. Programación matemática, 91:71–97, 2001.
5. RK Ahuja, JB Orlin y A. Tiwari. Un algoritmo genético codicioso para la asignación cuadrática problema mental. Computers and Operations Research, 27:917–934, 2000.
6. RM Aiex, S. Binato y MGC Resende. GRASP paralelo con reenlace de ruta para el trabajo programación de tiendas. Computación paralela, 29:393–430, 2003.
7. RM Aiex, PM Pardalos, MGC Resende y G. Toraldo. GRASP con reenlace de ruta para asignación de tres índices. INFORMA J. sobre Informática, 17:224–247, 2005.

28. SM Areibi. GRASP: Una técnica constructiva eficaz para la partición de circuitos VLSI. En Proc. Conferencia canadiense IEEE sobre ingeniería eléctrica e informática, páginas 462–467, Edmonton, Canadá, 1999.

29. MF Argüello, JF Bard y G. Yu. Un GRASP para el enrutamiento de aeronaves en respuesta a encallamientos y retrasos. J. de Optimización Combinatoria, 1:211-228, 1997.

30. MF Argüello, TA Feo y O. Goldschmidt. Métodos aleatorios para el problema de partición de números. Computadoras e investigación de operaciones, 23:103–111, 1996.

31. M. Armony, JC Klincewicz, H. Luss y MB Rosenwein. Diseño de autorreparación apilada anillos utilizando un algoritmo genético. J. of Heuristics, 6:85–105, 2000.

32. JEC Arroyo, PS Vieira y DS Vianna. Un algoritmo GRASP para el problema del árbol de expansión mínimo multicriterio. Annals of Operations Research, 159:125–133, 2008.

33. JB Atkinson. Una heurística de búsqueda aleatoria codiciosa para la programación de vehículos con limitaciones de tiempo y la incorporación de una estrategia de aprendizaje. J. de la Sociedad de Investigación Operacional, 49:700-708, 1998.

34. JF Bardo. Un análisis de un área de descarga de vagones de ferrocarril para un fabricante de productos de consumo. J. de la Sociedad de Investigación Operacional, 48:873-883, 1997.

35. JF Bard y TA Feo. Secuenciación de operaciones en la fabricación de piezas discretas. Ciencia administrativa, 35:249-255, 1989.

36. JF Bard y TA Feo. Un algoritmo para el problema de selección de equipos de fabricación. Transacciones IIE, 23:83–92, 1991.

37. JF Bard, TA Feo y S. Holland. Un GRASP para programar el montaje de la placa de circuito impreso. Transacciones del IIE, 28:155–165, 1996.

38. JF Bard, L. Huang, P. Jaillet y M. Dror. Un enfoque de descomposición para el problema de enrutamiento de inventario con instalaciones satelitales. Ciencias del Transporte, 32:189-203, 1998.

39. JF Bard, Y. Shao y Al Jarrah. Un GRASP secuencial para el enrutamiento y el horario del terapeuta problema de ing. J. of Scheduling, 17:109–133, 2014.

40. EB Baum. Descenso iterado: un mejor algoritmo para la búsqueda local en problemas de optimización combinatoria. Informe técnico, Instituto de Tecnología de California, 1986.

41. EB Baum. Hacia la computación 'neuronal' práctica para problemas de optimización combinatoria. En AIP Conference Proceedings 151 on Neural Networks for Computing, páginas 53–58, Woodbury, 1987. American Institute of Physics Inc.

42. J. Baxter. Evasión local óptima en la ubicación del depósito. J. de la Sociedad de Investigación Operativa, 32:815-819, 1981.

43. JE Beasley. Un algoritmo para problemas de cobertura de conjuntos. J. Europea de Investigación Operativa, 31:85–93, 1987.

44. JE Beasley. Una heurística lagrangiana para problemas de cobertura de conjuntos. Logística de investigación naval, 37: 151–164, 1990.

45. JE Beasley. Relajación lagrangiana. En CR Reeves, editor, Técnicas heurísticas modernas para problemas combinatorios, páginas 243–303. Publicaciones científicas de Blackwell, Oxford, 1993.

46. S. Binato, H. Faria Jr. y MGC Resende. Vuelva a vincular la ruta adaptativa aleatoria codiciosa. En JP Sousa, editor, Proceedings of the IV Metaheuristics International Conference, páginas 393–397, 2001.

47. S. Binato, WJ Hery, D. Loewenstern y MGC Resende. Un GRASP para la programación de talleres. En CC Ribeiro y P. Hansen, editores, Essays and Surveys in Metaheuristics, páginas 59–79. Editores académicos de Kluwer, 2002.

48. S. Binato y GC Oliveira. Un GRASP reactivo para la planificación de la expansión de la red de transmisión. En CC Ribeiro y P. Hansen, editores, Essays and Surveys in Metaheuristics, páginas 81–100. Editores académicos de Kluwer, 2002.

49. S. Binato, GC Oliveira y JL Araujo. Un procedimiento de búsqueda adaptativo aleatorio codicioso para la planificación de la expansión de la transmisión. IEEE Transactions on Power Systems, 16:247–253, 2001.

50. M. Boudia, MAO Louly y C. Prins. Un GRASP reactivo y la reconexión de rutas para un problema combinado de producción y distribución. Computadoras e investigación de operaciones, 34:3402–3419, 2007.

51. JL Bresina. Muestreo estocástico con sesgo heurístico. En Actas del Decimotercer Tribunal Nacional Conferencia sobre Inteligencia Artificial, páginas 271–278, Portland, 1996.
52. AM Campbell y BW Thomas. Problema probabilístico del viajante de comercio con plazos. Ciencia del transporte, 42: 1–21, 2008.
53. RG Cano, G. Kunigami, CC de Souza y PJ de Rezende. Una heurística GRASP híbrida para construir dibujos efectivos de mapas de símbolos proporcionales. Computadoras e investigación de operaciones, 40:1435–1447, 2013.
54. SA Canuto, MGC Resende y CC Ribeiro. Búsqueda local con perturbaciones para el problema del árbol de Steiner de recolección de premios en gráficos. Redes, 38:50–58, 2001.
55. C. Carreto y B. Baker. Un enfoque interactivo GRASP para el problema de enrutamiento de vehículos con backhauls. En CC Ribeiro y P. Hansen, editores, Essays and Surveys in Metaheuristics, páginas 185–199. Editores académicos de Kluwer, 2002.
56. WA ChaovaliTWongse, CAS Oliveira, B. Chiarini, PM Pardalos y MGC Resende. GRASP revisado con reenlace de rutas para el problema de ordenación lineal. J. de Optimización Combinatoria, 22:572–593, 2011.
57. I. Caronte y O. Hudry. El método Noising: un nuevo método para la optimización combinatoria. Cartas de investigación de operaciones, 14:133-137, 1993.
58. I. Caronte y O. Hudry. Los métodos ruidosos: una encuesta. En CC Ribeiro y P. Hansen, editores, Essays and Surveys in Metaheuristics, páginas 245–261. Editores académicos de Kluwer, 2002.
59. M. Chica, O. Cordón, S. Damas y J. Bautista. Un GRASP multiobjetivo para la variante 1/3 del problema de equilibrio de la línea de montaje de tiempo y espacio. En N. Garc´a-Pedrajas, F. Herrera, C. Fyfe, J. Ben´ytez y M. Ali, editores, Trends in Applied Intelligent Systems, volumen 6098 de Lecture Notes in Computer Science, páginas 656–665. Springer, Berlín, 2010.
60. R. Colomé y D. Serra. Elección del consumidor en modelos de ubicación competitivos: formulaciones y heurísticas. Artículos en ciencia regional, 80: 439–464, 2001.
61. C. Comandante, CAS Oliveira, PM Pardalos y MGC Resende. Una heurística GRASP para el problema de la comunicación cooperativa en redes ad hoc. En Actas de la VI Conferencia Internacional de Metaheurísticas, páginas 225–330, 2005.
62. Comandante de CW, SI Butenko, PM Pardalos y CAS Oliveira. GRASP reactivo con enlace de ruta para el problema de programación de transmisiones. En Actas de la 40.ª Conferencia Internacional Anual de Telemetría, páginas 792–800, 2004.
63. A. Corberan, R. Mart´y, and JM Sanch´ys. Una heurística GRASP para el cartero chino mixto problema. European J. of Operational Research, 142:70–80, 2002.
64. R. Cordone y G. Lulli. Una metaheurística GRASP para el análisis de datos de micromatrices. ordenadores & Operations Research, 40:3108–3120, 2013.
65. JF Correcher, MT Alonso, F. Parreno y R. Alvarez-Valdes. Resolviendo un gran problema de carga de contenedores múltiples en la industria de fabricación de automóviles. Computadoras e investigación de operaciones, 82: 139–152, 2017.
66. GL Cravo, GM Ribeiro y LA Nogueira Lorena. Un procedimiento de búsqueda adaptativo aleatorio codicioso para la colocación de etiquetas cartográficas de características puntuales. Informática y Geociencias, 34:373–386, 2008.
67. MM D'Apuzzo, A. Migdalas, PM Pardalos y G. Toraldo. Computación paralela en optimización global. En E. Kontoghiorghes, editor, Handbook of Parallel Computing and Statistics. Chapman & Hall / CRC, Boca Ratón, 2006.
68. S. Das y SM Idicula. Aplicación de GRASP reactivo a la biagrupación de datos de expresión génica. En Actas del Simposio Internacional sobre Biocomputación, página 14, Calicut, 2010. ACM.
69. P. De, JB Ghosj y CE Wells. Resolver un modelo generalizado para la asignación y secuenciación de fechas de vencimiento. International J. of Production Economics, 34:179-185, 1994.
70. H. Delmaire, JA D´yaz, E. Fernandez, and M. Ortega. Heurísticas basadas en GRASP reactivo y Tabu Search para el problema de ubicación de planta capacitada de fuente única. INFOR, 37:194-225, 1999.

71. X. Delorme, X. Gandibleux y F. Degoutin. Procedimientos evolutivos, constructivos e híbridos para el problema del empaquetamiento de conjuntos biobjetivo. J. Europea de Investigación Operativa, 204:206–217, 2010.
72. Y. Deng y JF Bard. Un GRASP reactivo con reconexión de rutas para agrupamiento capacitado. j de Heurística, 17:119–152, 2011.
73. Y. Deng, JF Bard, GR Chacon y J. Stuber. Programación de operaciones back-end en la fabricación de semiconductores. Transacciones IEEE sobre fabricación de semiconductores, 23:210–220, 2010.
74. AS Deshpande y E. Triantaphyllou. Un procedimiento de búsqueda adaptativo aleatorio codicioso (GRASP) para inferir cláusulas lógicas a partir de ejemplos en tiempo polinomial y algunas extensiones. Modelado matemático por computadora, 27: 75–99, 1998.
75. S. Dharan y AS Nair. Biclúster de datos de expresión génica utilizando un procedimiento de búsqueda adaptativa aleatorio codicioso reactivo. BMC Bioinformatics, 10 (suplemento 1): S27, 2009.
76. JA D´ÿaz, DE Luna, J.-F. Camacho-Vallejo, and M.-S. Casas-Ramírez. GRASP y heurística híbrida GRASP-Tabu para resolver un problema de ubicación de cobertura máxima con el cliente orden de preferencia. Sistemas expertos con aplicaciones, 82:67–76, 2017.
77. A. Drexl y F. Salewski. Requisitos de distribución y restricciones de compacidad en la escuela horarios European J. of Operational Research, 102:193-214, 1997.
78. A. Duarte y R. Mart´ÿ. Búsqueda tabú y GRASP para el problema de máxima diversidad. European J. of Operational Research, 178:71–84, 2007.
79. A. Duarte, R. Mart´ÿ, A. Alvarez, and F. Ángel-Bello. Metaheurísticas para el ordenamiento lineal problema con los costos acumulativos. European J. of Operational Research, 216:270–277, 2012.
80. A. Duarte, R. Mart´ÿ, MGC Resende y RMA Silva. GRASP con heuris de reenlace de ruta tics para el problema del anti-ancho de banda. Redes, 58:171–189, 2011.
81. A. Duarte, CC Ribeiro y S. Urrutia. Una heurística ILS híbrida para la asignación de árbitros problema con una estrategia MIP incrustada. Apuntes de conferencias sobre informática, 4771: 82–95, 2007.
82. A. Duarte, J. Sánchez-Oro, MGC Resende, F. Glover y R. Mart´ÿ. AGARRE con exterior Reconexión de rutas para la minimización de la dispersión diferencial. Ciencias de la Información, 296:46–60, 2015.
83. AR Duarte, CC Ribeiro, S. Urrutia y EH Haeusler. Asignación de árbitros en deportes ligas Lecture Notes in Computer Science, 3867:158–173, 2007.
84. M. Essafi, X. Delorme y Al Dolgui. Líneas de equilibrado con máquinas CNC: un arranque múltiple y heurística basada. CIRP J. de Ciencia y Tecnología de Manufactura, 2:176–182, 2010.
85. H. Faria Jr., S. Binato, MGC Resende y DJ Falcao. Diseño de red de transmisión mediante un enfoque de reconexión de ruta adaptativa aleatoria codiciosa. Transacciones IEEE en sistemas de potencia, 20:43–49, 2005.
86. TA Feo y JF Bard. Programación de vuelos y planificación base de mantenimiento. administración Ciencia, 35:1415-1432, 1989.
87. TA Feo y JF Bard. El problema de la selección de la ruta de corte y la herramienta en sistemas asistidos por computadora. planificación de procesos. J. of Manufacturing Systems, 8:17-26, 1989.
88. TA Feo, JF Bard y S. Holland. Planificación y programación de cableado impreso en toda la instalación montaje de tablero. Investigación de operaciones, 43:219-230, 1995.
89. TA Feo y JL González-Velarde. El problema de la asignación de remolques intermodales: modelos, algoritmos y heurísticas. Ciencias del Transporte, 29:330-341, 1995.
90. TA Feo y MGC Resende. Una heurística probabilística para un conjunto computacionalmente difícil problema de cobertura. Cartas de investigación de operaciones, 8:67–71, 1989.
91. TA Feo y MGC Resende. Procedimientos de búsqueda adaptativa aleatorios codiciosos. J de Global Optimización, 6:109–133, 1995.
92. TA Feo, MGC Resende y SH Smith. Un procedimiento de búsqueda adaptativo aleatorio codicioso para el máximo conjunto independiente. Investigación de operaciones, 42:860-878, 1994.
93. TA Feo, K. Sarathy y J. McGahan. Un GRASP para la programación de una sola máquina con costos de configuración dependientes de la secuencia y penalizaciones por retraso lineal. Computers and Operations Research, 23:881–895, 1996.

94. TA Feo, K. Venkatraman y JF Bard. GRASP para una programación difícil de una sola máquina problema. Computadoras e Investigación de Operaciones, 18:635-643, 1991.
95. E. Fernández y R. Mart´ÿ. GRASP para dibujo de costura en mosaicos de fotografías aéreas mapas J. of Heuristics, 5:181-197, 1999.
96. P. Fiesta. Sobre algunos problemas de optimización en biología molecular. biociencia matematica, 207:219–234, 2007.
97. P. Festa, PM Pardalos, LS Pitsoulis y MGC Resende. GRASP con reenlace de ruta para el problema ponderado de MAXSAT. ACM J. de Experimental Algorithmics, 11:1–16, 2006.
98. P. Festa, PM Pardalos y MGC Resende. Algoritmo 815: subrutinas FORTRAN para cálculo de la solución aproximada a problemas de conjunto de retroalimentación usando GRASP. Transacciones ACM sobre software matemático, 27:456–464, 2001.
99. P. Festa, PM Pardalos, MGC Resende y CC Ribeiro. Heurísticas aleatorias para el Problema de MAX-CUT. Métodos de optimización y software, 7:1033–1058, 2002.
100. P. Festa y MGC Resende. GRASP: Una bibliografía comentada. En CC Ribeiro y P. Hansen, editores, Essays and Surveys in Metaheuristics, páginas 325–367. Académico Kluwer Editores, 2002.
101. P. Festa y MGC Resende. Una bibliografía comentada de GRASP, Parte I: Algoritmos. Transacciones internacionales en investigación operativa, 16:1–24, 2009.
102. P. Festa y MGC Resende. Una bibliografía comentada de GRASP, Parte II: Aplicaciones. Transacciones internacionales en investigación operativa, 16:131–172, 2009.
103. P. Festa, MGC Resende, P. Pardalos y CC Ribeiro. GRASP y VNS para Max-Cut. en Resúmenes extendidos de la Cuarta Conferencia Internacional de Metaheurísticas, páginas 371–376, Oporto, julio de 2001.
104. ML Fisher. El método de relajación lagrangiana para resolver problemas de programación entera. Ciencias de la gestión, 50:1861–1871, 2004.
105. C. Fleurent y F. Glover. Estrategias de inicio múltiple constructivas mejoradas para el problema de asignación cuadrática usando memoria adaptativa. INFORMA J. sobre Informática, 11:198–204, 1999.
106. E. Fonseca, R. Fuchsuber, LFM Santos, A. Plastino y SL Martins. Explorando el híbrido DM-GRASP metaheurístico para replicación de servidor eficiente para multidifusión confiable. En Conferencia Internacional sobre Metaheurísticas y Computación Inspirada en la Naturaleza, Hammamet, 2008.
107. RD Frinhani, RM Silva, GR Mateus, P. Festa y MGC Resende. GRASP con vinculación de rutas para agrupamiento de datos: un estudio de caso para datos biológicos. En PM Pardalos y S. Reben Nack, editores, Algoritmos experimentales, volumen 6630 de Lecture Notes in Computer Science, páginas 410–420. Springer, Berlín, 2011.
108. JB Ghosh. Aspectos computacionales del problema de máxima diversidad. La investigación de operaciones Cartas, 19:175–181, 1996.
109. F. Glover. Nueva cadena de eyección y métodos de ruta alternativa para problemas de viajante de comercio. En O. Balci, R. Sharda y S. Zenios, editores, Computer Science and Operations Research: Nuevos desarrollos en sus interfaces, páginas 449–509. Elsevier, 1992.
110. F. Glover. Cadenas de eyección, estructuras de referencia y métodos de camino alternativo para viajar problemas del vendedor. Matemáticas aplicadas discretas, 65:223–254, 1996.
111. F. Glover. Búsqueda tabú y programación de memoria adaptativa: avances, aplicaciones y desafíos. En RS Barr, RV Helgason y JL Kennington, editores, Interfaces in Computer Investigación científica y operativa, páginas 1–75. Editores académicos de Kluwer, 1996.
112. F. Glover. Métodos de oscilación estratégica y de arranque múltiple: principios para explotar la memoria adaptativa. En M. Laguna y JL Gonzales-Velarde, editores, Computing Tools for Modeling, Optimization and Simulation: Interfaces in Computer Science and Operations Research, págs. 1–24. Editores académicos de Kluwer, 2000.
113. F. Glover. Reenlace de ruta exterior para optimización cero-uno. J Internacional de Aplicada Computación metaheurística, 5:1–8, 2014.
114. F. Glover y M. Laguna. Búsqueda tabú. Editores académicos de Kluwer, 1997.
115. F. Glover, M. Laguna y R. Mart´ÿ. Fundamentos de la búsqueda dispersa y la vinculación de rutas. Control and Cybernetics, 39:653–684, 2000.

116. MX Goemans y DP Williamson. El método primal dual para algoritmos de aproximación y su aplicación a problemas de diseño de redes. En D. Hochbaum, editor, Algoritmos de aproximación para problemas NP-difíciles, páginas 144–191. PWS Publishing Company, 1996.
117. FC Gomes, CS Oliveira, PM Pardalos y MGC Resende. GRASP reactivo con enlace de ruta para la asignación de canales en redes de telefonía móvil. En Actas del 5º Taller Internacional sobre Algoritmos y Métodos Discretos para Computación y Comunicaciones Móviles, páginas 60–67. Prensa ACM, 2001.
118. PL Hammer y DJ Rader, Jr. Soluciones máximamente disjuntas del problema de cobertura de conjuntos. J. of Heuristics, 7:131-144, 2001.
119. BT Han y VT Raja. Una heurística GRASP para resolver un problema de ubicación de concentrador capacitado extendido. International J. of Information Technology and Decision Making, 2:597–617, 2003.
120. P. Hansen y N. Mladenovic. Desarrollos de búsqueda de vecindad variable. en cc Ribeiro y P. Hansen, editores, Essays and Surveys in Metaheuristics, páginas 415–439. Editores académicos de Kluwer, 2002.
121. JP Hart y AW Shogan. Heurísticas semicodiciosas: un estudio empírico. Cartas de investigación de operaciones, 6:107-114, 1987.
122. M. Held y RM Karp. El problema del viajante de comercio y los árboles de expansión mínimos. Investigación de operaciones, 18:1138-1162, 1970.
123. M. Held y RM Karp. El problema del viajante de comercio y los árboles de expansión mínimos: Parte II. Programación matemática, 1: 6–25, 1971.
124. M. Held, P. Wolfe y HP Crowder. Validación de la optimización del subgradiente. Matemático Programación, 6:62–88, 1974.
125. C. Helmberg y F. Rendl. El método del paquete espectral para la programación semidefinida. SIAM J. sobre optimización, 10:673–696, 2000.
126. AJ Higgins, S. Hajkowicz y E. Bui. Un modelo multiobjetivo para la toma de decisiones de inversión ambiental. Computers & Operations Research, 35:253–266, 2008.
127. MJ Hirsch. Heurísticas basadas en GRASP para problemas de optimización global continua. Tesis de doctorado, Departamento de Ingeniería Industrial y de Sistemas, Universidad de Florida, Gainesville, 2006.
128. MJ Hirsch, CN Meneses, PM Pardalos y MGC Resende. Optimización global por AGARRE continuo. Cartas de optimización, 1:201–212, 2007.
129. MJ Hirsch, PM Pardalos y MGC Resende. Resolución de sistemas de ecuaciones no lineales con GRASP continuo. Análisis no lineal: aplicaciones del mundo real, 10:2000–2006, 2009.
130. MJ Hirsch, PM Pardalos y MGC Resende. Aceleración continua GRASP. Revista europea de investigación operativa, 205: 507–521, 2010.
131. MJ Hirsch, PM Pardalos y MGC Resende. Correspondencia de puntos y líneas 3D proyectados utilizando un GRASP continuo. Transacciones internacionales en investigación operativa, 18:493–511, 2011.
132. K. Holmqvist, A. Migdalas y PM Pardalos. Búsqueda adaptativa aleatoria codiciosa para un problema de ubicación con economías de escala. En IM Bomze et al., editor, Developments in Global Optimization, páginas 301–313. Editores académicos de Kluwer, 1997.
133. K. Holmqvist, A. Migdalas y PM Pardalos. Un algoritmo GRASP para el problema de flujo de red de costo cóncavo mínimo no capacitado de fuente única. En PM Pardalos y D.-Z. Du, editores, Diseño de redes: Conectividad y ubicación de instalaciones, volumen 40 de la serie DIMACS sobre matemáticas discretas e informática teórica, páginas 131–142. Sociedad Matemática Estadounidense, 1998.
134. HH Hoos y T. Stutzle. Evaluación de los algoritmos de Las Vegas - Trampas y remedios. En G. Cooper y S. Moral, editores, Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence, páginas 238–245, Madison, 1998.
135. R. Interián y CC Ribeiro. Una heurística GRASP que utiliza la vinculación de rutas y reinicios para el problema del viajante de comercio de Steiner. Transacciones internacionales en investigación operativa, 24:1307–1323, 2017.

136. DS Johnson. Optimización local y el problema del viajante de comercio. En Proceedings of the 17th Colloquium on Automata, volumen 443 de LNCS, páginas 446–461. Springer-Verlag, 1990.

137. EH Kampke, JEC Arroyo y AG Santos. GRASP reactivo con vinculación de ruta para resolver problemas de programación de máquinas paralelas con tiempos de configuración dependientes de la secuencia asignable de recursos. En Actas del Congreso Mundial sobre la Naturaleza y la Computación de Inspiración Biológica, páginas 924–929, Coimbatore, 2009. IEEE.

138. H. Kautz, E. Horvitz, Y. Ruan, C. Gomes y B. Selman. Políticas de reinicio dinámico. En Actas de la Decimoctava Conferencia Nacional sobre Inteligencia Artificial, páginas 674–681, Edmonton, 2002. Asociación Estadounidense de Inteligencia Artificial.

139. G. Kendall, S. Knust, CC Ribeiro y S. Urrutia. Programación en deportes: Una anotación bibliografía. Computadoras e investigación de operaciones, 37: 1–19, 2010.

140. JG Klincewicz. Evitar los óptimos locales en el problema de ubicación de p-hub usando la búsqueda tabú y SUJETAR. Annals of Operations Research, 40:283–302, 1992.

141. JG Klincewicz y A. Rajan. Uso de GRASP para resolver el problema de agrupación de componentes. Logística de Investigación Naval, 41:893-912, 1994.

142. G. Kontoravdis y JF Bard. Un GRASP para el problema de enrutamiento de vehículos con ventanas de tiempo. ORSA J. sobre Informática, 7:10–23, 1995.

143. M. Kulich, JJ Miranda-Bront y L. Preucil. Una estrategia de selección de objetivos basada en metaheurísticas para la búsqueda de robots móviles en un entorno desconocido. Computadoras e investigación de operaciones, 84:178–187, 2017.

144. N. Labadi, C. Prins y M. Reghioui. GRASP con reenlace de ruta para el problema de enrutamiento de arco capacitado con ventanas de tiempo. En A. Fink y F. Rothlauf, editores, Advances in Computer Intelligence in Transport, Logistics, and Supply Chain Management, páginas 111–135. Springer, Berlín, 2008.

145. M. Laguna, TA Feo y HC Elrod. Un procedimiento de búsqueda adaptativo aleatorio codicioso para el problema de dos particiones. Investigación de operaciones, 42:677-687, 1994.

146. M. Laguna y JL González-Velarde. Una heurística de búsqueda para la programación justo a tiempo en máquinas paralelas. J. de Fabricación Inteligente, 2:253-260, 1991.

147. M. Laguna y R. Mart´ÿ. GRASP y reenlace de ruta para minimizar el cruce de línea recta de 2 capas. INFORMA J. sobre Informática, 11:44–52, 1999.

148. M. Laguna y R. Mart´ÿ. Un GRASP para colorear gráficos dispersos. Aplicaciones y optimización computacional, 19:165–178, 2001.

149. R. De Leone, P. Festa y E. Marchitto. Resolviendo un problema de programación de un conductor de autobús con heurísticas aleatorias de arranque múltiple. Transacciones internacionales en investigación operativa, 18:707–727, 2011.

150. Y. Li, PM Pardalos y MGC Resende. Un proceso de búsqueda adaptativo aleatorio codicioso dura para el problema de asignación cuadrática. En PM Pardalos y H. Wolkowicz, editores, Asignación cuadrática y problemas relacionados, volumen 16 de la serie DIMACS sobre matemáticas discretas e informática teórica, páginas 237–261. Sociedad Matemática Estadounidense, 1994.

151. A. Lim, B. Rodrigues y C. Wang. Problemas de flow shop de dos máquinas con un solo servidor. J. de Programación, 9:515-543, 2006.

152. A. Lim y F. Wang. Una búsqueda tabú dinámica suavizada integrada GRASP para m-VRPTW. En Proceedings of ICTAI 2004, páginas 704–708, 2004.

153. X. Liu, PM Pardalos, S. Rajasekaran y MGC Resende. Un GRASP para la asignación de frecuencias en redes de radio móvil. En BR Badrinath, F. Hsu, PM Pardalos y S. Rajasejaran, editores, Mobile Networks and Computing, volumen 52 de DIMACS Series on Discrete Mathematics and Theoretical Computer Science, páginas 195–201. Sociedad Matemática Estadounidense, 2000.

154. HR Lourenço, OC Martin y T. Stutzle. Búsqueda local iterada. En F. Glover y G. Kochenberger, editores, Handbook of Metaheuristics, páginas 321–353. Editores académicos de Kluwer, 2003.

155. HR Lourenço y D. Serra. Heurísticas de enfoque adaptativo para el problema de asignación generalizada. Mathware y Soft Computing, 9:209–234, 2002.

156. M. Luby, A. Sinclair y D. Zuckerman. Aceleración óptima de los algoritmos de Las Vegas. información mation Processing Letters, 47:173–180, 1993.

157. M. Luis, S. Salhi y G. Nagy. Un GRASP reactivo guiado para la fuente múltiple capacitada Problema de Weber. Computadoras e investigación de operaciones, 38:1014–1024, 2011.

158. CLB Maia, RAF Carmo, FG Freitas, GAL Campos y JT Souza. prueba automatizada priorización de casos con GRASP reactivo. Avances en Ingeniería de Software, 2010, 2010. Número de artículo 428521.

159. R. Mart´ÿ. Minimización del cruce de arco en grafos con GRASP. Transacciones IEE, 33:913–919, 2001.

160. R. Mart´ÿ. Minimización del cruce de arco en grafos con GRASP. Transacciones IEEE, 33:913–919, 2002.

161. R. Mart´ÿ y V. Estruch. Problema de dibujo bipartito incremental. Computadoras y Operaciones Investigación, 28:1287-1298, 2001.

162. R. Mart´ÿ y M. Laguna. Heurísticas y metaheurísticas para cruce de línea recta de 2 capas minimización. Matemáticas aplicadas discretas, 127: 665–678, 2003.

163. R. Mart´ÿ y F. Sandoya. GRASP y reconexión de caminos para el problema de la dispersión equitativa. Computadoras e investigación de operaciones, 40:3091–3099, 2013.

164. O. Martín y SW Otto. Combinación de recocido simulado con heurística de búsqueda local. Un nals of Operations Research, 63:57-75, 1996.

165. O. Martin, SW Otto y EW Felten. Cadenas de Markov de gran paso para el viajante de comercio problema. Sistemas complejos, 5:299-326, 1991.

166. SL Martins, PM Pardalos, MGC Resende y CC Ribeiro. Greedy random adapt I tenía procedimientos de búsqueda para el problema de Steiner en gráficos. En PM Pardalos, S. Rajasejaran y J. Rolim, editores, Randomization Methods in Algorithmic Design, volumen 43 de DIMACS Series on Discrete Mathematics and Theoretical Computer Science, páginas 133–145. Americano Sociedad Matemática, 1999.

167. SL Martins, MGC Resende, CC Ribeiro y PM Pardalos. Un GRASP paralelo para el problema del árbol de Steiner en gráficos utilizando una estrategia de búsqueda local híbrida. Diario de Global Optimización, 17:267–283, 2000.

168. SL Martins, CC Ribeiro y MC Souza. Un GRASP paralelo para el problema de Steiner en gráficos. En A. Ferreira y J. Rolim, editores, Proceedings of IRREGULAR'98 – 5th International Symposium on Solving Irregularly Structured Problems in Parallel, volumen 1457 de Lecture Notes in Computer Science, páginas 285–297. Springer-Verlag, 1998.

169. GR Mateus, MGC Resende y RMA Silva. GRASP con reconexión de rutas para el problema de asignación cuadrática generalizada. J. of Heuristics, 17:527-565, 2011.

170. T. Mavridou, PM Pardalos, LS Pitsoulis y MGC Resende. Un GRASP para el problema de asignación bicuadrática. European J. of Operational Research, 105:613-621, 1998.

171. M. Mestre, LS Ochi y SL Martins. GRASP con reenlace de ruta para el problema del viajante de comercio agrupado clideano simétrico. Informática e investigación de operaciones, 40:3218–3229, 2013.

172. N. Mladenovic y P. Hansen. Búsqueda de vecindad variable. Computadoras y Operaciones Investigación, 24:1097-1100, 1997.

173. SK Monkman, DJ Morrice y JF Bard. Una heurística de programación de producción para un fabricante de productos electrónicos con costos de configuración dependientes de la secuencia. European J. of Operational Research, 187:1100–1114, 2008.

174. REN Moraes y CC Ribeiro. Optimización de potencia en control de topología de redes inalámbricas ad hoc con requerimientos de biconectividad. Informática e investigación de operaciones, 40:3188–3196, 2013,

175. LF Moran-Mirabal, JL González-Velarde y MGC Resende. Heurísticas aleatorias para el problema del vendedor ambulante familiar. Transacciones Internacionales en Operaciones Investigación, 21:41–57, 2014.

176. LF Moran-Mirabal, JL González-Velarde,MGC Resende, and RMA Silva. Heurística aleatoria para la minimización del traspaso en redes de movilidad. J. of Heuristics, 19:845–880, 2013.

177. RA Murphey, PM Pardalos y LS Pitsoulis. Un procedimiento de búsqueda adaptativo aleatorio codicioso para el problema de seguimiento multisensor de objetivos múltiples. En PM Pardalos y D.-Z. du, editores, Diseño de redes: Conectividad y ubicación de instalaciones, volumen 40 de la serie DIMACS sobre matemáticas discretas e informática teórica, páginas 277–301. Americano Sociedad Matemática, 1998.

178. RA Murphey, PM Pardalos y LS Pitsoulis. Un GRASP paralelo para la asociación de datos problema de asignación multidimensional. En PM Pardalos, editor, Parallel Processing of Discrete Problems, volumen 106 de The IMA Volumes in Mathematics and Its Applications, páginas 159–180. Springer-Verlag, 1998.

179. MCV Nascimento y L. Pitsoulis. Detección de comunidades por maximización de la modularidad usando GRASP con reenlace de ruta. Computadoras e investigación de operaciones, 40:3121–3131, 2013.

180. MCV Nascimento, MGC Resende y FMB Toledo. Heurística GRASP con reenlace de ruta para el problema de dimensionamiento de lotes capacitados de múltiples plantas. J. Europea de Operacional Investigación, 200:747-754, 2010.

181. V.-P. Nguyen, C. Prins y C. Prodhon. Resolviendo el problema de enrutamiento de ubicación de dos escalones por un GRASP reforzado por un proceso de aprendizaje y reconexión de caminos. J. Europea de Operacional Investigación, 216:113–126, 2012.

182. E. Nowicki y C. Smutnicki. Un algoritmo de búsqueda tabú avanzado para el problema del taller. J. of Scheduling, 8:145-159, 2005.

183. CA Oliveira, PM Pardalos y MGC Resende. GRASP con reconexión de caminos para el problema de asignación cuadrática. En CC Ribeiro y SL Martins, editores, Proceedings of III Workshop on Efficient and Experimental Algorithms, volumen 3059, páginas 356–368. saltador, 2004.

184. IH Osman, B. Al-Ayoubi y M. Barake. Un procedimiento de búsqueda adaptativo aleatorio codicioso para el problema del gráfico planar máximo ponderado. Informática e Ingeniería Industrial, 45:635–651, 2003.

185. AVF Pacheco, GM Ribeiro y GR Mauri. Un GRASP con reenlace de ruta para el problema de programación de la plataforma de reacondicionamiento. International J. of Natural Computing Research, 1:1–14, 2010.

186. JA Pacheco y S. Casado. Resolviendo dos modelos de ubicación con pocas instalaciones mediante el uso de una heurística híbrida: un caso real de recursos de salud. Informática e investigación de operaciones, 32:3075–3091, 2005.

187. G. Palubeckis. Estrategias de búsqueda tabú multicomienzo para el problema de optimización cuadrática binaria sin restricciones. Annals of Operations Research, 131:259–282, 2004.

188. PM Pardalos, T. Qian y MGC Resende. Un procedimiento de búsqueda adaptativo aleatorio codicioso para el problema del conjunto de vértices de retroalimentación. J. de Optimización Combinatoria, 2:399–412, 1999

189. PM Pardalos, LS Pitsoulis y MGC Resende. La implementación paralela de GRASP para El problema de asignación cuadrática. En A. Ferreira y J. Rolim, editores, Parallel Algorithms para problemas de estructura irregular - Irregular'94, páginas 115–133. Editores académicos de Kluwer, 1995.

190. PM Pardalos, LS Pitsoulis y MGC Resende. Un GRASP paralelo para MAX-SAT prob lemas Lecture Notes in Computer Science, 1184: 575–585, 1996.

191. PM Pardalos, LS Pitsoulis y MGC Resende. Algoritmo 769: subrutinas Fortran para la solución aproximada de problemas de asignación cuadráticos dispersos usando GRASP. YMCA Transacciones sobre software matemático, 23:196–208, 1997.

192. F. Parreno, R. Alvarez-Valdes, JM Tamarit y JF Oliveira. Un algoritmo de espacio máximo para el problema de carga de contenedores. INFORMA J. sobre Informática, 20:412–422, 2008.

193. RA Patterson, H. Pirkul y E. Rolland. La técnica de razonamiento adaptativo de memoria para Resolviendo el problema del árbol de expansión mínimo capacitado. J. of Heuristics, 5:159-180, 1999.

194. O. Pedrola, M. Ruiz, L. Velasco, D. Careglio, O. González de Dios y J. Comellas. Un GRASP con heurística de vinculación de rutas para la multicapa IP/MPLS sobre WSON superviviente problema de optimización de red. Computadoras e investigación de operaciones, 40:3174–3187, 2013.

195. LS Pessoa, MGC Resende y CC Ribeiro. Experimentos con la heurística LAGRASP para conjunto k-revestimiento. Cartas de optimización, 5:407–419, 2011.

196. LS Pessoa, MGC Resende y CC Ribeiro. Una heurística lagrangiana híbrida con GRASP y reenlace de rutas para cubrir conjuntos k. Computadoras e investigación de operaciones, 40:3132–3146, 2013.

197. E. Pinana, I. Plana, V. Campos y R. Mart´ÿ. GRASP y reenlace de rutas para la minimización del ancho de banda de la matriz. European J. of Operational Research, 153:200–210, 2004.
198. LS Pitsoulis, PM Pardalos y DW Hearn. Soluciones aproximadas al problema de balanceo de turbinas. European J. of Operational Research, 130:147-155, 2001.
199. F. Poppe, M. Pickavet, P. Arijs y P. Demeester. Técnicas de diseño de redes restaurables en malla SDH. En Actas de la Conferencia Europea sobre Redes y Comunicaciones Ópticas, Volumen 2: Core and ATM Networks, páginas 94–101, 1997.
200. M. Prais y CC Ribeiro. Variación de parámetros en implementaciones GRASP. En Extended Abstracts of the Third Metaheuristics International Conference, páginas 375–380, Angra dos Reis, 1999.

201. M. Prais y CC Ribeiro. Variación de parámetros en procedimientos GRASP. Investigación Operativa, 9:1–20, 2000.
202. M. Prais y CC Ribeiro. GRASP reactivo: una aplicación a un problema de descomposición de matrices en la asignación de tráfico TDMA. INFORMA J. sobre Informática, 12:164–176, 2000.
203. M. Rahmani, M. Rashidinejad, EM Carreño y RA Romero. Reenlace evolutivo de múltiples movimientos para la planificación de la expansión de la red de transmisión. En 2010 IEEE Power and Energy Society General Meeting, páginas 1 a 6, Minneapolis, 2010. IEEE.
204. MC Rangel, NMM Abreu y PO Boaventura Netto. GRASP en el QAP: Un límite de aceptación para soluciones iniciales. Investigación operativa, 20:45–58, 2000.
205. MG Ravetti, FG Nakamura, CN Meneses, MGC Resende, GR Mateus y PM gorriones Heurística híbrida para el problema del taller de flujo de permutación. Informe técnico, Informe técnico de investigación de AT&T Labs, Florham Park, 2006.
206. M. Reghioui, C. Prins y Nacima Labadi. GRASP con reenlace de ruta para el problema de enrutamiento de arco capacitado con ventanas de tiempo. En M. Giacobini et al., editor, Applications of Evolutionary Computing, volumen 4448 de Lecture Notes in Computer Science, páginas 722–731. Springer, 2007.
207. LIP Resende y MGC Resende. GRASP significa enrutamiento de circuito virtual permanente de retransmisión de tramas. En CC Ribeiro y P. Hansen, editores, Extended Abstracts of the III Metaheuristics International Conference, páginas 397–401, Angra dos Reis, 1999.
208. MGC Resende. Cálculo de soluciones aproximadas del problema de cobertura máxima usando GRASP. J. of Heuristics, 4:161-171, 1998.
209. MGC Resende y TA Feo. GRASP para la satisfacción. En DS Johnson y MA Trick, editores, Cliques, Coloring, and Satisfiability: The Second DIMACS Implementation Challenge, volumen 26 de DIMACS Series on Discrete Mathematics and Theoretical Computer Science, páginas 499–520. Sociedad Matemática Estadounidense, 1996.
210. MGC Resende, TA Feo y SH Smith. Algoritmo 787: Subrutinas de Fortran para la solución aproximada de problemas de máximos conjuntos independientes usando GRASP. ACM Trans. Matemáticas. Software, 24:386-394, 1998.
211. MGC Resende, R. Mart´ÿ, M. Gallego, and A. Duarte. GRASP y reenlace de rutas para el problema de diversidad máximo-mínimo. Computers & Operations Research, 37:498–508, 2010.
212. MGC Resende, PM Pardalos y Y. Li. Algoritmo 754: Subrutinas de Fortran para la solución aproximada de problemas de asignación cuadrática densa utilizando GRASP. ACM Transactions on Mathematical Software, 22:104–118, 1996.
213. MGC Resende, LS Pitsoulis y PM Pardalos. Solución aproximada de problemas ponderados MAX SAT utilizando GRASP. En J. Gu y PM Pardalos, editores, Satisfiability Problems, volumen 35 de DIMACS Series on Discrete Mathematics and Theoretical Computer Science, páginas 393–405. Sociedad Matemática Estadounidense, 1997.
214. MGC Resende, LS Pitsoulis y PM Pardalos. Subrutinas Fortran para computar soluciones aproximadas de problemas MAX-SAT utilizando GRASP. Matemáticas aplicadas discretas, 100: 95–113, 2000.

215. MGC Resende y CC Ribeiro. Un GRASP para la planarización de grafos. Redes, 29:173–189, 1997.

216. MGC Resende y CC Ribeiro. Planarización de grafos. En C. Floudas y PM Pardalos, editores, Encyclopedia of Optimization, volumen 2, páginas 368–373. Editores académicos de Kluwer, Boston, 2001.
217. MGC Resende y CC Ribeiro. Un GRASP con reenlace de ruta para circuito virtual privado enrutamiento Redes, 41:104–114, 2003.
218. MGC Resende y CC Ribeiro. GRASP con vinculación de rutas: avances y aplicaciones recientes. En T. Ibaraki, K. Nonobe y M. Yagiura, editores, Metaheuristics: Progress as Real Solucionadores de problemas, páginas 29–63. Springer, 2005.
219. MGC Resende y CC Ribeiro. Estrategias de reinicio para GRASP con heurísticas de vinculación de rutas. Cartas de optimización, 5:467–478, 2011.
220. MGC Resende y CC Ribeiro. Optimización por GRASP: Greedy Randomized Adaptive Procedimientos de búsqueda. Springer, 2016.
221. MGC Resende y RF Werneck. Una heurística híbrida para el problema de la p-mediana. j de Heurística, 10:59–88, 2004.
222. MGC Resende y RF Werneck. Una heurística híbrida de inicio múltiple para el problema de ubicación de instalaciones no capacitadas. European J. of Operational Research, 174:54–68, 2006.
223. AP Reynolds y B. de la Iglesia. El GRASP multiobjetivo para la clasificación parcial. suave Informática, 13:227–243, 2009. ,
224. CC Ribeiro. GRASP: Une metaheuristique gloutonę et probabiliste. En J. Teghem y , páginas 153–176. M. Pirlot, editores, Optimization Approchee en Recherche Operationnellé Hermes, 2002.
225. CC Ribeiro. Programación deportiva: problemas y aplicaciones. Transacciones Internacionales en Investigación operativa, 19:201–226, 2012.
226. CC Ribeiro, SL Martins e I. Rosseti. Metaheurísticas para problemas de optimización en com comunicaciones informáticas. Comunicaciones informáticas, 30:656–669, 2007.
227. CC Ribeiro y MGC Resende. Algoritmo 797: subrutinas Fortran para aproximado solución de problemas de planarización de grafos utilizando GRASP. Transacciones ACM en Matemáticas Software, 25:342–352, 1999.
228. CC Ribeiro, CD Ribeiro y RS Lanzelotte. Optimización de consultas en relaciones distribuidas bases de datos J. of Heuristics, 3:5-23, 1997.
229. CC Ribeiro e I. Rosseti. Implementaciones cooperativas paralelas eficientes de la heurística GRASP. Cómputo paralelo, 33:21–35, 2007.
230. CC Ribeiro e I. Rosseti. tttplots-compare: un programa de perl para comparar gráficos de tiempo hasta el objetivo o distribuciones generales de tiempo de ejecución de algoritmos aleatorios. Cartas de optimización, 9:601–614, 2015.
231. CC Ribeiro, I. Rosseti y R. Vallejos. Sobre el uso de distribuciones de tiempo de ejecución para evaluar y comparar algoritmos estocásticos de búsqueda local. En T. Sttzle, M. Biratari y HH Hoos, editores, Ingeniería de algoritmos de búsqueda local estocástica, volumen 5752 de Lecture Notes in Computer Ciencia, páginas 16–30. Springer, Berlín, 2009.
232. CC Ribeiro, I. Rosseti y R. Vallejos. Explotación de distribuciones de tiempo de ejecución para comparar algoritmos de búsqueda locales estocásticos paralelos y secuenciales. J. de Optimización Global, 54:405–429, 2012.
233. CC Ribeiro y MC Souza. Búsqueda de vecindad variable para el grado restringido Problema del árbol de expansión mínimo. Matemáticas aplicadas discretas, 118: 43–54, 2002.
234. CC Ribeiro, E. Uchoa y RF Werneck. Un GRASP híbrido con perturbaciones para el Problema de Steiner en gráficos. INFORMA J. sobre Informática, 14:228–246, 2002.
235. CC Ribeiro y S. Urrutia. Heurísticas para el problema del torneo itinerante reflejado. European J. of Operational Research, 179:775–787, 2007.
236. CC Ribeiro y DS Vianna. Una heurística GRASP/VND para el problema de filogenia usando un nueva estructura vecinal. Transacciones internacionales en investigación operativa, 12:325–338, 2005.
237. RZ Ríos-Mercado y JF Bard. Heurística para el problema de la línea de flujo con costos de instalación. European J. of Operational Research, 110:76-98, 1998.
238. RZ Ríos-Mercado y JF Bard. Una heurística basada en TSP mejorada para la minimización del intervalo de tiempo en un taller de flujo con costos de configuración. J. of Heuristics, 5:57-74, 1999.

239. RZ R´os-Mercado y E. Fernández. Un GRASP reactivo para un problema de diseño de territorio comercial con múltiples requisitos de equilibrio. Computers & Operations Research, 36:755–776, 2009.

240. A. Riva y F. Amigoni. Una metaheurística GRASP para la cobertura de entornos de cuadrícula con herramientas de huella limitada. En Actas de la 16.ª Conferencia sobre Agentes Autónomos y Sistemas Multiagente, AAMAS '17, páginas 484–491, Richland, SC, 2017. Fundación Internacional para Agentes Autónomos y Sistemas Multiagente.

241. AJ Robertson. Un conjunto de implementaciones de procedimientos de búsqueda local adaptativa aleatorios codiciosos (GRASP) para el problema de asignación multidimensional. Aplicaciones y optimización computacional, 19:145–164, 2001.

242. PL Rocha, MG Ravetti y GR Mateus. El GRASP metaheurístico como un límite superior para un algoritmo de ramificación y límite en un problema de programación con máquinas paralelas no relacionadas y tiempos de configuración dependientes de la secuencia. En Actas del 4º Taller EU/ME: Diseño y Evaluación de Metaheurísticas Híbridas Avanzadas, volumen 1, páginas 62–67, 2004.

243. FJ Rodríguez, F. Glover, C. Garc´a-Mart´ynez, R. Mart´y y M. Lozano. Comprender con la vinculación de rutas exteriores y la búsqueda local restringida para el problema de partición de números bidireccionales multidimensionales. Computadoras e investigación de operaciones, 78: 243–254, 2017.

244. FJ Rodríguez, C. Blum, C. García-Martínez y M. Lozano. GRASP con reenlace de ruta para el problema de programación de máquinas paralelas no idénticas con la minimización de los tiempos de finalización ponderados totales. Annals of Operations Research, 201:383–401, 2012.

245. MA Salazar-Aguilar, RZ R´os-Mercado, and JL Gonzalez-Velarde. Estrategias GRASP para un problema de diseño de territorio comercial bi-objetivo. J. of Heuristics, 19:179–200, 2013.

246. J. Santamar´ya, O. Cordon, S. Damas, R. Mart´y, and RJ Palma. GRASP y reconexión de rutas evolutivas para el registro de imágenes médicas basado en coincidencia de puntos. En el Congreso IEEE de 2010 sobre Computación Evolutiva, páginas 1–8. IEEE, 2010.

247. J. Santamar´ya, O. Cordon, S. Damas, R. Mart´y, and RJ Palma. GRASP e hibridaciones de reconexión de rutas para el problema de registro de imágenes basado en coincidencia de puntos. J. of Heuristics, 18:169–192, 2012.

248. D. Santos, A. de Sousa y F. Alvelos. Una generación de columnas híbridas con GRASP y reconexión de rutas para el problema de equilibrio de carga de la red. Computadoras e investigación de operaciones, 40:3147–3158, 2013.

249. M. Scaparra y R. Iglesia. Una heurística GRASP y de reconexión de caminos para redes de caminos rurales desarrollo. J. of Heuristics, 11:89–108, 2005.

250. IV Sergienko, VP Shilo y VA Roshchin. Paralelización de optimización para problemas de programación discreta. Cibernética y análisis de sistemas, 40:184–189, 2004.

251. OV Shylo, T. Middelkoop y PM Pardalos. Estrategias de reinicio en optimización: Paralelo y casos en serie. Computación paralela, 37:60–68, 2011.

252. OV Shylo, OA Prokopyev y J. Rajgopal. Sobre carteras de algoritmos y estrategias de reinicio. Cartas de investigación operativa, 39:49–52, 2011.

253. F. Silva y D. Serra. Localización de servicios de emergencia con diferentes prioridades: el problema de la ubicación de cobertura de colas prioritarias. J. de la Sociedad de Investigación Operativa, 59:1229-1238, 2007.

254. RMA Silva, MGC Resende, PM Pardalos y MJ Hirsch. La biblioteca de Python/C para la optimización global restringida con GRASP continuo. Cartas de optimización, 7:967–984, 2013.

255. RMA Silva, MGC Resende, PM Pardalos, GR Mateus y G. de Tomi. GRASP con enlace de ruta para el diseño de instalaciones. En BI Goldengorin, VA Kalyagin y PM Pardalos, editores, Modelos, algoritmos y tecnologías para el análisis de redes, volumen 59 de Springer Proceedings in Mathematics & Statistics, páginas 175–190. Springer, Berlín, 2013.

256. D. Sosnowska. Optimización de un problema simplificado de asignación de flotas con metaheurísticas: recocido simulado y GRASP. En PM Pardalos, editor, Aproximación y complejidad en optimización numérica. Editores académicos de Kluwer, 2000.

257. MC Souza, C. Duhamel y CC Ribeiro. Una heurística GRASP para el problema del árbol de expansión mínimo capacitado utilizando una estrategia de búsqueda local basada en memoria. En MGC Resende

y J. Souza, editores, Metaheuristics: Computer Decision-Making, páginas 627–658. Editorial académica Kluwer, 2004.

258. A. Srinivasan, KG Ramakrishnan, K. Kumaram, M. Aravamudam y S. Naqvi. Diseño óptimo de redes de señalización para telefonía por Internet. En IEEE INFOCOM 2000, volumen 2, páginas 707–716, 2000.

259. H. Takahashi y A. Matsuyama. Una solución aproximada para el problema de Steiner en gráficos. Mathematica Japónica, 24: 573–577, 1980.

260. TL Urbano. Procedimientos de solución al problema de disposición dinámica de instalaciones. Annals of Operations Research, 76:323–342, 1998.

261. TL Urbano, W.-C. Chiang y RA Russell. El problema integrado de distribución y asignación de máquinas. International J. of Production Research, 38:2913-2930, 2000.

262. FL Usberti, PM Francia y ALM Francia. GRASP con enlace evolutivo de rutas para el problema de enrutamiento de arco capacitado. Computadoras e investigación de operaciones, 40:3206–3217, 2013.

263. JX Vianna Neto, DLA Bernert y LS Coelho. Algoritmo GRASP continuo aplicado al problema de despacho económico de unidades térmicas. En Actas del 13º Congreso Brasileño de Ciencias e Ingeniería Térmicas, Uberlandia, 2010.

264. JG Villegas, C. Prins, C. Prodhon, AL Medaglia y N. Velasco. GRASP/VND y la búsqueda local evolutiva de inicio múltiple para el problema de enrutamiento de un solo camión y remolque con depósitos satelitales. Aplicaciones de ingeniería de la inteligencia artificial, 23:780–794, 2010.

265. JG Villegas, C. Prins, C. Prodhon, AL Medaglia y N. Velasco. Un GRASP con enlace evolutivo de ruta para el problema de enrutamiento de camiones y remolques. Computadoras e investigación de operaciones, 38:1319–1334, 2011.

266. DL Woodruff y E. Zemel. Hashing vectores para búsqueda tabú. Annals of Operations Research, 41:123–137, 1993.

267. JY Xu y SY Chiu. Procedimiento heurístico efectivo para un problema de programación de un técnico de campo. J. of Heuristics, 7:495-509, 2001.

268. J. Yen, M. Carlsson, M. Chang, JM García y H. Nguyen. Resolución de restricciones para el diseño de máscaras de impresión de inyección de tinta. J. of Imaging Science and Technology, 44:391-397, 2000.