Revista Facultad de Ingeniería (Rev. Fac. Ing.) Vol. 33, No. 68.
L-ISSN: 0121-1129, e-ISSN: 2357-5328.

DOI: 10.19053/01211129.v33.n68.2024.18079

ALGORITMO DE COLONIA DE ABEJAS ARTIFICIALES MODIFICADO PARA EL PROBLEMA DE COBERTURA DE CONJUNTOS

Modified Artificial Bee Colony Algorithm for the Set Covering Problem

Nelson-Enrique Quemá-Taimbud
Universidad del Cauca (Popayán, Colombia).
nquema@unicauca.edu.co

Martha-Eliana Mendoza-Becerra
Universidad del Cauca (Popayán, Colombia).
mmendoza@unicauca.edu.co

Fecha de recibido: 12-02-2024
Fecha de aceptado: 17-06-2024

RESUMEN

El Problema de Cobertura de Conjuntos (PCC) es un problema clásico de optimización combinatoria que pertenece a los NP-completos identificados por Karp. Tiene múltiples aplicaciones en el mundo real como la ubicación de servicios de emergencia, la planificación militar y la toma de decisiones en situaciones de pandemia, entre otros. En la literatura se identificó el uso de algoritmos metaheurísticos para resolver el PCC, destacando su capacidad para proporcionar soluciones de alta calidad. En este artículo se propuso la modificación de los métodos de inicialización y búsqueda local del algoritmo del estado del arte colonia de abejas artificiales (ABC por sus siglas en inglés) para el PCC (ABC_SCP), buscando equilibrar la exploración y explotación del espacio de búsqueda y obtener resultados iguales o cercanos a los mejores conocidos BKS de las instancias de gran escala de OR-Library. Para esto, se siguió una metodología basada en el Patrón de Investigación Iterativa (PII), propuesto por Pratt, la cual permitió identificar los métodos de inicialización y de búsqueda local del estado del arte aplicados recientemente al PCC, seleccionar algunos mediante criterios definidos por los autores, obtener variantes del algoritmo y seleccionar la que obtuvo un desempeño superior para ajustar un afinamiento de parámetros. El algoritmo modificado superó los resultados obtenidos por el ABC_SCP y otros enfoques del estado del arte, logrando una mejora en los valores promedio de ejecución de varias instancias. Así pues, esta investigación contribuye a la comunidad científica con un nuevo algoritmo que resuelve el PCC con componentes que pueden ser aplicados en otros problemas combinatorios de la literatura relacionados con este.

Keywords: algoritmos metaheurísticos; colonias de abejas artificiales; optimización combinatoria; Problema de Cobertura de Conjuntos.

ABSTRACT

The Set Covering Problem (SCP) is a classic combinatorial optimization problem that belongs to the NP-complete problems identified by Karp. It has multiple applications in the real world, such as the location of emergency services, military planning, and decision making in pandemic situations. The use of metaheuristic algorithms to solve the SCP is identified in the literature, highlighting their ability to provide high-quality solutions. This paper proposes the modification of the initialization and local search methods of the state-of-the-art artificial bee colony algorithm for SCP (ABC_SCP) to balance the exploration and exploitation of the search space and obtain results equal or close to the best known from large-scale instances of OR-Library. For this purpose, a methodology based on the Iterative Research Pattern proposed by Pratt was followed, which allowed us

Cómo citar este artículo: N. E. Quemá-Taimbud, y M. E. Mendoza-Becerra, Algoritmo de colonia de abejas artificiales modificado para el problema de cobertura de conjunto. *Revista Facultad de Ingeniería*, vol. 33, no. 68, e18079, 2024. https://doi.org/10.19053/01211129.v33.n68.2024.18079

to identify the initialization and local search methods of the state of the art recently applied to the SCP, select some of them using criteria defined by the authors, obtain variants of the algorithm, and select the one that obtained the best performance for a subsequent parameter tuning. The modified algorithm outperformed the results obtained by ABC_SCP and other state-of-the-art approaches, achieving an improvement in the average execution values for several instances. This research contributes to the scientific community a new algorithm that solves the SCP, with components that can be applied to other combinatorial problems related to the SCP.

Palabras clave: artificial bee colony; combinatorial optimization; metaheuristic algorithms; Set Covering Problem.

1. INTRODUCCIÓN

El Problema de Cobertura de Conjuntos (PCC) es un problema clásico de optimización combinatoria que hace parte de los 21 problemas NP-completos de Karp [1]. Muchas aplicaciones del mundo real pueden modelarse como PCC, por ejemplo, la ubicación de servicios de emergencia [2]and thus assist in saving patients' lives and improving their health outcomes. A considerable amount of spatial optimization research has been devoted to the development of models to support location planning in the context of EMS, with extensive applications in policy making around the world. However, in China, studies on the location of EMS facilities have not been paid enough attention to, let alone their practical applications. This paper conducted location optimization for EMS facilities in Chengdu, one of the biggest cities in southwest China with more than 16.5 million population, aiming to optimize the EMS system by adding (upgrading, la planificación militar [3], la toma de decisiones en situación de pandemia por COVID-19 [4] with its significant loss to human life and a huge economic toll. In this case, the fever clinics (FCs, entre otros.

Su solución está dada por un subconjunto de elementos disponible, sujeto a restricciones del problema. El PCC se define mediante una matriz binaria $A = [a_{ij}]$ de tamaño $m \times n$, donde cada elemento puede ser 0 o 1. Se definen $I = \{1,...,i,...,n\}$ y $J = \{1,...,j,...,m\}$ como los conjuntos de filas y columnas de A respectivamente, en donde una columna j cubre una fila i si $a_{ij} = 1$. Cada columna j de la matriz A está asociada a un costo no negativo c_j. El objetivo del problema es elegir un subconjunto de columnas $x_j \subseteq \{1,...,N\}$ que minimice la suma de los costos c_j, en donde cada fila debe estar cubierta por, al menos, una columna. El PCC se formula de acuerdo con las ecuaciones (1), (2) y (3).

Minimizar:

$$Min\ f(x) = \sum_{j=1}^{m} c_j x_j \qquad (1)$$

Sujeto a las restricciones:

$$\sum_{j=1}^{m} a_{ij} x_j > 1, \forall i \in I \qquad (2)$$

$$x_j \in \{0,1\} \qquad (3)$$

La ecuación (2) indica que cada fila debe estar cubierta por, al menos, una columna, y la ecuación (3) que las variables de decisión x_j pueden asumir solo valores binarios: 0 o 1. El PCC puede solucionarse desde un enfoque Unicosto en el que los costos del problema se igualan a 1, o No-Unicosto en el que los costos corresponden a los asignados por el problema.

En años recientes se ha identificado la aplicación de los algoritmos metaheurísticos al PCC, que son procedimientos iterativos que guían una heurística subordinada, combinando diferentes conceptos para explorar y explotar adecuadamente el espacio de búsqueda de forma inteligente. En estos algoritmos se reconocieron dos enfoques. Por un lado, algoritmos de enjambre que no habían sido aplicados al PCC, como el de hormigas león [5] is an area of interest at the level of basic and applied science. This is due to the fact that many of the problems addressed at industrial level are of a combinatorial type and a subset no less than these are of the NP-hard type. In this article, a mechanism of binarization of continuous metaheuristics that uses the concept of the percentile is proposed. This percentile concept is applied to the An Lion optimization algorithm, solving the set covering problem (SCP que mejora la búsqueda mediante movimientos perturbados de las hormigas; algoritmo basado en saltamontes [6] we explore a general binarization mechanism of continuous metaheuristics based on the percentile concept. In particular, we apply the percentile concept to the Grasshopper optimization algorithm in order to solve the set covering problem (SCP que, por su condición de plagas agrícolas, exhiben características de exploración y explotación en su comportamiento; mosca de la fruta binario [7] que emula el comportamiento de las moscas de la fruta en la búsqueda de alimento; basado en enseñanza-aprendizaje [8] que simula la dinámica de enseñanza en aulas escolares; y de optimización basado en redes sociales [9] que utiliza interacciones similares a las de plataformas sociales.

Por otro, algoritmos que ya se habían aplicado al PCC y que habían sido modificados para evaluar su impacto en los resultados como el híbrido que combina la colonia de abejas artificiales con búsqueda local [10]; de búsqueda local de ponderación de filas [11] que utiliza ponderación, prioridades y marcas de tiempo para mejorar la exploración; una variante del algoritmo de búsqueda de monos binario que incorpora escalada de montañas y cooperación entre monos [12]; de agujeros negros binario multidinámico [13] que ajusta probabilidades de transición y definición de restricciones; metaheurístico que usa descenso de vecindario variable y recocido simulado [14]; Ant-Set [15] que aplica la colonia de hormigas con búsqueda local; de gotas de agua adaptativo [16]; y uno basado en agujeros negros que adapta el número de estrellas de la población [17]. También se menciona un algoritmo de búsqueda adaptativo con recocido simulado, mutación y penalización de soluciones [18], y uno de colonia de abejas artificiales que se combina con otro genético para ajustar parámetros automáticamente [19].

Los algoritmos reportan resultados al resolver las instancias de prueba del repositorio de OR-Library. Se observó que en las instancias de mediana escala el promedio del mejor valor (Best) y el promedio de los valores logrados en cada iteración (Avg) de un algoritmo es igual al mejor resultado conocido (BKS), pero en las instancias de gran escala este no logra llegar al BKS.

Esta investigación se enmarcó en el enfoque de algoritmos que han sido aplicados al PCC y son modificados, tomando como referente uno de los algoritmos del estado del arte denominado ABC_SCP (Artificial Bee Colony – Set Covering Problem) [10], el cual obtiene resultados iguales a los *BKS* en la mayoría de instancias de OR-Library, con excepción de algunas de gran escala en las que no lo logra.

Así pues, se presentan las modificaciones realizadas a este algoritmo en el método de inicialización y búsqueda local, con base en métodos usados previamente en la literatura respecto al PCC, buscando hacer un balance entre la exploración y explotación del espacio de búsqueda.

Con esta investigación, la comunidad científica cuenta con un nuevo algoritmo para el PCC que supera los resultados de otros algoritmos del estado del arte en las instancias de OR-Library de gran escala. A su vez, el algoritmo propuesto puede ser aplicado en otros problemas combinatorios de la literatura similares al PCC.

2. METODOLOGÍA

Se siguió el Patrón de Investigación Iterativa (PII) propuesto por Pratt [20], diseñado para orientar el desarrollo de proyectos de investigación que involucran una solución computacional. Esta metodología consta de cuatro etapas: observación, identificación, desarrollo y prueba.

A. Observación

En esta etapa, se realizó un mapeo sistemático de la literatura [21] sobre los métodos de inicialización y búsqueda local que habían sido utilizados recientemente en el estado del arte para el PCC.

B. Identificación

Tomando como punto de partida los resultados del mapeo sistemático se seleccionaron los métodos para modificar el algoritmo base ABC_SCP. Para los métodos de inicialización (INIT) se definieron cuatro criterios: (C1) aleatoriedad, buscando generar soluciones que adicionen columnas de forma aleatoria para explorar el espacio de búsqueda con menor costo computacional [10, 22]; (C2) redundancia, que implica incluir columnas redundantes para mejorar la explotación de soluciones y reducir el costo computacional del operador de reparación en el algoritmo ABC_SPC [10]; (C3) adaptación al algoritmo base, verificando si el método es compatible con el algoritmo y si requiere recursos adicionales de *software*, y (C4) estrategia de inicialización que verifica si el método aplica una estrategia distinta de generación de soluciones.

La Tabla 1 muestra el resultado de la aplicación de los criterios a cada método. Los de inicialización aleatoria con heurística (RHeuristic), aleatorio (Random) y construcción iterativa (IterConstruct) cumplen (C) con los criterios C1 y C2 al generar soluciones seleccionando columnas de manera aleatoria y manteniendo columnas redundantes. En cuanto al criterio C3, los tres son compatibles y no requieren recursos adicionales de *software*, lo que permite su implementación.

Con respecto a los métodos basados en algoritmos aproximados de redondeo determinista (AAD), redondeo aleatorio 1 (AAR), redondeo aleatorio 2 (AAR2) y sus variantes con búsqueda local (LO), cumplen con los criterios C1 y C2, pero no cumplen (NC) con el C3, ya que para su ejecución requiere de soluciones relajadas que se generan con ayuda del *software* comercial CPLEX que, de implementarse con el algoritmo base, incrementaría su complejidad.

En el caso de las variantes de estos algoritmos con LO no cumplen con todos los criterios, ya que las soluciones generadas se optimizan acercándose a óptimos locales, eliminan columnas redundantes y cuentan con la dependencia del software CPLEX.

En la evaluación del criterio C4, tomando como referencia que el algoritmo base adiciona columnas a una nueva solución a partir de la lista restringida de columnas que pueden cubrir una fila (RCL),

los métodos RHeuristic y IterConstruct difieren de esta estrategia, adicionando de forma aleatoria columnas que cubran la mayor cantidad de filas. A su vez, el método Random adiciona un porcentaje de columnas de forma aleatoria a una solución y los métodos AAD, AAR y AAR2, al igual que sus variantes con LO, cumplen con el criterio, ya que inician a partir de una solución relajada para adicionar de forma aleatoria columnas a una solución.

Siguiendo el análisis realizado, los métodos seleccionados fueron RHeuristic, Random e IterConstruct por cumplir con los criterios evaluados.

Tabla 1. Aplicación de criterios de selección a métodos de inicialización

Método INIT	C1	C2	C3	C4	Método INIT	C1	C2	C3	C4
RHeuristic	C	C	C	C	AAR2	C	C	NC	C
Random	C	C	C	C	AAD + LO	NC	NC	NC	C
IterConstr	C	C	C	C	AAR + LO	NC	NC	NC	C
AAD	C	C	NC	C	AAR2 + LO	NC	NC	NC	C
AAR	C	C	NC	C	-	-	-	-	-

Posteriormente, se seleccionaron los métodos de búsqueda local (BL) considerando tres criterios: (C1) resultados reportados, buscando identificar los métodos aplicados a los grupos de instancias NRE, NRF, NRG, NRH del repositorio OR-Library con valores Avg destacados; (C2) adaptación al algoritmo base, verificando la disponibilidad de las especificaciones necesarias para la implementación de los métodos, y (C3) mejor explotación, buscando que el método contribuyera a mejorar la explotación del algoritmo evitando visitar nuevamente soluciones. Los criterios aplicados a los métodos de búsqueda local se muestran en la Tabla 2, indicando si estos cumplen con el criterio (C) o no (NC).

Tabla 2. Aplicación de criterios de selección a métodos de búsqueda local

Método BL	C1	C2	C3	Método BL	C1	C2	C3
RWLS	C	C	C	SLVS	NC	C	NC
IterLS	C	C	C	CCS	NC	NC	NC
KNN	C	NC	NC	JBLS	NC	NC	NC

En cuanto al criterio C1, la comparación de resultados se observa en la Tabla 3. El método Búsqueda Local de Ponderación de Filas (RWLS) se destaca al alcanzar un valor Avg igual al BKS en los grupos de instancias NRE, NRF, NRH y en las instancias NRG1 y NRG5, además de valores cercanos al BKS en las instancias NRG2, NRG3 y NRG4. También, el método búsqueda local Iterada (IterLS) muestra buenos resultados, igualando el BKS en tres instancias (NRE1, NRE5 y NRF2) y logrando valores cercanos en siete instancias más.

Tabla 3. Resultados reportados al aplicar los métodos de búsqueda local

Inst.	BKS	S-LV	KNN	IterLS	RWLS	Inst.	BKS	S-LV	KNN	IterLS	RWLS
		Avg	Avg	Avg	Avg			Avg	Avg	Avg	Avg
NRE1	29	**29.0**	**29.4**	**29.0**	**29.0**	*NRG1*	176	180.3	177.2	177.2	**176.0**
NRE2	30	32.1	**30.2**	**30.5**	**30.0**	*NRG2*	154	160.4	156.6	157.5	**154.2**
NRE3	27	28.7	**27.6**	**27.8**	**27.0**	*NRG3*	166	171.6	169.2	170.9	**166.6**
NRE4	28	29.6	**28.7**	**28.1**	**28.0**	*NRG4*	168	172.2	170.2	174.3	**168.2**
NRE5	28	**28.9**	**28.4**	**28.0**	**28.0**	*NRG5*	168	175.0	**168.6**	172.5	**168.0**
NRF1	14	15.0	**14.6**	**14.2**	**14.0**	*NRH1*	63	67.5	64.6	65.4	**63.0**
NRF2	15	**15.9**	**15.2**	**15.0**	**15.0**	*NRH2*	63	66.0	64.8	65.0	**63.0**
NRF3	14	16.7	**14.7**	**14.7**	**14.0**	*NRH3*	59	63.0	60.7	60.9	**59.0**
NRF4	14	15.0	**14.8**	**14.2**	**14.0**	*NRH4*	58	63.5	59.4	59.6	**58.0**
NRF5	13	15.10	**13.9**	**13.77**	**13.0**	*NRH5*	55	58.1	**55.2**	**55.8**	**55.0**

Por su parte, el método basado en el algoritmo de los K vecinos más cercanos (KNN) consigue 12 resultados cercanos al BKS, pero no logra igualar el valor promedio en ninguna instancia. El método de búsqueda basada en olor y visión (S-LV) solo logra un valor igual al BKS (NRE1) y dos valores cercanos (NRE5 y NRF2), por lo cual no cumple con el criterio. Los métodos de estrategia de verificación de configuración (CCS) y búsqueda local JB (JBLS) no cumplen con el criterio, ya que no reportaron resultados destacados.

En cuanto al criterio C2, se determinó que los métodos S-LV y RWLS cumplen parcialmente, requiriendo la adición de un ciclo de terminación para no incrementar el tiempo de ejecución del algoritmo base. El método IterLS lo cumple, sin embargo, se define adicionar tres parámetros de eliminación de columnas: P_b que es la probabilidad de adicionar una columna a un grupo de reinicio L_1 o L_2, *COL_DROP_1* y *COL_DROP_2* para eliminar columnas de una solución *S* según el número de columnas que contenga. Los métodos KNN y CCS, aunque disponen de la especificación necesaria incrementan la complejidad del algoritmo ABC_SCP, incumpliendo con el criterio. El método JBLS también fue descartado debido a una especificación incompleta en su estudio de referencia.

Con respecto al criterio C3, los métodos RWLS y IterLS lo cumplen, ya que implementan estrategias de ponderación de filas y asignación de puntajes a las columnas, y los otros métodos no lo cumplen al no contar con este tipo de estrategia. Después de aplicar los criterios definidos a los métodos de búsqueda local se seleccionaron los métodos IterLS y RWLS para modificar el algoritmo ABC_SCP.

C. Desarrollo de la solución

En esta etapa se definieron dos iteraciones. En la primera se implementó el algoritmo base (ABC_SCP_IMP), de acuerdo con las especificaciones del artículo y definición de parámetros de ejecución, y en la segunda se realizó la modificación del algoritmo, involucrando las adaptaciones hechas a los métodos de inicialización y búsqueda local seleccionados en la etapa anterior.

Como resultado de la primera iteración, se obtuvo el algoritmo ABC_SCP_IMP, el cual se ejecut**ó** con los grupo de instancias NRE, NRF, NRG y NRH del repositorio OR-Library [23]. Estos resultados se compararon con el algoritmo base (ABC_SCP), teniendo en cuenta el mejor valor de aptitud conocido de cada instancia (BKS), el mejor valor obtenido en las ejecuciones (Best) y el valor promedio (Avg) (Tabla 4).

Tabla 4. Resultados obtenidos algoritmo ABC_SCP_IMP

Inst.	BKS	ABC_SCP base		ABC_SCP IMP		Inst.	BKS	ABC_SCP base		ABC_SCP IMP	
		Best	*Avg*	*Best*	*Avg*			*Best*	*Avg*	*Best*	*Avg*
NRE1	29	29.0	29.0	29.0	29.0	*NRG1*	176	176.0	176.0	176.0	176.0
NRE2	30	30.0	30.0	30.0	30.0	*NRG2*	154	155.0	155.0	155.0	155.0
NRE3	27	27.0	27.0	27.0	27.0	*NRG3*	166	**166.0**	**166.0**	166.0	166.2
NRE4	28	28.0	28.0	28.0	28.0	*NRG4*	168	168.0	168.0	168.0	168.0
NRE5	28	28.0	28.0	28.0	28.0	*NRG5*	168	168.0	168.0	168.0	168.0
NRF1	14	14.0	14.0	14.0	14.0	*NRH1*	63	63.0	63.9	63.0	63.9
NRF2	15	15.0	15.0	15.0	15.0	*NRH2*	63	63.0	63.9	63.0	63.9
NRF3	14	14.0	14.0	14.0	14.0	*NRH3*	59	59.0	59.0	59.0	59.0
NRF4	14	14.0	14.0	14.0	14.0	*NRH4*	58	58.0	58.0	58.0	58.0
NRF5	13	13.0	13.7	**13.0**	**13.6**	*NRH5*	55	55.0	55.0	55.0	55.0

Al observar los resultados, estos son iguales en 18 de las 20 instancias (grupos NRE y NRH, instancias NRF1 a NRF4, NRG1, NRG2, NRG4, NRG5). En la instancia NRF5 se logró un resultado menor reportado en 0.1 unidades y en la instancia NRG3 se obtuvo un resultado 0.2 unidades mayor. En general, se observa que los resultados son similares a los reportados por el estudio de referencia [10].

Como resultado de la segunda iteración se obtuvo el algoritmo modificado. En la Tabla 5 se muestran las variantes resultado de la modificación del algoritmo ABC_SCP_IMP con los métodos de inicialización y búsqueda local seleccionados. Por ejemplo, ABC_SCP_IMP_RM_ILS identifica al algoritmo modificado que aplica el método de inicialización Random y el de búsqueda local basado en ponderación de filas RWLS.

Tabla 5. Variantes obtenidas al modificar el algoritmo ABC_SCP

Método de inicialización	Método de búsqueda local	
	IterLS	RWLS
Random	ABC_SCP_IMP_RM_ILS	ABC_SCP_IMP_RM_RLS
RHeuristic	ABC_SCP_IMP_RH_ILS	ABC_SCP_IMP_RH_RLS
IterConstruct	ABC_SCP_IMP_IC_ILS	ABC_SCP_IMP_IC_RLS

Los resultados de las variantes del algoritmo ABC_SCP_IMP modificado con los métodos de inicialización y con el método de búsqueda local RWLS se muestran en la Tabla 6, en la cual se resaltan los del algoritmo ABC_SCP_IMP que fueron mejorados en el valor Avg, los más cercanos al valor Best y aquellos que no fueron mejorados. A su vez, en esta tabla se observa que las variantes del método RWLS lograron resultados óptimos en los grupos de instancias de OR-Library, alcanzando valores Best y Avg iguales al BKS en 14 instancias y mejorándolos en tres instancias específicas: NRF5, NRG2 y NRH2. La variante que se destacó en la mayoría de las instancias fue ABC_SCP_IMP_RH_RLS. Mientras que, en algunas instancias específicas, otras variantes como ABC_SCP_IMP_IC_RLS también lograron mejoras significativas o resultados cercanos a los óptimos.

De igual forma, se verificaron los resultados de las variantes del algoritmo ABC_SCP_IMP modificado con los métodos de inicialización y con el método de búsqueda local IterLS, que se muestran en la Tabla 7, en la cual se resaltan los valores del algoritmo ABC_SCP_IMP que fueron superiores con respecto al valor Avg, los más cercanos al valor Best y los que no fueron mejorados.

Tabla 6. Resultados variantes que integran el método de búsqueda local RWLS

Instancia	BKS	ABC_SCP _IMP		ABC_SCP _IMP_RM_RLS		ABC_SCP _IMP_RH_RLS		ABC_SCP _IMP_IC_RLS	
		Best	Avg	Best	Avg	Best	Avg	Best	Avg
NRE1	29	29.0	29.0	29.0	29.0	29.0	29.0	29.0	29.0
NRE2	30	30.0	30.0	30.0	30.0	30.0	30.0	30.0	30.0
NRE3	27	27.0	27.0	27.0	27.0	27.0	27.0	27.0	27.0
NRE4	28	28.0	28.0	28.0	28.0	28.0	28.0	28.0	28.0
NRE5	28	28.0	28.0	28.0	28.0	28.0	28.0	28.0	28.0
NRF1	14	14.0	14.0	14.0	14.0	14.0	14.0	14.0	14.0
NRF2	15	15.0	15.0	15.0	15.0	15.0	15.0	15.0	15.0
NRF3	14	14.0	14.0	14.0	14.0	14.0	14.0	14.0	14.0
NRF4	14	14.0	14.0	14.0	14.0	14.0	14.0	14.0	14.0
NRF5	13	13.0	13.6	13.0	13.6	13.0	**13.2**	13.0	**13.2**
NRG1	176	176.0	176.0	176.0	176.0	176.0	176.0	176.0	176.0
NRG2	154	155.0	155.0	155.0	155.0	154.0	**154.8**	155.0	155.0
NRG3	166	166.0	166.2	**166.0**	**167.5**	**166.0**	**167.5**	167.0	168.7
NRG4	168	168.0	168.0	169.0	169.4	168.0	169.2	168.0	**168.9**
NRG5	168	168.0	168.0	168.0	168.0	168.0	168.0	168.0	168.0
NRH1	63	63.0	63.9	64.0	64.0	64.0	64.0	63.0	**63.8**
NRH2	63	63.0	63.9	63.0	**63.8**	63.0	**63.7**	63.0	**63.6**
NRH3	59	59.0	59.0	59.0	59.0	59.0	59.0	59.0	59.0
NRH4	58	58.0	58.0	58.0	58.0	58.0	58.0	58.0	58.0
NRH5	55	55.0	55.0	55.0	55.0	55.0	55.0	55.0	55.0

Las variantes que aplican el método de búsqueda local IterLS lograron resultados óptimos en las instancias del repositorio OR-Library, destacándose la variante ABC_SCP_IMP_RH_ILS, que igualó los valores Best y Avg del BKS en 15 instancias y mejoró los resultados del algoritmo base en tres casos (NRG2, NRH1 y NRH2). Aunque en las instancias NRG3 y NRG4 ninguna variante superó al algoritmo base, en otros casos, variantes lograron mejoras, especialmente en la instancia NRF5, donde todas igualaron al BKS.

Con base en los resultados obtenidos por la variante ABC_SCP_IMP_RH_ILS, como se observó en las Tablas 6 y 7, esta variante fue seleccionada para realizar un afinamiento de parámetros en busca de mejorar los resultados obtenidos.

Tabla 7. Resultados variantes que integran el método de búsqueda local IterLS

Instancia	BKS	ABC_SCP _IMP		ABC_SCP _IMP_RM_ILS		ABC_SCP _IMP_RH_ILS		ABC_SCP _IMP_IC_ILS	
		Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg
NRE1	29	29.0	29.0	29.0	29.0	29.0	29.0	29.0	29.0
NRE2	30	30.0	30.0	30.0	30.0	30.0	30.0	30.0	30.0
NRE3	27	27.0	27.0	27.0	27.0	27.0	27.0	27.0	27.0
NRE4	28	28.0	28.0	28.0	28.0	28.0	28.0	28.0	28.0
NRE5	28	28.0	28.0	28.0	28.0	28.0	28.0	28.0	28.0
NRF1	14	14.0	14.0	14.0	14.0	14.0	14.0	14.0	14.0
NRF2	15	15.0	15.0	15.0	15.0	15.0	15.0	15.0	15.0
NRF3	14	14.0	14.0	14.0	14.0	14.0	14.0	14.0	14.0
NRF4	14	14.0	14.0	14.0	14.0	14.0	14.0	14.0	14.0
NRF5	13	13.0	13.6	**13.0**	**13.0**	**13.0**	**13.0**	**13.0**	**13.0**
NRG1	176	176.0	176.0	176.0	176.0	176.0	176.0	176.0	176.0
NRG2	154	155.0	155.0	155.0	155.0	**154.0**	**154.9**	155.0	155.0
NRG3	166	166.0	166.2	166.0	167.1	166.0	167.5	**166.0**	**166.9**
NRG4	168	168.0	168.0	**168.0**	**168.3**	168.0	168.8	168.0	168.6
NRG5	168	168.0	168.0	168.0	168.0	168.0	168.0	168.0	168.0
NRH1	63	63.0	63.9	**63.0**	**63.8**	**63.0**	**63.8**	63.0	63.9
NRH2	63	63.0	63.9	63.0	63.7	63.0	63.6	**63.0**	**63.4**
NRH3	59	59.0	59.0	59.0	59.0	59.0	59.0	59.0	59.0
NRH4	58	58.0	58.0	58.0	58.0	58.0	58.0	58.0	58.0
NRH5	55	55.0	55.0	55.0	55.0	55.0	55.0	55.0	55.0

D. Prueba de la solución

En esta etapa, se realizó la evaluación de calidad del algoritmo ABC_SCP_IMP_RH_ILS para lo cual se utilizó el conjunto de datos del repositorio OR-Library para el PCC de mediana y gran escala. La Tabla 8 muestra los grupos de PCC, el número de instancias que los conforman y el número de filas (*m*) y de columnas (*n*) que los componen. En la presente investigación se abordaron los denominados de gran escala, conformados por los grupos NRE, NRF, NRG y NRH que contienen la mayor cantidad de filas y columnas.

Tabla 8. Grupos de PCC del repositorio OR-Library

Grupo(s)	No. Prob.	Filas (*m*)	Columnas (*n*)	Grupo(s)	No. Prob.	Filas (*m*)	Columnas (*n*)
4	10	200	1000	*C, D*	5	400	4000
5	10	200	2000	*NRE, NRF*	5	500	5000
6	5	200	1000	*NRG, NRH*	5	1000	10000
A, B	5	300	3000	-	-	-	-

La Tabla 9 muestra el afinamiento de parámetros realizado al algoritmo ABC_SCP_IMP_RH_ILS. Debido a la complejidad de las instancias de OR-Library los experimentos requieren de mucho tiempo de ejecución, por esto, no fue posible realizar un afinamiento de todos los parámetros involucrados en dicho algoritmo.

Tabla 9. Afinamiento de parámetros algoritmo ABC_SCP_IMP_RH_ILS

Parámetros algoritmo modificado			
Paso	**Parámetro**	**Descripción**	**Valor**
Fase abejas observadoras	*ONLOOKER_BEES*	Número de abejas observadoras	50
Parámetros Método IterLS			
Método	**Parámetro**	**Descripción**	**Valor**
IterLS	*COL_DROP_1*	Número de columnas a eliminar de forma aleatoria si $N>35$	20
IterLS	*COL_DROP_2*	Número de columnas a eliminar de forma aleatoria si $N\leq35$	6

Como el tiempo de ejecución de este algoritmo se incrementó considerablemente con respecto al ABC_SCP_IMP, se realizó una experimentación con pocas ejecuciones para identificar los parámetros que influían en este incremento. Uno de ellos fue el número de abejas exploradoras (ONLOOKER_BEES) cuyo valor inicial en ABC_SCP era de 150, ejecutando el algoritmo con valores de 140, 120, 100, 80, 60, 50, 40, encontrando con el valor 50 una disminución en el tiempo de ejecución y manteniendo la calidad de los resultados.

Con respecto al método de búsqueda local IterLS, para el afinamiento de los parámetros para eliminar columnas COL_DROP_1 y COL_DROP_2, se partió del valor en el algoritmo ABC_SCP y se realizó una variación de este buscando aumentar la diversidad en las soluciones generadas. Para el valor inicial fue 12, el cual se incrementó así: 14, 16, 18, 20, 22 y 24. Para COL_DROP_2 el valor inicial fue 5 y se varió en 1, es decir 4 y 6. Finalmente, la combinación COL_DROP_1= 20 y COL_DROP_2 = 6 obtuvo una mayor calidad de los resultados.

La evaluación del algoritmo modificado se realizó inicialmente comparando los resultados obtenidos con el algoritmo base (ABC_SCP base) y el algoritmo implementado (ABC_SCP_IMP) y, posteriormente, con otros algoritmos del estado del arte.

3. RESULTADOS

En esta investigación se obtuvieron dos resultados: el algoritmo modificado propuesto y los resultados de este algoritmo comparados con otros del estado del arte.

A. Algoritmo de colonia de abejas artificiales modificado para el PCC

El algoritmo modificado ABC_SCP_IMP_RH_ILS propuesto en esta investigación se muestra en la Figura 1, resaltando los pasos en los que se reemplazó el método de inicialización y de búsqueda local del algoritmo ABC_SCP por los seleccionados en este caso. La implementación del algoritmo fue realizada en lenguaje Java (https://shorturl.at/6FvRz).

El algoritmo comienza con la creación de variables para almacenar la población de fuentes de alimento FA_K, el número de intentos de mejora de una solución $trial_k$ y los valores de aptitud de las fuentes de alimento $fitness_k$ (líneas 1-4). Luego, para cada fuente de alimento FA_K, se inicializa una nueva solución aplicando el método de inicialización aleatoria con operador heurístico RHeuristic y se calcula su valor de aptitud para almacenarlo en $fitness_K$ (líneas 2-11). Una vez creadas las fuentes de alimento, se memoriza

el mejor valor de aptitud de la población y se crean las variables *bestSol* y bestFitness para almacenar la solución y el valor de aptitud más óptimos logrados en la ejecución, iniciándolas con valores iniciales y aplicando el procedimiento de memorización de la solución óptima global buscando la mejor solución de la población creada (líneas 10-12).

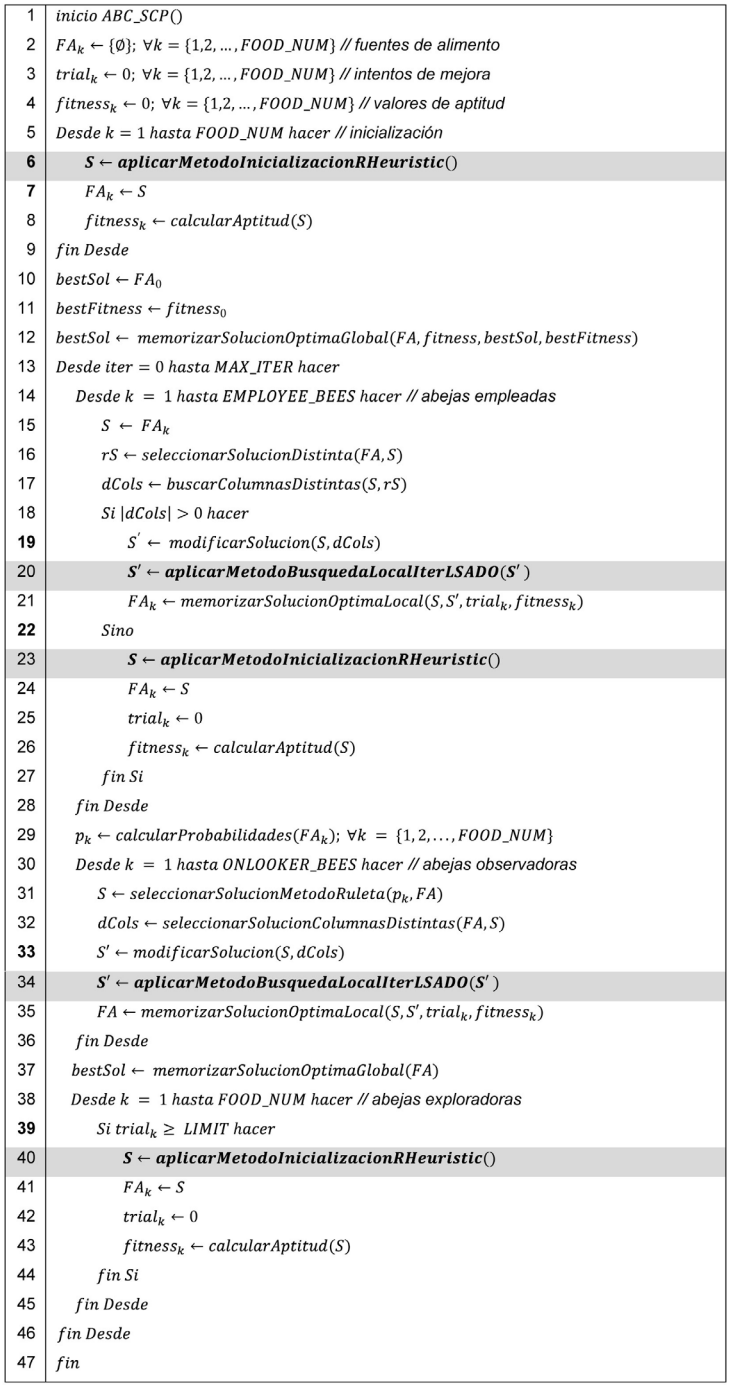

Figura 1. *Algoritmo modificado ABC_SCP_RH_ILS*

A continuación, inicia un ciclo que se ejecuta en un máximo de *MAX_ITER* iteraciones. Cada iteración comienza con la fase de abejas empleadas donde se recorren las soluciones *S* asociadas a las fuentes de alimento FA_K (líneas 13-15). Para cada *S*, se selecciona otra solución aleatoria *randS*, se buscan columnas en *randS* que fueran distintas a las que contenía *S* y se almacenan en la lista *dCols*. Si el número de *dCols* es mayor a cero, se procede a modificar *S* para obtener una nueva solución *S'* (líneas 16-19), la cual se intenta mejorar con la aplicación del método de búsqueda local iterada IterLS adaptado (líneas 20-21).

La solución *S* obtenida se compara con la solución actual, aplicando un procedimiento de memorización de soluciones local (líneas 22-24). Si no se encuentran columnas distintas, lo que indica una colisión entre soluciones, la abeja empleada se convertía en exploradora, abandona la fuente de alimento actual, explora el espacio de búsqueda y selecciona una nueva fuente para volver a ser empleada, reemplazando la solución actual *S* por una nueva generación (líneas 22-28).

Posteriormente, se calcula las probabilidades de selección de las fuentes de alimento *FA* aplicando la ecuación correspondiente para iniciar la fase de abejas exploradoras, seleccionando una solución mediante el método de la ruleta a partir de las probabilidades p_k calculadas (líneas 29-31). Se busca una solución *S'* que contuviera columnas distintas a las de *S* y estas se almacenan en *S'* (líneas 31-32).

A diferencia de la fase de abejas empleadas que detecta colisiones, en esta fase se busca una solución *S'* hasta que contuviera columnas distintas descartando las colisiones. Una vez se tiene la lista *dCols* de columnas distintas, se aplica los procedimientos de modificación de la solución (líneas 33-34), aplicación del método de búsqueda local iterada IterLS adaptado (líneas 35-36) y memorización de la solución óptima local (líneas 33-36), similares a los de la fase de abejas empleadas.

Finalmente, se busca en la población de fuentes de alimento la solución óptima global aplicando nuevamente el procedimiento de memorización correspondiente (líneas 37-38).

Después, comienza la fase de abejas exploradoras recorriendo las fuentes de alimento en busca de aquellas con un número de intentos mayor al parámetro *LIMIT*. Las fuentes que cumplen la condición son abandonadas y sustituidas por nuevas fuentes, asociadas con la inicialización de soluciones novedosas con el método RHeuristic, su almacenamiento en la fuente FA_K correspondiente y el cálculo de su valor de aptitud, reiniciando el número de intentos de mejora $trial_k$ a cero (líneas 39-47). El algoritmo finaliza la iteración y realiza nuevas iteraciones hasta cumplir con el número máximo de iteraciones *MAX_ITER*.

B. Evaluación

La Tabla 10 muestra los resultados del algoritmo modificado propuesto ABC_SCP_IMP_RH_ILS con las 20 instancias de prueba respecto al BKS, el Best y el Avg, los cuales se compararon con ABC_SCP base y ABC_SCP_IMP.

Como se observa, el algoritmo ABC_SCP_IMP_RH_ILS logra valores Avg iguales a los BKS en 18 instancias (NRE1 a NRE5, NRF1 a NRF5, NRG1, NRG2, NRG4, NRG5, NRH2 a NRH5) y resultados cercanos en dos instancias (NRG3, NRH1). Se destacan los resultados de tres instancias (NRF5, NRG2, NRH2) en los que ABC_SCP_IMP_RH_ILS supera los resultados del algoritmo base y el algoritmo base implementado logrando el valor Avg igual al BKS.

Tabla 10. Resultados algoritmo ABC_SCP_IMP_RH_ILS

Inst.	BKS	ABC_SCP base		ABC_SCP_IMP		ABC_SCP_IMP _RH_ILS	
		Best	Avg	Best	Avg	Best	Avg
NRE1	29	29.0	29.0	29.0	29.0	29.0	29.0
NRE2	30	30.0	30.0	30.0	30.0	30.0	30.0
NRE3	27	27.0	27.0	27.0	27.0	27.0	27.0
NRE4	28	28.0	28.0	28.0	28.0	28.0	28.0
NRE5	28	28.0	28.0	28.0	28.0	28.0	28.0
NRF1	14	14.0	14.0	14.0	14.0	14.0	14.0
NRF2	15	15.0	15.0	15.0	15.0	15.0	15.0
NRF3	14	14.0	14.0	14.0	14.0	14.0	14.0
NRF4	14	14.0	14.0	14.0	14.0	14.0	14.0
NRF5	13	13.0	13.7	13.0	13.6	**13.0**	**13.0**
NRG1	176	176.0	176.0	176.0	176.0	176.0	176.0
NRG2	154	155.0	155.0	155.0	155.0	**154.0**	**154.0**
NRG3	166	**166.0**	**166.0**	166.0	166.2	166.0	166.1
NRG4	168	168.0	168.0	168.0	168.0	168.0	168.0
NRG5	168	168.0	168.0	168.0	168.0	168.0	168.0
NRH1	63	63.0	63.9	63.0	63.9	**63.0**	**63.1**
NRH2	63	63.0	63.9	63.0	63.9	**63.0**	**63.0**
NRH3	59	59.0	59.0	59.0	59.0	59.0	59.0
NRH4	58	58.0	58.0	58.0	58.0	58.0	58.0
NRH5	55	55.0	55.0	55.0	55.0	55.0	55.0

También, el resultado de la instancia NRH1 que es mejorado con respecto a los otros dos algoritmos con un valor Avg de mayor cercanía al BKS. Por último, el resultado de la instancia NRG3 fue mejorado con respecto a ABC_SCP_IMP, pero no superó el valor del algoritmo base.

Sumado a lo anterior, los resultados del algoritmo ABC_SCP_IMP_RH_ILS se compararon con otros reportados por algoritmos del estado del arte (ver Tabla 11): optimización con enjambre de partículas saltadoras (Jumping Particle Swarm Optimization, JSPO) [24] y algoritmo memético codificado híbrido (Hybrid Encoded Memetic Approach, HEMA) [11]. Los resultados del algoritmo JSPO solo reportaron el valor de aptitud promedio obtenido (Avg), el cual fue utilizado en la comparación.

Considerando que los algoritmos del estado del arte realizaron 30 ejecuciones para obtener los resultados reportados, el algoritmo ABC_SCP_IMP_RH_ILS es ejecutado el mismo número de veces (RUNTIME) con el fin de realizar la comparación en condiciones de ejecución similares.

Tabla 11. Comparación de resultados con otros algoritmos del estado del arte

Inst.	BKS	JSPO		HEMA		ABC_SCP_IMP_RH_ILS	
		Best	Avg	Best	Avg	Best	Avg
NRE1	29	-	29.0	29.0	29.0	29.0	29.0
NRE2	30	-	30.0	30.0	30.0	30.0	30.0
NRE3	27	-	27.0	27.0	27.0	27.0	27.0
NRE4	28	-	28.0	28.0	28.0	28.0	28.0
NRE5	28	-	28.0	28.0	28.0	28.0	28.0
NRF1	14	-	14.0	14.0	14.0	14.0	14.0
NRF2	15	-	15.0	15.0	15.0	15.0	15.0
NRF3	14	-	14.0	14.0	14.0	14.0	14.0
NRF4	14	-	14.0	14.0	14.0	14.0	14.0
NRF5	13	-	13.0	13.0	13.0	13.0	13.0
NRG1	176	-	176.0	176.0	176.0	176.0	1760
NRG2	154	-	155.0	154.0	154.2	**154.0**	**154.0.**
NRG3	166	-	167.2	166.0	166.6	**166.0**	**166.1**
NRG4	168	-	168.2	168.0	168.2	**168.0**	**168.0**
NRG5	168	-	168.0	168.0	168.0	168.0	168.0
NRH1	63	-	64.0	**63.0**	**63.0**	63.0	63.1
NRH2	63	-	63.0	63.0	63.0	63.0	63.0
NRH3	59	-	59.2	59.0	59.0	59.0	59.0
NRH4	58	-	58.3	58.0	58.0	58.0	58.0
NRH5	55	-	55.0	55.0	55.0	55.0	55.0

Los resultados de la Tabla 11 muestran que los tres algoritmos logran el valor Avg igual al BKS en 14 instancias (NRE1 a NRE5, NRF1 a NRF5, NRG1, NRG5, NRH2, NRH5); que ABC_SCP_IMP_RH_ILS y HEMA logran el valor promedio igual al BKS, superando a JSPO, en dos instancias (NRH3 y NRH4), y que en tres instancias (NRG2, NRG3 y NRG4) ABC_SCP_IMP_RH_ILS supera los resultados de los otros algoritmos, alcanzando el valor de BKS en el valor Avg en dos instancias (NRG2, NRG4) y obteniendo un mejor valor Avg en uno (NRG3).

A su vez, en la instancia NRH1, el mejor resultado lo reporta el algoritmo HEMA con el valor Avg igual al BKS, seguido del algoritmo ABC_SCP_IMP_RH_ILS y, en último lugar, el algoritmo JSPO.

C. Limitaciones

El algoritmo modificado ABC_SCP_IMP_RH_ILS presentó resultados competitivos en comparación con otros algoritmos del estado del arte, pero no logró el valor de BKS en todos los casos. Esto se puede observar en la instancia NRG3 donde el algoritmo base ABC_SCP obtuvo el valor del BKS (166) y nuestro algoritmo 166.1, que es ligeramente inferior. De manera similar, en la instancia NRH1 el algoritmo HEMA obtuvo el valor del BKS (63), y nuestro algoritmo 63.1, también ligeramente inferior.

Aunque el ABC_SCP_IMP_RH_ILS puede ser efectivo para ciertos problemas, su rendimiento es dependiente de la naturaleza específica de los mismos. Esto lo explica el teorema No Free Lunch (*NFL*)

[25], el cual afirma que el rendimiento de un algoritmo no puede ser garantizado de manera consistente en todos los problemas. Así, la efectividad de nuestro algoritmo se debe a su adaptación a las características particulares de las instancias de OR-Library, lo que no implica que se logré el BKS en otras instancias de prueba o, que supere los resultados de otros algoritmos aplicados al PCC.

4. CONCLUSIONES

Para resolver el PCC en este artículo se propone el algoritmo ABC_SCP_IMP_RH_ILS, el cual es una modificación del algoritmo base de colonias de abejas artificiales del estado del arte ABC_SCP con el método de inicialización aleatorio con operador heurístico (RHeuristic) para la generación de soluciones, a partir de la identificación de mejores filas y columnas y el método de búsqueda local iterada (IterLS) para aplicar el reinicio y la penalización de columnas en las soluciones e intentar mejorarlas.

Para la selección de los métodos de inicialización y búsqueda local, que serían modificados en el algoritmo base ABC_SCP, se llevó a cabo un mapeo sistemático de la literatura. Tomando como base estos resultados y algunos criterios definidos por los autores, fueron seleccionados métodos de inicialización que generaran soluciones utilizando componentes de aleatoriedad para la exploración del espacio de búsqueda a un costo computacional menor y redundancia, favoreciendo la explotación de soluciones del algoritmo.

Al mismo tiempo, fueron seleccionados los métodos de búsqueda local basados en ponderación de filas y búsqueda local iterada por reportar en los estudios primarios los resultados más cercanos al BKS en comparación con otros métodos identificados, por disponer de la especificación necesaria para su adaptación al algoritmo base y por mejorar la explotación del algoritmo al implementar estrategias de ponderación de filas y asignación de puntajes a las columnas.

En la evaluación del algoritmo propuesto ABC_SCP_IMP_RH_ILS con el algoritmo base ABC_SCP y el implementado por los autores (ABC_SCP_IMP) se encontró una mejora en el valor de aptitud promedio (Avg) de cuatro instancias (NRF5, NRG2, NRH1, NRH2), logrando igualar el valor promedio de la mejor solución conocida (BKS) en tres instancias (NRF5, NRG2, NRH2).

Esto se alcanzó adicionando en la inicialización de soluciones una estrategia de identificación de mejores filas y columnas mediante el método RHeuristic y en la búsqueda local de soluciones una estrategia codiciosa con penalización de columnas mediante el método IterLS. En conjunto, estos métodos agregan componentes que logran mayor diversificación en la evolución que el algoritmo base ABC_SCP.

Además, se realizó una evaluación del algoritmo propuesto ABC_SCP_IMP_RH_ILS con otros algoritmos del estado del arte en la cual se observa la mejora de resultados del algoritmo de enjambre de partículas saltadoras (JPSO) en seis instancias (NRG2, NRG3, NRG4, NRH1, NRH3, NRH4) y la mejora en los resultados del algoritmo memético codificado híbrido (HEMA) en tres instancias (NRG2, NRG3, NRG4). Con estos resultados, se muestra que el ABC_SCP_IMP_RH_ILS es un algoritmo metaheurístico competitivo en el ámbito de investigaciones relacionadas con el PCC.

En futuros estudios, buscando mejorar los resultados obtenidos por el algoritmo ABC_SCP_IMP_RH_ILS en las instancias de OR-Library, se pueden identificar en la literatura otros métodos de inicialización y búsqueda local que se adapten al algoritmo para resolver este problema de cobertura de conjuntos. También, es importante realizar un análisis experimental de la eficiencia computacional de nuestro algoritmo comparándolo con el ABC_SCP. Por último, se podría ejecutar el ABC_SCP_IMP_RH_ILS en

instancias de prueba de otros repositorios, realizando las adaptaciones requeridas para analizar los resultados obtenidos.

CONTRIBUCIÓN DE LOS AUTORES

Nelson-Enrique Quemá-Taimbud: Conceptualización, Investigación, Metodología, Software, Redacción-borrador original.

Martha-Eliana Mendoza-Becerra: Conceptualización, Metodología, Supervisión, Redacción-revisión y edición.

AGRADECIMIENTOS

Los autores agradecen el aporte a la Universidad del Cauca para la realización de esta investigación.

REFERENCIAS

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, *Introduction to Algorithms*, Third edition. Cambridge, MA: MIT Press, 2009.

[2] Y. Deng, Y. Zhang, J. Pan, "Optimization for Locating Emergency Medical Service Facilities: A Case Study for Health Planning from China," *Risk Management and Healthcare Policy*, vol. 14, pp. 1791-1802, Apr. 2021. https://doi.org/10.2147/RMHP.S304475

[3] J. Purnomo, Z. Fanani, T. Domai, A. Hariswanto, "The Model For Determining Location Of Naval Base Using AHP Method And Set Covering Problem," *Russian Journal of Agricultural and Socio-economic Sciences*, vol. 101, no. 5, pp. 122-131, May 2020. https://doi.org/10.18551/rjoas.2020-05.13

[4] Q. Yong, D. Liu, G. Li, W. Wu, W. Sun, S. Liu, "Reducing exposure to COVID-19 by improving access to fever clinics: an empirical research of the Shenzhen area of China," *BMC Health Services Research.*, vol. 21, no. 1, pp. 1-10, Sep. 2021. https://doi.org/10.1186/S12913-021-06831-4

[5] L. Jorquera, P. Valenzuela, M. Valenzuela, H. Pinto, "A Binary Ant Lion Optimisation Algorithm Applied to the Set Covering Problem," in *Artificial Intelligence Methods in Intelligent Algorithms*, R. Silhavy, Ed. Cham: Springer International Publishing, 2019, pp. 156-167. https://doi.org/10.1007/978-3-030-19810-7_16

[6] B. Crawford, R. Soto, A. Peña, G. Astorga, "A Binary Grasshopper Optimisation Algorithm Applied to the Set Covering Problem," in *Cybernetics and Algorithms in Intelligent Systems*, R. Silhavy, Ed. Cham: Springer International Publishing, 2019, pp. 1-12. https://doi.org/10.1007/978-3-319-91192-2_1

[7] B. Crawford *et al.*, "Binary Fruit Fly Swarm Algorithms for the Set Covering Problem," *Computers, Materials and Continua*, vol. 71, no. 2, pp. 4295-4318, Jan. 2022. https://doi.org/10.32604/cmc.2022.023068

[8] B. Crawford *et al.*, "A teaching-learning-based optimization algorithm for the weighted set-covering problem," *Tehnički Vjesnik*, vol. 27, no. 5, pp. 1678-1684, 2020. https://doi.org/10.17559/TV-20180501230511

[9] B. Crawford, R. Soto, G. Cabrera, A. Salas-Fernández, F. Paredes, "Using a Social Media Inspired Optimization Algorithm to Solve the Set Covering Problem," in *Social Computing and Social Media.*

Design, Human Behavior and Analytics, G. Meiselwitz, Ed. Cham: Springer International Publishing, 2019, pp. 43-52. https://doi.org/10.1007/978-3-030-21902-4_4

[10] S. Sundar, A. Singh, "A hybrid heuristic for the set covering problem," *Operations Research*, vol. 12, no. 3, pp. 345-365, Nov. 2012. https://doi.org/10.1007/s12351-010-0086-y

[11] F. Xu, J. Li, "A hybrid encoded memetic algorithm for set covering problem," in *Proceedings 10th International Conference on Advanced Computational Intelligence*, ICACI 2018, IEEE, Xiamen, China, 2018, pp. 552-557. https://doi.org/10.1109/ICACI.2018.8377519

[12] B. Crawford *et al.*, "A binary monkey search algorithm variation for solving the set covering problem," *Nature Computational*, vol. 19, no. 4, pp. 825-841, Dic. 2020. https://doi.org/10.1007/s11047-019-09752-8

[13] J. García, B. Crawford, R. Soto, P. García, "A Multi Dynamic Binary Black Hole Algorithm Applied to Set Covering Problem," in *Harmony Search Algorithm*, J. Del Ser, Ed. Singapore: Springer Singapore, 2017, pp. 42-51. https://doi.org/10.1007/978-981-10-3728-3_6

[14] E. Meca Castro, "Two Neighbourhood-based Approaches for the Set Covering Problem," *U.Porto Journal of Engineering.*, vol. 5, no. 1, pp. 1-15, Feb. 2019. https://doi.org/10.24840/2183-6493_005.001_0001

[15] M. F. L. Schmitt, M. H. Mulati, A. A. Constantino, F. Hernandes, T. A. Hild, "Ant-set: A subset-oriented ant colony optimization algorithm for the set covering problem," *Journal of Universal Computer Science*, vol. 26, no. 2, pp. 293-316, Feb. 2020. https://doi.org/10.3897/jucs.2020.016

[16] B. Crawford, R. Soto, G. Astorga, J. Lemus-Romani, S. Misra, J.-M. Rubio, "An Adaptive Intelligent Water Drops Algorithm for Set Covering Problem," in *19th International Conference on Computational Science and Its Applications*, IEEE, St. Petersburg, Russia, 2019, pp. 39-45. https://doi.org/10.1109/ICCSA.2019.000-6

[17] R. Soto *et al.*, "Adaptive Black Hole Algorithm for Solving the Set Covering Problem," *Mathematical Problems in Engineering*, vol. 2018, pp. 1-23, Oct. 2018. https://doi.org/10.1155/2018/2183214

[18] G. Lin, J. Luo, X. Chen, H. Xu, M. Xu, "An Adaptive Feasible and Infeasible Search Algorithm for Solving the Set Covering Problem," in *Advances in Natural Computation, Fuzzy Systems and Knowledge Discovery*, H. Meng, T. Lei, M. Li, K. Li, N. Xiong, y L. Wang, Eds. Cham: Springer International Publishing, 2021, pp. 271-278. https://doi.org/10.1007/978-3-030-70665-4_31

[19] B. Crawford, R. Soto, E. Monfroy, G. Astorga, J. García, E. Cortes, "A Meta-Optimization Approach for Covering Problems in Facility Location," in *Communications in Computer and Information Science*, J. Figueroa-García, E. López-Santana, J. Villa-Ramírez, and R. Ferro-Escobar, Eds. Cham: Springer Verlag, 2017, pp. 565-578. https://doi.org/10.1007/978-3-319-66963-2_50

[20] K. S. Pratt, K. S. Pratt, H. Bright, "Design Patterns for Research Methods: Iterative Field Research," in *AAAI Spring Symposia*, California, United States, 2009, pp. 1-7.

[21] N.-E. Quemá-Taimbud, M.-E. Mendoza-Becerra, O.-F. Bedoya-Leyva, "Initialization and Local Search Methods Applied to the Set Covering Problem: A Systematic Mapping," *Revista Facultad de Ingeniería*, vol. 32, no. 63, e15235, Feb. 2023. https://doi.org/10.19053/01211129.v32.n63.2023.15235

[22] H. Razip, N. Zakaria, "Genetic Algorithm with Approximation Algorithm Based Initial Population for the Set Covering Problem," in *Proceedings of International Conference on Communication and Computational Technologies*, Singapore: Springer, 2021, pp. 59-78. https://doi.org/10.1007/978-981-16-3246-4_6

[23] J. E. Beasley, *OR-Library: A collection of test data sets for a variety of operations research (or) problems*. http://people.brunel.ac.uk/~mastjjb/jeb/orlib/scpinfo.html

[24] S. Balaji, N. Revathi, "A new approach for solving set covering problem using jumping particle swarm optimization method," *Natural Computacional*, vol. 15, no. 3, pp. 503-517, Sep. 2016. https://doi.org/10.1007/s11047-015-9509-2

[25] D. H. Wolpert, W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67-82, Apr. 1997. https://doi.org/10.1109/4235.585893

Nelson-Enrique Quemá-Taimbud,
Martha-Eliana Mendoza-Becerra

Algoritmo de colonia de abejas artificiales modificado para el problema de cobertura de conjuntos
Modified Artificial Bee Colony Algorithm for the Set Covering Problem

Revista Facultad de Ingeniería
vol. 33, núm. 68, e18079, 2024
Universidad Pedagógica y Tecnológica de Colombia,
ISSN: 0121-1129
ISSN-E: 2357-5328

DOI: https://doi.org/10.19053/01211129.v33.n68.2024.18079