# ESTUDIO DE VARIACIÓN POBLACIONAL USANDO HARMONY SEARCH PARA LA RESOLUCIÓN DEL SET COVERING PROBLEM

\*Juan Salas1, Mauricio Mora1, Hugo Barriga1, José-Miguel Rubio1, Broderick Crawford2

1 Universidad Tecnológica de Chile INACAP

2 Pontificia Universidad Católica de Valparaíso

{jsalasf,mmorad,hbarriga,jrubiol}@inacap.cl

{broderick.crawford}@pucv.cl

# RESUMEN

El presente trabajo se titula: Estudio de variación poblacional usando Harmony Search (HS) para la resolución del Set Covering Problem (SCP). Su aporte principal consiste en demostrar los efectos del tamaño de la población de soluciones, en la obtención de resultados a un problema de optimización. Los efectos del presente estudio son relevantes, debido a que la mayoría de las metaheurísticas actuales usan población de soluciones de tamaño fijo. Nuestro equipo logró comprobar que una población de soluciones variable permite obtener mejores resultados en el 93.3% de los casos. Este documento se estructura de la siguiente forma: En la sección introducción se realiza una revisión y justificación de este estudio; en la sección materiales y métodos se define el marco de trabajo de esta investigación; en la sección resultados se presentan los valores obtenidos de los diferentes experimentos y finalmente en la sección discusión y conclusión se muestran los comentarios más importantes respecto a los resultados obtenidos.

**Palabras Claves:** Metaheurística, Harmony Search, Población fija, Población Variable, Set Covering Problem.

# ABSTRACT

The present work is entitled: Study of population variation using Harmony Search (HS) for the resolution of Set Covering Problem (SCP). Its main contribution is to demonstrate the effects of the population size of solutions, in obtaining results to an optimization problem. The effects of the present study are relevant, since most of the current metaheuristics use population of fixed size solutions. Our team was able to verify that a population of variable solutions allows to obtain better results in 93.3% of cases. This document is structured as follows: In the introduction section, a review and justification of this study is carried out; in the materials and methods section, the framework of this research is defined; In the results section the values ​​obtained from the different experiments are presented and finally in the discussion and conclusion section the most important comments regarding the obtained results are shown.

**Keywords:** Metaheuristic, Harmony Search, Fixed size population, Variable population, Set Covering Problem.

# INTRODUCCIÓN

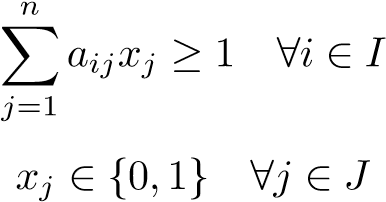
Las metaheurísticas (MH) son algoritmos de optimización muy utilizados en las ciencias de la computación y matemáticas aplicadas. Las MH se aplican a problemas de optimización que no tienen algoritmo o heurística específica que dé una solución satisfactoria; o bien cuando no es posible implementar una técnica completa dado su alto costo computacional. Este tipo de algoritmos pertenece a la Inteligencia Artificial (IA). Las MH generan interés en diversas tecnologías, industrias y servicios dada su eficiencia en resolver problemas de optimización de diferentes dominios (Torres-Jiménez & Pavón, 2014). La parametrización de la MH es crucial para resolver las instancias del Set Covering Problem (SCP) (Birattari et al., 2002). Dentro de los parámetros que recibe la MH, se encuentra el tamaño de la población de soluciones o cantidad de soluciones en memoria. La división de población de soluciones fue tratada en (Turky & Abdullah, 2014), con buenos resultados. La versión propuesta por Turky y Abdullah no trabaja con SCP. En este trabajo se identificó el efecto positivo de aumentar la cantidad de soluciones en el espacio de memoria durante la ejecución del algoritmo, mediante un nuevo operador de HS. Se comparó el comportamiento de la MH utilizando Población de soluciones Fijas (PF) contra Población de soluciones Variables (PV). Ambos métodos fueron probados contra las instancias de la librería o benchmark OR (Beasley, 1990). La meta de este trabajo consistió en determinar la mejor estrategia para solucionar el SCP ya sea usando PF o PV. El trabajo se estructura de la siguiente manera: En la sección Introducción se realiza una descripción de alto nivel de la técnica de inteligencia artificial usada y se presenta brevemente la estructura de trabajo que proporcionó los resultados que permiten hacer comparativas y conclusiones. Posteriormente en la sección Materiales y Métodos se aborda la MH Harmony Search (HS) original, pasando por una pequeña muestra de algunas variaciones presentes en otros estudios, adicionalmente se propone una nueva variación de HS con la cual se resolverá el SCP. De igual modo se describe formalmente en términos matemáticos el problema a resolver. En la sección Resultados se despliegan los valores obtenidos por la MH, los cuales son analizados en profundidad para pasar a la sección Discusión y Conclusiones.

# MATERIALES Y MÉTODOS

El SCP es un problema matemático, el cual intenta cubrir un conjunto de necesidades al menor costo posible (minimización). El SCP fue incluido en la lista de 21 problemas NP-*completos* de Karp (Karp, 2010). Existen muchos usos prácticos para este problema como son: Localización de unidades de conexión en redes vehiculares (Sarubbi et al., 2017), ubicación de instalaciones (Crawford et al., 2017), ruteo de vehículos (Cacchiani, Hemmelmayr, & Tricoire, 2014; Prata, 2016; Yakici, 2017), asignación de tráfico en sistemas de comunicación satelitales (Ribeiro, Minoux, & Penna, 1989; Ceria, Nobili, & Sassano, 1998), asignación de tripulantes en líneas aéreas (Housos & Elmroth, 1997), asignación y ruteo de trenes (Samà et al., 2015), control de acceso basado en roles (Chen & Crampton, 2009) entre otros

El SCP puede ser formulado como sigue:

(1)

Sujeto a:

(2)

(3)

Sea una matriz donde y son las filas y columnas respectivamente. Formalmente podemos decir que la columna *j* puede cubrir a la fila *i* si *aij* = 1. El vector *cj* contiene valores no negativos que representan el costo de seleccionar la columna *j* y *xj* es la variable de decisión, la cual puede tomar el valor 1 si la columna *j* es seleccionada o 0 si no.

HS es un algoritmo metaheurístico del tipo poblacional (Beheshti & Shamsuddin, 2013) como son: Artificial Bee Colony (ABC) (Crawford et al., 2014), Black Hole (BH) (Gómez et al., 2016) y Soccer League Competition (SLC) (Jaramillo et al., 2016). HS está inspirada en el proceso de búsqueda de un estado de armonía perfecto o Aesthetic Quality of Harmony (AQH) en la improvisación musical. Fue propuesto por (Geem, Kim, & Loganathan, 2001). La idea de esta MH es mimetizar el proceso realizado por músicos cuando improvisan. Las armonías que pueden ser generadas, dependen directamente del tono o rango de frecuencia del instrumento (Geem, 2009). En el proceso de improvisación musical, todos los músicos ejecutan acordes en un posible rango al mismo tiempo, para generar una armonía. Si todos los acordes generan una buena armonía, cada músico almacena en su memoria esa experiencia y posibilita la ejecución de una buena armonía en la siguiente improvisación. Lo mismo en la optimización: La solución inicial es generada aleatoriamente con las variables de decisión en el rango predefinido. Si al evaluar la solución en la función objetivo, se obtiene un buen valor, quiere decir que hay una buena oportunidad de mejorar en la próxima iteración. La función objetivo permite definir una relación de orden total entre cualquier par de soluciones del espacio de búsqueda. El óptimo global en minimización, es definido como una solución *s*∗ ∈ *S*, donde *S* corresponde al conjunto total de soluciones. *s*∗ posee el mejor fitness (menor valor) al ser evaluada en la función objetivo que cualquier otra solución del espacio de búsqueda, i.e., ∀*s* ∈ *S*, *f*(*s*∗) ≤ *f*(*s*). El fitness corresponde al valor obtenido al evaluar la solución en la función objetivo.

La MH HS, consta de 5 pasos en su versión original (Zou et al., 2011). Dichos pasos serán revisados en detalle a continuación:

Paso 1: Inicialización del problema y los parámetros del algoritmo. Los parámetros *xiL* y *xiU* y corresponden a los valores inferiores y superiores de los dominios de las variables de decisión. Los parámetros requeridos para solucionar el problema de optimización son los siguientes: Harmony Memory (HM), el cual es un espacio de memoria donde se almacenan las armonías, Harmony Memory Size (*HMS)* que corresponde al número de vectores de soluciones en HM, Harmony Memory Consideration Rate (*HMCR*) el cual determina el ratio de selección de valores desde memoria, Pitch Adjusting Rate (*PAR*) la cual determina la probabilidad de mejoras locales y finalmente el número de improvisaciones (*NI*).

Paso 2: Inicialización del espacio de memoria HM hasta completar el *HMS*. Las armonías se improvisan de acuerdo a la ecuación (4).

(4)

donde:

*t* = Iteración o improvisación actual.

*d* = *i*-ésima variable del vector solución o armonía.

*i* = *i*-ésima solución o armonía

*xdi*(*t* + 1) = Representa a la siguiente solución.

*xdiL*(*t*) = Valor inferior de la variable de decisión, dado *d* y *i.*

*xdiU*(*t*) = Valor superior de la variable de decisión, dado *d* y *i.*

*rand*() = Corresponde a un valor aleatorio de una distribución uniforme dentro del rango [0*,*1].

Paso 3: La improvisación de una nueva armonía se especifica en Algoritmo 1. De acuerdo al proceso, se generan vectores del tipo . Estos vectores componen una matriz como la Figura 1.

**Figura 1.** Matriz de espacio en memoria HM

|  |  |  |  |
| --- | --- | --- | --- |
| **Pseudo Código: Operador Improvisación**   |  | | --- | | **for** i←1 **to** n **do**  **if** ≤ HMCR **then**  = //Consideración de uso de memoria.  **if** ≤ PAR **then**  = ± r(bw) // BW corresponde al ancho de banda  **Else**  = //Selección Aleatoria |   **Algoritmo 1.** Generación de una nueva armonía en HS tradicional | **Pseudo Código: Operador Actualización**   |  | | --- | | **if** () > ()  = |   **Algoritmo 2.** Procedimiento para reemplazar peor armonía. |

Paso 4: Actualización de *HM*. Si la nueva armonía generada tiene mejor fitness que la peor armonía en *HM*, entonces se reemplaza la peor armonía en *HM* con la nueva armonía tal como se muestra en el Algoritmo 2; de lo contrario se continua con el paso siguiente.

Paso 5: Si el criterio de detención (e.g. máximo número de Improvisaciones *NI*) se ha alcanzado, entonces la ejecución del algoritmo finaliza. Por el contrario, se continua la ejecución a partir del paso 3.

Con el propósito de mejorar el desempeño en la cobertura de HS tradicional y mejorar ciertas deficiencias de esta MH, se presenta Global-Best Harmony Search (GHS), desarrollada por (Omran & Mahdavi, 2008). La característica de esta variación de HS, es la generación de la población inicial aleatoriamente usando un Proceso de Bernoulli (Bertsekas & Tsitsiklis, 2002). Adicionalmente, se genera un vector solución de manera elitista, basado en la idea de una relación de ganancia (profit ratio), donde se verifica si es mejor que *xworst* en HM entonces *xworst* es reemplazado por . La metaheurística GHS ajusta dinámicamente el parámetro PAR de acuerdo a la ecuación (5):

(5)

donde:

*PAR*(*t*) = Pitch adjustment rate para la iteración *t*.

*PARmin* = Valor mínimo que puede tomar *PAR*.

*PARmax* = Valor máximo que puede tomar *PAR*.

*NI* = Número de improvisaciones.

*t* = Improvisación o iteración actual.

HS tradicional es buena explorando el espacio de búsqueda, pero no lo es tanto a la hora de explotar (Xiang et al., 2014). No existe un equilibrio entre la exploración y la explotación. HS fue pensada para ser utilizada en un espacio de búsqueda continua y no puede ser aplicada para resolver problemas de optimización combinatoria discretos. Para mejorar estos y otros aspectos se propone la variación de HS denominada Binary Global-Best Harmony Search (BGBHS). BGBHS, hace uso de los siguientes operadores: Pitch Adjustment Rate (PAR), Harmony Memory Consideration Rate (HMCR), reparación ADD y DROP, mecanismo de selección elitista y ajuste dinámico del parámetro *p* de Bernoulli entre otros.

El parámetro HMCR con valores altos, acelerar la velocidad de convergencia, con valores bajos permite escapar de óptimos locales, lo cual puede ser de gran utilidad hacia el fin de la ejecución. Detalle en ecuación (6).

(6)

donde:

*HMCR*(*t*) = Harmony Memory Consideration Rate para la iteración *t*.

*HMCRmin* = Mínimo valor que puede tomar *HMCR*.

*HMCR*max = Máximo valor que puede tomar HMCR.

*NI* = Número de improvisaciones o iteraciones.

*t* = Iteración o improvisación actual.

Con el fin de mejorar la exploración en BGBHS (Salas et al., 2016), se propone una variación dinámica del parámetro de probabilidad *p* de Bernoulli al generar la población inicial. El parámetro *p* parte con el valor 0*,*5 en la iteración o improvisación *t* = 1 y decrece con cada iteración, tendiendo a cero. El comportamiento del parámetro se ajusta a la ecuación (7):

(7)

Este procedimiento permite que las soluciones al inicio de la generación tengan una mayor cantidad de variables activadas, debido a la alta probabilidad de éxito en el ensayo de Bernoulli. A medida que las iteraciones transcurran, el valor de *p* disminuye, lo cual produce que las soluciones tengan menos variables activadas. HM, presentará un amplio espectro de soluciones desde el inicio, lo cual contribuye a una mejor exploración del espacio de búsqueda.

Todas las armonías deben ser reparadas, dado que el proceso de Bernoulli no asegura la factibilidad de las soluciones generadas. El operador de reparación consiste de dos fases: la primera llamada *ADD*, donde se procede a activar columnas del vector de manera tal que este se vuelva factible; la segunda se denomina *DROP* y se aplica para eliminar columnas redundantes sin que ello afecte la factibilidad de la armonía. El detalle de este operador puede ser revisado en el Algoritmo 3.

Se agrega un comportamiento elitista al funcionamiento de la MH, tratando de mejorar en cada iteración la mejor solución o bien mejorando la peor solución existente en HM. Un detalle de este comportamiento puede ser revisado en el algoritmo 4.

|  |  |  |  |
| --- | --- | --- | --- |
| **Pseudo Código: Operador de Reparación**   |  | | --- | | //FASE ADD  M ←  Ai aijxj,i ∈ M  **for** j←1 **to** n **do**  **if** *xj* = 0 and ∃*i* ∈ *M,Ai <* 1 **then**      **endif**  **endfor**  //FASE DROP  **for** j←1 **to** n **do**  **if** *xj* = 1 and ∃*i* ∈ *M,Ai*  - ≥ 1 **then**      **endif**  **endfor** |   **Algoritmo 3.** Operador de reparación ADD y DROP | **Pseudo Código: Operador Elitista**   |  | | --- | | **if** () < ()  =  **if** () < ()  = |   **Algoritmo 4.** Mecanismo de selección elitista. |

La operación que permite agregar soluciones al HM logrando la implementación automática de PV, se basa en el proceso de generación inicial de armonías, tomando en cuenta un porcentaje de no mejora. Si transcurridas 100 iteraciones, se detecta estancamiento en óptimo local, se desencadena un proceso que se explica en algoritmo 5. Esta operación permite al algoritmo escapar de óptimos locales, alcanzando una mejor exploración del espacio de búsqueda.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Pseudo Código: Operador Incremento de Población**   |  | | --- | | nuevas\_armonias[] = inicialización de vector largo 10  **for** j←0 **to** 8 **do**  nueva\_armonia = nueva armonía con Proceso de Bernoulli usando  nueva\_armonia = reparacion(nueva\_armonia)  nuevas\_armonias[j]=nueva\_armonia  **endfor**  nuevas\_armonias[9] = nueva armonía elitista  mejor\_armonia = obtención de mejor armonía en base a fitness (nuevas\_armonias)  **if** fitness(mejor\_armonia) < fitness(x\_best)  = mejor\_armonia  **if** fitness(mejor\_armonia) < fitness(x\_worst)  = mejor\_armonia |   **Algoritmo 5.** Inclusión de nuevas armonías en HM |  |

Extendiendo la inspiración original de la MH, podríamos decir que el proceso mencionado anteriormente correspondería a realizar una improvisación paralela y la mejor armonía obtenida de ese proceso sería incluida en el repertorio de la improvisación principal. La población paralela creada es eliminada.

# RESULTADOS

Con el fin de probar la correcta ejecución de la MH, se realizó 30 ejecuciones independientes de cada instancia del benchmark, tanto para HS PF como HS PV. El tiempo de ejecución de la MH contra cada uno del set de datos varió entre 2 y 120 minutos dependiendo del nivel de complejidad. La MH fue programada en lenguaje Python 2.7, haciendo uso de las librerías Numpy y Scipy. Como persistencia se utilizó archivos de texto y como procesamiento se contó con una máquina virtual con CPU doble núcleo 2.0 GHz (64 bit), 8GB de RAM, 80 GB de Disco Duro y sistema operativo Windows 7 Profesional de 64 bit. En las tablas expuestas en esta sección, la columna RPD reporta la desviación porcentual relativa entre el menor valor obtenido experimentalmente y el óptimo global para esa instancia (*s*∗). El RPD se calcula mediante la ecuación (8).

(8)

El representa la diferencia entre el RPD de la PV y el RPD de la PF.

Tal como apreciamos, en la Tabla 1, los resultados obtenidos por PV para la instancia 4, fueron más cercanos al óptimo global, es decir: presentaron un mejor fitness. De todos los conjuntos de datos, con ambas técnicas no se llegó a obtener un óptimo global, sin embargo, los RPD se mantienen bajos.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  | **POBLACION**  **FIJA** | | | **POBLACION VARIABLE** | | |  | | Instancia |  |  |  |  |  |  |  | | 4.1 | 429 | 468 | 505 | 9,09 | 456 | 488 | 6,29 | | 4.2 | 512 | 611 | 675 | 19,34 | 583 | 675 | 13,87 | | 4.3 | 516 | 587 | 643 | 13,76 | 565 | 637 | 9,50 | | 4.4 | 494 | 569 | 623 | 15,18 | 548 | 615 | 10,93 | | 4.5 | 512 | 581 | 646 | 13,48 | 552 | 643 | 7,81 | | 4.6 | 560 | 648 | 712 | 15,71 | 624 | 685 | 11,43 | | 4.7 | 430 | 495 | 552 | 15,12 | 475 | 541 | 10,47 | | 4.8 | 492 | 560 | 607 | 13,82 | 526 | 588 | 6,91 | | 4.9 | 641 | 775 | 877 | 20,90 | 748 | 810 | 16,69 | | 4.10 | 514 | 596 | 648 | 15,95 | 580 | 624 | 12,84 |   **Tabla 1.** Resultados instancias 4. | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  | **POBLACION**  **FIJA** | | | **POBLACION VARIABLE** | | |  | | Instancia |  |  |  |  |  |  |  | | 5.1 | 253 | 291 | 317 | 15,02 | 289 | 314 | 14,23 | | 5.2 | 302 | 355 | 388 | 17,55 | 341 | 389 | 12,91 | | 5.3 | 226 | 257 | 272 | 13,72 | 253 | 272 | 11,95 | | 5.4 | 242 | 266 | 291 | 9,92 | 264 | 283 | 9,09 | | 5.5 | 211 | 246 | 265 | 16,59 | 240 | 258 | 13,74 | | 5.6 | 213 | 264 | 286 | 23,94 | 253 | 295 | 18,78 | | 5.7 | 293 | 342 | 376 | 16,72 | 334 | 365 | 13,99 | | 5.8\* | 288 | 331 | 364 | 14,93 | 335 | 371 | 16,32 | | 5.9 | 279 | 330 | 359 | 18,28 | 325 | 360 | 16,49 | | 5.10 | 265 | 294 | 314 | 10,94 | 290 | 317 | 9,43 |   **Tabla 2.** Resultados instancia 5. |

En la Tabla 2, se puede ver como la estrategia PV obtiene mejores los mejores resultados, excepto para el set 5.8\*. Las tablas 3 y 4 muestran como consistentemente la técnica con PV presentó mejor desempeño que la técnica con PF.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  | **POBLACION**  **FIJA** | | | **POBLACION VARIABLE** | | |  | | Instancia |  |  |  |  |  |  |  | | 6.1 | 138 | 163 | 179 | 18,12 | 155 | 168 | 12,32 | | 6.2 | 146 | 170 | 190 | 16,44 | 164 | 184 | 12,33 | | 6.3 | 145 | 169 | 189 | 16,55 | 165 | 179 | 13,79 | | 6.4 | 131 | 148 | 160 | 12,98 | 142 | 155 | 8,40 | | 6.5 | 161 | 193 | 212 | 19,88 | 186 | 205 | 15,53 |   **Tabla 3.** Resultados instancia 6. | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  | **POBLACION**  **FIJA** | | | **POBLACION VARIABLE** | | |  | | Instancia |  |  |  |  |  |  |  | | A.1 | 253 | 283 | 307 | 11,86 | 280 | 304 | 10,67 | | A.2 | 252 | 294 | 324 | 16,67 | 291 | 324 | 15,48 | | A.3 | 232 | 274 | 295 | 18,10 | 268 | 288 | 15,52 | | A.4 | 234 | 269 | 294 | 14,96 | 267 | 287 | 14,10 | | A.5 | 236 | 271 | 292 | 14,83 | 261 | 293 | 10,59 |   **Tabla 4.** Resultados instancia A. |

La Tabla 5, muestra que la técnica PF solo fue mejor en uno de los conjuntos de datos de la instancia B.5\*.

En las instancias superiores, C y D se presentan comportamientos donde ambas técnicas llegan a igualdad de soluciones, como es el caso de C.3\*\*, C.5\*\*, D.1\*\* y D.2\*\*. Solamente en la instancia D.5\* la técnica con PF es mejor. Para el resto, PV obtiene resultados más cercanos al óptimo global.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  | **POBLACION**  **FIJA** | | | **POBLACION VARIABLE** | | |  | | Instancia |  |  |  |  |  |  |  | | B.1 | 69 | 85 | 93 | 23,19 | 82 | 92 | 18,84 | | B.2 | 76 | 92 | 105 | 21,05 | 90 | 97 | 18,42 | | B.3 | 80 | 89 | 95 | 11,25 | 87 | 94 | 8,75 | | B.4 | 79 | 92 | 104 | 16,46 | 91 | 102 | 15,19 | | B.5\* | 72 | 82 | 94 | 13,89 | 86 | 91 | 19,44 |   **Tabla 5.** Resultados instancia B. | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  | **POBLACION**  **FIJA** | | | **POBLACION**  **VARIABLE** | | |  | | Instancia |  |  |  |  |  |  |  | | C.1 | 227 | 261 | 275 | 14,98 | 256 | 279 | 12,78 | | C.2 | 219 | 260 | 280 | 18,72 | 256 | 275 | 16,89 | | C.3\*\* | 243 | 298 | 342 | 22,63 | 298 | 318 | 22,63 | | C.4 | 219 | 264 | 287 | 20,55 | 261 | 290 | 19,18 | | C.5\*\* | 215 | 244 | 290 | 13,49 | 244 | 273 | 13,49 |   **Tabla 6.** Resultados instancia C. |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **POBLACION**  **FIJA** | | | **POBLACION**  **VARIABLE** | | |  |
| Instancia |  |  |  |  |  |  |  |
| D.1\*\* | 60 | 66 | 76 | 10,00 | 66 | 87 | 10,00 |
| D.2\*\* | 66 | 76 | 86 | 15,15 | 76 | 88 | 15,15 |
| D.3 | 72 | 82 | 93 | 13,89 | 81 | 104 | 12,50 |
| D.4 | 62 | 70 | 83 | 12,90 | 69 | 96 | 11,29 |
| D.5\* | 61 | 69 | 80 | 13,11 | 70 | 76 | 14,75 |

**Tabla 7.** Resultados instancia D.

En definitiva, PV obtuvo mejores o iguales resultados en 42 de los 45 set de datos, logrando una efectividad del 93.3%. Restando los empates, se presenta una ventaja de 84.4% para PV.

Para realizar un análisis estadístico de PF vs PF se deben eliminar los outliers (observación estadística marcadamente diferente a otros valores de la muestra) de los resultados obtenidos. Una técnica de detección de outliers, puede ser mediante gráfica de cajas (figura 2), donde se determina la existencia de ellos para los resultados del SCP41 con PF.



**Figura 2.** Boxplot instancia SCP41 para la detección de outliers.

Luego de que se retiraron los outliers, se verificó la normalidad de los datos mediante los test Shapiro-Wilk (Shapiro & Wilk., 1965) y Kolmogorov-Smirnov-Lilliefors (Lilliefors, 1967), para lo cual se plantearon las siguientes hipótesis:

Dado los p-valores obtenidos en los test, se rechaza . Esto permitió aplicar Wilcoxon-Mann-Whitney (Mann & Whitney, 1947). Para verificar superioridad de la estrategia de resolución con PV sobre PF, se define las hipótesis:

El programa estadístico R, obtuvo un p-valor < 0.05 por lo cual se rechaza y se acepta esto implica que PV proporciona mejores resultados. Este procedimiento se extiende a cada instancia del Benchmark. Los resultados de este análisis son coincidentes con lo verificado mediante RPD.

Al comparar los resultados de instancias superiores de HS con PV contra BH (Gómez et al., 2016) nos damos cuenta que tanto en la instancia B.1 como D.3 ambos algoritmos llegan a los mismos resultados.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Instancia |  | HS PV | BH | MEJOR |
| A.1 | 253 | 280 | 298 | HS PV |
| A.2 | 252 | 291 | 301 | HS PV |
| A.3 | 232 | 268 | 256 | BH |
| A.4 | 234 | 267 | 268 | HS PV |
| A.5 | 236 | 261 | 266 | HS PV |
| B.1 | 69 | 82 | 82 | = |
| B.2 | 76 | 90 | 99 | HS PV |
| B.3 | 80 | 87 | 89 | HS PV |
| B.4 | 79 | 91 | 88 | BH |
| B.5 | 72 | 86 | 88 | HS PV |
| C.1 | 227 | 256 | 252 | BH |
| C.2 | 219 | 256 | 245 | BH |
| C.3 | 243 | 298 | 266 | BH |
| C.4 | 219 | 261 | 252 | BH |
| C.5 | 215 | 244 | 247 | HS PV |
| D.1 | 60 | 66 | 71 | HS PV |
| D.2 | 66 | 76 | 73 | BH |
| D.3 | 72 | 81 | 81 | = |
| D.4 | 62 | 69 | 70 | HS PV |
| D.5 | 61 | 70 | 72 | HS PV |

De las restantes 18 instancias, HS con PV obtiene mejores resultados en un total de 11 instancias. Esto representa un total de 61.1% de eficacia en comparación con BH. Por grupo de instancias, el mejor desempeño para HS PV es la instancia A, el peor desempeño se presenta en la instancia C.

# DISCUSIÓN Y CONCLUSIÓN

Los resultados comparados directamente a través de muestran que: en la mayoría de las

instancias probadas, la versión de HS con PV es mejor. Los resultados obtenidos por instancias son representados en la figura 3.

**Figura 3.** Resultados por instancia para técnicas PF y PV

Consistentemente, el análisis estadístico realizado demuestra que el comportamiento de la MH con PV obtiene mejores o iguales resultados en 93.3% de las instancias. Llama la atención el hecho de que la variación de población encuentre mejores resultados incluso a menor cantidad de iteraciones lo cual es bastante prometedor debido a que disminuye el tiempo de procesamiento computacional, con lo cual muchos problemas pueden ser resueltos en una menor cantidad de tiempo.

En el presente trabajo se realizó una variación positiva en el tamaño de la población (Agregando soluciones al espacio en memoria hasta un máximo predefinido), pero no se profundizó en realizar un ajuste dinámico, en el sentido de agregar o quitar soluciones del espacio de memoria. Esto abre claramente la opción para investigar la posibilidad de utilizar una población completamente adaptativa o paralelas.

Si miramos la Figura 3, correspondiente al gráfico de convergencia muestra estancamiento en óptimos locales, claramente hacia el final la tendencia es asintótica a . Por el contrario, la Figura 4 presenta un comportamiento siempre decreciente, sin estancamientos en óptimos locales, mostrando una gran capacidad de exploración, con tendencia al óptimo global.



|  |  |
| --- | --- |
| **Figura 3.** Gráfica de convergencia SCP48, obteniendo como mejor solución 560. | **Figura 4.** Gráfica de convergencia SCP48, obteniendo como mejor solución 526. |

Finalmente queda demostrado que para resolver problemas de optimización como SCP, es útil emplear PV en contraposición a PF.

# REFERENCIAS

Beasley, J. (1990). OR-library: distributing test problems by electronic mail. *Journal of the*

*Operational Research Society*, *41*(11), 1069–1072.

Beheshti, Z., & Shamsuddin, S. (2013). A review of population-based meta-heuristic algorithms.

*Int. J. Adv. Soft Comput. Appl*, *5*(1), 1-35.

Birattari, M., Stützle, T., Paquete, L., & Varrentrapp, K. (2002, July). A racing algorithm for

configuring metaheuristics. In *Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation* (pp. 11-18). Morgan Kaufmann Publishers Inc.

Bertsekas, D., & Tsitsiklis, J. (2002). *Introduction to probability* (Vol. 1). Belmont, MA: Athena

Scientific.

Cacchiani, V., Hemmelmayr, V. C., & Tricoire, F. (2014). A set-covering based heuristic algorithm for

the periodic vehicle routing problem. *Discrete Applied Mathematics*, *163*, 53-64.

Ceria, S., Nobili, P., & Sassano, A. (1998). A Lagrangian-based heuristic for large-scale set covering

problems. *Mathematical Programming*, *81*(2), 215-228.

Chen, L., & Crampton, J. (2009, September). Set Covering Problems in Role-Based Access Control. In

*ESORICS* (pp. 689-704).

Crawford, B., Soto, R., Monfroy, E., Astorga, G., García, J., & Cortes, E. (2017, September). A Meta-

Optimization Approach for Covering Problems in Facility Location. In *Workshop on Engineering Applications* (pp. 565-578). Springer, Cham.

Crawford, B., Soto, R., Cuesta, R., & Paredes, F. (2014). Application of the artificial bee colony

algorithm for solving the set covering problem. The Scientific World Journal, 2014.

Geem, Z. W. (2009). Music-inspired harmony search Algoritmo: theory and applications (Vol. 191).

Springer.

Geem, Z. W., Kim, J. H., & Loganathan, G. V. (2001). A new heuristic optimization algorithm: harmony

search. *simulation*, *76*(2), 60-68.

Gómez, A., Crawford, B., Soto, R., Jaramillo, A., Mansilla, S., Salas, J., & Olguín, E. (2016, June). An

binary black hole algorithm to solve the set covering problem. In Information Systems and Technologies (CISTI), 2016 11th Iberian Conference on (pp. 1-5). IEEE.

Gómez, A., Crawford, B., Soto, R., Olguín, E., Misra, S., Jaramillo, A. & Salas, J. (2016, July). Solving

the Set Covering Problem with a Binary Black Hole Inspired Algorithm. In *International Conference on Computational Science and Its Applications* (pp. 207-219). Springer International Publishing.

Housos, E., & Elmroth, T. (1997). Automatic optimization of subproblems in scheduling airline crews.

*Interfaces*, *27*(5), 68-77.

Jaramillo, A., Crawford, B., Soto, R., Villablanca, S. M., Rubio, Á. G., Salas, J., & Olguín, E. (2016,

August). Solving the set covering problem with the Soccer League Competition algorithm. In International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (pp. 884-891). Springer International Publishing.

Karp, R. (2010). Reducibility among combinatorial problems. In *50 Years of Integer Programming*

*1958-2008* (pp. 219-241). Springer Berlin Heidelberg.

Lilliefors, H. (1967). On the Kolmogorov-Smirnov test for normality with mean and variance

unknown. *Journal of the American statistical Association*, *62*(318), 399-402.

Mann, H., & Whitney, D. (1947). On a test of whether one of two random variables is stochastically

larger than the other. *The annals of mathematical statistics*, 50-60.

Omran, M., & Mahdavi, M. (2008). Global-best harmony search. *Applied Mathematics and*

*Computation*, *198*(2), 643–656.

Prata, B. (2016). A multiobjective metaheuristic approach for the integrated vehicle and crew

scheduling. *Journal of Transport Literature*, *10*(2), 10-14.

Ribeiro, C., Minoux, M., & Penna, M. (1989). An optimal column-generation-with-ranking

Algoritmo for very large scale set partitioning problems in traffic assignment. *European Journal of Operational Research*, *41*(2), 232–239.

Salas, J., Crawford, B., Soto, R., Rubio, Á. G., Jaramillo, A., Villablanca, S. M., & Olguín, E. (2016,

August). Binary Harmony Search Algorithm for Solving Set-Covering Problem. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems* (pp. 917-930). Springer International Publishing.

Samà, M., DAriano, A., Toli, A., Pacciarelli, D., & Corman, F. (2015, September). Metaheuristics for

real-time near-optimal train scheduling and routing. In Intelligent Transportation Systems (ITSC),

2015 IEEE 18th International Conference on (pp. 1678-1683). IEEE.

Sarubbi, J., Silva, T., Martins, F., Wanner, E., & Silva, C. (2017, May). A GRASP based

heuristic for Deployment Roadside Units in VANETs. In *Integrated Network and Service Management (IM), 2017 IFIP/IEEE Symposium on* (pp. 369-376). IEEE.

Shapiro, S., & Wilk, M. (1965). An analysis of variance test for normality (complete samples).

Biometrika, 52(3/4), 591-611.

Torres-Jiménez, J., & Pavón, J. (2014). Applications of metaheuristics in real-life problems.

Turky, A., & Abdullah, S. (2014). A multi-population harmony search algorithm with external

archive for dynamic optimization problems. Information Sciences, 272, 84-95.

Xiang, W., An, M., Li, Y., He, R., & Zhang, J. (2014). An improved global-best harmony search

Algoritmo for faster optimization. *Expert Syst. Appl.*, *41*(13), 5788–5803.

Yakıcı, E. (2017). A heuristic approach for solving a rich min-max vehicle routing problem with mixed

fleet and mixed demand. *Computers & Industrial Engineering*, *109*, 288-294.

Zou, D., Gao, L., Li, S., & Wu, J. (2011). Solving 0-1 knapsack problem by a novel global harmony

search Algoritmo. *Appl. Soft Comput.*, *11*(2), 1556–1564.