ISSN 1870-4069

# Algoritmo de evolución diferencial con reparador cromosómico aplicado a un problema de secuenciación de vehículos

Elvi Malintzin Sánchez Márquez, Héctor José Puga Soberanes, Luis Ernesto Mancilla Espinoza, Juan Martín Carpio Valadez, Manuel Ornelas Rodríguez, Javier Iván Manzanares Cuadro

Tecnológico Nacional de México / Instituto Tecnológico de León, Departamento de Estudios de Posgrado e Investigación, León, Guanajuato, México
martinezbucita@gmail.com,pugahector@yahoo.com, lmancilla01@hotmail.com,jmcarpio61@hotmail.com, mornelas67@yahoo.com.mx,d2241427@itleon.edu.mx

**Resumen.** En el artículo se propone un algoritmo de Evolución Diferencial con Reparador Cromosómico (EDRC) aplicado a la secuenciación de vehículos, que consiste en encontrar una secuencia de producción de diferentes modelos de automóviles en una línea de ensamblaje. Este es un problema de satisfacción de restricciones multiobjetivo NP-Duro [1], en el que se busca violar la menor cantidad de restricciones. Para generar la población inicial se utiliza un operador de mutación basado en el cambio. Además, se propone un reparador cromosómico que toma en cuenta las características del problema y asegura la generación de individuos factibles. Las soluciones del EDRC fueron comparadas con los resultados reportados por el algoritmo de recocido simulado usado por la Renault [2] y el equipo que aplicó búsqueda tabú y búsqueda codiciosa greedy [3], mostrando competencia (39% de los casos), mejora (22% de los casos), no logrando competir en el 39% de los casos. En la etapa que dará continuidad al proyecto se analizarán operadores de mutación especializados para mejorar el desempeño de la propuesta.

**Palabras clave:** problema de secuenciación de vehículos, evolución diferencial, reparador cromosómico

# Differential Evolution Algorithm with Chromosome Repair Applied to a Problem of Vehicle Sequencing

**Abstract.** The article proposes a differential evolution algorithm with chromosome repair (EDRC), applied to sequencing of vehicles, this problem aims to find a production sequence of different car models in an assembly line. This is a NP-Hard multiobjective restrictions satisfaction problem [1], in which it is expected to violate the least amount of restrictions. A change-based mutation operator is used to generate the initial population. It also proposes a chromosomal repairman that takes into account the characteristics of the problem and ensures the generation of feasible individuals. EDRC solutions were compared with the

results reported by the simulated annealing algorithm used by Renault [2] and the team that applied tabu search and greedy search [3], viewing competition (22% of cases), improvement (21% of cases), failing to compete in 37% of cases. The next phase of project will analyze specialized mutation operators to improve the performance of the proposal.

**Keywords:** vehicle sequencing problem, differential evolution, chromosomal repairman.

## 1. Introducción

El problema de secuenciación de vehículos fue descrito por Parello y Kabat en 1986 como la programación de vehículos a lo largo de una línea de ensamble, con el fin de instalar distintas opciones por cada vehículo, lo que se conoce como línea de ensamblaje mixto [4]. Este problema se presentó como un problema de satisfacción de restricciones (CSP) por primera vez utilizando una programación de restricción que tiene que ver con el aspecto declarativo de la lógica y la eficiencia con la técnica de manipulación de restricciones denominada CHIP [4].

En el estado del arte hay una amplia cantidad de investigaciones que abordan el problema, algunas referencias de estas investigaciones son [5,6].

En 2005, el problema de secuenciación fue utilizado como punto de referencia para el Desafío Roadef, donde se establecieron restricciones duras y blandas, además de introducir restricciones a los lotes de pintura para minimizar el consumo de solventes en dicho taller. Para resolver el problema, se aplicaron métodos exactos como programación entera y método ad-hoc , además de métodos heurísticos greedy [3], búsqueda local [2] , algoritmos genéticos [7] y búsqueda tabú iterativa [6] por mencionar algunos. El desafío ofreció una visión integral respecto a los métodos propuestos, y el algoritmo ganador fue implementado en Renault y fábricas piloto europeas.

Como resultado de este concurso se publicaron un total de 6 artículos, entre los cuales se encuentra "Búsqueda tabú para el problema de secuenciación de vehículos", el cual hemos tomado como base para nuestra investigación [3].

El problema abordado en este artículo es un problema multiobjetivo con restricciones de suavizado. La función objetivo contiene tres componentes $F_1$, $F_2$ , $F_3$ y es evaluada mediante un enfoque lexicográfico escalonado, respetando $F_1 > F_2 > F_3$ ; el orden de prioridad de los objetivos viene dado por las instancias utilizadas en el concurso.

En este trabajo se tomaron las instancias del conjunto A que formaban parte de un grupo integrado por el conjunto X y B, también se utilizaron como referente los resultados del concurso Roadef, que son vigentes en la comparación de trabajos recientes como son [3,8].

Los objetivos son minimizar las sobrecargas o violaciones de restricciones de proporción del taller de línea de ensamblaje (suavizado) y minimizar los cambios de color del taller de pintura.

Las restricciones de proporción $N_i/P_i$ están relacionadas con la *i-ésima* estación de trabajo y representan las capacidades de cada estación. La restricción indica que hay

un máximo de $N_i$ vehículos que requieren la opción $i$, que pueden programarse en una secuencia de vehículos $P_i$ en la estación de trabajo, de lo contrario ocurre una sobrecarga. En este modelo, una solución consiste en una secuencia de los últimos automóviles del día de producción D-1 (anterior, que ya fueron programados) y todos los automóviles del día D (actual).

## 2. Descripción del problema

El problema consiste en que, dado un conjunto de vehículos se desea encontrar la mejor secuencia *S* de éstos, que satisfaga las restricciones establecidas en cada estación de trabajo de una línea de ensamblaje. En este sentido, la secuencia $S$ se puede identificar como un arreglo lineal de $c'$ vehículos de la producción del día anterior (D-1), los cuales ya fueron programados y no se deben modificar, más $c$ carros del día de producción actual (D) véase ecuación (1) y (2). Se establecen dos tipos de restricciones de proporción: las restricciones H consideradas como de alta prioridad ya que debido a las características del automóvil se requiere una gran carga de trabajo en la línea de ensamblaje y las restricciones L como de baja prioridad ya que las características del automóvil causan pequeños inconvenientes en la línea de ensamblaje. Ambas consideradas como restricciones suaves, ya que no se puede garantizar la satisfacción completa al programar un día de producción véase ecuación (4) y (5).

Se toma en cuenta una restricción dura, que es el tamaño de lote, la cual nos dice el número máximo de carros consecutivos del mismo color tolerable véase ecuación (3).

En este contexto se definen:

$$S_p = l, \tag{1}$$

representa que el vehículo *l* está en la posición *p* de la solución *S*.

$$(l) \in \{D-1, D\}, \tag{2}$$

representa día de producción del automóvil *l*.

$$Color\ (l) \in \{1,\ldots,50\}, \tag{3}$$

$$H_i \in \{0,1\} \tag{4}$$

$$L_i \in \{0,1\}, \tag{5}$$

donde $H_i = 1$ si el automóvil requiere la opción y $H_i = 0$ cuando no, mismo caso para $L_i$.

Los objetivos que se buscan cumplir son: (1) suavizado de restricciones y (2) disminuir el número de cambios de color en el taller de pintura.

La función fitness se establece de acuerdo a los objetivos, siendo evaluada con un enfoque lexicográfico escalonado que se define en la ecuación (6):

$$F(S) = F_1(S) + F_2(S) + F_3(S), \tag{6}$$

donde $F_1$, $F_2$ , $F_3 \in \{FH, FL, FC\}$ de acuerdo a la prioridad de optimización de objetivos de cada instancia. $FH$ es el número de sobrecargas de restricciones de alta prioridad en la solución $S$, $FL$ es el número de sobrecargas de restricciones de baja

prioridad en la solución *S* (ambos corresponden al objetivo de suavizado) y $FC$ que son los cambios de color en la solución.

Las sobrecargas se calculan mediante las siguientes expresiones ecuación (7) y (8):

$$FH(S) = \sum_{i=1}^{|H|} FH_i(S), \tag{7}$$

$$FL(S) = \sum_{i=1}^{|L|} FL_i(S). \tag{8}$$

En el cálculo de sobrecargas se hace uso de ventanas deslizantes $W_k$ de tamaño *k*. El tamaño de cada ventana, en la *i-ésima* estación de trabajo, es definido mediante las restricciones de proporción $N_i/P_i$ que son únicas para cada estación de trabajo, por ello se realiza un cálculo para la primer ventana y las ventanas finales. Las ventanas intermedias mantienen constante su tamaño $k = P_i$ durante todos sus deslizamientos hasta comenzar las ventanas finales.

La primera ventana de tamaño $P_i$ contemplará vehículos del día $D - 1$ en conjunto con vehículos del día D, para conocer la cantidad de vehículos de cada día se requieren las siguientes expresiones ecuación (9) y (10):

$$\#c' = P_i - 1, \tag{9}$$

$$\#c = P_i - \#c', \tag{10}$$

donde $\#c'$ es la cantidad de vehículos del día $D - 1$ que se deben tomar de la secuencia, y $\#c$ es la cantidad de vehículos del día D. Esto únicamente para la primera ventana.

En el caso de las ventanas finales:

$$NewP_i = P_i - 1, \tag{11}$$

$$K = N_i + 1, \tag{12}$$

donde $NewP_i$ nos indica el tamaño de la primera ventana, en el conjunto de ventanas finales. Este valor va en decremento hasta llegar a $K$ que es el tamaño de la última del conjunto de ventanas finales ecuaciones (11-12).

Este suceso de las últimas ventanas se debe a que la secuencia es infinita y se consideran los carros del día $D - 1$, $D$ y se toma como supuesto que existirán vehículos del día $D + 1$.

Teniendo los tamaños de las ventanas se realiza el proceso de evaluación de la función fitness para cada estación de trabajo mediante la expresión ecuaciones (13):

$$FE_i(S) = \sum_{a}^{b} FE_i(S, W_{Pi}) + \sum_{k}^{NewP_i} FE_i\left(S, W_{NewP_i}\right), \tag{13}$$

donde $FE_i(S)$, representa a la función fitness $FH_i(S)$ para las prioridades *H*, o $FL_i(S)$ para las prioridades *L*. La variable $a$ es la posición donde inicia la primera ventana y $b$ es la posición donde inicia la primera ventana del conjunto de últimas ventanas. Para

obtener los valores de las variables se utilizan las siguientes expresiones ecuaciones (14-15):

$$a = |c'| - \#c', \tag{14}$$

$$b=c-NewPi. \tag{15}$$

El número de sobrecargas en la ventana $W_k$ al momento del deslizamiento de la misma en una secuencia de $k$ carros consecutivos se define como $FE_i(S, W_k)$ y se calcula mediante la expresión ecuación (16-17):

$$FE_i(S, W_k) = \max\left\{\left(\sum_{i=1}^{|E|} E_i(l)\ en\ la\ ventana\ W_k\right) - N_i\right\}, \tag{16}$$

$$\text{Si } FE_i(S, W_k) < 0 \text{ será igual a 0.} \tag{17}$$

## 3. Algoritmo de evolución diferencial con reparador cromosómico (EDRC)

En esta sección se describe el algoritmo de evolución diferencial con un reparador cromosómico, así como la agrupación de los vehículos en clases de equivalencia usando la distancia de Hamming. Se detalla la estructura del individuo, así como los procesos de cruza y mutación; utilizando para este último el método $(DE/rand/1/bin)$, también se puntualiza el proceso para la generación de la población inicial, implementando un operador de mutación denominado "basado en el cambio" propuesto por Syswerda [9] y se muestra el diagrama general del EDRC.

### 3.1 Agrupación de vehículos en clases de equivalencia

Como primer paso los automóviles son agrupados en clases de equivalencia definidas por aquellos automóviles que cuentan con los mismos requerimientos u opciones de alta y baja prioridad, $H$ y $L$ respectivamente [7]. Este agrupamiento se realiza mediante el cálculo de la distancia de Hamming entre dos vehículos [3]. El resultado nos indica la similitud o diferencia en cuanto a su configuración. La expresión para este cálculo se muestra en la ecuación (18):

$$\begin{aligned} dist(x,y) = distH(x,y)\left\lfloor\sum_{i=1}^{|H|}\lfloor H_i(x) + H_i(y)\rfloor mod\ 2\right\rfloor \\ +\ distL(x,y)\left\lfloor\sum_{i=1}^{|L|}\lfloor L_i(x) + L_i(y)\rfloor mod\ 2\right\rfloor. \end{aligned} \tag{18}$$

### 3.2 Individuo

La Tabla 1 muestra la configuración de un individuo (secuenciación de vehículos). El individuo está conformado por dos cromosomas: cromosoma (a) donde cada gen representa una clase y el cromosoma (b) donde cada gen representa un color que le corresponde a la clase del vector (a) [7].

**Tabla 1**. Ejemplo de configuración de un individuo.

| **INDIVIDUO** | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SECUENCIA | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| CROMOSOMA (A) | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 1 | 1 | 1 | 1 | 3 | 1 | 1 |
| CROMOSOMA (B) | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 5 | 4 | 4 | 4 | 4 | 6 | 5 | 6 |

### 3.3 Generación de población inicial

Los individuos de la población inicial son generados con un operador de mutación denominado "basado en el cambio" a partir del orden de la instancia original, este operador consiste en seleccionar aleatoriamente de la secuencia una subsecuencia de autos, cambiarles el orden y a continuación insertar la subsecuencia en la misma posición que fue extraída [9] .

### 3.4 Operadores de mutación y cruza

Los operadores de mutación *(DE/rand/1/bin)* y cruza utilizados en esta investigación se muestran en la ecuación (19) y ecuación (20) respectivamente [10]:

*Operador de mutación clásico (DE/rand/1/bin).* El operador de mutación, definido en la ecuación (19), lleva implícito el proceso de selección, en el cual se eligen de la población, de manera aleatoria 3 individuos $X_{r1}, X_{r2}\ y\ X_{r3}$ diferentes entre sí, y diferentes de $X_{ri}$ que representa al individuo que se busca reemplazar. Observe que el individuo mutado $X_m$ se obtiene realizando operaciones aritméticas, donde F representa al factor de mutación.

$$X_m = X_{r1} + F(X_{r2} - X_{r3}), \tag{19}$$

$$\text{donde: } r1 \neq r2 \neq r3 \neq ri \in [1, N], \qquad 0 < F \leq 2.$$

*Operador de cruza.* Para el proceso de cruza se utilizan los individuos, $X_m$ y $X_{ri}$, como padres para generar el individuo de prueba $X_t$.

El modelo para generar cada $X_t$ se muestra en la Ecuación (20) donde $Xm_j$ representa el j-ésimo gen de $X_m$ , $Xri_j$ representa el j-ésimo gen de $X_{ri}$.

En este caso, el j-ésimo gen de $X_t$, se obtiene a partir de generar dos números aleatorios, $rand1$ en un rango de $[0,1]$ y $rand2$ en un rango de $[1, total\ de\ genes]$ y tomando en cuenta la probabilidad de cruza CR, de acuerdo a la regla definida en ecuación (20).

$$X_t = \begin{cases} Xm_j, si \;\; rand1 < CR \text{ ó } rand2 == j, \\ Xri_j, \qquad en\ otros\ casos, \end{cases} \quad donde: j = 1,2,3, \ldots |X_{ri}|. \qquad (20)$$

### 3.5 Reparador cromosómico

La naturaleza del algoritmo de evolución diferencial (DE) por sus siglas en inglés, requiere de evaluar al individuo para posteriormente, si presenta alguna mejora, elegirlo.

Debido a esta naturaleza al momento de aplicar DE a un problema combinatorio, como es el caso del problema de secuenciación de vehículos, puede darse el caso de que después del proceso evolutivo se generen individuos no factibles, no permitiendo realizar la evaluación de la función objetivo. Por esta situación, es deseable generar un proceso de reparación a dichos individuos convirtiéndolos en factibles, este proceso es conocido como reparador cromosómico.

Se han generado reparadores para otros algoritmos, los cuales permiten reparar fallos en la representación de las soluciones en investigaciones sobre asignación de horarios en una institución educativa y también como operador de selección [11]. Para el problema que abordamos, se desarrollaron dos reparadores cromosómicos, Rclase asociado al objetivo de suavizado y Rcolor asociado al objetivo de cambios de color.

Un cromosoma es un vector cuyos componentes son genes que contienen identificadores de todas las clases. En un cromosoma un mismo identificador puede aparecer varias veces. Un cromosoma es no factible cuando al menos en un gen aparece un identificador de clase que no existe y/o cuando la aparición de un identificador en el cromosoma excede la cardinalidad de la clase.

En este sentido reparar un cromosoma consiste en cambiar aquellos genes que vuelven el cromosoma no factible por genes que contengan identificadores de clase dentro del conjunto de las clases sin que la aparición de los identificadores exceda la cardinalidad de las clases. El proceso para realizar la reparación cromosómica se describe en el Algoritmo 1.

En el Algoritmo 1 se puntualiza el funcionamiento del reparador Rclase, la parte 1 consiste en la reparación del cromosoma (a) del individuo y la parte 2 consiste en la reparación del cromosoma (b), a partir de la secuencia generada por el paso previo. El algoritmo del reparador cromosómico Rcolor (que no se presenta), tiene la misma estructura, pero su ejecución es realizada intercambiando el orden de las partes 1 y 2.

**Algoritmo 1:** Reparador cromosómico (Rclase)

*Parte 1*

1. Se toma el individuo $\boldsymbol{Xtclases}$ que se genera después del proceso de mutación y cruza.
2. Se recorre de izquierda a derecha respetando el orden.
3. Para $\boldsymbol{i = 1}$ hasta $|\boldsymbol{Xtclases}|$.
   A. Si $\boldsymbol{Xtclases_i < 0\ or\ Xtclases_i > \#C}$ entonces:
      (1) Eliminar $\boldsymbol{Xtclases_i}$.
   B. Contar cuantas veces existe $\boldsymbol{Xtclases_i}$ en $\boldsymbol{Xtclases}$.
   C. Si $\boldsymbol{\#Xtclases_i > \#c\ C_i}$ entonces:
      (1) Eliminar $\boldsymbol{Xtclases_i}$.
4. Verificar que clases faltan en $\boldsymbol{Xt}$clases.

5. En las posiciones eliminadas, agregar de izquierda a derecha:
6. Las clases faltantes en $\boldsymbol{Xtclases}$.
7. Los carros faltantes para cada clase de acuerdo a $\#\boldsymbol{c}\ \boldsymbol{C_i}$.
8. Regresar el individuo $\boldsymbol{Xtclases}$.

*Parte 2*

9. Se toma el individuo $\boldsymbol{Xtcolores}$ que se genera después del proceso de mutación y cruza.
10. Se obtiene el $\boldsymbol{color_{max}}$ y el $\boldsymbol{color_{min}}$ que fungirán como el rango.
11. Se recorre de izquierda a derecha respetando el orden.
12. Para $\boldsymbol{i = 1}$ hasta $|\boldsymbol{Xtcolores}|$.
    A. Si $\boldsymbol{Xtcolores_i} \in \boldsymbol{Xtclases_i}$ entonces:
        (1) Contar cuantas veces existe $\boldsymbol{Xtcolores_i}$ en $\boldsymbol{Xtcolores}$.
        (2) Si $\#\boldsymbol{Xtcolores_i} > \#\boldsymbol{ccolor_i}$ entonces:
            I. Eliminar $\boldsymbol{Xtcolores_i}$.
            II. Sino:
                (A) Si $\boldsymbol{Xtcolores_i} < \boldsymbol{color_{min}}\ \boldsymbol{or}\ \boldsymbol{Xtcolores_i} > \boldsymbol{color_{max}}$ entonces:
                    A. Eliminar $\boldsymbol{Xtcolores_i}$.
13. Verificar que colores faltan de acuerdo a su clase en $\boldsymbol{Xt}$colores.
14. En las posiciones eliminadas, agregar de izquierda a derecha:
15. Los colores faltantes en $\boldsymbol{Xtcolores}$.
16. Los carros faltantes para cada color de acuerdo a $\#\boldsymbol{ccolor_i}$.
17. Regresar el individuo completo reparado, tanto el vector de clases como de colores.

### 3.6 Reemplazo de individuos

El reemplazo utilizado depende de la prioridad de los objetivos, por lo que el proceso de reemplazo se da en tres casos, en cualquiera caso se aplica la proposición que se presenta como Ecuación (21). Si la proposición no se satisface se descarta $X_t$:

$$Si\ F_b(X_t) \leq F_b(X_{ri})\ entonces\ X_{ri} == X_t \qquad b = 1,2,\ldots|F|. \tag{21}$$

El caso 1 la proposición se ejecuta una sola vez, sin embargo, en el caso 2 y 3, la proposición se ejecuta dos y tres veces respectivamente de manera cíclica, siempre y cuando el paso anterior sea verdadero.

### 3.7 Aplicación del EDRC

La ejecución del EDRC depende de la forma de evaluar la función objetivo. Para evaluarla se tomó en cuenta la prioridad de los objetivos de manera escalonada, descartando individuos que empeoren el valor fitness del objetivo previo. La población inicial generada en el EDRC es aplicada para $F_1$ ; la población resultante se toma como inicial para $F_2$ , y así sucesivamente hasta $F_3$ .

El diagrama general del algoritmo EDRC se muestra en la fig. 1, al primer criterio de decisión le corresponden 25 generaciones. Al proceso evolutivo (remarcado con rectángulo azul) le corresponden el resto de los criterios de decisión, los cuales se ejecutan durante 100 generaciones cada uno. El proceso evolutivo, sigue el orden: mutación, cruza, reparación cromosómica y reemplazo por cada objetivo.

Se realizaron 10 experimentos por instancia, cada experimento consta de las 25 generaciones del primer criterio de decisión, estas generaciones proporcionan 10

soluciones, de las cuales se elige la de mejor fitness. Al final se obtuvo una solución por cada experimento. En el artículo se reportan el promedio de los 10 experimentos por instancia.

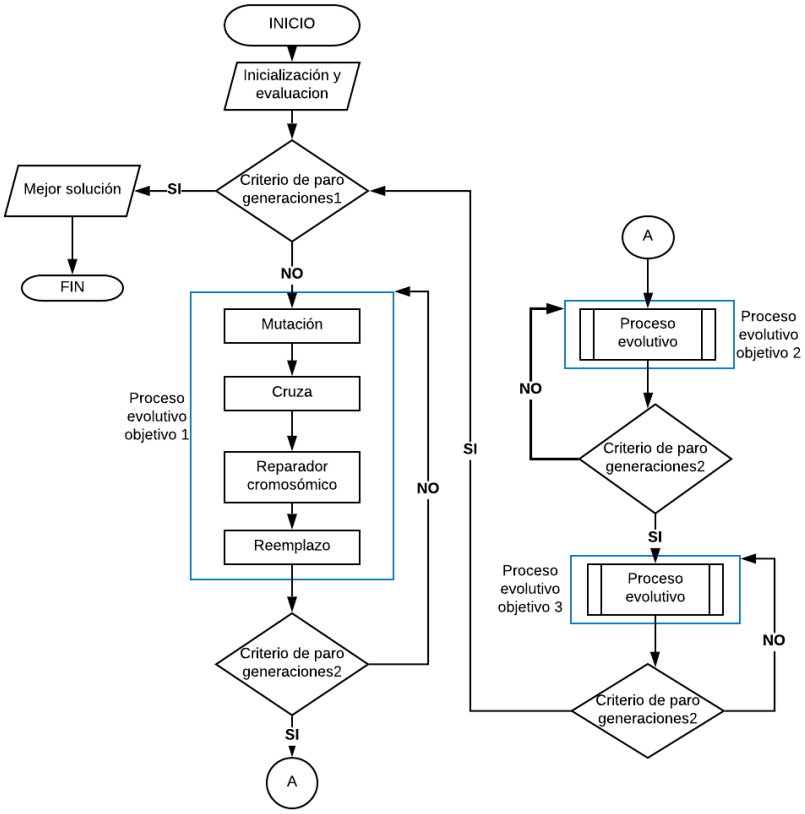


**Fig. 1.** Diagrama general del Algoritmo EDRC.

## 4. Diseño de experimentos

El algoritmo EDRC propuesto fue implementado en java y compilado con Intellij IDEA Community Edition 2018. Los experimentos fueron corridos en una computadora Toshiba con un procesador Intel i3-3217u y 6GB de memoria RAM con Windows 10.

Los parámetros utilizados para el algoritmo EDRC, fueron: factor de mutación F=1, este valor fue asignado para evitar la presencia de clases y colores que no existen dentro del cromosoma, factor de cruza CR=0.9 y tamaño de población N=10, ambos parámetros utilizados en el estado del arte.

El conjunto de instancias utilizadas pertenecen al conjunto denominado A de tres, tomadas de la Constraint Satisfaction Problem Lib repository (ver www.csplib.org),

que son un total de 16 instancias. Cada instancia tiene 4 archivos: *vehicles, ratios, paint_batch_limit y optimization_objectives.*

En el archivo *vehicles*, se encuentran los parámetros *c* y *c'* con toda su configuración respecto a los objetivos H, L y color. El archivo de *ratios* muestra las restricciones de proporción para cada opción H y L. El archivo *paint_batch_limit* muestra la restricción dura en cuanto al tamaño de lote de colores consecutivos y *optimization objetives* muestra la prioridad de los objetivos.

## 5. Resultados

El desempeño del EDRC fue comparado con los resultados reportados por Renault quien utilizo recocido simulado y los resultados reportados en el artículo participantes del concurso Roadef con un algoritmo búsqueda tabú y un greedy para generar la población inicial. Ambos resolviendo el mismo problema.

Los resultados son mostrados en cuatro tablas, una para las instancias de color, otra para las instancias consideradas fáciles para Renault y las dos últimas para las instancias difíciles. Los resultados muestran el desempeño promedio del EDRC propuesto.

**Tabla 2**. Resultados para las instancias de color.

| | **PHEL1** | **PHEL2** | **PHEL3** | **PHE** |
|---|---|---|---|---|
| (C´; C) | (99;335) | (14;485) | (29;875) | (27; 954) |
| (B;\|C\|) | (15; 12) | (450;12) | (15; 14) | (15;14) |
| (\|H\|;\|L\|) | (4; 2) | (3; 6) | (7; 2) | (5;-) |
| | µ | µ | µ | µ |
| ARTICULO FC | 27 | 11 | 64 | 68 |
| RENAULT FC | 30 | 11 | 64 | 69 |
| PROPUESTA FC | 31,2 | 11 | 73,88 | 74,37 |
| ARTICULO FH | 367.8 | 39.4 | 436 | 244.6 |
| RENAULT FH | 197 | 48 | 462 | 392 |
| PROPUESTA FH | 200 | 49,8 | 502,55 | 435,37 |
| ARTICULO FL | 101.2 | 151.4 | 832.4 | NA |
| RENAULT FL | 61 | 5 | 883 | NA |
| PROPUESTA FL | 63,5 | 4,7 | 854,44 | NA |

En las tablas, los nombres asignados a cada instancia corresponden a sus características en cuanto a nivel de dificultad y prioridad de objetivos. Por ejemplo, PHEL1 significa que FC (representado por P) es más importante que FH (representado por H) y este a su vez mas importante que FL (representado por L), la 'E' significa que H fue considerada como fácil de optimizar para la Renault y cuando aparece 'D' es

considerada difícil de optimizar; si existe más de una instancia de este tipo se le asigna un número entero consecutivo. En el segundo renglón de las tablas, es el par ordenado $(c'; c)$, se indica cuantos carros del día $D-1$ $(c')$ y cuantos del día $D$ $(c')$ tiene esa instancia; en el tercer renglón el par $(B; |C|)$, B indica la restricción dura del objetivo de color y C la cantidad de colores, finalmente en el cuarto renglón el par (|H|;|L|), $|H|$ indica la cantidad de estaciones de trabajo para las restricciones de alta prioridad y $|L|$la cantidad de estaciones de trabajo para las restricciones de baja prioridad. En algunas instancias en el objetivo de $FL$ aparece $NA$, eso significa que en esa instancia no existen vehículos que requieran pasar por las estaciones de $L$. Los resultados obtenidos son resaltados con tres distintos colores: azul rey para los resultados donde competimos, verde donde los resultados son mejores que los comparados y rojos donde la propuesta no compite.

**Tabla 3**. Resultados para las instancias fáciles.

| | **HEPL1** | **HEPL2** | **HEPL3** | **HEPL4** | **HELP** |
|---|---|---|---|---|---|
| (C´; C) | (99;335) | (14;485) | (29;875) | (228; 1,004) | (228;1004) |
| (B;\|C\|) | (15; 12) | (450;12) | (15; 14) | (10;24) | (10;24) |
| (\|H\|;\|L\|) | (4; 2) | (3; 6) | (7; 2) | (4;18) | (4;18) |
| | μ | μ | μ | μ | μ |
| ARTICULO FC | 38.8 | 38.8 | 137.4 | 232.8 | 833.2 |
| RENAULT FC | 46 | 70 | 195 | 290 | 290 |
| PROPUESTA FC | 121,77 | 256,22 | 195,88 | 300,44 | 291,66 |
| ARTICULO FH | 0 | 0 | 0 | 0 | 0 |
| RENAULT FH | 28 | 2 | 2 | 0 | 0 |
| PROPUESTA FH | 0 | 3,22 | 2,66 | 26,88 | 0,1 |
| ARTICULO FL | 107.58 | 49.4 | 801.4 | 3705.2 | 377.2 |
| RENAULT FL | 50 | 2 | 787 | 2075 | 2075 |
| PROPUESTA FL | 50,88 | 14,66 | 785 | 2163,44 | 2041 |

**Tabla 4**. Resultados para las instancias difíciles parte 1.

| | **HDP** | **HDPL1** | **HDPL2** | **HDPL3** |
|---|---|---|---|---|
| (C´; C) | 27;954) | (18;600) | (14;1,315) | (14; 1,260) |
| (B;\|C\|) | (20; 14) | (10;12) | (10; 13) | (10;13) |
| (\|H\|;\|L\|) | (5; –) | (5; 12) | (5; 8) | (5;8) |
| | μ | μ | μ | μ |
| ARTICULO FC | 137.2 | 180.8 | 303.2 | 296.2 |
| RENAULT FC | 229 | 182 | 468 | 363 |
| PROPUESTA FC | 397,33 | 187,66 | 537,33 | 536 |
| ARTICULO FH | 13.4 | 0 | 4.2 | 4 |
| RENAULT FH | 115 | 35 | 98 | 73 |
| PROPUESTA FH | 117,33 | 61,55 | 256,44 | 77,66 |
| ARTICULO FL | NA | 1181.4 | 168.8 | 293.4 |
| RENAULT FL | NA | 861 | 99 | 205 |
| PROPUESTA FL | NA | 852,55 | 198,66 | 218,55 |

**Tabla 5.** Resultados para las instancias difíciles parte 2

| | **HDLP1** | **HDLP2** | **HDLP3** |
|---|---|---|---|
| (C´; C) | (18;600) | (14;1,315) | (14;1,260) |
| (B;\|C\|) | (10;12) | (10;13) | (10;13) |
| (\|H\|;\|L\|) | (5;12) | (5;8) | (5;8) |
| | μ | μ | μ |
| ARTICULO FC | 368.2 | 394 | 366,8 |
| RENAULT FC | 334 | 392 | 464 |
| PROPUESTA FC | 199,33 | 464,22 | 563,88 |
| ARTICULO FH | 0.2 | 4 | 4 |
| RENAULT FH | 42 | 106 | 82 |
| PROPUESTA FH | 59,11 | 299,88 | 89,44 |
| ARTICULO FL | 99.4 | 63.6 | 29,6 |
| RENAULT FL | 98 | 134 | 77 |
| PROPUESTA FL | 828,44 | 249,77 | 132,66 |

## 6. Análisis de resultados

Los porcentajes relativos al tipo de instancias fueron los siguientes. Para las instancias de color el 37% (4/11) fue ganado por EDRC, 55% (6/11) compitió y 8% (1/11) obtuvo malos resultados. Para las instancias fáciles el 26% (4/15) fue ganado por EDRC, 53% (8/15) compitió y 21% (3/15) obtuvo malos resultados.

Finalmente, para instancias difíciles el 10% (2/20) fue ganado por EDRC, 20% (4/20) compitió y 70% (14/20) obtuvo malos resultados.

En el total de las 46 comparaciones entre los resultados de la Renault, el artículo seleccionado y EDRC, aproximadamente en el 22% (10/46) fue ganado por EDRC, 39% (18/46) compitió y 39% (18/46) obtuvo malos resultados.

Para determinar la significancia de los resultados obtenidos en los casos donde, de acuerdo a los porcentajes relativos, nuestra propuesta aparece como mejor o compite, se realizó la prueba de normalidad de los resultados con un valor $p$=0.54, la cual mostro que los datos no se comportan como una distribución normal. Con base en lo anterior se realizaron las siguientes cuatro pruebas no paramétricas: prueba de signos, prueba de rangos con signo de Wilcoxon, análisis de varianza de Friedman y coeficiente de concordancia de Kendall usando un valor de significancia de 0.05.

La hipótesis nula que se planteó para estas pruebas fue que el desempeño de los algoritmos era el mismo. Aunque en la Tabla 2, se observa que EDRC compite, los resultados de las pruebas estadísticas mostraron que la diferencia no es significativa respecto al desempeño de los algoritmos. Lo anterior se concluyó debido a que el valor $p$ que se obtuvo para cada objetivo fue: FC $p$=0.250, FL $p$=1 y FH $p$=0.068; en todos los casos mayores a 0.05.

En el caso de la Tabla 3 donde también se observa que EDRC compite, los valores $p$ de las pruebas estadísticas fueron: para FH $p$= 0.500 en las pruebas de Wilcoxon y de signos, y $p$=0.180 para el caso de las pruebas Friedman y Kendal. Para FL $p$=0.686 para Wilcoxon y prueba de signos, y $p$=0.655 para Friedman y Kendall. Se observa que en todos los casos los valores $p$ son superiores a 0.05, es decir, en estos casos no existe de diferencia estadística significativa en el desempeño de los algoritmos. En el caso del

objetivo FC, el valor p=0.025, lo que indica que hay diferencia significativa que nos permite concluir que los resultados del estado del arte son mejores a la propuesta.

En el caso de las instancias difíciles (Tabla 4 y 5) donde se aprecia que la propuesta obtuvo malos resultados, las pruebas estadísticas solo confirman que, si existe diferencia significativa en el desempeño de los algoritmos, obteniendo los valores *p* en un rango de $[0.008, 0.046]$.

Algunos aspectos que se observaron durante la experimentación que son importantes de destacar son los siguientes:

Los resultados obtenidos fueron influenciados por la naturaleza contradictoria de los objetivos debido a que, por una parte, el objetivo de suavizado (FH, FL) busca crear agrupaciones pequeñas de vehículos al tratar de respetar las restricciones de proporción, y, por otra parte, el objetivo de color (FC) busca agrupar la mayor cantidad de vehículos del mismo color sin sobrepasar la restricción dura que es el tamaño del lote. Otro aspecto es que la implementación de reparadores cromosómicos en cada objetivo, que se estableció con la finalidad de obtener solo individuos factibles, disminuyo el conflicto de satisfacción de restricciones entre los objetivos, sin embargo, esto fue insuficiente para obtener resultados competitivos en general.

También se hizo evidente que las restricciones de proporción cercanas a la unidad como $2/3$ o $4/5$ en los objetivos de suavizado, provocan una mayor saturación de trabajo en las estaciones y mayor dificultad de crear una secuencia óptima. Los aspectos anteriores deben ser tomados para mejorar el desempeño de la propuesta.

## 7. Conclusiones

El objetivo de este artículo fue el estudio del desempeño del algoritmo de evolución diferencial en el problema de secuenciación de vehículos (no aplicado en el estado del arte) dotando su estructura clásica con un reparador cromosómico. Se observó que el algoritmo EDRC propuesto mostró, en general, competencia (39% de los casos), mejora (22% de los casos), no logrando competir en el 39% de los casos, particularmente en mayor porcentaje en las instancias difíciles.

El análisis de los resultados mostro la factibilidad del algoritmo EDRC en el problema propuesto y nos permiten establecer, como continuidad de la presente investigación, realizar mayor experimentación e incluir operadores de mutación especializados para cada uno de los objetivos del problema, con la finalidad de buscar mejorar y sustentar el desempeño del EDRC buscando soluciones significativamente mejores a las del estado del arte.

**Agradecimientos.** Los autores desean agradecer al Consejo Nacional de Ciencia y Tecnología (CONACYT) de México, a través de la beca para estudios de posgrado: 634738 (E. Sánchez) y al Tecnológico Nacional de México/Instituto Tecnológico de León, por el apoyo brindado para la realización de esta investigación.

## Referencias

1. Kis, T.: On the complexity of the car sequencing problem. Operations Research Letters 32, pp. 13–17 (2004)

2. Solnon, C.: The car sequencing problem: overview of state-of-the-art methods and industrial case-study of the ROADEF'2005 challenge problem. European Journal of Operational Research 191, pp. 912–927 (2008)
3. Zufferey, N.: Metaheuristics for production systems. Springer, Francia (2016)
4. Dincbas Mehmet, S.H V H P.: Solving the car-sequencing problem in constraint logic programming. In: Proceedings of the European Conference on Artificial Intelligence, pp. 290–295 (1988)
5. Ribeiro, C.C.: A hybrid heuristic for a multi-objetive real-life car sequencing problem with painting and assembly line constraints. European Journal of Operational Research 191, pp. 981–992 (2008)
6. Cordeau, G. J.-F.: Iterated tabu search for the car sequencing problem. European Journal of Operational Research 191, pp. 945–956 (2008)
7. Zinflou, C. a. M.: Design of an Efficient Genetic Algorithm to Solve the Industrial Car Sequencing Problem. Advances in Evolutionary Algorithms (2008)
8. Xiang-yang. Z., and H.Z.-D. Xiang.: A hybrid algorithm based on tabu search and large neigbourdhood search for car sequencing problem. Journal of Central South University 25, pp. 1–16 (2018)
9. Syswerda, G.: Schedule optimization using genetic algorithms. New York (1991)
10. Simon, D.: Evolutionary Optimization Algorithms. Wiley, Estados Unidos de América (2013)
11. Espinal-Dueñas, J.A.A.F.J.: Aplicación de un Reparador Cromosómico al PSO y GA para la solución del problema de las n-Reinas. En: CIINDET (2010)