Instituto Nacional Electoral

COMITÉ TÉCNICO
DISTRITACIÓN NACIONAL
2021-2023

MODELO MATEMÁTICO Y ALGORITMOS

OCTUBRE 2021

Índice general

Contenido

Capítulo 1

Modelo Matemático

Un Modelo de Optimización Combinatoria tiene dos componentes:

- Una Función Objetivo.
- Un Conjunto de Restricciones.

La tarea de la función objetivo, dentro del modelo de optimización, es calificar la calidad de las soluciones (o escenarios) que cumplen con todas las restricciones (soluciones factibles). Esta calificación se basa en tres criterios; el equilibrio poblacional en cada uno de los distritos, la forma de los distritos de acuerdo a su compacidad geométrica y los tiempos de traslado entre localidades de 2500 habitantes o más. La solución del modelo es cuando se obtiene aquel escenario cuyo valor objetivo es el menor posible. Este escenario óptimo puede ser único o puede que exista más de uno.

Otra tarea muy importante de la función objetivo es la de guiar a los algoritmos de búsqueda, del mejor escenario, a ir mejorando las soluciones que se van obteniendo de tal manera que al final encuentren la mejor o una de las mejores soluciones al modelo.

1.1. Función Objetivo

La función objetivo debe conducir a los algoritmos a encontrar distritos con tres características: menor desviación poblacional; las formas geométricas de los distritos sean lo más cercano a polígonos regulares; y los tiempos de traslado entre localidades de 2500 o más habitantes sean lo más cercanas posibles. Pero además los valores de estos tres objetivos deben ser comparables para que sean “competitivos” dentro de esta función.

1.1.1 Objetivo Poblacional

La desviación del promedio poblacional de cada distrito electoral, debe estar a ±15 %, esto significa que si *PD* = Población del distrito y *PM* = Población media estatal, entonces la población de cualquier distrito está dentro de este criterio si cumple con la siguiente desigualdad:

$$PM - 0.15PM \leq PD \leq PM + 0.15PM$$

Haciendo algunas operaciones algebraicas se llega a lo siguiente:

$$-0.15PM \leq PD - PM \leq +0.15PM$$

$$-1 \leq \frac{1}{0.15}\left(\frac{PD}{PM} - 1\right) \leq 1$$

o equivalentemente

$$0 \leq \left|\frac{1}{0.15}\left(\frac{PD}{PM} - 1\right)\right| \leq 1$$

De esta forma, para penalizar de forma más estricta a distritos fuera de rango, se toma un valor de *a* mayor que 1, como se muestra en la siguiente expresión:

$$0 \leq \left|\frac{1}{0.15}\left(\frac{PD}{PM} - 1\right)\right|^{a} \leq 1 \tag{1.1}$$

Se probaron algunos valores de $a \geq 1$ tales como: 1, 1.2, 1.5, 1.8 y 2 y después de varias experimentaciones se llegó a que el valor apropiado de *a* es igual a 2, ya que, para valores menores de dos, en algunos ejercicios de prueba, se obtuvieron distritos fuera de rango.

En virtud de lo anterior, la fórmula de equilibrio poblacional, para un distrito *D*, es igual a:

$$C_1(D) \leq \left(\frac{1-\frac{PD}{PM}}{0.15}\right)^2 \tag{1.2}$$

Donde:

PD = Población del distrito *D*,

PM = Población media estatal,

0.15 = Desviación máxima permitida de la media estatal.

Se puede observar que $C_1(D)$ es un valor adimensional y que posteriormente se puede comparar con los otros dos objetivos.

1.1.2. Objetivo Compacidad Geométrica

La compacidad geométrica de cada distrito electoral, se puede entender como la situación en la que el perímetro de los distritos adquiera una forma geométrica lo más cercana a un polígono regular. Este criterio se incluye en la función objetivo. La función que evalúa la compacidad se definirá como un cociente del perímetro del distrito entre la raíz cuadrada del área del mismo, multiplicada por un coeficiente definido experimentalmente para normalizarla con respecto al objetivo del equilibrio poblacional. En la literatura sobre estudios de diseño de zonas, y en particular sobre distritación electoral, se proponen muchas medidas de compacidad tales como envolventes conexas, cuadriculado, comparación de círculos inscritos, etc.; una fórmula para el cálculo de la compacidad geométrica que da muy buenos resultados además que su cálculo numérico requiere de pocas operaciones es:

$$k * \left(\frac{\text{Perímetro del distrito}}{\sqrt{\text{Área del distrito}}} \right) \tag{1.3}$$

Donde *k* es una constante que hace la expresión 1.3 igual a uno dependiendo del polígono regular que se quiera. Por ejemplo, si deseamos que el polígono regular sea un triángulo equilátero la expresión 1.3 es igual a uno, *k* = 0.21935; para un cuadrado, *k* = 0.25, etc. En general para un polígono regular de *m* lados la expresión 1.3 vale uno, si el valor de *k* es igual a:

$$k = \frac{1}{2\sqrt{m \tan\left(\frac{\pi}{m}\right)}} \tag{1.4}$$

En la Tabla 1.1 se calculan algunos valores de *k* usando la expresión 1.4, tomando polígonos regulares con *m* lados, para *m* = 3, 4, ..., 10, ...∞.

Por lo expresado anteriormente, para el cálculo de la compacidad geométrica de un distrito *D* se propone la expresión:

$$C_1(D) = \left(\frac{\text{Perímetro del distrito } D}{\sqrt{\text{Área del distrito } D}} * 0.25 - 1 \right) \tag{1.5}$$

m	*k*
3	0.21935
4	0.25000
5	0.26233
6	0.26864
7	0.27233
8	0.27467
9	0.27626
10	0.27738
—	—
∞	0.28209

Tabla 1.1: Valores de *k* calculados con la expresión 1.4 tomando polígonos regulares de *m* lados.

Donde 0.25 corresponde al coeficiente que hace la expresión 1.3 igual a 1 si la forma del distrito corresponde a un cuadrado. Se eligió un cuadrado porque dentro de los polígonos regulares, los únicos capaces de formar una teselación son: el triángulo equilátero, el cuadrado y el hexágono. Por lo que se consideró restar a la expresión anterior la cantidad 1 para permitir que la mejor compacidad se encuentre cercana a cero y sea exactamente igual a cero cuando se forme un distrito de forma cuadrada.

1.1.3. Objetivo Tiempos de Traslado

Es muy importante que los distritos electorales, en su interior, estén lo mejor comunicados que se pueda, por lo que la tercera componente en la función objetivo se refiere a la comunicación del

distrito. Para obtener la tercera componente, sea $E = \{D_1, D_2, \dots, D_n\}$ un escenario de un estado dividido en n distritos y sean $l_1, l_2, \dots, l_k$ los centros de todas las localidades de más de 2,500 habitantes del estado **asociados a una sección** en particular. Sea $L = \{l_1, l_2, \dots, l_k\}$ el conjunto de todos los centros de las localidades. El escenario E induce una partición en conjuntos de localidades $P = \{L_1, L_2, \dots, L_n\}$ donde $l_i \epsilon L_j$ si el centroide l_i cae dentro del distrito D_j.

Un indicador del tiempo de traslado dentro de un distrito D_j, viene dado por:

$$T_{D_j} = \sum_{l_p \epsilon L_j} \sum_{l_q \epsilon L_j} t(l_p, l_q) \text{ tomando en cuenta que } t(l_p, l_q) = 0 \text{ si } l_p = l_q$$

donde $t(l_p, l_q)$ es el tiempo de traslado del centroide l_p al l_q.

Cada doble sumatoria tiene $|L_j| \cdot (|L_j| - 1)$ sumandos, donde $|\cdot|$ es el número de elementos que tiene el conjunto $(\cdot)$. Así que el tiempo promedio de traslado en el distrito D_j viene dado por:

$$\bar{T}_{D_j} = \frac{\sum_{l_p \epsilon L_j} \sum_{l_q \epsilon L_j} t(l_p, l_q)}{|L_j| \cdot (|L_j| - 1)} \tag{1.6}$$

Un indicador de los tiempos de traslado del escenario E es igual a:

$$\bar{T}_{D_1} + \bar{T}_{D_2} + \cdots + \bar{T}_{D_n}$$

Estos valores dependen del escenario E, si el escenario cambia, los valores cambian.

Ahora, si se calcula el tiempo de traslado dentro de todo el estado es igual a:

$$T_{Es} = \sum_{l_p \epsilon L} \sum_{l_q \epsilon L} t(l_p, l_q) \text{ tomando } t(l_p, l_q) = 0 \text{ si } l_p = l_q \tag{1.7}$$

Pero esta última doble sumatoria tiene todas las sumatorias en $T_1, T_2, \dots, T_n$ y más, ya que $k = |L_1| + |L_2| + \cdots + |L_k|$ así que T_{Es} tiene $k(k-1)$ términos, es decir,

El tiempo promedio de traslado dentro de todo el estado es:

$$\bar{T}_{Es} = \frac{\sum_{l_p \epsilon L} \sum_{l_q \epsilon L} t(l_p, l_q)}{k(k-1)} \tag{1.8}$$

$\bar{T}_{Es}$ no depende del escenario, por lo que es un valor inherente a la entidad federativa en cuestión e indica que tan bien está comunicado el estado.

Si tomamos como índice de tiempos de traslado para el escenario *E* el siguiente:

$$I_E = \frac{\bar{T}_{D_1} + \bar{T}_{D_2} + \cdots + \bar{T}_{Dn}}{\bar{T}_{Es}} \tag{1.9}$$

tenemos un índice adimensional cuyo valor indica que tan comunicado está el escenario propuesto. Valores "grandes" indicarían que el escenario propuesto está mal comunicado en uno o más distritos y valores "pequeños" indicarían que el escenario tiene buena comunicación dentro de sus distritos.

Por lo expresado anteriormente, para el cálculo del tiempo de traslado dentro de un distrito *D* se propone la expresión:

$$C_3(D) = \frac{\bar{T}_D}{\bar{T}_{Es}} \tag{1.10}$$

tenemos un índice adimensional cuyo valor indica qué tan comunicado está el distrito *D* propuesto. Valores "grandes" de $C_3(D)$ indicarían que el distrito propuesto está mal comunicado y valores "pequeños" indicarían que el distrito tiene buena comunicación.

1.1.4. Función Objetivo del Modelo Matemático

La función multiobjetivo que se propone para la distritación electoral será la suma ponderada del equilibrio poblacional (ecuación (1.2), la compacidad geométrica (ecuación (1.5) y los tiempos de traslado (ecuación (1.10) dada por la siguiente expresión:

$$f(E) = \sum_{j=1}^{k} \left(\frac{1 - {P_{D_j}}/{P_M}}{0.15} \right)^2 + 0.5 \sum_{j=1}^{k} \left(\frac{\textit{Perímetro del distrito } j}{\sqrt{\textit{Área del distrito}}} * 0.25 - 1 \right) + 0.4 \sum_{j=1}^{k} \frac{\bar{T}_j}{\bar{T}_{Es}} \qquad (1.11)$$

Donde, en la expresión (1.11), $E = \{D_1, D_2, \dots, D_k\}$ es un escenario o plan distrital y el factor 0.5 y 0.4 refleja la importancia relativa de la compacidad geométrica y los tiempos de traslado con respecto al equilibrio poblacional. Se asigna al equilibrio poblacional un peso de 1, el doble con respecto a la compacidad geométrica, y de 2.5 veces, con respecto a los tiempos de traslado, debido a la importancia relativa de los tres objetivos.

1.2. Restricciones del Modelo

Las restricciones que el modelo considera son las siguientes:

1. El estado se dividirá en exactamente *n* distritos electorales de acuerdo a la constitución federal o estatal según sea el caso de distritación federal o local.
2. Se conformarán distritos con municipios que cuenten con un porcentaje 40% o más de población indígena; de acuerdo a una tipología de municipios.
3. Integridad municipal, al considerar a los municipios que en forma integral se agruparán para formar distritos. Todo aquel municipio que pueda ser separado en un número entero de distritos dentro del 15 por ciento de desviación poblacional le serán asignados ese número de distritos y serán divididos hacia su interior; de acuerdo a una tipología de municipios.
4. Los distritos serán continuos y contiguos.

Capítulo 2

Recocido Simulado.

Recocido simulado es una de las técnicas heurísticas más conocidas por su simplicidad y buenos resultados en numerosos problemas, se ha convertido en una herramienta muy popular, con aplicaciones en diferentes áreas de optimización. El concepto fue introducido en el campo de la optimización combinatoria a inicios de la década de los 80 por Kirkpatrick [5] y Cerny [4]. Esta heurística, se inspira en una analogía entre el proceso de recocido de sólidos y la forma en que se resuelven problemas de optimización combinatoria. Dicha analogía resulta importante para comprender la forma en que trabaja este algoritmo.

El recocido de sólidos es un proceso de tratamiento térmico que se aplica a varios materiales como el vidrio y ciertos metales y aleaciones para hacerlos menos quebradizos y más resistentes a la fractura. El objetivo de este proceso es minimizar la energía interna de la estructura atómica del material y eliminar posibles tensiones internas provocadas en las etapas anteriores de su procesado. Para lograrlo, los metales ferrosos y el vidrio se recuecen calentándolos a alta temperatura y enfriándolos lentamente. Cada vez que se baja la temperatura, las partículas se reacomodan en estados de más baja energía hasta que se obtiene un sólido con partículas acomodadas conforme a una estructura con energía mínima.

El algoritmo de recocido simulado está basado en técnicas de Monte Carlo y genera una sucesión de estados del sólido de la siguiente manera. Dado un estado actual i del sólido con energía S_i, entonces el siguiente estado j es generado al aplicar una pequeña perturbación al estado actual. La temperatura del nuevo estado es S_j. Si la diferencia de energía $S_i - S_j$, es menor o igual que cero, el estado j es aceptado como estado actual. Si la diferencia es mayor que cero, el estado j es aceptado con una probabilidad dada por:

$$\exp\left(\frac{S_i - S_j}{k_B T}\right) \tag{2.1}$$

Donde T denota la temperatura a la cual se encuentra el sólido y k_B es una constante física llamada la constante de Boltzmann. Este criterio de aceptación es conocido como el criterio de Metrópolis. Si el decremento de la temperatura es suficientemente lento, el sólido alcanzará un equilibrio térmico en cada temperatura.

2.1. Algoritmo de Recocido Simulado

La técnica de recocido simulado, se inspiró en el hecho de que el algoritmo de Metrópolis puede ser utilizado para generar soluciones a problemas de optimización combinatoria si se hacen las siguientes consideraciones:

- Las soluciones del problema de optimización son equivalentes a los estados del sólido.
- El costo de una solución, denotado por $f(E_i)$, es equivalente a la energía de cada estado.
- La temperatura será sustituida por un parámetro de control que regulará la probabilidad de aceptación de nuevas soluciones.

El algoritmo de Recocido Simulado comienza con una solución inicial y una temperatura inicial T_0. Se recomienda que al inicio del algoritmo la temperatura sea suficientemente alta para permitir todo, o casi todo, movimiento, es decir, que la probabilidad de pasar de la solución actual E_i a la solución vecina E_i sea muy alta, sin importar la diferencia entre los costos de ambas soluciones, $f(E_i) - f(E_j)$.

En cada iteración se genera una solución vecina de manera aleatoria, si la nueva solución mejora el valor de la función objetivo con respecto a la solución actual, esta última es reemplazada. Cuando la nueva solución no mejora el valor de la función objetivo, se puede aceptar el cambio de la solución actual con cierta probabilidad dada por:

$$\exp\left(\frac{f(E_i) - f(E_j)}{k_B T}\right) \tag{2.2}$$

Donde, $f(E_i)$ es el costo de la solución actual, $f(E_j)$ es el costo de la solución vecina y T es la temperatura del proceso. Conforme el algoritmo avanza, el valor de la temperatura disminuye mediante un coeficiente de enfriamiento, $0 < \alpha < 1$, pero cada valor de T se mantiene constante durante L iteraciones, para permitir que el algoritmo explore distintas soluciones con la misma probabilidad de aceptación.

Debe observarse que, al inicio, cuando la temperatura es alta, se tiene una mayor probabilidad de aceptar soluciones de menor calidad, lo cual permite la exploración del espacio de soluciones y evita la convergencia prematura a mínimos locales. Sin embargo, conforme el valor de la temperatura disminuye, el algoritmo se hace más selectivo y difícilmente acepta soluciones de menor calidad, iniciando una búsqueda que lo guía hacia un mínimo local. Finalmente, el algoritmo se detiene cuando la temperatura alcanza un valor límite, T_f, y devuelve la mejor solución encontrada.

Los parámetros de temperatura inicial T_0, coeficiente de enfriamiento α, temperatura final T_f y número de soluciones visitadas en cada temperatura L, son conocidos como *programa de enfriamiento* y juegan un papel importante en el desarrollo del algoritmo. Si se utilizan valores demasiado grandes el tiempo de ejecución podría ser excesivo, mientras que valores demasiado pequeños podrían provocar una convergencia prematura a un mínimo local. Para mayores detalles sobre esta técnica ver [1].

Sea T_f denote el valor de la temperatura y L_k el número de transiciones generadas en la *k*-ésima iteración del algoritmo de Metropolis. El algoritmo de Recocido Simulado puede describirse en pseudo-código como se muestra en el Algoritmo 1.

El algoritmo de Recocido Simulado comienza llamando a un procedimiento de inicialización donde se definen la solución inicial, parámetro de control inicial y el número inicial de generaciones necesarias para alcanzar el equilibrio térmico para la temperatura inicial. La parte medular del algoritmo consta de dos ciclos. El externo **Repite**...**hasta** y el interno **Para**... **finpara**. El ciclo interno mantiene fija la temperatura hasta que se generan L_k soluciones y se acepta o se rechaza la solución generada conforme el criterio de aceptación ya discutido. El ciclo externo disminuye el valor de la temperatura mediante el procedimiento CALCULA-CONTROL y calcula

el número de soluciones a generar para alcanzar equilibrio térmico mediante el procedimiento CALCULA-LONGITUD. Este ciclo finaliza cuando la condición de paro se cumple.

Algoritmo 1: Algoritmo de Recocido Simulado en pseudo-código

```
1  INICIALIZA (E_{i_inicial}, T_0, L_0)
2  k := 0
3  E_i := E_{i_inicial}
4  Repite
5      Para l := 1 a L_k hacer
6          GENERA (E_j vecino de E_i)
7          si f(E_j) ≤ f(E_i) entonces
8              E_i := E_j
9          fin
10         en otro caso
11             si exp((f(E_i)−f(E_j))/T_k) > número aleatorio en [0,1) entonces
12                 E_i := E_j
13             fin
14         fin
15     fin
16     k := k + 1
17     CALCULA-LONGITUD (L_k)
18     CALCULA-CONTROL (T_k)
19 hasta Cumplir criterio de paro;
20 Termina algoritmo
```

Un rasgo característico del algoritmo de Recocido Simulado es que, además de aceptar mejoras en el costo, también acepta soluciones peores en costo. Inicialmente, para valores grandes de T, puede aceptar grandes soluciones deterioradas; cuando T decrece, únicamente pequeñas desviaciones serán aceptadas y finalmente, cuando el valor de T se aproxima a cero, no se aceptarán desviaciones. Este hecho significa que el algoritmo de Recocido Simulado tiene la capacidad de escapar de mínimos locales.

Note que la probabilidad de aceptar desviaciones está implementada al comparar el valor de $\exp\big((f(E_i) - f(E_j))/T\big)$ con un número aleatorio generado de una distribución uniforme en el intervalo [0,1). Además, debe ser obvio que la velocidad de convergencia del algoritmo está determinada al escoger los parámetros L_k y T_k, k = 0,1,… Si los valores T_k decrecen rápidamente o los valores de L_k no son grandes, se tendrá una convergencia más rápida que cuando los valores de T_k decrecen lentamente o los valores de L_k son grandes.

2.2. Aplicación a Distritos Electorales

La creación de distritos electorales, mediante recocido simulado, se realiza en dos etapas. Primero se crea una solución inicial formada por r distritos conexos, donde r es el número de distritos indicado para cada conjunto territorial. Posteriormente, se realiza un proceso de mejora destinado a explorar diferentes soluciones, a partir de la solución inicial, de tal forma que al final del proceso se obtenga una solución de buena calidad. Estas etapas se describen con más detalle en las siguientes secciones.

2.2.1 Solución inicial

El primer paso del algoritmo consiste en construir una solución inicial con distritos conexos, para lo cual selecciona de manera aleatoria r Unidades Geográficas (UG) que asigna a distritos diferentes, y las marca como UG no disponibles. Después se realizan las siguientes instrucciones hasta que todas las UG están marcadas como no disponibles:

- Elegir un distrito.
- Generar una lista con las UG disponibles que colindan con el distrito seleccionado.
- Seleccionar al azar una UG de la lista.
- Incluir en el distrito la UG elegida y marcarla como no disponible.

De esta forma se obtiene una solución con r distritos conexos ajenos que incluyen a todas las UG, cuya calidad no necesariamente es buena pero que podrá mejorarse en el proceso de búsqueda.

2.2.2 Solución vecina

El algoritmo de RS inicia con la construcción de una solución con distritos conexos, como ya se explicó en la sección 2.2.1, y durante el proceso de búsqueda y mejora, se garantizará que las nuevas soluciones conserven esta característica.

Para generar una solución vecina se elige de manera aleatoria un distrito, D, y se genera una lista con las UG que pueden ser enviadas a un distrito contiguo. Por lo tanto, en la lista se incluyen las UG que se encuentran en colindancia con otros distritos. Se selecciona aleatoriamente una UG, i, de la lista, y se cambia al distrito con el cual colinda, D'; en caso de que colinde con dos o más distritos se hace una elección aleatoria. En caso de que el distrito elegido inicialmente esté formado por una sola UG se evita el cambio, ya que esto implicaría una disminución en el número de distritos.

Cuando el cambio de la UG provoca una desconexión en D se recupera la conexidad de la siguiente forma:

1) Se identifican las diferentes componentes conexas en que se dividió D.
2) Se cuenta el número de UG que forman a cada componente conexa.
3) La componente conexa con el mayor número de UG es considerada como el distrito original, D.
4) El resto de las componentes conexas es enviado a D' junto con i.

Siguiendo este procedimiento, cada solución vecina es una solución con distritos conexos que se diferencia de la anterior por la ubicación de un conjunto de UG. Se debe mencionar que es importante elegir de manera aleatoria las UG que son cambiadas para evitar que el algoritmo favorezca algunas soluciones y para aumentar las posibilidades de visitar un mínimo global.

Capítulo 3

Algoritmo ABC-RS

La idea de combinar varios conceptos y técnicas de diferentes algoritmos es una de las líneas de investigación más fuertemente desarrolladas en el campo de la optimización desde los años noventa [2]. El objetivo de una hibridación es crear técnicas mejores y más robustas, que combinen y exploten las ventajas de diferentes métodos.

En la literatura especializada, no existe un consenso sobre el concepto **heurística híbrida** [3, 7, 8]. Sin embargo, en términos generales, una técnica heurística híbrida puede definirse como una estrategia robusta que combina de forma armónica las características de diferentes heurísticas. El objetivo principal de generar un híbrido es explotar de forma sinérgica los beneficios de diferentes técnicas de optimización, mientras que sus aspectos negativos se disminuyen, de tal forma que permite producir un algoritmo capaz de resolver problemas difíciles de forma eficiente. Para lograrlo, se requiere la combinación adecuada de ideas, paradigmas o conceptos complementarios.

La creación de una estrategia híbrida es un proceso que implica conocimiento y creatividad. Se requieren conocimientos sobre el problema de interés, métodos heurísticos y técnicas de optimización, de tal forma que se puedan seleccionar los elementos que serán utilizados en el híbrido. Por otra parte, se requiere la creatividad para combinar adecuadamente estas ideas. Por lo anterior, se puede decir que la hibridación no es un proceso trivial, que se basa en la experiencia, la creatividad y la inteligencia para la creación de un algoritmo capaz de resolver de forma eficiente problemas de optimización difíciles [6]. Por lo tanto, una combinación adecuada de ideas complementarias, paradigmas o conceptos de optimización puede ser la clave para lograr el máximo rendimiento en la resolución de muchos problemas.

Existen diferentes taxonomías, o clasificaciones, de las técnicas de hibridación. Por ejemplo, Talbi propuso en [8] una clasificación basada en las funciones y en la arquitectura del algoritmo híbrido. De acuerdo a esta clasificación, se pueden considerar algoritmos híbridos de bajo y de alto nivel. Un híbrido es de bajo nivel cuando una de las estrategias de optimización de una técnica heurística es reemplazada por otra heurística. Por otro lado, cuando varias heurísticas trabajan dentro del

mismo algoritmo se trata de un híbrido de alto nivel. Los híbridos de alto nivel se subdividen en dos categorías: por relevos y cooperativos. Se dice que es por relevos cuando la salida de una técnica es la entrada de otra, y se le llama cooperativos cuando varias estrategias trabajan por separado y periódicamente comparten información. Bajo esta taxonomía, el algoritmo ABC-RS puede clasificarse como un híbrido cooperativo de alto nivel que combina técnicas de recocido simulado (RS) y de colonia de abejas artificiales (ABC).

En la siguiente sección se da una descripción de la forma en que opera el algoritmo ABC-RS.

3.1 Aplicación a Distritos Electorales

Por ser un algoritmo híbrido que incluye recocido simulado, hereda muchos de los conceptos mencionados en la sección 2, por ejemplo, será necesario establecer la temperatura inicial, T_0, la temperatura final, T_f, el procedimiento CALCULA-LONGITUD para determinar el número de iteraciones que se realizará en cada temperatura y el procedimiento CALCULA-CONTROL empleado para disminuir la temperatura.

Asimismo, hereda algunas de las características de la técnica colonia de abejas artificiales (ABC), la cual no será empleada en el presente proceso de distritación, por lo que no se describirá de forma completa, pero se usará parte de su notación, por ejemplo, toda solución será llamada fuente de alimento y se representará como $E = \{D_1, D_2, .., D_n\}$, donde D_s es un conjunto de UG para $s = 1,2, \dots, n$. Además, se trabajará con un conjunto inicial de M fuentes de alimento.

Al iniciar el algoritmo, se genera una población de M fuentes de alimento utilizando la estrategia descrita en la sección 2.2.1, de tal manera que cada solución está formada por n distritos conexos.

Posteriormente, cada fuente de alimento, E_i, entra a un proceso de búsqueda y mejora de acuerdo a la siguiente estrategia.

Primero se emplea RS. Se genera una solución vecina, E_j, como se describe en la sección 2.2.2. Si E_j mejora el valor de la función objetivo con respecto a E_i, entonces esta última es reemplazada

por E_j. En caso contrario, E_j puede reemplazar a E_i con una probabilidad dada por la Ecuación 2.2.

Posteriormente, se elige de forma aleatoria una solución $E_{Destino}$, la cual es combinada con la solución E_i siguiendo los siguientes pasos.

Se elige una UG, u, de forma aleatoria. Entonces, existen un distrito $D_k \in E_i$ y un distrito $D_j \in E_{Destino}$ tales que $u \in D_k \cap D_j$. Ahora se deben considerar los siguientes conjuntos:

$$H_1 = \{l: x_{lk} = 0, x_{lj} = 1\} \tag{3.1}$$

$$H_2 = \{l: x_{lk} = 1, x_{lj} = 0\} \tag{3.2}$$

Donde:

$$x_{lk} = \begin{cases} 1, & \text{si la UG } l \text{ pertenece al distrito } k \\ 0, & e\text{n otro caso} \end{cases}$$

Entonces una UG en H_1 es insertada en D_k, y una UG en H_2 es extraída de D_k, e insertada en un distrito contiguo a D_k.

Es importante observar que estos movimientos pueden producir desconexión en el distrito D_k, por lo que debe realizarse un proceso que la repare. Para esto, se cuenta el número de componentes conexas en D_k. Si el número de componentes conexas es 1, entonces el distrito no perdió su conexidad. En otro caso, la componente conexa que contiene a u (i.e., la UG usada en el proceso de combinación mencionado anteriormente) se define como el distrito D_k, mientras que el resto de las componentes son asignadas a otros distritos adyacentes.

Una vez que han concluido los procesos de combinación y reparación antes mencionados, se obtiene una nueva fuente da alimento E'_i. Si la nueva solución E'_i tiene un costo mejor que E_i, E'_i sustituye a E'_i y se convierte en una nueva fuente de alimento. En otro caso, E'_i es rechazada y E_i es conservada.

En cada iteración se lleva una estadística del número de veces que cada fuente de alimento sufre un cambio. Si una fuente de alimento no cambia después de un número N de iteraciones, entonces

una abeja exploradora sustituye esta fuente de alimento usando la estrategia descrita en la sección 2.2.1.

Este proceso de creación y reemplazo de soluciones se repite durante L_k iteraciones, dadas por el procedimiento CALCULA-LONGITUD, al término de las cuales se pasa a la siguiente fuente de alimento, E_{i+1}, en la cual se aplican los mismos pasos. Cuando todas las fuentes de alimento han sido afectadas por este ciclo, se realiza un decremento en la temperatura mediante el procedimiento CALCULA-CONTROL, y se ejecuta nuevamente el ciclo. Este proceso termina cuando se alcanza la temperatura final propuesta por el usuario. En el Algoritmo 2 se presenta el pseudocódigo del algoritmo ABC-RS.

Algoritmo 2: Algoritmo ABC-RS en pseudo-código

```
1  INICIALIZA (E_{i_inicial}, T_0, L_0)
2  k := 0
3  E_i := E_{i_inicial}
4  Generar de forma aleatoria M fuentes de alimento E_1, ..., E_M.
5  Repite
6      Para i := 1 a M hacer
7          Para l := 1 a L_k hacer
8              Inicia RS
9              GENERA (E_j vecino de E_i)
10             si f(E_j) ≤ f(E_i) entonces
11                 E_i ⟵ E_j
12             fin
13             en otro caso
14                 si exp((f(E_i)−f(E_j))/T_k) > número aleatorio en [0,1) entonces
15                     E_i ⟵ E_j
16                 fin
17             fin
18             Seleccionar de forma aleatoria una fuente de alimento E_Destino.
19             Combinar las fuentes de alimento E_i y E_Destino para obtener una
               fuente de alimento nueva E'_i.
20             Determinar el costo de la solución E'_i mediante la función objetivo.
21             si la nueva solución es mejor entonces
22                 E_i ⟵ E'_i
23             fin
24         fin
25     fin
26     k := k + 1
27     CALCULA-LONGITUD (L_k)
28     CALCULA-CONTROL (T_k)
29 hasta Cumplir criterio de paro;
30 Termina algoritmo
```

Bibliografía

1) S. G. de los Cobos, J. Goddard, M. A. Gutiérrez y A. E. Martínez, "Búsqueda y exploración estocástica", Ed. México: UAM-I (2010)
2) C. Blum, J. Puchinger, G. R. Raidl y A. Roli, "A brief survey on hybrid metaheuristics", 4th International Conference on Bioinspired Optimization Methods and their Applications, pp 3-16 (2010)
3) C. Blum, J. Puchinger, G. R. Raidl, A. Roli, "Hybrid metaheuristics in combinatorial optimization: A survey". Applied Soft Computing, 11(6), pp.4135-4151 (2011)
4) V. Cerny, "A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm", Journal of Optimization Theory and Applications, 45, pp. 41-55 (1985)
5) S. Kirkpatrick, C. D. Gellat y M. P. Vecchi, "Optimization by simulated annealing", Science, 220, pp. 671-680 (1983)
6) J. T. Palma-Méndez y R. Marín-Morales, "Inteligencia artificial. Técnicas, métodos y aplicaciones". Mc Graw Hill (2008)
7) R. G. Raidl, "A unified view on hybrid metaheuristics". Hybrid Metaheuristics, Springer Berlin Heidelberg (2006)
8) E. G. Talbi, "A taxonomy of hybrid metaheuristics". Journal of heuristics,8(5), 541-564 (2002)