UNIVERSIDAD BERNARDO O'HIGGINS

Algoritmo de Optimización Anaconda Verde

Magíster Ingeniería informática

Proyecto de Tesis de Grado

Profesor: Broderick Crawford

Paulo Salinas Lueyza

Contenido

RESUMEN 3
I. INTRODUCCIÓN 3
I-A. Metaheurística: 3
I-B. Optimización Anaconda Verde: 3
I-C. Set Covering Problem 4
II. DEFINICION DE OBJETIVOS 4
II-A. Objetivo general 4
II-B. Objetivos Específicos 4
II-C. Estado del Arte 5
III. SET COVERING PROBLEM 5
III-A. Formulación Matemática 5
III-B. Funciones de Binariazación 6
III-B1. Función de Binariazación con forma sigmoide: 7
IV. GREEN ANACONDA OPTIMIZATION (GAO) 7
IV-A. Exploración 8
IV-B. Explotación 11
IV-C. Seudocódigo del algoritmo 11
V. GOOGLE COLAB 12
Hardware: 12
Software: 13
Almacenamiento: 13
V. RESULTADOS 13
VI. CONCLUSIONES 13
REFERENCIAS 13

Los páginas no tienen número ... ¿?

RESUMEN

En este artículo se aborda la resolución del problema de Set Covering Problem (SCP) utilizando el algoritmo de optimización Green Anaconda Optimization (GAO). El Set Covering Problem es un desafío esencial en la teoría de optimización, donde se busca encontrar una forma eficiente de cubrir un conjunto de elementos con un conjunto de subconjuntos predefinidos. El GAO, inspirado en el comportamiento de la anaconda verde, propone una estrategia bioinspirada para la búsqueda de soluciones optimas. El algoritmo GAO se basa en dos fases: exploración y explotación. Durante la exploración, las "hembras" candidatas se seleccionan en base a la función objetivo y las soluciones se actualizan en conformidad con las probabilidades y desplazamientos aleatorios. La fase de explotación busca mejorar las soluciones cerca de los resultados descubiertos. El artículo presenta el proceso de GAO en detalle y propone un seudocódigo del algoritmo. Se espera que la combinación del GAO y el SCP ofrezca una nueva perspectiva para abordar problemas de optimización desafiantes.

Index Terms: Optimización, Set Covering Problem, Green Anaconda Optimization, Metaheurística, Algoritmo bioinspirado, Exploración, Explotación, Solución óptima.

I. INTRODUCCIÓN

I-A. Metaheurística:

Las metaheurísticas son algoritmos que revolucionan la resolución de problemas complejos, dirigiendo la búsqueda de soluciones optimas mediante la imitación de procesos naturales [1]. A diferencia de los algoritmos tradicionales, son flexibles y se adaptan para resolver diversos desafíos en distintas áreas [2]. Estas herramientas poderosas han transformado la manera en que abordamos problemas insolubles, logrando soluciones de alta calidad en la optimización moderna, si bien este tipo de algoritmos representa una solución ideal para encontrar solucione optimas de cara a la capacidad de cómputo necesaria resolver, suelen quedar se en mínimos locales, es por lo cual es posible apreciar un gran número de metaheurísticas que buscan acercarse de mejor manera a soluciones más optimas.

I-B. Optimización Anaconda Verde:

Para efectos de este informe es que se determina utilizar el algoritmo de optimización llamado GAO (Green Anaconda Optimization), que trata de un algoritmo bioinspirado en como su nombre lo indica en la anaconda verde (Eunectes murinus) es una especie de boa que habita en Sudamérica y es

conocida por varios nombres, como anaconda común, anaconda gigante, boa de agua común o sucuri. Es una de las más grandes y pesadas serpientes, similar a un boa, pero con la característica que no es venenosa, puede alcanzar una longitud de hasta 5,21 metros y pesar entre 30 y 70 kg, su color es verde oliva con manchas negras a lo largo del cuerpo y tiene una cabeza estrecha con rayas de color amarillo anaranjado, sus ojos están posicionados de modo que le permite nadar sin exponer su cuerpo fuera del agua, su dieta incluye peces, aves, reptiles y mamíferos, y ocasionalmente cazas presas más grandes. La anaconda verde pasa la mayor parte del tiempo en el agua y se esconde bajo la superficie para emboscar a sus presas, su estrategia de caza implica golpear, envolver y asfixiar a la presa, estos comportamientos naturales son fundamentales en el diseño de la metaheurística GAO.

I-C. Set Covering Problem

El proyecto propone resolver el problema de conjuntos SCP o Set Covering Problem, este es un desafío fundamental en la teoría de la optimización y la teoría de la complejidad. En este problema, se busca encontrar la manera más eficiente de cubrir un conjunto de elementos con un conjunto de subconjuntos predefinidos, donde cada subconjunto contiene ciertos elementos y el objetivo es seleccionar un subconjunto mínimo que cubra todos los elementos al menos una vez. El problema tiene aplicaciones en diversos campos, como la planificación de rutas, asignación de recursos, diseño de redes y otros escenarios en los que se debe optimizar la asignación de elementos a conjuntos para satisfacer ciertas restricciones.

II. DEFINICION DE OBJETIVOS

II-A. Objetivo general

La resolución de Set Covering Problem (SCP) utilizando un algoritmo bioinspirado, en específico GAO.

II-B. Objetivos Específicos

1. Resolver SCP con algoritmo GAO en lenguaje Python.
2. Analizar calidad de la solución en relación con la capacidad para aproximarse a mínimo o máximo global según sea el caso.
3. Comparación de solución con otros algoritmos bioispirados.
4. Implementación de solución en plataforma Google Colab y Google Firebase.

II-C. Estado del Arte

En la actualidad el problema SCP es un problema ampliamente resulto con diferentes técnicas metaheurísticas, al tratarse de un estándar dentro del conjunto de problemas a solucionar cada vez que se presenta una técnica nueva a la comunidad. En este proyecto se busca abordar la solución de SCP desde una perspectiva nueva utilizando una técnica para la cual no tenemos conocimiento de que se haya resuelto este tipo de problemas, al tratarse GAO de una metaheurística recientemente publicada.

El problema SCP se ha resuelto anteriormente con algoritmos bioisnpirados con un gran buen performance, teniendo una gran numero de publicación que se pueden visitar ya sea para efectos de análisis o comparativa con GAO, por ejemplo, en [3], él hablaba sobre un sistema de hormigas (ant system), también se encuentra el trabajo realizado en [4], el cual consiste en presentar nuevas ideas para aplicar la metaheurística ACO; [5], realizaron una comparación entre dos algoritmos de la metaheurística ACO para resolver el Set Covering Problem, también se desarrolló un trabajo relacionado con el SCP en [6], él cual consistía en la realización de un sistema paralelo de la metaheurística ACO para resolver el Set Covering Problem, además está el trabajo realizado en [7], el cual consistía en un algoritmo ACO orientado linealmente, para resolver el SCP

III. SET COVERING PROBLEM

El Set Covering Problem es un desafío fundamental en la teoría de la optimización combinatoria. En este problema, se enfrenta la tarea de seleccionar un conjunto mínimo de subconjuntos de una colección predefinida para cubrir todos los elementos de un conjunto dado. Cada subconjunto tiene un costo asociado y el objetivo es minimizar la suma de los costos de los subconjuntos seleccionados, mientras se garantiza que cada elemento este cubierto al menos una vez. Formalmente, dado un conjunto universal U con n elementos y un conjunto de subconjuntos $S = \{S_1, S_2, \ldots, S_m\}$, donde S_i contiene los elementos que cubren, se busca encontrar un subconjunto $C \subseteq S$ de manera que la unión de los elementos en C sea igual a U y el costo total sea mínimo.

III-A. Formulación Matemática

Sea:

- U un conjunto universal con m elementos.
- $S = \{S_1, S_2, \ldots, S_m\}$ un conjunto de subconjuntos, donde S_i contiene los elementos que cubren.

- $C \subseteq S$ un subconjunto de manera que la union de los elementos en C sea igual a U y el costo total sea mínimo.
- c_i *el costo del subconjunto* S_i.
- x_i una variable binaria que indica si el subconjunto S_i es seleccionado o no.

El objetivo es minimizar la siguiente función:

$$min \sum_{i=1}^{m} c_i * x_i$$

Se sugiere usar modelo matemático matricial (meta a_{ij})

Sujeto a las siguientes restricciones:

$$\sum_{S_j \, contiene \, e_i} x_j \geq 1 \; \forall e_i \in U$$

Donde x_j es una variable binaria que indica si el subconjunto S_j es una solución C. Las restricciones aseguran que cada elemento de U este cubierto al menos una vez por los subconjuntos seleccionados. El Set Covering Problem tiene diversas aplicaciones en la vida real, como la planificación de rutas de entrega, asignación de recursos, programación de horarios y diseno de redes de telecomunicaciones. Por ejemplo, en la programacion de horarios, los subconjuntos podrían representar clases y los elementos podrían representar intervalos de tiempo, donde el objetivo es programar las clases en horarios de manera que todos los intervalos estén cubiertos y se minimice el costo. El Set Covering Problem es un problema combinatorio importante con aplicaciones en diversas áreas. La optimización de la selección de sub-conjuntos para cubrir un conjunto de elementos de manera eficiente tiene implicaciones practicas significativas y ha sido abordada mediante diversas técnicas de optimización, incluidas las metaheurísticas como el algoritmo GAO que se presenta en este trabajo.

III-B. Funciones de Binarización

Un aspecto fundamental en el desarrollo de esta solución reside en la necesidad de binarizar las soluciones generadas por la metaheurística GAO. Dado que GAO opera en un espacio continuo, es imprescindible convertir las soluciones obtenidas en un formato discreto para poder compararlas con las soluciones del SCP. Esta etapa de binariazación resulta esencial para mapear las soluciones continúas generadas por GAO al espacio discreto del SCP y, así, evaluar su calidad y viabilidad en términos de la optimización del problema. Para llevar a cabo este proceso, una estrategia común implica la aplicación de una función de

binariazación a las soluciones continúas generadas por GAO. En este sentido, se suele emplear una función sigmoide u otras funciones de transformación que permiten asignar valores binarios $(0\ o\ 1)$ en función de ciertos umbrales. La principal característica que deben tener estas funciones es su rango es $y \in R$: $0 < y < 1$.

III-B1. Función de Binariazación con forma sigmoide:

A partir de una modificación a la tangente hiperbólica, se obtienen funciones de binariazación con forma sigmoide, las cuales se utilizan para binarizar las soluciones de la metaheurística GAO, estas funciones se observan en la figura 1.

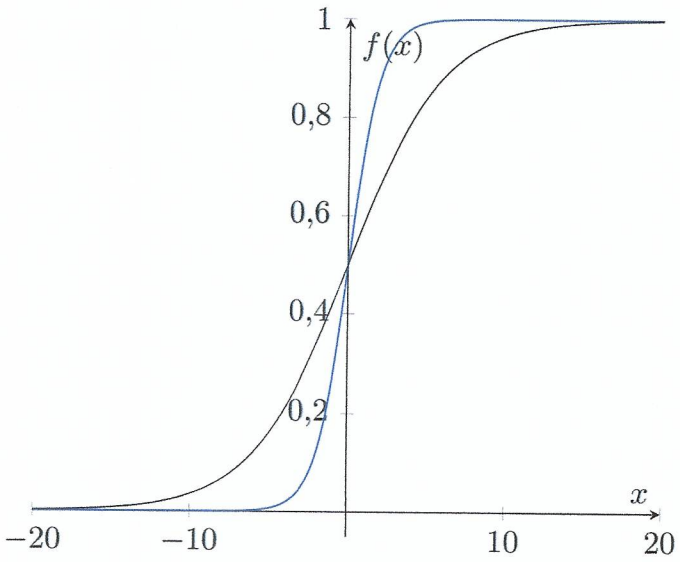

Figura 1 Función de Binarización con forma sigmoide

Las fórmulas de estas funciones son:

$$f(x) = \frac{1}{1+e^{-x}}$$

$$f(x) = \frac{1}{1+e^{-x/3}}$$

Para este experimento se usó la Binarización dada por la formula (4).

IV. GREEN ANACONDA OPTIMIZATION (GAO)

El GAO es un algoritmo metaheurístico poblacional basado en las anacondas verdes como miembros de la población. Cada anaconda representa una solución candidate con una posición en el espacio de búsqueda, representada por un vector. La posición inicial de cada anaconda se genera de manera aleatoria el inicio del algoritmo.

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N \times m} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,d} & \cdots & x_{1,m} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ x_{i,1} & \cdots & x_{i,d} & \cdots & x_{i,m} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,d} & \cdots & x_{N,m} \end{bmatrix}_{N \times m} \tag{5}$$

$$x_{i,d} = lb_d + r_{i,d} \cdot (ub_d - lb_d) \tag{6}$$

donde $i = 1, 2, \ldots, N$ y $d = 1, 2, \ldots, m$

Dado:

- X: como la matriz de población.
- $x_{i,d}$: la solución candidata y su dimensión en el espacio de búsqueda.

Correspondiendo a los valores sugeridos de cada anaconda verde para las variables de decisión, se puede evaluar la función objetivo del problema. Este conjunto de valores calculados para la función objetivo puede representarse desde un punto de vista matemático mediante un vector según la ecuación.

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N \times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N \times 1} \tag{7}$$

Dado:

- F: Vector de la función objetivo calculada.
- F_i: Función objetivo calculada con la i-ésima dimensión.

A partir de la comparación de los valores calculados para la función objetivo, el miembro correspondiente al mejor valor calculado para la función objetivo se identifica como el mejor miembro (la mejor solución candidata). Dado que, en cada iteración de GAO, las posiciones de las anacondas verdes y, por tanto, los valores de la función objetivo se actualizan, el mejor también debe actualizarse.

IV-A. Exploración

En el diseño del GAO de la posición de las anacondas verdes en el espacio de búsqueda se ha actualizado basándose en la simulación del comportamiento de la anaconda verde en dos fases con el fin de proporcionar exploración y

explotación en el proceso de búsqueda. La capacidad de exploración de GAO en la búsqueda global y la exploración precisa del espacio de resolución de problemas para evitar quedarse atascado en regiones locales optimas en el diseño del GAO, para cada anaconda verde, los miembros de la población que tienen un mejor valor de la función objetivo que ella se consideran como la especie hembra de las anacondas verdes. El conjunto de especies hembras candidatas para cada anaconda verde se determina mediante la ecuación (8).

$$CFL_i = \{x_{ki} : F_{ki} < F_i \text{ y } k_i \neq i\} \quad (8)$$

$$\text{donde } i = 1, 2, \ldots, N \; y \; ki \in \{1, 2, \ldots, N\}$$

Dado:

- CLF_i: es el conjunto de ubicaciones de hembras candidatas para la i-esima iteración
- k_i: es el numero de fila de la anaconda verde en la matriz de población

Para el modelo cuando mejor sea el valor de la función objetivo de un miembro, mayor la probabilidad de seleccionarlo. La función de probabilidad de selección a cada miembro de la matriz se calcula mediante la ecuación (9).

$$PC_j^i = \frac{CFF_j^i - CFF_{\text{máx}}^i}{\sum_{n=1}^{n_j} CFF_n^i - CFF_{\text{máx}}^i} \quad (9)$$

Dado:

- PC_j^i: como la probabilidad de selección de j para la solución i
- CFF^i: como el vector del conjunto de valores de la función objetivo para la i-ésima iteración
- CFF_j^i: es su j-ésimo valor
- $CFF_{máx}^i$: es el valor máximo, es el número de candidatos para la i-esima solución

En el diseño GAO, se selecciona aleatoriamente uno de los candidatos y se desplaza hacia la solución. Para proceso de selección, primero se calcula la función de probabilidad acumulada mediante la ecuación (10). A continuación, a partir de la comparación de la función de probabilidad acumulada con un numero aleatorio con una distribución normal en el intervalo de $[0, 1]$, se determina el valor de acuerdo con la siguiente ecuación (11).

$$C_j^i = PC_j^i + C_{j-1}^i \tag{10}$$

donde $i = 1, 2, \ldots, N$ y $j = 1, 2, \ldots, m$ y $C_0^i = 0$

$$SF^i = CFL_j^i : C_{j-1}^i < r_{i,j} < C_j^i \tag{11}$$

Dado:

- C_j^i: como la función de probabilidad acumulada de la j-esima solución para la i-esima anaconda verde.
- SF^i: como la solución/hembra seleccionada para la i-esima anaconda/solución.
- r: como un numero aleatorio con una distribución normal en el intervalo de $[0, 1]$

Una vez identificada la hembra seleccionada, basada en la simulación del movimiento de la anaconda verde hacia ella, se procede a calcular un numero aleatorio para cada individuo de la población de anacondas verdes. A partir de estos valores, se determina una nueva posición aleatoria en el espacio de búsqueda de la anaconda verde mediante la ecuación (12). Se evalúa entonces si esta nueva posición mejora el valor de la función objetivo. Si el valor de la función objetivo efectivamente mejora en esta nueva posición, tal como se establece en la ecuación (13), la posición correspondiente de la anaconda verde se actualiza con esta nueva posición. En caso contrario, la anaconda verde mantiene su posición actual en el espacio de búsqueda.

$$x_{i,d}^{P1} = x_{i,d} + r_{i,d}(SF_{i,d} - I_{i,d} \cdot x_{i,d}) \tag{12}$$

donde $i = 1, 2, \ldots, N$ y $d = 1, 2, \ldots, m$

$$X_i = \begin{cases} x_i^{P1} & \text{si } F_i^{P1} < F_i \\ x_i & \text{en otro caso} \end{cases} \tag{13}$$

Dado:

- x_i^{P1}: es la nueva posición sugerida de la i-esima anaconda verde basada en la primera fase de GAO
- $x_{i,d}^{P1}$: es su d-esima dimensión
- F_i^{P1}: es el valor de su función objetivo
- $r_{i,d}$: es un numero aleatorio con una distribución normal en el intervalo de $[0, 1]$

- $SF_{i,d}$: es la enésima dimensión de la hembra seleccionada para la i-esima anaconda verde
- $I_{i,d}$: son numeros aleatorios del conjunto $f1, 2g$
- N: es el número de anacondas verdes (soluciones)
- m: es el número de dimensiones (variables de decisión)

IV-B. Explotación

Durante la segunda fase del GAO, se procede a actualizar las posiciones de los miembros de la población mediante pequeños desplazamientos en el espacio de búsqueda. Esta fase refleja la capacidad de explotación del GAO al buscar soluciones optimas en las proximidades de las soluciones previamente descubiertas. En este proceso, se inicia generando una posición cercana a cada anaconda verde, empleando la ecuación (14). Posteriormente, se evalúa si esta nueva posición, como se describe en la ecuación (15), resulta en una mejora en el valor de la función objetivo. Si esta mejora es observable, la posición se actualiza según la nueva posición generada. En caso contrario, la posición original se mantiene sin cambios.

$$x_{i,d}^{P2} = x_{i,d} + (1 - 2r_{i,d})\frac{ub_d - lb_d}{t} \tag{14}$$

donde $i = 1, 2, \ldots, N$, $d = 1, 2, \ldots, m$ y $t = 1, 2, \ldots, T$

$$X_i = \begin{cases} x_i^{P2} & \text{si } F_i^{P2} < F_i \\ x_i & \text{en otro caso} \end{cases} \tag{15}$$

IV-C. Seudocódigo del algoritmo

A continuación, se presenta el seudocódigo del algoritmo GAO.

Algoritmo 1: Seudocódigo de GAO

```
Información de entrada del problema: variables, función objetivo y restricciones;
Establecer el tamaño de la población GAO (N) y las iteraciones (T);
Generar la matriz de población inicial al azar (6);
Evaluar la función objetivo de cada anaconda verde;
para t = 1 a T hacer
    para i = 1 a N hacer
        Fase 1: Exploración de la primera fase;
        Identificar candidatos (8);
        Calcular la función de concentración de las hembras candidatas (9);
        Calcular la función de probabilidad acumulada de las hembras candidatas (10);
        Determinar la hembra seleccionada (11);
        Calcular la nueva posición de la anaconda verde (i-ésimo miembro del GAO) (12);
        Actualizar el miembro i-ésimo del GAO (13);
        Fase 2: Exploración de la segunda fase;
        Calcular la nueva posición de la anaconda verde (i-ésimo miembro del GAO) (14);
        Actualizar el miembro i-ésimo del GAO (15);
    fin
    Guardar la mejor solución encontrada hasta el momento;
fin
Resultado: Mejor solución encontrada
```

V. GOOGLE COLAB

Google Colab, también conocido como Google Colaboratory, es un entorno de computación en la nube que facilita la ejecución de código Python en el navegador web. Esta plataforma permite a los usuarios realizar tareas que requieran de muchos recursos computacionales, como el entrenamiento de modelos de aprendizaje automático y el análisis de grandes conjuntos de datos, sin necesidad de instalar software o hardware adicional.

Google Colab nos entrega un entorno que se actualiza constantemente al momento de describir este informe el reléase disponible cuenta con la configuración descrita a continuación.

Hardware:

- CPU: Intel Xeon a 2.20 GHz o superior
- RAM: 13 GB