ALGORITMO DE FORRAJEO DE BACTERIAS APLICADO AL ALINEAMIENTO MÚLTIPLE DE SECUENCIAS GENÉTICAS

BACTERIAL FORAGING ALGORITHM APPLIED TO MULTIPLE ALIGNMENT OF GENETIC SEQUENCES

Rios Willars Ernesto, Silva Diaz Abigail

Resumen

Las bacterias son organismos unicelulares ubicuos que han evolucionado para adaptarse a diversos entornos, desempeñando roles cruciales en los ecosistemas. Aunque muchas son beneficiosas, algunas pueden causar enfermedades. En el contexto de la biotecnología y la inteligencia artificial, se han desarrollado algoritmos inspirados en las bacterias, como el algoritmo de forrajeo de bacterias (BFOA), que simula su comportamiento para resolver problemas complejos de optimización. Este algoritmo se basa en el quimiotaxis de la bacteria Escherichia coli, utilizando su capacidad para detectar nutrientes y evitar sustancias nocivas mediante desplazamientos aleatorios.

El BFOA es útil en áreas como la optimización de funciones no lineales, la detección de fallos en sistemas industriales y la predicción de estructuras de proteínas. En este proyecto, se busca implementar un BFOA para optimizar la ubicación y ruta de vehículos de una flota de distribución de paquetes, con el objetivo de reducir costos operativos y mejorar la eficiencia del servicio. El algoritmo se diseñará para emular la adaptación bacteriana, aplicando operaciones de mutación y ajuste en cada generación, con el fin de identificar la bacteria más fuerte, definida por su secuencia genética.

Palabras Clave: BFOA, Bacterias, Biotecnología, Mutación, Optimización, Quimiotaxis.

Abstract

Bacteria are ubiquitous single-celled organisms that have evolved to adapt to diverse environments, playing crucial roles in ecosystems. While many are beneficial, some can cause disease. In the context of biotechnology and artificial intelligence, bacteria-inspired algorithms have been developed, such as the Bacterial Foraging Algorithm (BFOA), which simulates their behavior to solve complex optimization problems. This algorithm is based on the chemotaxis of the Escherichia coli bacteria, using its ability to detect nutrients and avoid harmful substances through random displacements.

BFOA is useful in areas such as optimization of non-linear functions, fault detection in industrial systems, and prediction of protein structures. In this project, we seek to implement a BFOA to optimize the location and route of vehicles in a package delivery fleet, with the aim of reducing operating costs and improving service efficiency. The algorithm will be designed to emulate bacterial adaptation, applying mutation and adjustment operations in each generation, in order to identify the strongest bacteria, defined by its genetic sequence.

Keywords: BFOA, Bacteria, Biotechnology, Chemotaxis, Mutation, Optimization.

1. Introducción

Las bacterias son organismos microscópicos unicelulares que se encuentran en todas partes de nuestro entorno, desde el suelo hasta el cuerpo humano. Estos organismos han evolucionado a lo largo de millones de años para sobrevivir y adaptarse a diferentes entornos, gracias a sus complejos procesos biológicos y a su capacidad de comunicación y cooperación entre sí. Por su eficiencia en las tareas, desde la síntesis de proteínas hasta la detección de nutrientes, los algoritmos inspirados en las bacterias se han convertido en una herramienta valiosa para resolver problemas complejos en diferentes campos. Estos microorganismos son vitales para los ecosistemas del planeta. La mayoría de las bacterias que se encuentran en el organismo no producen ningún daño, al contrario, algunas son beneficiosas. También pueden tener distintas formas: esféricas, alargadas o espirales. Existen bacterias perjudiciales, llamadas patogénicas, las cuales causan enfermedades y las bacterias benéficas se encuentran frecuentemente en el sistema digestivo, en el intestino, tenemos bacterias que son muy necesarias para que nuestro cuerpo funcione correctamente[1,2,3].

Las bacterias son organismos muy eficientes en la realización de diversas tareas, gracias a sus complejos procesos biológicos y a su capacidad de adaptación a diferentes entornos. Por lo tanto, es posible desarrollar algoritmos inspirados en los mecanismos de las bacterias para resolver problemas complejos de diferentes áreas, tales como la optimización, el control y la detección de fallos en sistemas complejos, la inteligencia artificial y el aprendizaje automático, la síntesis de proteínas, entre otros. Además, se han utilizado en sistemas de detección de fallos en la industria, en la síntesis de proteínas, en la predicción de estructuras de proteínas, entre otros[1,2,3]. Este proyecto se basa en el desarrollo de un algoritmo de forrajeo de bacterias (BFOA por sus siglas en inglés) para resolver un problema de optimización específico en el problema de alineamiento múltiple de secuencias genéticas. El objetivo es optimizar la ubicación y la ruta de los vehículos de una flota de distribución de paquetes, con el fin de reducir los costos operativos y mejorar la eficiencia del servicio. Para lograr este objetivo, se diseñará un algoritmo de colonia de bacterias que emula el comportamiento de las bacterias en su capacidad de adaptarse y encontrar la mejor solución.

1.1 ¿Qué es un algoritmo?

Un algoritmo informático es un conjunto de instrucciones definidas, ordenadas y acotadas para resolver un problema, realizar un cálculo o desarrollar una tarea. Es decir, un algoritmo es un procedimiento paso a paso para conseguir un fin. A partir de un estado e información iniciales, se siguen una serie de pasos ordenados para llegar a la solución de una situación.

En programación, un algoritmo supone el paso previo a ponerse a escribir el código. Primero debemos encontrar la forma de obtener la solución al problema (definir el algoritmo informático), para luego, a través del código, poder indicarle a la máquina qué acciones queremos que lleve a cabo. De este modo, un programa informático no sería más que un conjunto de algoritmos ordenados y codificados en un lenguaje de programación para poder ser ejecutados en un ordenador[4].

1.2 Algoritmo BFOA

La optimización por enjambre de bacterias o bacterial foraging optimization algorithm (BFOA) representa una aproximación diferente a la búsqueda de valores óptimos en funciones no lineales, desarrollado por Passino (2010), se basa en el comportamiento quimiotáctico de la bacteria Escherichia Coli (E. Coli). Si bien, utilizar la quimiotaxis como modelo para optimización se propuso por primera vez en Bremermann (1974) y se ha utilizado en trabajos de Leiviskä (2006), el trabajo de Passino incluye algunas modificaciones como la reproducción y la dispersión de los agentes. La E.Coli es quizá el microorganismo más comprendido, ya que su comportamiento y estructura genética están bien estudiados.

Esta es una cápsula con órganos y flagelos, que utiliza para su locomoción; posee capacidad de reproducirse por división y puede intercambiar información genética con sus congéneres. Además, es capaz de detectar nutrientes y evitar sustancias nocivas, efectuando un tipo de búsqueda aleatoria, basado en dos estados de locomoción: el

desplazamiento o nado (swim) y el giro o tumbo (tumble). La decisión de permanecer en alguno de estos dos estados se basa en la concentración de nutrientes o sustancias nocivas en el medio. Este comportamiento se denomina quimiotaxis. A continuación, se describen los pasos de la optimización con el algoritmo BFOA[2].

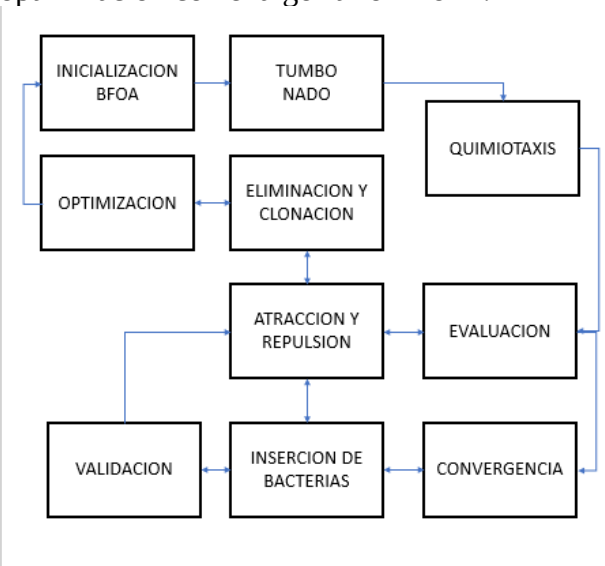

Figura 1. Proceso del algoritmo de forrajeo de bacterias.

1.3 Objetivos

El objetivo de este trabajo es diseñar y desarrollar un algoritmo computacional que pueda emular el comportamiento de las bacterias en su capacidad de adaptarse a diferentes entornos y de resolver problemas de manera eficiente.

En cada generación, se aplican operaciones de mutación y ajuste en las secuencias de bacterias para simular la evolución y adaptación de las bacterias. Se busca encontrar la bacteria más fuerte, donde la fuerza se define como el número de caracteres '-' en la secuencia de la bacteria, probando una serie de métodos para que el procesamiento y el calculo para que las evaluaciones las realice de una manera óptima.

2. Materiales y Métodos

En el desarrollo de este proyecto se utilizó el lenguaje de programación Java de la mano con NetBeans como el editor de lenguaje para su codificación. Con un equipo de cómputo con procesador AMD Ryzen 5 5500U que cuenta con 8 GB de memoria RAM

Se integran tres conjuntos de secuencias (Conjunto A, Conjunto B y Conjunto C) para la realización de los alineamientos. Dichas secuencias fueron obtenidas por consulta a la base de datos NCBI.

2.1 Repositorio GitHub código del algoritmo

https://github.com/AbigailDiaz09/SeminarioII

2.2 Proceso del algoritmo original

Este algoritmo de forrajeo bacterias se basa en el metabolismo de la bacteria estérica coli. Se tomo como base 5 clases, la clase que inicia todo el proceso consiste en un ciclo en el cual se realizan iteraciones, una clase llamada evaluador bloosom que se refiere a las matrices de evaluación, la clase bacteria que integra las funciones principales, la cual representa en si misma una matriz de alineamiento, una clase que hace las veces de lecto del archivo fasta, en donde se encuentran las secuencias de prueba, la clase quimiotaxis que consiste en el proceso de forrajeo de las bacterias y la realización de la quimiotaxis donde aquí mismo se realiza la evaluación.

1. Inicialización. El algoritmo comienza creando una población de bacterias, donde cada bacteria representa una posible solución al problema de alineación de secuencias. En este caso, cada bacteria tiene una serie de secuencias que serán alineadas introduciendo gaps en diferentes posiciones para maximizar el puntaje de BLOSUM (una matriz que mide la similitud entre pares de aminoácidos).
2. Proceso de Tumbo-Nado. El comportamiento de tumbo y nado representa las fases de exploración y explotación del algoritmo.
3. Evaluación. Cada bacteria evalúa su propio fitness basado en dos factores principales:

Puntaje blosum y la interacción quimiotáctica: Las bacterias interactúan entre sí, y hay una atracción (colaboración) o repulsión (competencia) que influye en su fitness.

4. Quimiotaxis. El proceso de quimiotaxis permite que las bacterias ajusten su posición en el espacio de soluciones. El algoritmo intenta encontrar

soluciones óptimas mediante el movimiento de las bacterias hacia áreas con mayor fitness.
5. Eliminación y Clonación: Este proceso de eliminación y clonación es un mecanismo de selección natural que asegura que las mejores soluciones sobrevivan y se exploren más a fondo en iteraciones futuras.
6. Inserción de Nuevas Bacterias Aleatorias. Para mantener la diversidad en la población, el algoritmo también introduce nuevas bacterias aleatorias en la población durante cada iteración. Esto asegura que el algoritmo no se quede atrapado en un óptimo local y continúe explorando el espacio de búsqueda.
7. Validación: Compara las secuencias alineadas con las secuencias originales para verificar que las secuencias modificadas siguen siendo coherentes con las secuencias originales.
8. Convergencia. El algoritmo continúa iterando hasta que se alcanza un número máximo de iteraciones o evaluaciones, o hasta que se cumpla un criterio de convergencia, como cuando el fitness deja de mejorar.

2.3 Mejoras al algoritmo

Matriz de bacterias: En lugar de String[] bacterias, ahora tenemos una matriz 2D String[][] bacterias, donde cada bacteria tiene un conjunto de secuencias.
1.Clase Bacteria: encapsula las secuencias, el puntaje BLOSUM, y el fitness. Esto mejora la claridad y permite manejar mejor la lógica relacionada a cada bacteria.
Puntaje BLOSUM: añadí un método en la clase para calcular el puntaje BLOSUM (aún es un valor aleatorio).
2. Interacción entre bacterias (Quimiotaxis): implementé la quimiotaxis, con reglas de atracción y repulsión entre bacterias para ajustar su fitness. Si el puntaje BLOSUM entre dos bacterias es similar, se atraen; si es muy diferente, se repelen.
3. Eliminación y Clonación: al final de cada iteración, eliminamos las bacterias con menor fitness. Las bacterias con mejor fitness se clonan y reemplazan a las eliminadas.
4. Diversidad de Población: introduje un mecanismo para insertar nuevas bacterias aleatorias en cada generación, manteniendo la diversidad de la población y evitando que el algoritmo se estanque en un óptimo local.
5. Balanceo de Secuencias: implementé el balanceo final para asegurar que las secuencias alineadas tengan la misma longitud después de cada iteración, similar al ajuste de gaps en las secuencias alineadas.

2.4 Experimento

Se realizan alineamientos utilizando el algoritmo de forrajeo de bacterias con los siguientes parámetros.

Parámetro	Valor
Attract	0.2
Repel	10
Número de bacterias	3
Ciclos	10

2.5 Alineamiento de pares de secuencias

Se realizan pruebas con el conjunto A para generar una tabla comparativa donde se describe:
• El Número de corrida: es un secuencial de 1 a 30 repeticiones.

2.6 Alineamiento Múltiple

Se realizan pruebas con el conjunto B y conjunto C para generar una tabla comparativa donde se describe:
• El Número de corrida: es un secuencial de 1 a 30 repeticiones.
• Núm. de Generaciones: es de 15 en cada una.
• Fitness.
• Número de funciones evaluadas (NFE).

2.7Tablas.

Tabla 1: Conjunto A de secuencias genéticas

Secuencia	Identificador	Descripción	Longitud
1	>NC_003310.1:c121870-119195	Monkeypox virus, complete genoma	1381 caracteres.
2	>NC_004718.3:21492-25259	SARS coronavirus Tor2, complete genoma	2664 caracteres.

Tabla 2: Conjunto B de secuencias genéticas

Secuencia	Identificador	Descripción	Longitud
1	>OQ319145.1	Helicobacter pylori isolate JHP_101 cytotoxin-associated protein A (cagA) gene, partial cds	258 caracteres.
2	>NZ_JAEUYS010000057.1	Escherichia coli strain 126 NODE_128_151_length_2297_cov_48.9748/1-2297, whole genome shotgun sequence	1874 caracteres.
3	>NZ_JASOFQ010000035.1	Enterococcus faecalis strain UMB9329NODE_35_length_2805_cov_16.548387, whole genome shotgun sequence	1496 caracteres.
4	>NZ_JAEMBN010000087.1	Porphyromonas gingivalis strain Kyudai-4 231, whole genome shotgun sequence	579 caracteres
5	>NZ_BQZG01000119.1	Staphylococcus aureus strain JARB-OU1147 sequence119, whole genome shotgun sequence	319 caracteres
6	>NZ_JUOA01001180.1	Bacteroides fragilis strain 915_BFRA3914_6177_403191, whole genome shotgun sequence	2172 caracteres

Tabla 3. Conjunto C de la proteína app.

Secuencia	Identificador	Descripción	Longitud
1	>NM_000484.4	Homo sapiens amyloid beta precursor protein (APP), transcript variant 1, mRNA	3583 caracteres
2	>NM_001198823.1	Mus musculus amyloid beta precursor protein (App), transcript variant 1, mRNA	3377 caracteres
3	>NM_001259804.2	Drosophila melanogaster approximated (app), transcript variant S, mRNA	3314 caracteres
4	>AF389401.1	Danio rerio amyloid precursor protein (app) mRNA, complete cds	3317 caracteres

3. Resultados y Discusión

Los resultados no fueron los esperados ya que las modificaciones que realice no fueron las óptimas y mejores para que las evaluaciones fueran las que se esperaban. Por lo que este código tal cual esta no cumple con lo que se tenía planteado cuando se hizo el planteamiento de este proyecto.

4. Conclusiones

Se necesitará trabajar y analizar muy detalladamente que es lo que pudo haber fallado en el desarrollo del código, y plantear mejor la estrategia de los métodos y cálculos para que nos entregue mejores resultados de optimización en el algoritmo. Todo esto debido a que los resultados de ejecución del algoritmo no fueron los esperados aun asi se haya modificado los códigos de distintas formas. Necesitare profundizar mas sobre este tema de investigación, ya que como no se pudo lograr los objetivos del propósito principal por la cual se hizo la realización de este proyecto se puede concluir que hace falta pulir el conocimiento e información.

Referencias

Artículos de revistas:

[1] Ernesto Ríos Willars, E. L. (Mayo-Agosto de 2015). Evaluación de una mejora al algoritmo genético clasico aplicado al alineamiento de secuencias. Obtenido de Komputer Sapiens.

[2] Ernesto Ríos Willars, E. L. (Noviembre de 2016). SCIELO. Obtenido de SCIELO: https://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1405-77432016000400479

Recursos de Internet:

[3] GENOME. (s.f.). GENOME. Obtenido de NATIONAL HUMAN GENOME RESEARCH INSTITUTE: https://www.genome.gov/es/genetics-glossary/Bacteria

[4] Maluenda, R. (21 de Enero de 2021). Profile. Obtenido de Profile: https://profile.es/blog/que-es-un-algoritmo-informatico/

[5] Noriega Gabriel Marcelo, R. J. (Marzo de 2010). SCIELO. Obtenido de SCIELO: https://ve.scielo.org/scielo.php?script=sci_arttext&pid=S1316-48212010000100005#:~:text=algoritmo%20basa%20su%20selecci%C3%B3n%20de,ser%20resuelto%20cuando%20se%20a%C3%B1adan

[6] Rasmussen, S. (4 de Junio de 2021). Sald Digital. Obtenido de Salud Digital: https://saluddigital.com/es/avance-de-la-ciencia/cientificos-utilizan-algoritmo-de-inteligencia-artificial-para-la-identificacion-de-bacterias/