UNIVERSIDAD DE CONCEPCION
Facultad de Ingeniería
Departamento de Ing. Civil Informática y
Ciencias de la Computación

Profesora Patrocinante:
María Angélica Pinninghoff J.

Comisión:
Ricardo Contreras A.
John Atkinson A.

Algoritmo de Conjugación Bacteriana con Enfrentamiento

Claudio Antonio Torres Méndez

Informe de Memoria de Título
Para optar al título de

Ingeniero Civil Informático

3 de octubre de 2016

Con cariño para todas las personas que me han ayudado, en especial para mi mamá y mi hermana que sin su apoyo incondicional nunca hubiera logrado esto.

Resumen

Las bacterias han existido en la tierra desde hace aproximadamente 3500 millones de años, sobrevivieron las cinco extinciones masivas de animales y logran vivir en las condiciones más extremas del planeta. Todo esto se debe a que han logrado adaptarse al ambiente.

Las buenas características que poseen las bacterias lograron que surgiera el Algoritmo de conjugación bacteriana, una nueva Metaheurística basada en la interacción de bacterias.

El objetivo de esta memoria es unir los siguientes modelos: Algoritmo de conjugación Bacteriana (ACB) y Algoritmo Metaheurístico Basado en Evolución horizontal de microorganismos. El resultado es el algoritmo de conjugación bacteriana con enfrentamiento (ACBCE), cuya efectividad fue probada en CVRP utilizando las instancias de Augerat.

Los resultados que obtuvo el ACBCE se compararon con los que obtuvieron los modelos de Algoritmos genéticos (AG), Algoritmo de Colonia Artificial de Abejas (ABC) y el Algoritmo de Conjugación Bacteriana con Control de la Población (ACBCP). Los resultados mostraron que los valores obtenidos por el ACBCE fueron cercanos a AG y ABC y superior en todas las instancias al ACBCP.

Índice

Índice de figuras

Índice de cuadros

1. Introducción

Existen problemas muy complejos, donde encontrar la solución exacta puede tomar mucho tiempo debido a su explosión combinatoria de su espacio de búsqueda. Por este motivo han surgido nuevos métodos que entregan soluciones cercanas al óptimo en un tiempo razonable. Algunos de estos nuevos métodos están basados en la naturaleza y se conocen como algoritmos bioinspirados, los cuales permiten resolver problemas de optimización en un tiempo razonable, gracias a complejos mecanismos de supervivencia presentes en la naturaleza que permiten a los seres vivos adaptarse a cambios en su entorno modificando su forma de vida o incluso su estructura. Los algoritmos bioinspirados han encontrado buenas soluciones en tiempos más acotados que los métodos exactos, como son los casos del Algoritmo de la colonia de hormigas (ACO) [2], Algoritmos Genéticos (AG) [6] y el Algoritmo de Colonia Artificial de Abejas (ABC) [7].

En esta memoria se intenta mejorar un algoritmo bioinspirado basado en la conjugación bacteriana (ACB) [3][4]. Para ello se hará uso de varias características del Algoritmo Metaheurístico Basado en Evolución Horizontal de Microorganismos [5]. El resultado es una fusión con el nombre de algoritmo de conjugación bacteriana con enfrentamiento (ACBCE).

El ACBCE se ha aplicado a un problema de optimización combinatoria de enrutamiento vehicular para evaluar su efectividad. Los resultados obtenidos por el ACBCE finalmente se compararán con los obtenidos con otros métodos [6][7].

1.1. Objetivo General

El objetivo de esta memoria es desarrollar un algoritmo basado en: Algoritmo de conjugación Bacteriana [3][4] y el Algoritmo Metaheurístico Basado en Evolución horizontal [5].

1.2. Objetivos específicos

- Estudiar las características de las bacterias y su comportamiento colaborativo.
- Definir los problemas de optimización combinatoria y sus características generales.

- Definir los modelos bioinspirados y estudiar las características de los algoritmos genéticos (AG) y de colonia artificial de abejas (ABC).
- Estudiar los algoritmos de conjugación bacteriana y Metaheurístico Basado en Evolución horizontal de microorganismos.
- Desarrollar un algoritmo basado en los anteriores y aplicarlo al problema de enrutamiento vehicular con capacidad constante.
- Evaluar y comparar ACBCE con Algoritmos genéticos, Algoritmo de colonia artificial de abejas y el Algoritmo de conjugación Bacteriana.

1.3. Organización de la memoria

La memoria está dividida en 5 capítulos: Introducción, Marco teórico, Algoritmo de conjugación bacteriana con enfrentamiento, Experimentos y Resultados y Conclusiones. A continuación se detalla cada uno de los capítulos:

- **Marco teórico:** se definen los conceptos y modelos biológicos en que está basado el algoritmo de conjugación bacteriana. Se explica el problema de Enrutamiento Vehicular (VRP). Finalmente se definen los algoritmos bioinspirados.
- **Algoritmo de conjugación bacteriana con enfrentamiento:** Se da a conocer el algoritmo desarrollado que se implementará para resolver el problema de enrutamiento vehicular con capacidad constante.
- **Experimentos y Resultados:** Se muestran los resultados obtenidos con ACBCE y se compara con AG, ACB y ACBCP.
- **Conclusiones:** se presentan las conclusiones obtenidas del trabajo realizado.

Finalmente en el anexo, se muestran los resultados que obtuvo el ACBCE y un manual de usuario del software.

2. Marco Teórico

2.1. Computación Evolutiva

La idea de que los organismos pueden evolucionar a través del tiempo y que un tipo de organismo da origen a otro es muy antigua [1]. Ya en la antigua Grecia poseían una concepción del mundo biológico en que el origen y

la transformación de especies eran producto de procesos naturales. Pasaron casi dos milenios para que se produjera un nuevo avance, Charles Darwin postula su teoría sobre el origen de las especies indicando que la evolución se origina a través de cambios aleatorios de características hereditarias, combinados con un proceso de selección natural. Tiempo después Johann Gregor Mendel realizó una serie de experimentos con guisantes durante una buena parte de su vida, enunciando a partir de ellos las leyes básicas que gobiernan la herencia [8]. De lo planteado por Darwin, Weismann, Mendel nace lo que actualmente se Conoce como Neo-Darwinismo el cual establece que toda la vida en el planeta puede ser explicada a través de sólo 3 procesos: Reproducción, Mutación y Competencia.

La computación evolutiva es una rama de la computación y la inteligencia artificial que emula a la evolución natural para la resolución de problemas. Su origen se remonta a la segunda mitad del siglo XX donde varios científicos de modo independiente comenzaron a estudiar los sistemas evolutivos pensando en que podrían aplicarlo como una herramienta para la resolución de problemas de optimización en ingeniería. La idea era lograr obtener una población de candidatos a ser solución de un problema conocido, utilizando operadores inspirados en la selección natural y la variación genética natural. A mediados de la década del sesenta se produjeron grandes avances que serían claves para esta área.

2.2. Microorganismos y Bacterias

Los microorganismos constituyen un grupo sumamente heterogéneo que incluyen las bacterias, hongos, virus, protozoos y algas microscópicas. Una característica común es su diminuto tamaño, ho que los hace invisibles para el ojo humano [9].

Las bacterias son microorganismos unicelulares sin núcleo (conocida como procariota). Son los organismos más simples y abundantes de la tierra, pueden vivir prácticamente cualquier sitio (tierra, agua, materia orgánica, plantas, animales) [11].

Las bacterias al ser células procariotas, no poseen membrana nuclear por lo que su ADN está libre en la célula bajo la forma de un único cromosoma. Las bacterias también poseen plásmidos que son moléculas de ADN circulares, que cumplen funciones particulares.

2.2.1. Estructura de una bacteria

Cada bacteria está compuesta de una envoltura (formada por la membrana plasmática, pared celular y la cápsula), un medio interno conocido como

citoplasma (solución acuosa contiene el material genético, ribosomas e inclusiones) y apéndices externos (flagelos y los Pilus). En la figura 1 se puede apreciar la estructura de la bacteria.

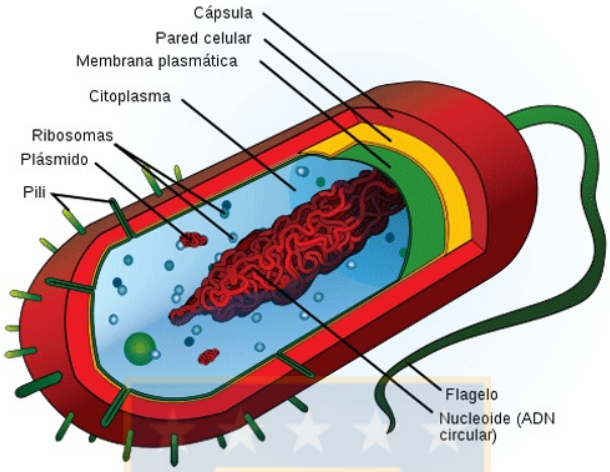

Figura 1: Estructura de una bacteria

En la superficie externa de la bacteria se encuentran los pilus (estructuras en forma de pelo) que se utilizan para transferir material genético. Un pilus sexual interconecta dos bacterias construyendo un puente entre ambos citoplasmas.

En el citoplasma se encuentra el material genético de la bacteria formado por una molécula circular de ADN de doble cadena enrollado sobre sí mismo. Algunas bacterias poseen material genético extra cromosómico, denominados plásmidos. Los plásmidos son elementos genéticos constituidos por secuencias de ADN cortas circulares que se replican en forma autónoma. En general los plásmidos no contienen información esencial, sino que confieren ventajas a la bacteria en condiciones de crecimiento determinada. Un ejemplo común de un plásmido es el que contiene genes de resistencia a un determinado antibiótico, es decir únicamente tendrá una ventaja en presencia de este.

2.2.2. Genética bacteriana

Las bacterias se reproducen por un proceso asexual, pero también poseen mecanismos para obtener la variabilidad genética necesaria para adaptarse al entorno. Existen dos formas de cambiar la dotación genética de una bacteria:

1. **Mutaciones:** Son cambios puntuales de una molécula de ADN. Una

mutación se produce por errores en el proceso de replicación del ADN. Estos errores ocurren con baja probabilidad.

Para las bacterias las mutaciones son una fuente importante de variabilidad, debido a que son poblaciones muy numerosas con tiempos de generación muy cortos.

2. **Recombinación:** Es un proceso que lleva a la obtención de un nuevo genotipo (información genética) a través del intercambio de material genético entre secuencias homólogas de ADN de dos orígenes diferentes (dos bacterias distintas).

   Se conocen dos formas de recombinación: la recombinación homóloga y la transposición. En la recombinación homóloga el fragmento da ADN aceptado es muy similar a una parte de los genes de la bacteria y, tras situarse al lado de este, se intercambian sus genes por un mecanismo de rotura, entrecruzamiento y reunión de sus cadenas de ADN. [10].

   En la figura 2 se puede observar cómo se produce la rotura, entrecruzamiento y reunión entre las cadenas de ADN.

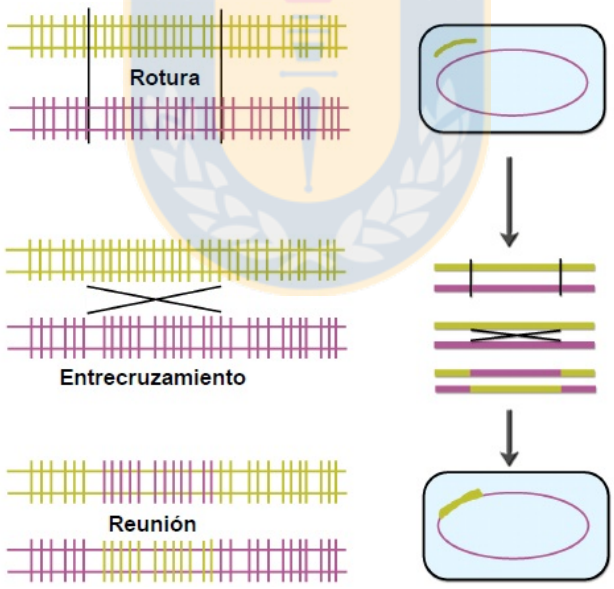

Figura 2: Mecanismo de recombinación homóloga

La transferencia previa a este tipo de recombinación puede ocurrir mediante tres vías:

a) **Transformación:** Penetra el ADN desnudo obtenido directamente desde el ambiente, a través de la pared de la célula receptora.

b) **Transducción:** Transfiere material genético mediante la intervención de un virus.

c) **Conjugación:** Transfiere plásmidos y en su caso genes cromosómicos arrastrados por plásmidos desde las bacterias donadoras a las receptoras.

2.2.3. Conjugación Bacteriana

La conjugación bacteriana es una forma de transferencia de material genético entre una célula bacteriana donadora y una receptora mediante contacto directo [12].

En la conjugación bacteriana, se puede transferir información de una bacteria a otra, en la forma de plásmidos y genomas. En este proceso ambas bacterias tienen contacto a través de un puente conocido como "Pili sexual".

La capacidad de donar o transferir material genético la proporciona un plásmido conjugativo denominado factor de fertilidad o plásmido sexual (F).

2.2.4. Plásmidos

Los plásmidos (definidos en la sección 2.2.1) son mucho más pequeños que el cromosoma. Las bacterias pueden poseer desde un par de genes de plásmido hasta más de cien. Algunos se replican en forma sincrónica de manera que cada bacteria replicada (mediante la reproducción asexual) tiene una sola copia del plásmido, pero existen bacterias que pueden contener múltiples copias (existen casos en que se ha detectado más de 50 estos plásmidos solo en una de ellas).

El plásmido F

El primer plásmido que se identificó fue el factor F de Escherichia coli (bacteria que se encuentra generalmente en los intestinos animales) [13]. Las bacterias que contienen este plásmido se denominan masculinas (Donadoras) o F+ y las que carecen de este se llaman femeninas (receptoras) o F-.

Por medio de los pilus (definidos en la sección 2.2.1) una bacteria F+ se adhiere a una F- y le transfiere el plásmido F a través de puentes. En la figura 3 se observa todo el proceso de transferencia del plásmido F.

El factor F transferido le confiere a las bacterias receptoras la capacidad de producir pilus y transferir plásmido F a otra que no lo posea, transformándose en F+.

Existen ocasiones que el plásmido F puede integrarse al cromosoma. Una bacteria que posea el plásmido F integrado en su cromosoma se conoce como

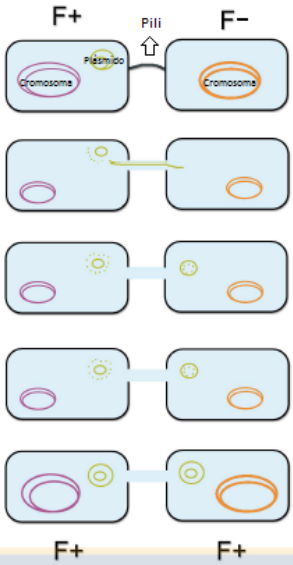

Figura 3: Conjugación entre una bacteria F+ y una F-

Hfr (alta frecuencia de recombinación), la cual puede transferir su cromosoma (replicado) o parte de él, generando una nueva combinación genética en la célula receptora.

El plásmido R

El plásmido R provee resistencia a la bacteria a los antibióticos, y puede transferirse entre bacterias por medio de conjugación. En un solo plásmido la bacteria puede recolectar hasta 10 genes de resistencia para distintos antibióticos y que pueden transferirse incluso a otro tipo de bacterias.

Antibiótico

Un antibiótico es una sustancia química que mata o impide el crecimiento de ciertas clases de microorganismos. Generalmente, los antibióticos son fármacos usados en el tratamiento de infecciones por bacterias.

2.3. Problemas de Optimización Combinatoria

La Optimización Combinatoria es una rama de la optimización, dedicada a abordar problemas de optimización donde el conjunto de posibles soluciones es discreto o se puede reducir a un conjunto discreto.

En los problemas de optimización combinatoria, se busca encontrar la mejor solución posible o solución óptima, aquella que minimiza una función de coste dada. Si un problema optimización combinatoria no es complejo, existen técnicas para resolverlo como el Branch and Bound (técnica de Ramificación y poda) , pero a medida que la complejidad del espacio de búsqueda aumenta, el coste de ejecución de dichos algoritmos puede aumentar de forma exponencial, haciendo la resolución de este problema por Branch and Bound prácticamente inviable.

Otra forma de abordar los problema optimización combinatoria consiste en buscar una solución subóptima, pero en un tiempo razonable. En algunos casos es posible encontrar incluso la solución óptima al problema. Este tipo de técnicas se pueden dividir en dos grandes grupos: Heurísticas y Metaheurísticas.

Problema de Enrutamiento vehicular (VRP)

El VRP es Problema clásico de optimización combinatoria donde se intenta encontrar un conjunto de rutas óptimas para que una flota de vehículos (localizados en un depósito) atienda a un conjunto clientes distribuidos geográficamente. El VRP es una variación del problema del vendedor viajero. La figura 4 muestra un ejemplo de enrutamiento donde cuatro vehículos cumplen con la demanda de todos los clientes.

Participantes del VRP

Vehículos: son los encargados de atender a los clientes, y deben comenzar y terminan en el depósito.

Depósito: Lugar geográfico donde comienza y termina cada vehículo.

Cliente: Persona que debe ser atendida por los vehículos.

Características generales del VRP

- La flota de vehículos es homogénea y está ubicada en un único depósito.
- La demanda del depósito es cero.

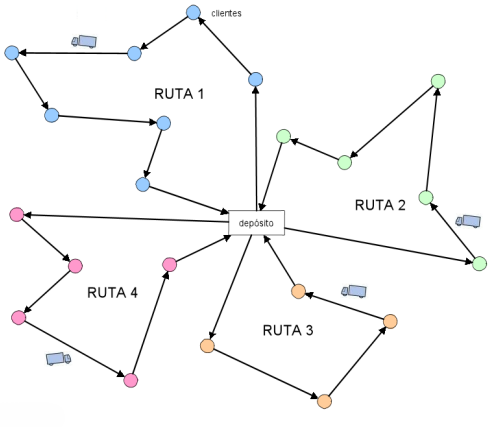

Figura 4: Ejemplo de un problema VRP

- Cada cliente tiene una demanda conocida.
- Un solo vehículo debe satisfacer la demanda de un cliente.
- La demanda de cada cliente es constante a lo largo del problema.
- Los clientes pueden ser visitados en cualquier momento.
- El orden de visita de los clientes no es importante.
- Los vehículos sólo entregan o reciben carga.
- Existe una variante del VRP llamado problema enrutamiento vehicular con capacidad constante (CVRP) donde cada vehículo tiene una capacidad limitada y constante [15].

2.4. Metaheurística

Las técnicas metaheurísticas son una clase de métodos aproximados que están diseñados para resolver problemas difíciles de optimización combinatoria, en los que los métodos heurísticos clásicos no son efectivos [14].

Las técnicas Metaheurísticas se basan en la aplicación de reglas relativamente sencillas. A diferencia de las heurísticas, tratan de huir de los óptimos locales [16].

Funcionamiento de las técnicas metaheurísticas:

1. Inicializar con una solución candidata.

2. Elegir soluciones similares simulares que satisfacen algunos criterios.

3. Repetir el proceso hasta cumplir alguna condición.

2.4.1. Computación bioinspirada

La computación bioinspirada se basa en emplear similitudes con sistemas pertenecientes a la naturaleza para diseñar métodos heurísticos no determinísticos de búsqueda, de aprendizaje o de imitación de comportamiento [17].

2.4.2. Algoritmos bioinspirados

Un algoritmo bioinspirado modela un fenómeno ya existente en la naturaleza, crean una metáfora biológica para resolver un problema. Un algoritmo bioinspirado es no determinístico y a menudo presentan, una estructura de múltiples agentes paralelos (entidad capaz de de percibir su entorno, procesar las percepciones y responder o actuar en este entorno).

2.4.3. Algoritmos Evolutivos

Los algoritmos evolutivos son un tipo de modelo bioinspirado que operan con una población de individuos, cada uno de los cuales representa una solución candidata a un problema. Esta población es sometida a ciertas transformaciones y luego a un proceso de selección donde se favorece a los mejores individuos. Cada ciclo de transformación y selección constituye una generación. Luego de un cierto número de generaciones se espera que el mejor individuo de la población represente la solución buscada.

2.4.4. Algoritmos Genéticos

Un Algoritmo Genético es un tipo de algoritmo evolutivo basado en la teoría de la evolución de Charles Darwin. Los algoritmos genéticos fueron creados en la década de los sesenta por John Holland y modelan el proceso de evolución como una sucesión de cambios genéticos [18].

Los genes se codifican en una cadena llamada cromosoma donde cada individuo de la población es representado como una posible solución. Cada una de estas es evaluada mediante una función. Si algún individuo alcanza el criterio de parada se detiene la ejecución y entrega la mejor o mejores soluciones. De no ser así se realiza una selección de individuos [19]. Luego se cruzan los individuos seleccionados para que sus genes se propaguen a su

descendencia. Puede ocurrir una mutación que modifica un gen para mantener la variabilidad dentro de la población. Finalmente se repite el ciclo hasta alcanzar algún criterio de parada [20].

2.4.5. Algoritmo de Colonia Artificial de Abejas

El algoritmo de colonia artificial de abejas o ABC (Artificial Bee Colony) [7] busca emular el comportamiento de las abejas en la búsqueda y explotación de fuentes alimentos. Para esto se define una colmena artificial y tres tipos de abejas (obreras, observadoras y exploradoras) para luego realizar la recolección de polen. Para esto, se distinguen los siguientes elementos:

- **Fuente de alimento:** Representa a una solución a un problema de optimización.
- **Abejas Obreras:** Están asociadas a una fuente de alimento. Cada abeja obrera lleva información sobre una fuente en particular, su distancia, ubicación y rentabilidad para compartirla con sus demás compañeras.
- **Abejas Exploradoras:** se encargan de buscar nuevas fuentes de alimento en el entorno que rodea a la colmena.
- **Abejas Observadoras:** Están en constante búsqueda de una fuente de alimento, a través de información compartida por las obreras o por otras exploradoras en el nido.

Un ABC típico funciona de la siguiente manera:

1. Se generan un conjunto de soluciones iniciales al problema (fuentes de alimento).
2. Se evalúa cada solución con respecto a la función de evaluación.
3. Se seleccionan las mejores soluciones que fueron evaluadas en el punto 2.
4. Se generan nuevas soluciones a partir de las mejores utilizando operadores de vecindad.
5. Se evalúan las nuevas soluciones y se escogen las que entrarán en la nueva iteración (vuelve al punto 3).

2.4.6. Algoritmo de conjugación Bacteriana

Un algoritmo de conjugación bacteriana (ACB) se basa en la transferencia genética realizada entre bacterias, a través del contacto físico entre bacterias donadoras y receptoras.

Debido a la naturaleza colaborativa no existe competencia por la supervivencia, por lo que cualquier bacteria donadora tiene las mismas posibilidades de transferir genes a una bacteria receptora.

Un ACB se compone de 7 operaciones:

1. **Inicialización:** Se genera la población inicial de bacterias.
2. **Evaluación:** Se evalúa cada una de las bacterias mediante una función.
3. **Clasificación:** Se clasifican las bacterias en donadoras y receptoras (dependiendo si poseen o no plásmido F).
4. **Selección:** Se seleccionan pares de bacterias (donadora y receptora).
5. **Conjugación:** Se realiza el intercambio genético entre las bacterias seleccionadas (una donadora y una receptora).
6. **Mutación:** Aleatoriamente se intercambia la posición del material genético de una bacteria.
7. **Curación:** Aleatoriamente se le quita a bacterias donadoras su plásmido F.

Algoritmo de conjugación Bacteriana con control de la población

Un Algoritmo de conjugación Bacteriana con control de la población (ACBCP) a diferencia del ACB no posee un tamaño fijo de población de bacterias sino que es un parámetro variable.

3. Algoritmo de conjugación bacteriana con enfrentamiento

En esta memoria se propone una adaptación al algoritmo de conjugación bacteriana que considera un proceso de enfrentamiento (presente en el Algoritmo Metaheurístico Basado en Evolución horizontal) y se denomina

ACBCE. A diferencia del ACB y ACBCP no existe un intercambio genético sino un traspaso de materia genético de una donadora a una receptora.

En el ACBCE incorpora dos nuevos actores: plásmidos R y antibióticos que está directamente relacionado el proceso de enfrentamiento.

3.1. Actores participantes del ACBCE

Bacteria: Posee su propio material genético. Se logran comunicar a través de estímulos para mantener el tamaño de la población. Las bacterias pueden poseer plásmidos F y R. La presencia del segundo vuelve a las bacterias inmune a la acción de ciertos antibióticos. Estos plásmidos pueden ser transferidos de una bacteria a otra conjuntamente.

A diferencia del modelo original no se produce un intercambio de genes sino un traspaso de material hacia la bacteria receptora. La bacteria donadora mantiene intacto su material genético.

1. **Bacteria Donadora:** Una bacteria donadora es aquella que posee el plásmido F. Este tipo de bacteria a su vez se clasifica en F+ y Hfr con la única diferencia de que el plásmido está unido al cromosoma en esta última y no puede transformar a una bacteria receptora en donadora. Si una bacteria donadora posee el plásmido R existe una probabilidad de transmitirlo a una bacteria receptora.

2. **Bacteria Receptora:** No posee el plásmido F y se les conoce como F-. Si una bacteria receptora posee el plásmido R, esta no puede transferirlo a menos que se convierta en donadora.

Para realizar traspaso de material genético es necesario una bacteria donadora y una receptora.

Antibiótico: Es una sustancia química artificial que es suministrada a la población de bacterias cada cierto tiempo. Si una bacteria entra en contacto directo con un antibiótico y no posee inmunidad para este tipo (un plásmido R adecuado), irremediablemente la bacteria muere. Luego del ataque, esta sustancia se vuelve inofensiva y las bacterias que no fueron afectadas entran en contacto directo con estos residuos (están esparcidos por toda la muestra). Los residuos les puede permitir a las bacterias generar inmunidad para este tipo de antibiótico.

3.2. Etapas ACBCE

En la figura 5 se muestra el ACBCE.

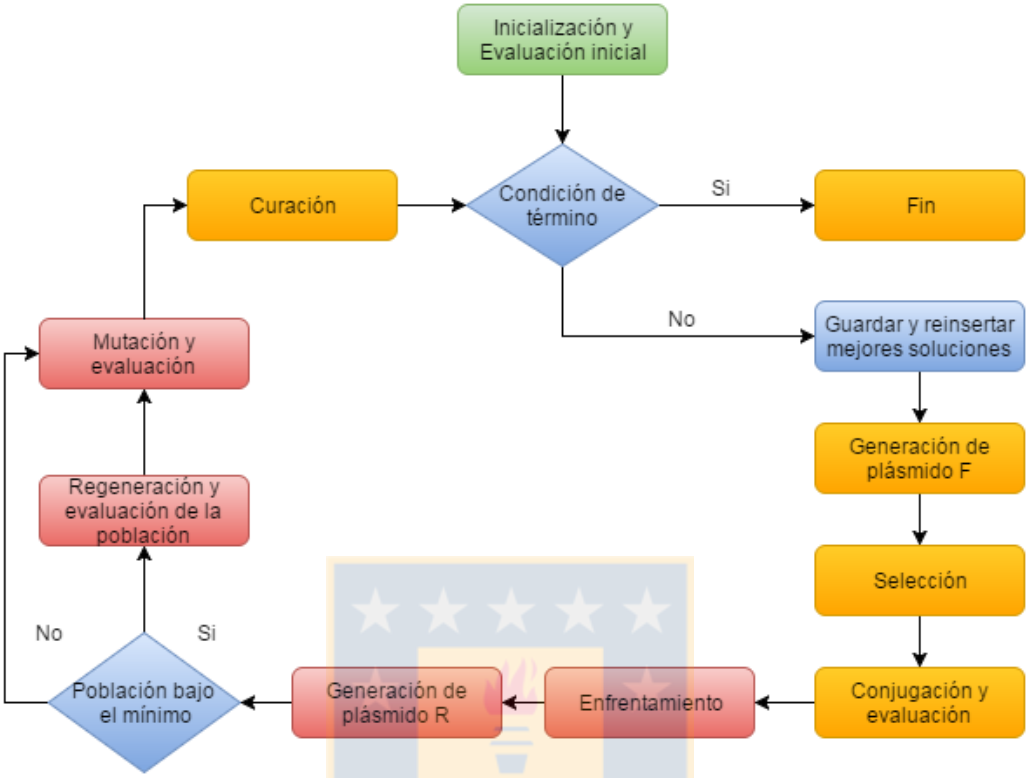

Figura 5: ACBCE

1. **Inicialización y Evaluación inicial:** Se genera la población inicial de bacterias de forma aleatoria. Todas las bacterias inicialmente son receptoras y no poseen ningún plásmido. Luego se evalúa mediante una función matemática cada una de las bacterias.

2. **Almacenar y reinsertar las mejores soluciones:** Se buscan las mejores bacterias de la población y se guardan. Luego en la siguiente generación son reinsertadas remplazando a las peores bacterias de la población. Estas son desechadas (eliminadas de la población).

3. **Clasificar plásmido F:** Se separan las bacterias en donadoras (F+ o Hfr) y receptoras (F-). Una bacteria cuya función de evaluación es menor debe tener una mayor probabilidad de ser una bacteria donadora. Para lograr esto, la siguiente fórmula define la probabilidad de que una bacteria B sea donadora.

$$P(B = donadora) = \frac{(MinF - F(B)) * 100 * K}{MinF - MaxF}$$

Donde:

- MaxF: es el fitness de la mejor solución encontrada en la población actual.
- MinF: es el fitness de la peor solución encontrada en la población actual.
- F(B): es el fitness de la bacteria B.
- K: es un factor (entre 0 y 1) que define que tan selectiva es la clasificación. Mientras más pequeño es menos probable que la bacteria puede convertirse en donadora.

Si K=1 entonces la probabilidad de ser donadora es de un 100 % a la mejor bacteria y 0 % a la peor. El resto de las soluciones se encuentra distribuido entre estos dos valores.

4. **Seleccionar:** Realiza el emparejamiento entre las bacterias donadoras y receptoras de forma aleatoria para el posterior traspaso genético.

5. **Conjugar y evaluar:** Se traspasan genes en las bacterias que fueron seleccionadas previamente. La conjugación puede ocurrir entre una célula F+ y F- o Hfr y F-. Sin embargo solo la primera opción convierte a la bacteria F- en F+, pero tiene una menor probabilidad de ocurrir.

   Para la conjugar una bacteria se utiliza un operador Order Crossover (OX) que modifica la bacteria receptora y mantiene intacta la bacteria donadora. El operador OX modificado funciona como sigue:

   Sean B_r la bacteria receptora y B_d la donadora.

$$B_r = 9\ 8\ 4\ 5\ 6\ 7\ 1\ 2\ 3\ 10$$
$$B_d = 8\ 7\ 1\ 2\ 3\ 10\ 9\ 5\ 4\ 6$$

   a) Se seleccionan los puntos de cruce aleatoriamente de B_r formando una subcadena (Sub-cadena 5 6 7 1)

   b) Se produce una nueva bacteria copiando la sub-cadena en las posiciones correspondientes a B_r

$$B'_r = \text{X X X } 5\ 6\ 7\ 1 \text{ X X X}$$

   c) Se Borran los valores que ya se encuentren en la sub-cadena de B'_d. La secuencia resultante contiene los valores faltantes.

$$B'_d = 8 \text{ X X } 2\ 3\ 10\ 9 \text{ X } 4 \text{ X}$$

d) Se colocan los valores en posiciones no conocidas de la bacteria de izquierda a derecha.

$$B'_r = 8\ 2\ 3\ 5\ 6\ 7\ 1\ 10\ 9\ 4$$

e) Finalmente B'_r reemplaza B_r en la población.

También se produce una transferencia de plásmidos R solo si la bacteria donadora lo posee. Finalmente se evalúa la población mediante la función de evaluación.

6. **Enfrentamiento**: Un antibiótico es elegido al azar y dependiendo de su tipo afecta a un cierto porcentaje de la población. Las bacterias son elegidas al azar hasta llegar al porcentaje de acción del antibiótico.

   Una bacteria que sea alcanzada por el antibiótico y no tenga inmunidad contra este enemigo en específico morirá, en caso contrario sobrevivirá. En cualquiera de los casos, una vez aplicado el antibiótico deja de hacer efecto, debido a que el tiempo de acción es igual a 1 iteración. Se definió esto para que para que solo exista un tipo de antibiótico atacante en cada iteración.

7. **Generar plásmido R:** Las bacterias que sobrevivieron al antibiótico pueden entrar en contacto con los residuos (restos inofensivos del antibiótico atacante) y dependiendo de la calidad de cada bacteria tienen una probabilidad de mutar y generar un plásmido R de inmunidad para este tipo de antibiótico atacante. El cálculo de esta probabilidad es igual a la descrita en la etapa 3.

$$P(B = GeneraplasmidoR) = \frac{(MinF - F(B)) * 100 * K}{MinF - MaxF}$$

8. **Regenerar y evaluar la población:** Si la población baja del mínimo de bacterias permitido (parámetro definido al comienzo) se regenera hasta alcanzar el tamaño de la población inicial. Las bacterias son regeneradas de la misma forma que en fase de Inicialización. Finalmente se realiza la evaluación para los nuevos individuos.

9. **Mutación:** La posición de dos genes de la bacteria (elegidos aleatoriamente) se intercambian para así mantener la variabilidad de la población. La mutación ocurre con cierta probabilidad.

10. **Curación:** Dependiendo de la calidad de una bacteria existe la probabilidad de perder un plásmido R elegido de forma aleatoria. Mientras

peor sea su fitness mayor es la probabilidad de perder un plásmido R. De lo contrario todas las bacterias podrían volverse inmunes a los antibióticos. También es necesario regular el plásmido F con el fin de que la mayor parte de las bacterias puedan participar en la conjugación es necesario regular las bacterias donadoras. Las bacterias pueden perder su plásmido F con cierta probabilidad que depende de la calidad de su fitness. Esto se repite hasta que estén reguladas. El cálculo de esta probabilidad es igual a la descrita en la etapa 3.

$$P(B = PerderPlasmido) = 100 - \frac{(MinF - F(B)) * 100 * K}{MinF - MaxF}$$

3.3. Analogía biológica computacional del ACBCE

Se ha simplificado la anatomía bacteriana (vista en la sección 2.2.1, no se incluyen ribosomas y solo dos tipos de plásmido) para su adaptación al modelo. En la figura 6 se muestra la bacteria simplificada.

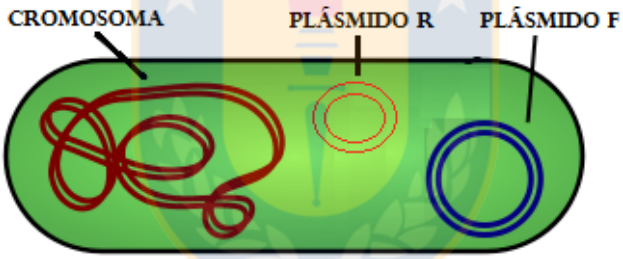

Figura 6: Bacteria simplificada

A continuación se detalla cada parte de la bacteria y su respectiva analogía computacional:

1. **Cromosoma bacteriano:** es codificado por un arreglo de datos. Donde cada posición representa un gen (ver figura 7).

2. **Plásmido F:** La presencia o ausencia del plásmido F en las bacterias nos permite clasificarlas en donadoras (conocidas como F+ y Hfr) y receptoras (F-) respectivamente (ver figura 8).

   Una bacteria donadora le traspasa parte de su material genético a la receptora. Dependiendo del tipo de bacteria donadora, la receptora se puede convertir en donadora. Solo las F+ pueden convertir a la bacteria receptora en donadora. Esto tiene su explicación en la parte biológica, pues el plásmido en la Hfr está integrado en el cromosoma.

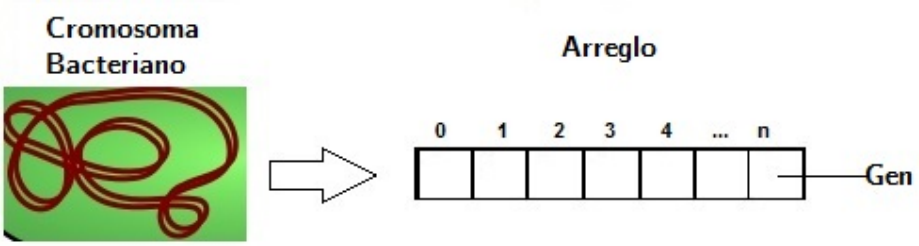

Figura 7: Condificación del cromosoma computacionalmente.

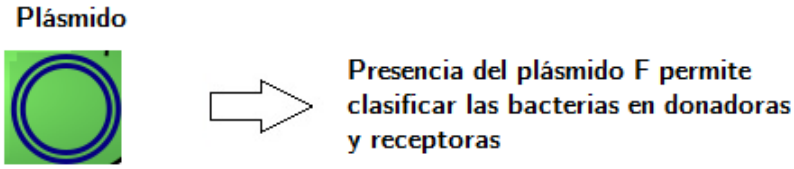

Figura 8: Representación del plásmido

3. **Plásmido R:** Un plásmido R entrega inmunidad a cierto antibiótico. Para que una bacteria sea inmune a cualquier antibiótico debe poseer todos los plásmidos R que existen.

   Los plásmidos R que posee una bacteria son representados por un arreglo de tamaño n (Siendo n el número de antibióticos distintos que existen) Cada antibiótico tiene una posición definida en el arreglo y si su identificador es distinto al número del antibiótico en el arreglo quiere decir que no posee plásmido de inmunidad para este antibiótico. En la figura 9 se muestra la analogía del plásmido R.

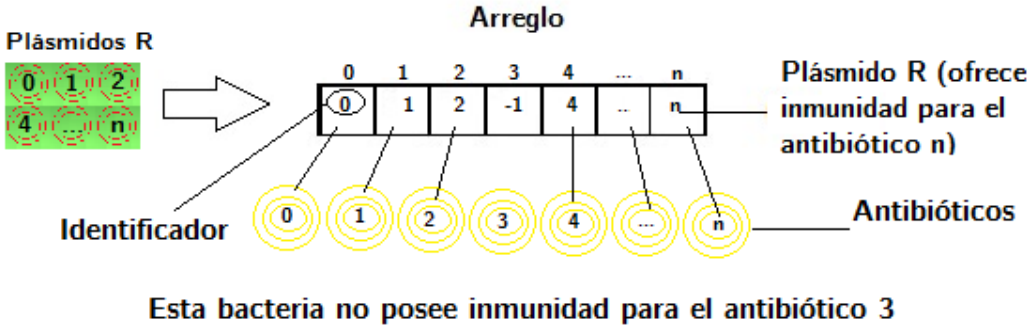

Figura 9: Analogía biológica computacional para los plásmidos R

Antibiótico R: en la parte computacional es identificado por un número. Cada número es un tipo distinto de antibiótico y dependiendo de este

afecta a un tamaño distinto de la población. En la figura 10 se observa esta equivalencia.

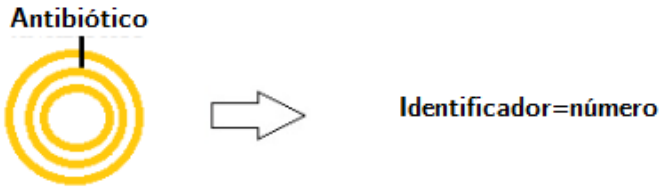

Figura 10: Analogía biológica computacional antibiótico

3.4. ACBCE orientado a CVRP

Para evaluar la efectividad del algoritmo se utiliza el problema de optimización combinatorio de enrutamiento vehicular con capacidad constante que fue definido en la sección 2.3. En el CVRP a diferencia del VRP cada vehículo posee una capacidad constante.

3.4.1. Cromosoma de una bacteria de ACBCE para CVRP

El cromosoma circular en una bacteria es representado mediante un arreglo de tamaño fijo (determinado por el número de clientes que posee el problema) y está formado por las rutas que recorren los vehículos. En la figura 11 se muestra la analogía entre el enrutamiento óptimo para una instancia tipo A de Augerat y su representación en un arreglo. Cabe mencionar que una ruta es un subconjunto de clientes que son visitados en el mismo orden que aparecen (Cada vehículo realiza una ruta). El tamaño de la ruta depende de la demanda del cliente.

Cada vehículo tiene una capacidad de carga y una vez satisfecha ésta, tiene que volver al depósito. Para crear las rutas se suma la demanda de cada cliente mientras no sobrepase la capacidad del vehículo. Se realiza esto hasta incluir a todos los clientes en alguna ruta.

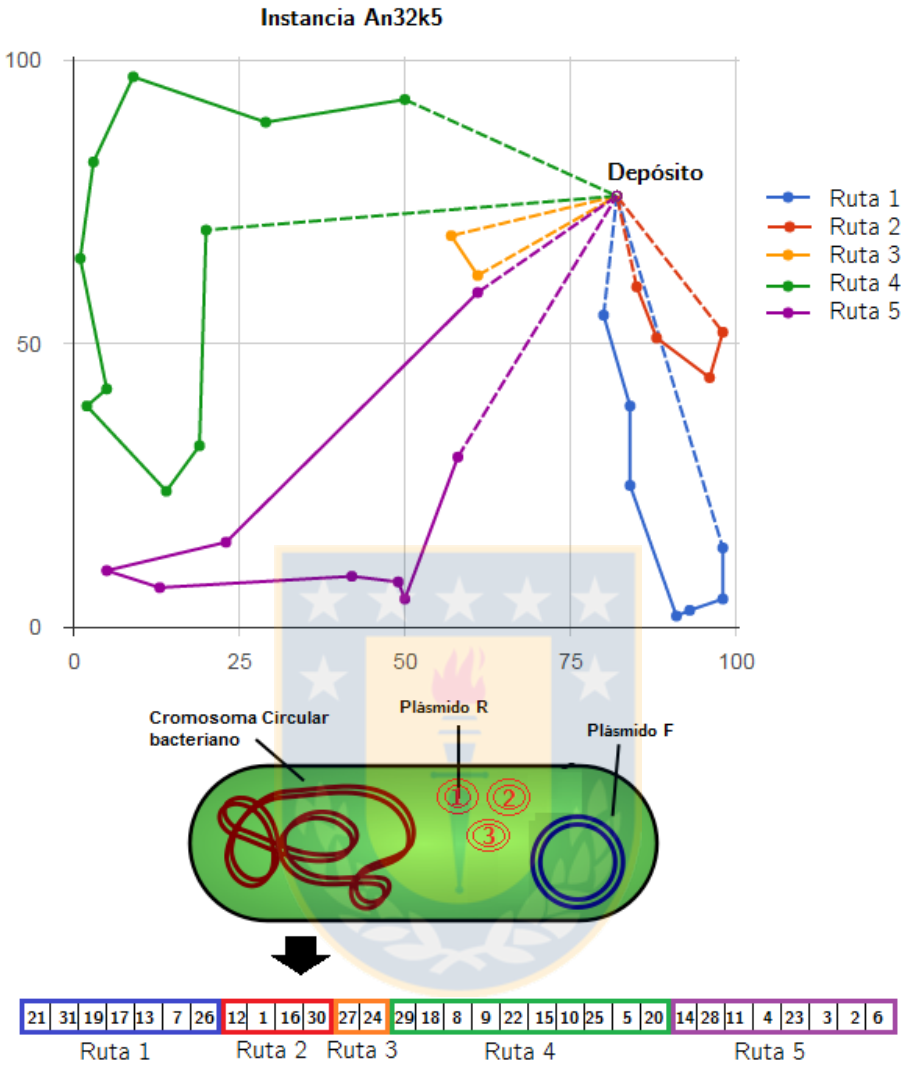

Figura 11: Representación de la bacteria

4. Experimentos y Resultados

Con el fin de medir la calidad del ACBCE se realizan experimentos con instancias de Augerat. Se replicó los mismos experimentos realizados con otros modelos [5][7] para logar una comparación válida. Se espera mejorar los resultados obtenidos por el ACB con control de la población que utiliza como base el ACBCE.

4.1. Datos Utilizados

Se utilizan las instancias de Augerat, las cuales consisten en distintos problemas de enrutamiento vehicular para las que su óptimo se encuentra calculado. Existen tres tipos de estas instancias A, B, P. En este caso se hará uso solo de los grupos A y B.

- **Grupo A:** Consta de 27 instancias que varían desde los 31 a los 79 clientes sin incluir el depósito. Estas instancias se caracterizan porque los clientes se encuentran dispersos alrededor del depósito. El cual puede situarse en cualquier parte del sector geográfico que abarca el problema.
- **Grupo B:** Consta de 23 instancias que varían desde los 30 a los 78 clientes. Se caracterizan por tener a los clientes agrupados en distintas zonas alrededor del depósito, el cual al igual que las instancias del grupo A, puede situarse geográficamente en cualquier parte del sector que abarca el problema.

4.2. Ajuste de parámetros

Para la evaluación del ACBCE es necesario realizar un ajuste de parámetros para lograr obtener el mejor resultado posible.

Luego de las pruebas realizadas se llegó a los valores ideales para cada parámetro. En el cuadro 1 se muestran los parámetros finales con los que se analizara el rendimiento del método.

Nombre del Parámetro	Valor
Número de iteraciones	1.000.000
Tamaño de la población	100
Número de antibióticos	10
Método de selección	Aleatoria
Método de conjugación	OX modificado
Factor de conjugación	75
Porcentaje mutación	10
Porcentaje de acción del antibiótico	variable

Cuadro 1: Parámetros finales

4.3. Experimentos realizados

Las pruebas realizadas al ACBCE abarcan 27 instancias Tipo A y 23 Tipo B de Augerat. Para cada una de las instancias se realizaron 20 ejecuciones con los parámetros finales.

La medida que se usó para evaluar los resultados es el porcentaje de desviación relativa o error porcentual con respecto a la solución óptima, RPD (Relative Percentage Desviation), la cual permite tener una noción concreta de la calidad de los resultados como:

$$RDP(\,\%) = \frac{C_{heur} - C_{opt}}{C_{opt}} * 100$$

Donde:
C_{heur}: costo de transporte (distancia total recorrida)
C_{opt}:valor óptimo del problema en cuestión.

En total se realizaron 1000 ejecuciones (50 instancias x 20 ejecuciones por cada una) con la finalidad de obtener el mejor valor posible para las instancias de Augerat.

Resultados Obtenidos

Se observa en las figuras 12 y 13 que los resultados del ACBCE son cercanos al óptimo y en algunos casos no se logra distinguir entre las líneas de cada uno. También es notorio que a medida que aumentan el número de clientes el ACBCE se aleja del óptimo.

El VRP es un problema de optimización combinatoria, o sea cuando aumentas el número los clientes el número de soluciones aumenta de forma exponencial. Esto puede desencadenar que el ACBCE pueda estar cayendo en un óptimo local y no pueda salir de allí (al cada vez tener una mayor cantidad de posibles soluciones). También es notorio en los gráficos que el ACBCE tiene un mejor desempeño cuando los clientes estas agrupados en zonas (instancias Augerat tipo B).

Cabe mencionar que los resultados del ACBCE mostrados en los gráficos corresponden a la mejor solución encontrada para cada instancia de Augerat. Estas mismas soluciones se utilizaron para el cálculo de los RPDs.

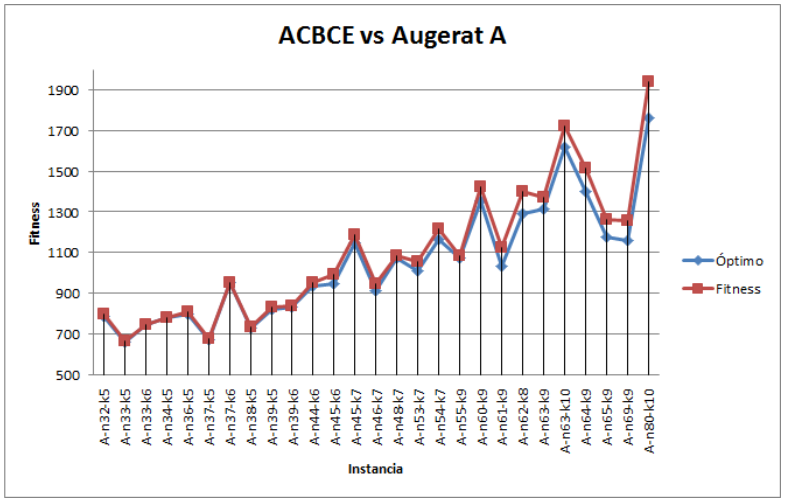

Figura 12: ACBCE vs Augerat A

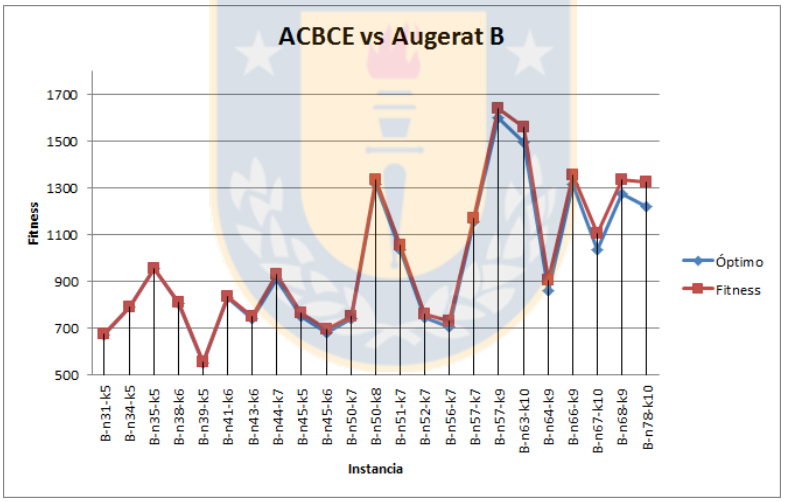

Figura 13: ACBCE vs Augerat B

Las figuras 14 y 15 muestran de manera más notoria que al aumentar el número de clientes los resultados se alejan más del óptimo.

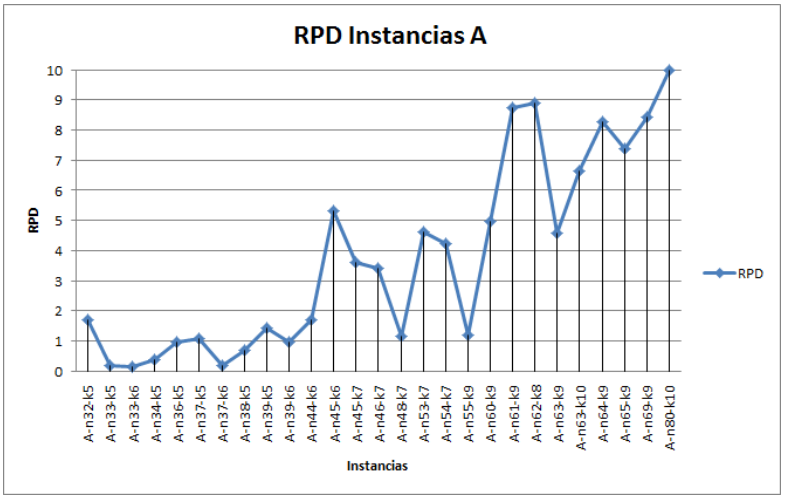

Figura 14: RDP calculado para cada instancia de Augerat tipo A

Figura 15: RDP calculado para cada instancia de Augerat tipo B

En la figura 16 se muestra lo que sucede con el tiempo al ir aumentando el número de clientes. Se observa que el tiempo es prácticamente lineal al aumentar el número de clientes y no se ve afectado por el número de soluciones aumenta exponencialmente.

Cabe mencionar que para el ACB se realizaron solo 300.000 iteraciones por cada instancia ejecutada, mientras que en el ACBCE se realizaron 1.000.000 de iteraciones por cada una de las instancias de Augerat.

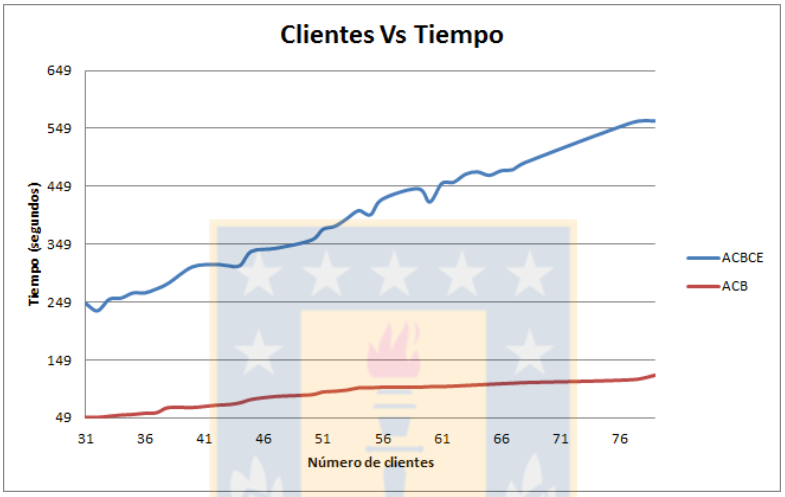

Figura 16: Tiempo según el número de clientes a visitar.

Comparación con otros modelos

Para medir la calidad del ACBCE se comparó con otros modelos bioinspirados tales como: Algoritmos Genéticos, Algoritmo de Colonia Artificial de Abejas y ACBCP. Tal como se muestra en las figuras 17 y 18. Se observa que no solo ACBCE empeora al aumentar el número de clientes, sino que es generalizado en todos los modelos.

Los resultados obtenidos por ACBCE muestran que mejora los de ACBCP. También se observa que los resultados son cercanos a los de algoritmos genéticos y colonia artificial de Abejas, pero esto modelos obtienen mejores resultados al ir aumentado la cantidad de clientes. Cabe mencionar que estos dos modelos cuentan con varios enrutamientos que incluyen un vehículo extra que el óptimo no posee. De esta manera logra resultados más bajos que el óptimo.

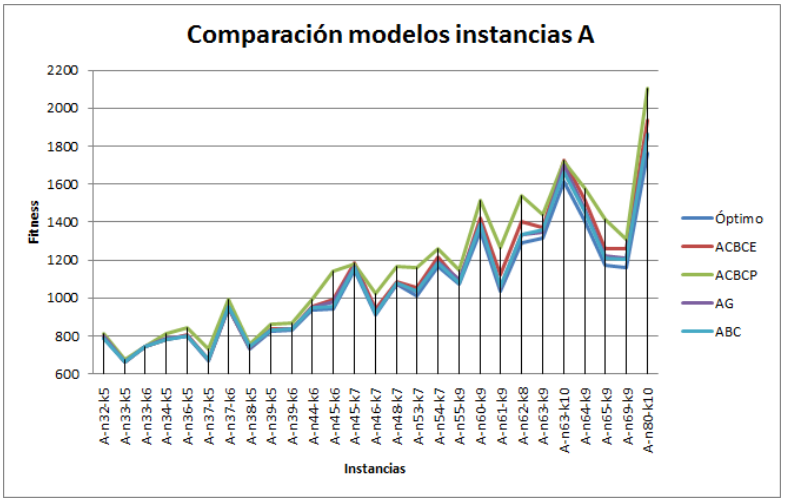

Figura 17: Comparación de modelos instancias A

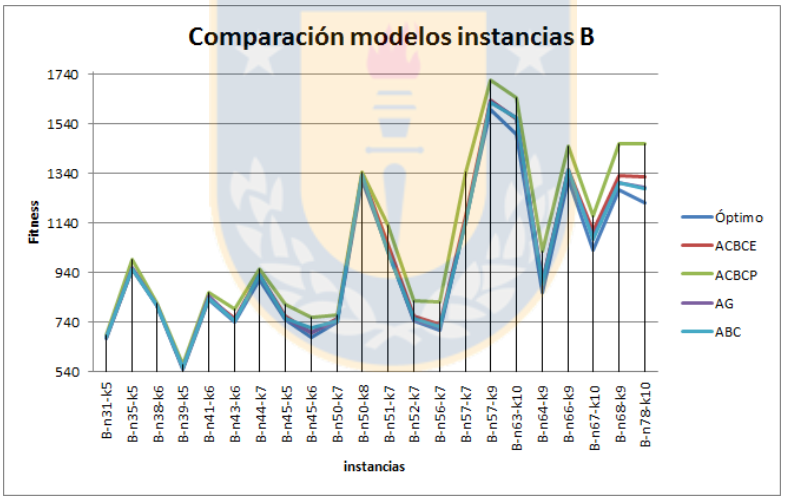

Figura 18: Comparación de modelos instancias B

Las figuras 19 y 20 muestran una visión completa del comportamiento en las instancias Tipo A y B. Se aprecia que existe una clara tendencia donde la tasa de error porcentual crece a la medida que aumenta la cantidad de clientes. Esto se hace más notorio a partir de los 60 clientes.

En las figuras 19 y 20 se comparan los RPDs de los diferentes modelos.

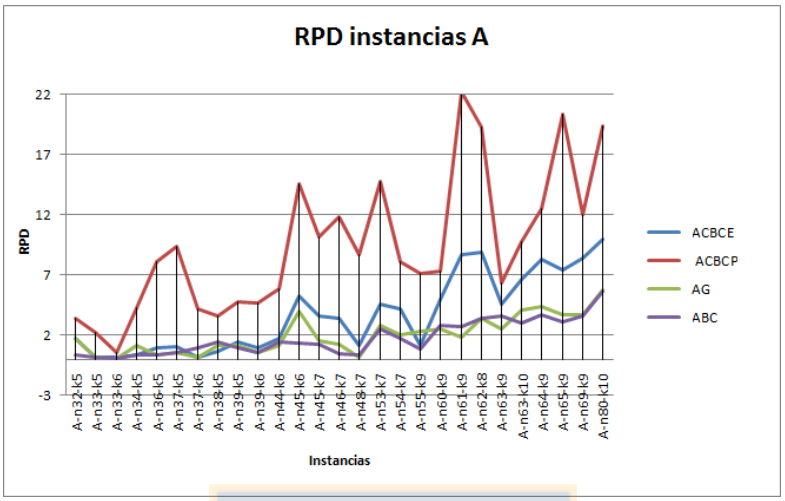

Figura 19: RDPs para la instancias A

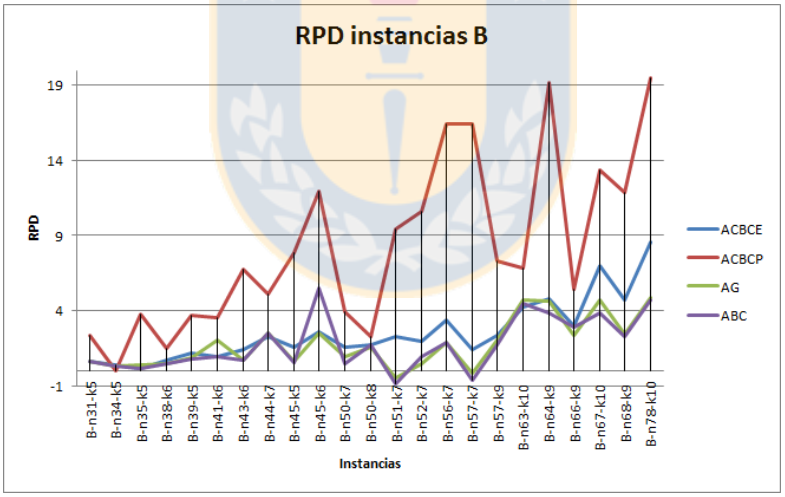

Figura 20: RDPs para la instancias B

El cuadro 2 ejemplifica de mejor manera que el ACBCE empeora en mayor medida que otros modelos al aumentar el número de clientes.

	Número de Clientes < 60				Número de Clientes ≥ 60			
	Instancias	ACBCE	AG	ABC	Instancias	ACBCE	AG	ABC
Grupo A	18	1.83	1.23	0.88	9	7.86	3.54	3.50
Grupo B	17	1.54	1,00	1,02	6	5.37	3.96	3.66

Cuadro 2: RPD promedio agrupado por número de clientes

Finalmente luego de comparar los resultados obtenidos por el ACBCE con el ACBCP se observa que los cambios realizados cumplieron su objetivo mejorando casi en un 60 % su rendimiento.

5. Conclusiones

En esta memoria se propuso una adaptación al algoritmo de conjugación bacteriana que considera un proceso de enfrentamiento y se le denominó ACBCE. A diferencia del ACB y ACBCP no existe un intercambio genético sino un traspaso de materia genético de una donadora a una receptora.

En el ACBCE incorporó dos nuevos actores: Plásmidos R y antibióticos que está directamente relacionado el proceso de enfrentamiento

Para medir el comportamiento de este nuevo modelo se le aplicó al problema del CVRP. utilizando las instancias de Augerat, las cuales consisten en distintos problemas CVRP para las que su óptimo se encuentra calculado.

Los resultados obtenidos muestran que el ACBCE es superior al ACBCP en casi un 60 %. El ACBCE Tiene un mejor rendimiento cuando los clientes se encuentran agrupados en zonas localizadas y/o cuando la cantidad de clientes no supera los 60.

Las comparaciones realizadas con AG y ABC muestran que los valores obtenidos por el ACBCE son cercanos a estos modelos, pero a partir de las instancias de más de 60 clientes es superado indiscutiblemente por AG y ABC. Cabe mencionar también que estos modelos cuentan con varios ruteamientos que incluyen más vehículos que en el utilizado en el óptimo.

Finalmente el ACBCE mejora los resultados del ACBCP, pero todavía le falta para estar a la par de los modelos de AG y ABC.

Referencias

[1] Curtis, Barnes, Schnek y Massarini, *Curtis Biología*, séptima edición en español, Sección 4 Evolución. pág. 332-350, 2008.

[2] Benjamín Arenas F, Alejandro Pavez y Rodrigo Vidal, *Algoritmo ACO aplicado al TSP: Resumen de una experiencia práctica*, Universidad Técnica Federico Santa María, Departamento de Informática.

[3] Hernán Fernández Carné *Metaheurística basada en la conjugación bacteriana*, Universidad de Concepción, 2014.

[4] Victor Ortiz Baltierra *Memoria de título: Metaheurística basada en el comportamiento colaborativo de microorganismos celulares, para la solución de un problemas de optimización combinatoria*, Universidad de Concepción, 2014.

[5] Rodrigo Neira Jara, *Algoritmo Metaheurístico basado en evolución horizontal de microorganismos*, Universidad de Concepción, 2014.

[6] Carlos Pantoja, *Resolución del Problema de Enrutamiento Vehicular con Capacidad Homogénea utilizando Algoritmos Genéticos*, Universidad de Concepción, Abril 2015.

[7] Germán Álvarez Sánchez, *Resolución del Problema de Enrutamiento Vehicular con Capacidad Homogénea Utilizando un Algoritmo de Colonia Artificial de Abejas*, Universidad de Concepción, 2013.

[8] Erik SCHWARZBACH, Petr SMÝKAL2, Ondřej DOSTÁL, Michaela JARKOVSKÁ y Simona VALOVÁ, *Gregor J. Mendel – Genetics Founding Father*, Miroslav, Czech Republic; Faculty of Science, Palacký University Olomouc, Olomouc, Czech Republic; Mendel Museum and Faculty of Science, Masaryk University, Brno, Czech Republic, pág. 43-51, 2014.

[9] Noé Manuel Montaño Arias, Ana Lidia Sandoval Pérez, Sara Lucía Camargo Ricalde y Juan Manuel Sánchez Yáñe, *Los microorganismos: pequeños gigantes*, Elementos, Pág 15–23, 2010.

[10] L. Betancor, M. Gadea y K. Flores, *Genética bacteriana*, Facultad de Medicina del Uruguay, 10 de Febrero de 2009.

[11] Curtis, Barnes, Schnek y Massarini, *Curtis Biología*, séptima edición en español, Sección 5 La diversidad de la vida. pág. 455-478, 2008.

[12] Gerard Tortora, Berdell Funke y Christene Case, *Intruducción a la microbiología*, novena edición en español, capítulo 1 El mundo microbiano y usted. pág 1-25, 2007.

[13] NEVILLE FIRTH, KARIN IPPEN-IHLER y RONALD A. SKURRAY, *Structure and Function of the F Factor and Mechanism of Conjugation*, SECTION C Gene Transfer: Conjugation.

[14] Ibrahim H. Osman y , John Paul Kelly *Meta-Heuristics: Theory and Applications*, Capítulo2. Heurística y metaheuristica, Boston USA Ed. Kluwer Academic, 1996.

[15] Juan Pablo Orrego Cardozo, *Solución al problema de ruteo de vehículos con capacidad limitada "CVRP" a través de la heurística de barrido y la implementación del algoritmo genético de CHU-BEASLEY*, Universidad Tecnológica de Pereira, Diciembre de 2013.

[16] Álvaro García Sánchez, *Técnicas metaheurísticas* , Universidad interino.

[17] Alfonso Mateos Andaluz, *Algoritmos Evolutivos y Algoritmos Genéticos* , Universidad Calos III de Madrid.

[18] Marcos Gestal, Daniel Rivero, Juan Ramón Rabuñal, Julián Dorado y Alejandro Pazos, *Introducción a los Algoritmos Genéticos y la Programación Genética*, Universidade da Coruña, 2010.

[19] Victor Arroyo Apablaza, *Modelo de un algoritmo genético con selección discriminatoria de individuos bajo un esquema de ponderación de probabilidades de mutación*, Pág 25-26, Universidad Católica San Pablo, Diciembre 2013.

[20] Belén Melián Batista, José A. Moreno Pérez y J. Marcos Moreno Vega, *Algoritmos Genéticos. Una visión práctica*, NÚNEROS Volumen 71, Pág 29–47 Agosto de 2009.

[21] Ms. Monica Sehrawat y Mr. Sukhvir Singh, *Modified Order Crossover (OX) Operator*, International Journal on Computer Science and Engineering, Vol. 3 No, 5 May 2011 .

ANEXOS

1. Trabajos Previos

1.1. Algoritmo de conjugación Bacteriana (ACB)

Este algoritmo está basado en la conjugación bacteriana (explicada en detalle en la sección 2.2.3) donde mediante contacto físico entre dos células bacterianas (una donadora y una receptora) se produce una transferencia de material genético.

1.1.1. Analogía biológica computacional del algoritmo

Se ha simplificado la anatomía bacteriana (vista en la sección **??**) para su adaptación al modelo. En la figura 21 se representa esto.

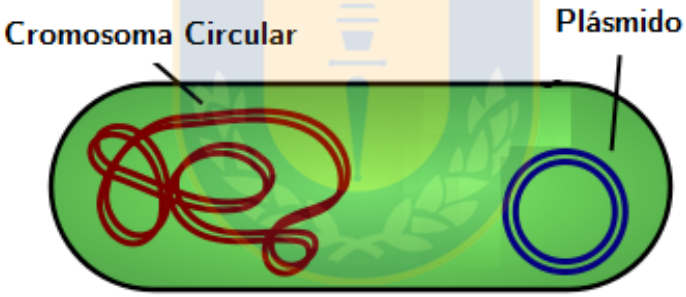

Figura 21: Bacteria simplificada

El cromosoma bacteriano es codificado por un arreglo de datos. Donde cada posición representa un gen (ver figura 22).

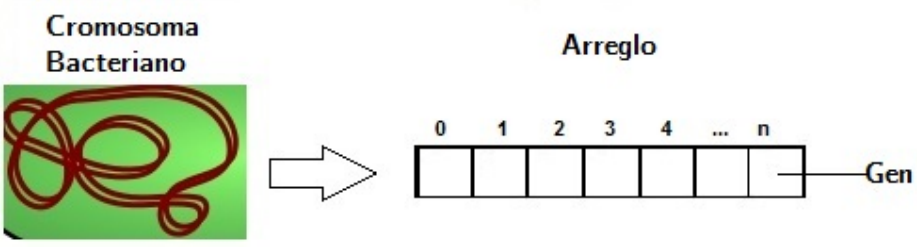

Figura 22: Condificación del cromosoma computacionalmente.

La presencia o ausencia del plásmido en la bacteria nos permite clasificar las bacterias en donadoras (conocidas como F+ y Hfr) y receptoras (F-) respectivamente (ver figura23).

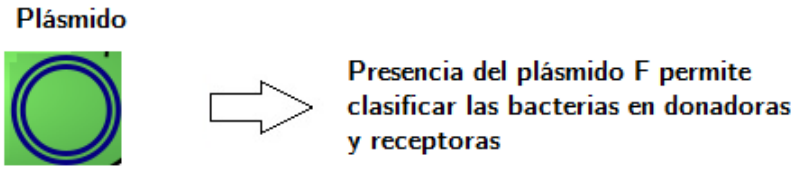

Figura 23: Representación del plásmido

Finalmente una bacteria donadora y receptora intercambiaran material genético (ver figura 24). Dependiendo del tipo de bacteria donadora, la receptora se puede convertir en donadora.

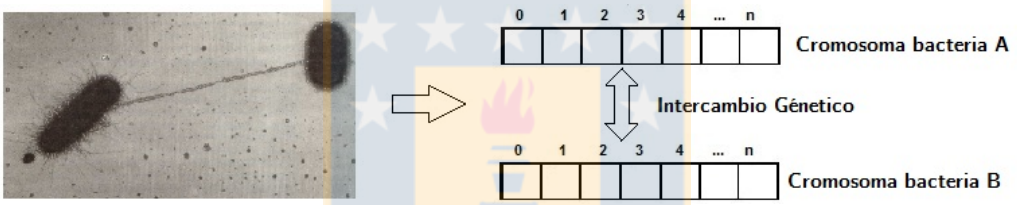

Figura 24: Codificación del intercambio genético

1.1.2. Características esenciales

- El Algoritmo de conjugación bacteriana es evolutivo, es decir va mejorando la población mediante principios de evolución.
- Se asume que cada bacteria tiene un cromosoma circular bacteriano y un plásmido.
- Existen bacterias F+ (Donadoras).Tienen la capacidad de convertir a la bacteria receptora (F-) en donadora transfiriendo parte de su plásmido.
- Existen bacterias Hfr que también son donadoras, pero a diferencia de las F+ no convierten a la bacteria receptora en donadora. Esto tiene su explicación en la parte biológica, pues el plásmido en la Hfr está integrado en el cromosoma.
- Existe la probabilidad de perder el plásmido F+ obtenido. Sin esta probabilidad todas las baterías se convertirían en donadoras en poco tiempo.

- La cantidad de información genética a intercambiar depende del tipo de bacteria donadora que participa en el intercambio (F+ o Hfr). Si es Hfr la cantidad de material genético intercambiado es mayor que si fuera F+.

- El algoritmo es colaborativo, las bacterias que forman la población cooperan entre ellas en vez de competir para su supervivencia. La colaboración se interpreta como el resultado del traspaso de material genético de bacterias donadoras a receptoras en un determinado periodo de tiempo representado por una iteración en el algoritmo.

1.1.3. El ACB se compone de las siguientes operaciones

1. **Inicialización:** En esta fase se genera la población inicial de forma aleatoria. Cada individuo representa un cromosoma bacteriano circular, que puede ser visto como una bacteria. Luego una bacteria representa una solución candidata.

2. **Evaluación:** En esta fase se evalúa mediante una función cada una de las bacterias (posibles soluciones). Esta varía dependiendo del contexto del problema y entrega un valor escalar y sirve para medir la calidad de las soluciones.

3. **Clasificación:** En esta fase se agrupan cada una de las bacterias en donadoras (F+, Hfr) y receptoras (F-). Aquellas que maximicen o minimicen, dependiendo del problema el valor de la función, tienen más probabilidad de ser una célula donadora. Esto solo se realiza al inicio del algoritmo.

4. **Selección:** Consiste en el emparejamiento entre las bacterias donadoras y receptoras de forma aleatoria para el posterior intercambio genético. Cada una tiene la misma probabilidad de ser escogida.

5. **Conjugación:** Se realiza el intercambio de genes entre las bacterias seleccionadas. El intercambio puede ocurrir entre una bacteria F+ y F- o Hfr y F-. Solo la primera opción convierte a la bacteria F- en F+, pero tiene una menor probabilidad de ocurrir que Hfr y F-.

6. **Mutación:** La posición del material genético de una bacteria se puede intercambiar aleatoriamente para mantener la variabilidad de la población.

7. **Curación:** A una bacteria se le quita aleatoriamente su plásmido convirtiéndola en receptora (F-). Esto se realiza para mantener el equilibrio entre bacterias donadoras y receptoras. Sin esta etapa todas las bacterias receptoras en algún punto se convertirían en donadoras, luego el algoritmo llegaría a su fin ante la imposibilidad de nuevas conjugaciones.

8. **Control de la población:** Pese a que existen mecanismos para que se mantengan en equilibrio la cantidad de bacterias donadoras y receptoras (curación), estos no siempre son exitosos. Por este motivo es necesario ir controlando la cantidad de bacterias donadoras y receptoras en cada iteración para que se mantengan en equilibrio.

En la figura 25 se muestra el esquema del algoritmo de conjugación bacteriana.

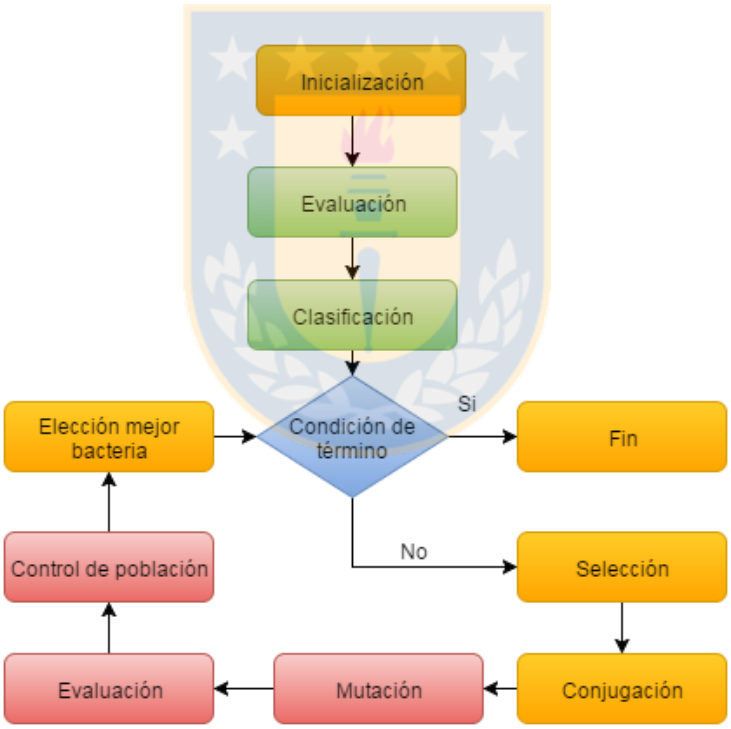

Figura 25: Esquema ACB

1.1.4. Parámetros del algoritmo de conjugación bacteriana

Cepa inicial: Determina la cantidad de bacterias que se generan inicialmente. Esta se mantiene constante durante todas las iteraciones. No debe ser un número muy pequeño, pues no sería representativo.

Número de iteraciones: Cantidad de iteraciones que se utilizarán la ejecución del algoritmo

Cantidad inicial de bacterias donadoras: Aquí se estable la cantidad de bacterias que serán donadoras. Definiendo el número de F+ y Hfr.

Probabilidad de conjugación: Esta probabilidad determina si una pareja seleccionada realiza el intercambio genético.

Criterio de supervivencia: Un cierto porcentaje de bacterias pasan a la siguiente iteración directamente. Con esto se consigue no perder las mejores soluciones.

Porcentaje de curación: Es la probabilidad de que una bacteria pase de donadora a receptora de una iteración a otra.

1.2. Algoritmo Metaheurístico Basado en Evolución horizontal de microorganismos

1.2.1. Modelo Planteado

Se busca crear un nuevo algoritmo bioinspirado que pueda resolver problemas de optimización, basado en la evolución horizontal de microorganismos

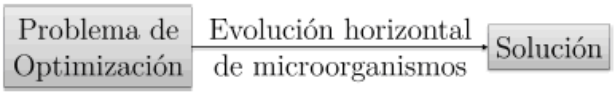

Figura 26: Esquema general

El concepto de evolución horizontal reside en la evolución genética conjunta de todo el grupo en determinado ambiente, se tiende a estabilizar a través del tiempo. Los microorganismos logran esto gracias a:

1. Evolución horizontal: Modificación, transmisión y asimilación de material genético en tiempo real.

2. Percepción de quórum: Comunicación mediante sustancias químicas que induce o no la expresión genética colectiva y coordinación.

3. Estabilidad en colonia: Autorregulación en producción y consumo de recursos, y tendencia a un estado común de los individuos.

El algoritmo surge como la idea de que los microorganismos forman parte de un grupo que tiene reglas de comportamiento. A través de éste se logrará la estabilidad del grupo

1.2.2. Reglas generales

Reglas generales para la creación del algoritmo se derivan de las características 1 y 3 mencionadas anteriormente:

- Modificación o mutación genética: Los microorganismos constantemente modifican o mutan sus códigos genéticos.

- Interacción entre un microorganismo y un enemigo: (Por enemigo se debe considerar cualquier tipo de elemento distinto al microorganismo). Si hay algún enemigo cerca de un microorganismo, este se enfrentará a él (Si el microorganismo posee un código genético superior al de su enemigo, entonces el enemigo morirá y el microorganismo sobrevivirá al enfrentamiento, en caso contrario, el enemigo sobrevivirá y el microorganismo morirá.

- Transferencia horizontal genética: Los microorganismos constantemente transmiten sus códigos genéticos entre sí.

1.2.3. Descripción del algoritmo

Las soluciones se identifican con códigos genéticos de individuos que pueden formar parte de la población de búsqueda. La codificación de una solución se interpreta como el código genético del individuo compuesto de un cierto número de genes.

Se consideran tres operaciones fundamentales: la mutación, el enfrentamiento y la transferencia horizontal de material genético.

Mutación: consiste en modificación de un gen de un individuo. Este cambio se produce al azar.

Enfrentamiento: Se comparan los dos microorganismos teniendo en cuenta su función de evaluación. Sobrevive el que tiene mejor código genético, mientras el otro muere.

Transferencia horizontal de material genético: Consiste en el traspaso de material genético entre un par de microorganismos vecinos.

Cada microorganismo tiene un límite de códigos genéticos que puede llegar a poseer. Esto impide que adquiera códigos sin control por medio de la Transferencia horizontal.

Si un microorganismo aún no ha llegado a su límite, se le agregan los códigos genéticos nuevos que posee el aliado vecino. De no ser así se compara el peor código genético que posee microorganismo con código genético del vecino. Si el microorganismo es peor es remplazado por el código del vecino.

En la figura 27 se muestra el esquema del Algoritmo Metaheurístico Basado en Evolución horizontal de microorganismos

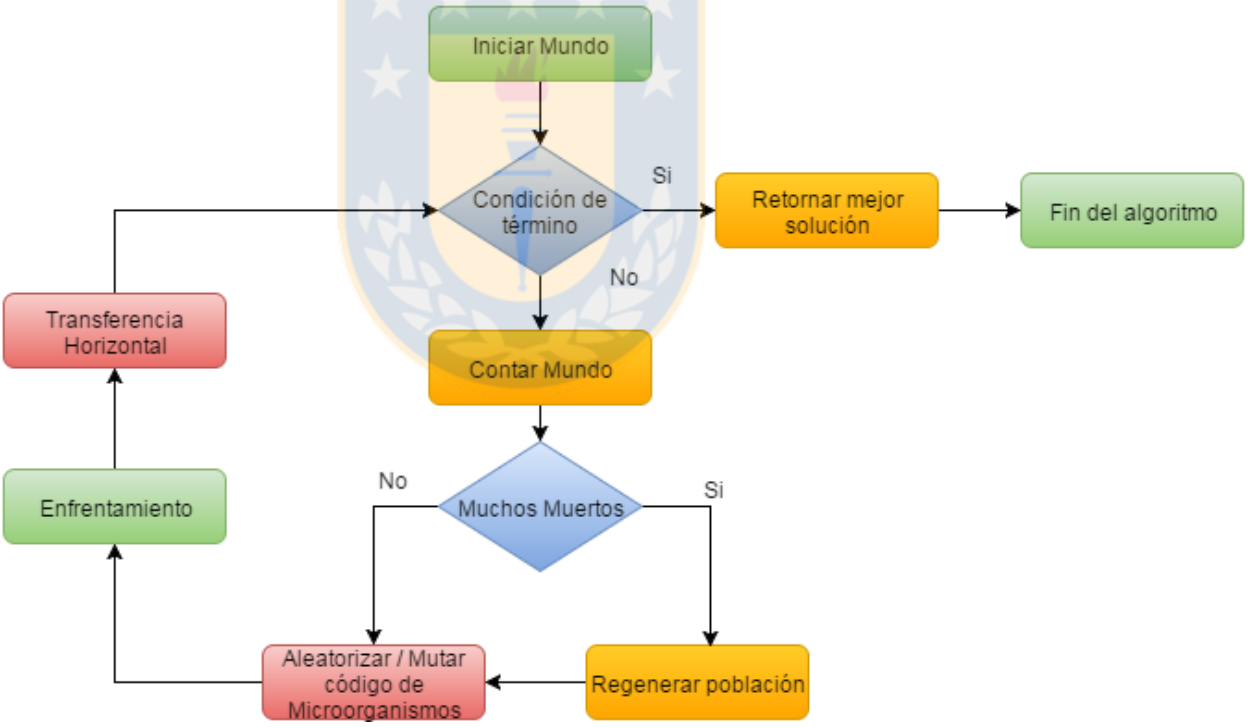

Figura 27: Esquema general

1.2.4. Parámetros del algoritmo

Cantidad de genes en un código genético: Valor que especifica cuántos genes tendrá cada código genético del organismo.

Tamaño del mundo: Par de valores (x,y) que especifican el tamaño del mundo bidimensional en el que interactuarán los individuos.

Población inicial de microorganismos: Cantidad inicial de microorganismos.

Población inicial de enemigos: Cantidad inicial de enemigos.

Porcentaje de muertes de microorganismos que debe cumplirse para desencadenar una regeneración de la población. Esta regeneración genera individuos con códigos genéticos al azar. Con esto se intenta lograr el nacimiento de mejores microorganismos y se evitar el estancamiento evolutivo.

Porcentaje de muertes de enemigos que debe cumplirse para desencadenar una regeneración de la población. A diferencia de la regeneración de microorganismos, estos no se regeneran aleatoriamente, sino que son iguales al mejor código genético (de los microorganismos) encontrado hasta aquel momento. Esto es necesario pues a medida que pasan las iteraciones la calidad de los microorganismos va mejorando (evolución) haciendo menos influyente a los enemigos.

Criterio de detención

Existen varios criterios que sirven como detención, cierta cantidad de iteraciones sin mejorar, cierta cantidad total de iteraciones o cierta cantidad de tiempo.

2. Detalle Ajuste de parámetros ACBCE

2.1. Probabilidad de mutación

Con el fin de encontrar la probabilidad de mutación que optimice los resultados del programa se realizaron pruebas con distintos porcentajes de mutación: 0 %, 10 %, 20 %, 30 %, 40 %, 50 %, 60 %, 70 %, 80 %, 90 % y 100 %. Para cada porcentaje se realizaron cinco ejecuciones del programa y se guardó

la mejor encontrada. La instancia elegida para las pruebas de manera aleatoria fue A-n39-k5.

El valor de los otros parámetros se mantuvo fijo y se muestran en la siguiente tabla:

Parámetro	Valor
Número de iteraciones	1.000.000
Tamaño de la población	100
Número de antibióticos	10
Método de selección	Aleatoria
Método de conjugación	OX
Porcentaje de acción del antibiótico	20
Factor de conjugación	75

Cuadro 3: Parámetros de pruebas probabilidad de mutación

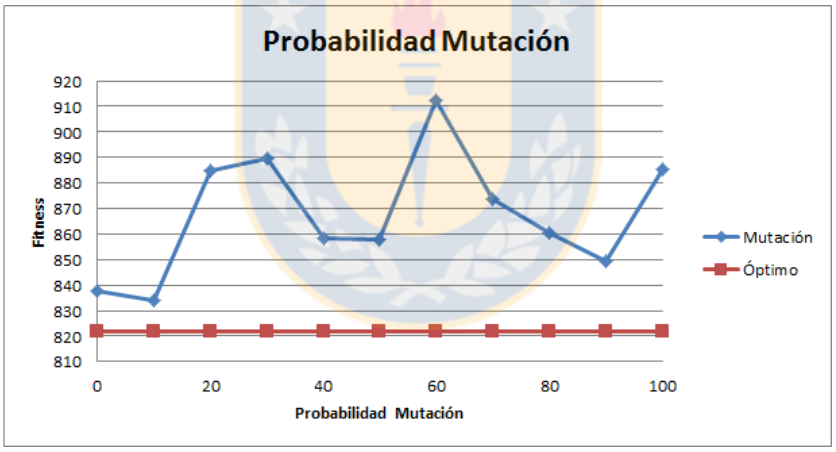

Figura 28: Variación Probabilidad de Mutación

Los resultados presentados en la figura 28 muestran que el mejor valor para el parámetro de probabilidad de mutación es de un 10 %, donde se alcanza el mejor fitness.

2.2. Tamaño de la población

Para encontrar el tamaño ideal de la población se realizaron pruebas en la instancia A-n39-k5. Para ella se realizaron cinco pruebas por cada tamaño distinto.

Los parámetros restantes fueron los mismos que en el cuadro 3 y el valor de la mutación fue de que 10 % se definió anteriormente.

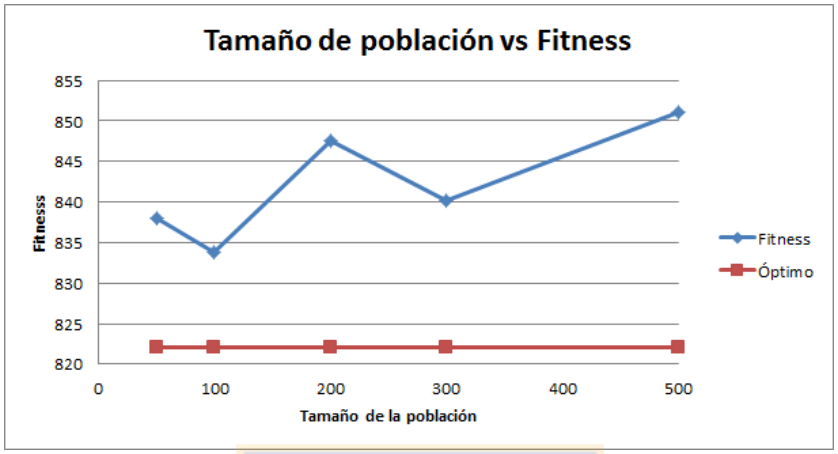

Figura 29: Variación de tamaño de la población

Los resultados presentados en la figura 29 muestran que el tamaño de la población ideal es de 100 bacterias. Cabe mencionar que el tiempo aumenta considerablemente al disponer de más bacterias.

2.3. Número de iteraciones

El número de iteraciones que se ejecutan en el programa se fijó en 1.000.000 ya que se observa que no mejora una vez que ha pasado este número. Se probó con A-n39-k5 y A-n37-k5, pero solo esta última el fitness mejoraba en las últimas iteraciones. En la figura 30 se muestra que para la instancia A-n37-k5 va disminuyendo el fitness a medida que pasan las iteraciones y que paulatinamente se estanca en un valor menor al millón fijado.

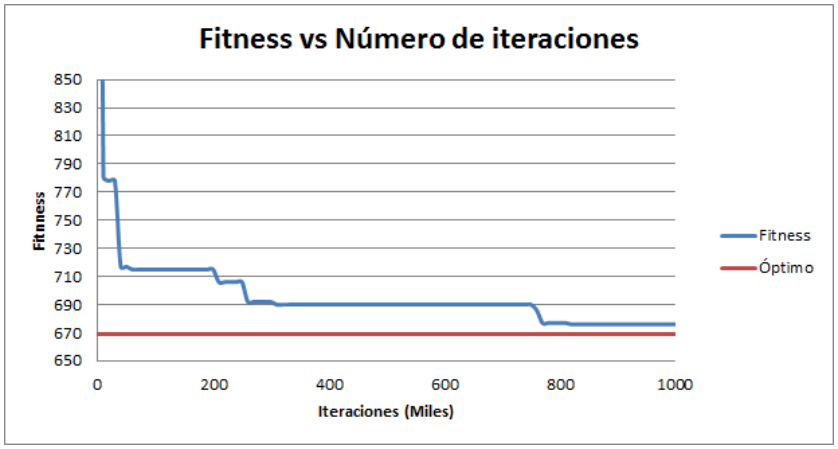

Figura 30: Fitness vs Iteraciones

2.4. Porcentaje de acción del antibiótico

Se realizaron pruebas con distintos valores para este parámetro, por cada instancia se ejecutó 5 veces el programa y se guardó la mejor.

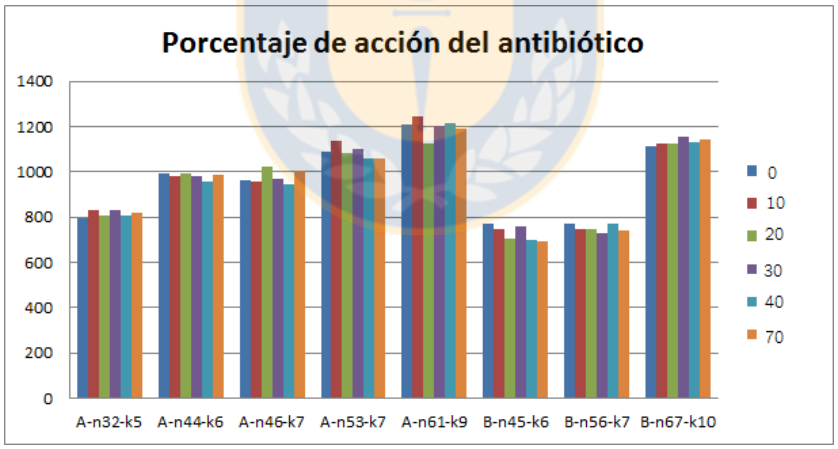

Figura 31: Porcentaje de acción del antibiótico

Los resultados vistos en la figura 32 no muestran un Porcentaje de acción que sea mejor todas las instancias. Sino que algunas funcionan bien con un porcentaje en específico, pero funciona mal en otras. Por lo tanto este parámetro será variable dependiendo de la instancia para no limitarse con un porcentaje y obtener los mejores resultados posibles.

2.5. Número de antibióticos

Luego de realizar distintas pruebas, se obtuvo mejores resultados con 10 antibióticos distintos.

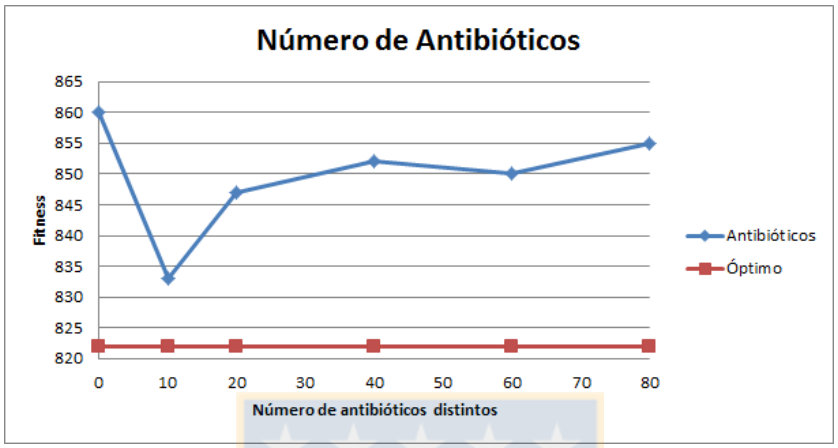

Figura 32: Número de antibióticos distintos

2.6. Factor de conjugación

Se mantuvo el valor ocupado en el ACBCP.

3. Detalle resultados obtenidos

3.1. Resultados ACBCE

Las pruebas fueron realizadas en un PC (Windows 7 de 64 bits) con procesador Intel Core i7 4970 de 3.60 GHz y 8GB de RAM.

En los cuadros 4 y 5 se muestran el mejor resultado obtenido por el ACBCE para cada instacia.

Instancia	Óptimo	ACBCE	RPD
A-n32-k5	784	797,45123	1,715718112
A-n33-k5	661	662,1101	0,167942511
A-n33-k6	742	743,004	0,135309973
A-n34-k5	778	780,9359	0,377365039
A-n36-k5	799	806,7826	0,974042553
A-n37-k5	669	676,2488	1,083527653
A-n37-k6	949	950,8522	0,195173867
A-n38-k5	730	735,0513	0,691958904
A-n39-k5	822	833,8419	1,440620438
A-n39-k6	831	838,92	0,953068592
A-n44-k6	937	952,933	1,700426894
A-n45-k6	944	994,1217	5,309502119
A-n45-k7	1146	1187,1437	3,590200698
A-n46-k7	914	945,2072	3,414354486
A-n48-k7	1073	1085,492	1,164212488
A-n53-k7	1010	1056,5367	4,607594059
A-n54-k7	1167	1216,2593	4,221019709
A-n55-k9	1073	1085,625	1,176607642
A-n60-k9	1354	1421,3276	4,972496307
A-n61-k9	1034	1124,3027	8,733336557
A-n62-k8	1288	1402,6434	8,900885093
A-n63-k9	1314	1373,9944	4,565783866
A-n63-k10	1616	1723,2069	6,634090347
A-n64-k9	1401	1516,8025	8,265703069
A-n65-k9	1174	1260,7625	7,390332198
A-n69-k9	1159	1256,7014	8,429801553
A-n80-k10	1763	1938,7191	9,967050482

Cuadro 4: Mejor solución encontrada con ACBCE para cada instancia.

Resultados instancias tipo B.

Instancia	Óptimo	ACBCE	RPD
B-n31-k5	672	676,0885	0,608407738
B-n34-k5	788	790,6982	0,342411168
B-n35-k5	955	956,294	0,135497382
B-n38-k6	805	810,7558	0,715006211
B-n39-k5	549	555,5166	1,186994536
B-n41-k6	829	836,7947	0,940253317
B-n43-k6	742	752,11835	1,36365903
B-n44-k7	909	929,9146	2,300836084
B-n45-k5	751	762,92804	1,588287617
B-n45-k6	678	695,5625	2,590339233
B-n50-k7	741	752,31165	1,526538462
B-n50-k8	1312	1334,5969	1,722324695
B-n51-k7	1032	1055,205	2,248546512
B-n52-k7	747	761,43726	1,932698795
B-n56-k7	707	730,67615	3,348818953
B-n57-k7	1153	1169,3256	1,415923677
B-n57-k9	1598	1635,8316	2,367434293
B-n63-k10	1496	1559,1841	4,223536096
B-n64-k9	861	902,11766	4,775570267
B-n66-k9	1316	1355,0751	2,969232523
B-n67-k10	1032	1104,3969	7,015203488
B-n68-k9	1272	1331,568	4,683018868
B-n78-k10	1221	1325,2302	8,536461916

Cuadro 5: Mejor solución encontrada con ACBCE para cada instancia.

3.2. Resultados otros modelos

En el cuadro 6 y 7 se muestran los resultados obtenidos por ACBCP, AG y ABC.

Instancia	Óptimo	ACBCP	RPD (ACB)	AG	RPD(AG)	ABC	RPD(ABC)
A-n32-k5	784	811	3,44388	797,45	1,72	787,08	0,39
A-n33-k5	661	676	2,26929	662,11	0,17	662,26	0,19
A-n33-k6	742	746	0,53908	742,69	0,09	742,69	0,09
A-n34-k5	778	811	4,24165	786,8	1,13	780,94	0,38
A-n36-k5	799	842	8,08729	802,13	0,39	802,13	0,39
A-n37-k5	669	732	9,41704	672,47	0,52	672,47	0,52
A-n37-k6	949	989	4,21496	950,85	0,2	957,79	0,93
A-n38-k5	730	756	3,56164	738,01	1,1	740,78	1,48
A-n39-k5	822	861	4,74453	831,75	1,19	830,21	0,99
A-n39-k6	831	870	4,69314	835,25	0,51	835,25	0,51
A-n44-k6	937	992	5,8698	947,74	1,15	950,4	1,43
A-n45-k6	944	1139	14,58753	981,17	3,94	956,76	1,35
A-n45-k7	1146	1177	10,12216	1164,09	1,58	1160,43	1,23
A-n46-k7	914	1022	11,81619	925,35	1,24	918,13	0,45
A-n48-k7	1073	1166	8,66729	1074,34	0,12	1077,3	0,4
A-n53-k7	1010	1159	14,75248	1037,99	2,77	1035,65	2,54
A-n54-k7	1167	1262	8,14053	1190,22	1,99	1187,09	1,72
A-n55-k9	1073	1149	7,08295	1098,29	2,36	1081,71	0,81
A-n60-k9	1354	1511	7,31534	1387,88	2,5	1391,51	2,77
A-n61-k9	1034	1264	22,24371	1053,85	1,82	1061,61	2,67
A-n62-k8	1288	1539	19,30233	1333,78	3,39	1331,28	3,36
A-n63-k9	1314	1442	6,37376	1348,56	2,55	1361,16	3,58
A-n63-k10	1616	1719	9,74125	1700,28	4,06	1664,86	3,02
A-n64-k9	1401	1578	12,5535	1463,56	4,39	1452,8	3,69
A-n65-k9	1174	1417	20,39082	1220,64	3,71	1210,75	3,13
A-n69-k9	1159	1308	11,9863	1211,37	3,71	1200,92	3,62
A-n80-k10	1763	2104	19,34203	1864,66	5,71	1862,32	5,63

Cuadro 6: Resultados instancias A

Resultados instancias tipo B.

Instancia	Óptimo	ACBCP	RPD (ACB)	AG	RPD(AG)	ABC	RPD(ABC)
B-n31-k5	672	688	2,38	676,09	0,61	676,09	0,61
B-n35-k5	955	991	3,76	958,89	0,41	956,29	0,14
B-n38-k6	805	817	1,49	808,7	0,46	808,7	0,46
B-n39-k5	549	569	3,64	553,52	0,82	553,16	0,76
B-n41-k6	829	858	3,49	846,06	2,06	836,99	0,96
B-n43-k6	742	792	6,73	746,98	0,67	746,98	0,67
B-n44-k7	909	955	5,06	931,72	2,5	931,51	2,48
B-n45-k5	751	810	7,85	755,72	0,63	755,11	0,55
B-n45-k6	678	759	11,94	695,02	2,51	715,58	5,46
B-n50-k7	741	770	3,91	747,69	0,9	744,34	0,45
B-n50-k8	1312	1343	2,28	1331,98	1,52	1334,22	1,69
B-n51-k7	1032	1129	9,39	1026,94	-0,49	1022,8	-0,89
B-n52-k7	747	826	10,57	750,54	0,47	753,63	0,89
B-n56-k7	707	823	16,40	720,23	1,87	720,06	1,85
B-n57-k7	1153	1342	16,39	1150,78	-0,19	1145,4	-0,66
B-n57-k9	1598	1715	7,32	1629,66	2	1625,8	1,74
B-n63-k10	1496	1642	6,83	1563,64	4,72	1562,75	4,46
B-n64-k9	861	1026	19,16	901,08	4,66	894,16	3,85
B-n66-k9	1316	1448	5,38	1346,61	2,33	1353,73	2,87
B-n67-k10	1032	1170	13,37	1080,22	4,67	1071,47	3,82
B-n68-k9	1272	1459	11,88	1303,31	2,46	1300,83	2,27
B-n78-k10	1221	1459	19,49	1280,78	4,9	1278,58	4,72

Cuadro 7: Resultados instancias B

En el cuadro 8 se muestra un resumen de los resultados de los RPDs obtenidos de las instancias de Augerat en los distintos modelos que se están comparando.

	Grupo A				Grupo B			
RPD	ACBCE	ACBCP	AG	ABC	ACBCE	ACBCP	AG	ABC
Mínimo	0,135	0,539	0,09	0,09	0.14	1.49	-0.49	-0.49
Máxino	9,967	22,243	5,71	5,63	8.54	19.16	4.90	5.54
Promedio	3,732	9,462	2,00	1,79	2.53	8.58	1.77	1.72

Cuadro 8: Resumen comparativo de los resultados

En el cuadro 5 y en la figura 33 se muestran el número de instancias que pertenecen a cada rango definido por su RPD.

Rango RPD	Número de instancias
0 y 0,99	12
1 y 2,99	18
3 y 4.99	10
5 y 9.99	10

Cuadro 9: instancias clasificadas por su RDP

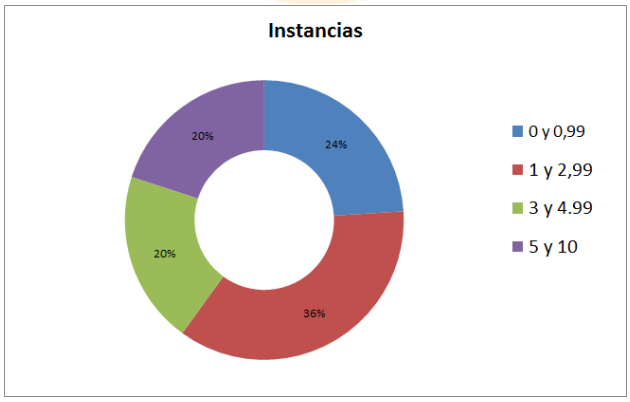

Figura 33: Rango de RPD vs cantidad de instancias para ABC

3.3. Resultados obtenidos variando el valor de los parámetros

En los cuadros 10,11,y 12 se muestran los resultados obtenidos al variar cada uno de los parámetros del algoritmo.

Mutación	Fitness	Óptimo
0 %	837.6252	822
10 %	833.8419	822
20 %	884.5998	822
30 %	889.8351	822
40 %	858.3431	822
50 %	857.7806	822
60 %	912.3794	822
70 %	973.7997	822
80 %	860.2726	822
90 %	849.3252	822
100 %	885.1462	822

Cuadro 10: Mejor fitness encontrado para cada porcentaje de mutación

Tamaño de la población	Fitness	Tiempo ejecución (s)
50	838.0286	130
100	833.8419	273
200	847.4919	625
300	840.1228	1082
500	851.9311	2402

Cuadro 11: Variación del tamaño de la población

Instancia/Porcentaje	0	10	20	30	40	70
A-n32-k5	797	832	808	832	804	819
A-n44-k6	990	979	995	979	958	989
A-n46-k7	963	959	1020	969	945	1004
A-n53-k7	1088	1138	1085	1098	1056	1059
A-n61-k9	1208	1245	1124	1203	1217	1188
B-n45-k6	771	746	702	760	699	695
B-n56-k7	771	749	745	730	769	741
B-n67-k10	1115	1126	1125	1152	1130	1140

Cuadro 12: Pruebas realizadas al porcentaje de acción del antibiótico.

3.4. Tiempo ejecución algoritmo ACBCE

En el cuadro 13 se muestra el tiempo promedio de ejecución (en segundos) dependiendo de el número de clientes del problema.

Numero Clientes	Tiempo ejecución(segundos)
31	247
32	234
33	254
34	256
35	265
36	265
37	272
38	282
40	310
42	314
43	312
44	312
45	337
47	342
50	356
51	375
52	380
53	393
54	407
55	400
56	427
59	445
60	422
61	454
62	456
63	470
64	474
65	468
66	476
67	478
68	490
77	559
79	562

Cuadro 13: Tiempo de ejecución ACBCE

4. Manual de usuario del programa

Cuando se ejecuta el programa abrirá la interfaz mostrada en la figura 34. Luego se asignan los valores que se desean a los parámetros y posteriormente se ingresan los nombres de los archivos distancias y demandas que para cada instancia de Augerat es distinto. Estas equivalencias se muestran en los cuadro 14 y 15. Finalmente es necesario hace clic en comenzar para iniciar la prueba.

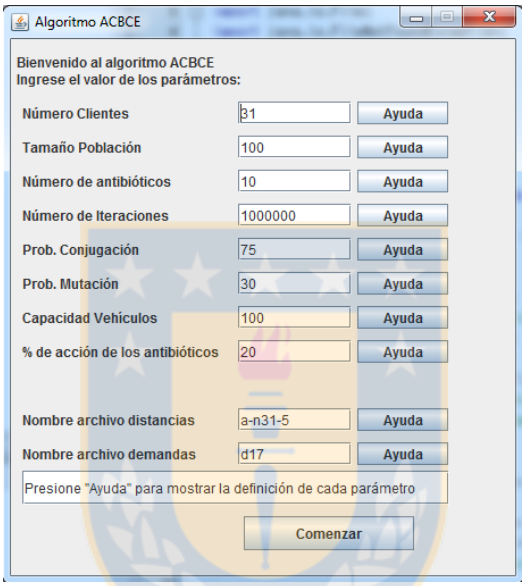

Figura 34: Interfaz del programa

	Archivos de entrada	
Instancia	Distancia	Demanda
A-n32-k5	a-n31-5	d17
A-n33-k5	a-n32-5	d1
A-n33-k6	a-n32-6	d18
A-n34-k5	a-n33-5	d19
A-n36-k5	a-n35-5	d20
A-n37-k5	a-n36-5	d2
A-n37-k6	a-n36-6	d21
A-n38-k5	a-n37-5	d22
A-n39-k5	a-n38-5	d3
A-n39-k6	a-n38-6	d4
A-n44-k6	a-n43-6	d23
A-n45-k6	a-n44-6	d5
A-n45-k7	a-n44-7	d24
A-n46-k7	a-n45-7	d25
A-n48-k7	a-n47-7	d26
A-n53-k7	a-n52-7	d6
A-n54-k7	a-n53-7	d27
A-n55-k9	a-n54-9	d7
A-n60-k9	a-n59-9	d28
A-n61-k9	a-n60-9	d8
A-n62-k8	a-n61-8	d29
A-n63-k9	a-n62-9	d9
A-n63-k10	a-n62-10	d10
A-n64-k9	a-n63-9	d30
A-n65-k9	a-n64-9	d31
A-n69-k9	a-n68-9	d32
A-n80-k10	a-n79-10	d11

Cuadro 14: Equivalencia archivos de entrada y Augerat tipo A

	Archivos de entrada	
Instancia	Distancia	Demanda
B-n31-k5	b-n30-5	d33
B-n34-k5	b-n33-5	d34
B-n35-k5	b-n34-5	d35
B-n38-k6	b-n37-6	d36
B-n39-k5	b-n38-5	d37
B-n41-k6	b-n40-6	d38
B-n43-k6	b-n42-6	d39
B-n44-k7	b-n43-7	d40
B-n45-k5	b-n44-5	d41
B-n45-k6	b-n44-6	d12
B-n50-k7	b-n49-7	d42
B-n50-k8	b-n49-8	d43
B-n51-k7	b-n50-8	d13
B-n52-k7	b-n51-7	d44
B-n56-k7	b-n55-7	d45
B-n57-k7	b-n56-6	d14
B-n57-k9	b-n56-9	d46
B-n63-k10	b-n62-10	d47
B-n64-k9	b-n63-9	d48
B-n66-k9	b-n65-9	d49
B-n67-k10	b-n66-9	d15
B-n68-k9	b-n67-9	d50
B-n78-k10	b-n77-10	d16

Cuadro 15: Equivalencia archivos de entrada y Augerat tipo B

5. Mejores rutas encontradas

Acontinuacion se muestran la mejor ruta encontrada por cada instancia.

Instancia	Ruta
A-n32-k5	{1 4 3 24 29 5 12 9 19 15 1}, {1 21 6 26 11 16 10 23 30 1} { 1 7 18 20 32 22 14 27 1 }, { 1 31 17 8 2 13 1 }, { 1 25 28 1}
A-n33-k5	{1 25 7 20 15 22 2 32 12 1}, {1 13 6 27 8 9 14 33 3 1}, {1 30 17 4 10 18 16 1} {1 11 31 26 28 5 21 1} {1 24 19 29 23 1 }
A-n33-k6	{1 22 13 11 1} {1 14 7 19 2 15 1} {1 8 20 30 12 18 1} {1 29 28 31 17 26 33 1} {1 21 10 16 3 4 5 9 6 1} {1 32 24 25 27 23 1}
A-n34-k5	{1 15 30 9 16 7 8 1} {1 5 27 6 31 25 1} {1 21 34 17 23 4 13 10 3 1} {1 11 18 20 12 24 2 28 1} {1 14 26 32 29 33 22 19 1}
A-n36-k5	{1 2 23 33 14 18 31 30 34 19 22 1} {1 13 32 20 5 4 7 24 15 1} {1 16 9 36 3 35 29 10 1} {1 17 12 25 28 26 6 21 1} {1 27 8 11 1}
A-n37-k5	{1 23 14 11 7 6 34 5 8 1} {1 2 13 3 20 21 24 15 18 1} {1 16 35 27 36 19 32 29 33 30 37 1} {1 31 26 9 28 12 10 25 4 1} {1 22 17 1}
A-n37-k6	{1 34 3 29 24 23 13 12 11 1} {1 14 31 16 33 28 1} {1 21 9 6 4 2 35 18 19 1} {1 32 20 10 22 27 5 1} {1 25 30 37 7 15 1} {1 17 36 26 8 1}
A-n38-k5	{1 22 27 13 4 2 5 17 26 7 20 1} {1 15 34 36 24 9 10 1} {1 25 3 18 37 14 16 33 21 8 1} {1 29 32 38 12 28 23 6 1} {1 19 35 30 31 11 1}
A-n39-k5	{1 3 23 4 8 17 33 11 32 1} {1 20 26 34 2 24 21 30 6 16 39 1} {1 15 13 10 19 5 1} {1 9 27 28 35 38 36 25 18 1} {1 12 7 37 29 14 31 22 1}
A-n39-k6	{1 14 16 1} {1 19 23 35 17 11 28 33 1} {1 3 34 20 5 9 8 21 1} {1 12 37 18 24 22 2 7 1} {1 4 13 39 38 32 36 26 15 31 1} {1 25 30 29 10 6 27 1}
A-n44-k6	{1 27 12 17 21 19 36 2 41 1} {1 35 40 31 24 26 1} {1 9 16 29 28 20 32 1} {1 5 18 13 4 7 3 1} {1 23 37 10 30 44 14 39 15 42 1} {1 25 34 11 38 43 33 22 6 8 1}
A-n45-k6	{1 25 38 35 19 30 1} {1 7 45 2 36 32 15 1} {1 39 3 26 13 40 37 23 16 10 1} {1 29 8 14 21 4 11 1} {1 33 6 22 34 42 9 17 5 43 1} {1 24 44 18 41 31 20 12 28 27 1}
A-n45-k7	{1 7 29 4 12 45 25 22 1} {1 13 3 5 35 36 40 1} {1 34 28 30 42 44 1} {1 41 8 39 18 24 26 43 16 9 1} {1 11 2 38 31 23 1} {1 6 27 37 20 32 15 10 1} {1 19 21 17 1}

A-n46-k7	{1 37 6 4 45 26 19 36 13 1} {1 24 15 27 14 31 5 39 1} {1 29 42 21 28 2 41 35 32 30 22 1} {1 40 23 8 11 12 18 1} {1 9 34 43 17 25 16 20 1} {1 33 38 3 44 7 46 1} {1 10 1}
A-n48-k7	{1 42 3 11 48 18 15 1} {1 17 34 22 31 47 14 30 29 1} {1 24 44 32 2 6 13 1} {1 7 23 26 20 39 37 33 1} {1 10 25 5 12 43 16 28 35 46 1} {1 38 4 21 27 40 9 8 41 1} {1 45 36 19 1}
A-n53-k7	{1 47 9 36 28 52 1} {1 29 23 46 49 43 44 37 45 2 1} {1 26 22 14 15 6 4 40 1} {1 39 19 41 27 11 30 50 31 1} {1 48 5 32 21 7 34 1} {1 35 53 12 25 42 18 10 1} {1 38 3 51 24 20 16 33 17 13 8 1}
A-n54-k7	{1 24 21 50 37 2 46 27 18 11 36 1} {1 17 54 4 23 14 1} {1 20 9 32 41 49 38 28 1} {1 31 35 47 52 43 25 13 3 15 53 1} {1 39 22 34 10 45 12 1} {1 26 48 42 33 7 16 30 44 1} {1 19 6 51 40 8 29 5 1}
A-n55-k9	{1 31 23 20 28 14 55 29 1} {1 48 40 50 10 36 44 1} {1 13 33 39 17 41 54 6 11 1} {1 19 51 2 45 25 53 24 30 1} {1 38 4 47 21 32 43 1} {1 35 18 27 15 8 5 1} {1 16 12 52 3 34 37 1} {1 49 46 7 1} {1 26 42 9 22 1}
A-n60-k9	{1 44 57 13 52 10 33 51 1} {1 36 56 16 27 9 14 1} {1 8 30 38 58 18 28 20 1} {1 15 40 43 55 23 2 1} {1 48 59 46 6 11 37 3 1} {1 35 25 49 24 1} {1 7 29 45 50 31 54 32 1} {1 26 47 41 12 22 5 4 21 17 1} {1 42 34 39 60 53 19 1}
A-n61-k9	{1 7 42 54 6 3 50 1} {1 16 39 4 1} {1 59 26 30 37 22 28 57 48 40 1} {1 14 51 13 23 5 27 44 1} {1 24 25 11 60 9 21 41 1} {1 35 36 19 49 17 2 29 18 1} {1 10 45 61 12 32 53 58 1} {1 52 55 47 46 38 31 43 1} {1 56 33 34 20 8 15 1}
A-n62-k8	{1 53 37 38 3 47 31 19 14 1} {1 29 54 2 55 4 40 28 24 42 22 56 32 35 1} {1 8 62 30 44 9 11 52 1} {1 17 23 45 20 39 13 49 1} {1 16 7 33 58 41 21 1} {1 51 18 59 34 27 60 1} {1 12 5 36 50 26 61 6 15 57 1} {1 10 43 25 48 46 1}
A-n63-k9	{1 54 56 33 59 52 58 63 48 44 8 1} {1 29 23 51 15 41 22 1} {1 13 42 4 62 1} {1 27 20 21 55 30 26 1} {1 57 16 45 34 12 1} {1 28 60 25 9 3 31 1} {1 5 19 2 47 32 39 10 43 17 1} {1 14 35 6 46 50 37 49 61 1} {1 18 36 38 7 53 24 40 11 1}
A-n63-k10	{1 62 61 34 38 55 1} {1 63 51 29 58 33 35 12 48 45 1} {1 49 5 27 14 21 1} {1 36 7 28 9 17 1} {1 57 30 26 16 19 60 11 1} {1 22 47 24 8 1} {1 53 44 39 31 37 1} {1 32 52 59 23 56 25 1} {1 18 41 54 46 40 4 43 20 6 1} {1 15 3 13 10 42 2 50 1}
A-n64-k9	{1 10 35 5 6 51 24 1} {1 57 53 14 62 25 33 43 1} {1 55 59 3 64 48 47 9 17 61 42 1} {1 11 21 20 52 49 1} {1 34 63 46 1} {1 19 23 13 15 45 32 60 38 1} {1 54 4 56 50 40 41 37 8 22 27 36 1} {1 16 31 28 2 18 29 44 39 1} {1 26 58 12 30 7 1}

A-n65-k9	{1 54 57 45 56 1} {1 60 41 11 9 53 20 14 19 30 1} {1 46 62 43 39 3 42 17 51 61 1} {1 47 65 7 27 32 35 48 1} {1 18 52 40 33 59 21 6 1} {1 16 23 10 15 28 44 1} {1 29 24 58 49 55 64 12 8 1} {1 63 22 26 25 13 2 34 1} {1 50 31 38 36 37 4 5 1}
A-n69-k9	{1 24 31 10 50 34 3 19 32 1} {1 53 11 18 33 38 55 27 1} {1 23 14 45 16 4 7 52 54 1} {1 59 67 42 21 60 9 37 2 49 17 51 1} {1 44 6 47 64 12 43 58 1} {1 66 56 61 5 48 26 15 35 1} {1 29 28 22 62 65 68 8 1} {1 13 39 69 46 36 20 1} {1 25 30 41 40 57 63 1}
A-n80-k10	{1 26 42 34 16 57 70 66 36 27 58 62 52 1} {1 77 73 55 10 56 48 20 76 21 31 1} {1 78 4 30 28 60 47 65 40 1} {1 11 64 45 13 54 37 74 1} {1 72 15 49 19 80 29 53 1} {1 59 33 5 23 46 51 50 1} {1 75 18 32 61 71 39 67 68 1} {1 63 24 25 7 6 43 1} {1 12 35 3 38 9 44 17 69 79 14 1} {1 2 8 22 41 1}
B-n31-k5	{1 30 5 26 6 19 17 22 1} {1 31 24 9 13 29 27 1} {1 4 2 20 25 12 16 15 1} {1 23 7 10 18 14 8 1} {1 21 28 11 3 1}
B-n35-k5	{1 7 34 4 19 27 2 24 10 8 21 1} {1 16 31 6 12 20 3 33 14 1} {1 35 28 25 29 5 15 23 1} {1 18 30 11 17 13 1} {1 9 32 26 22 1}
B-n38-k6	{1 21 11 2 16 15 1} {1 37 18 9 35 25 30 13 8 26 1} {1 24 34 19 5 6 28 1} {1 31 3 36 38 14 29 22 1} {1 33 4 12 23 17 27 1} {1 32 7 20 10 1}
B-n39-k5	{1 15 14 19 7 21 20 11 30 1} {1 2 27 22 25 24 31 17 23 38 34 3 1} {1 4 29 5 36 18 37 32 1} {1 28 39 33 26 13 12 16 6 1} {1 10 35 8 9 1}
B-n41-k6	{1 14 31 8 17 41 21 23 1} {1 40 20 27 28 39 5 1} {1 26 7 12 34 35 29 1} {1 2 18 15 30 9 32 16 6 1} {1 25 36 3 4 19 22 38 10 1} {1 37 11 24 13 1 33}
B-n43-k6	{1 8 17 16 13 39 37 35 43 1} {1 27 7 25 23 14 12 4 1} {1 11 2 38 29 15 20 19 6 1} {1 31 30 33 26 28 41 42 36 1} {1 10 32 3 24 9 40 18 5 1} {1 34 22 21 1 }
B-n44-k7	{1 14 9 43 31 23 39 10 1} {1 33 12 24 11 1} {1 20 5 27 25 40 15 37 3 1} {1 16 34 44 32 28 36 21 1} {1 26 29 4 13 2 41 1} {1 19 30 38 42 35 22 7 1} {1 6 18 17 8 1}
B-n45-k5	{1 38 33 8 35 28 15 45 19 1} {1 42 23 5 17 34 20 27 26 12 30 1} {1 40 6 44 9 22 13 7 41 43 1} {1 29 14 3 16 10 2 37 24 1} {1 31 18 11 25 4 32 39 36 21 1}
B-n50-k7	{1 7 29 33 50 1} {1 26 19 15 44 25 28 17 11 43 4 1} {1 36 5 30 24 47 39 13 23 34 1} {1 10 2 14 27 40 48 1} {1 38 12 21 9 6 37 46 22 1} {1 41 16 3 49 18 20 45 35 1} {1 32 42 8 31 1}
B-n50-k8	{1 7 28 13 38 8 35 1} {1 24 36 40 46 42 1} {1 6 17 45 1} {1 2 29 30 20 33 1} {1 16 18 43 9 22 11 1} {1 10 34 50 44 37 21 31 15 32 1} {1 12 25 48 27 19 5 26 41 49 3 1} {1 39 23 47 14 4 1}
B-n51-k7	{1 5 14 43 15 29 27 44 1} {1 37 11 35 39 3 18 17 12 1} {1 36 4 20 34 49 7 8 1} {1 6 41 19 30 40 1} {1 42 33 38 13 9 32 28 1} {1 47 26 46 48 16 51 24 45 22 1} {1 23 25 31 50 10 2 21 1}

B-n52-k7	{ 1 47 14 23 18 50 16 20 33 35 39 3 1} {1 17 10 27 51 13 24 46 5 1} {1 41 43 21 19 31 2 37 1} {1 12 29 4 32 25 40 15 22 49 1} {1 38 30 28 9 6 11 45 1} {1 48 52 8 44 36 34 1} {1 26 7 42 1}
B-n56-k7	{1 41 6 13 10 39 51 23 24 44 1} {1 43 38 5 40 27 22 48 1} {1 26 20 8 16 30 42 2 53 1} {1 14 50 34 12 56 21 32 55 31 35 3 1} {1 47 29 45 15 52 18 9 11 19 1} {1 28 17 36 49 33 37 4 1} {1 46 25 7 54 1}
B-n57-k7	{1 38 33 48 21 44 8 1} {1 57 29 26 37 39 18 20 27 1} {1 42 35 52 10 9 54 13 19 1} {1 47 25 45 4 11 22 46 50 49 2 56 1} {1 55 36 41 30 5 1} { 1 32 7 12 16 1} {1 51 53 43 23 15 40 31 3 1} {1 17 28 34 14 6 24 1}
B-n57-k9	{1 13 49 2 28 29 10 11 18 1} {1 46 25 53 55 44 1} {1 57 23 4 7 1} {1 5 31 50 39 34 36 54 1} {1 47 32 56 51 41 33 1} {1 12 17 14 45 35 1} {1 15 20 40 27 24 3 1} {1 8 43 21 38 52 26 16 9 19 1} ({1 6 37 22 42 30 48 1}
B-n63-k10	{1 35 53 20 52 3 63 40 1} {1 29 30 14 33 49 1} {1 4 41 17 45 56 1} {1 26 36 7 27 34 22 23 1} {1 11 42 55 13 59 15 62 43 1} {1 38 54 24 12 18 44 25 58 39 1} {1 31 9 47 60 57 50 32 1} {1 16 48 10 5 6 61 1} {1 46 21 8 37 28 1} {1 2 51 19 1}
B-n64-k9	{1 31 18 32 39 14 41 28 9 47 1} {1 16 15 36 37 61 49 38 1} {1 8 55 22 35 42 2 1} {1 13 11 51 19 62 1} {1 20 25 6 5 60 52 56 23 10 33 4 1} {1 63 53 40 43 64 3 24 26 1} {1 34 48 30 12 29 57 17 1} {1 59 54 7 46 45 21 58 1} {1 50 44 27 1}
B-n66-k9	{1 53 55 24 19 54 21 1} {1 14 44 52 35 23 57 46 1} {1 25 13 27 37 22 40 15 1} {1 42 18 56 32 58 12 1} {1 28 6 5 61 16 4 30 10 50 1} {1 43 34 66 9 41 59 33 1} {1 2 8 60 65 45 64 47 26 1} {1 51 48 62 49 38 31 36 20 39 1} {1 29 17 63 11 3 7 1}
B-n67-k10	{1 48 34 14 29 1} {1 54 45 41 65 43 40 47 1} {1 3 33 66 38 24 9 42 1} {1 15 51 12 7 35 37 1} {1 61 4 58 31 62 63 64 1} {1 5 2 50 36 20 21 8 1} {1 32 19 27 67 52 60 25 30 1} {1 59 16 56 55 23 53 46 13 1} {1 6 57 22 26 49 10 1} {1 17 44 28 11 39 18 1 }
B-n68-k9	{1 65 36 61 5 32 57 2 1} {1 48 66 44 14 67 28 1} {1 54 39 56 24 40 41 51 1} {1 47 46 62 52 3 9 42 38 34 1} {1 31 60 35 45 49 10 21 53 18 16 1} {1 30 4 12 26 64 20 1} {1 50 23 58 27 1} {1 8 22 13 17 55 63 15 6 1} {1 43 29 19 25 11 59 7 33 37 68 1}
B-n78-k10	{1 42 15 37 20 14 34 5 60 36 56 62 1} {1 78 24 27 41 69 19 43 1} {1 45 12 65 54 11 26 17 67 61 1} {1 39 70 68 44 22 25 53 2 1} {1 64 76 8 33 21 55 4 66 48 1} {1 16 35 52 32 74 77 58 1} {1 38 30 71 63 7 1} {1 10 29 75 49 40 51 1} {1 57 9 13 47 46 23 72 1} {1 3 6 31 50 28 59 73 18 1}