UN ALGORITMO GENETICO MULTIMODAL Y SU APLICACION AL PROBLEMA DE RUTEO DE VUELOS CON MULTIPLES PARADAS

TESIS

DOCTORADO EN INFORMATICA

INSTITUTO TECNOLOGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

POR

EDUARDO URESTI CHARRE

MAYO, 2003

UN ALGORITMO GENETICO MULTIMODAL Y SU APLICACION AL PROBLEMA DE RUTEO DE VUELOS CON MULTIPLES PARADAS

T E S I S

DOCTORADO EN INFORMATICA

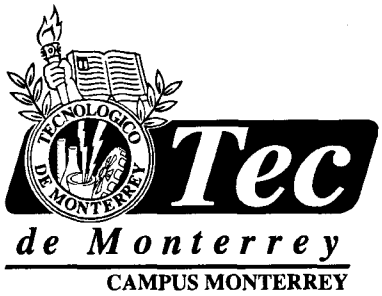
INSTITUTO TECNOLOGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

POR

EDUARDO URESTI CHARRE

MAYO, 2003

UN ALGORITMO GENÉTICO MULTIMODAL Y SU APLICACIÓN AL PROBLEMA DE RUTEO DE VUELOS CON MÚLTIPLES PARADAS



TESIS

DOCTORADO EN INFORMÁTICA
INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

POR
EDUARDO URESTI CHARRE
MAYO, 2003

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

DIVISIÓN DE GRADUADOS EN ELECTRÓNICA, COMPUTACIÓN, INFORMACIÓN Y COMUNICACIONES

PROGRAMAS DE POSGRADO EN ELECTRÓNICA, COMPUTACIÓN, INFORMACIÓN Y COMUNICACIONES

Los miembros del comité de tesis recomendamos que la presente tesis del Lic. Eduardo Uresti Charre sea aceptada como requisito parcial para obtener el grado de **Doctor en Informática**.

Comité de Tesis

Dr. Manuel Valenzuela Rendón
Asesor

Dr. Hugo Terashima Marín
Sinodal

Dr. José Luis González Velarde
Sinodal

Dr. Carlos Artemio Coello Coello
Sinodal

Dr. David Garza Salazar
Director de los Programas de Posgrado en Electrónica, Computación, Información y Comunicaciones
Mayo de 2003

Un Algoritmo Genético Multimodal y su Aplicación al Problema de Ruteo de Vuelos con Múltiples Paradas

Tesis Doctoral por

Eduardo Uresti Charre

Presentada al Programa de Graduados en Electrónica, Computación, Información y Comunicaciones.
Este trabajo es requisito parcial para obtener el grado de Doctor en Informática.

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

Mayo, 2003

Declaración

Yo declaro que este documento de tesis fue compuesto completamente por mi mismo y que describe mi propia investigación.

Eduardo Uresti Charre
Monterrey, N.L. México.
Mayo, 2003.

Dedicatoria

A mi esposa Martha Patricia y a mis hijos Eduardo, Martha Patricia, y Gabriela Alejandra. Por ser la motivación de la presente imvestigación, y de vez en cuando, sólo de vez en cuando, un obstáculo para terminarla.

Agradecimientos

Quiero dar las gracias a Manuel Valenzuela, asesor de mi investigación doctoral; por el tiempo que invirtió discutiendo ideas conmigo; su guía y sentido crítico fueron determinates para el inicio, desarrollo y terminación de la presente investigación. Esta investigación la cual no hubiera sido posible sin él.

Doy gracias a todos los miembros del comité de disertación. Gracias a José Luis González Velarde, a Hugo Terashima Marín, y a Carlos Coello Coello por su ayuda. Sus comentarios permitieron en mucho la mejora este documento de tesis.

Quiero agradecer a Francisco Javier Cantú por haberme invitado a trabajar en el Centro de Sistemas Inteligentes; por motivarme a entrar en el programa doctoral, y por el apoyo que me brindó durante su estancia como director del centro.

Agradezco la ayuda y confianza que me ha brindado Rogelio Soto, director del Centro de Sistemas Inteligentes, lo cual me ha facilitado la culminación de mi investigación.

A mis compañeros del programa doctoral debo darles las gracias y un fuerte abrazo: Leonardo Garrido, Olivia Barrón, y a Santiago Conant. Por su sincero interés y compañerismo, y porque de vez en cuando sabían que no había que preguntar por el avance de tesis.

Quiero agradecer a Tomás Sánchez, director del departamento de Matemáticas, por la ayuda que me proporcionó en la selección y horario de clases que me asignó en el tiempo de la investigación doctoral lo cual me hizo llevar una carga más ligera, así como en diversos apoyos logísticos.

Gracias a todos los profesores del departamento de Matemáticas por su amistad y apoyo.

Y por último, quiero dar las gracias a mi esposa y a mis hijos por el apoyo y cariño que me han brindado el cual es el soporte no sólo de esta investigación, sino de mi vida misma. Les pido perdón por tantos fines de semana en casa y las cenas basadas en cereal.

Resumen

En los algoritmos genéticos la población tiene un doble papel. Por un lado representa lo que puede ser el resultado entregado por el algoritmo, y por otro, la población es la materia prima para la exploración de espacio de búsqueda. Siendo la población lo que el algoritmo entrega en su finalización ésta debe ser cuidadosamente reemplazada por nuevas soluciones que mejoren lo ya encontrado y no destruyan soluciones adecuadas ya descubiertas. Si el problema consiste en encontrar la mejor solución posible en el espacio de búsqueda, es sencillo cuidar el mejor elemento a lo largo de las generaciones. Cuando la solución al problema consiste en encontrar un conjunto múltiple y diverso de puntos, la población debe modificarse con un cuidado adecuado. Este cuidado puede comprometer sustancialmente la capacidad de exploración del espacio de búsqueda. Con el propósito de mantener un nivel adecuado de diversidad en la población el aumentar el tamaño de la población aumenta sustancialmente los recursos requeridos para hacer la evolución de la población y, en el contexto de un problema donde el número de evaluaciones disponible sea limitado o costoso, también puede compromenter el nivel de exploración.

El presente trabajo propone y analiza un algoritmo genético que se aplica a problemas de optimización multimodal donde el rol de la población está separado. Este proceso de división es llevado a cabo mediante el manejo de dos poblaciones. Una primera población que hace el papel de una población memoria que representa el conjunto respuesta al problema de búsqueda. Esta población memoria es concebida como un *salón de la fama* donde se almacenan los mejores individuos encontrados en el proceso de búsqueda. El concepto de *mejor* es el resultado de una combinación entre la aptitud del individuo, una medida descriptiva de la sobrepoblación en el nicho que ocupa, y de un indicador de cómo se compara su evaluación respecto a la de sus compañeros en el nicho. La segunda población representa el medio de exploración en el espacio de soluciones. A la par con estas dos poblaciones, un mecanismo de administración es desarrollado. Este mecanismo se encargará de reemplazar elementos en la población memoria por aún mejores elementos encontrados en el proceso de búsqueda. La otra función importante a su cargo consistirá en formar las poblaciones para cada una de las nuevas exploraciones.

El algoritmo desarrollado es comparado experimentalmente con dos de los algoritmos genéticos que mejor se han desempeñado en la solución a problemas de optimización multimodal resultando con ventajas sobre ellos. Este algoritmo es aplicado al problema de ruteo de vuelos con múltiples paradas el cual es un problema de optimización discreta de alta complejidad y relevancia práctica. En este problema la evaluación de los individuos tiene un costo computacional elevado, de manera que el *overhead* computacional causado por la administración de la población memoria es reducido cuando se le compara contra los costos de evaluación.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Este capítulo presenta una visión general del presente trabajo de investigación. Primeramente, y desde una perspectiva general, aquí se mencionan los problemas que enfrenta un algoritmo genético cuando se aplica a problemas de optimización donde es fundamental la diversidad de la población entregada como solución al problema. Seguidamente, traza las ideas y desarrollo de una estrategia que pretende remediar las debilidades naturales de los algoritmos genéticos en este tipo de problemas. También, se presentan y discuten las limitaciones, metodología y los objetivos de la presente investigación. Finalmente, este capítulo concluye con una descripción capítulo a capítulo del contenido de este escrito. El tono del mismo es totalmente de expresión de ideas. Para detalles y formalidades sugerimos al lector una revisión de los capítulos correspondientes.

1.1. Algoritmos genéticos: inconvenientes en el caso multimodal

Primeramente daremos una descripción informal de los algoritmos genéticos para después comentar los problemas que enfrentan al ser aplicados a problemas donde la respuesta es un subconjunto solución del espacio búsqueda.

1.1.1. Algoritmos genéticos

Desde una perspectiva general, los algoritmos genéticos (AGs) son modelos computacionales y matemáticos de búsqueda basados en la genética poblacional. Los AGs pueden ser considerados como mecanismos de búsqueda guiados por la aleatoriedad que imitan la adaptación de los sistemas naturales a un medio ambiente, donde se enfatiza la relación directa entre el nivel de adaptación al medio de un individuo y su capacidad reproductiva.
Básicamente, un AG opera mediante la evolución de un conjunto de estructuras artificiales que desempeñan el papel de una población, o conjunto de individuos. La evolución de la población ocurre cuando nuevas estructuras reemplazan a algunos o

todos los individuos de la población. Estos nuevos elementos son construidos combinando las características de pares de individuos de la actual población, los cuales son seleccionados de acuerdo a su nivel de adaptación al medio ambiente. La producción de nuevos individuos es la forma como el AG muestrea el espacio de soluciones. Al final de su ejecución, el AG contiene una población que pretende representar los elementos mejor adaptados a su medio ambiente.
En cada iteración, la población representa o contiene la información sobre lo mejor que el AG ha encontrado a lo largo de la simulación. Simultáneamente, la información contenida en los individuos será utilizada como base para continuar la exploración del espacio de búsqueda mediante procesos recombinativos. Podemos afirmar que la población es fin y medio a la vez. Por un lado representa la respuesta que la estrategia proporciona como solución al problema, y por otro lado contiene la información utilizada como base para explorar el espacio de búsqueda del problema.

1.1.2. Retención contra exploración

En problemas como optimización multimodal o multiobjetivo la solución esperada es un conjunto de puntos en el espacio de búsqueda; en el caso del problema multimodal es el conjunto de *óptimos locales*, mientras que en el caso del problema multiobjetivo es el *frente de Pareto*. En general, tales conjuntos solución constan de más de un punto y en algunos casos podrían ser infinitos. Cuando un AG es aplicado para resolver alguno de estos problemas, debe proceder cuidadosamente al modificar la población reemplazando individuos actuales por nuevos individuos de manera tal que no sean destruidas soluciones adecuadas ya descubiertas. Pero al mismo tiempo debe mantener la habilidad de continuar el muestreo de regiones del espacio de búsqueda.

Algunos AGs han sido desarrollados para atacar el problema de la optimización multimodal y han tenido éxito parcial, notablemente entre ellos la estrategia de las funciones compartidas (*sharing*) y el amontonamiento determinístico (*deterministic crowding*). Pero algunos resultados recientes muestran que no existe una respuesta satisfactoria a una estrategia en AGs al problema de la optimización multimodal. Los resultados recientes se pueden resumir diciendo que existen problemas de retención de óptimos locales así como de convergencia a ellos.

Reconocemos la dificultad de la aplicación de AGs al problema de optimización multimodal; conservar ciertos elementos ya encontrados y continuar la exploración del espacio problema son dos acciones que compiten entre sí. También reconocemos la utilidad práctica de tener una estrategia general que permita aproximar soluciones a este tipo de problemas.

1.2. Tesis: propósito, solución, y delimitaciones

El propósito de la presente investigación consiste en desarrollar, analizar y experimentalmente probar una estrategia basada en AGs que enfrente con mejor suerte que las técnicas disponibles problemas de optimización multimodal. Por ello queremos decir

que la estrategia propuesta deberá poseer una mayor capacidad de mantener dentro de su población activa representantes de los óptimos locales encontrados, sin que ello vaya en detrimento de su capacidad de exploración del espacio de búsqueda del problema que esté siendo resuelto.

La principal idea en el desarrrollo de la estrategia desarrollada en este documento consiste en separar los mecanismos de retención de información y de exploración del espacio de búsqueda. Para ello, la estrategia propuesta contempla el manejo de dos conjuntos de poblaciones, o estructuras artificiales. Una que representa lo mejor que el proceso de búsqueda completa ha encontrado, lo cual podría ser considerado como un *salón de la fama*, y que será prácticamente la respuesta que la estrategia entrega como solución al problema. Y una segunda población, que hace las veces de una población exploradora la cual tendrá como objetivo muestrear el espacio de búsqueda. La utilización y la explotación de tales poblaciones requieren de una adecuada administración. Mecanismos de administración de lo mejor encontrado y de determinación de zonas de explotación serán desarrollados como parte esencial de la estrategia propuesta en esta investigación. Uno de estos mecanismos está dedicado al reemplazo de elementos en la población memoria por nuevos elementos recién encontrados, y el otro dedicado a la formación de la población exploradora.

Esta investigación parte del supuesto que tales mecanismos se pueden describir como mecanismos de comparación y selección con base en un indicador numérico adecuadamente definido. Para el mecanismo de reemplazo hemos formulado que debe ser un indicador numérico basado en tres informaciones que consideramos importantes. La evaluación del individuo mediante la función objetivo del elemento aparece en primer lugar en esta lista. Otra es una medida numérica del grado de sobrerrepresentatividad del individuo en la población memoria, el cual estará relacionado con lo que posteriormente llamaremos *conteo del nicho*. Y como elemento nuevo, hemos incluido un indicador referente a cómo se compara la evaluación del individuo con respecto a las evaluaciones de los otros individuos en el mismo nicho. También hemos propuesto que el indicador de selección para conformar la población exploradora es un indicador numérico basado también en tres informaciones relevantes. La evaluación del individuo, la sobrerrepresentatividad del nicho, y adicionalmente hemos incluido un delimitador para evitar la sobreexplotación del uso de un individuo.
Planteado así, la estrategia tiene un buen número de elementos a considerar que pueden impactar seriamente en su desempeño. Tamaños de poblaciones, mecanismos precisos de selección, mecanismos de trabajo en la fase se exploración, encabezan la lista de elementos a considerar.

El número de elementos en el conjunto solución a un problema propuesto es una característica del problema en sí mismo. Por consiguiente, no puede ser un parámetro preestablecido; debería ajustarse dinámicamente por la estrategia. Dos escenarios son posibles. Uno simple y limitado donde el tamaño del salón de la fama se encuentre fijo. Y otro donde éste puede autoajustarse. Por razones de simplicidad en el análisis y desarrollo hemos tratado el caso donde esta población tiene tamaño fijo.

El tamaño de la población exploradora es clave en el proceso completo. Si este número es muy grande; el esfuerzo computacional invertido en la exploración crece. Si

es muy pequeño, la velocidad con la cual se encuentran nuevos individuos disminuye en forma importante. Es fácil imaginar situaciones donde diferentes tiempos en el proceso de búsqueda de soluciones en un mismo problema y simulación tengan requerimientos diferentes en cuanto al tamaño de la población exploradora. Por consiguiente este parámetro debería modificarse a lo largo de la simulación. Esta alternativa no se ha explorado. De momento hemos centrado nuestro esfuerzo en una situación donde se mantiene fijo y hemos supuesto que es predeterminado por el usuario.

Un principio, o predilección personal, del cual partimos en esta investigación es la ventaja que pueden tener los procesos recombinativos sobre aquellos que basan su exploración en la búsqueda local alrededor de un punto en ciertos problemas. La caracterización de aquellos problemas donde una estrategia es superior a la otra es tema abierto en general. Sin embargo, hemos mantenido el interés y nuestro esfuerzo en utilizar como base en la exploración la utilización de un algoritmo genético, es decir, de una estrategia basada en la recombinación. El presente desarrollo permitiría validar o hacer confirmaciones parciales sobre lo justificado o injustificado de tal predilección. La estrategia general de búsqueda planteada en la presente investigación permitiría hacer comparaciones entre los resultados obtenidos al aplicar un algoritmo basado en recombinación y otro basado en información puntual cual se aplique el proceso de exploración. Por ejemplo, darle a la estrategia la decisión de usar un AG o bien hacer búsqueda local dependiendo de sus resultados en diferentes etapas de una misma simulación o corrida. Este trabajo no ha sido extendido en ese sentido.

1.3. Organización del escrito

El presente escrito está organizado de la siguiente forma. El capítulo 2 presenta una descripción general de los algoritmos genéticos, de cuáles son sus elementos principales, y de cómo operan en lo general. El capítulo incluye la definición del problema de optimización multimodal, así como de algunos de los intentos más exitosos de estrategias basadas en AGs para resolver problemas de este tipo. Entre las estrategias que se comentan están la estrategia de amontonamiento (*crowding*, CR), citada aquí más por su valor en el desarrollo de las ideas que por su efectividad en problemas de comparación, la estrategia de amontonamiento determinístico (*deterministic crowding*, CD), la estrategia de las funciones de compartición (*sharing*, SH), y la estrategia de nicheo secuencial (*sequential niching, SN*). Así mismo se hace una descripción de los AGs basados en dos o más poblaciones para contrastarlos con el esfuerzo de la presente investigación. En esta sección se incluye también el marco de referencia experimental comúnmente utilizado para parcialmente evaluar y comparar estrategias en esta área. Finalmente, también se plantea el problema de optimización discreta conocido como el problema de ruteo de vuelos con múltiples paradas. Este problema es incluido buscando darle al presente desarrollo un cierto matiz de aplicación. Este problema es naturalmente de optimización global y en nuestro contexto es planteado como un problema multimodal. Este problema tiene un perfil que resulta adecuado para mostrar las ventajas del presente desarrollo; independientemente de su valor práctico, el costo

de evaluar individuos es relativamente alto, y por consiguiente hacer ahorros en las evaluaciones puede ser atractivo. Nuestra estrategia pretende hacer un menor gasto en el número de evaluaciones de la función objetivo que las estrategias contra las cuales se compara.

El capítulo 3 presenta y discute el algoritmo propuesto. Primeramente, y como motivación, se presentan resultados negativos sobre CD y se discuten las razones de tal comportamiento. Estos resultados negativos se argumentan en experimentos limitados donde se muestra que la capacidad que tiene CD para mantener subpoblaciones en las vecindades de los óptimos locales está muy relacionada con la codificación y el perfil de la función objetivo. Los experimentos conducidos no tienen que ver con funciones desarrolladas en forma deliberada para hacer fracasar una estrategia, sino con variaciones mínimas al marco de referencia experimental usando en problemas de optimización multimodal. Sobre SH, se argumenta sobre su debilidad para retener también subpoblaciones, en menor cantidad de problemas pero presente, y su impacto sobre el tamaño de población requerido, y adicionalmente la baja convergencia a los óptimos locales. Seguidamente se presenta el algoritmo propuesto y se discuten las tareas de un mecanismo de administración de la información. Las dos tareas principales de la administración que se proponen son del reemplazo de individuos en la población memoria por individuos recién generados. El reemplazo se plantea como un proceso basado determinísticamente en un indicador numérico. Este indicador numérico resulta de la evaluación del individuo, penalizada por una expresión que cuantifica la cantidad de individuos en el nicho en el cual se encuentra el individuo y de cómo él se compara en evaluación respecto a ellos. El nivel de penalización se plantea como un parámetro numérico a analizar. El mecanismo de exploración se basa en la conformación de una población exploradora, la cual servirá como población inicial de un algoritmo de recombinación. Para pertenecer a esta población, los individuos de la población memoria serán seleccionados probabilísticamente con base en un indicador numérico que será el resultado de tres elementos: la evaluación, el grado de sobrerrepresentación y a un indicador que contabiliza el número de veces que el individuo ha sido seleccionado para este proceso de formación de la población exploradora. Se propone que este indicador actúa como un proceso de penalización, cuyo castigo se puede graduar mediante la introducción de un parámetro.

En el capítulo 4 se analizan las figuras de mérito dedicadas a la administración de la población memoria. Básicamente, este análisis se centra en el efecto de la selección de los parámetros de penalización definidos en las dos figuras de mérito. Primeramente se analiza el indicador de reemplazo; este análisis conduce a que el parámetro de esta figura impacta en forma directa sobre la diversidad de la población memoria. Seguidamente se analiza el impacto del parámetro que aparece en la figura de selección. En este apartado se concluye que tal parámetro impacta en forma directa en la diversidad de la búsqueda a lo largo de las regiones identificadas en memoria, sirviendo como un regulador de la exploración. El capítulo incluye experimentación para determinar valores para los parámetros de regulación o control que pueden tomarse como referencia en problemas particulares. Una de las conclusiones obtenidas es la relación entre los valores de los parámetros de control y el desempeño óptimo del algoritmo, y de la natu-

raleza propia del problema que esté siendo resuelto. Incluye asimismo experimentación a favor de la robustez de la estrategia mostrando la capacidad de retener óptimos locales y de converger a ellos. Los experimentos allí desarrollados ilustran que el algoritmo introducido con parámetros fijos mantiene su desempeño cualitativo haciendo modificaciones a un problema que ha resuelto adecuadamente. Asimismo, el capítulo incluye experimentos para determinar la relación entre el desempeño del algoritmo y el tamaño de la población exploradora.

En el capítulo 5 se presentan resultados experimentales donde se compara la presente estrategia con las estrategias de amontonamiento determinístico y el método de las funciones de compartición. Para ello, se utiliza un marco de referencia experimental basado en los problemas comúnmente usados, adicionado con algunas extensiones. Los resultados mostrados ilustran la ventaja de la estrategia desarrollada en la presente investigación. La ventaja se manifiesta en problemas cuando la función tiene un número grande de óptimos locales. La ventaja se presenta tanto en número de evaluaciones utilizadas así como en el tamaño de la población.

El capítulo 6 presenta la aplicación de la estrategia propuesta en esta investigación a un problema práctico. El problema práctico seleccionado es de optimización combinatoria y se le conoce como optimización de ruteo de vuelos con múltiples paradas. Se propone una codificación y una evaluación para poder ser atacado mediante AGs, asimismo se presentan resultados experimentales sobre la aplicación de la estrategia aquí presentada comparada contra amontonamiento determinístico y el método de las funciones de compartición. En los experimentos desarrollados, *crowding* determinístico no presenta identificación de los óptimos a ser considerados importantes, *sharing* muestra un desempeño adecuado y la estrategia propuesta muestra su competencia al utilizar una población menor que *sharing* y un número de evaluaciones ligeramente menor, todo ello manteniendo su capacidad de retener óptimos, de forma tal que existe mayor garantía de continuar el proceso de solución sin poner el riesgo de perder lo mejor de lo ya encontrado.

El capítulo 7 presenta las principales conclusiones de la presente investigación. Esta investigación avala la existencia de un esquema que a pesar de estar basado en la retención de soluciones sin que ello vaya en detrimento de su capacidad de exploración. El trabajo abre múltiples puertas hacia diferentes líneas de investigación, varias de ellas se delinean y comentan en la conclusión de este trabajo.

Capítulo 2

Temas relacionados

Los algoritmos genéticos se han planteado como una alternativa importante para resolver problemas de optimización, esto debido a la poca información que necesitan para abordar un problema específico. Esto incluye a muchas aplicaciones a problemas de optimización discreta.

Por su propia naturaleza al utilizar una población de soluciones posibles como plataforma para muestrear el espacio de búsqueda, los algoritmos genéticos se han aplicado con relativo éxito a problemas de optimización multimodal.

En este capítulo se presentan los elementos sobre los cuales recae la presente investigación. Primeramente se describirán los algoritmos genéticos, cuáles son sus componentes básicos y cuáles son sus principales variantes. Seguidamente se planteará el problema de optimización conocido como el problema de optimización multimodal. Después se hablará de algunas de las estrategias basadas en algoritmos genéticos para resolver este tipo de problemas. Se describirá el marco de referencia experimental, que es un conjunto reducido de problemas de optimización multimodal, sobre el cual se han comparado en la literatura diferentes estrategias basas en algoritmos genéticos. Por último, se formulará un problema de relevancia práctica que será un campo de prueba para el nuevo algoritmo que se propone: el problema del ruteo de vuelos con múltiples paradas.

2.1. Algoritmos genéticos

La presente sección tiene como propósito hacer una descripción en lo general sobre los algoritmos genéticos. A un lector que desee un punto de vista con mayor profundidad sugerimos la lectura de los libros de Holland [1], Goldberg [2], Mitchell [3], o Davis [4]. Para una introducción rápida sobre aspectos básicos puede consultar el tutorial de Whitley [5] sobre este tema.

Los algoritmos genéticos(AGs) fueron introducidos y desarrollados por John Holland y sus colegas en la universidad de Michigan en los años sesentas [1]. Los AGs pueden ser considerados como toda una familia de modelos computacionales que tiene como su base fundamental de operación los principios de la genética poblacional y el hecho de que operan de forma análoga al proceso de evolución y selección natural.

Entendida ésta como el proceso de evolución darwiniana donde el individuo más apto tiene una mayor probabilidad de apareo que un individuo débil, de manera que las características que le dieron supremacía tengan posibilidades de repetirse en futuras generaciones [6, 7]. Como muchas otras meta-heurísticas, los AGs tienen un enorme número de variantes, pero podemos decir que todos ellos tienen las siguientes características distintivas.

- Muestrean el espacio de posibles soluciones mediante un conjunto de puntos o *población* y no con sólo un punto.
- El muestreo del espacio de búsqueda que realizan es producido primeramente *seleccionando* individuos de la población.
- Los nuevos individuos son construidos *recombinando* partes de la información contenida en individuos actuales, los cuales pueden ser considerados como padres.
- Repetidamente modifican la población reemplazándola total o parcialmente por nuevos elementos encontrados; este concepto lo llamaremos *evolución*.

Este proceso de selección, generación y reemplazo favorece a que parte de la información contenida en los mejores individuos continúe siendo utilizada en la generación de los nuevos individuos por construir.

2.1.1. Componentes de un AG simple

En esta parte desglozaremos con cierto grado de detalle los elementos que conforman un algoritmo genético simple.

Individuos y población

En un AG, la *población* es un conjunto de individuos. Los individuos son estructuras artificiales que en general poseen tres diferentes partes: cromosoma, fenotipo y aptitud. Típicamente, el cromosoma de un individuo está representado computacionalmente por una cadena de ceros y unos, es decir por una cadena binaria. Esta alternativa de representación no es única; existen otras posibles. Desde números reales [8, 9]; alfabetos de más de dos letras [10]; hasta programas computacionales de alto nivel [11]. Los cromosomas en general son de longitud fija, aunque pueden ser de longitud variable [12]. Un cromosoma es pensado como una concatenación de subcomponentes llamados *genes*, los cuales ocurren en varias posiciones o lugares (*loci*) y toman algunos valores llamados *alelos*. El cromosoma contiene encriptada la descripción del individuo.

La característica visible del individuo es el *fenotipo*. En problemas de optimización usualmente el fenotipo de un individuo representa una posible solución al problema o un punto del espacio de búsqueda del problema. Una *codificación* es una función que asocia a cada posible cromosoma un fenotipo; es la función de *desencriptamiento* del individuo contenido en la representación genética. Para que el problema sea resuelto por un AG, esta función debe ser suprayectiva, es decir que cualquier punto posible del espacio de

búsqueda está codificado al menos en un cromosoma. Aunque en optimización sobre conjuntos continuos basta con una resolución adecuada para el espacio de búsqueda. Lo deseable es encontrar un mapeo que sea una biyección entre todos los cromosomas y todos los puntos del espacio de búsqueda del problema que se desee resolver.

Sin embargo, esto es muy difícil de conseguir. Hay dos alternativas que se pueden considerar cuando la función de codificación no es inyectiva, o precisamente sobre el espacio de búsqueda si no sobre un conjunto aun más grande que el espacio de búsqueda. En la primera de ellas existen puntos en el espacio solución que tienen varios cromosomas asociados mediante la función de codificación. En este caso se dice que ocurre una *sobrerrepresentación* de ese punto en el espacio de búsqueda. Si el nivel de sobrerrepresentación es grande, esto puede influir en la distribución en la cual se muestrea el espacio de búsqueda por parte del AG, y por consiguiente la búsqueda basada en esta codificación tiene un sesgo. No es fácil estimar el impacto de esto en la evolución de la población. En la segunda alternativa, la función de codificación produce individuos inaceptables como elementos del espacio de búsqueda. Esto puede ocurrir con frecuencia en problemas de optimización con restricciones. Dos alternativas para manejar este tipo de inconveniencia son las siguientes. Reconvertir el punto asociado a un punto que sí está en el espacio de búsqueda y una segunda consiste en modificar la evaluación de tal individuo de forma tal que tenga bajas probabilidades de recombinarse.

La evaluación es la componente restante. La evaluación de un individuo mide el nivel de adaptación de él a su medio ambiente. Salvo detalles, en problemas de maximización global la evaluación de un individuo es el valor de la función en el punto del espacio que el individuo representa. Cuando se trata de optimización con restricciones usualmente la evaluación de un individuo es la evaluación de su fenotipo penalizada por castigos que miden el cumplimiento o incumplimiento del fenotipo respecto a las restricciones [13, 14].

Inicio y evolución de la población en un AG

Normalmente un AG parte de una población generada aleatoriamente. Esto se hace usualmente generando cromosomas cuyos alelos han sido producidos aleatoriamente mediante una distribución uniforme. Esta distribución inicial podría ser alterada o afectada por la función de codificación debido a sesgos pero estos detalles no han sido extensamente investigados. Otra alternativa para una población inicial totalmente aleatoria puede consistir en que una población siga una distribución diferente; Kallel y Schoenauer [15] muestran experimentalmente que diferentes distribuciones en los ceros y unos en los cromosomas binarios pueden influir en el resultado que proporciona un AG. Otras alternativas a poblaciones iniciales podrían relacionarse con *sembrar* en la población inicial individuos bien adaptados o distribuidos en ciertas regiones del espacio de búsqueda [16, 17, 15].

Un AG simple (AGS) evoluciona una población reemplazándola totalmente por nuevos individuos en instantes de tiempo determinados llamados generaciones. Alternativamente a reemplazar el total de la población actual por nuevos individuos, está el sólo reemplazar una fracción de la misma. Cuando el reemplazo es completo, el AG

se llama *generacional.* Cuando sólo una fracción de la población es reemplazada se dice que ocurre un *traslape generacional*, al convivir en la misma generación ancestros y descendientes. El dejar individuos de previas generaciones bien adaptados al medio ambiente apresura la *convergencia* de la población. Es decir, generación a generación los individuos en la población tienden a ser semejantes entre sí cada vez más. Para ver un estudio comparativo de las respuestas entre estos dos tipos de AGs se recomienda en parte el análisis hecho por De Jong [18] o la comparación de Syswerda [19].

Selección de padres

Ls *selección* en un AG consiste en determinar aquellos individuos de la población que aparecerán, posiblemente varias veces alguno o varios de ellos, en la aportación de material genético para la exploración del espacio de búsqueda al producir nuevos individuos. En un AGS el número de individuos a seleccionar es el tamaño de la población y cada individuo tiene como probabilidad de ser seleccionado:

$$p(x) = \frac{f(x)}{\sum_{y \in Pob} f(y)}, \tag{2.1}$$

donde $f(x)$ es el valor de la función de aptitud evaluada en el individuo x. Una condición para que efectivamente p resulte en una función de probabilidad, es que las evaluaciones de los individuos sean no negativas. En caso de no serlo, una estrategia comúnmente usada es *subir* la función f una cantidad fija adecuada. Aunque este traslado no modifica los valores de x donde ocurren los óptimos, hacerlo en mayor o menor grado puede modificar sustancialmente la distribución p y por consiguiente puede afectar el comportamiento del AG.

La selección de individuos se hace mediante selecciones de un solo individuo a la vez, y en cada una de ellas participa el total de la población. Es importante decir que el individuo seleccionado sigue teniendo la posibilidad de ser seleccionado nuevamente. Este método de selección es conocido como *selección de rueda de ruleta*, debido a la analogía que presenta con el tirado de dardos a una rueda donde cada sector circular está asociado a un individuo x y el área del sector corresponde a la proporción $p(x)$. Este tipo de selección presenta mucho ruido estocástico y es preferido un método conocido como *selección universal estocástica* o SUS, el cual tiene el mismo valor esperado en la distribución de la selección de individuos en la población pero con una menor desviación [20]. La descripción visual de este método es similar a la anterior. Pero en este caso existen n marcas, donde asumimos que n es el número de individuos en la población, igualmente espaciadas alrededor de la ruleta; aquellos individuos asociados a sectores donde caigan cada una de las marcas será seleccionado. Si un individuo tiene proporcionalmente una gran evaluación, y por consiguiente el sector circular asociado a él puede resultar grande, puede ser entonces que varias marcas caigan en su sector después del *giro de la ruleta*; en este hipotético caso dicho individuo será seleccionado más de una vez; precisamente el número de marcas que caen en el sector asociado. Otras alternativas de selección han sido desarrolladas. Uno de los más utilizados en la práctica es el de *selección de torneo de tamaño k* [21, 22]. En él, la población es primeramente revuelta en forma aleatoria. Entonces se conforman conjuntos de k individuos

consecutivos. El individuo con una mejor evaluación es el individuo seleccionado de la muestra. Para estos torneos, la lista de individuos se asume circular haciendo contiguos el último y el primer elemento. Los ganadores de los *torneos* individuales conforman el *pool* de apareamiento. Observe que individuos que corresponden a máximos locales, ganarán sus torneos respectivos haciendo que en el pool de apareamiento existan k copias de ellos. Así, el valor del parámeto k influye sensiblemente en la velocidad de convergencia de la población. Debido a ello autores prefieren torneos de tamaño 2.

Selección por *Ranqueo* [23, 22] es otra alternativa de selección. La idea es la siguiente; todos los individuos en la población actual se colocan en una lista ordenada en forma decreciente de acuerdo a la función de aptitud. Posteriormente a cada individuo se le asigna un número fijo de copias de acuerdo a la posición que ocupa en la lista ordenada. Este número es calculado por una función no creciente definida sobre las posiciones. El total de elementos seleccionados debe coincidir con el tamaño de la población. La selección de esta función es decisiva en la convergencia del algoritmo.

Generación de nuevos individuos

Habiendo seleccionado el material genético base para la exploración del espacio de búsqueda, el siguiente paso es la recombinación. Los individuos seleccionados son apareados en forma aleatoria. Formadas las parejas de individuos, los futuros padres son sometidos primeramente a un proceso de *mutación*. En el caso de la representación binaria, este proceso consiste en cambiar los valores de los alelos de 0 a 1 o de 1 a 0 según el caso. Este cambio se hace usualmente con una probabilidad muy baja. Existen modelos donde diferentes posiciones del cromosoma tienen diferentes posibilidades de cambiar y otros modelos donde esta probabilidad de mutación cambia de generación a generación.

La nueva pareja de padres mutados pasa ahora a la etapa de recombinación o de *cruce*. Existen en la literatura diferentes operadores de cruce. El más sencillo de ellos es el operador de *cruce en un punto*. Este operador se puede describir de la siguiente manera. Los cromosomas de los padres son alineados a lo largo de su extensión. Un punto entre alelos es elegido al azar; el *punto de cruce*. Este punto se usa como referencia para cortar en dos partes cada uno de los cromosomas de los ancestros. Los cromosomas de los descendientes son formandos cruzando partes diferentes de los cromosomas como en la figura 2.1. Holland [1] analiza la destrucción de esquemas asociados a este tipo de cruce y este análisis es extendido a cotas más justas por Bridges y Goldberg [24]. Las nuevos individuos son decodificados y posteriormente evaluados. Se describen otros cruces y se analizan en varios puntos de la literatura [25, 26, 27].

La idea detrás del operador de cruce es combinar las características de los padres. En un buen número de aplicaciones utilizar el operador de cruce en un punto puede resultar poco significativo, al producir individuos que no contienen características de los padres. El operador de cruce afecta seriamente el éxito de la aplicación del AG en el problema en cuestión. En un gran número de casos es conveniente definir nuevos operadores de cruce que son muy dependientes del problema a resolver. En la literatura podemos encontrar un gran número de operadores específicos a la aplicación [28, 29].

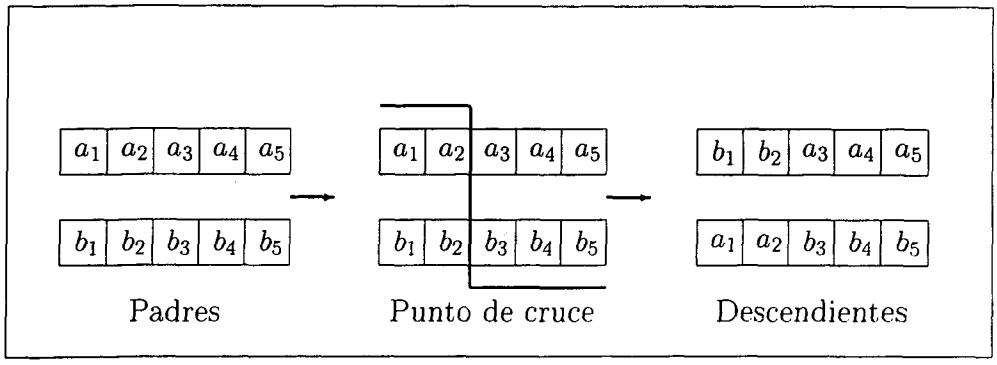


Figura 2.1: Ejemplo que ilustra el operador de cruce en un punto aplicado a dos cromosomas binarios. El punto de corte es aleatorio.

Criterios de paro de un AG

Un punto importante es cuándo detener un AG. Una primera respuesta es mediante tiempo; usualmente esto se puede definir fijando el número máximo de evaluaciones o generaciones usadas por el algoritmo.

Una segunda y más elaborada respuesta tiene que ver con la convergencia. El concepto intuitivo de convergencia tiene que ver un mínimo o nulo cambio manifiesto. El cambio puede ser medido directamente en el indicador del promedio de la evaluación de la población, o en la diversidad genética o fenotípica de la población. Algunos criterios específicos de paro pueden ser los siguientes.

- Detenerse si la diferencia entre el promedio de la evaluación de la población actual y el promedio del indicador similar de un cierto número de generaciones previas es menor que un cierto número fijo y pequeño previamente establecido.
- Detenerse si la diferencia entre el promedio de la evaluación de la población y la evaluación del mejor individuo en la población es menor que un cierto número fijo y pequeño previamente establecido.
- Detenerse si el promedio de las distancias entre los elementos de la población es menor que un cierto número fijo previamente establecido.

Habiendo definido los elementos relativos a un AG una descripción en seudocódigo para un AGS se encuentra en la figura 2.2.

2.1.2. AGs célebres

El modelo de AG básico utilizado en problemas de optimización es un AGS donde generación a generación el mejor individuo es determinado y comparado contra el mejor individuo de la generación previa para retener siempre el mejor. Otros AG que han probado su éxito en problemas de optimización son Genitor y CHC. Genitor [30] es un

1. Construya una población aleatoria de n individuos.
2. Mientras que la población no cumpla el criterio de paro haga:
   a) seleccione n individuos proporcionalmente a su evaluación,
   b) aparee en forma aleatoria los individuos seleccionados,
   c) aplique los operadores de mutación y cruce a cada par de individuos para generar dos indiviudos ,
   d) constituya la nueva población con la totalidad de los individuos generados.

Figura 2.2: Descripción general de los pasos de un algoritmo genético simple.

AG no generacional desarrollado por D. Whitley donde sólo un individuo nuevo es producido en cada generación; dicho elemento es añadido a la población y posteriormente el peor elemento de ella es borrado. Este nuevo individuo se produce escogiendo dos padres mediante selección por ranqueo, y recombinando sus características mediante los operadores de cruce y mutación.

CHC [31] es un AG no generacional desarrollado por L. Eshelman que tiene numerosas variantes. Sistemáticamente mantiene en la población al mejor individuo encontrado. El proceso de selección es totalmente aleatorio. El cruce entre individuos es posible sólo cuando los individuos sean suficientemente diferentes entre sí. El mecanismo de cruce utilizado en CHC es el cruce uniforme. En la evaluación de CHC, se hace una supervisión de la población para detectar cuándo los elementos de ella son muy parecidos entre sí, en cuyo caso individuos aleatoriamente generados entran a la población reemplazando a un buen número de los peores elementos.

AGs más específicos a problemas también han sido desarrollados. Entre sus elementos distintivos están sus procedimientos de mutación particulares y sus operadores de cruce. En lo general podemos decir que mucho del éxito de la aplicación de un AG en un problema depende de encontrar la codificación adecuada, o los operadores de mutación y cruce convenientes.

2.1.3. Detrás del manejo de cromosomas

La teoría que subyace detrás de la búsqueda en un AG involucra el concepto de *esquemas*. En el caso particular de cromosoma binarios, un esquema es una cadena sobre el alfabeto $\{0, 1, *\}$. Un esquema es la representación de un conjunto de cromosomas: precisamente aquellos que *coinciden* con el patrón definido por el esquema. El elemento $*$ corresponde a un comodín: 0 ó 1. Así por ejemplo, en el caso de cromosomas binarios de longitud 3, el esquema $0 * 1$ representa el conjunto $\{001, 011\}$, mientras que el esquema $1 * *$ representa el conjunto $\{100, 101, 110, 111\}$.

La población representa un muestreo de algunos esquemas particulares; los procesos de selección y recombinación permiten que los nuevos individuos sean elementos de esquemas de un subgrupo de éste y otros esquemas nuevos. La idea principal detrás del AG consiste en muestrear mayormente aquellos esquemas que tienen en promedio una evaluación mayor debido a la muestra en la población, sobre esquemas poco afortunados. Este proceso hace que la producción de nuevos hijos se convierta en la generación de individuos de ciertos esquemas representados ya en la población.

2.2. Optimización multimodal

Complementario al problema del optimización global de encontrar el valor x en el dominio de una función, donde ésta toma su mayor o menor valor, conocido como el problema de optimización global, tenemos el problema de optimización *multimodal*. El objetivo para este tipo de problemas consiste en encontrar todos los *óptimos locales de una función*. Sea S un espacio de búsqueda sobre el cual se encuentra definida una cierta métrica $d : S \times S \rightarrow \Re$, es decir una función que satisface los axiomas de una distancia. A saber, que la distancia entre dos elementos de S cualesquiera es siempre positiva. Que la función d es simétrica, es decir que la distancia de un punto a otro es la misma que la distancia del segundo al primer punto. Que d satisface el axioma de la desigualdad triangular, es decir que la distancia entre dos puntos es siempre menor o igual a la suma de las distancias de ellos a un tercer punto. Asumimos también definida sobre S una función $f : S \rightarrow \Re$ a maximizar. Un punto x_o en S se dice *máximo local*, si existe una vecindad del punto x_o donde no existe otro punto de S que tenga una evaluación mejor que la que tienen x_o. Formalmente esto quiere decir que existe un número positivo $\delta > 0$, tal que para cualquier otro punto y del espacio de búsqueda S que satisfaga $d(y, x_o) < \delta$ debe cumplir $f(y) \leq f(x_o)$. Mientras que x_o se dice un *mínimo local* si $d(y, x_o) < \delta$ implica $f(x_o) \leq f(y)$. El concepto de *óptimo local* se asocia a máximo local o mínimo local si el problema es de maximización o minimización respectivamente. El problema de optimización multimodal asociado a un espacio S, una métrica d y una función f consiste en encontrar todos los óptimos locales de la función f en el espacio S. Existen otras variantes al problema propuesto.

- Encontrar todos los óptimos globales.
- Encontrar un óptimo global.
- Encontrar un óptimo local.
- Encontrar todos aquellos óptimos locales cuya evaluación se encuentre dentro de una banda predeterminada por la evaluación de un óptimo global.

2.3. AGs en optimización multimodal

Los AGs tienen un primer punto a favor para atacar este tipo de problemas. Esto es debido a que en general la solución a ellos consiste en un conjunto de puntos y los AGs

trabajan con una población de individuos que representa un conjunto de puntos en el espacio de búsqueda. Un AG utilizado en problemas de optimización global en general no resuelve el problema general de encontrar el conjunto de óptimos locales. Esto se debe a que un AG exhibe un comportamiento conocido como *genetic drifting*. Esto se manifiesta en la convergencia de la población entera a un único óptimo local; la población tiende a poseer un número creciente de copias de uno de estos individuos, dando al traste con la diversidad esperada como solución. Las principales razones por las cuales un AG pierde soluciones o subsoluciones son el mecanismo de selección y los operadores genéticos. El mecanismo de selección de individuos aplica una *presión selectiva* a la población favoreciendo individuos para formar el *pool* de apareamiento sólo por su evaluación. Una medida de esta presión selectiva es el concepto denominado *take-over time*, o tiempo en el cual la población se convertiría en copias de un solo óptimo global. Este mecanismo de selección tiene asociado un cierto ruido estocástico intrínseco a él. La presión selectiva está relacionada a la cantidad de oportunidades reproductivas asignadas a un individuo debido a su evaluación. Ella depende del mecanismo de selección utilizado.

Presentaremos ahora algunos de los intentos que pretenden promover la diversidad en la solución esperada.

2.3.1. *Crowding*

De Jong [32] propuso un AG no generacional llamado *crowding* (CR). Este tipo de algoritmo es del tipo estado estable en el que sólo parte de la población puede ser reemplazada por nuevos individuos. En este algoritmo sólo parte de la población se reproduce. Un cierto número de individuos, determinados por un parámetro fijo llamado brecha generacional (*generational gap*), G_g, es elegido mediante selección proporcional usando como evaluación la función a optimizar. Este conjunto de individuos es apareado al azar y sobre los pares se utiliza mutación y recombinación. Los individuos nuevos así generados reemplazarán a un igual número de individuos en la población actual. El criterio de reemplazo es el siguiente. Para cada uno de los nuevos individuos se escogen al azar entre la población un cierto número fijo de elementos, llamado factor de amontonamiento (*crowding factor*), CF. Entre ellos, aquél que esté lo más cercano al nuevo elemento será reemplazado por él. La cercanía es medida utilizando la distancia de Hamming a nivel del cromosoma binario. La *distancia de Hamming* entre dos cromosomas binarios es el número de alelos en los cuales los dos cromosomas no coinciden. CR intenta promover la competencia entre similiares. Una descripción en seudocódigo aparece en la figura 2.3.

Goldberg y Deb [33] probaron experimentalmente que CR es incapaz de mantener en la población individuos en cada una de la vecindades de los óptimos locales de problemas de optimización conocidos; la población presentaba a lo largo de las generaciones un fenómeno de convergencia hacia uno de los picos de la función.

1. Genere una población aleatoria con n elementos, P.
2. Mientras que la población no cumpla el criterio de paro haga:
   - *a)* mediante selección proporcional a la evaluación escoja $CG \cdot n$ elementos de la población,
   - *b)* aparee aleatoriamente los elementos seleccionados,
   - *c)* a cada pareja formada aplique mutación y cruce para obtener 2 elementos,
   - *d)* para cada individuo nuevo x, haga:
     - escoja CF individuos en la población, $S_x \subset P$
     - escoja de S_x el individuo *más cercano* a x, digamos y_x.
     - x reemplaza a y_x.

Figura 2.3: Descripción general del algoritmo de *crowding*.

2.3.2. El método de las funciones de compartición

Otra de la estrategias basadas en AGs para problemas multimodales es una desarrollada por Goldberg y Richardson [33] conocida bajo el nombre del método de las funciones de compartición o simplemente *sharing, SH*. SH intenta promover el concepto de nicho ecológico. Para ello, fuerza a que individuos similares *compartan* recursos o evaluación. Esto es computacionalmente implementado decrementando la función de adaptación al medio mediante un factor positivo y mayor o igual a uno, que mide aproximadamente el número de individuos de la población que se encuentran cercanos al individuo a evaluar. La idea que se persigue es que individuos en la misma vecindad ecológica deberían compartir recursos. Una sobrepoblación en un nicho implicaría una evaluación modificada reducida para los individuos en el nicho. El método de las funciones de compartición, como es presentado en [33, 34], utiliza un AG generacional que distribuye las oportunidades reproductivas mediante selección proporcional usando una evaluación ajustada. La evaluación ajustada $f'(x)$ de un individuo se determina mediante la siguiente fórmula:

$$f'(x) = \frac{f(x)}{N(x)}, \tag{2.2}$$

donde f es la función a optimizar y $N(x)$ es una medida que estima la cantidad de individuos en el nicho al cual pertenece x en la población actual. La fórmula para este factor es:

$$N(x) = \sum_{y \in \text{Población}} sh(d(x,y)), \tag{2.3}$$

donde

$$sh(d) = \begin{cases} 1 - \left(\frac{d}{\sigma_{share}}\right)^a, & \text{si } d < \sigma_{share} \\ 0, & \text{en otro caso.} \end{cases} \quad (2.4)$$

El parámetro σ_{share} es una estimación del radio del nicho al cual pertenecen los elementos. En el desarrollo de la estrategia de las funciones de compartición está implícita la suposición que los nicho ecológicos son todos del mismo radio. La función *sh* es una función que penaliza más a elementos que están muy cerca unos de otros, es decir aquellos que tienen una distancia entre sí pequeña, mientras que elementos que están relativamente retirados no se penalizan mutuamente entre sí. En la fórmula de *sh*, la constante a sirve para regular el castigo que elementos en un mismo nicho se provocan uno a otro. Usualmente se toma el parámetro a como 1.0 ó 2.0.

La figura 2.4 muestra el perfil de la forma como la proximidad entre dos elementos puede influir en el decremento de la función objetivo. El parámetro σ_{share} se puede estimar con alguna facilidad si se tiene un estimado del número de óptimos locales de la función problema q y cuando el espacio de búsqueda es un hipercubo en $\Re^p$. Con estos datos el radio de compartición se puede estimar mediante la fórmula desarrollada por Deb y Goldberg

$$\sigma_{share} = \frac{\sqrt{\sum_{k=1}^{p} \left(x_{k,\text{máx}} - x_{k,\text{min}}\right)^2}}{2\sqrt[p]{q}}. \quad (2.5)$$

Dentro de ciertos márgenes se puede afirmar que la población, en el transcurso de las generaciones, se va a distribuir en las vecindades de los óptimos locales en número en una forma proporcional a la altura del pico del óptimo local. Para ser precisos, si el pico x es en evaluación r veces mayor que el pico y, entonces el número de individuos en la vecindad del pico x debería ser r veces mayor que el número de individuos en la vecindad del pico y.

2.3.3. Nicheo secuencial

Alternativamente a encontrar todos los óptimos locales en la población que está siendo evolucionada, es decir en paralelo, está la posibilidad de encontrarlos *secuencialmente*. El principal problema subyacente en encontrarlos secuencialmente es que el algoritmo puede repetidamente encontrar un óptimo local previamente encontrado y despreciar otros óptimos o subóptimos con evaluación menor. Una posible estrategia es tomar en cuenta los óptimos ya encontrados y reorientar la búsqueda lejos de éstos: Beasley, Bull, y Martin desarrollaron esta idea [35]. Para alejar la búsqueda de óptimos ya encontrados, deprecian la función objetivo por una serie de factores entre cero y uno que se decrementan al aproximarse a los óptimos encontrados. La idea de nicheo secuencial (SN) se describe en seudocódigo en la figura 2.6. Los factores que decrementan la función objetivo tienen la forma:

$$G(x, s_n) = \begin{cases} (d(x, s_n)/r)^a & \text{si } d(x, s_n) < r \\ 1 & \text{en otro caso} \end{cases} \quad (2.6)$$

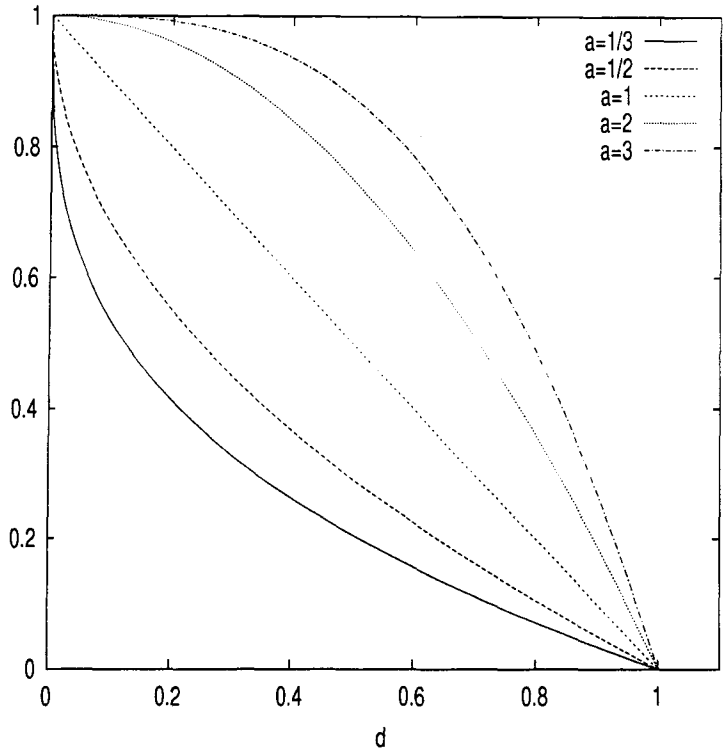


Figura 2.4: Gráficas de las funciones de compartición para los valores del parámetro a 1/3, 1/2, 1, 2, y 3.

1. Genere una población aleatoria con n elementos, P.
2. Mientras que la población no cumpla el criterio de paro haga:
   - *a*) para cada individuo x en la población calcule $N(x)$,
   - *b*) para cada individuo x, determine $f'(x) = f(x)/N(x)$,
   - *c*) mediante selección proporcional sobre f' escoja n elementos de la población,
   - *d*) aparee aleatoriamente los elementos seleccionados,
   - *e*) a cada pareja formada aplique mutación y cruce para obtener 2 descendientes,
   - *f*) la totalidad de los descendientes forman la nueva población.

Figura 2.5: Descripción de los pasos que sigue el algoritmo de las funciones de compartición o *sharing*.

1. Fase de inicialización:
   a) $optimos = \{\}$,
   b) $M_0(x) = f(x)$,
   c) $i = 0$.
2. Mientras que la población no cumpla el criterio de paro haga:
   a) ejecute un AG simple usando la función objetivo $M_i(x)$ pero manteniendo el mejor individuo encontrado a lo largo de todas las generaciones,
   b) añada el mejor individuo encontrado, s_i, al conjunto de óptimos:
      - $optimos = optimos \cup \{s_i\}$
   c) defina $M_{i+1}(x) = M_i(x)\, G(x, s_i)$, tome $i = i + 1$ y continúe el ciclo.

Figura 2.6: Descripción de los pasos del algoritmo de nicheo secuencial.

En estos factores, r es el radio de los nichos por determinar, y su forma que depende del parámetro a se ilustra en la figura 2.7 para los valores $a = 0.5, 1, 2, 4$ y 8. Uno de los problemas potenciales que puede traer el uso de SN es la creación de falsos óptimos locales presentes en la función ajustada y ausentes en la función a optimizar. Por ejemplo, si la función obetivo es $f(x) = \text{sen}^2(2\,pi\,x)$, los parámetros utilizados son $r = 0.25$, $a = 2.0$, y se ha encontrado primeramente el primer óptimo local $x = 0.25$ entonces la función objetivo que optimiza la segunda iteración es la función que tiene como gráfica lo que ilustra la figura 2.8. En general los óptimos falsos producidos pueden decrementarse tomando valores de a mayores, pero según declaraciones de los autores *no es posible desaparecerlos totalmente.* Sobreestimaciones o subestimaciones del radio de los nichos también puede traer problemas en la creación de falsos óptimos.

2.3.4. *Crowding* determinístico

Desarrollado por Samir Mahfoud [36]. *Crowding* determinístico es una variante de *crowding* tal y como la formuló inicialmente De Jong [32]. Como se comentó previamente *crowding* no es capaz de mantener subpoblaciones alrededor de los óptimos como se demuestra experimentalmente en [33]; Mahfoud [36] plantea que esta deficiencia se debe a que existen errores de reemplazo en la selección del individuo reemplazado. lo cual conlleva, en la evolución del algoritmo genético, a la pérdida de subpoblaciones en algunos nichos. Para corregir estos reemplazos equivocados Mahfoud [36] propone no muestrear el espacio de búsqueda, sino más bien buscar que el reemplazo por

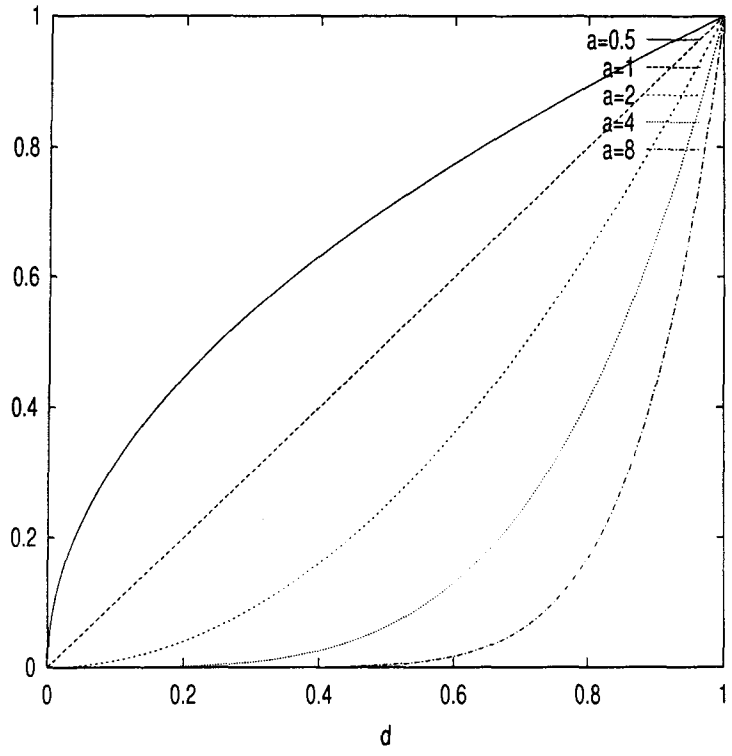


Figura 2.7: Gráficas de los perfiles de decremento usados en SN para diversos valores del parámetro a.

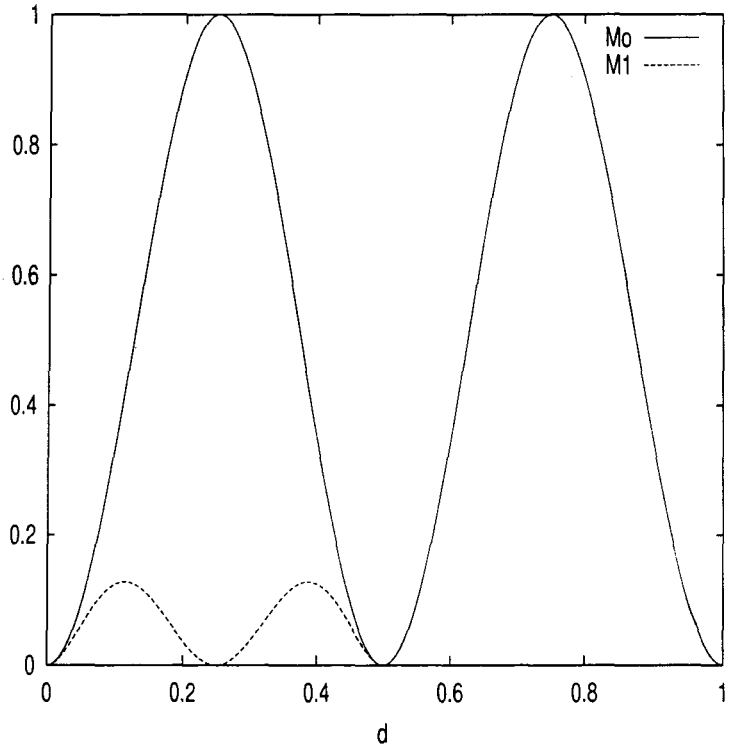


Figura 2.8: Gráficas de la función problema $M_0(x) = f(x) = \text{sen}^2(2\,\pi\,x)$ y de la función a optimizar por SN, M_1, después de haber encontrado el primer óptimo local. Note la aparición de óptimos locales que la función del problema inicial no poseía.

1. Construya una población aleatoria con n individuos
2. Mientras la población no satisfaga el criterio de paro hacer
   a) aparee aleatoriamente la población,
   b) aplique mutación y cruce a cada pareja formada para obtener dos descendientes,
   c) realice entre cada par de individuos en la población p_1 y p_2 y sus descendientes s_1 y s_2
      1) calcule
      $$d_1 = d(p_1, s_1) + d(p_2, s_2)$$
      $$d_2 = d(p_1, s_2) + d(p_1, s_2) \text{ ,}$$
      2) si $d_1 \leq d_2$, realice la siguientes dos competencias por permanecer en la población para la próxima generación:
         - si $f(s_1) > f(p_1)$, s_1 reemplaza p_1,
         - si $f(s_2) > f(p_2)$, s_2 reemplaza p_2.
      3) si $d_2 < d_1$, realice la siguientes dos competencias por permanecer en la población para la próxima generación:
         - si $f(s_1) > f(p_2)$, s_1 reemplaza p_2,
         - si $f(s_2) > f(p_1)$, s_2 reemplaza p_1.

Figura 2.9: Descripción de los pasos en el algoritmo de *crowding* determinístico.

un individuo nuevo ocurra por alguno de los padres. El algoritmo resultante, llamado *crowding determinístico* (CD), no es uno dirigido por la selección (de hecho cada individuo tiene una oportunidad reproductiva y las parejas son formadas aleatoriamente) sino más bien uno orientado a la sobrevivencia: el hijo reemplazará uno de los padres si acaso tiene una mejor evaluación. Si f es la función a maximizar, n es el tamaño de la población, y d es la función de distancia entre dos individuos, ya sea fenotípica o genotípica, *crowding* determinístico está formulado en el seudocódigo de la figura 2.9.

2.4. Marco de referencia experimental

Ante la dificultad de hacer análisis rigurosos sobre el desempeño general de un algoritmo propuesto, es común realizar pruebas estadísticas sobre el desempeño de algoritmos sobre un cierto conjunto de problemas de dificultad caracterizada. El conjunto de problemas encontrados en la literatura cuando se aplican AGs a problemas de optimización multimodal incluye las siguientes funciones.

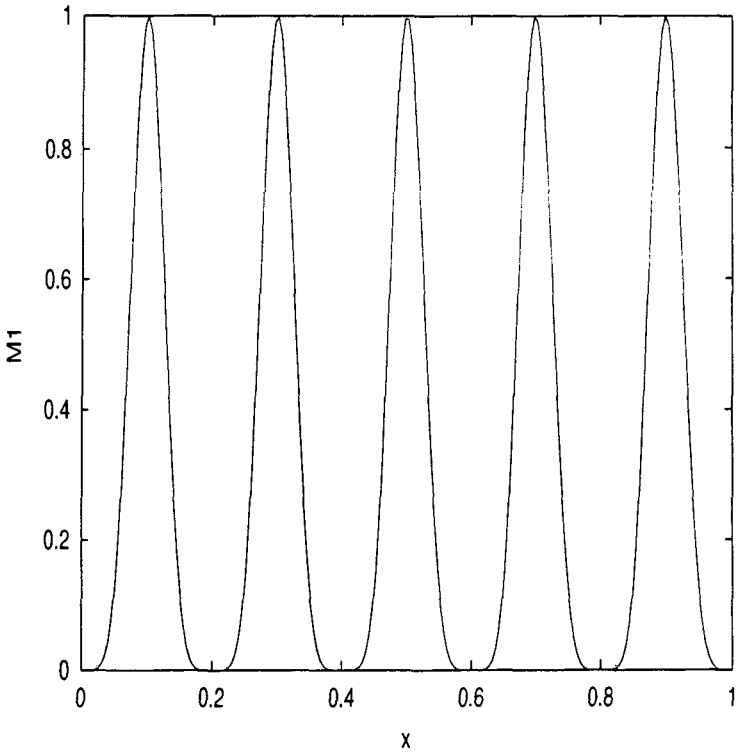


Figura 2.10: Gráfica de la función M_1.

2.4.1. Funciones senoidales: M_1 a M_4

Las funciones comúnmente referidas en la literatura como M_1, M_2, M_3 y M_4 fueron utilizadas por primera vez por Goldberg y Richardson [34] y han sido posteriormente utilizadas como un primer marco de referencia experimental [35, 34, 33, 36]. A pesar de la simplicidad de las mismas suelen referenciarse debido a que han sido utilizadas ampliamente en la literatura lo que permite hacer comparaciones entre diferentes algoritmos. Asimismo han permanecido como un primer marco de referencia debido a que AGs tradicionales convergen a un solo óptimo en ellas. En las funciones M_1-M_4, la variable x está restringida al intervalo $[0, 1]$ y se encuentra codificada mediante un cromosoma binario de longitud 30. La función de codificación toma el cromosoma, de él obtiene el número binario asociado y posteriormente es dividido entre el número $2^{30} - 1$. La función M_1 está definida por la ecuación 2.7.

$$M_1(x) = \text{sen}^6(5\,\pi\,x). \tag{2.7}$$

Los óptimos locales de M_1 están igualmente espaciados en $[0, 1]$ y todos tienen altura 1. La gráfica de M_1 se muestra en la figura 2.10. La función M_2 con ecuación 2.8 tiene sus picos prácticamente uniformemente espaciados pero de altura decreciente. Los óptimos locales están localizados en 0.100, 0.299, 0.498, 0.698 y 0.897, con alturas 1.000, 0.917,

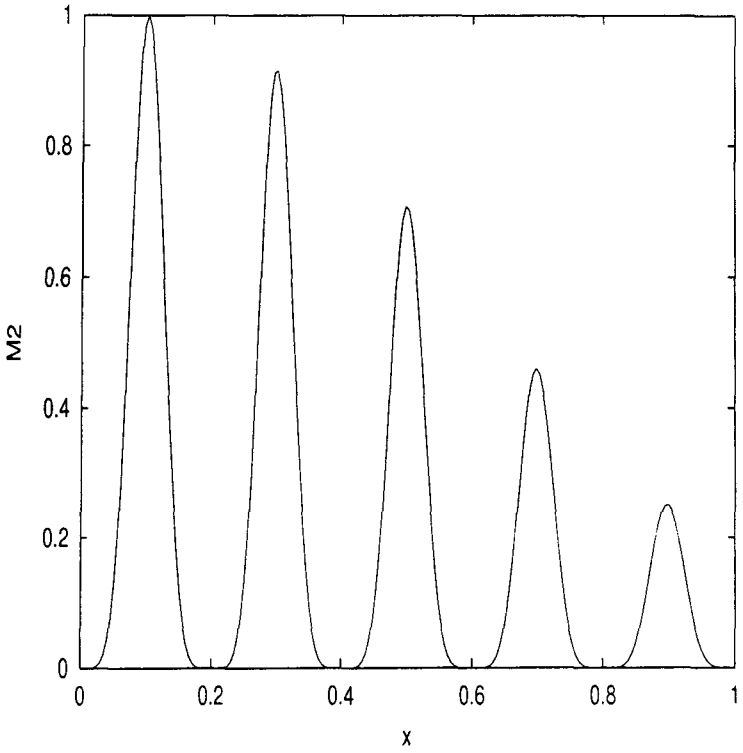


Figura 2.11: Gráfica de la función M_2.

0.707, 0.459, y 0.250 respectivamente. La gráfica de M_2 se ilustra en la figura 2.11.

$$M_2(x) = e^{-2\ln(2)\left(\frac{x-0.1}{0.8}\right)^2} \operatorname{sen}^6(5\,\pi\,x). \tag{2.8}$$

Contrario a M_1 y a M_2, M_3 no tiene sus óptimos locales uniformemente espaciados. Ellos son 0.079, 0.246, 0.450, 0.681, y 0.933 aproximadamente. Cada uno de los óptimos tiene por evaluación 1.0. La fórmula para M_3 aparece en la ecuación 2.9 y la gráfica aparece en la figura 2.12. Cabe hacer notar que la función M_3 tiene 6 óptimos locales realmente; el valor $x = 0.0$ corresponde al óptimo que no aparece listado.

$$M_3(x) = \operatorname{sen}^6\left(5\,\pi\,\left(x^{0.75} - 0.05\right)\right). \tag{2.9}$$

Complementando a las funciones M_1-M_3, la función M_4 tiene sus óptimos desigualmente espaciados y con alturas diferentes. Los óptimos de M_4 los alcanza en 0.0796, 0.2462, 0.4494, 0.6791, y 0.9301 aproximadamente, y tienen por evaluación 1.00, 0.948, 0.770, 0.503, y 0.250 respectivamente. La fórmula que define a M_4 aparece en la ecuación 2.10 y la gráfica se despliega en la figura 2.13. Igual que en la función M_3, la función M_4 tiene 6 óptimos locales ; $x = 0.0$ es el óptimo faltante.

$$M_4(x) = e^{-2\ln(2)\left(\frac{x-0.08}{0.854}\right)^2} \operatorname{sen}^6\left(5\,\pi\,\left(x^{0.75} - 0.05\right)\right). \tag{2.10}$$

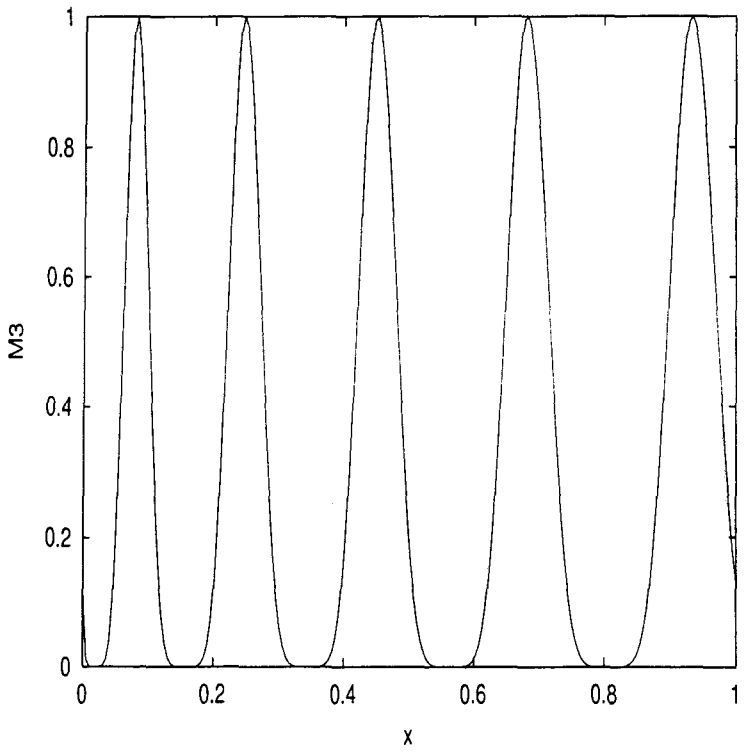


Figura 2.12: Gráfica de la función M_3.

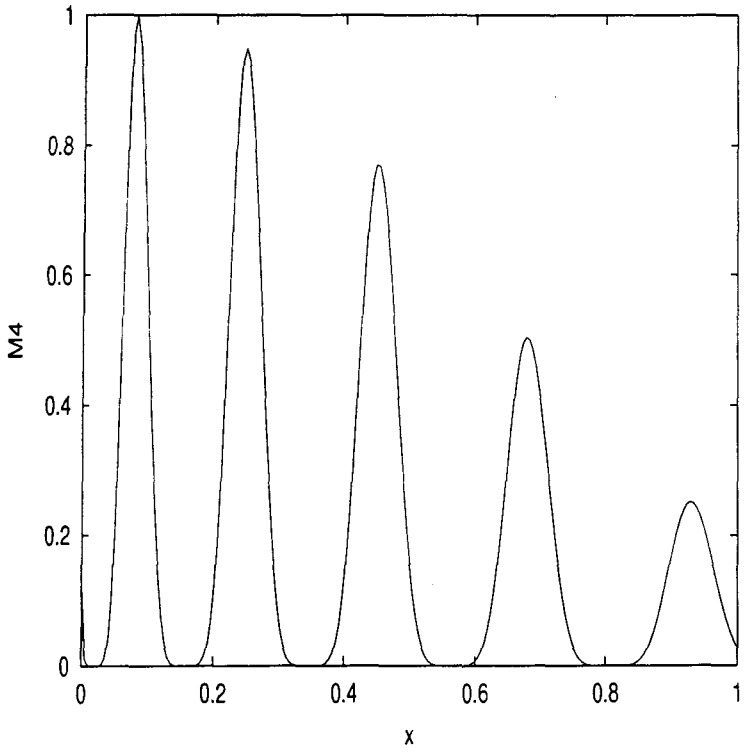


Figura 2.13: Gráfica de la función M_4.

	Óptimos locales				
M_1	$(0.100, 1.000)$	$(0.300, 1.000)$	$(0.500, 1.000)$	$(0.700, 1.000)$	$(0.900, 1.000)$
M_2	$(0.100, 1.000)$	$(0.299, 0.917)$	$(0.498, 0.707)$	$(0.698, 0.459)$	$(0.897, 0.250)$
M_3	$(0.079, 1.000)$	$(0.246, 1.000)$	$(0.450, 1.000)$	$(0.681, 1.000)$	$(0.933, 1.000)$
M_4	$(0.079, 1.000)$	$(0.246, 0.948)$	$(0.449, 0.770)$	$(0.679, 0.503)$	$(0.930, 0.250)$

Tabla 2.1: Óptimos para las funciones M_1 a M_4.

La información sobre los óptimos locales está sumarizada en la tabla 2.1. Estos valores han sido calculados utilizando el software llamado *Mathematica* con 8 dígitos de precisión y difieren ligeramente de los reportados en la literatura. Se han corroborado estas cifras utilizando la versión 2.2 y 4.0 de *Mathematica* y el software computacional llamado *Maple*. Las diferencias son muy pequeñas. Nuevamente comentamos la falta del valor $x = 0$ dentro de la lista de óptimos locales para M_3 y M_4.

2.4.2. La función de Himmelblau M_5

La función M_5 que tiene como fórmula la ecuación 2.11 es la versión modificada por Mahfoud [36], la cual es a su vez una modificación a la función de Himmelblau definida por Deb [34]. Esta función tiene dos variables las cuales son restringidas al intervalo $[-6, 6]$. Esta función tiene 4 óptimos locales los cuales tienen altura 1.0. Estos óptimos ocurren en los puntos $(3.0, 2.0)$, $(3.584, -1.848)$, $(-3.779, -3.283)$, y $(-2.805, 3.131)$. En la codificación empleada un punto (x, y) en el espacio de búsqueda es codificado en un cromosoma de longitud 30, correspondiendo 15 bits a cada variable. Las variables son codificadas linealmente: los cromosomas son convertidos en el número binario que los representa, posteriormente se divide entre $2^{15} - 1$, el número obtenido, un real en $[0, 1]$ es multiplicado por 12.0 y al resultado se le resta 6.0. Una idea gráfica de M_5 aparece en la figura 2.14.

$$M_5(x, y) = \frac{2186 - (x^2 + y - 11)^2 - (x + y^2 - 7)^2}{2186}. \tag{2.11}$$

2.4.3. La función de Sekel modificada M_6

La función de Sekel aparece reportada en la tesis doctoral de De Jong [32]. Esta es una función en dos variables reales las cuales están restringidas al intervalo $[-65.536, 65.535]$. Al aplicar un algoritmo genético a este problema, las variables se codifican en un cromosoma de 34 bits de longitud, correspondiendo 17 bits a cada variable. El proceso de codificación es lineal. Esto es, primeramente el subcromosoma es convertido al número binario correspondiente, es dividido entre el número $2^{17} - 1$ y posteriormente es convertido linealmente en un punto del intervalo $[-65.356, 65.535]$; cero corresponde al extremo inferior y 1.0 al superior. La gráfica 2.15 tiene una imagen

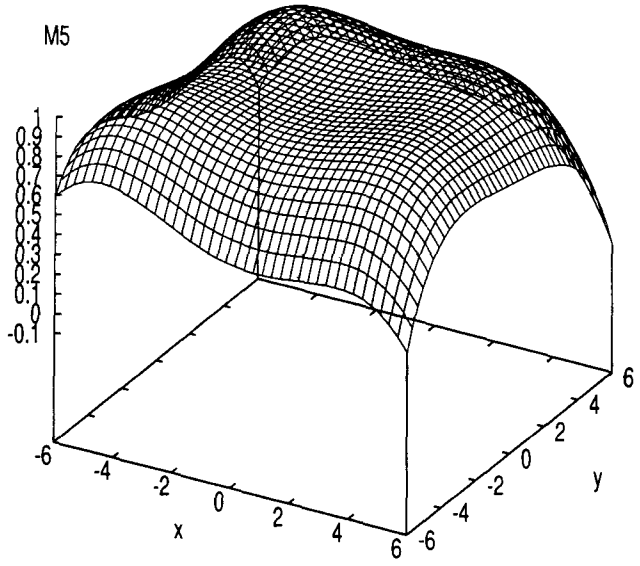


Figura 2.14: Gráfica de la función modificada de Himmelblau M_5.

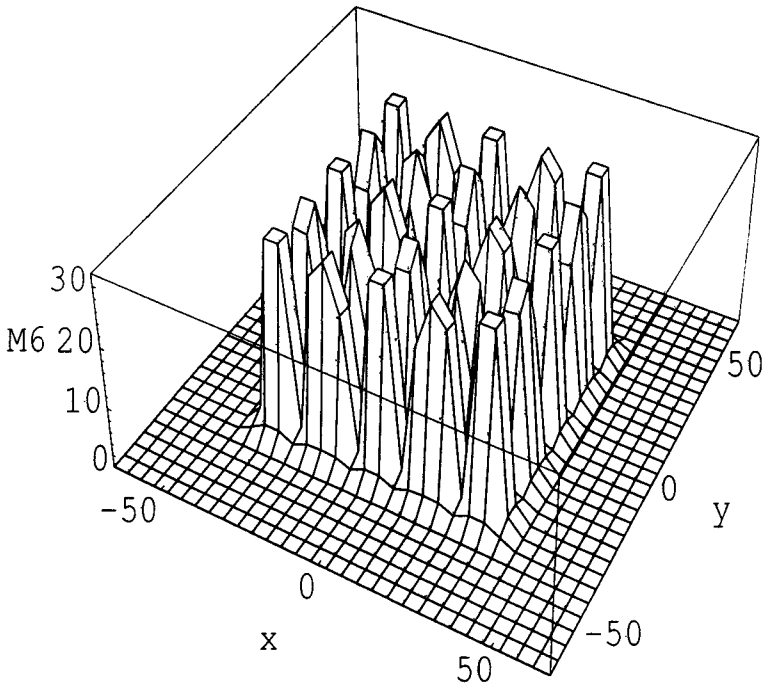


Figura 2.15: Gráfica de la función de Sekel modificada M_6.

aproximada de M_6 y la fórmula que la define aparece en la fórmula 2.12. En dicha fórmula $a(i) = 16 \times ((i \bmod 5) - 2)$ y $b(i) = 16(\lfloor i/5 \rfloor - 2)$. M_6 tiene 25 óptimos locales los cuales se encuentran en los puntos $(16\,i, 16\,j)$ donde i y j representan números enteros entre -2 y 2 inclusive. Las evaluaciones de dichos óptimos están entre 476.191 y 499.002.

$$M_6(x,y) = 500 - \frac{1}{0.002 + \sum_{i=0}^{24} \frac{1}{1 + i + (x - a(i))^6 + (y - b(i))^6}}. \qquad (2.12)$$

2.4.4. Funciones altamente multimodales: M_7 y M_8

Goldberg, Deb y Horn [37] construyeron funciones masivamente multimodales y engañosas de las cuales un caso particular es M_7. El dominio de M_7 es la totalidad de los cromosomas binarios de longitud 30. M_7 evaluada en un cromosoma c es el resultado de la suma de cinco subfunciones idénticas que actúan sobre cinco partes del cromosoma original c, dichas partes o subcromosomas se obtienen dividiendo el cromosoma en cinco partes iguales de 6 bits cada una de ellas. La subfunción f utilizada opera

$unos(c')$	$f(unos(c'))$
0	1.0
1	0.0
2	0.360384
3	0.640576
4	0.360384
5	0.0
6	1.0

Tabla 2.2: Valores de la subfunción utilizada para el cálculo de M_7. Esta función se aplica al conteo de unos en el subcromosoma designado por c'.

sobre un cromosoma de longitud 6, o subcromosoma si se quiere ver como parte del original, y su valor es calculado primeramente contando el número de unos presentes en dicho subcromosoma y posteriormente asociando a este número un valor que depende del resultado de este conteo. Los valores de dicha función aparece en la tabla 2.2 y la versión continua de ella aparece en la gráfica 2.16. La función M_7 posee 32 óptimos globales de altura 5, 20×5 óptimos locales de altura $4 + 0.640$, y además de otra gran cantidad de óptimos locales hasta completar una totalidad de $5,153,632$ óptimos entre locales y globales. La función M_8 es casi la misma que M_7 salvo que está *exponencialmente escalada.* Este escalamiento hace que los óptimos locales caigan rápidamente. La fórmula para la definición de M_8 se encuentra en la fórmula 2.13. A pesar de que M_8 sigue teniendo los mismos óptimos locales y globales ya no es engañosa.

$$M_8(c) = 5\left(\frac{M_7(c)}{5}\right)^{15}. \tag{2.13}$$

2.5. El problema de ruteo de vuelos con múltiples paradas

La literatura atestigua un gran interés por la aplicación de AGs en el contexto de la optimización discreta. La tesis doctoral de Levine [14] desarrolla un AG paralelo para resolver el problema conocido como el problema de la partición de un conjunto (*set partition problem*), la disertación de Hjorring [38] está relacionada con la aplicación de AGs al problema de ruteo de vehículos. Beasley y Chu plantean un algoritmo genético para el problema de la cubierta de un conjunto (*set covering problem*)[39]. El problema conocido como *job shop scheduling* ha sido ha sido atacado mediante AGs por Fang y colaboradores [40], por citar algunos ejemplos. La presente investigación tiene la intención indirecta de aplicar el algoritmo multimodal desarrollado a un problema de optimización discreta más que desarrollar un algoritmo específico. El problema seleccionado fue el problema de ruteo de vuelos con múltiples paradas.

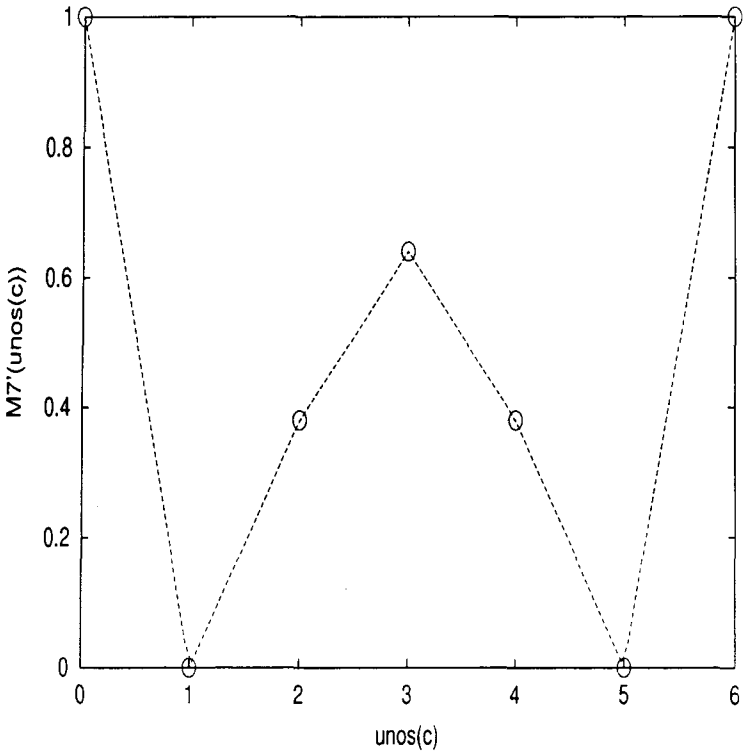


Figura 2.16: Gráfica de la función auxiliar para M_7. En el eje horizontal aparecen los valores enteros 0 a 6 los cuales corresponden a la cantidad de unos en el subcromosoma al cual se aplica.

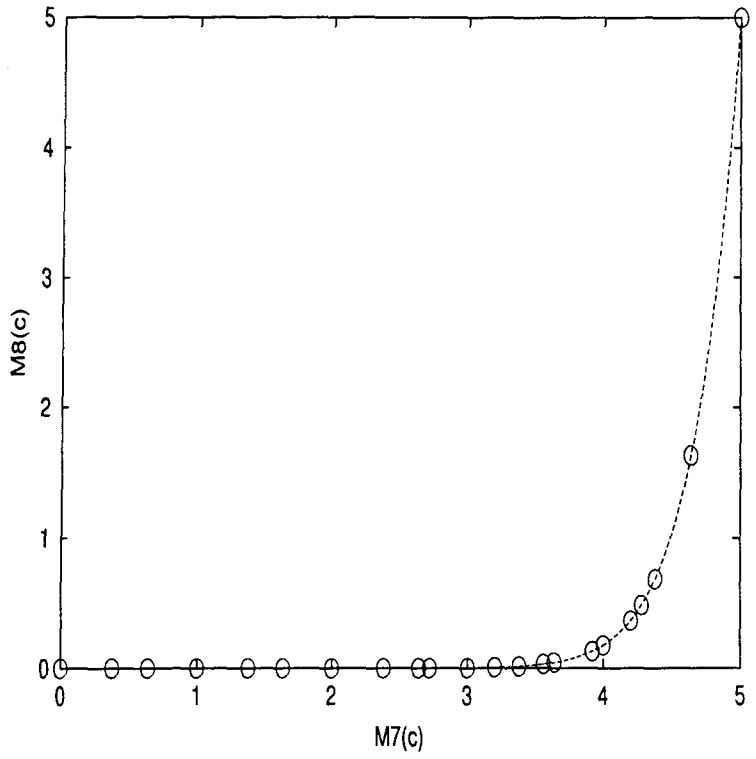


Figura 2.17: Gráfica del perfil de decaimiento usado en M_8.

La industria de las aerolíneas es una industria millonaria. Sus costos, insumos y ganancias involucran cada uno cifras enormes. Hacer ahorros en algunas de estas áreas conduce también a ahorros enormes. La planeación o programación de rutas es una área que impacta sensiblemente en las ganancias y gastos de las compañías y es un área que abre el paso para la investigación científica y la investigación de operaciones en lo particular.

La decisión de cómo trazar un itinerario y ruta de un avión es uno de los problemas más relevantes para las aereolíneas, pero este problema no es sencillo; es un problema tal que a pesar de que su formulación no resulta tan compleja, su proceso de solución ciertamente sí lo es, al involucrar una cantidad excesiva de cálculos. El problema de *rutear* un avión consiste en desarrollar una tabla de planeación de tiempos y movimientos; de gastos y ganancias para diferentes vuelos de la compañía aérea. El proceso de programación de vuelos en general consiste de dos fases. Una fase de construcción de una programación de operaciones y otra fase de evaluación. En general el proceso de solución consiste de un ciclo que itera entre dos fases. Una fase de construcción de una planeación y otra en la cual se evalúa dicha planeación. Para la fase de construcción de las rutas un gran número de factores deben tomarse en consideración. El conjunto de rutas preestablecidas, la frecuencia del servicio, los tráficos estimados, así como la ganancia que obtiene la compañía trasladando pasajeros entre ciudades de cada ruta. También intervienen las características propias de la flotilla de aviones y sus gastos de operación. Con todos esos elementos, la fase de construcción de la ruta con la planeación de tiempos de salida y llegada, asignando tipos de aviones específicos para satisfacer las decisiones de frecuencia en el ruteo. Durante la fase de evaluación, el personal de operaciones examina los resultados considerando factibilidad y tomando en cuenta consideraciones de desempeño. Cualquier deseo manifiesto de mejora, crítica o comentario se proporciona como retroalimentación y se regresa a la fase de construcción de la ruta continuando este ciclo hasta la obtención de una programación satisfactoria.

El ruteo de aviones con múltiples paradas es uno de los principales problemas que se enfrentan en la fase de construcción. El problema ha sido atacado en la tesis doctoral de Salvador García-Lumbreras [41], pero en el enfoque ahí propuesto el problema está formulado como un problema de optimización global. La presente investigación tiene como objetivo aplicar las técnicas de optimización multimodal desarrolladas aquí como una propuesta de aceleración del proceso de construcción y evaluación de las rutas construidas así como en el enriquecimiento en la toma de decisiones al presentarse un conjunto de diferentes soluciones en lugar de una única solución.
El deseo de desarrollar esta aplicación tiene los siguientes elementos.

- El problema es económicamente relevante.
- El problema es teóricamente importante, debido a que es del tipo NP-difícil [42, 43].
- Proveer soluciones subóptimas tiene sentido, debido a que no es un problema de decisión o existencia.

- Proveer un conjunto de soluciones diversas presenta un marco enriquecedor para los tomadores de decisiones, y podría ayudar a acelerar el proceso de decisión.

2.5.1. Formulación del problema

Presentaremos una descripción formal del problema conocido como Ruteo de Vuelos con Múltiples Páradas (RVMP). El problema se formula sobre un grafo dirigido, $G(N, A)$, donde N es un conjunto de nodos, los cuales representarán las ciudades donde los aviones hacen escala, saliendo o llegando. A representa un conjunto de arcos dirigidos entre nodos, los cuales a su vez representarán los únicos trayectos de vuelos entre ciudades intermedias. Ciertas consideraciones se hacen sobre este grafo dirigido G. Todo lo anterior podría resumirse diciendo simplemente que G es una *red dirigida.* Con esto se dice los siguiente. Que el grafo, sin tomar en cuenta la dirección es conexo. Es decir que si eliminamos la dirección de las flechas, todos los nodos están conectados. Existe un nodo al cual no llega ninguna flecha y otro del cual no sale ninguna flecha. Adicionalmente que el grafo no tiene ciclos. Lo cual podría a su vez resumirse diciendo que debe existir una forma de enumerar los nodos con números del 0 a $n-1$, siendo $n = |N|$, de tal suerte que el conjunto de arcos A puede considerarse como un subconjunto de $\{(i,j)|i < j\}$. Los nodos que corresponden a los números 0 y $n-1$ son especiales. Al nodo con posición 0 se le llama *base principal* y el nodo con etiqueta $n-1$ se llama *nodo terminal.* Un lado (i,j) está en A tendrá como significado que el servicio directo de la ciudad i a la ciudad j es posible. Con esta representación, la ruta de un avión se entiende como un camino dirigido desde el nodo 0 hasta el nodo $n-1$.

Con este contexto el problema podría formularse matemáticamente como:

$$\text{máx}\left(\sum_{(i,j)\in A}\rho_{ij}y_{ij} - \sum_{(i,j)\in A}\sum_{t}\gamma_{ijt}x_{ijt}\right) \tag{2.14}$$

sujeto a las restricciones:

$$\sum_{t}\sum_{j\in S(i)}x_{ijt} - \sum_{t}\sum_{j\in P(i)}x_{jit} = \begin{cases} 1, & \text{si } i = 0 \\ 0, & \text{si } 2 \leq i \leq n-1 \\ -1, & \text{si } i = n-1 \end{cases} \tag{2.15}$$

$$\sum_{k\in Q(i)}\sum_{m\in R(j)}y_{km} \leq \bar{C} + \sum_{t}(C_t - \bar{C})x_{ijt}\forall(i,j) \in A;\ \forall t \tag{2.16}$$

$$y_{ij} \leq \sum_{t}\sum_{k\in P(j)}D_{ij}x_{ikt}\forall(i,j) \in M;\ \forall t \tag{2.17}$$

$$y_{ij} \leq \sum_{t}\sum_{k\in S(i)}D_{ij}x_{kjt}\forall(i,j) \in M;\ \forall t \tag{2.18}$$

$$y_{ij} \geq 0; x_{ijt} \in \{0,1\}\ \forall(i,j) \in M;\ \forall t \tag{2.19}$$

donde los conjuntos son:

$$\begin{aligned} N &= \text{el conjunto de nodos} \\ A &= \text{el conjunto de arcos} \\ P(i) &= \{j : (j,i) \in A\} \\ S(i) &= \{j : (i,j) \in A\} \\ Q(i) &= \{j : i \text{ es alcanzable desde } j\} \\ R(i) &= \{j : j \text{ es alcanzable desde } i\} \\ M &= \{(i,j) : i = 1,2,\ldots,n-1 \text{ y } j \in R(i); i \neq j\} \end{aligned}$$

Índices:

$$\begin{aligned} i &= \text{ciudad } i(=0,1,2,\ldots,n-2) \\ j &= \text{ciudad } j(=1,3,\ldots,n-1) \\ t &= \text{tipo de avión } j(=0,1,2,\ldots,T-1) \\ (i,j) &= \text{arco } (i,j) \in A \end{aligned}$$

Parámetros:

$$\begin{aligned} C_t &= \text{cantidad de asientos en el avión del tipo } t \\ \bar{C} &= \text{máx}_t\{C_t\} \\ n &= \text{número de ciudades en la red} \\ T &= \text{número del tipo de aviones diferentes} \\ D_{ij} &= \text{máxima demanda conocida de pasajeros para el segmento origen-destino } (i,\ j) \\ \rho_{ij} &= \text{ganancia por pasajero del origen } i \text{ al destino } j \\ \gamma_{ijt} &= \text{costo fijo para un avión del tipo } t \text{ al viajar en el segmento } (i,j) \end{aligned}$$

Variables de decisión:

$$\begin{aligned} x_{ijt} &= \begin{cases} 1, & \text{si el avión del tipo } t \text{ vuela en el segmento } (i,j) \\ 0, & \text{otro caso} \end{cases} \\ y_{ij} &= \text{el número de pasajeros transportados de la ciudad } i \text{ a la ciudad } j \end{aligned}$$

El objetivo consiste en maximizar la ganancia total definida como el pago recibido, el cual corresponde a la primera suma en la fórmula 2.14, menos el costo de la ruta, que corresponde a la segunda suma también en 2.14. En este modelo, la posibilidad de usar uno de diferentes aviones disponibles se representa permitiendo diferentes arcos entre dos ciudades directamente conectadas en la red.

Las restricciones 2.17 y 2.18 permiten a los pasajeros viajar de la ciudad i a la ciudad j únicamente si la ruta pasa efectivamente primero por la ciudad i y posteriormente, aunque posiblemente no en forma inmediata, por la ciudad j dentro de la ruta. Por otro lado las restricciones 2.17 y 2.18 simultáneamente limitan el número de pasajeros en el trayecto (i,j) a una valor que no exceda la demanda entre dichas ciudades. Las restricciones 2.16 requieren que el total del número de pasajeros transportados en el trayecto (i,j) no exceda la capacidad del avión seleccionado. El número total de pasajeros transportados en cualquier trayecto (i,j) es la suma de todos los pasajeros que provienen de una ciudad desde la cual se puede alcanzar la ciudad j y que tienen como destino una ciudad alcanzable desde la ciudad i. Si el trayecto (i,j) no está incluido

en la ruta el lado derecho de 2.16 asume el valor no restrictivo $\bar{C}$. Las restricciones del tipo 2.15 requieren que cualquier ruta sea construida arrancando de la ciudad 0 como origen y terminando en la ciudad $n-1$. Debido a la naturaleza de las variables de decisión, el problema planteado aquí es un problema entero mixto que es ubicado en la categoría NP-difícil.

2.6. Sumario

En el presente capítulo hemos primeramente hablado de los algoritmos genéticos; de cuáles son sus componentes principales y cuál es la idea principal que está detrás de su accionar. Hemos hablado del problema de optimización multimodal y de algunas variantes basadas en algoritmos genéticos que han sido desarrolladas para este problema en lo específico. Asimismo, hemos mostrado el marco de referencia experimental el cual sirve para comparar propuestas usadas comúnmente en la literatura. Finalmente, en esta introducción hemos también cubierto la definición de un problema de optimización discreta de relevancia práctica, conocido como el problema de ruteo de vuelos con múltiples paradas en el cual aplicaremos la estrategia desarrollada. En esta sección se ha presentado un pequeño compendio del conocimiento relativo a esta investigación. Lejos de pretender cubrir todos los temas y las investigaciones recientes hemos puesto énfasis en conceptos e investigaciones relativas directamente al presente trabajo.

Capítulo 3

Un algoritmo genético para problemas multimodales

En este capítulo se presenta el algoritmo propuesto. En su primer apartado y como motivación se presentan algunos resultados indeseables que manifiestan dos de las estrategias que han sido reportadas en la literatura como más exitosas. Primeramente se discutirán algunos problemas que presenta *crowding* determinístico (CD). Estos problemas dan la motivación para cuestionar la capacidad de CD al tratar de retener subpoblaciones alrededor de algunos óptimos locales a lo largo de las generaciones. La argumentación aquí presentada asocia esta debilidad con la codificación del problema, y en particular con una relación entre el perfil de la función y la codificación del problema, lo cual conlleva en la evolución del algoritmo a la emigración de los descendientes de picos menores hacia picos mayores. Seguidamente, el método de las funciones de compartición (SH) es puesto en la mesa de discusión. Al hacer experimentos similares a los relacionados con CD, SH también muestra dificultad de retener óptimos locales y adicionalmente se encuentra que SH presenta problemas en la convergencia a los óptimos locales.

Habiendo planteado los problemas que manifiestan las estrategias citadas, este capítulo presenta una propuesta de algoritmo al problema de optimización multimodal. La primera motivación en su desarrollo está relacionada con la capacidad de retener una población diversa que pueda ser considerada como solución a problemas de optimización multimodal. La segunda motivación es la búsqueda de un mecanismo de administración adecuado, para que la retención de óptimos identificados como picos altos no esté en detrimento de la capacidad de explorar el espacio de búsqueda.

En términos generales podríamos describir la estrategia de la siguiente manera. En la estrategia formulada, se trabaja un modelo de dos poblaciones y un sistema de administración. Una de las poblaciones hace el papel de un salón de la fama o memoria donde se acumula lo mejor que el proceso de búsqueda ha localizado. La segunda población es una población exploradora. En cada iteración el mecanismo de administración construye la población exploradora usando como base la población memoria, y entonces aplica un algoritmo para hacerla evolucionar. La población final de ella es entonces comparada con la población memoria y, a través de un mecanismo de reem-

plazo, posiblemente nuevos individuos reemplacen a otros en el salón de la fama. Los mecanismos para formar la población exploradora y de reemplazo en la memoria son discutidos.

3.1. Motivación

Dos de las estrategias que han sido reportadas como más exitosas en la solución a problemas de optimización multimodal son *crowding* determinístico (CD) [36] y el método de las funciones de compartición (SH) [33, 34, 36]. Para detalles se puede consultar la disertación doctoral de Samir Mahfoud [36]. En esta sección cuestionaremos este éxito y discutiremos las razones que dan origen a las fallas que presentan. Este cuestionamiento gira en torno a la capacidad de retener un conjunto de soluciones al problema multimodal. Consideramos que retener soluciones es importante debido a que en ciertas situaciones existe la posibilidad de hacer simulaciones o cálculos adicionales, y en general no se tiene información suficiente para saber si se ha alcanzado la solución o no; en cuyo caso esperaríamos no perder lo ya encontrado. En los dos siguientes apartados veremos que tanto CD como SH tienen problemas de retención de óptimos locales.

3.1.1. *Crowding* Determinístico: resultados negativos

CD ha sido reportado con muy buenos resultados en la literatura [36]. Dentro de las funciones de evaluación experimental con la cual ha sido evaluado se encuentra la función M_2, cuya definición aparece en la ecuación 2.8. Repetiremos algunos experimentos ya hechos en la literatura para estimar la dificultad que tiene para resolver esta función y para observar su desempeño en la retención de óptimos locales en condiciones adecuadas de espacio y tiempo.

Estimación de recursos requeridos por un AG

Los recursos necesarios por un AG para resolver un problema específico son dependientes de la propia estategia que sigue el algoritmo y del problema en lo particular. En general, los recursos que se desean medir son cantidad de material genético y el tiempo invertido en el proceso de solución. Y aquí debemos mencionar que la solución al problema deberá ser conocida de antemano a fin de saber si la solución propuesta por la estrategia es aceptable como solución y así estimar estos dos parámetros. La cantidad de material genético la mediremos en tamaño de la población y el tiempo lo estimaremos usando el número de generaciones o evaluaciones.

Nuestra metodología para determinar la cantidad de recursos, que requiere para resolver un problema conocido será la siguiente. Iniciaremos la simulación con una población aleatoria y *reducida en tamaño*. El concepto de población reducida aplica en el sentido que con muy baja probabilidad una población aleatoria con ese número de individuos, o con un número menor, podría ser una solución al problema multimodal. Por ejemplo, si el problema tiene 6 óptimos locales, una población con 6 individuos, o

menos, sería reducida. Sobre esta población se aplicará la estrategia a evaluar y evolucionará la población hasta que un cierto criterio de paro detenga la simulación. La población evolucionada, o población final, se revisará para saber si es aceptada o no como una solución al problema. En la estrategia de determinación de recursos descrita en 3.1, la duplicación del tamaño de la población es un elemento utilizado por Mahfoud [36] en la comparación de SH, SN, y CD; aquí hemos seguido esta estrategia aunque pensamos que en general el tamaño de población obtenido de esa forma no reflejará la cantidad *mínima* de recursos invertidos para obtener la solución del problema multimodal.

En la estrategia de obtención de recursos, cuando la población final no es aceptada como solución, la estrategia a evaluar será aplicada a una población con el doble del tamaño que la población de la simulación previa. Esta nueva población se compone de la población inicial de la simulación previa, y no la población final evolucionada, complementada con una población de tamaño idéntico pero aleatoria. En caso que la población final sea aceptada como solución al problema conocido, se contabilizará el número de individuos de la población con que inició la simulación, y el número de evaluaciones. Esta estrategia está descrita en la figura 3.1. En la descripción del algoritmo (f, c) representa lo siguiente. f es la función asociada al problema y c representa la codificación utilizada para representar los puntos del espacio de soluciones como cromosomas. El criterio de paro S utilizado en los algoritmos a ser evaluados en su desempeño sobre un problema particular está dado por la fórmula 3.1. Este criterio ha sido previamente utilizado por Beasley [35] y por Mahfoud [36].

En la fómula de S, $\bar{f}_k$ representa la evaluación promedio de la población en la generación k. La idea involucrada en el criterio de detención S es que la diferencia entre el promedio de la evaluación de la población y el promedio de los promedios correspondientes en las 4 anteriores generaciones sea menor que una cantidad pequeña. Esta diferencia mímima de cambio se asume un número constante y positivo pequeño preestablecido, que designaremos como s_o.

$$S(\text{generación}_j) = \begin{cases} \text{Veradero}, & \text{si } \left|\bar{f}_j - \left(\frac{\bar{f}_{j-1}+\bar{f}_{j-2}+\bar{f}_{j-3}+\bar{f}_{j-4}}{4}\right)\right| < s_o; \\ \text{Falso}, & \text{otro caso.} \end{cases} \tag{3.1}$$

Este criterio de paro puede no reflejar la convergencia de la población; el movimiento de individuos de la población de la vecindad de un óptimo local a otro, o la aparición de un punto en una vecindad de un óptimo antes vacía puede *esconderse* en el promedio de la evaluación de una generación a otra. Sin embargo, es un criterio sencillo de implementar y que ha sido utilizado en las referencias donde se utilizan las estrategias contra las cuales deseamos comparar la estrategia que se está desarrollando.

La descripción de lo que entendemos como una solución o como un conjunto que se acepta como solución al problema (f, c) de optimización multimodal será la siguiente. Diremos que un conjunto S de puntos en el espacio de búsqueda es una solución o que se acepta como solución, al problema multimodal, si para cada uno de los óptimos locales de f existe un elemento en S que está cerca de él a una distancia menor que una cierta toleracia fijada de antemano que designaremos mediante r_o.

1. Construya una población aleatoria de n individuos, P_o.
2. Tome $i = 0$, $P_i = P_o$.
3. Repita
   - *a)* Tome
     - Número de generaciones = 0,
     - Número de evaluaciones = 0
   - *b)* Aplique la estrategia E_o al problema (f, c) partiendo de la población inicial P_i. Utilice el criterio de paro S.
   - *c)* Si la población final de la simulación es aceptada como solución a f, terminar.
   - *d)* Si la población final no es aceptada como solución a f, generar $|P_i|$ individuos al azar, Pop_{nueva}, y tomar:
     - $i = i + 1$,
     - $P_i = P_{i-1} \cup Pop_{\text{nueva}}$
4. Reporte
   - $|P_i|$,
   - Número de generaciones,
   - Número de evaluaciones

Figura 3.1: Algoritmo de determinación de recursos computacionales, tamaño de la población, número de generaciones, y número de evaluaciones, para una estrategia E_o al resolver el problema (f, c). f representa la función y c representa la codificación utilizada.

CD sobre M_2								
Población			Generaciones			Evaluaciones		
μ	σ	máximo	μ	σ	máximo	μ	σ	máximo
14.56	7.53	32	20.60	13.10	57	373.76	449.0	1824

Tabla 3.1: Promedios y desviaciones estándar correspondientes sobre 50 simulaciones independientes de los recursos utilizados por CD para resolver M_2.

Resultados de CD aplicado a M_2

En esta sección evaluaremos la habilidad que tiene CD para retener, a lo largo de las generaciones, un conjunto aceptado como solución al problema asociado a la función M_2. Para ello primeramente determinaremos la cantidad de recursos suficientes para que CD resuelva dicho problema, y posteriormente, y asignándole una cantidad de recursos adecuados, determinaremos su capacidad de retención de la solución encontrada. Usaremos el algoritmo de determinación formulado previamente con los siguientes parámetros. Iniciaremos con una población aleatoria de cuatro elementos, $n = 4$ (Recordemos que el número de óptimos es 5). La tolerancia utilizada en el criterio de paro será fijada en el valor $s_o = 0.001$, como fue utilizada por Mahfoud [36]. La distancia utilizada para identificar si un punto se encuentra en la vecindad de un óptimo local la fijaremos en 0.1, es decir $r_o = 0.1$. Repitiendo estos experimentos 50 veces, iniciando cada uno de ellos con una población aleatoria diferente, se obtuvieron los resultados de la tabla 3.1. Teniendo una idea de la cantidad de recursos necesarios para que CD resuelva el problema asociado a M_2, determinemos ahora la capacidad de CD para retener soluciones. Para ello utilizaremos una población que garantice suficiente material genético y evaluaremos si en el transcurso de las generaciones la solución sigue estando representada en la población. Fijaremos el tamaño de la población en $n = 32$, el cual corresponde al valor máximo como tamaño de población requerido en las 50 simulaciones.

Vale la pena comentar lo siguiente; los datos obtenidos sobre los tamaños de las poblaciones no se ajustaron a una distribución normal. No se investigó sobre el tipo de distribución que mejor los describía. El valor $n = 32$ es un número suficientemente grande como para esperar que, haciendo una estimación ruda, en el 50/51 %(98 %) de las veces, una población aleatoriamente generada de ese tamaño contenga la suficiente información genética como para que CD la evolucione en la solución al problema. Habiendo fijado el tamaño de la población en 32 dejaremos correr el algoritmo genético, partiendo cada vez de una población aleatoria, hasta un cierto número de generaciones al término de las cuales revisaremos la población final contando cuántas vecindades de los óptimos locales están vacías. Nuevamente el concepto de vecindad vacía se entenderá como que no exista en la población un individuo que diste del óptimo en menos que $r_o = 0.1$. Llevaremos un registro de este número. La tabla 3.2 presenta el promedio de estos conteos sobre 50 muestras e indica, sobre una muestra de 50 simulaciones diferentes, el promedio del número de vecindades vacías en una generación determinada.

CD sobre M_2		
Generaciones	Conteo del vecindades vacías	
	promedio	desviación estándar
50	0.02	0.141
100	0.04	0.198
150	0.04	0.198
200	0.04	0.198
250	0.04	0.198
300	0.04	0.198
350	0.04	0.198
400	0.04	0.198
450	0.04	0.198
500	0.04	0.198
550	0.04	0.198
600	0.04	0.198
650	0.04	0.198
700	0.04	0.198
750	0.06	0.240
800	0.06	0.240
850	0.06	0.240
900	0.06	0.240
950	0.06	0.240
1000	0.06	0.240

Tabla 3.2: Promedio sobre 50 simulaciones del conteo, a lo largo de diferentes generaciones, del número de vecindades vacías al aplicar CD a M_2 utilizando una población de 32 individuos aleatorios a lo largo de diferentes generaciones.

Como vemos en la información de las tablas 3.1 y 3.2, CD resuelve con relativamente pocos recursos el problema multimodal y muestra su capacidad de retener en su población la solución al problema. La pregunta que nos formulamos ahora es, ¿cuál es el desempeño en problemas similares a M_2?, es decir ¿qué tan robusto es CD?

Variaciones simples a M_2: removiendo y adicionando un óptimo, M_{2a} y M_{2b}

Recientes investigaciones teóricas [44] han demostrado que dos estrategias optimizadoras cualquiera tienen en promedio el mismo resultado, cuando este promedio es calculado sobre todas las posibles instancias de problemas. Es decir que toda estrategia que se considera exitosa en un tipo o familia particular de problemas, cuando se piensa que exitosa significa que tiene un desempeño superior al de las otras estrategias, tiene un conjunto de problemas donde tiene un comportamiento por debajo del promedio de las otras estrategias. El conjunto de resultados teóricos es conocidos como los teoremas NFL, de las siglas en inglés *no free lunch*. Pretendiendo hacer un análogo a los teoremas

x	$M_{2a}(x)$
0.1250 (= 1/8)	1.000
0.3738 (≈ 3/8)	0.874
0.6227 (≈ 5/8)	0.583
0.8715 (≈ 7/8)	0.297

Tabla 3.3: Tabla de los óptimos locales para M_{2a}.

NFL, que aplican en el caso de optimización global, hacia otros que abarquen al caso de optimización multimodal, podríamos llegar a pensar que existen una infinidad de problemas que no serán resueltos satisfactoriamente por CD. En esta sección veremos que existen problemas muy cercanos al problema asociado a M_2 donde CD no tiene un desempeño tan espectacular, y que casi parece una fortuna haber seleccionado M_2.

Existen un buen número de variantes sencillas para M_2. Estas variantes podrían incluir cambiar el número de picos, modificar el perfil de decaimiento de los picos, modificar el perfil de los picos, cambiar la codificación, o inclusive cambiar la representación y los operadores genéticos utilizados. Tratando de mantener la simplicidad de la situación hemos modificado M_2 cambiando el número de óptimos locales. Esto puede hacerse fácilmente cambiando el coeficiente de x en la función seno. Dos variaciones simples son:

$$M_{2a}(x) = g_{2a}(x)\,\mathrm{sen}^6(4\,\pi\,x) \tag{3.2}$$

y

$$M_{2b}(x) = g_{2b}(x)\,\mathrm{sen}^6(6\,\pi\,x), \tag{3.3}$$

donde

$$g_{2a}(x) = \exp\left(-2\,\log(2)\,\times\left(\frac{x-1/8}{0.8}\right)^2\right) \tag{3.4}$$

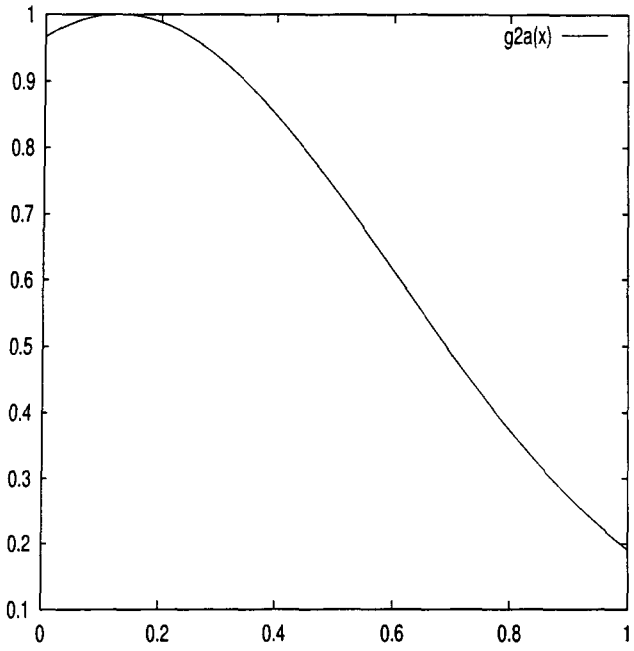
y

$$g_{2b}(x) = \exp\left(-2\,\log(2)\,\times\left(\frac{x-1/12}{0.8}\right)^2\right). \tag{3.5}$$

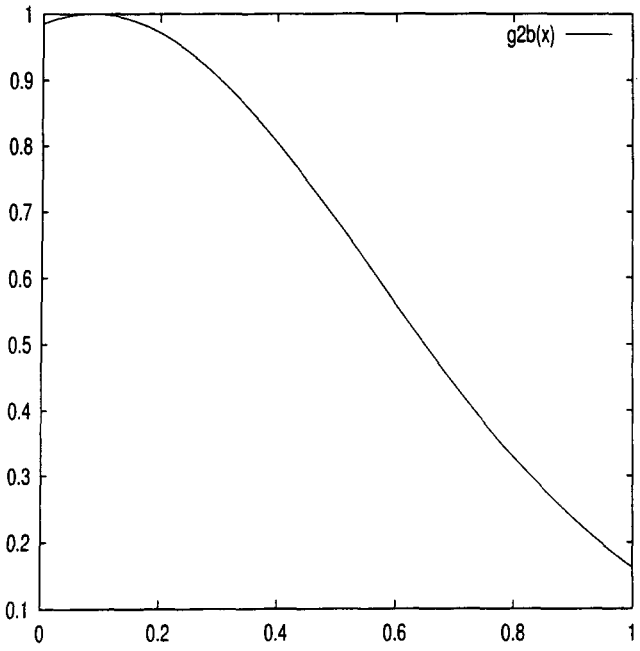
Las funciones de perfil g_{2a} y g_{2b} fueron definidas para conservar la idea de que el primer óptmo local conserve altura 1.0. Los óptimos locales para estas funciones se proporcionan en las tablas 3.3 y 3.4. Similarmente a M_2, los óptimos locales no coinciden exactamente con las fracciones $(2\,i-1)/(2\,n), n = 4, 6$ debido a que las funciones perfil son decrecientes en $[0, 1]$, lo cual se puede ver en las gráficas de la figura 3.2.

Resultados de CD aplicado a M_{2a} y a M_{2b}

Idénticas simulaciones que para M_2 fueron repetidas para M_{2a} y M_{2b}. Es decir, primeramente se determinaron la cantidad de recursos requeridos por CD para resolver ambos problemas y posteriormente, y con abundante material genético, se determinó su capacidad para retener una población solución. Los resultados obtenidos no fueron tan

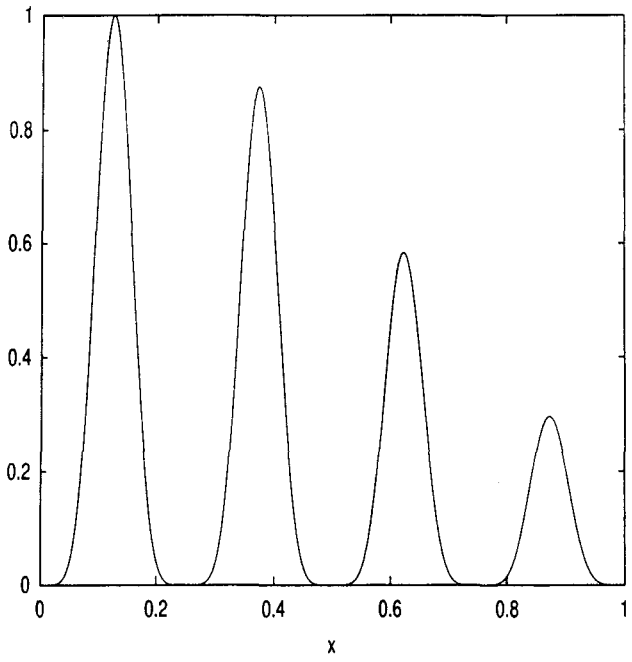


(a) g_{2a}

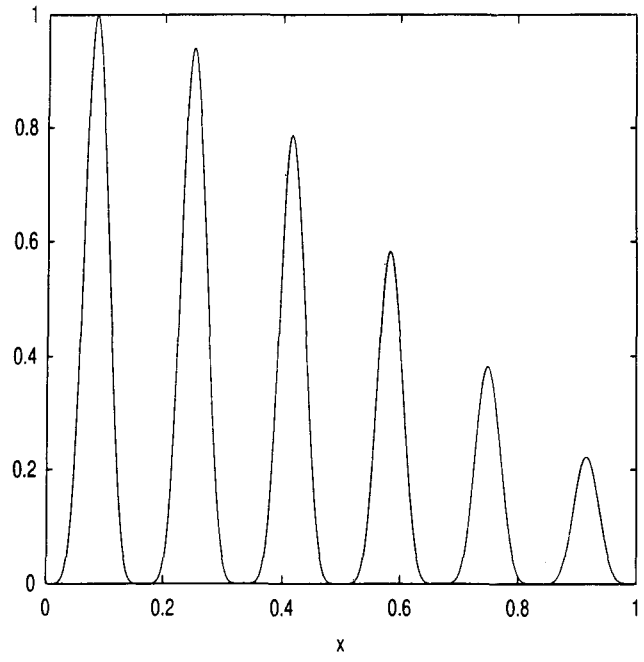


(b) g_{2b}

Figura 3.2: Gráficas de los perfiles de descenso de los óptimos para las funciones M_{2a} y M_{2b}.



(a) M_{2a}



(b) M_{2b}

Figura 3.3: Gráficas de la funciones M_{2a} y M_{2b}.

x	$F_{2a}(x)$
0.0833 (= 1/12)	1.000
0.2497 (≈ 3/12)	0.941
0.4160 (≈ 5/12)	0.786
0.5823 (≈ 7/12)	0.582
0.7486 (≈ 9/12)	0.383
0.9150 (≈ 11/12)	0.223

Tabla 3.4: Tabla de los óptimos locales para M_{2b}.

CD sobre M_{2a} y M_{2b}									
	Población			Generaciones			Evaluaciones		
	$\bar{s}$	$\sigma_{\bar{s}}$	$\bar{s}_{\text{máx}}$	$\bar{g}$	$\sigma_{\bar{g}}$	$\bar{s}_{\text{máx}}$	$\bar{e}$	$\sigma_{\bar{e}}$	$\bar{s}_{\text{máx}}$
M_{2a}	10.56	4.48	16	18.74	18.74	45	226.24	201.96	720
M_{2b}	22.56	8.19	32	33.64	16.06	66	848.64	605.23	2112

Tabla 3.5: Promedios y desviaciones estándar correspondientes, sobre 50 simulaciones independientes de los recursos utilizados por CD para resolver M_{2a} y M_{2b}.

adecuados esta vez. La tabla 3.5 contiene los recursos requeridos para las funciones M_{2a} y M_{2b}. La tabla 3.6 contiene el promedio sobre 50 simulaciones diferentes del conteo de nichos vacíos en la población al cabo de un número fijo de generaciones. Como podemos observar, CD no fue capaz de retener una población que fuera admitida como una solución al problema multimodal; en el caso de M_{2a} a lo largo de las generaciones dos vecindades quedaban vacías, mientras que para M_{2b} siempre quedaban 4. Es decir que en cualquiera de estas dos variaciones CD no exhibía la capacidad de retener individuos alrededor de los picos. Damos por hecho que la población inicial contenía siempre individuos en cada vecindad. Este hecho se asume debido que en ambos casos los tamaños de la población se tomáron como el valor máximo del tamaño de población requerido en los experimentos previos, lo cual garantiza que haciendo una estimación ruda para el 98% de las veces, había suficiente material genético en una población aleatoria como para que CD hubiera resuelto el problema de optimización multimodal. Lo que ocurre es un proceso de emigración de la población; uno o los dos descendientes de ciertos picos aparecen con una mejor evaluación y en la vecindad de otro óptimo abandonando, en la repetición de este proceso, algunos óptimos locales. ¿Cuáles son las razones para este comportamiento negativo? Resultados sobre el comportamiento a largo o mediano plazo del desepeño fueron obtenidos por Sareni y Krahenbuhl [45] sobre la misma función M4; ellos observan que CD determina los cinco picos de la función en las primeras generaciones y que posteriormente la población abandona uno de ellos ($x = 0.451$) para emigrar a otro ($x = 0.681$). Como veremos más adelante, la clave para el problema de la emigración está en la codificación; y más que en la codificación en la relación entre la codificación y el perfil de la función problema. Es

CD: Conteo de vecindades vacías				
	Sobre M_{2a} ($n = 16$)		Sobre M_{2b} ($n = 32$)	
Generación	$\bar{v}$	$\sigma_{\bar{v}}$	$\bar{v}$	$\sigma_{\bar{v}}$
50	0.10	0.303	0.14	0.351
100	0.40	0.606	0.64	0.802
150	0.84	0.738	1.52	1.015
200	1.42	0.642	2.42	1.032
250	1.68	0.513	3.10	0.931
300	1.76	0.476	3.58	0.673
350	1.80	0.404	3.78	0.507
400	1.88	0.328	3.86	0.405
450	1.94	0.240	3.90	0.303
500	1.94	0.240	3.96	0.198
550	1.96	0.198	3.96	0.198
600	2.00	0.000	3.96	0.198
650	2.00	0.000	3.96	0.198
700	2.00	0.000	3.96	0.198
750	2.00	0.000	3.96	0.198
800	2.00	0.000	3.96	0.198
850	2.00	0.000	3.96	0.198
900	2.00	0.000	3.96	0.198
950	2.00	0.000	3.96	0.198
1000	2.00	0.000	3.96	0.198

Tabla 3.6: Promedio sobre 50 simulaciones independientes del conteo de vecindades vacías ($\bar{v}$) y su desviación estándar ($\sigma_{\bar{v}}$) de los óptimos locales al aplicar CD a las funciones M_{2a} y M_{2b}.

decir, la clave está en lo que en inglés se engloba bajo el término: *fitness landscape*. En la siguiente sección haremos un análisis parcialmente detallado de la situación.

Análisis detallado del comportamiento de CD sobre M_{2a}

Revisaremos solamente el caso de M_{2a}. Para entender cómo ocurre la emigración de un nicho a otro dejando uno de éstos vacío y poder hacer algunas simulaciones, trataremos de reducir la complejidad del problema. Reduciremos la longitud de los cromosomas binarios sustancialmente y distribuiremos los individuos por clases. Fijaremos la longitud del cromosoma binario en 6. También asumiremos que el operador de cruce en un punto está siendo utilizado. Para simplificar la escritura de los puntos o individuos los representaremos en su forma entera la cual corresponde al binario representado por el cromosoma. Con un cromosoma de longitud 6, el número de los posibles individuos que se pueden codificar son 64. Los elementos en la vecindad del óptimo local $x = 0.125$ son $\{0, 1, \ldots, 14, 15\}$, pero aquellos con evaluación del 70 por ciento de la altura del pico son $\{8, 7, 9\}$, los correspondientes a $x = 0.3738$ son $\{24, 23, 25, 22\}$, para $x = 0.6227$ son $\{39, 40, 38\}$ mientras que para $x = 0.8715$ son $\{55, 54, 56\}$. Estos conjuntos serían los representantes más adecuados para los óptimos locales.
Suponga que al aplicar CD al problema de la función M_{2a} en una generación determinada la población contiene sólo los elementos:

$$S = \{8, 7, 9, 24, 23, 25, 22, 39, 40, 38, 54, 55, 56\}. \tag{3.6}$$

Es decir que en la población quedan sólo aquellos individuos que tienen por evaluación el 75 % o más de la altura del óptimos al cual están cercanos. Veamos cómo CD puede generar errores de reemplazo. Recordemos cómo procede en cada generación CD; primeramente aparea en forma aleatoria a la población; después los operadores de cruce y mutación son aplicados en cada uno de los pares que se forman. Para cada uno de los pares formados los hijos compiten contra los padres por quedarse en la población para la siguiente generación.

Entremos en un ejemplo específico. ¿Qué pasa si se forma una pareja de padres con los elementos $p_1 = 24$ y $p_2 = 39$?. Vea la tabla 3.7 para detalles sobre los posibles sucesores de la pareja. De acuerdo con la operación de CD, a p_1 y p_2 se les aplicarán los operadores de cruce y mutación para producir dos descendientes, s_1 y s_2. Estos competirán contra p_1 y p_2 por permanecer en la población. Recordando los detalles de operación de CD, cuya descripción aparece en tabla 2.9, la competencia se llevará a cabo dependiendo de qué pareja ofrezca un mejor parecido fenotípico, lo cual será estimado usando las suma de las distancias $d_1 = d(p_1, s_1) + d(p_2, s_2)$ y $d_2 = d(p_1, s_2) + d(p_2, s_1)$. Usando cruce en un punto y de acuerdo a la posición i donde ocurre, se pueden obtener diferentes descendientes; la lista de los descendientes posibles aparecen en la tabla 3.8. Usando la información en las tablas 3.7 y 3.8 podemos obtener los posibles pares que se obtienen en la competencia entre pares y descendientes del par (24,39), esta información está en la tabla 3.9. De acuerdo a ella, en uno de cinco casos cuando se aparean los elementos 24 y 39, que están en las clases 2 y 3, los individuos resultantes pertencen a las clases 1 y 3. Es decir, el elemento 39 permite que el elemento 24 *emigre* de la

	p_1				p_2				
	N_o	fenotipo	evaluación	clase		N_o	fenotipo	evaluación	clase
011000	24	0.3809	0.8532	2	100111	39	0.6190	0.5795	3

Tabla 3.7: Información sobre los individuos 24 y 39 para F_{2a}. Esta información incluye el cromosoma del individuo, la posición en la enumeración que corresponde al entero (N_o), el fenotipo, la evaluación, y la clase o posición del óptimo al cual está asignado.

i	s_1				s_1					
	N_o	fenotipo	evaluación	clase	N_o	fenotipo	evaluación	clase	d_1	d_2
1	7	0.1111	0.9118	1	56	0.8888	0.2577	4	0.539	1.015
2	23	0.3650	0.8423	2	40	0.6349	0.5433	3	0.031	0.507
3	31	0.4920	0.0000	2	32	0.5079	0.0000	3	0.222	0.253
4	27	0.4285	0.1870	2	36	0.5714	0.1483	3	0.095	0.380
5	25	0.3968	0.6780	2	38	0.6031	0.4848	3	0.031	0.444

Tabla 3.8: Posibles descendientes del par (24,39) dependiendo del punto de cruce i que se elija. Se incluye la información sobre el número de descendiente, el fenotipo, la evaluación, y la clase a la cual pertenece.

	p_1'		p_2'	
i	N_o	clase	N_o	clase
1	7	1	39	3
2	24	2	39	3
3	24	2	39	3
4	24	2	39	3
5	24	2	39	3

Tabla 3.9: Individuos y clases correspondientes que resultan de la aplicación de cruce al par (24,39) dependiendo del punto de cruce i.

Par formado				Par resultante				
N_o	clase	N_o	clase	i	m	clase	m	clase
24	2	39	3	1	7	1	39	3
24	2	40	3	1	8	1	40	3
23	2	39	3	1	7	1	39	3
23	2	40	3	1	8	1	40	3
25	2	39	3	1	7	1	39	3
25	2	40	3	1	8	1	40	3
8	1	54	4	1	8	1	40	3
7	1	54	4	1	7	1	39	3
9	1	54	4	1	9	1	41	3
8	1	55	4	1	8	1	40	3
7	1	55	4	1	7	1	39	3
9	1	55	4	1	9	1	41	3
8	1	56	4	1	8	1	40	3
7	1	56	4	1	7	1	39	3
9	1	56	4	1	9	1	41	3

Tabla 3.10: Pares de S que producen errores de reemplazo para M_{2a} y el punto de cruce responsable del error, así como el par producido. Se especifican el número de individuo (m) y su clase.

clase 2 a la clase 1. Haciendo un cálculo exhaustivo mediante *Mathematica* podemos repetir para todos los posibles pares de S (vea ecuación 3.6) y anotar solamente aquellos que producen una pareja donde existe emigración hacia otra clase. Los resultados referentes a errores de reemplazo arrojados aparecen en la tabla 3.10. Los datos nos ayudan a suponer que al principio el error de reemplazo era ocasional, debido a que la probabilidad de aparear el par problemático y hacer el cruce en el punto equivocado era baja. Pero conforme un error ocurre, la probabilidad de aparear pares equivocados aumenta. Esto hace que el proceso entre en un círculo vicioso produciendo el abandono de los nichos por parte de la población.

El anterior razonamiento se conserva para cromosomas de longitud superior, pero ciertamente la probabilidad del cruce en puntos que originaría reemplazos erróneos irá disminuyendo conforme aumente la longitud del cromosoma. Esto fue experimentalmente corroborado haciendo idénticas simulaciones para el problema asociado a la función M_{2a} pero codificando en cromosomas de diferente longitud. La tabla 3.11 ilustra los resultados obtenidos que ratifican parcialemente las conclusiones. Los resultados ilustran que existe una dependencia entre la longitud del cromosoma binario y la velocidad de CD en la pérdida de representatividad en la solución de un problema multimodal.

	CD: Conteo de vecindades vacías							
	$l = 10, n = 16$		$l = 20, n = 16$		$l = 30, n = 16$		$l = 40, n = 16$	
Generación	μ	σ	μ	σ	μ	σ	μ	σ
50	0.64	0.66	0.04	0.20	0.00	0.00	0.04	0.20
100	1.78	0.42	0.58	0.67	0.20	0.40	0.12	0.38
150	1.98	0.14	1.14	0.76	0.68	0.68	0.34	0.56
200	1.98	0.14	1.76	0.52	1.00	0.70	0.62	0.72
250	2.00	0.00	1.92	0.27	1.28	0.73	0.86	0.73
300	2.00	0.00	2.00	0.00	1.64	0.48	1.18	0.75
350	2.00	0.00	2.00	0.00	1.74	0.44	1.44	0.67
400	2.00	0.00	2.00	0.00	1.80	0.40	1.64	0.60
450	2.00	0.00	2.00	0.00	1.90	0.30	1.82	0.39
500	2.00	0.00	2.00	0.00	1.96	0.20	1.90	0.30

Tabla 3.11: Promedio sobre 50 simulaciones independientes del conteo, a lo largo de las generaciones, del número de vecindades vacías al aplicar CD a M_{2a} con diferentes longitudes de cromosomas.

Desempeño de CD sobre variaciones adicionales a M_2

Los resultados presentados por CD sobre M_{2a} y M_{2b} se podrían considerar solamente desafortunados, pues tenemos la intuición de que siempre habrá ejemplos negativos para cualquier estrategia. Pero hay algo más; lo que queremos demostrar en esta sección es que lejos de tener sólo dos ejemplos negativos similares a M_2, este es un caso afortunado para CD.

Los siguientes problemas son similares al problema asociado a M_2:

- Misma codificación que para M_2 pero considerado la función:

$$M_{2c}(x) = e^{-4.0\,(x-0.1)^2}\,\mathrm{sen}^6(5\,\pi\,x). \tag{3.7}$$

  Esta función es semejante a M_2 pero el los picos caen más rápidamente.

- Misma codificación que para M_2 pero considerado la función:

$$M_{2d}(x) = e^{-0.5\,(x-0.1)^2}\,\mathrm{sen}^6(5\,\pi\,x). \tag{3.8}$$

  Esta función es semejante a M_2 pero ahora los picos caen menos rápidamente.

- Misma codificación que para M_2 pero considerando la función:

$$M_{2e}(x) = e^{-2\,\ln 2\,\frac{(x-0.1)^2}{0.8}}\,\mathrm{sen}^{12}(5\,\pi\,x). \tag{3.9}$$

  Esta función es semejante a M_2 pero la forma de los picos es más aguda.

- Misma codificación que para M_2 pero considerando la función:

$$M_{2f}(x) = e^{-2\ln 2 \frac{(x-0.1)^2}{0.8}} \sqrt{|\text{sen}(5\pi x)|}. \tag{3.10}$$

  Esta función es semejante a M_2 pero la forma de los picos es menos aguda.

- Misma función a optimizar pero ahora la codificación es mediante código *gray*, abusando de la notación diremos que éste es el problema M_{2g}.

- Misma función y codificación pero utilizando cruce en dos puntos, referido como problema M_{2h}.

- Misma función y codificación pero utilizando el operador de cruce uniforme, referido como problema M_{2i}.

Experimentos similares a los previos fueron repetidos para los problemas anteriormente descritos y los resultados se resumen en la tabla 3.12. Como podemos observar de dicha tabla, CD presenta la capacidad de retener soluciones en los problemas M_{2c}, M_{2d}, y M_{2i}, contrario al desempeño en los problemas restantes en esta lista.

Conclusiones sobre el desempeño de CD

Lo anterior permite asegurar que M_2 es un caso afortunado en su *vecindad* dentro del conjunto de problemas para el algoritmo CD. La conclusión que podemos sacar es que la capacidad para retener poblaciones solución por parte de CD es muy dependiente del problema y su codificación, y que es relativamente sencillo encontrar ejemplos donde CD exhiba una baja retención de la representación del conjunto solución. Esto nos lleva a la conclusión de que *dejar corriendo* CD generaciones adicionales para mejorar la solución parcial que ha encontrado corre el peligro de perder las soluciones encontradas. Bien podríamos pensar que entonces está bien aceptar o tomar la solución que entrega CD cuando se activa el criterio de paro en un problema específico, pero veremos, posteriormente en el capítulo destinado a la comparación experimental, que hay ejemplos de problemas de optimización donde los tiempos de emigración de la población son menores que los tiempos de convergencia, impidiéndole a CD llegar a la solución esperada. Mantener poblaciones en los nichos ya encontrados sigue siendo un tema pendiente para CD.

3.1.2. Funciones de compartición, retención y convergencia

A la par con CD el método de las funciones de compartición o *sharing* [33] ha sido reportado en la literatura como exitoso cuando se aplica a problemas de optimización multimodal como muestra la tesis de Mahfoud [36].

En esta sección veremos cual es la capacidad de SH para retener subpoblaciones en las vecindades de los óptimos locales, así como su capacidad de acercar los puntos de la población a ellos generación a generación. Anticipando resultados diremos que, a la par con lo que ocurre con CD, la capacidad de retención y convergencia de SH cuando

CD: Conteo de vecindades vacías							
	M_{2c}	M_{2d}	M_{2e}	M_{2f}	M_{2g}	M_{2h}	M_{2i}
	$n_c = 32$	$n_d = 32$	$n_e = 32$	$n_f = 32$	$n_g = 32$	$n_h = 32$	$n_i = 64$
g	μ	μ	μ	μ	μ	μ	μ
50	0.00	0.00	0.00	0.02	0.00	0.00	0.02
100	0.00	0.00	0.00	0.18	0.06	0.00	0.22
150	0.00	0.00	0.00	0.46	0.12	0.00	0.50
200	0.00	0.00	0.00	0.72	0.28	0.00	0.54
250	0.00	0.00	0.00	0.88	0.40	0.00	0.58
300	0.00	0.00	0.00	0.98	0.46	0.00	0.58
350	0.00	0.00	0.00	1.06	0.64	0.00	0.70
400	0.00	0.00	0.00	1.08	0.74	0.00	0.76
450	0.00	0.00	0.00	1.10	0.82	0.00	0.80
500	0.00	0.00	0.00	1.12	0.88	0.00	0.84
550	0.00	0.00	0.00	1.14	0.88	0.02	0.92
600	0.00	0.00	0.00	1.22	0.92	0.02	0.92
650	0.00	0.00	0.00	1.24	0.96	0.02	0.96
700	0.00	0.00	0.00	1.26	1.02	0.02	0.98
750	0.00	0.00	0.00	1.32	1.02	0.02	1.00
800	0.00	0.00	0.00	1.38	1.02	0.02	1.02
850	0.00	0.00	0.00	1.44	1.06	0.02	1.02
900	0.00	0.00	0.00	1.46	1.08	0.02	1.02
950	0.00	0.00	0.00	1.46	1.12	0.02	1.02
1000	0.00	0.00	0.00	1.48	1.12	0.02	1.02

Tabla 3.12: Promedio sobre 50 simulaciones independientes del conteo, a lo largo de las generaciones, del número de vecindades vacías al aplicar CD a las funciones M_{2c}, M_{2d}, M_{2e}, M_{2f}, M_{2g}, M_{2h}, y M_{2i}.

se aplica a un problema específico está fuertemente asociada al problema al que se aplique en particular. Además tal relación lleva a la conclusión que dejar corriendo SH generaciones adicionales puede no mejorar el resultado. Repetiremos los experimentos hechos en la sección anterior pero considerando ahora el método de las funciones de compartición.

Desempeño de SH sobre M_2

Para evaluar el desempeño de SH sobre M_2 repetiremos los experimentos hechos en la sección anterior para CD ahora con SH. Es decir, primeramente se determinará la cantidad de recursos que utiliza para resolver el problema asociado a M_2, y posteriormente se pondrá a prueba su capacidad para retener a lo largo de las generaciones individuos en las vecindades de los óptimos locales de dicha función. Adicionalemente, revisaremos también su capacidad para hacer converger la población a los óptimos locales. Esta convergencia será descrita mediante un parámetro numérico de la siguiente manera.

Definición

El *nivel de convergencia* de la población S al conjunto de óptimos locales O de la función f dado el número positivo r_o se define como

$$D(S, O, r_o) = \frac{1}{|O|} \sum_{o \in O} DP(o, S, r_o) \tag{3.11}$$

donde

$$DP(S, o, r_o) = \frac{1}{r_o} \frac{1}{|\{x \in S | d(o, x) < r_o\}|} \sum_{x \in S, d(o,x) < r_o} d(o, x). \tag{3.12}$$

El nivel de convergencia D entrega el promedio ponderado sobre la distancia r_o de la distancia entre los elementos en la población S a los óptimos O de f. Así por ejemplo $PD(S, O, r_o) = 0.1$, diría que en promedio los individuos en la población S que están próximos a los óptimos en O están a un 10 % de la distancia respecto a r_o. Mientras que $D(S, O, r_o) = 0$ diría que la población sólo consta de óptimos locales. De esta forma, el número $D(S, O, r_o)$ es una medida numérica algo aproximada de representación o proximidad de S a O.

Para determinar los recursos de SH sobre M_2 usamos la estrategia de la figura 3.1, con el mismo criterio de paro ($s_o = 0.001$), el mismo tamaño de población inicial ($n = 4$), y la misma forma de duplicar la población en caso de que ocurra un fracaso, en la solución al problema multimodal. Asimismo se usará la misma codificación y operadores genéticos, también con probabilidades de cruce y mutación de 1.0 y 0.0 respectivamente, e idéntico criterio de aceptación de un conjunto como solución. Los resultados obtenidos se ilustran en la tabla 3.13. Como podemos observar, SH resuelve este problema con relativa facilidad. Veamos ahora cuál es su capacidad para retener individuos alrededor de los óptimos locales con generaciones adicionales. Para ello, repetiremos simulaciones similares a las realizadas para CD. A saber SH iniciará a partir de una población aleatoria suficientemente grande, $n = 32$ y será detenido a un cierto número de generaciones para determinar si existen vecindades de óptimos locales que

SH sobre M_2										
Población			Generaciones			Evaluaciones			Proximidad	
μ	σ	máx	μ	σ	máx	μ	σ	máx	D	σ
38.08	14.42	64	11.78	4.58	25	431.36	175.63	832	0.157	0.0488

Tabla 3.13: Recursos requeridos por SH para resolver M_2.

están vacías. En este caso se contabilizarán y promediarán. Se repitieron 50 simulaciones independientes y los resultados se resumen en la tabla 3.14. El desempeño de SH respecto a la retención era esperado de alguna forma. Esto debido a lo que se infería de las fórmulas desarrolladas en el artículo de Goldberg y Richarson [33] referentes al tamaño que debería tener la población, para, que en una cierta generación, en cada una de las vecindades de los óptimos locales tuviera un elemento de la población. En dicha fórmulas, el tamaño de la población dependía exponencialmente del número de generación. Por consiguiente era de esperarse que en el paso de las generaciones aparecieran vecindades vacías.

Desempeño de SH sobre problemas similares a M_2

Siguiendo la ruta experimental trazada para la revisión del desempeño de CD, pero aplicándola para SH, el siguiente paso corresponde a experimentar el desempeño de SH con problemas similares a M_2. Para ello, usemos el mismo marco extendido para M_2 y compuesto por las funciones M_{2a}, M_{2b}, M_{2c}, M_{2d}, M_{2e}, M_{2f}, y M_{2g}. Repitiendo experimentos idénticos con tamaños de población siempre correspondiendo al máximo valor de la población encontrado en 50 simulaciones hechas para determinar la cantidad de recursos utilizados por SH en cada uno de los problemas de las funciones anteriores, se obtuvieron los resultados resumidos en las tablas 3.15 y 3.16.

Revisión de resultados

Como las tablas 3.15 y 3.16 lo atestiguan, el comportamiento de SH en problemas similares a M_2 presenta a veces problemas, tanto para la retención de una solución admisible, como para la convergencia hacia los óptimos locales. Mientras que la retención de subpoblaciones en los óptimos locales para las funciones M_{2a}, M_{2a}, M_{2d} y M_{2d} es muy bueno, para las funciones M_{2e} y M_{2f} es un poco el esperado debido a la fórmulas de retención desarrolladas por Mahfoud [36]. Pero para las funciones M_{2c} y M_{2h} la retención es bastante baja. Por otro lado, la convergencia a los óptimos encontrados y mantenidos en la población para las funciones M_{2f} y M_{2h} es bastante pobre y al menos para M_{2f} da la impresión de que la población está casi uniformemente en el intervalo $[0, 1]$.

SH sobre M_2				
Generación	Conteo del vecindades vacías		Proximidad	
	μ	σ	D	σ
50	0.02	0.141	0.108	0.0284
100	0.02	0.141	0.104	0.0308
150	0.02	0.141	0.109	0.0272
200	0.04	0.198	0.105	0.0298
250	0.04	0.198	0.103	0.0283
300	0.04	0.198	0.104	0.0249
350	0.04	0.198	0.105	0.0285
400	0.06	0.240	0.105	0.0259
450	0.08	0.274	0.113	0.0294
500	0.08	0.274	0.106	0.0257
550	0.08	0.274	0.102	0.0229
600	0.12	0.328	0.111	0.0252
650	0.12	0.328	0.111	0.0294
700	0.14	0.351	0.110	0.0270
750	0.16	0.370	0.106	0.0281
800	0.22	0.418	0.106	0.0248
850	0.22	0.418	0.102	0.0268
900	0.20	0.404	0.100	0.0277
950	0.30	0.463	0.103	0.0257
1000	0.30	0.463	0.107	0.0297

Tabla 3.14: Promedio sobre 50 simulaciones independientes del número de vecindades vacías y del promedio ponderado de la distancia de los individuos de la población a los óptimos locales en cuya vecindad se encuentran, a lo largo de las generaciones, cuando se aplica SH a M_2.

SH: Conteo de vecindades vacías									
	M_{2a}	M_{2b}	M_{2c}	M_{2d}	M_{2e}	M_{2f}	M_{2g}	M_{2h}	M_{2i}
g	μ	μ	μ	μ	μ	μ	μ	μ	μ
50	0.00	0.00	0.92	0.00	0.10	0.00	0.20	0.06	0.26
100	0.00	0.00	0.96	0.00	0.16	0.00	0.00	0.16	0.44
150	0.00	0.00	0.94	0.00	0.24	0.00	0.00	0.18	0.70
200	0.00	0.00	0.96	0.00	0.26	0.00	0.00	0.16	0.74
250	0.00	0.02	0.96	0.00	0.32	0.00	0.00	0.30	0.74
300	0.00	0.00	0.92	0.00	0.30	0.00	0.00	0.26	0.94
350	0.00	0.00	0.92	0.00	0.30	0.00	0.00	0.32	0.96
400	0.00	0.02	0.94	0.00	0.40	0.20	0.00	0.36	1.18
450	0.00	0.00	1.00	0.00	0.36	0.00	0.00	0.38	1.18
500	0.00	0.02	0.90	0.00	0.38	0.40	0.00	0.52	1.38
550	0.00	0.02	0.98	0.00	0.40	0.00	0.00	0.48	1.32
600	0.00	0.00	0.90	0.00	0.36	0.00	0.00	0.46	1.36
650	0.00	0.00	0.98	0.00	0.42	0.00	0.00	0.44	1.48
700	0.00	0.00	0.94	0.00	0.48	0.20	0.00	0.46	1.46
750	0.00	0.04	0.92	0.00	0.48	0.00	0.00	0.48	1.60
800	0.00	0.02	0.94	0.00	0.46	0.00	0.00	0.48	1.66
850	0.00	0.00	0.98	0.00	0.52	0.00	0.00	0.52	1.70
900	0.00	0.00	0.96	0.00	0.48	0.00	0.00	0.48	1.90
950	0.00	0.00	0.98	0.00	0.54	0.00	0.00	0.48	1.86
1000	0.00	0.02	0.96	0.00	0.54	0.00	0.00	0.46	1.86

Tabla 3.15: Promedio, sobre 50 simulaciones independientes, del conteo a lo largo de generaciones del número de vecindades vacías al aplicar SH a los problemas M_{2a}, M_{2b}, M_{2c}, M_{2d}, M_{2e}, M_{2f}, M_{2g}, M_{2h}, y M_{2i}. SH se aplica a una población aleatoria de 50 individuos.

SH: Convergencia hacia los óptimos									
	M_{2a}	M_{2b}	M_{2c}	M_{2d}	M_{2e}	M_{2f}	M_{2g}	M_{2h}	M_{2i}
g	D	D	D	D	D	D	D	D	D
50	0.052	0.103	0.098	0.114	0.097	0.444	0.102	0.150	0.263
100	0.028	0.096	0.092	0.114	0.084	0.457	0.095	0.148	0.255
150	0.023	0.093	0.100	0.109	0.076	0.452	0.091	0.136	0.256
200	0.020	0.091	0.091	0.112	0.074	0.461	0.089	0.141	0.251
250	0.018	0.095	0.099	0.114	0.077	0.465	0.090	0.140	0.258
300	0.016	0.085	0.093	0.103	0.080	0.464	0.095	0.144	0.250
350	0.015	0.091	0.099	0.111	0.076	0.466	0.090	0.143	0.248
400	0.015	0.095	0.094	0.101	0.073	0.461	0.092	0.137	0.243
450	0.015	0.090	0.093	0.109	0.076	0.462	0.092	0.132	0.235
500	0.014	0.094	0.097	0.116	0.075	0.471	0.093	0.135	0.248
550	0.014	0.098	0.095	0.110	0.076	0.467	0.090	0.143	0.248
600	0.013	0.095	0.094	0.118	0.077	0.464	0.095	0.141	0.239
650	0.013	0.096	0.096	0.117	0.077	0.467	0.094	0.143	0.239
700	0.013	0.092	0.101	0.107	0.076	0.473	0.089	0.133	0.239
750	0.013	0.099	0.098	0.110	0.073	0.470	0.096	0.136	0.240
800	0.013	0.085	0.091	0.107	0.073	0.471	0.090	0.138	0.228
850	0.013	0.092	0.091	0.108	0.072	0.468	0.094	0.135	0.242
900	0.012	0.099	0.098	0.116	0.073	0.471	0.097	0.138	0.227
950	0.012	0.102	0.087	0.107	0.076	0.470	0.093	0.138	0.231
1000	0.012	0.092	0.098	0.116	0.075	0.469	0.090	0.144	0.233

Tabla 3.16: Promedio sobre 50 simulaciones independientes del promedio a lo largo de las generaciones en la evolución, de la distancia de los individuos en la población solución a los óptimos locales en cuya vecindad se encuentran. Resultados de SH aplicado a los problemas M_{2a}, M_{2b}, M_{2c}, M_{2d}, M_{2e}, M_{2f}, M_{2g}, M_{2h}, y M_{2i}. a M_{2a}.

3.1.3. Discusión: población, búsqueda y solución

Como podemos ver de los experimentos anteriores, las dos estrategias que son reportadas como adecuadas en la solución a problemas de optimización multimodales tienen serios inconvenientes relacionados a la retención de óptimos para problemas muy similares a los problemas donde se han reportado como exitosas en la literatura. La cualidad de retención es una cualidad deseable en un contexto incremental de la solución a un problema.

Cuando se intenta resolver un problema de optimización global mediante un algoritmo iterativo, una estrategia razonable y comúnmente usada es conservar aquél punto que representa lo mejor que se ha encontrado durante la búsqueda. De esta forma proceden los usuarios de estrategias como los buscadores de alpinista [46], de recocido simulado [47], de búsqueda tabú [48], algoritmos genéticos como CHC [31] o GENITOR [30], etc. Algunas de estas estrategias conservan ese punto por separado, y algunas otras como punto de referencia en la búsqueda. Cuando otro punto es encontrado, se compara contra el mejor punto obtenido; si en la evaluación lo es superior entonces el nuevo punto reemplaza al primero. En optimización global la decisión de este reemplazo es simple. Cuando la solución al problema es un conjunto de soluciones y no sólo una, donde este conjunto debe cumplir un cierto criterio de diversidad, la retención de un conjunto de individuos encontrados durante la búsqueda se complica.

En los algoritmos genéticos, a cada generación la población representa el resultado parcial de la estrategia y también sirve como un medio de exploración del espacio de búsqueda en la generación siguiente. El título de medio de exploración se justifica debido a que la población contiene la información genética de la cual parte el algoritmo para generar nuevos individuos mediante sus procesos de selección y recombinación: los nuevos individuos creados se forman seleccionando características de los individuos en la población. En un problema como el de optimización multimodal, donde lo que se busca es un conjunto, el algoritmo debe evolucionar cuidadosamente para no perder lo ya encontrado. Desde nuestro punto de vista, esto hace que la capacidad de exploración tenga una seria limitación. Nosotros pensamos que este doble rol que juega la memoria en un algoritmo genético puede tener desventajas en problemas donde lo que se debe conservar es un conjunto amplio y diverso de posibles soluciones. Como hemos visto, las estrategias que han resultado existosas en la literatura como CD y SH no presentan la robustez adecuada, cuando uno pone como requisito la capacidad de retener subpoblaciones identificando a los óptimos locales a lo largo de las generaciones.

3.2. Un nuevo algoritmo basado en dos poblaciones

En el presente apartado se propone un algoritmo orientado hacia la retención de soluciones diversas en un contexto de optimización. La retención será realizada mediante la creación de una población memoria mientras que la exploración del espacio del problema será realizada por una población adicional. Es decir, que el algoritmo propuesto está basado en dos poblaciones. La población memoria contendrá lo mejor que se ha encotrado en el proceso de búsqueda. El concepto *mejor* es una suma de dos elementos.

Por una lado la evaluación del individuo, que está relacionado directamente con la función a optimizar. Por otro el grado de sobrerrepresentatividad de ese individuo como representante de un nicho o del óptimo local. La otra población, llamada exploradora, se encargará de muestrear el espacio de búsqueda. Esta población de tamaño reducido se podrá considerar un micro algoritmo genético (μAG), el cual se refiere a un algoritmo genético con una población reducida, el cual opera con reinicializaciones [49]. La estrategia basada en el uso de un micro AG ha sido investigada por Krishnakumar [50] comparándola contra un AG simple en un problema de optimización, reportando buenos resultados a favor del micro AG. Han existido otras experiencias basadas en micro AGs [51], pero la más cercana a la presente investigación es una que se desarrolló en el transcurso de mi investigación y se debe a Coello y Toscano [52, 53]. El desarrollo se aplica a problemas de optimización multiobjetivo; tiene una población memoria y de ella se forma la población exploradora a la cual se aplicará un proceso evolutivo. La población memoria se considera dividida en dos partes; una parte reemplazable y otra no reemplazable, la cual no cambiará a lo largo del proceso completo y será considerada como la fuente de diversidad. El micro algoritmo genético que sirve como base de exploración utiliza elitismo manteniendo un individuo no dominado aleatorio sembrándolo en la población de la siguiente generación. El algoritmo genético evoluciona hasta que ocurre la convergencia nominal; entonces dos individuos no dominados, o sólo uno si fuera el caso, son tomados de la población final del micro AG. Estos individuos son comparados contra la memoria externa; si uno o los dos permanecen no dominados se añaden a la memoria. De esta población son removidos los individuos dominados que aparecen en las competencias o comparaciones. En el desarrollo propuesto allá no existe ninguna forma explícita de nicheo basada en funciones de compartición pero busca la diversidad usando un mapa dividido del espacio de búsqueda (*grid*) como medida de amontonamiento en un proceso similar al desarrollado por Knowles y Corne [54], el cual tiene un costo computacional reducido. En este proceso de nicheo indirecto, una malla de zonas cubre el espacio y cada una de ellas tiene una medida del número de individuos en dicha la zona. El algoritmo planteado por Coello y Toscano [52, 53] tiene una arquitectura muy similar a la arquitectura del algoritmo planteado pero las componentes internas operan en principios diferentes y hacia una aplicación diferente. Knowles y Corne [54] plantean que el costo computacional de este tipo de arquitecturas es elevado y que en general no mejora el desempeño de la estrategia, en ese caso multiobjetivo. Como veremos posteriormente esta técnica puede conducir a ahorros importantes en el número de evaluaciones, que en un problema donde el costo de evaluación sea elevado puede ser una opción práctica.

La existencia de estas dos poblaciones constituyen la primera parte, la parte que los complementa es el mecanismo de administración; este mecanismo se encargará de reemplazar elementos en la memoria y construir la población exploradora. En los siguientes apartados, describiremos las componentes del algoritmo propuesto, intentando describir la idea que persigue cada componente.

3.2.1. Figuras de mérito

La parte principal del algoritmo propuesto es la parte de administración. Las dos tareas que realiza son el reemplazo de elementos en la memoria por nuevos elementos recién encontrados, y la selección de individuos de la memoria para conformar la población exploradora. Estas tareas son realizadas mediante la definición de dos indicadores numéricos o figuras de mérito. El mecanismo de administración utilizará la figura de reemplazo determinísticamente para saber cuándo un individuo recién generado reemplaza a otro punto que radica en la población memoria. Mientras que para conformar la población exploradora usará la figura de selección estocásticamente asociando a cada individuo en memoria una probabilidad de hacer una copia de él en la construcción de la población exploradora.

Reemplazo

En esta sección describiremos cómo se construye el indicador numérico para determinar los elementos que formarán la población exploradora. En el problema de la optimización global no hay duda si se reemplaza el punto conservado por algún otro punto recién hallado: la función a optimizar da la medida numérica clara si el reemplazo procede o no. En el caso de la optimización multimodal, la función objetivo no puede ser el único indicador a considerar pues en este caso estamos también interesados en los óptimos locales de la función objetivo.

Pensemos que la idea de un solo indicador numérico para el posible reemplazo se puede aplicar al caso del problema de optimización multimodal. Si se usa como criterio único la función a optimizar para el reemplazo, será inevitable la pérdida de máximos locales que queden por abajo en evaluación de los máximos globales. Por otro lado, el uso de la función ajustada usada en un algoritmo genético que usa funciones de compartición puede tener problemas como se puede ver a continuación. Recordemos que cuando se usan funciones de compartición se define el *conteo de nicho* de un elemento x de la siguiente forma:

$$N(x) = \sum_{y \in S} sh(d(x,y)), \tag{3.13}$$

donde $d(x,y)$ es la distancia entre x y y, y

$$sh(d) = \begin{cases} 0 & \text{si } d > \sigma_{share} \\ 1 - (d/\sigma_{share})^a & \text{si } d \leq \sigma_{share}. \end{cases} \tag{3.14}$$

En la definición de sh eligiremos, como en parte de la literatura reciente [36, 35], $a = 2$. De esta forma $sh(d(x,x)) = 1$ y por consiguiente la ecuación 3.13 puede escribirse ahora como:

$$N(x) = 1.0 + \sum_{y \in Pop - \{x\}} sh(d(x,y)). \tag{3.15}$$

El conteo de nichos de un elemento es una buena aproximación a la cantidad de individuos próximos a un elemento. La función o *aptitud ajustada* en *sharing* se calcula

como:

$$f_a(x) = \frac{f(x)}{N(x)}. \tag{3.16}$$

La interpretación para la fórmula anterior es que la *aptitud natural*, $f(x)$, de un individuo se ve disminuida por la presencia de otros individuos que están en el mismo nicho que el individuo con los cuales habrá de compartir el recurso, el cual representa la altura del la función en el óptimo local: el grado de disminución de la aptitud está definido por la cantidad de individuos en ese nicho, la cual está estimado por $N(x)$.

Para tener un indicio del porqué esta función ajustada no es conveniente como indicador veamos el siguiente ejemplo. Imaginemos el caso hipotético en el cual tengamos en lo mejor encontrado los siguientes puntos y sus evaluaciones:

$$\begin{array}{rcll} x_1 & = & 1, & f(x_1) = 1.5 \\ x_2 & = & 1.1, & f(x_2) = 1.4 \\ x_3 & = & 2, & f(x_3) = 1.0 \\ x_4 & = & 3, & f(x_4) = 0.9. \end{array} \tag{3.17}$$

Donde además el radio de compartición es $\sigma_{share} = 0.8$, así los contadores de nicho para esos puntos serán $N(x_1) = 1.98$, $N(x_2) = 1.98$, $N(x_3) = 1$, y $N(x_4) = 1$. En este caso las evaluaciones ajustadas de los puntos serán: $f_a(x_1) = 0.756$, $f_a(x_2) = 0.705$. $f_a(x_3) = 1.0$, y $f_a(x_4) = 0.9$. Esto tiene el efecto negativo de que los representantes del pico mayor tienen un valor del indicador menor que los de picos menores. En este reducido ejemplo, vemos que sería conveniente para motivos de reemplazo que al menos un representante del pico mayor tuviera como indicador su propia evaluación, mientras que el segundo representante fuese penalizado.

Una alternativa para introducir la idea anterior es la siguiente variante del conteo de nichos. Revisemos los contadores de nicho de la pareja de puntos x y y. En ambos aparece el sumando $sh(d(x,y))$, este sumando hace que las evaluaciones de ellos sean penalizadas sin importar sus evaluaciones y esto trae como resultado que la evaluación ajustada de ellos pueda ser pequeña para ambos. La idea propuesta es que uno de ellos no sea penalizado; aquél con mejor evaluación. Para esto, el sumando $Sh(d(x,y))$ no debería aparecer en un contador de nicho de ese individuo. Para realizar esta idea definamos el contador de nichos auxiliar:

$$N_1(x) = \sum_{y \in Pop} sh_1(x,y), \tag{3.18}$$

donde

$$sh_1(x,y) = \begin{cases} 0 & \text{si } d > \sigma_{share} \text{ ó } f(x) \geq f(y) \\ sh(d(x,y)) & \text{otro caso.} \end{cases} \tag{3.19}$$

El caso $f(x) = f(y)$ será resuelto en favor de aquél con menor contador de nichos o aletoriamente en caso de empate aún en esta situación.
Con este contador auxiliar definimos una figura de mérito para el reemplazo de individuos:

$$R_0(x) = \frac{f(x)}{N_1(x)}. \tag{3.20}$$

De acuerdo con esta figura de mérito, la evaluación de un punto x en el espacio de búsqueda es penalizada por el término $N_1(x)$, el cual castiga con preferencia ciertos elementos sobre otros; las copias son penalizadas de una manera singular. A modo de ejemplo, si se tiene un grupo de k idénticos individuos dentro de la población con evaluación f_1, y descartando al resto de la población, uno de ellos tendrá un indicador de f_1; otro de $f_1/2$; otro de $f_1/3$ y así sucesivamente.

Una pregunta que surge es si el nivel de penalización resulta adecuado de un problema a otro. La intuición indica que la penalización parece adecuada para evitar duplicidad en lo que pretendemos sea la memoria de lo mejor encontrado. Pero es válido poner en duda el nivel de penalización. Cambiemos el modelo de reemplazo R_o por otro similar donde tengamos la posibilidad de ajustar la intensidad de la penalización:

$$R(x) = \frac{f(x)}{N_1(x)^{\alpha}}. \tag{3.21}$$

Haciendo uso de este indicador para el reemplazo de individuos las copias de ellos serán reemplazadas primeramente y al menos un individuo representará al nicho con su evaluación. Es motivo de cuestionamiento si esta penalización es excesiva o insuficiente. Todo esto con respecto al reemplazo de individuos en la problación. El indicador R es la única medida numérica para competencia al decidir la permanencia entre dos individuos. Esta medida es una resultante entre la altura del pico en el cual se encuentra un individuo, y de alguna manera, de cómo se compara él contra los otros elementos del nicho, tanto en calidad como en número. El elemento clave en esta medida es el factor de penalización N_1. En secciones siguientes haremos un análisis de este indicador.

Selección

El retener elementos en la población puede tener efectos negativos. Si la probabilidad de seleccionar un individuo para conformar la población exploradora sólo dependiera de su evaluación, esto podría viciar negativamente la búsqueda al darle excesivas oportunidades. Este efecto puede degradar la diversidad de la búsqueda. Para limitar el número de oportunidades reproductivas es necesario introducir un mecanismo de control. Una alternativa para imponer limitaciones en la elección de un individuo para formar la población exploradora es usar el número de veces que ha sido seleccionado para este fin. Este número debería imponer una penalización para la selección. Para llevar a cabo esta idea hemos introducido un contador en cada invididuo que indica cuántas veces ha sido seleccionado para formar la población exploradora. Para los individuos iniciales de la memoria este contador será 1. Cuando un individuo es seleccionado para construir la población exploradora este contador se incrementará en uno. Cuando un individuo resultante de la población exploradora se ha seleccionado para reemplazar un individuo en la población memoria dos alternativas para iniciar este contador pueden ser posibles. La primera es inicializar su número de selecciones en 1, mientras que la segunda alternativa es iniciar su contador en el promedio de este contador de aquellos elementos que formaron la población exploradora. Hemos tomado esta segunda opción.

Una de las ideas involucradas en *sharing* consistía en asignar oportunidades reproductivas por nicho. Para explicar esto recurramos al siguiente ejemplo. Si x y y son

dos individuos iguales en la población y suficientemente alejados del resto de ella, de tal suerte que pueden los dos ser considerados como los únicos representantes del nicho donde se encuentran, entonces las oportunidades reproductivas se calculaban con base en la aptitud ajustada $f_a(x) = f(x)/N(x) = f(x)/2$ y $f_a(y) = f(y)/N(y) = f(y)/2 = f(x)/2$. Es decir que el nicho tiene asignadas oportunidades reproductivas dependiendo cómo se compare su altura contra la altura de los demás picos, y los representantes se dividen entre sí estas oportunidades. Como puede haber varias copias de óptimos locales en memoria utilizaremos esta idea para construir la figura de selección.
Tomando en cuenta el concepto del contador de selecciones C y la figura proveniente de SH podemos proponer una figura de mérito para seleccionar elementos para formar la población exploradora de la población memoria:

$$S_o(x) = \frac{f(x)}{N(x)} \times \frac{1}{C(x)}. \tag{3.22}$$

Donde $C(x)$ es el contador del número de veces que el individuo x ha sido seleccionado de la memoria para formar la población exploradora. El término $f(x)/N(x)$ en la ecuación 3.22, es el término *atractivo* que se usa como indicador en la selección en SH. Atractivo porque reparte individuos de un mismo nicho el número de selecciones. El contador de selecciones $C(x)$ de un individuo impone una penalización en el número de oportunidades para formar la selección de la población exploradora. Como en el caso de R, podríamos dudar sobre si la intensidad de la penalización resulta excesiva o insuficiente y proponer en el caso general un indicador de la forma:

$$S(x) = \frac{f(x)}{N(x)\, C(x)^{\gamma}}. \tag{3.23}$$

En esta sección hemos dado las ideas básicas de los criterios para el reemplazo en la población memoria y para la construcción de la población exploradora. En el siguiente apartado introduciremos la arquitectura del algoritmo genético propuesto.

Estableciendo la arquitectura

Habiendo definido las figuras de mérito para la selección y reemplazo queda por definir organización y operación. En esta sección se completa la descripción del algoritmo para la solución de problemas de optimización multimodal. Primeramente se hace una descripción detallada de la arquitectura y operación del mismo. Posteriormente se discute en términos cualitativos el posible desempeño. Este algoritmo se basa en el uso de dos poblaciones. Una población exploradora y una población memoria la cual contiene los mejores individuos encontrados durante el momento actual de la búsqueda. La población exploradora representa el medio por el cual se muestrea el espacio de búsqueda. El mecanismo de operación podría describirse en general de la siguiente manera. La población memoria tiene n elementos, los cuales inicialmente son generados en forma aleatoria. Repetidamente una población exploradora es generada y evolucionada. La población exploradora es construida tomando k elementos de la población memoria mediante la figura de mérito S. Para este propósito se utiliza selección universal

estocástica, conocida por sus siglas como SUS [20]. Esta población es evolucionada mediante un algoritmo genético. En los experimentos aquí desarrollados se utilizó CD como estrategia de evolución debido a su baja complejidad computacional durante su evolución. Al término de su evolución, la población exploradora final es confrontada contra la población memoria. De acuerdo a esta confrontación son seleccionados los peores k individuos de la población memoria y los individuos de la población exploradora final de acuerdo a la figura R. Estos mejores individuos reemplazan a los peores individuos de la memoria si acaso son mejores que ellos. La ejecucción de este ciclo se lleva a cabo en forma iterada hasta que un cierto criterio de convergencia se obtenga.

La forma como se confrontan la población memoria y la población exploradora es la siguiente. Con ambas poblaciones se forma una única población sobre la cual se realiza el conteo de nichos y el conteo de nichos modificado. Usando esta información se calcula la figura R para cada elemento tanto en la población memoria como en la población exploradora.

El criterio de paro en el algoritmo propuesto es uno que pretende determinar una forma de convergencia. Esta convergencia es medida mediante el promedio de la evaluación de la población; cuando la diferencia entre el promedio de la evaluación de la generación actual y el correspondiente promedio de la evaluación de las cuatro generaciones previas no sea mayor que un cierto número pequeño se asume que la convergencia se ha alcanzado. Este criterio de paro coincide con aquel desarrollado en la parte de evaluación de los algoritmos CD y SH. La figura 3.4 da una ilustración del algoritmo propuesto, al cual llamaremos memGA, en su parte central, mientras que el seudocódigo del algoritmo propuesto aparece en la figura 3.5.

Una de las principales críticas que podría hacerse al algoritmo propuesto es el cálculo adicional u *overhead* debido al cálculo del conteo de nicho y las comparaciones para el reemplazo y selección. Así por ejemplo en cada ciclo selección-evolución-reemplazo debe hacerse

1. Aplicación de SUS sobre la población de n elementos: $O(n)$. Asumiendo que el indicador S está calculado.

2. Aplicación de CD, orden no estimado.

3. Reemplazo:

   a) determinación de R (y S para la siguiente generación), orden $O((n+k)^2)$,

   b) determinación de los k peores de la población memoria $O(k\,n)$, quizá menor si se utiliza un método de torneo,

   c) competencia directa por el reemplazo de k individuos contra k individuos, orden $O(2\,k)$, procediendo con una estrategia de *merge-sort*.

A pesar que cada etapa tiene su costo computacional, la etapa que se vuelve un cuello de botella es la comparación entre la población memoria y la población final en la fase de evolución. Uno de los factores por elegir una población exploradora pequeña es su impacto computacional en esta fase.

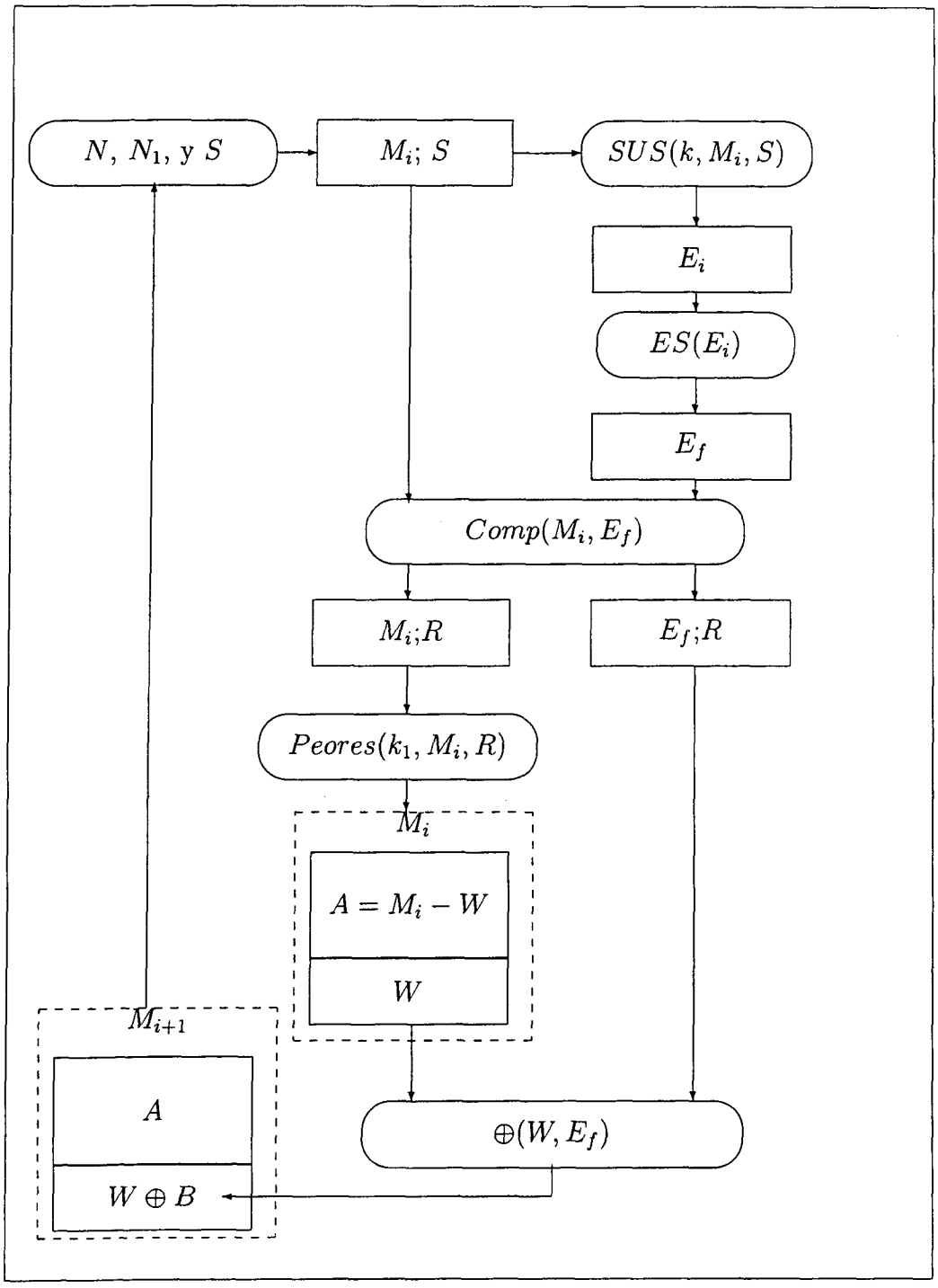


Figura 3.4: Ciclo principal de memGA

1. Genera aleatoriamente una población M_o de n individuos.
2. Mientras no se cumpla un criterio de paro haz:
   - *a)* Determine los indicadores N y N_1 para de la población.
   - *b)* Obtenga el indicador S de cada elemento.
   - *c)* $SUS(M_i, S)$
     Forma la población exploradora E_i seleccionando k individuos de M_i mediante **SUS** tomando el indicador S.
   - *d)* $ES(E_i)$
     Aplique un procedimiento de evolución **ES** a E_i, obteniendo E_f.
   - *e)* $Comp(M_i, E_f)$
     Determine los indiciadores N y N_1 para $M_i \cup E_f$.
   - *f)* Determine los valores de R para M_i y E_f.
   - *g)* $Peores(k, M_i, R)$
     Determine los k peores individuos de M_i de acuerdo a R, W.
   - *h)* $\oplus(W, E_f)$
     Reemplace los individuos de W con aquellos elementos de E_f que sean mejores de acuerdo a R.

Figura 3.5: Seudocódigo del algoritmo propuesto: memGA.

Pretendiendo disminuir el costo computacional de esta fase de comparación, puede ser suficiente hacer estimaciones para los indicadores R y S en lugar de hacer un cálculo exacto, pero ello cae de momento fuera de los límites de la presente investigación. Podría seguirse una estrategia similar a la desarrollada por Yin y Germay [55, 56] que utiliza técnicas de agrupamiento (*clustering*) para reducir la complejidad de la determinación del conteo de nichos en cada generación de $O(n^2)$ a $O(m\,n)$ donde m es el número de agrupamientos. Pero nuestra intención primordial consiste en estudiar y comparar experimentalmente el modelo. Como veremos posteriormente, en los experimentos propuestos con funciones de moderada complejidad el ahorro hecho en las evaluaciones de la función objetivo hace que el desempeño del algoritmo propuesto sea atractivo cuando se le compara contra CD o SH.

3.3. Resumen

En este capítulo hemos puesto de manifiesto que la retención de individos es importante cuando se pretende resolver un problema en un contexto incremental, esto es en un contexto donde queremos continuar con el proceso de solución. Mediante experimentos hemos visto que esta propiedad de retención la exhiben muy limitadamente algoritmos como CD y SH. Los experimentos desarrollados muestran que ambas estrategias no son robustas debido a que tomando un problema específico donde se desarrollan muy bien, pequeñas variaciones dan origen a problemas donde el desempeño cambia radicalmente cuando se hacen experimentos donde lo que se busca es la retención. Tomando tales fracasos parciales como motivación hemos planteado una nueva propuesta de algoritmo en la solución al problema de optimización multimodal. Hemos descrito su arquitectura, sus componentes básicas y los indicadores numéricos que utiliza. Hemos argumentado algunas de sus componentes. En la siguiente sección haremos un análisis parcial de sus figuras de mérito que nos ayudará a escoger los parámetros que en esta parte sólo fueron esbozados.

Capítulo 4

Análisis del algoritmo propuesto

En el capítulo anterior hemos planteado el algoritmo propuesto el cual fue bautizado con el nombre de memGA, pero no hemos definido los parámetros que aparecen como exponentes en las fórmulas de las dos figuras de mérito. El propósito de esta sección consiste en plantear una forma de análisis y experimentación que ayudará a la determinación de valores adecuados para tales parámetros. Primeramente se realizará un análisis reducido para determinar el impacto cuantitativo y cualitativo de la selección de los parámetros α y γ. Posteriormente haremos una evaluación experimental del impacto de la elección de los parámetros en el desempeño de memGA en diferentes problemas. Posteriormente veremos cuál es la sensibilidad en el desempeño del memGA con cierta selección de parámetros cuando cambia la naturaleza del problema. Este capítulo también incluye una comprobación experimental de la forma como memGA evoluciona una población manteniendo óptimos localizados y acercando los individuos paulatinamente más a los óptimos conforme el número de evaluaciones invertidas crece. El capítulo concluye con un breve análisis de sensibilidad del algoritmo ante cambios en el tamaño de la población exploradora.

4.1. Discusión y delimitaciones

Existen diferentes estrategias para analizar el comportamiento de un AG. Entre ellas podemos citar las basadas en modelos de Markov [57, 58, 59]. También existen otros modelos basados en la distribución de esquemas [60], en los tiempos de convergencia [61], y otros que hacen un análisis de la intensidad de la presión selectiva [62]. Pero a veces estas garantías no revelan revelan el comportamiento de un AG en un problema específico de optimización, ello debido a que su desempeño es demasiado dependiente del problema y su codificación. Pero sin querer tomar una validación puramente experimental sobre la presente propuesta, presentaremos argumentos que ayudarán a construir una idea del comportamiento del algoritmo y que pemitirán hacer ajustes que pueden resultar adecuados cuando se aplica a un problema desconocido pero del cual se tienen presentes ciertas características.

Nuestro análisis está orientado hacia la determinación del efecto de la elección de los parámetros sobre la búsqueda en el espacio del problema y sobre administración de

la información almacenada. Nuestro enfoque considera por separado la revisión de las figuras de mérito. Primeramente revisaremos el indicador de reemplazo.

4.2. Figura de reemplazo

Recordamos que el reemplazo de un individuo que radica en memoria por otro recién encontrado es un proceso determinístico basado en el indicador numérico:

$$R(x) = \frac{f(x)}{N_1(x)^{\alpha}}. \tag{4.1}$$

Primero revisemos el indicador N_1. Sabemos que

$$\begin{aligned} N_1(x) &= \textstyle\sum_{z \in P} sh_1(x,z) \\ &= sh_1(x,x) + \textstyle\sum_{z \in P-\{x\}} sh_1(x,z) \\ &= 1 + \textstyle\sum_{z \in P-\{x\}} sh_1(x,z). \end{aligned} \tag{4.2}$$

Debido a que $sh_1(x,z) \geq 0$ para todo z, entonces $N_1(x) \geq 1.0$ para todo x, por consiguiente si $\alpha \geq 0$, el termino $N_1(x)^{\alpha}$ actua como factor de decremento en la fórmula de $R(x)$.
Veamos el significado de que $N_1(x) = 1$. De la fórmula 4.2 se deduciría en este caso que

$$\sum_{z \in P-\{x\}} sh_1(x,z) = 0, \tag{4.3}$$

y por consiguiente

$$sh_1(x,z) = 0 \ \forall z \in P - \{x\}. \tag{4.4}$$

Revisando la definición de sh_1 y recordando la fijación de $a = 2$ en la fórmula de Sh y Sh_1:

$$sh_1(x,y) = \begin{cases} 1 - \left(\frac{d(x,y)}{\sigma_{share}}\right)^2 & \text{si } f(x) < f(y) \text{ y } d(x,y) < \sigma_{share} \\ 0 & \text{en caso contrario.} \end{cases} \tag{4.5}$$

Por consiguiente, $N_1(x) = 1$ implica que para todo y en $P-\{x\}$ $f(x) \geq f(y)$ ó $d(x,y) \geq \sigma_{share}$, es decir que no existe ningún y que esté lo suficientemente cerca de x (es decir que $d(x,y) < \sigma_{share}$) y que a la vez tenga una mejor evaluación que x. De esta forma aquellos individuos presentes en la población P que tienen $N_1(x) = 1$ serían los representantes de la población para ser los óptimos locales que se pretende localizar. Todo ello bajo la suposición de que el valor de σ_{share} está adecuadamente establecido. Este tipo de elementos recibirán un nombre especial.

Definición
Un elemento x en la población P se llamará *significativo* si $N_1(x) = 1$.

Por otro lado si x es un elemento de la población que cumple $N_1(x) \gg 1$, entonces esto significará que

$$\sum_{z \in P-\{x\}} Sh_1(x,z) \gg 1, \tag{4.6}$$

y por consiguiente existe en la población un número suficientemente grande de elementos y que cumplen $Sh_1(x,y) > 0$, es decir que existe un número suficientemente grande de elementos que están suficientemente cerca de x ($(d(x,y) < \sigma_{share})$ y que son mejores que x ($f(x) \leq f(y)$). Desde un punto utilitario en la solución reportada, estos elementos no aportan información adicional a la solución. Debido a esto la solución reportada por memGA consistirá sólo de aquellos elementos que tengan un contador de nicho modificado relativamente bajo; el límite lo fijaremos por regla empírica en 2. Los individuos con un contador alto recibirán un nombre especial.

Definición

Un elemento x en la población P se llamará *poco significativo* si $N_1(x) \gg 1$.

4.2.1. Competencias entre elementos

La competencia entre elementos por permanecer en la memoria se decide haciendo uso del indicador R. De acuerdo a las definiciones anteriores las competencias podrían hacerse en la siguiente forma:

- significativo vs significativo,
- significativo vs poco significativo,
- poco significativo vs poco significativo.

Puesto que los elementos significativos representan los óptimos locales, deseamos que tengan permanencia en la población memoria, y por consiguiente estamos interesados en situaciones donde puedan perder una competencia.

Competencia: significativo contra significativo

Supongamos que x y y son dos elementos significativos en la población memoria P. Así $N_1(x) = N(y) = 1$, y por consiguiente $R(x) = f(x)$ y $R(y) = f(y)$, es decir que los dos individuos, o picos si nos toleramos esa libertad de nombrarlos así, compiten por permancer utilizando como criterio la altura de los picos. A pesar de que esto es razonable, tiene como desventaja la pérdida de representatividad en la población del pico que resulte perdedor. Una competencia en esta situación ocurre cuando el tamaño de la población resulta insuficiente para representar la totalidad de los óptimos locales para el problema que esté siendo resuelto. Dos acciones podrían tomarse en caso que se monitoreen estas competencias. Decidir aumentar el tamaño de la población para no perder ninguno de estos individuos, o bajo el reconocimiento de que picos por debajo de un cierto umbral no interesan, dejar que la competencia los elimine. En la estrategia desarrollada en esta investigación hemos optado por dejar que la competencia los elimine.

Competencia: significativo contra poco significativo

Supongamos que x es un elemento significativo y que y no lo es. Por consiguiente, debe existir en la población otro elemento y_o suficientemente cerca de y ($d(y, y_o) < \sigma_{share}$) y con una mejor evaluación. Reconocemos dos alternativa: que el elemento y sea una copia de y_o y otra en la cual no lo es.

Competencia: significativo contra copias de otro significativo

Si y es una copia de y_o entonces

$$\begin{aligned} N_1(y) &= \textstyle\sum_{z\in P} sh_1(d(y,z)) \\ &= sh_1(d(y,y)) + sh_1(d(y,y_o)) + \textstyle\sum_{z\in P-\{y,y_o\}} Sh_1(y,z) \qquad (4.7) \\ &= 2 + \textstyle\sum_{z\in P-\{x\}} sh_1(d(x,z)). \end{aligned}$$

Por consiguiente $N_1(y) \geq 2$ y así la competencia entre x y y se resuelve a favor de y si

$$R(x) = f(x) > \frac{f(y_o)}{2^\alpha} = \frac{f(y)}{2^\alpha} > \frac{f(y)}{N_1(y)^\alpha} = R(y), \qquad (4.8)$$

es decir si

$$\frac{f(x)}{f(y_o)} > \frac{1}{2^\alpha}. \qquad (4.9)$$

Es decir, que el representante de la población de un pico compite ganando contra las copias del representante de otro pico cuando la relación entre la altura menor contra la del mayor es mayor que la fracción $1/2^\alpha$. Por ejemplo, para $\alpha = 2$, los picos que están a un 25 % de la altura del pico mayor sobreviven cuando compiten contra copias del pico mayor. Mientras que para $\alpha = 3$, los picos que están a un 12.5 % también lo hacen. Como podemos observar valores relativamente grandes de α permiten la permanencia en la población memoria de una buena cantidad de picos menores cuando la población es lo suficientemente grande. En la versión primera del problema de optimización multimodal estamos interesados en todos los picos, sin importar qué tan abajo estén de los óptimos globales; en situaciones prácticas ciertamente no. A menos que los picos menores contengan los bloques constructores para alcanzar los óptimos globales y en aquellos que pudieramos estar interesados. Esta situación es compleja de analizar. En la parte experimental veremos su desempeño.

Competencia: significativo contra individuos cercanos a picos mayores

La otra alternativa de la competencia es cuando y no es una copia del pico en cuya vecindad está. Supongamos que y se encuentra a una distancia relativa r del óptimo y_o. Es decir, que $d(x,y) = r\,\sigma_{share}$, donde $0 < r < 1$. Así

$$\begin{aligned} N_1(y) &= \textstyle\sum_{z\in P} sh_1(d(y,z)) \\ &= sh_1(d(y,y)) + sh_1(d(y,y_o)) + \textstyle\sum_{z\in P-\{y,y_o\}} sh_1(y,z), \end{aligned} \qquad (4.10)$$

donde

$$\begin{aligned} sh_1(d(y,y_o)) &= 1 - \left(\frac{d(y,y_o)}{\sigma_{share}}\right)^2 \\ &= 1 - r^2, \end{aligned} \tag{4.11}$$

por consiguiente

$$N_1(y) = 2 - r^2 + \sum_{z \in P - \{y,y_o\}} sh_1(d(y,z)). \tag{4.12}$$

Por tanto

$$R(y) = \frac{f(y)}{\left(2 - r^2 + \sum_{z \in P-\{y,y_o\}} sh_1(d(y,z))\right)^{\alpha}}, \tag{4.13}$$

y así la competencia entre x y y se decidirá a favor de x si

$$f(x) \geq \frac{f(y)}{(2-r^2)^{\alpha}} > \frac{f(y)}{\left(2 - r^2 + \sum_{z \in P-\{y,y_o\}} sh_1(d(y,z))\right)^{\alpha}}. \tag{4.14}$$

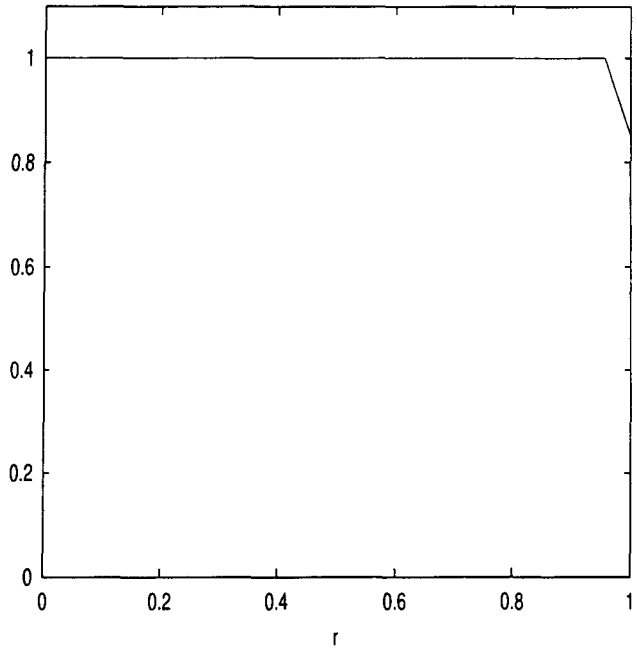
Rearreglando la ecuación anterior y bajo la suposición que $f(y_o) > 0$ obtenemos

$$\frac{f(x)}{f(y_o)} \geq \frac{f(y)}{f(y_o)} \frac{1}{(2-r^2)^{\alpha}}. \tag{4.15}$$

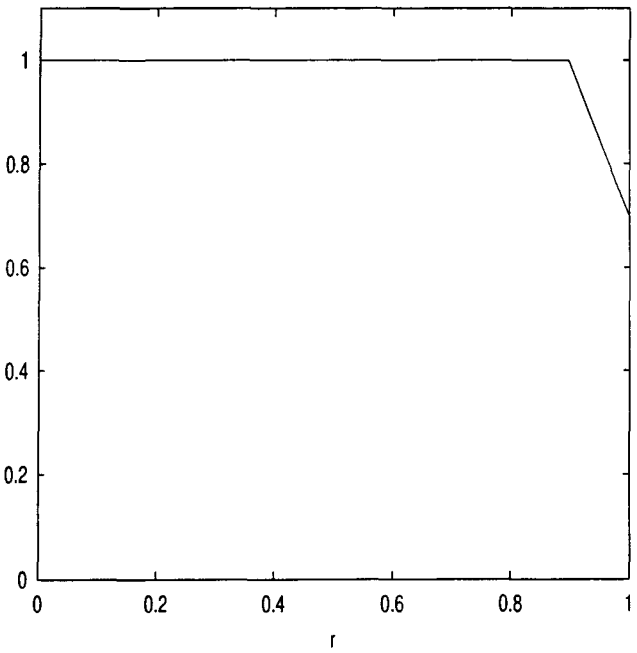
La relación $f(x)/f(y_o)$ se interpreta como la relación entre la altura del pico menor y la altura del pico mayor. Mientras que $f(y)/f(y_o)$, cuando se relaciona con la variable r, es el perfil de decaimiento de la función en el pico asociado a y_o. Con estos elementos leemos la desigualdad anterior en la siguiente forma. Para que el representante de un pico menor x gane la competencia contra un elemento que no es representante de un pico mayor una condición es que el perfil de decaimiento de la función alrededor del pico mayor cumpla

$$\frac{f(y)}{f(y_o)} \leq (2-r^2)^{\alpha} \frac{f(x)}{f(y_o)}. \tag{4.16}$$

Por ejemplo para los valores $f(x)/f(y_o) = a = 0.85, 0.65, 0.35, 0.25$ y $\alpha = 2.0$ el perfil de la función, pensémoslo simétrico en el espacio dimensional adecuado, debe quedar por debajo de la curva que ilustran las figuras 4.1 y 4.2. Como pondemos ver de las figuras 4.1 y 4.2, para picos que no están muy por abajo de un óptimo global, la permanencia de óptimos locales menores ocurre aún para picos que tienen una forma casi plana. Picos definitivamente muy por debajo de un óptimo local perderán competencias por permanecer contra elementos que están en las vecindades de los óptimos locales. Esto es bajo la suposición de que los óptimos locales estén plenamente identificados. Es decir, un individuo que sea representante de un óptimo local pero que no alcance la evaluación máxima del pico sino un porcentaje batallará aún más en la competencia por permancer cuando enfrente a los individuos que están en las vecindades de los óptimos globales.

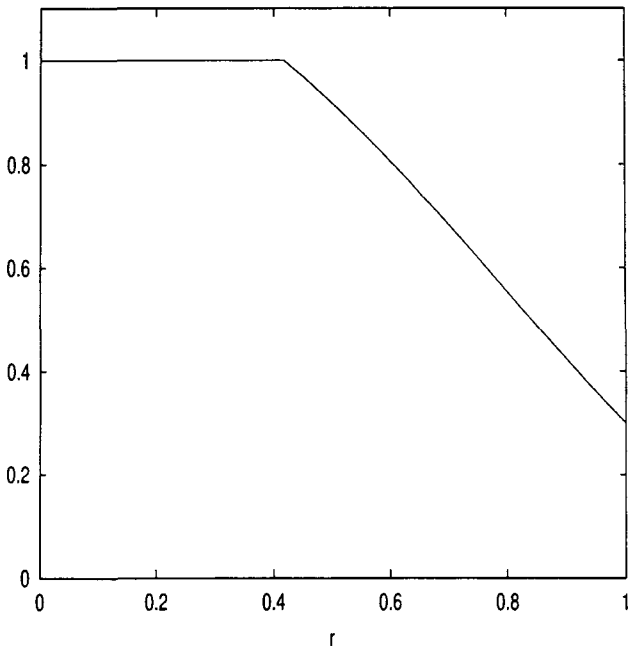


(a) $a = 0.85$

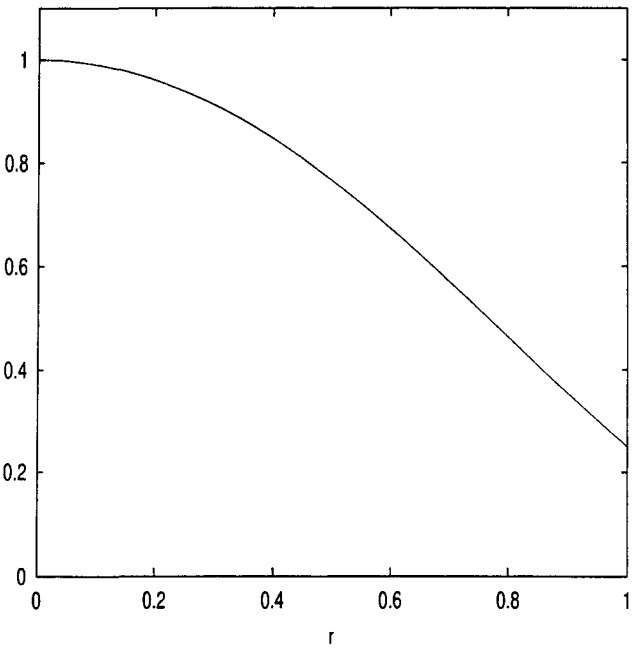


(b) $a = 0.70$

Figura 4.1: Perfiles de la función a optimizar alrededor de y_o abajo de los cuales existe sobrevivencia del representante de un pico a veces menor.



(a) $a = 0.30$



(b) $a = 0.25$

Figura 4.2: Perfiles de la función a optimizar alrededor de y_o abajo de los cuales existe sobrevivencia del representante de un pico a veces menor (continuación).

4.2.2. Resumen

En esta sección hemos analizado el comportamiento de la retención de óptimos locales ante variaciones del parámetro α. Hemos visto que valores grandes de tal parámetro, digamos por ejemplo $\alpha \geq 3$, permiten la subsistencia de óptimos locales que están muy por abajo de los óptimos globales. Y que para valores pequeños los picos pequeños pierden la competencia ante copias de los picos mayores. Hemos visto también que la forma o perfil de los picos mayores puede afectar la competencia. Esto significará que, ante un tamaño de población adecuado, la diversidad de la población aumenta al aumentar el parámetro α. Y que valores relativamente pequeños aumentarán la posibilidad de tener varios representantes en los óptimos. Es decir el parámetro α es un parámetro de control sobre la diversidad de la población memoria, y por consiguiente de la solución que el algoritmo entregará.

4.3. Figura de selección

La figura de selección de la estrategia propuesta está dada por la fórmula:

$$S(x) = \frac{f(x)}{N(x)\,C(x)^{\gamma}}. \tag{4.17}$$

En el marco de la sección anterior hemos visto que para valores de α grandes el número de individuos cercanos a los óptimos disminuye sensiblemente debido a que ellos deberían de competir en condiciones muy desiguales contra representantes de picos más pequeños. De tal forma que podemos asumir que

$$N(x) \approx 1. \tag{4.18}$$

Estamos interesados en poder comparar el número de veces que es seleccionado un pico respecto al número de veces que es seleccionado otro que tiene diferente altura. Siendo específicos, queremos determinar el comportamiento de la proporción del número de selecciones de uno entre el número de selecciones del otro. Una de las preguntas que puede hacerse es sobre si el comportamiento resulta caótico o si acaso presenta signos de convergencia. Supongamos que x y y son los representantes únicos de sus picos y que se seleccionará uno cada vez. El indicador usando para la selección será:

$$S(x) = \frac{f(x)}{N(x)\,C(x)^{\gamma}} \quad \text{y} \quad S(y) = \frac{f(y)}{N(y)\,C(y)^{\gamma}}. \tag{4.19}$$

Bajo la suposicón $N(x) \approx 1$ y $N(y) \approx 1$ las relaciones anteriores quedan:

$$S(x) = \frac{f(x)}{C(x)^{\gamma}} \text{ y } S(y) = \frac{f(y)}{C(y)^{\gamma}}. \tag{4.20}$$

Si restringimos la competencia por seleccionamiento sólo entre x y y y recordamos que el proceso de selección utiliza selección proporcional sobre dichos indicadores entonces

la probabilidad de elegir x será:

$$P(x) = \frac{\frac{f(x)}{C(x)^\gamma}}{\frac{f(x)}{C(x)^\gamma} + \frac{f(y)}{C(y)^\gamma}}. \tag{4.21}$$

Si dividimos entre $f(x)/C(x)^\gamma$ numerador y denominador el lado derecho de la ecuación anterior se obtiene la expresión

$$P(x) = \frac{1}{1 + \frac{f(y)}{f(x)}\left(\frac{(x)}{C(y)}\right)^\gamma}. \tag{4.22}$$

Si definimos $f(y)/f(x) = r$ la ecuación anterior queda

$$P(x) = \frac{1}{1 + r\left(\frac{C(x)}{C(y)}\right)^\gamma}. \tag{4.23}$$

Mientras que la probabilidad de seleccionar y quedará

$$P(y) = 1 - P(x). \tag{4.24}$$

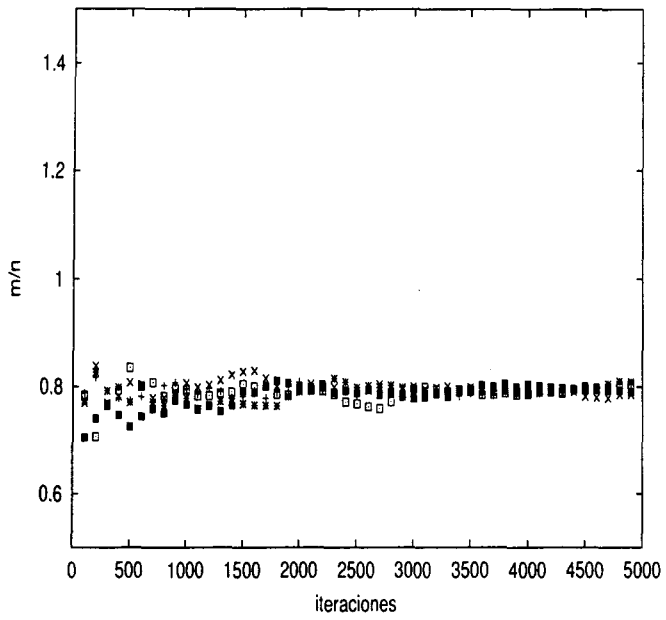
Simplifiquemos la notación tomando $n = C(x)$ y $m = C(y)$. De esta forma podemos imaginar el plano $n - m$; un punto *inicial* (n_o, m_o) correspondería a una configuración donde n_o sería igual al número de veces que fue seleccionado anteriormente el representante x, y m_o sería el correspondiente de y, Seleccionar el individuo x equivaldría a moverse al punto $(n_o + 1, m_o)$, mientras que seleccionar y equivaldría a moverse al punto $(n_o, m_o + 1)$. Asimismo las probabilidades de hacer un movimiento en el eje n sería

$$P(n \to n+1) = \frac{1}{1 + r\left(\frac{n}{m}\right)^\gamma}, \tag{4.25}$$

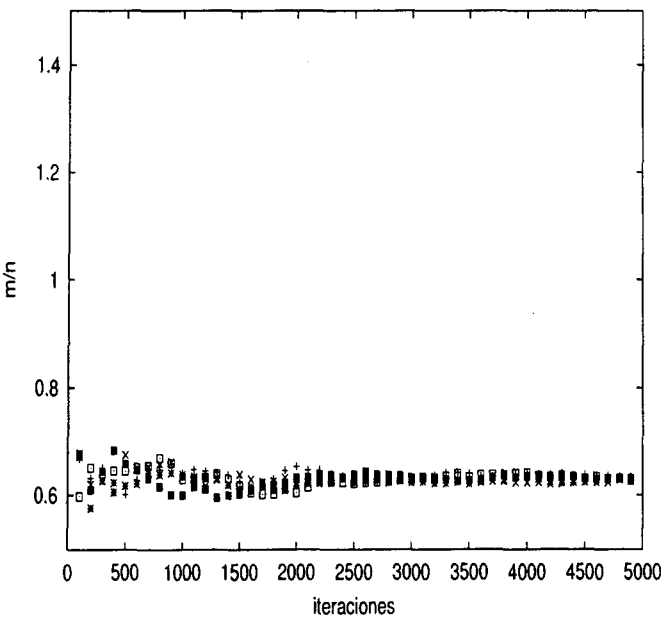
mientras que la de moverse en el eje m sería

$$P(m \to m+1) = 1 - P(n \to n+1). \tag{4.26}$$

Las ecuaciones 4.25 y 4.26 definen el movimiento estocástico de un punto en el plano $n - m$. En este contexto estamos interesados en saber cuál es el comportamiento de la relación m/n cuando n es grande. Las figuras 4.3, 4.4 y 4.5 representan diferentes simulaciones partiendo de puntos aleatorios en el plano $n - m$ y para diferentes valores de r y γ. Como podemos observar de las figuras 4.3, 4.4, y 4.5 existe evidencia experimental de un comportamiento convergente del punto (n, m) a una línea del plano $n - m$ cuya pendiente es precisamente la relación que con la cual se prefiere el pico x sobre el pico y sabiendo la relación entre sus alturas, r. Alrededor de esta *línea de equilibrio* la trayectoria del punto oscila, lo cual nos hace pensar que la probabilidad de moverse de un lado hacia el otro de esa recta inclinada es la misma. Utilicemos la figura 4.6 como referencia para determinar la relación entre $P_n = P(n \to n+1)$ y $P_m = P(m \to m+1)$; bajo la suposición de que las probabilidades de moverse hacia

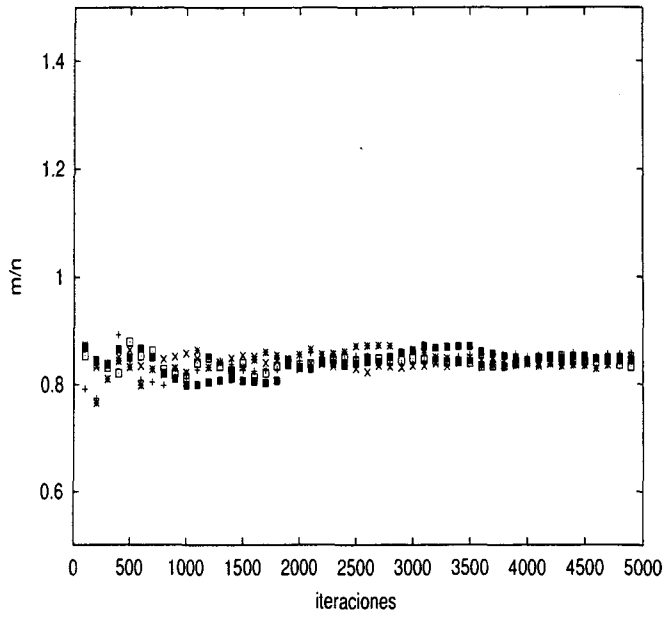


(a) $\gamma = 2, r = 0.5$

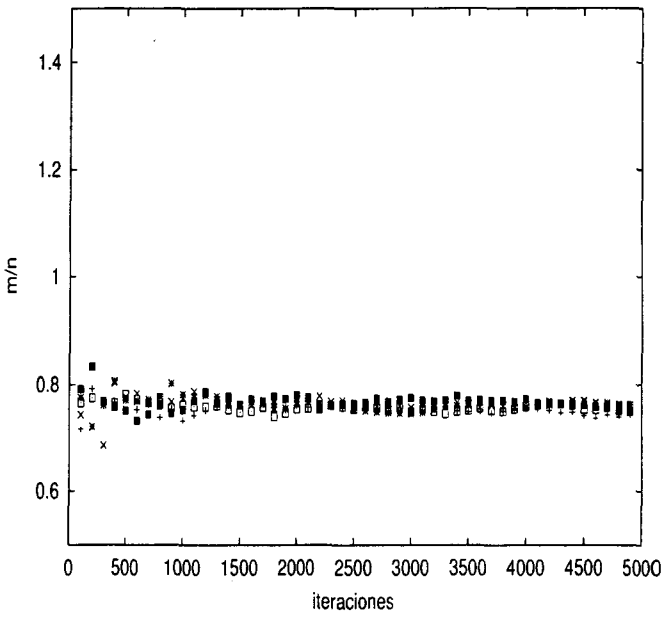


(b) $\gamma = 2, r = 0.25$

Figura 4.3: Proporción m/n de las trayectorias del punto (n, m) sujeto a la figura de selección S realizando simulaciones aleatorias con parámetros fijos.

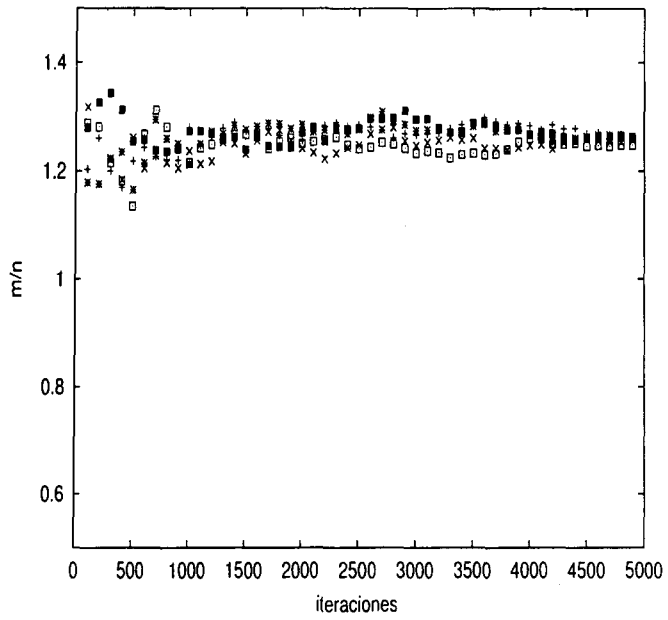


(a) $\gamma = 3, r = 0.5$

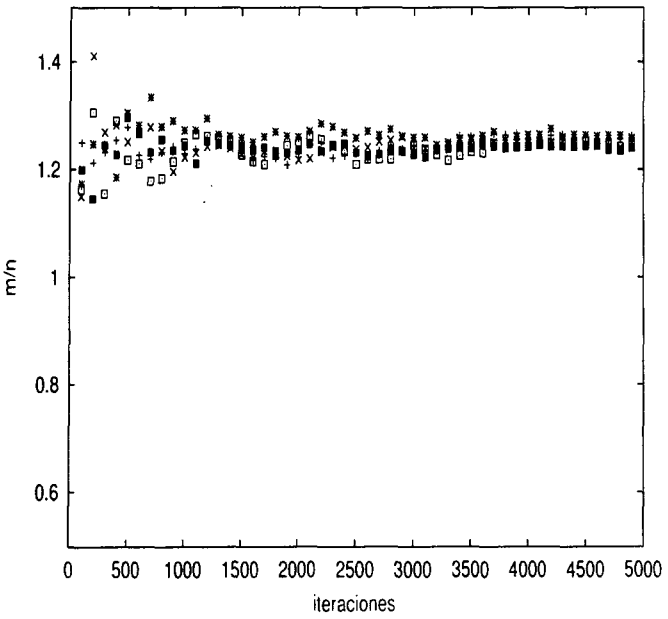


(b) $\gamma = 4, r = 0.25$

Figura 4.4: Proporción m/n de las trayectorias del punto (n, m) sujeto a la figura de selección S realizando simulaciones aleatorias con parámetros fijos (continuación).



(a) $\gamma = 2, r = 2.0$



(b) $\gamma = 4, r = 3.0$

Figura 4.5: Proporción m/n de las trayectorias del punto (n, m) sujeto a la figura de selección S realizando simulaciones aleatorias con parámetros fijos (continuación).

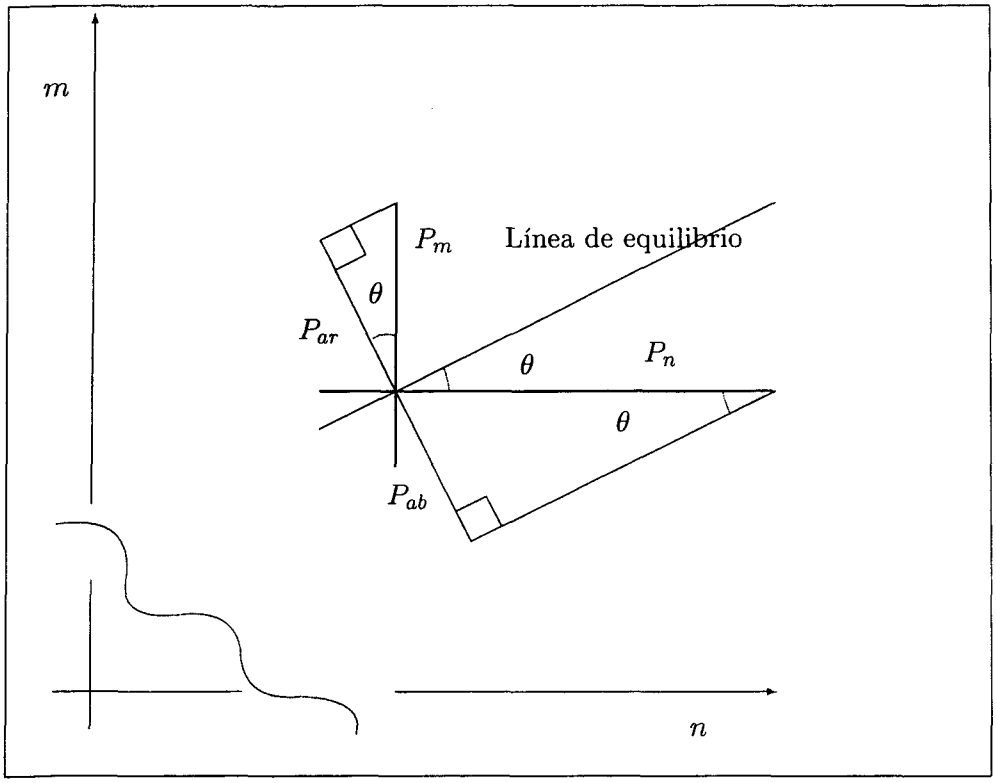


Figura 4.6: Diagrama que relaciona las probabilidades de subir o bajar sobre la línea de equilibrio y las probabilidades P_m y P_n

arriba de la línea de equilibrio, P_{ar}, y de moverse hacia abajo de ella, P_{ab}, sean iguales. Del triángulo superior de la figura 4.6 se deduce que :

$$P_{arr} = P_m \cos(\theta), \tag{4.27}$$

y del inferior que

$$P_{ab} = P_n \text{sen}(\theta). \tag{4.28}$$

Bajo la suposición de que ambas probabilidades son iguales se deduce que

$$P_m \cos(\theta) = P_n \text{sen}(\theta), \tag{4.29}$$

de donde

$$\begin{aligned} \tan(\theta) &= \frac{P_m}{P_n} \\ &= \frac{\frac{r\left(\frac{n}{m}\right)^\gamma}{1+r\left(\frac{n}{m}\right)^\gamma}}{\frac{1}{1+r\left(\frac{n}{m}\right)^\gamma}} \\ &= r\left(\frac{n}{m}\right)^\gamma. \end{aligned} \tag{4.30}$$

Por otro lado, la línea de equilibrio en el plano $n - m$ debería ser de la forma

$$m = \tan(\theta)\, n + b, \tag{4.31}$$

para alguna constante real b. De donde

$$\tan(\theta) = \frac{m-b}{n} = \frac{m}{n} - \frac{b}{n}. \tag{4.32}$$

Para valores grandes de n el término b/n resulta pequeño pues b está fijo, de tal manera que la ecuación anterior quedaría como

$$\frac{m}{n} \approx \tan(\theta) \text{ ó } \frac{n}{m} \approx \cot(\theta). \tag{4.33}$$

La cual, al ser sustituida en la ecuación 4.30 nos da la ecuación

$$\tan(\theta) \approx r \cot(\theta)^\gamma, \tag{4.34}$$

y por consiguiente, para valores grandes de n y m, los puntos de la recta de que satisfacen que la probabilidad de salir perpendicularmente hacia arriba y la probabilidad de salir perpendicularmente hacia abajo satisfacen la relación

$$\frac{m}{n} \approx \tan(\theta) \approx r^{\frac{1}{\gamma+1}}, \tag{4.35}$$

o simplemente

$$\frac{m}{n} \approx \sqrt[\gamma+1]{r}. \tag{4.36}$$

γ	r	$\sqrt[\gamma+1]{r}$
2.0	0.50	0.794
2.0	0.25	0.629
3.0	0.50	0.841
4.0	0.25	0.758
2.0	2.00	1.260
4.0	3.00	1.246

Tabla 4.1: Pendiente estimada de la línea de equilibrio por la fórmula 4.36 para los valores de r y γ que aparecen en las simulaciones de las gráficas 4.3, 4.4, y 4.5.

Para validar experimentalmente dicho resultado usamos los datos y gráficas ilustradas en las figuras 4.3 a 4.5. La tabla 4.1 ilustra los valores de la fórmula anterior para los valores de γ y r de dichas gráficas. Las figuras 4.7, 4.8 y 4.7 comparan las simulaciones hechas y graficadas en 4.3,4.4, y 4.5 contra el límite estimado. Por cuestiones de visibilidad sólo se han graficado dos de las cinco simulaciones hechas en aquel caso. Como podemos ver existe congruencia entre las estimaciones y el comportamiento de las simulaciones.

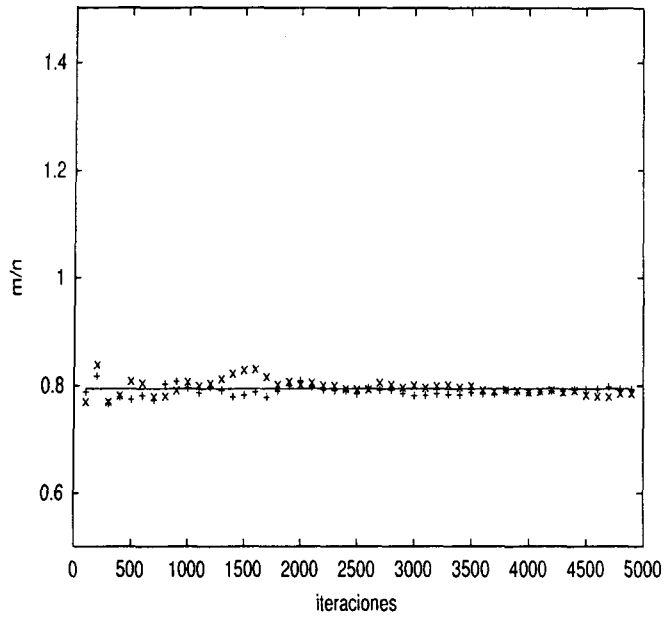
Algunas de las conclusiones a que se pueden llegar a partir de la fórmula 4.36 son las siguientes.

- Que efectivamente picos mayores que otros tienen mayores probabilidades de ser selecionados para conformar la población exploradora. Esto se deduce de la ecuación 4.36, debido a que si $f(x) > f(y)$, entonces $r = f(y)/f(x) < 1$ y así $\sqrt[\gamma+1]{r} < 1$ el cual correspondería a la proporción esperada a largo plazo entre el número de selecciones de y respecto al número de selecciones de x.

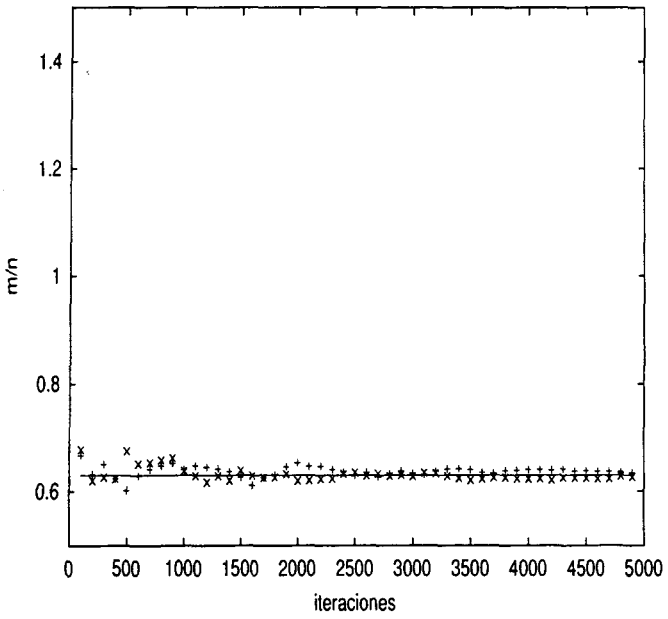
- Que el parámetro γ puede actuar como un regulador en la diversidad en la búsqueda. Debido a que para valores de $r < 1$ y $0 < \gamma$, se obtendría que $r < \sqrt[\gamma+1]{r}$, es decir, que la probabilidad de seleccionar picos menores se puede aumentar incrementando el valor γ. Por ejemplo, si se tratara de comparar las selecciones entre dos picos, uno con una altura 10 % del mayor $r = 0.1$. Para el valor $\gamma = 2.0$, $\sqrt[\gamma+1]{r} = 0.464$, es decir que en el largo plazo el número de veces que sería seleccionado el mayor pico es ligeramente mayor que el doble de veces (2.12 veces) que es seleccionado el menor. Mientras que para $\gamma = 4$, el porcentaje decrece a 1.28 veces.

4.3.1. Resumen

Uno de los problemas previstos al retener individuos bien evaluados en memoria fue el posible enviciamiento de la búsqueda hacia las regiones del espacio donde ellos radican. Un medio previsto para solucionar esta problemática negativa en la búsqueda fue la introducción de un contador de las veces que un individuo ha sido seleccionado

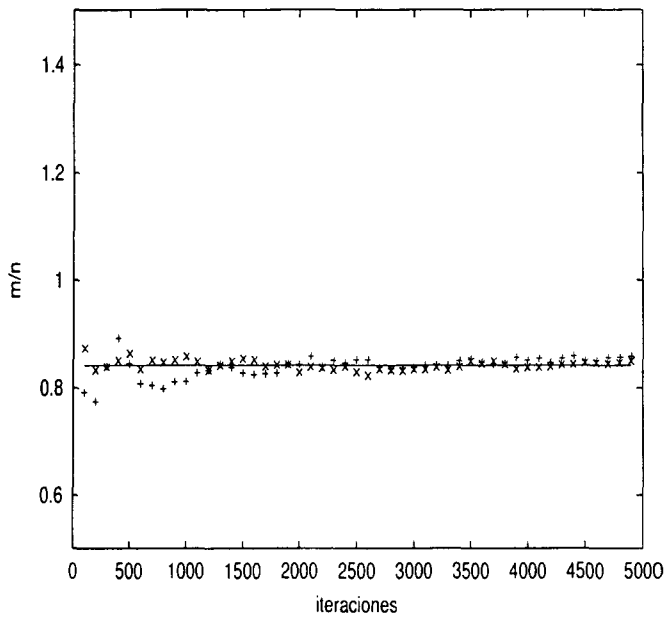


(a) $\gamma = 2, r = 0.5$

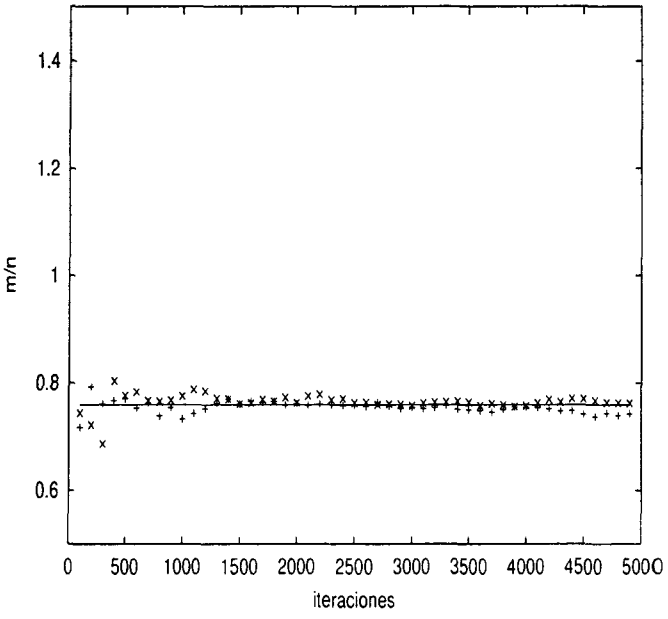


(b) $\gamma = 2, r = 0.25$

Figura 4.7: Algunas de las simulaciones contenidas en las gráficas de la figura 4.3 junto con los correspondientes límites estimados por la fórmula 4.36.

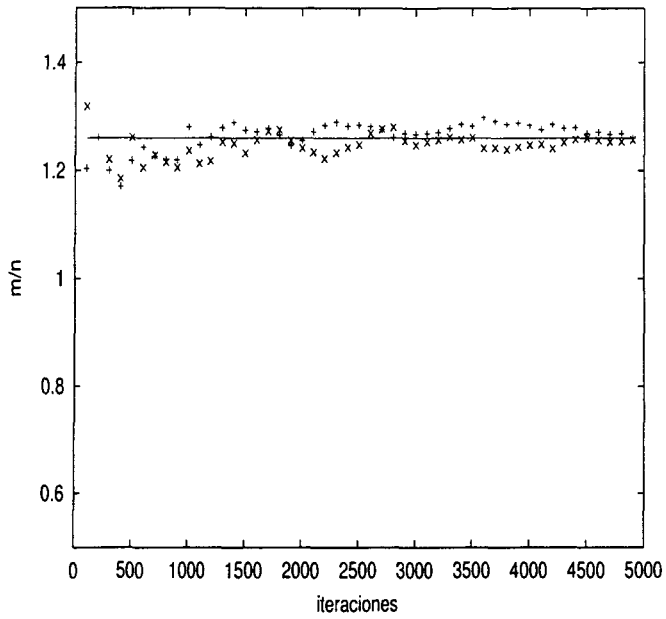


(a) $\gamma = 3, r = 0.5$

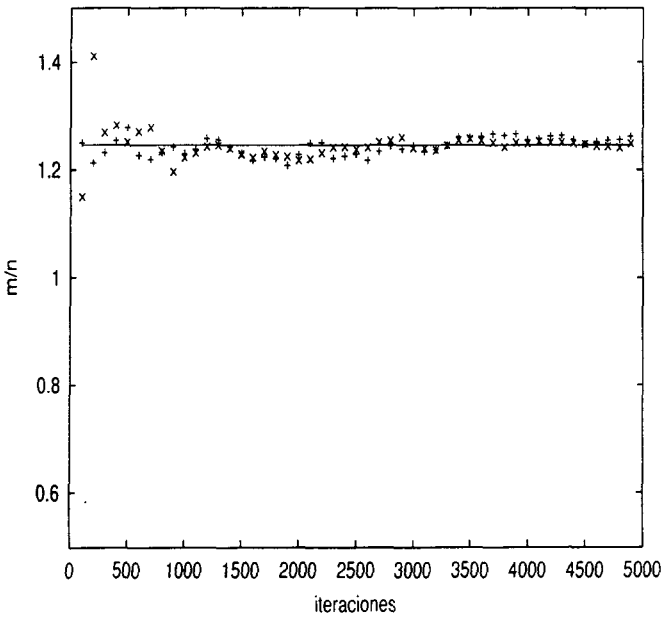


(b) $\gamma = 4, r = 0.25$

Figura 4.8: Algunas de las simulaciones contenidas en las gráficas de la figura 4.4 junto con los correspondientes límites estimados por la fórmula 4.36 (continuación).



(a) $\gamma = 2, r = 2.0$



(b) $\gamma = 4, r = 3.0$

Figura 4.9: Algunas de las simulaciones contenidas en las gráficas de la figura 4.5 junto con los correspondientes límites estimados por la fórmula 4.36.

para constituir la población exploradora. La primera intuición fue que este contador debía de actuar como un agente de penalización en la figura de selección. El nivel de penalización se estableció como un parámetro a determinar. En este contexto, la pregunta que se formuló entonces fue cuál sería la distribución de las selecciones sobre los individuos en memoria. Las alternativas que se plantearon fueron sobre si el comportamiento llegaría a ser caótico y presentaría alguna forma de convergencia.

En este apartado hemos hecho un análisis parcial del nivel de penalización del contador de selecciones dentro de la figura de mérito para seleccionar individuos en memoria. Las conclusiones a las cuales se llegaron después de este análisis fueron las siguientes. Que cuando se comparan las selecciones entre picos de diferente altura, el comportamiento inducido no es caótico; más bien presenta un nivel de convergencia a un valor que depende en forma directa de la proporción que guardan las alturas de los picos. Es decir, que picos mayores tendrán mayor probabilidad de ser seleccionados, y que esta proporción puede ser regulada por el parámetro γ, de tal suerte que picos muy por debajo de los óptimos globales pueden tener no tan bajas probabilidades de ser seleccionados para valores de γ grandes. Esto impacta la diversidad de la búsqueda.

4.4. Experimentación

En la sección anterior hemos visto que los parámetros α y γ tienen efectos sobre la diversidad de la respuesta y la diversidad de la búsqueda. Sin embargo, la pregunta que parece razonable es ¿qué tanta diversidad es necesaria para resolver problemas particulares?

Con el fin de tener una estimación de las diversidades adecuadas haremos algunas comparaciones experimentales de diferentes combinaciones entre los parámetros. Para ello utilizaremos un problema prestablecido de optimización multimodal viendo cuáles serían valores adecuados para los parámetros α y γ, y posteriormente revisaremos cuál es la sensibilidad de la elección ante cambios en la función problema.

4.4.1. Marco de referencia experimental para la comparación

El problema que utilizaremos para iniciar la experimentación que permita una comparación entre diferentes selecciones para los parámetros α y γ es la función senoidal definida en el marco de referencia experimental $M_2(x)$. Esta función fue definida en el capítulo 2 por la fórmula 2.8, pero aquí la repetiremos; la ecuación que la define es:

$$M_2(x) = e^{-2\ln(2)\left(\frac{x-0.1}{0.8}\right)^2} \operatorname{sen}^6(5\,\pi\,x). \tag{4.37}$$

Recordemos que el dominio de la función se restringe al intervalo $[0, 1]$. Para aplicar el algoritmo genético, los números en el dominio serán codificados mediante un cromosoma binario de longitud 30. La función de codificación será lineal; esto es, primeramente el cromosoma será convertido al número entero que representa la cadena binaria y posteriormente será dividido entre el número $2^{30} - 1$. La función tiene 5 óptimos locales los cuales son aproximadamente los números 0.1, 0.3, 0.5, 0.7, y 0.7.

4.4.2. Modelos a comparar

El conjunto de valores numéricos de donde tomaremos los parámetros usados en las figuras de mérito será el conjunto

$$V = \{1.00, 2.00, 4.00, 8.00, 16.0\}. \tag{4.38}$$

Recordemos los indicadores de reemplazo y selección definidos. La figura de mérito para el reemplazo está dada por la función

$$R(x) = \frac{f(x)}{N_1(x)^{\alpha}}. \tag{4.39}$$

La figura de métiro para la selección es

$$S(x) = \frac{f(x)}{N(x)C(x)^{\gamma}}. \tag{4.40}$$

Para simplificar la referencia al algoritmo propuesto aceptaremos que la notación $\text{memGA}_{\alpha_o,\gamma_o}$ hace referencia a él donde el parámetro $\alpha = \alpha_o$ y el parámetro $\gamma = \gamma_o$. Con el muestreo que hemos definido tenemos una familia de 25 algoritmos a considerar.

4.4.3. Diseño de experimentos

Habiendo planteado la función que servirá como marco de referencia y los algoritmos a comparar lo siguiente es diseñar los experimentos. El objetivo de esta comparación es la determinación de aquel o aquellos modelos que tienen un *mejor* desempeño en este problema. Para realizar esta comparación utilizaremos la estrategia de comparación utilizada por Mahfoud [36] la cual posee dos elementos principales.

Calidad de la solución entregada

Uno de los principales criterios para comparar algoritmos tiene que ver con la calidad de las respuesta entregada por el algoritmo. Esta calidad será medida por el esfuerzo computacional invertido por un algoritmo de optimización local cuando aplica a la solución entregada. El esfuerzo computacional es medido por el número de evaluaciones de la función objetivo utilizadas para *mover* los puntos en la solución proporcionada hacia los óptimos locales más próximos. El proceso de optimización local utilizado es el procedimiento definido por Mahfoud [36], el cual implementa el buscador de alpinista del *próximo paso* previamente desarrollado por Mühlenbein [63] y que aparece descrito en la figura 4.10, cuando el espacio de fenotipos es un subconjunto de $\Re^n$. Si el fenotipo es el mismo cromosoma binario el optimizador local se reduce a la experimentación de todos los posibles alelos, iniciando en una posición aleatoria del cromosoma. Nos referiremos a esta estrategia como el algoritmo PHC. En el caso de fenotipos en variables reales la estrategia de optimización local inicia fijando una vecindad relativamente grande. Cada elemento de la población *intenta mejorar* cambiándose posiblemente por otro. El intento por mejorar en un individuo consiste en generar un nuevo individuo

sumándole o restándole el radio de la vecindad en alguna de las variables codificadas en el fenotipo del individuo. Todas las variables codificadas son intentadas; se recorren todas ellas en forma cíclica iniciando en una variable escogida aleatoriamente. Si el nuevo individuo tiene una evaluación mayor que el inicial, éste lo reemplaza. En caso que tenga una menor o igual evaluación no hay reemplazo. Este proceso concluye hasta que no existe reemplazo para ningún elemento en la población. Cuando esto ocurre la distancia de búsqueda es dividida a la mitad y el proceso de búsqueda por individuos mejores se repite. Esta repetición avanza hasta que el radio de vecindad utilizado es menor que el menor incremento codificado por el AG. Este número se designa en el algoritmo por ϵ, y en el caso de la codificación lineal es $(b-a)/(2^l-1)$, donde $[a,b]$ corresponde al espacio del fenotipo y l es la longitud del cromosoma. El valor inicial del radio de la vecindad N_s se fija en el valor de σ_{share}, y la idea principal es que un individuo converja hacia el óptimo local más cercano; un valor mayor podría hacer que el punto se moviera hacia la vecindad de otro óptimo local y de allí quizá la convergencia a él. Un valor mucho menor podría hacer que el número de pasos para que un punto se mueva al óptimo local más cercano sea grande. El algoritmo entrega la población de puntos hacia donde la población se han movido, y además una contabilización total del número de evaluaciones realizadas; inclusive aquellas que no han conducido a un reemplazo.

Recursos utilizados por una estrategia

El segundo elemento a considerar son los recursos computacionales utilizados por una estrategia. Los criterios de desempeño que utilizaremos serán el tamaño de población y el número de evaluaciones utilizadas. Es decir, que una alternativa que minimiza estos dos recursos simultáneamente sería considerada mejor. Esta situación se inscribe plenamente en el concepto del problema de optimización multiobjetivo. Pero de momento no entraremos en posibles detalles.

Para determinar los recursos computacionales utilizados por un algoritmo utilizaremos la metodología adaptada por Mahfoud [36] que fue introducida en el capítulo anterior y cuya descripción aparece en la figura 3.1. Recordemos que el proceso procedía aplicando la estrategia a considerar sobre una población reducida. La población era evolucionada por la estrategia hasta que se detectaba convergencia. Sobre la población entregada se aplicaba un proceso para determinar si era aceptada como solución al problema multimodal. En caso negativo, la población inicial a la estrategia, y no la final, era aumentada con una población aleatoria del mismo tamaño; la estrategia se aplicaba a esta nueva población nuevamente. En caso que la población sea aceptada como solución se contabilizaban los recursos: tamaño de población, número de evaluaciones, y número de generaciones usadas por la estrategia. El criterio de paro utilizado será el definido por la fórmula 3.1, el cual revisa la diferencia entre el promedio de la evaluación de la población y el promedio del promedio correspondiente al de las cuatro últimas generaciones. Si la diferencia era inferior de la cantidad s_o el criterio de paro se habilitaba. El criterio de aceptación de un conjunto como solución al problema multimodal determinaba si para cada uno de los elementos del conjunto conocido de

1. Inicialice el tamaño de la vecindad N_s.
2. Mientras que $N_s \geq \epsilon$ haga:
   - *a)* Para cada uno de los elementos de la población haga:
     - Escoja aleatoriamente una variable en el fenotipo como la primera y ordénalas para recorrerlas todas en forma cíclica
     - Defina *cambio* como 1
     - Mientras que *cambio* valga 1 haga:
       - Defina *cambio* como 0
       - Para cada variable en el fenotipo haga:
         - Si adicionando N_s a la variable seleccionada en el fenotipo del individuo, la evaluación del individuo aumenta entonces
           - Cambie el invididuo
           - Defina *cambio* como 1
         - Si restando N_s a la variable seleccionada en el fenotipo del individuo, la evaluación del individuo aumenta entonces
           - Cambie el invididuo
           - Defina *cambio* como 1
   - *b)* Defina N_s como $N_s/2$

Figura 4.10: Seudocódigo para el optimizador local paralelo (PHC) utilizado sobre el fenotipo real. Se asumen incrementos o decrementos iguales sobre las variables en el fenotipo. El algoritmo entrega los puntos hacia los cuales se han *movido* los individuos de la población de entrada; el algoritmo contabiliza el número de evaluaciones hechas.

	Tamaño de población al resolver M_2				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	32.6(13.4)	30.7(16.6)	31.4(11.8)	35.2(17.9)	33.3(13.0)
2.0	25.9(14.1)	24.0(11.8)	26.2(14.3)	25.9(8.4)	25.0(14.5)
4.0	23.4(11.8)	20.5(11.6)	19.8(8.0)	19.5(8.3)	20.8(9.8)
8.0	17.9(8.7)	**15.4**(7.3)	19.8(12.7)	17.9(11.1)	19.5(7.3)
16.0	16.6(6.5)	19.8(15.5)	18.6(8.2)	21.8(11.9)	17.6(8.0)

Tabla 4.2: Promedio sobre 50 simulaciones independientes del tamaño de población utilizado por memGA para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_2. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

óptimos locales al problema existía al menos un elemento en el conjunto propuesto como solución cuya distancia a ese óptimo fuese menor que la distancia predefinida r_o.

Para determinar cuál de las variantes $\text{memGA}_{\alpha,\gamma}$ tiene un mejor desempeño utilizaremos la estrategia construida por Mahfoud [36], en la cual primeramente se aplicaba la estrategia de determinación de recursos y sobre el conjunto solución se aplicaba el proceso PHC. Los resultados sumarizados totales contabilizaban el tamaño de población requerido, el número de evaluaciones usadas por el AG, y el número de evaluaciones totales las cuales suman las evaluaciones del AG y las evaluaciones usadas por el proceso PHC. Los parámetros que utilizaremos serán: tamaño de la población de inicio $n = 4$, mismo criterio de paro con $s_o = 0.001$, y mismo el criterio de aceptación de un conjunto como solución con $r_o = 0.1$. Estos experimentos serán repetidos para cada uno de los modelos $\text{memGA}_{\alpha,\beta}$ 50 veces iniciando siempre con una población inicial diferente.

4.4.4. Resultados sobre M_2, discusión y experimentación adicional

Las tablas 4.2, 4.3 y 4.4 contienen los promedios sobre 50 simulaciones independientes del tamaño de la población, el número de evaluaciones utilizadas por el AG, y el número total de evaluaciones, las cuales incluyen tanto las evaluaciones del algoritmo memGA como las utilizadas por el proceso PHC aplicado al conjunto solución que propone memGA. Si utilizamos una prueba estadística básica, como la formulada por ejemplo en el libro de Hines y Montgomery [64], para comparar los promedios de cada uno de los pares posibles de las tablas 4.2, 4.3 y 4.4 con un índice de confianza del 95 %, de suerte que aquella combinación de parámetros α y γ que tiene una cantidad de recursos mayor que otra con una confianza del 95 %, se declara perdedora y efectuaremos este proceso para todas las parejas de la tablas y nos *quedamos* con aquellas combinaciones que no *perdieron* ninguna de estas competencias estadísticas de medias obtendremos las siguientes listas de *ganadores* para cada una de las tablas.

	Conteo de evaluaciones al resolver M_2				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	546.4(297.1)	525.3(204.8)	547.5(200.5)	522.7(259.2)	525.3(190.3)
2.0	407.4(141.7)	415.2(162.4)	492.8(244.5)	400.3(116.4)	392.3(169.0)
4.0	420.2(180.1)	348.5(131.4)	372.3(138.9)	336.3(169.4)	364.8(144.2)
8.0	385.9(117.3)	**310.9**(108.2)	389.8(176.4)	339.5(190.2)	323.7(142.0)
16.0	350.2(143.5)	366.9(203.4)	349.9(138.8)	383.8(154.5)	351.5(150.1)

Tabla 4.3: Promedio sobre 50 simulaciones independientes del número de evaluaciones utilizado por memGA para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_2. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Conteo de evaluaciones totales al resolver M_2				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	951.6(306.0)	881.8(214.3)	867.6(200.0)	847.3(274.6)	839.7(186.1)
2.0	802.2(179.8)	772.9(182.3)	827.4(243.7)	757.8(114.2)	730.7(172.6)
4.0	801.7(198.4)	725.2(149.7)	732.5(141.1)	684.6(159.3)	699.7(134.7)
8.0	768.7(127.8)	**682.7**(137.4)	751.5(182.3)	708.5(181.8)	692.0(119.1)
16.0	752.1(167.9)	780.0(214.4)	724.0(154.9)	742.0(141.0)	713.5(145.8)

Tabla 4.4: Promedio sobre 50 simulaciones de la suma del número de evaluaciones utilizadas por memGA y el número de evaluaciones utilizadas por PHC aplicado a la solución provista por memGA en cada simulación, para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_2. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

- Para el tamaño de población no perdieron los modelos:
  $\text{memGA}_{8,1}$, $\text{memGA}_{16,1}$, $\text{memGA}_{8,2}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,16}$.
- Para el contador de evaluaciones utilizadas por sólo el AG no perdieron los modelos:
  $\text{memGA}_{16,1}$, $\text{memGA}_{4,2}$, $\text{memGA}_{8,2}$, $\text{memGA}_{16,4}$, $\text{memGA}_{4,8}$, $\text{memGA}_{8,8}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$.
- Para el contador de evaluaciones totales no perdieron los modelos:
  $\text{memGA}_{4,2}$, $\text{memGA}_{8,2}$, $\text{memGA}_{16,4}$, $\text{memGA}_{4,8}$, $\text{memGA}_{8,8}$, $\text{memGA}_{2,16}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$.

De los anteriores listas vemos que $\text{memGA}_{8,2}$, $\text{memGA}_{8,8}$, y $\text{memGA}_{16,16}$ son modelos que minimizan los tres recursos simultáneamente.

Un cuestionamiento natural que cabe al ver estos resultados es sobre la sensibilidad del buen desempeño de memGA con alguna de dichas combinaciones de parámetros cuando ocurre un cambio de la naturaleza del problema.

4.4.5. Resultados sobre variaciones a M_2

Para responder a estas dudas, al menos experimentalmente, se hicieron simulaciones similares a los anteriores para las funciones M_{2a}, M_{2b}, M_{2c}, M_{2d}, M_{2e}, M_{2f}, M_{2g}, M_{2h}, y M_{2i}. Las tablas que resumen los resultados se muestran en el apéndice 1. En ese mismo apéndice 1 aparece las listas de ganadores, es decir no perdedores con un índice de confianza del 95 % en ninguna competencia. Los modelos obtenidos de memGA para cada una de las funciones fueron los siguientes.

1. Para el problema M_{2a}: Mejores simultáneos:
   $\text{memGA}_{4,4}$, $\text{memGA}_{8,8}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,4}$, $\text{memGA}_{16,16}$
2. Para el problema M_{2b}: Mejores simultáneos:
   $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{16,16}$
3. Para el problema M_{2c}: Mejores simultáneos:
   $\text{memGA}_{8,4}$, $\text{memGA}_{16,2}$, $\text{memGA}_{16,4}$, $\text{memGA}_{16,8}$
4. Para el problema M_{2d}: Mejores simultáneos:
   $\text{memGA}_{8,4}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,8}$, $\text{memGA}_{16,16}$
5. Para el problema M_{2e}: Mejores simultáneos:
   $\text{memGA}_{16,2}$, $\text{memGA}_{16,8}$, $\text{memGA}_{16,16}$
6. Para el problema M_{2f}: Mejores simultáneos:
   $\text{memGA}_{2,16}$, $\text{memGA}_{4,8}$, $\text{memGA}_{4,16}$, $\text{memGA}_{16,8}$
7. Para el problema M_{2g}: Mejores simultáneos:
   $\text{memGA}_{2,16}$, $\text{memGA}_{4,8}$, $\text{memGA}_{4,16}$, $\text{memGA}_{16,8}$

8. Para el problema M_{2h}: Mejores simultáneos:
   $\text{memGA}_{8,4}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,8}$, $\text{memGA}_{16,16}$

9. Para el problema M_{2i}: Mejores simultáneos:
   $\text{memGA}_{8,1}$, $\text{memGA}_{16,1}$

Lo que observamos de estas simulaciones es que, en general, para estas variaciones al problema M_2 que hemos planteado, los valores para α y γ para los cuales memGA se comporta mejor son valores grandes. Que a valores mayores del parámetro α el tamaño de la población requerida para resolver el problema disminuya es consistente con los análisis previos. Esto debido a que se aumenta la presión por permanecer en la población memoria y por el hecho de que el número de nichos es reducido.

4.4.6. Resultados sobre M_6

Si cambiamos la naturaleza del problema y ahora pasamos a la función M_6 definida en el capítulo 2, y repetimos experimentos idénticos a los hechos para la función M_2 ahora con parámetros $r_o = 0.1$, tal y como es escogido por Mahfoud en su disertación [36], y con valor de σ_{share} como 8.0 obtenemos los resultados de las tablas B.1,B.3, y B.2 del apéndice 2. Asimismo se aplicaron las mismas comparaciones estadísticas hechas para M_2 ahora para los resultados obtenidos para la función M_6, y determinamos la selección de parámetros α y γ que dan un mejor desempeño promedio con un índice del 95 % podemos afirmar que los mejores son los siguientes.

- En tamaño:
  $\text{memGA}_{2,1}$, $\text{memGA}_{4,1}$, $\text{memGA}_{8,1}$, $\text{memGA}_{16,1}$, $\text{memGA}_{2,2}$,
  $\text{memGA}_{4,2}$, $\text{memGA}_{4,4}$, $\text{memGA}_{4,8}$, $\text{memGA}_{8,8}$

- En evaluaciones:
  $\text{memGA}_{16,1}$, $\text{memGA}_{16,2}$, $\text{memGA}_{16,4}$, $\text{memGA}_{16,8}$, $\text{memGA}_{16,16}$

- En evaluaciones totales:
  $\text{memGA}_{1,1}$, $\text{memGA}_{2,1}$, $\text{memGA}_{2,2}$, $\text{memGA}_{1,8}$, $\text{memGA}_{1,16}$

Como podemos observar no existe una combinación de valores para los parámetros α y γ que minimice simultáneamente los tres recursos que hemos definido como relevantes. También podemos observar, contrariamente a las simulaciones para M_2 y las variaciones para ella que hemos contemplado, que para M_6 ahora los valores pequeños de los parámetros son más convenientes. Este problema no puede ser resuelto por CD como fue experimentalmente comprobado por Mahfoud [36] y en su argumentación se comentaba que en este problema CD presentaba una emigración hacia nichos mejor evaluados cuando se combinaban mediante cruce elementos de nichos inferiores. Notamos que en este problema un buen porcentaje, 11 de 19, de las mejores selecciones de los parámetros involucren valores pequeños de γ. Es decir, y a la luz de los análisis teóricos desarrollados, en este tipo de problema no hace falta aumentar la diversidad de la búsqueda. Mediante estos ejemplos reducidos podemos concluir que no es deseable

fijar los parámetros α y γ en valores altos debido a que esto impactará negativamente en el desempeño de memGA en algunos problemas debido a su propia naturaleza y codificación, la cual no es fácil de entender antes de ser resuelto el problema.

4.4.7. Resultados sobre una función engañosa

Buscando un contraste, en esta parte del análisis haremos simulaciones similares a las anteriores ahora sobre un problema más complejo. La función que lo define es una que está definida sobre las cadenas binarias de longitud 12, y es el resultado de sumar la aplicación de la función f sobre el número de unos contenidos en la primera mitad del cromosoma con la aplicación de esa misma función sobre el número de unos de la segunda mitad. La función f es una función conocida entre los practicantes de AGs como una función bipolar y engañosa, su definición es:

$$f(i) = \begin{cases} 1.0 & \text{si } i = 0 \text{ ó } i = 6 \\ 0.0 & \text{si } i = 1 \text{ ó } i = 5 \\ 0.640576 & \text{si } i = 2 \text{ ó } i = 4 \\ 0.360384 & \text{si } i = 3 \end{cases} \tag{4.41}$$

Si se utiliza la métrica de Hamming, la función resultante, que llamaremos E_2 y cuya fórmula aparece en la ecuación 4.42, es una función multimodal y consta de 4 óptimos globales y 22^2 óptimos locales.

$$E_2(c) = f(unos(subc(c, 1, 6))) + f(unos(subc(c, 7, 12))) \tag{4.42}$$

La expresión $subc(c, i, j)$ entrega el subcromosoma del cromosoma c comprendido entre las posiciones i y j inclusive. La función *unos* entrega el número de dígitos 1 contenidos en una cadena binaria. El problema, que es multimodal por naturaleza, es convertido en uno problema multimodal reducido; un conjunto es admitido como solución a este problema multimodal, si contiene individuos alrededor de los 4 óptimos globales, que en este problema son las cadenas 000000000000, 000000111111, 111111000000 y 111111111111. El radio de vecidad utilizado es 2.5, es decir que una cadena se considera en la vecindad de un óptimo global, si sólo difiere del óptimo en un máximo de dos bits. Este valor coincide con el valor de σ_{share} que utilizaremos. A diferencia de los problemas anteriores, en éste utilizaremos la distancia de Hamming sobre los cromosomas, a saber, la distancia entre dos cromosomas es la cantidad de posiciones en las que difieren.

Aplicaremos a este problema, similares experimentos a los ejecutados sobre M_2 y M_6. Usaremos la misma estrategia de medición de recursos con parámetros, y también mantendremos el mismo criterio de paro para el algoritmo con parámetro s_o igual a 0.001. Mantendremos el operador de cruce en un punto con probabilidad 1 de apliación y la probabilidad de mutación la fijaremos en cero. Los resultados de estas simulaciones aparecen en la tablas C.1, C.2 y C.3 del apéndice 3. Siguiendo con la estrategia de comparación de los modelos en el mismo apéndice 3 aparece la competencia entre las variaciones a memGA sobre cada uno de los parámetros. En este caso los ganadores

simultáneos son memGA$_{2,4}$ y memGA$_{8,2}$. Es decir, en este caso no hay una regla para determinar los valores de los parámetros para el modelos ganador de acuerdo a la regla desarrollada para M_2 y problemas similares, y M_6. Una regla empírica que sugerimos es conservar los parámetros α y γ en valores reducidos, digamos 2.0 como sugieren convincentemente los experimentos sobre M_6, y que apenas insinuan los resultados sobre E_2. Esta selección puede justificarse parcialmente diciendo que finalmente los problemas complejos son los más motivadores de resolver.

4.4.8. Desempeño de memGA a lo largo de las generaciones

Habiendo analizado el impacto que tiene la selección de los parámetros α y γ en el desempeño de memGA en problemas particulares es deseable observar la forma como la población es evolucionada por el algoritmo; también debemos mostrar experimentalmente la capacidad de retención que presenta el mismo de óptimos locales localizados. Asimismo, deseamos observar si acaso ocurre la convergencia de la población a los óptimos locales. Ello debido a que en el capítulo 2 se formularon este tipo de comportamientos negativos como fallas que presentaban CD y SH, y debido a ello se presentó a memGA como una alternativa. En los siguientes apartados diseñaremos, realizaremos y analizaremos una serie de experimentos encaminados a revisar experimentalmente el desempeño de memGA en este sentido.

Diseño experimental

En los siguientes experimentos usaremos parte del marco de referencia experimental usado en este capítulo, a saber, las funciones M_2 y M_6. Como nuestro objetivo tiene como fin ver cómo la población es evolucionada, detendremos la evolución de memGA por intervalos medidos respecto a un número aproximado de evaluaciones, y revisaremos la población memoria disponible; contaremos la cantidad de vecindades vacías, y el promedio de las distancias a los óptimos locales donde los elementos de la población se encuentren. Este último indicador corresponde a la calidad de un conjunto de puntos como solución a un problema multimodal como había sido previamente definido en el capítulo anterior.

Para la función M_2 se parte de una población aleatoria de 40 individuos. Note que 40 individuos es una cantidad suficientemente grande ya que de acuerdo a la tabla 4.2 cubre la mayor de las poblaciones con un poco menos de la desviación estándar. Los resultados de la evolución se muestran en la tabla 4.5.

En el caso de la función M_6, fijaremos el tamaño de la población en 400 individuos. Dejaremos correr la población y la detendremos a intervalos iguales del número de evaluaciones para observar cómo está siendo evolucionada la población. Los resultados de estas simulaciones aparecen en la tabla 4.6. Revisemos ahora las tablas 4.5 y 4.6 con los resultados de las 50 simulaciones independientes. En el caso de las simulaciones correspondientes a la función M_2 notamos lo siguiente. Que un tamaño de población de 400 individuos no era tan generoso como lo fue 40 en las simulaciones correspondientes para M_2. De hecho un indicador inicial de 0.50 indica que inicialmente la población no

memGA$_{2,2}$ aplicado a M_2 población 40								
	Indicadores de la población a lo largo de las evaluaciones							
	$\bar{f}$		$\bar{N}$		E		D	
e	μ	σ	μ	σ	μ	σ	μ	σ
40	0.205	0.045	5.92	0.417	0.02	0.141	0.175	0.071
60	0.398	0.052	6.47	0.305	0.02	0.141	0.159	0.074
80	0.476	0.055	6.86	0.326	0.00	0.000	0.150	0.079
100	0.512	0.057	7.06	0.378	0.00	0.000	0.147	0.080
120	0.535	0.056	7.18	0.406	0.00	0.000	0.145	0.077
140	0.546	0.057	7.24	0.414	0.00	0.000	0.142	0.076
160	0.555	0.061	7.28	0.430	0.00	0.000	0.142	0.079
180	0.560	0.060	7.32	0.425	0.00	0.000	0.139	0.080
200	0.566	0.061	7.35	0.431	0.00	0.000	0.138	0.080
220	0.573	0.062	7.39	0.433	0.00	0.000	0.130	0.079
240	0.577	0.060	7.42	0.436	0.00	0.000	0.127	0.077
260	0.581	0.057	7.44	0.423	0.00	0.000	0.121	0.075
280	0.584	0.056	7.46	0.397	0.00	0.000	0.116	0.073
300	0.586	0.054	7.49	0.400	0.00	0.000	0.115	0.074
400	0.590	0.056	7.50	0.426	0.00	0.000	0.105	0.072
500	0.599	0.052	7.61	0.486	0.00	0.000	0.096	0.069
600	0.605	0.054	7.63	0.506	0.00	0.000	0.091	0.070
700	0.608	0.053	7.64	0.508	0.00	0.000	0.089	0.070
800	0.611	0.053	7.66	0.509	0.00	0.000	0.086	0.069
900	0.615	0.049	7.72	0.469	0.00	0.000	0.080	0.064
1000	0.617	0.049	7.71	0.466	0.00	0.000	0.076	0.062

Tabla 4.5: Promedio sobre 50 simulaciones independientes de los indicadores principales de la población evolucionada por memGA a un determinado número de evaluaciones y aplicado a M_2. Se incluyen la evaluación promedio($\bar{f}$) y su desviación, el contador de nichos ($\bar{N}$) y su desviación estándar, el contador nichos vacíos E y su desviación, y el promedio de las distancias a los óptimos locales más próximos y su desviación.

memGA$_{2,2}$ aplicado a M_6, población 400								
	Indicadores de la población a lo largo de las evaluaciones							
	$\bar{f}$		$\bar{N}$		E		D	
e	μ	σ	μ	σ	μ	σ	μ	σ
400	26.68	5.27	3.24	0.083	0.50	0.678	0.553	0.034
500	36.32	5.40	3.27	0.085	0.46	0.676	0.516	0.038
600	46.67	6.04	3.34	0.102	0.40	0.639	0.482	0.044
700	57.19	6.18	3.46	0.139	0.34	0.626	0.453	0.047
800	67.72	6.85	3.63	0.206	0.32	0.621	0.432	0.049
900	79.07	6.95	3.88	0.286	0.26	0.565	0.410	0.054
1000	91.69	7.77	4.21	0.365	0.26	0.565	0.389	0.058
1500	153.53	12.04	6.98	0.988	0.20	0.452	0.328	0.068
2000	219.82	13.94	12.13	2.292	0.18	0.438	0.297	0.076
2500	287.53	16.16	19.25	3.843	0.16	0.422	0.268	0.080
3000	336.18	16.53	22.13	2.877	0.16	0.422	0.251	0.081
4000	383.25	17.91	20.86	1.628	0.06	0.240	0.234	0.071
5000	408.25	15.15	20.56	1.456	0.02	0.141	0.217	0.065
6000	418.78	13.90	20.03	1.404	0.02	0.141	0.201	0.068
7000	426.02	12.86	19.35	1.335	0.02	0.141	0.179	0.067
8000	431.23	12.77	18.74	1.193	0.02	0.141	0.163	0.067
9000	436.49	13.07	18.32	1.064	0.02	0.141	0.151	0.066
10000	441.00	13.03	17.91	0.980	0.02	0.141	0.145	0.065

Tabla 4.6: Promedio sobre 50 simulaciones independientes de los indicadores principales de la población evolucionada por memGA aplicada a M_6. Se incluyen la evaluación promedio($\bar{f}$) y su desviación, el contador de nichos ($\bar{N}$) y su desviación, el contador nichos vacíos E y su desviación, y el promedio de las distancias a los óptimos locales más próximos y su desviación.

cubría todas las vecindades de los óptimos locales. De hecho, revisando los datos no incluidos en la tabla se pudo revisar que efectivamente en 20 de las 50 simulaciones la población inicial no tenía un representante en alguna vecindad. Si no existe coincidencia entre el número 0.50 y el correspondiente 20/50 se debe a que en algunos casos la población inicial no abarcaba dos o más vecindades. Sin embargo, en el proceso de evolución de la población, paulatinamente ésta abarcaba esas vecindades antes vacías de representante. De hecho el número final 0.02 indica, como se pudo cotejar con los datos que no se presentan, que en una sola simulación, de las cincuenta, existió una sola vecindad vacía. Otra de los hechos experimentales que podemos observar es que existe un proceso de convergencia de la población hacia los óptimos locales lo cual se refleja en el indicador D, el cual representa el promedio de las distancias a los óptimos más cercanos sobre cada elemento de la población ponderado por el radio del nicho. Esta convergencia también se ve reflejada en el indicador del promedio de evaluación de los individuos de la población; efectivamente observamos que la evaluación sube rápidamente lo cual coincide con la imagen de la gráfica de M_6 que aparece en la figura 2.15. El indicador de promedio del conteo de nicho aproximado tiene un comportamiento interesante. Inicialmente tiene un valor bajo, lo cual es indicio de que la población está distribuida en el espacio de búsqueda. Posteriormente crece indicando que se han formado nubes de puntos; éstas deberían de corresponder a los óptimos locales; posteriormente disminuye ligeramente. Esta disminución ligera corresponde a los tiempos en los cuales disminuye el contador de vecindades vacías; lo cual significa que parte de la población se ha *ido* hacia la vecindad del óptimo recién encontrado. En resumen, la población evoluciona manteniéndose en las vecindades de los óptimos y acercándose a ellos. El hecho de que la convergencia no disminuya más puede deberse a que en ausencia de mutación hay una falta de riqueza de alelos que le impide al algoritmo generar la información de los individuos aún más próximos.

4.4.9. Sensibilidad en el parámetro k

Habiendo seleccionado valores para los parámetros de las figuras de mérito, otro de los elementos importantes es el tamaño de la población exploradora. En esta sección realizaremos experimentos limitados para tener una visión de cuál es el efecto de la selección de k en el desempeño de memGA. Para ello se realizaron experimentos en los que se utilizaron las funciones M_2, M_6 y E_2 adoptadas recientemente el capítulo. Se fijaron los parámetros $\alpha = 2.0$, y $\gamma = 2.0$, y se ejecutaron los mismos experimentos para determinar cuál es la variante de memGA que se desempeña mejor para un problema determinado. Ahora el parámetro que varía es el tamaño de la población exploradora. Los resultados obtenidos sobre las funciones M_2, M_6 y E_2 se encuentran en las tablas 4.7, 4.8, y 4.9. La tabla 4.7 nos indica que al aumentar el tamaño de la población exploradora aumenta sustancialmente el número de evaluaciones invertidas, tanto por el AG como por PHC. En cierta manera k grande es innecesaria e implica un desperdicio de esfuerzo. El tamaño de la población requerido para resolver el problema no se ve significativamente disminuido. La tabla 4.7 en las evaluaciones confirma la

memGA$_{2,2}$ sobre M_2			
k	Tamaño de población $\bar{s}(\sigma_{\bar{s}})$	Evaluaciones AG $\bar{e}(\sigma_{\bar{t}})$	Evaluaciones totales $\bar{t}(\sigma_{\bar{t}})$
4	24.0(8.08)	431.3(171.41)	805.5(187.95)
6	25.6(11.20)	589.8(242.61)	936.7(243.65)
8	26.6(13.56)	689.4(260.99)	1029.4(276.11)
10	25.3(12.56)	749.5(248.53)	1078.6(252.25)
12	25.9(11.15)	963.6(336.53)	1283.7(332.97)
14	27.8(9.03)	1045.6(259.86)	1369.2(267.07)
16	24.0(9.83)	1117.8(345.56)	1442.8(342.11)

Tabla 4.7: Desempeño de memGA$_{2,2}$ para diferentes tamaños de la población exploradora (k) sobre M_2.

misma situación. Pero observamos que el tamaño de la población requerida aumenta al aumentar el tamaño de la población exploradora. La razón que suponemos influye es el comportamiento de CD; recordemos que en el problema M_6 presentaba un fenómeno de emigración a nichos con evaluación superior dejando vacías otras vecindades. Al crecer la población exploradora los individuos que pretenden entrar son abundantes y localizados en los óptimos más altos; como el valor de α es pequeño estas copias tienden a subsistir en las competencias viciando la búsqueda. La tabla 4.9 muestra un comportamiento interesante. Cuando se comparan los tamaños de población requeridos para $k = 4$ y $k = 6$ el tamaño decae importantemente, y el número de evaluaciones tiene un decremento apenas significativo. ¿Habrá algún concepto de *masa crítica* de búsqueda para el problema E_2? Se hicieron experimentos adicionales para la función E_3 que es la extensión de la función E_2 pero a cromosomas de longitud 18; ésta tiene por fórmula:

$$E_3(c) = f(unos(subc(c, 1, 6))) + f(unos(subc(c, 7, 12))) + f(unos(subc(c, 13, 18))). \tag{4.43}$$

Esta función es multimodal y engañosa donde son requeridos solamente los óptimos globales, que son los que tienen evaluación 3. Estos óptimos son las cadenas de cromosomas de 18 bits de longitud formadas con segmentos de 6 bits de lóngitud que sólo contienen o ceros o unos. Esta función tiene 22^3 óptimos locales, y corresponde a las funciones masivamente multimodales definidas por Goldberg, Deb y Horn [37]. Los resultados sobre esta función aparecen en la tabla 4.10. Son interesantes los resultados del efecto de diferentes valores en el tamaño de la población población requerida y el número de evaluaciones totales que aparece en la tabla 4.10. Salvo detalles estadísticos, confirmamos que el tamaño de la población exploradora está relacionado con el desempeño no sólo en forma negativa cuando éste aumenta. Observamos que al aumentar el tamaño de población en el rango de 6 a 18 el tamaño de la población requerida va disminuyendo, primero en forma importante y posteriormente sigue disminuyendo pero en forma más ligera. En este mismo rango el número de evaluaciones disminuye.

memGA$_{2,2}$ sobre M_6			
k	Tamaño de población $\bar{s}(\sigma_{\bar{s}})$	Evaluaciones AG $\bar{e}(\sigma_{\bar{t}})$	Evaluaciones totales $\bar{t}(\sigma_{\bar{t}})$
4	314.9(121.87)	4025.8(1298.40)	7587.7(1760.76)
6	391.7(204.29)	6795.6(2911.44)	10806.3(3371.99)
8	412.2(181.55)	8908.0(3340.44)	13401.8(3836.81)
10	384.0(163.55)	10278.4(3843.71)	14781.2(4476.12)
12	409.6(155.16)	12224.6(3989.93)	16962.7(4598.28)
14	409.6(126.69)	13735.4(4323.89)	18650.9(4698.19)
16	432.6(177.08)	16517.1(5804.20)	21343.2(6382.74)

Tabla 4.8: Desempeño de memGA$_{2,2}$ para diferentes tamaños de la población exploradora (k) sobre M_6.

memGA$_{2,2}$ sobre E_2			
k	Tamaño de población $\bar{s}(\sigma_{\bar{s}})$	Evaluaciones AG $\bar{e}(\sigma_{\bar{t}})$	Evaluaciones totales $\bar{t}(\sigma_{\bar{t}})$
4	256.0(219.71)	2630.3(1430.56)	3981.0(1611.81)
6	118.4(87.99)	2649.8(944.45)	3688.0(1243.23)
8	109.1(88.03)	3450.9(1402.07)	4400.1(1626.51)
10	108.5(70.32)	4555.5(1692.62)	5561.8(1998.48)
12	89.0(58.91)	5474.3(2576.77)	6365.7(2803.45)
14	70.7(43.74)	5980.1(2662.58)	6749.3(2823.05)
16	69.4(44.04)	7453.4(3430.20)	8227.0(3641.00)

Tabla 4.9: Desempeño de memGA$_{2,2}$ para diferentes tamaños de la población exploradora (k) sobre E_2.

memGA$_{2,2}$ sobre E_3			
k	Tamaño de población $\bar{s}(\sigma_{\bar{s}})$	Evaluaciones AG $\bar{e}(\sigma_{\bar{t}})$	Evaluaciones totales $\bar{t}(\sigma_{\bar{t}})$
6	477.1(494.45)	7842.4(4228.35)	17306.4(11168.31)
8	366.8(386.26)	8645.5(4163.43)	16947.0(11810.87)
10	252.3(307.78)	10351.1(4548.16)	16278.7(10567.66)
12	115.7(95.43)	11712.6(4469.39)	14540.7(4895.23)
14	128.0(100.38)	11615.5(4257.22)	14978.8(5198.93)
16	124.3(122.88)	13843.7(5691.91)	17168.8(6403.84)
18	83.7(48.81)	15713.2(10452.86)	17906.5(10233.14)
20	89.8(48.76)	14112.1(5319.69)	16517.9(5738.62)
22	98.5(59.30)	18557.3(9018.62)	21206.8(9221.02)
24	94.8(58.29)	18068.9(7807.09)	20701.1(7642.90)
26	93.5(50.33)	22784.5(13061.20)	25289.3(13195.38)
28	68.9(35.85)	24543.1(14565.61)	26337.5(14351.41)
30	86.1(97.80)	23501.1(12889.39)	25833.5(13331.49)
32	57.8(30.04)	27416.6(11355.68)	28880.1(11410.29)

Tabla 4.10: Desempeño de memGA$_{2,2}$ para diferentes tamaños de la población exploradora (k) sobre E_3.

Indicado lo adecuado de un tamaño de población exploradora alrededor de los 12 o 14 individuos. Fuera de ese rango el tamaño de población decrece pero el número de evaluaciones crece fuertemente. Estas limitadas simulaciones hacen suponer la idea de una *tamaño de masa crítica* en la población exploradora. Esta idea no fue explorada. Faltan por discutir aspectos relacionados con el esfuerzo computacional que requieren los cálculos de las figuras de reemplazo y selección para la población. Sin querer cambiar el flujo del texto, pero adelantando información para aquel improbable y escéptico lector, diremos que la recompensa obtenida por memGA involucra un ahorro importante en el número de evaluaciones cuando se compara con otras estrategias, lo cual hace pensar que este esfuerzo computacional derivado del cálculo de los indicadores puede ser considerado reducido al ver sus beneficios.

Habiendo llegado al momento de decidir la selección de parámetros adecuados no tenemos un resultado contundente de la elección. Valores de α y γ son adecuados para funciones relativamente simples. Con los problemas que hemos experimentado, para funciones más complejas los valores pequeños para tales parámetros son los mejores. Nuestra propuesta de algoritmo es mantener los valores de los parámetros relativamente bajos: $\alpha = 2.0$ y $\beta = 2.0$. En los simulaciones del próximo capítulo justificaremos experimentalmente esta selección.

4.5. Resumen

En la presente sección hemos hecho un análisis de la figuras de mérito utilizadas por el algoritmo memGA. Hemos visto cuál es el impacto de los parámetros α y γ de sus figuras de reemplazo y selección sobre su desempeño. En particular, se ha hecho un análisis reducido de sus efectos sobre la diversidad de la respuesta entregada y la diversidad de la exploración del espacio de búsqueda. Posteriormente hemos hecho algunos experimentos que indican valores adecuados para estos parámetros, utilizando un grupo de funciones fijas. Hemos descubierto la dependencia de la selección adecuada de ellos y la naturaleza del problema. Específicamente, a partir de la experimentación hemos descubierto que mientras que para ciertos problemas el desempeño es mejor cuanto más alto son los valores de esos parámetros, para otros problemas la situación es opuesta. Como en general no se tiene información sobre la naturaleza de un problema se ha propuesto que estos parámetros permanezcan en valores reducidos. Siendo precisos, el algoritmo memGA corresponde a $\text{memGA}_{2,2}$. En el siguiente capítulo compararemos la propuesta contra los algoritmos de CD y SH utilizando el marco de referencia experimental propuesto en la literatura así como por un grupo problemas adicionales.

Capítulo 5

Comparación experimental

En los capítulos previos hemos propuesto y parcialmente analizado un algoritmo para resolver problemas de optimización multimodal que está basado en el manejo de dos poblaciones. Este algoritmo basa la exploración del espacio de búsqueda en un micro algoritmo genético. Dos figuras de mérito le permiten mantener la diversidad de lo encontrado y la diversidad de la búsqueda. El ajuste de los parámetros que determinan la intensidad de tales diversidades han sido sugeridos con base en experimentación y análisis.

En esta sección haremos una evaluación experimental del algoritmo propuesto mediante la comparación contra *crowding* determinístico (CD), contra el método de las funciones de compartición o *sharing* (SH), y contra un método denominado *clearing* (CL) el cual ha aparecido en el largo proceso de la escritura de este documento. Hemos seleccionado estas estrategias debido a que son de las estrategias más exitosas para resolver problemas de optimización multimodal reportadas en la literatura.

Para hacer estas comparaciones utilizaremos las funciones comprendidas en general en el marco de referencia experimental usado en la literatura, adicionado funciones similares a ellas y que requieren un mayor esfuerzo computacional. Pondremos especial énfasis en tres principales indicadores. El primer indicador es el tamaño de la población requerida en la solución, que representa la cantidad de información que requiere el algoritmo para llegar a la solución del problema. El segundo indicador importante es el número de evaluaciones que utiliza el algoritmo para resolver el problema. Este indicador mide el esfuerzo invertido por el algoritmo para llegar a una solución al problema. El tercer indicador es uno que indirectamente mide la calidad del conjunto propuesto como solución. Esta calidad es medida contabilizando el número de evaluaciones que un optimizador local utilizaría para *mover* cada uno de los puntos dados en el conjunto solución a los óptimos locales más próximos a cada uno de ellos.

El presente capítulo está organizado de la siguiente forma. Primeramente se describe el marco de referencia experimental utilizado en las comparaciones. Después se hace una descripción del método conocido como *clearing*, y seguidamente se presenta el diseño de los experimentos para realizar las mismas. Posteriormente se presentan los estadísticos importantes de los resultados experimentales obtenidos al aplicar las estrategias a comparar. Finalmente, el capítulo concluye haciendo una comparación

sumaria de las estrategias comparadas.

5.1. Marco de referencia experimental

Los problemas sobre los cuales serán comparadas las estrategias incluyen las funciones senoidales M_1 a M_4, la función de Himmelblau M_5, la función de Sekel modificada M_6, la función altamente multimodal y engañosa M_7, y la función M_8 que es la versión escalada de M_7 la cual ya no es engañosa. Todas estas funciones de prueba han sido utilizadas previamente en la literatura [33, 34, 36, 35]. Las funciones M_1 a M_5 han sido resueltas con relativa facilidad por *crowding* determinístico y *sharing*. La función M_6 presenta problemas para ser resuelta por CD pero es fácilmente resuelta por SH. La función M_7 no es resuelta por SH pero es posible resolverla con CD con un número grande de recursos; tanto en el tamaño de la población requerido, como en el número de evaluaciones. La función M_8 es resuelta tanto por CD como por SH. Este es el marco de referencia comúnmente usado en la literatura. Este marco de referencia experimental aparece documentado en el capítulo 2. Adicionalmente a estas funciones, utilizaremos un marco extendido que nos permita tener una visión adicional de ventajas de un algoritmo sobre otro. Este marco adicional estará constituido por tres familias de problemas. La primera familia de este marco se constituye por las variaciones a la función M_2 que fueron introducidas en el capítulo anterior. Esta familia está constituida por los problemas M_{2a}, M_{2b}, M_{2c}, M_{2d}, M_{2e}, M_{2f}, M_{2g}, M_{2h}, y M_{2i}. Esta familia permitirá estudiar la sensibilidad del algoritmo ante cambios ligeros en el problema.

La segunda familia de problemas que consideraremos es la asociada a las funciones senoidales que tienen como función problema la ecuación:

$$S_n(x) = e^{-2x^2} \operatorname{sen}^6(n\pi x). \tag{5.1}$$

Los problemas multimodales asociados a esta familia de funciones tendrán como espacio de búsqueda el intervalo $[0, 1]$, y la función de codificación será idéntica a la utilizada para la función M_2. Es decir, que primeramente el cromosoma será convertido en el número entero que representa y posteriormente será dividido entre el número $2^l - 1$. Con la diferencia que utilizaremos cromosomas binarios de longitud 40 ($l = 40$), en lugar de longitud 30. Esto con el fin de aumentar la *fineza* con la cual se muestrea el espacio de búsqueda. La función S_n de esta familia tiene n óptimos locales en el intervalo $[0, 1]$ los cuales están, para motivos prácticos, uniformemente distribuidos. Estos óptimos se calculan aproximadamente por la fórmula $(2i-1)/(2n)$, para valores de i desde 1 hasta n. Los óptimos exactos no se encuentran en esos valores, sino en un valor muy cercano y a la izquierda de ellos debido a que la función que define el perfil de decaimiento de los óptimos es decreciente; no están exactamente distribuidos uniformemente debido a que el mismo perfil de decaimiento decrece a diferentes velocidades en el mismo intervalo. La familia $\{S_n\}$ es interesante debido a que para diferentes valores del parámetro n, el problema correspondiente presenta diferentes comportamientos relacionados con la aparición de descedientes en vecindades de óptimos con mejor evaluación que la correspondiente a los padres, dando origen esto a la emigración de la población hacia

la vecindad de óptimos con evaluación alta, dejando vacías en este proceso vecindades de óptimos con evaluación baja. Este fenómeno fue detectado *a posteriori* al observar e interpretar los resultados de los experimentos hechos en esta sección. En el mismo proceso de experimentación observaremos cómo este efecto nocivo en CD impacta en el desempeño de memGA.

La última familia de funciones que consideramos está basada en la función M_7. Las funciones de los problemas que componen dicha familia las podríamos definir como

$$E_i(c) = \frac{1}{i}\sum_{j=1}^{j=i} f(unos(subc(c, 6\,(j-1)+1, 6\,j))), \tag{5.2}$$

donde la expresión $subc(c, n, m)$ entrega el segmento del cromosoma c entre las posiciones n y m inclusive, la función $unos(c')$ determina el número de unos que posee el cromosoma c', y la función f es la función con valores $f(0) = f(6) = 1$, $f(1) = f(5) = 0$, $f(2) = f(4) = 0.360384$, y $f(3) = 0.640576$. La función $f \circ unos$ es una función engañosa de dos máximos globales definida sobre un cromosoma de longitud 6. El número de óptimos de ella es de 22. Cada una de las funciones E_i se convierte en una función engañosa definida sobre un cromosoma de longitud $6\,i$ que tiene un total de 2^i óptimos globales y una gran cantidad de subóptimos, por ejemplo en el caso $i = 5$ el número de óptimos locales es de 22^5. En nuestro caso experimentaremos con los casos $i = 1, 2, 3, 4, 5$.

5.2. El método de *clearing*

El método de *clearing* desarollado por Alain Pétrowski [65] es una estrategia de nicheo basada en algoritmos genéticos. El método opera como una variante de un AG que usa como estrategia de selección el mecanismo de SUS ya descrito anteriormente. La variante sólo implica un paso adicional; un proceso de *limpieza* que se aplica antes del proceso de selección. Este proceso hace cero literalmente la evaluación de individuos próximos a individuos con mejor evaluación. La proximidad es definida en forma definitiva: si la distancia entre dos individuos es menor que la cantidad definida como el radio de nicho aceptado. En un nicho el individuo que conserva su evaluación es aquél con mayor evaluación; la evaluación de los otros en el nicho se hace cero. Esta es una verdad parcial; la estrategia tiene un parámetro adicional denominado *Kappa*, el cual es un entero que regula el tamaño de los nichos, de tal suerte que no hace cero la evaluación de los *Kappa* individuos mejores en el nicho. El proceso de *clearing* puede utilizarse en un AG generacional o bien en un AG elitista que conserva los mejor evaluados. Por claridad, el proceso de *clearing* consiste en el paso en el cual se hacen cero las evaluaciones de individuos no deseables para el proceso de selección, mientras que el método de *clearing* o simplemente *clearing* es el AG que utiliza el procedimiento de *clearing* previo al proceso de selección. El proceso de *clearing* aparece descrito en la figura 5.1.

Existe una similaridad importante entre *clearing* y memGA; si en *clearing* elegimos el valor *Kappa* como uno, sólo el mejor representante de cada nicho es candidato a ser

1. Ordene la población P de acuerdo a f
2. Desde $i = 1$ hasta $n = |P|$ haga:
   a) Si la evaluación de $P[i]$ es mayor que cero, haga:
      1) nNicho $= 1$
      2) Desde $j = i + 1$ hasta n haga:
         i) Si la evaluación de $P[j]$ es positiva y la distancia entre $P[i]$ y $P[j]$ es menor que σ_{share}, si nNicho $< Kappa$ tome nNicho $= n$Nicho $+ 1$, en otro caso haga cero la evaluación de $P[j]$.

Figura 5.1: Seudocódigo para el procedimiento de *clearing*

elegido para el proceso de selección y cuando se utiliza en un AG con elitismo estos individuos son los que paulatinamente sobrevivirán en el transcurso de las generaciones; mientras que en memGA el indicador de reemplazo de un individuo x en una cierta generación es:

$$R(x) = \frac{f(x)}{N_1(x)^\alpha}$$

para valores grandes del parámetro α, digamos que para cuando α tiende a infinito, tienen como efecto resultante que hacen cero el indicador $R(x)$.

En las comparaciones siguientes utilizaremos una versión de *clearing* elitista: si n es el tamaño de la población, después del procedimiento de *clearing* nos quedaremos con los individuos que poseen una evaluación positiva digamos m ($m \leq n$); el número de selecciones para participar en el *pool* de apareamiento, y por consiguiente el número de nuevos elementos, será el mayor número par que es menor o igual a $n - m$. Los nuevos individuos generados por cruce y mutación a partir del *pool* de apareamiento reemplazarán sólo a los individuos con evaluación cero. Preservando como mencionábamos a los individuos con evaluación positiva. El parámetro $Kappa$ que indica cuántos elementos elementos de un pico en la población actual son retenidos se ha mantenido en 1 como en los experimentos descritos por Pétrowski [65].

5.3. Diseño de experimentos e invariantes en ellos

Los experimentos que se desarrollarán tienen como fin establecer un criterio de calidad sobre el desempeño de los algoritmos a comparar. Este criterio de calidad es el resultado de tres indicadores: el esfuerzo computacional invertido en llegar a una solución, la cantidad de información inicial utilizada para llegar a ella, y la calidad de la solución entregada. El esfuerzo computacional se medirá contabilizando el número

de evaluaciones utilizadas por el algoritmo. La cantidad de información utilizada se medirá por el tamaño de la población inicial requerida por el algoritmo para llevar esa población aleatoria a una solución. La calidad de la solución entregada es un criterio difícil de medir. Para ello utilizaremos la estrategia adaptada por Mahfoud [36] la cual mide la calidad de la solución determinando el esfuerzo computacional, medido en términos del número de evaluaciones de la función objetivo, invertidas en *acercar* los individuos de la población final propuesta como solución a sus óptimos locales más próximos. Este proceso ha sido utilizado previamente en el capítulo anterior y al algoritmo se le ha designado por PHC y que fue presentado en el capítulo previo y cuyo código aparece en la figura 4.10. El optimizador local PHC aparece descrito en la figura 4.10 y es la implementación de Mahfoud [36] del buscador de alpinista del *próximo paso* desarrollado por Mühlenbein [63]. La vecindad inicial que es utilizada corresponde al valor dado de σ_{share} y el valor de ϵ corresponde al valor $(b-a)/(2^l)$, donde $[a, b]$ corresponde al intervalo donde la variable toma sus valores y l corresponde a la longitud del subcromosoma que la codifica.

Para poder establecer comparaciones imitaremos la estrategia seguida por Mahfoud [36] al evaluar la estrategia SN desarrollada por Beasley y colaboradores [35] y compararla contra SH, y CD. Esta estrategia también fue desarrollada en el capítulo anterior y aparece descrita en la figura 3.1 consistiendo de los siguiente. La estrategia a considerar se aplica a una población aleatoria y reducida en tamaño. El algoritmo evoluciona la población hasta que un criterio de paro se habilita. La población final es revisada y se determina si cada uno de los óptimos locales de la función problema tiene al menos un punto en la población lo suficientemente cerca a él. Esta condición será abreviada diciendo que el conjunto es *admitido como solución* al problema multimodal. El concepto de estar suficientemente cerca se define indicando una distancia mínima, o radio del nicho, r_0, de cercanía para la pertenecia al óptimo local. En caso de que la población entregada no sea admitida como solución, se duplicará en tamaño de la población inicial adicionando una población aleatoria a la población inicial que fue el punto de partida a la simulación; la estrategia es aplicada nuevamente partiendo de esta nueva población. En cada una de las reinicializaciones el contador de evaluaciones es colocado en cero. En caso de que la población final sea admitida como solución, la población se contabiliza en tamaño y también se contabiliza el número de evaluaciones desde el posible reinicio de la simulación. Sobre la población solución se aplica el proceso de optimización local PHC; se contabiliza el número de evaluaciones utilizada por este proceso. Finalmente se reportan tanto el tamaño de la población como el número de evaluaciones utilizadas por el algoritmo, y el número de evaluaciones utilizadas por el optimizador local.

El conteo de evaluaciones utilizadas por la estrategia de optimización local visto como un indicador de calidad en la solución puede no ser del todo claro; por ejemplo, imaginemos la comparación de la calidad entre dos conjuntos solución propuestos: uno que es un conjunto formado por 3 copias de un óptimo local, y el otro que es un conjunto con dos elementos, uno el óptimo local y otro un punto que está exactamente a la distancia inicial de la vecindad. Si m es el número de evaluaciones invertidas en revisar si un individuo es un óptimo local, en el primer caso el algoritmo de optimi-

zación invertirá en el conjunto solución $3m$ evaluaciones, mientras que en el segundo invertirá $m + m + 2$ evaluaciones. Es decir, que usando el criterio de un mejor número de evaluaciones invertidas por el algoritmo de optimización local, uno supondría que el segundo conjunto solución tiene una mejor calidad. Esta conclusión es muy custionable. Este tipo de ejemplos nos lleva a pensar que el número de evaluaciones usadas por el el optimizador local no es un claro indicador de la calidad de un conjunto propuesto como solución. Este indicador está fuertemente influenciado por el tamaño del conjunto solución propuesto. Este efecto combinado con el esquema de desarrollo en los experimentos de duplicar el tamaño de la población en caso de que la población no sea admisible como solución puede falsear las conclusiones respecto a qué algoritmo tiene un mejor comportamiento que otro. Sin embargo hemos mantenido este diseño en los experimentos con fines de comparación con los trabajos de investigación previos.

El criterio de paro en las simulaciones puede también ser un elemento de discusión. Afirmar que la convergencia en el promedio de la evaluación de la población efectivamente indica que no hay mejora puede ser falso. La aparición de un nuevo óptimo local en la población o una redistribución de la población a lo largo de las vecindades de los óptimos puede esconderse y no reflejarse en un cambio en el promedio de las evaluaciones. Sin embargo en este caso no hay una solución clara y de fácil implementación que nos permita indicar cuando efectivamente no hay una mejora, y por consiguiente usaremos el criterio de paro del promedio de las evaluaciones de la población.

Como en los experimentos desarrollados por Mahfoud [36], usaremos el operador de cruce en un punto, fijando la probabilidad de cruce en 1.0, de tal suerte que siempre se ejecutará dicho cruce para producir nuevos individuos. La probabilidad de mutación se fijará en 0. Este selección de parámetros de recombinación parece adecuada; en ausencia de mutación los resultados obtenidos son más dependientes de la recombinación aplicada por el AG.

Por razones de claridad repetiremos los elementos que serán utilizados a lo largo de todos los experimentos.

- Para todos los problemas el único operador de cruce utilizado fue el operador de cruce en un punto. En cada creación de nuevos elementos siempre fue utilizado, es decir que la probabilidad de cruce se fijó en 1.0.

- En ninguna de las simulaciones se utilizó mutación. Esta determinación tuvo el fin de medir la capacidad de construir las soluciones a partir sólo de la información inicial, esto es, sin reintroducir información que pudiera perderse en el proceso de solución.

- El criterio de paro utilizado fue el mismo que el planteado en el capítulo 3; el criterio se activa en la generación k, si la diferencia entre el promedio de la evaluación de la población actual y el promedio del correspondiente promedio de la evaluación de las cuatro generaciones previas es menor que el número positivo s_o. A menos que se indique lo contrario el valor de s_o, se fijará en 0.001, el cual es consistente con los experimentos hechos por Mahfoud [36].

- El exponente utilizado en el cálculo de la función de compartición se fijó en $a = 2$. Este valor es el mismo que utilizó Mahfoud [36], el cual lo tomó de Beasley [35] argumentando *producir menores óptimos falsos en la función ajustada* utilizada en *sharing*.

- El criterio de aceptación de un conjunto de puntos P como solución al problema multimodal con conjunto de solución aceptada fue el mismo que el que se mencionó en el capítulo 3; P es aceptado como solución si para cada óptimo local existe al menos un elemento del conjunto propuesto que dista de él en menos que una cierta cantidad positiva llamada el radio nicho y que designaremos por r_o. A menos que se diga lo contrario esta cantidad será igual al valor de σ_{share} definido en el problema a resolver.

- En el algoritmo memGA, el microalgoritmo genético utilizado fue CD, debido a su mínimo *overhead* computacional.

- La población final reportada por memGA fueron la totalidad de individuos cuyo indicador de reemplazo fue menor que 2.

- A menos que se especifique lo contrario el tamaño de la población utilizada en la población exploradora de memGA fue 4.

- La distancia utilizada fue la distancia euclidiana fenotípica, salvo en los problemas M_7, M_8 y en los problemas de la familia $\{E_i\}$, en los cuales se utilizó la distancia de Hamming. Asimismo, en estos problemas se hizo uso del optimizador local basado puramente en el cromosoma del individuo. Este optimizador intenta todos los posibles cambios en todos alelos del cromosoma hasta que no hay mejora posible.

- Cada uno de los resultados reportados corresponde al promedio sobre 50 simulaciones que inician con una semilla diferente para el generador de números seudoaleatorios.

Por lo cambiante de los escenarios hemos preferido hacer el reporte de resultados para establecer las comparaciones agrupando los datos de acuerdo a la natulareza de los problemas.

5.4. Resultados experimentales

5.4.1. M_1 a M_5

Todos los parámetros especificados por omisión fueron respetados. Los valores para σ_{share} utilizados fueron los siguientes. Para las funciones de M_1 a M_4 el valor se fijó en 0.1, tal y como aparece en la literatura [36, 35, 33, 34]; para M_5, el valor fue de 4.24 por el mismo argumento. Los resultados promedio sobre las 50 simulaciones independientes se reportan en la tabla 5.1. En ella hemos reportado el desempeño tanto de memGA,

que corresponde a memGA$_{2,2}$, como el de memGA$_{16,16}$ que de acuerdo al análisis experimental del capítulo anterior tendría un mejor desempeño en este tipo de problemas. Hemos incluido esta información con el fin de contrastar las diferencias en los resultados obtenidos por las variantes de memGA. En la tabla se reportan los promedios del tamaño de la población, del número de generaciones, del número de evaluaciones empleadas por el AG, y el promedio de evaluaciones totales las cuales incluyen las del propio algoritmo genético y las del optimizador local. En negritas se resaltan los valores mínimos por cada función de prueba.

Observamos de la tabla 5.1 que referente al tamaño de población requerido para determinar con éxito un conjunto solución en general CD requiere menos recursos que sus contrincantes. Aunque podemos concluir que memGA no gana en ningún caso. Y los algoritmos restantes se intercambian la población más reducida. Referente al número de evaluaciones sí hay un claro ganador cuando se usa tanto en el número de evaluaciones por parte del algoritmo genético, como en el conteo de evaluaciones totales. Omitiendo el renglón correspondiente a memGA$_{16,16}$, el algoritmo memGA es el que presenta el menor número posible de evaluaciones tanto en evaluaciones del algoritmo como evaluaciones totales, salvo en el problema asociado a M_5 en el cual el algoritmo con menor número de evaluaciones totales en promedio es *clearing* pero la diferencia con memGA no es estadísticamente significativa. En lo relativo al tamaño de la población inicial el algoritmo que requiere menos recursos en general en promedio es CD, aunque en algunos casos particulares no existe diferencia significativa.

5.4.2. Experimentos sobre M_6

La función M_6 es la función *Sekel's Foxholes* la cual tiene 25 óptimos. Esta función aparece descrita en el capítulo 2. Los parámetros utilizados para los experimentos son los siguientes. En el criterio de paro se tomó con parámetro $s_o = 0.1$, valor que utilizó Mahfoud en su disertación [36]. Un valor tan grande respecto al empleado para las funciones M_1 a M_5 se justica a la luz de notar que las evaluaciones de los picos de la función M_6 se encuentran en el rango de 476.191 a 499.002, este último valor corresponde al único máximo global. La distancia utilizada fue la distancia euclideana sobre el fenotipo. El valor de σ_{share} se fijó en 8.0, el cual corresponde a la mitad de la mínima distancia entre dos óptimos locales los cuales tienen la forma $(16\,i, 16\,j)$ para i y j enteros en el rango de -2 a 2. Los experimentos fueron repetidos 50 veces y los resultados se ilustran en la tabla 5.2. De los resultados observamos que para M_6, DC consume más de los recursos asignados como límite en las simulaciones, de suerte que al ocurrir la convergencia en el promedio de las evaluaciones no todos los óptimos locales tienen en su vecindad puntos de la población. Lo que ocurre es que individuos en la vecindad de tales óptimos locales emigran a picos superiores dejándolas vacías antes que el criterio de paro se habilite. Este problema es uno donde la velocidad de emigración supera la velocidad de convergencia para DC. Observamos también que *sharing* es el algoritmo que requiere de una población menor que los otros para resolver el problema cuando se compara contra memGA o contra PHC. En el conteo de evaluaciones utilizadas solamente por el AG indica como claro ganador a memGA.

	Población		Generaciones		Evaluaciones del AG		Evaluaciones totales	
Método	n	σ	g	σ	μ	σ	μ	σ
M_1								
PHC	17.3	7.6					1121.8	497.6
SH	32.6	12.1	14.8	9.0	449.9	228.0	2498.8	833.7
CD	**15.5**	6.5	25.9	14.7	442.6	396.0	1416.2	732.9
Clearing	85.4	103.4	7.6	2.7	597.3	765.9	907.9	763.9
memGA$_{16,16}$	17.0	9.6	22.4	11.2	**218.3**	110.3	**597.5**	120.2
memGA$_{2,2}$	21.6	10.1	26.7	13.2	266.1	130.2	651.3	147.6
M_2								
PHC	15.8	6.9					1030.6	450.1
SH	42.2	17.0	10.1	3.8	412.8	183.4	3064.0	1197.9
CD	**14.6**	7.5	20.6	13.1	373.8	449.0	1294.9	891.5
Clearing	115.5	158.8	7.2	2.2	762.6	964.6	1073.3	962.8
memGA$_{16,16}$	15.3	6.0	18.6	8.0	**178.2**	79.2	**576.0**	108.1
memGA$_{2,2}$	23.84	10.3	26.0	11.5	261.1	117.3	657.7	131.8
M_3								
PHC	16.6	8.1					1081.5	520.1
SH	31.4	15.1	14.3	8.6	399.7	193.5	2356.4	1053.5
CD	**16.1**	9.0	25.1	15.5	494.8	559.0	1500.3	1082.1
Clearing	139.5	151.9	7.7	1.9	978.4	1113.7	1294.5	1117.1
memGA$_{16,16}$	19.3	11.6	22.8	11.3	**229.3**	115.4	**617.8**	129.1
memGA$_{2,2}$	24.6	11.6	30.8	13.0	311.6	189.5	696.66	156.1
M_4								
PHC	20.0	14.5					1295.0	941.6
SH	33.3	14.0	11.6	8.6	350.7	183.6	2434.3	951.2
CD	**15.3**	7.7	24.0	13.1	434.6	554.2	1396.2	1008.6
Clearing	241.9	322.2	7.8	2.4	1544.3	1970.4	1858.6	1971.8
memGA$_{16,16}$	17.6	9.7	21.1	9.1	**206.3**	89.1	**596.3**	104.6
memGA$_{2,2}$	27.0	9.6	29.1	10.6	297.0	113.2	692.8	144.9
M_5								
PHC	12.0	6.5					1025.5	554.4
SH	**10.6**	4.3	36.4	31.7	337.7	263.5	1231.7	464.3
CD	15.4	10.3	19.0	4.8	319.7	285.8	1538.0	1082.8
Clearing	29.9	24.4	8.7	3.7	210.0	141.7	**572.1**	159.9
memGA$_{16,16}$	19.2	28.0	14.8	6.6	**163.8**	81.0	604.4	149.4
memGA$_{2,2}$	15.5	25.4	16.0	7.2	173.6	92.8	593.9	147.0

Tabla 5.1: Resultados sobre 50 simulaciones sobre las funciones $M_1 - M_5$. La tabla reporta los promedios del tamaño de población requerida por el algoritmo genético (n), el número de generaciones que invirtió (g), el número de evaluaciones considerando solamente el AG, y el número de evaluaciones totales las cuales incluyen además las utilizadas por el optimizador local sobre el conjunto solución.

	Población		Generaciones		Evaluaciones del AG		Evaluaciones totales	
Método	n	σ	g	σ	μ	σ	μ	σ
M_6								
PHC	284.2	160.0					20270.4	11427.7
SH	**122.9**	42.6	51.2	45.0	5329.9	3087.6	12951.3	3929.8
CD					$>> 100k$			
Clearing	537.6	330.2	35.1	15.1	16681.9	10879.3	18159.2	10877.9
$\text{memGA}_{2,2}$	327.7	124.4	104.0	34.12	**4195.2**	1451.4	**7763.8**	1931.5

Tabla 5.2: Resultados sobre 50 simulaciones sobre la función M_6. La tabla reporta los promedios del tamaño de población requerida por el algoritmo genético (n), el número de generaciones que invirtió (g), el número de evaluaciones considerando solamente el AG, y el número de evaluaciones totales las cuales incluyen además las utilizadas por el optimizador local sobre el conjunto solución.

5.4.3. Variaciones a la función M_2

Los experimentos realizados para las funciones M_1 a M_6 se repitieron para las funciones obtenidas al modificar la función M_2. El tamaño de la población exploradora se fijó en 4 individuos, el tamaño de la población memoria inicial también fue de 4. El valor del parámtro s_o utilizado en el criterio de paro tuvo el valor 0.001. El valor del radio de compartición o σ_{share} se fijó en el valor 0.1 para las funciones M_{2c} a M_{2i}; para la función M_{2a} el valor utilizado fue de $1/(2 \times 4) = 0.125$, para la función M_{2b}, $1/(2 \times 6) = 0.833$. El valor utilizado fue de r_o utilizado fue el de σ_{share}. Los resultados sobre 50 simulaciones independientes aparecen en las tablas 5.3 y 5.4. En las tablas no hay un ganador consistente. En tamaño de la población requerida CD y PHC son las dos estrategias que requieren una población menor en general, aunque en algunos casos no hay una diferencia estadísticamente significativa entre ellos. MemGA es la estrategia que ocupa en general el tercer lugar, y ocasionalmente el primero aunque sin una diferencia significativa. Es de llamar la atención el gran tamaño de población requerido por *clearing* comparativamente. Respecto al número de evaluaciones requeridas por la estrategia y en el número de evaluaciones totales la estrategia que requiere menos es memGA, siendo seguida en varias ocaciones por *clearing*, lo cual es de llamar la atención debido al tamaño de la población que es grande en promedio, pero ello es consistente con los experimentos inmediatos anteriores.

5.4.4. Familia $\{S_i\}_{i=6,\cdots,35}$

Para cada uno de los problemas de la familia $\{S_i\}_{i=6,\cdots,35}$ se repitieron 50 experimentos independientes. El parámetro s_o utilizado en el criterio de paro se fijó en 0.001. La función S_i tiene i óptimos locales, distribuidos casi uniformemente en $[0, 1]$ y se encuentran aproximadamente en los puntos $(2j + 1)/(2i)$ para $j = 0, 1, ...(i - 1)$, los valores de estos óptimos calculados por Mathematica aparecen en el apéndice 1. El va-

	Población		Generaciones		Evaluaciones del AG		Evaluaciones totales	
Método	n	σ	g	σ	μ	σ	μ	σ
M_{2a}								
PHC	14.7	9.0	.0	.0			952.2	579.8
SH	23.4	9.0	12.5	4.6	287.5	150.0	1738.3	666.7
CD	**11.6**	5.7	19.6	12.4	270.8	298.5	995.8	622.3
Clearing	79.4	155.2	8.1	3.2	546.1	933.0	793.4	933.1
memGA$_{2,2}$	17.1	7.8	21.0	6.8	**209.8**	74.1	**502.7**	100.1
M_{2b}								
PHC	21.1	11.6					1374.0	758.9
SH	47.0	20.0	10.6	6.0	460.8	181.5	3409.4	1385.4
CD	**20.0**	8.7	32.0	20.3	752.8	725.1	2001.6	1202.1
Clearing	204.2	266.1	8.4	3.3	1570.0	2118.4	1940.3	2115.9
memGA$_{2,2}$	28.5	12.1	32.5	12.1	**333.8**	126.5	**811.4**	164.7
M_{2c}								
PHC	**14.7**	7.5					955.8	484.5
SH	67.2	29.8	8.0	2.2	519.7	212.0	4733.5	2046.5
CD	16.2	8.3	20.0	14.2	396.8	474.5	1416.2	935.2
Clearing	256.0	254.1	6.1	1.1	1562.3	1565.4	1873.2	1564.9
memGA$_{2,2}$	26.9	12.3	25.5	10.4	**255.4**	103.1	**672.4**	119.7
M_{2d}								
PHC	**13.4**	5.5					876.0	360.1
SH	30.1	12.4	13.4	7.5	370.6	180.4	2261.7	849.3
CD	14.4	7.0	21.0	13.0	345.3	329.1	1256.6	734.1
Clearing	141.8	217.9	6.9	2.2	864.7	1215.2	1178.0	1212.1
memGA$_{2,2}$	25.0	11.3	28.2	10.6	**284.1**	115.5	**680.9**	136.1
M_{2e}								
PHC	18.7	8.9					1232.3	587.6
SH	62.1	23.7	9.6	4.1	561.9	227.8	4451.9	1626.8
CD	**15.8**	6.4	15.8	12.7	299.4	391.3	1314.6	742.3
Clearing	133.8	126.3	7.2	2.5	889.3	778.1	1202.1	778.0
memGA$_{2,2}$	41.3	17.1	36.1	15.2	**369.9**	156.1	**778.2**	175.3

Tabla 5.3: Resultados sobre 50 simulaciones sobre variaciones a la función M_2. La tabla reporta los promedios del tamaño de población requerida por el algoritmo genético (n), el número de generaciones que invirtió (g), el número de evaluaciones considerando solamente el AG, y el número de evaluaciones totales las cuales incluyen además las utilizadas por el optimizador local sobre el conjunto solución.

	Población		Generaciones		Evaluaciones del AG		Evaluaciones totales	
Método	n	σ	g	σ	μ	σ	μ	σ
M_{2f}								
PHC	16.2	7.3					1034.4	468.7
SH	32.5	15.7	8.7	3.4	276.8	173.3	2319.7	1118.4
CD	16.6	7.0	20.2	9.5	367.7	299.3	1408.3	706.0
Clearing	56.3	28.6	6.6	2.1	350.1	234.7	798.4	292.9
memGA$_{2,2}$	**15.5**	6.6	20.1	8.1	**199.4**	81.8	**652.2**	133.5
M_{2g}								
PHC	**16**	9.7					1043.5	633.4
SH	35.5	13.4	11.3	4.4	377.6	139.4	2603.4	911.8
CD	17.9	8.2	26.5	15.5	548.8	495.5	1667.9	950.0
Clearing	115.5	150.3	6.8	1.8	736.6	909.4	1048.6	908.1
memGA$_{2,2}$	25.4	11.4	28.7	12.0	**292.0**	121.4	**701.7**	114.1
M_{2h}								
PHC	18.1	8.1					1174.1	521.6
SH	37.9	14.7	10.3	3.7	372.5	137.1	2757.7	1005.6
CD	**13.3**	5.5	25.4	11.9	369.4	302.9	1200.4	612.2
Clearing	123.5	168.2	8.2	2.6	891.4	1089.5	1202.1	1086.6
memGA$_{2,2}$	28.8	13.2	31.5	14.6	**326.8**	155.8	**742.4**	176.0
M_{2i}								
PHC	**16.8**	7.5					1092.5	484.5
SH	45.8	17.4	9.5	4.6	404.2	166.3	3308.8	1189.6
CD	19.8	14.0	26.7	11.4	654.2	770.0	1877.3	1603.6
Clearing	93.8	84.7	8.2	3.8	679.9	535.0	993.4	533.0
memGA$_{2,2}$	22.6	11.7	25.1	9.8	**266.3**	105.9	**659.3**	143.3

Tabla 5.4: Resultados sobre 50 simulaciones sobre variaciones a la función M_2 (Conclusión).

lor de σ_{share} así como el radio de nicho, r_0, utilizado para determinar si un conjunto es aceptado como solución para la función S_i se fijó en $1/(2\,i)$. Los resultados obtenidos se ilustran en las tablas 5.5, 5.6, 5.7, 5.8, y 5.9. En los resultados de las tablas observamos comportamientos extraños de CD; existen valores de n que dan origen a problemas que no pueden ser resueltos por CD con pocos recursos. Esto se debe a problemas con emigración de nichos de baja evaluación a nichos de mayor evaluación. La cantidad de recursos requeridos por SH para resolver los problemas crece muy rápidamente cuando n aumenta; tanto en número de evaluaciones, evaluaciones totales y tamaño de población requerido. El comportamiento de memGA es mejor que sus oponentes en evaluaciones del AG y en evaluaciones totales en las comparaciones contra el método de *clearing* que es su más cercano oponente respecto al número de evaluaciones, y el cual tiene el defecto del tamaño de población requerido que aumenta considerablemente. Respecto al tamaño de población en la particular el algoritmo que mejor se desempeña en promedio sobre el total de elementos de la familia de problemas es PHC, seguido de cerca y cuando no presenta comportamientos negativos por DC; memGA ocupa consistentemente el tercer lugar y cuando DC presenta anomalías, el segundo.

5.4.5. Familia $\{E_i\}_{n=1..5}$

Para las simulaciones relativas a la familia $\{E_i\}_{n=1..5}$, el diseño de los experimentos ha sido cambiado. Por limitaciones en el tiempo de cómputo, se ha modificado el esquema de simulación: ahora toda simulación inicia con una población aleatoria de 32 individuos, la población es evolucionada por la estrategia hasta que el criterio de paro se dispara; si la población no es admitida como solución, la población aleatoria inicial es aumentada en 32 individuos aleatoriamente generados y la simulación inicia nuevamente. La estrategia de duplicar la población realizada en las simulaciones anteriores demandaba una cantidad de recursos excesivos para la realización de un número importante de experimentos, ello debido a que los tamaños de población aumentan ahora en forma considerable y la evaluación de individuos requiere un mayor tiempo de cálculo. Asimismo, las simulaciones realizadas con memGA mostraron un comportamiento deficiente que experimentalmente fue mejorado seleccionando el tamaño de la población exploradora: en los problemas asociados a las funciones E_1 a E_4 el tamaño de la población exploradora se fijó en 16, el cual condujo a deficiencias en el desempeño en la función E_5, el cual fue corregido aumentando la misma a un valor de 24. Las deficiencias se refieren a tamaños elevados en la población requerida, lo cual conlleva a números elevados de comparaciones en la determinación del parámetro N_1. Algo que vale la pena comentar es que memGA intenta retener la totalidad de los óptimos localizados en la evolución de la población, cuyo número crece grandemente aumentando i, por lo cual el tamaño de la población requerido aumenta sensiblemente por la cantidad de subóptimos locales de la función. Estos ajustes del tamaño hacen injusta la comparación de memGA contra las otras estrategias, pero refleja el potencial propio del algoritmo. Estos experimentos muestran el buen desepeño de *clearing* en los problemas con mayor número de óptimos; por otro lado memGA requiere un bajo tamaño de población pero no es mejor que *clearing*.

	Población		Generaciones		Evaluaciones del AG		Evaluaciones totales	
Método	n	σ	g	σ	μ	σ	μ	σ
				S_6				
PHC	22.2	12.3					1884.9	1041.1
SH	53.1	24.7	11.0	4.3	534.4	190.4	4910.1	2159.6
DC	**22.1**	10.0	23.8	14.9	627.5	674.2	2453.9	1449.0
Clearing	284.8	307.0	7.6	2.2	2009.5	2297.3	2499.8	2294.1
memGA	30.7	12.0	22.6	8.2	**478.6**	169.0	**1084.9**	189.9
				S_7				
PHC	**24.3**	11.3					2068.0	961.3
SH	61.4	24.0	8.9	2.7	**526.7**	201.3	5610.3	2133.7
DC	25.1	10.0	25.7	14.4	717.8	594.4	2801.3	1346.0
Clearing	351.4	367.0	7.3	2.1	2374.3	2722.5	2947.3	2718.3
memGA	43.8	17.0	27.6	10.1	583.0	205.0	**1321.9**	224.0
				S_8				
PHC	**30.9**	14.5					2562.5	1203.5
SH	57.0	20.7	13.2	5.2	688.6	198.2	5277.1	1756.9
DC	42.6	36.7	48.6	18.9	2485.1	3831.3	5806.4	6565.2
Clearing	357.1	345.8	7.3	2.2	2260.3	1963.5	2895.5	1964.0
memGA	44.2	15.7	26.0	9.3	**626.6**	231.8	**1454.9**	243.8
				S_9				
PHC	45.4	23.8					3781.7	1979.0
SH	92.8	33.1	8.4	2.1	766.7	282.2	8270.9	2904.3
DC	**32.0**	11.2	38.3	17.4	1329.6	968.0	3925.5	1795.1
Clearing	486.4	410.5	7.1	2.4	3078.3	2493.8	3796.9	2492.2
memGA	55.7	22.0	31.6	12.9	**700.7**	283.7	**1667.5**	322.1
				S_{10}				
PHC	**43.2**	17.8					3596.8	1480.2
SH	90.2	41.2	9.8	2.8	840.3	316.0	8132.3	3585.2
DC	45.8	26.1	50.8	22.0	2696.0	2760.1	6351.2	4714.2
Clearing	521.0	417.6	7.8	2.8	3525.7	2795.1	4324.0	2790.7
memGA	65.3	31.0	34.7	11.7	**793.6**	269.8	**1854.4**	294.6
				S_{11}				
PHC	48.0	20.4					3994.0	1696.4
SH	117.8	39.6	8.6	1.9	1002.2	347.6	10512.5	3479.9
DC	**45.8**	16.5	47.5	17.4	2387.8	1513.5	6069.5	2772.0
Clearing	622.1	433.1	7.4	2.2	3981.3	2614.3	4858.1	2611.8
memGA	71.0	29.1	35.3	13.0	**805.3**	297.1	**1987.2**	314.2

Tabla 5.5: Resultados sobre 50 simulaciones sobre la familia de funciones S_i, $i = 6, 7, \cdots, 35$. La tabla reporta los promedios del tamaño de población requerida por el algoritmo genético (n), el número de generaciones que invirtió (g), el número de evaluaciones considerando solamente el AG, y el número de evaluaciones totales las cuales incluyen además las utilizadas por el optimizador local sobre el conjunto solución.

	Población		Generaciones		Evaluaciones del AG		Evaluaciones totales	
Método	n	σ	g	σ	μ	σ	μ	σ
S_{12}								
PHC	**53.8**	21.9					4472.4	1816.7
SH	120.3	49.5	10.3	3.5	1145.6	349.2	10840.0	4240.9
DC	261.8	281.6	91.5	32.8	31625.6	43228.5	51399.7	63994.9
Clearing	527.4	409.0	8.6	3.0	4042.9	3121.7	4994.5	3116.2
memGA	69.8	27.9	33.5	10.2	**833.3**	260.4	**2129.7**	301.6
S_{13}								
PHC	58.2	16.7					4835.9	1387.3
SH	153.6	64.6	8.5	1.9	1253.1	444.0	13646.2	5617.9
DC	**55.7**	18.1	55.2	20.4	3319.0	2064.1	7781.2	3393.6
Clearing	565.8	420.2	7.7	2.6	3668.4	2327.4	4702.6	2327.1
memGA	95.4	45.9	39.5	17.4	**935.0**	409.3	**2408.1**	446.1
S_{14}								
PHC	**67.8**	27.1					5629.5	2252.0
SH	139.5	75.0	10.1	3.9	1264.6	449.1	12533.4	6435.2
DC	92.8	53.4	75.0	29.0	8256.0	8351.9	15518.2	12313.5
Clearing	619.5	415.7	8.0	3.4	4102.9	2413.4	5220.5	2412.4
memGA	101.1	45.0	39.9	13.9	**1005.2**	355.7	**2624.7**	383.7
S_{15}								
PHC	74.9	39.1					6217.4	3244.6
SH	166.4	63.3	8.4	1.6	1363.2	478.3	14801.7	5543.0
DC	**67.2**	25.2	62.3	14.3	4409.0	2527.6	9777.0	4463.0
Clearing	651.5	417.3	7.3	2.2	4121.6	2504.7	5321.2	2506.9
memGA	105.0	38.3	45.9	14.6	**1073.6**	335.7	**2740.8**	372.3
S_{16}								
PHC	**89.0**	39.5					7214.3	3209.0
SH	158.7	55.2	11.3	2.6	1707.5	353.3	14176.5	4621.1
DC	779.5	371.8	110.5	30.7	96272.6	53424.2	153107.0	79968.6
Clearing	445.4	359.5	9.1	3.9	3259.8	1911.0	4499.7	1910.5
memGA	107.5	49.1	43.0	12.7	**1229.3**	394.9	**3015.9**	458.2
S_{17}								
PHC	90.9	42.1					7382.1	3423.2
SH	215.0	75.1	8.2	1.8	1728.0	545.1	18699.1	6381.6
DC	**82.6**	30.4	65.9	17.2	5693.4	3099.1	12130.6	5303.1
Clearing	675.8	403.1	7.6	2.4	4375.6	2113.7	5704.8	2114.5
memGA	113.9	34.9	44.2	13.4	**1075.8**	316.4	**2963.2**	341.6
S_{18}								
PHC	**96.0**	39.3					7803.6	3200.0
SH	197.1	64.4	9.9	2.4	1889.3	569.8	17439.6	5539.4
DC	157.4	90.7	92.9	32.2	17070.1	15673.0	28949.1	22152.2
Clearing	688.6	365.6	8.3	4.0	4764.2	1990.6	6167.0	1988.6
memGA	125.4	40.8	45.5	15.8	**1167.9**	384.6	**3187.4**	401.3

Tabla 5.6: Resultados sobre 50 simulaciones sobre la familia de funciones S_i, $i = 6, 7, \cdots, 35$ (Continuación)

	Población		Generaciones		Evaluaciones del AG		Evaluaciones totales	
Método	n	σ	g	σ	μ	σ	μ	σ
S_{19}								
PHC	101.1	50.2					8205.9	4072.5
SH	227.8	69.8	8.5	1.8	1889.3	503.1	19813.1	5908.8
DC	**94.7**	39.3	67.7	17.3	6853.1	4309.0	14198.1	7247.5
Clearing	706.6	370.5	7.6	2.7	4657.2	1867.9	6138.2	1872.7
memGA	126.7	24.2	47.9	12.5	**1170.3**	258.4	**3357.2**	300.1
S_{20}								
PHC	**120.3**	49.5					9751.2	3996.7
SH	227.8	82.9	9.5	2.4	2081.3	626.4	20008.8	7018.4
CD					$>> 100k$			
Clearing	663.0	375.4	9.0	4.4	4817.2	2057.0	6377.5	2058.2
memGA	135.7	38.0	50.5	14.2	**1359.7**	396.9	**3624.1**	427.3
S_{21}								
PHC	111.4	48.0					9037.0	3912.0
SH	284.2	116.5	8.2	1.6	2245.1	767.2	24634.2	9851.7
DC	**110.1**	43.0	71.5	13.8	8090.9	3967.2	16617.0	7162.5
Clearing	696.3	357.5	8.4	4.0	4875.1	1890.7	6514.6	1892.8
memGA	166.4	74.3	57.9	22.7	**1447.1**	585.5	**3856.5**	599.3
S_{22}								
PHC	**111.4**	42.5					9024.9	3443.3
SH	276.5	113.8	9.0	2.1	2368.0	732.0	24148.1	9580.8
DC	358.4	306.2	120.3	41.5	53456.6	58714.1	80073.5	80911.6
Clearing	693.8	358.4	7.9	2.8	4732.8	1873.2	6445.2	1871.6
memGA	161.3	61.0	52.3	18.7	**1396.5**	477.0	**3963.5**	529.2
S_{23}								
PHC	**131.8**	47.3					10692.3	3845.7
SH	294.4	97.6	8.7	2.3	2455.0	645.8	25670.9	8172.4
DC	134.4	59.6	72.3	18.3	10439.7	6651.5	20842.3	11109.7
Clearing	686.1	340.9	7.9	2.6	4753.6	1808.3	6550.2	1815.4
memGA	192.0	64.6	62.7	19.3	**1618.2**	497.1	**4211.1**	490.4
S_{24}								
PHC	**131.8**	41.7					10690.9	3385.0
SH	256.0	77.6	10.6	2.8	2588.2	520.0	22682.4	6432.2
CD					$>> 100k$			
Clearing	640.0	332.2	10.1	5.3	5344.0	2158.6	7203.9	2154.0
memGA	157.4	56.7	52.7	16.3	**1519.8**	498.8	**4292.7**	605.6
S_{25}								
PHC	**153.6**	60.6					12454.5	4910.1
SH	337.9	150.2	8.4	1.8	2703.4	874.9	29314.6	12613.0
DC	169.0	78.4	78.2	13.5	13683.2	8020.1	26712.5	13921.3
Clearing	675.8	349.9	9.4	4.5	5273.8	2167.4	7225.2	2170.4
memGA	209.9	76.5	63.3	23.2	**1636.6**	597.6	**4588.2**	653.2

Tabla 5.7: Resultados sobre 50 simulaciones sobre la familia de funciones S_i, $i = 6, 7, \cdots, 35$. (Continuación)

	Población		Generaciones		Evaluaciones del AG		Evaluaciones totales	
Método	n	σ	g	σ	μ	σ	μ	σ
S_{26}								
PHC	**138.2**	52.3					11222.2	4251.5
SH	294.4	97.6	9.3	2.2	2670.1	761.0	25845.9	8241.9
DC	506.9	325.0	135.7	35.6	78871.0	63930.4	116294.0	87366.6
Clearing	655.4	331.8	9.5	5.4	5065.4	1913.5	7092.7	1917.8
memGA	189.4	64.6	60.7	19.8	**1625.5**	522.9	**4631.3**	533.3
S_{27}								
PHC	156.2	58.1					12648.5	4697.1
SH	358.4	126.7	8.1	1.4	2805.8	763.6	31012.6	10663.3
DC	**152.3**	53.1	74.3	14.7	11544.3	5371.6	23299.3	9255.6
Clearing	783.4	332.8	8.5	4.7	5432.1	1727.2	7545.2	1726.6
memGA	243.2	74.3	73.4	20.7	**1912.0**	541.0	**5049.4**	492.1
S_{28}								
PHC	**151.0**	54.5					12254.2	4427.6
SH	325.1	119.1	9.7	2.1	3018.2	804.6	28631.0	10059.2
CD					$>> 100k$			
Clearing	696.3	319.0	9.0	3.6	5501.4	2158.1	7683.6	2157.1
memGA	222.7	72.3	68.5	20.5	**1960.6**	553.6	**5255.5**	598.1
S_{29}								
PHC	172.8	63.7					14018.4	5162.7
SH	399.4	128.4	8.4	1.9	3215.4	763.9	34693.7	10765.6
DC	**169.0**	64.3	78.8	13.6	13833.0	6833.0	26844.9	11648.0
Clearing	742.4	307.1	7.7	3.2	5121.7	1736.1	7386.1	1740.1
memGA	279.0	102.4	81.3	30.0	**2162.9**	787.9	**5570.3**	781.5
S_{30}								
PHC	**170.2**	62.8					13814.0	5097.6
SH	399.4	156.5	9.0	2.1	3415.0	1015.6	34900.3	13201.6
CD					$>> 100k$			
Clearing	768.0	314.6	8.2	3.7	5414.7	1691.7	7765.1	1697.3
memGA	243.2	86.7	73.2	24.4	**1986.6**	641.5	**5560.9**	698.9
S_{31}								
PHC	**174.1**	62.1					14109.9	5015.8
SH	491.5	144.8	8.0	1.6	3804.2	885.3	42522.2	12143.8
DC	189.4	64.6	82.5	11.4	15849.0	6403.5	30416.1	11176.5
Clearing	809.0	285.0	7.9	3.2	5697.7	1656.4	8121.6	1653.7
memGA	297.0	104.8	85.6	27.2	**2282.6**	728.4	**5895.2**	709.1
S_{32}								
PHC	**176.6**	62.8					13967.9	4962.9
SH	325.1	119.1	11.8	3.7	3576.3	744.3	28466.6	9690.9
CD					$>> 100k$			
Clearing	824.3	289.3	9.2	4.0	6590.7	1943.1	9015.3	1938.0
memGA	238.1	63.4	66.6	17.3	**2146.0**	521.4	**5889.2**	609.9

Tabla 5.8: Resultados sobre 50 simulaciones sobre la familia de funciones S_i, $i = 6, 7, \cdots, 35$ (Continuación)

	Población		Generaciones		Evaluaciones del AG		Evaluaciones totales	
Método	n	σ	g	σ	μ	σ	μ	σ
S_{33}								
PHC	**197.1**	64.4					15589.2	5092.4
SH	471.0	130.4	8.2	1.4	3747.8	754.4	39964.4	10663.7
DC	212.5	61.2	82.2	10.0	17756.2	6244.7	33711.7	10663.5
Clearing	788.5	287.2	8.6	4.0	5824.5	1445.0	8336.7	1450.5
memGA	289.3	92.6	83.3	21.4	**2235.1**	615.2	**5970.4**	617.4
S_{34}								
PHC	**202.2**	63.8					16000.4	5056.5
SH	455.7	107.1	8.9	1.7	3921.9	788.5	38885.9	8803.6
CD					$>> 100k$			
Clearing	798.7	271.9	8.8	4.8	6134.6	2196.8	8729.8	2198.4
memGA	289.3	92.6	80.0	23.1	**2260.1**	632.6	**6190.1**	646.5
S_{35}								
PHC	**220.2**	85.9					17399.6	6801.2
SH	547.8	171.6	8.1	1.1	4357.1	1019.4	46408.6	14052.2
DC	220.2	73.3	82.4	12.3	18544.6	8211.6	35023.4	13466.6
Clearing	819.2	253.4	7.8	2.8	5839.7	1442.9	8510.7	1438.3
memGA	291.8	89.7	82.6	24.5	**2221.4**	668.7	**6218.7**	659.7

Tabla 5.9: Resultados sobre 50 simulaciones sobre la familia de funciones S_i, $i = 6, 7, \cdots, 35$ (Conclusión)

	Población		Generaciones		Evaluaciones del AG		Evaluaciones totales	
Método	n	σ	g	σ	μ	σ	μ	σ
E_1								
PHC	59.5	27.4					**655.7**	305.0
SH	**49.9**	20.6	19.6	13.0	918.4	606.8	1363.2	703.1
CD	54.4	22.6	15.7	3.0	862.7	394.6	1191.9	520.5
Clearing	91.5	54.1	10.0	9.8	**650.2**	290.4	808.6	298.1
memGA	79.4	43.0	10.2	2.7	908.5	310.7	1018.5	325.1
E_2								
PHC	954.2	331.8					22493.0	7799.7
SH	101.8	32.8	40.9	27.7	**3860.5**	2671.6	5964.6	2774.9
CD	76.8	25.9	79.0	15.4	6220.2	2768.4	7141.8	3056.6
Clearing	120.3	56.2	49.7	31.6	4793.2	3894.5	**5557.6**	4004.6
memGA	**67.8**	24.7	26.4	10.3	7999.0	3317.9	8774.5	3407.6
E_3								
PHC	1825.3	318.7					66213.4	11556.1
SH	339.8	81.8	57.9	47.5	19094.4	15385.7	30457.1	15971.1
CD	107.5	32.1	89.2	20.0	9840.0	4448.0	11776.4	4970.7
Clearing	179.8	66.8	74.9	48.3	**9378.5**	5582.4	**11696.2**	5780.6
memGA	**77.4**	36.1	40.3	23.8	14753.6	7730.5	16699.4	7807.4
E_4								
PHC					$>> 100k$			
SH	1580.0	294.3	34.8	17.3	55525.3	32115.6	128624.3	39293.3
CD	194.3	60.1	97.1	11.0	19187.4	7034.6	23855.0	8427.0
Clearing	212.9	88.7	107.7	46.5	**15247.0**	6266.0	**18782.1**	7118.8
memGA	**126.9**	70.3	39.5	17.2	18633.5	6566.5	23747.5	7102.4
E_5								
PHC					$>> 100k$			
SH					$>> 100k$			
CD	412.2	84.0	106.4	8.4	48111.4	10931.7	60480.4	13361.8
Clearing	256.0	95.9	111.9	50.8	**19630.9**	9473.8	**24786.7**	10317.2
memGA	**167.2**	64.8	38.4	10.1	35232.7	8266.1	43925.9	9130.5

Tabla 5.10: Resultados sobre 50 simulaciones de la aplicación de PHC, CD, SH, y memGA sobre la familia de funciones E_i, $i = 1, \cdots, 5$.

	Población		Generaciones		Evaluaciones del AG		Evaluaciones totales	
Método	n	σ	g	σ	μ	σ	μ	σ
M_7								
memGA(24)	284.8	80.5	33.3	12.61	16656.2	6952.18	31863.2	10805.5
memGA(16)	368.0	139.3	64.9	15.7	17324.6	5236.3	36944.6	11617.7

Tabla 5.11: Resultados sobre 50 simulaciones de memGA sobre M_7 utilizando sobre la población exploradora *clearing* con poblaciones de tamaño 16 y 24.

Adicionalmente a los experimentos anteriores, se realizaron otros cambiando sólo el algoritmo genético en la población exploradora; ahora se experimentó utilizando el mismo método de *clearing* con un diferente tamaño en la población exploradora; los resultados obtenidos pueden resumirse en la tabla 5.11. Como podemos observar, en este problema se mejora el rendimiento de memGA utilizando *clearing* pero sin embargo la mejora no supera los resultados obtenidos por el mismo método de *clearing*. También se experimentó usando *sharing* pero los resultados obtenidos fueron negativos; los recursos requeridos para la solución rebasaron el límite fijado y se consideró que no pudo obtener la solución multimodal. Como podemos notar, no existe una clara respuesta respecto a qué algoritmo utilizar en la población exploradora; hemos elegido CD pero otras opciones pueden ser redituables y otras no.

5.4.6. Problema asociado a M_8

Los parámetros en este problema son los mismos que los usados para la función E_5. En el problema M_8, ni PHC ni memGA compiten; la cantidad de recursos que requieren es excesiva; esto se debe a que el criterio de paro se habilita muy rápidamente debido a que los óptimos no globales tienen una altura muy baja respecto a los óptimos globales y a pesar de que se incorporen varios de ellos en la población memoria el incremento del promedio de la evaluación de la población es muy bajo, y antes que se localizen óptimos nuevos óptimos el criterio de paro se habilita. Las simulaciones que reporta la tabla 5.12 corresponden a la metodología de duplicar el tamaño de población adicionando una población aleatoria. En este problema, el desempeño de SH es adecuado; es estadísticamente igual en tamaño de la población requerida al que requiere menos población y es la estrategia que requiere un menor número de evaluaciones.

5.5. Resumen

En este capítulo hemos hecho una comparación experimental entre CD, SH, y al algoritmo memGA = $\text{memGA}_{2,2}$ en un marco que en general abarca el marco de referencia experimental usado en la literatura y un marco de referencia adicional. En problemas considerados de dificultad media y baja el algoritmo se desempeña mejor

	Población		Generaciones		Evaluaciones del AG		Evaluaciones totales	
Método	n	σ	g	σ	μ	σ	μ	σ
M_8								
PHC					$>> 100k$			
SH	428.2	131.7	28.2	17.9	**15603.8**	7160.3	**31946.0**	9784.0
CD	**427.5**	101.6	127.6	9.7	55031.0	15274.7	67859.0	18240.3
Clearing	674.6	113.8	81.3	22.8	54687.4	23191.0	56685.4	23274.3
memGA					$>> 100k$			

Tabla 5.12: Resultados sobre 50 simulaciones sobre el problema M_8

que los algoritmos contra los que se compara; a saber *crowding* determinístico, el método de las funciones compartidas, y contra *clearing*. En problemas de dificultad alta, donde el número de picos crece y muchos de ellos son de altura muy baja el algoritmo tiende a retener un número excesivo de ellos haciendo elevado el consumo de recursos, primordialmente la población. Algunos de estos problemas, por ejemplo los de la familia $\{E_i\}$, pueden ser resueltos adecuadamente por la estrategia en caso de tener un mecanismo de control del tamaño de la población exploradora. En algunos otros como M_8, otros elementos intervienen para que simplemente incrementar la poblaciones conduzcan al algoritmo memGA a la solución del problema.

Capítulo 6

El problema de ruteo de aviones con múltiples paradas

En esta sección se presenta la aplicación del algoritmo desarrollado en la presente investigación al problema de ruteo de vuelos con múltiples paradas. Primeramente se formulan los objetivos que se persiguen en la realización de esta aplicación cuando el problema asociado es formulado como un problema de optimización multimodal. Posteriormente se proporciona una codificación binaria para las posibles soluciones y una forma de obtener la aptitud de una posible solución. Utilizando un marco de referencia experimental reducido se evalúan las ventajas de formular el problema como un problema multimodal. Finalmente se presentan las conclusiones obtenidas.

6.1. Metas

Debido a que la industria de la aviación comercial es multimillonaria, los ahorros en algunos de sus gastos o costos de operación involucran cifras considerablemente grandes. Uno de los renglones importantes de su operación es la planeación o programación de rutas. Este rubro impacta importantemente en las ganancias y gastos de las compañias y es un área directamente ligada a la investigación de operaciones.

El problema de la programación de rutas áreas es un problema de alta complejidad, no tanto en la formulación matemática del mismo, sino en la búsqueda de sus soluciones debido a la cantidad excesiva de cálculos que involucra, cuyo número aumenta exponencialmente con el tamaño del problema.

Como fue formulado en el capítulo 2, el problema que abordamos está íntimamente relacionado con los procesos de creación y planeación y rutas. La descripción del problema puede enunciarse como la determinación de una ruta a lo largo de una red de ciudades con sólo una ciudad origen y una ciudad término. La solución buscada, además de la ruta, debe indicar el tipo de avión escogido dentro de un conjunto de aviones con diferentes características, así como de la indicación de las cantidades y movimientos de pasajeros que transporta a lo largo de la ruta.

El proceso de construcción de una ruta es un ciclo entre dos fases. Una de ellas involucra la creación de una posible solución y la otra consiste de un proceso de revisión

de factibilidad de la solución propuesta. En general, la factibilidad se revisa tomando en cuenta consideraciones que en general son difíciles de codificar matemáticamente.

Para la fase de construcción de las rutas, un gran número de factores deben tomarse en consideración. El conjunto de rutas preestablecidas, la frecuencia del servicio, los tráficos estimados, así como la ganancia que obtiene la compañía trasladando pasajeros entre ciudades de cada ruta. También intervienen las características propias de la flotilla de aviones y sus gastos de operación. Con todos esos elementos, la fase de construcción de la ruta con la planeación de tiempos de salida y llegada, asignando tipos de aviones específicos para satisfacer las decisiones de frecuencia en el ruteo. Durante la fase de evaluación, el personal de operaciones examina los resultados tomando en cuenta factibilidad y desempeño. Cualquier deseo manifiesto de mejora, crítica o comentario se proporciona como retroalimentación y se regresa a la fase de construcción de la ruta continuando este ciclo hasta la obtención de una programación satisfactoria.

El ruteo de aviones con múltiples paradas es uno de los principales problemas que se enfrentan en la fase de contrucción. El problema ha sido considerado en la tesis doctoral de Salvador García-Lumbreras [41], pero en el enfoque ahí propuesto el problema está formulado como un problema de optimización global. En este enfoque la idea principal consiste en encontrar planos cortantes ajustados (*strong cutting planes*) para reducir sustancialmente el espacio de búsqueda. Estos planos cortantes son desigualdades válidas que se añaden al conjunto de restricciones para apresurar la solución al resolver el problema entero mixto [66]. La presente investigación tiene como objetivo aplicar las técnicas de optimización multimodal presentadas en esta investigación como una propuesta de aceleración del proceso de construcción y evaluación de las rutas construidas así como en el enriquecimiento en la toma de decisiones al presentarse un conjunto de diferentes soluciones en lugar de una única solución.

El deseo de desarrollar esta aplicación tiene los siguientes elementos:

- El problema es económicamente relevante.
- El problema es teóricamente importante, debido a que es del tipo NP-difícil, al ser equivalente al problema de flujo en una red con argo fijo, esl cual es reconocido como del tipo NP-difícil [41, 66].
- Proveer soluciones subóptimas al problema tiene sentido, debido a que no es un problema de decisión o existencia.
- Proveer un conjunto de soluciones diversas presenta un marco enriquecedor para los tomadores de decisiones, y podría ayudar a mejorar y/o acelerar el proceso de decisión.
- La forma como está construido el problema hace que el costo computacional de la evaluación de individuos sea bastante mayor que el costo correspondiente al de la determinación de los indicadores de reemplazo y selección. Podríamos decir que si el ahorro en número de evaluaciones que ha mostrado memGA respecto a SH y a CD en los problemas donde se ha experimentado se vuelve a repetir,

entonces bien vale la pena invertir esfuerzo computacional en el cálculo de los indicadores mencionados.

6.2. Codificando las posibles soluciones

Uno de los factores de éxito en la aplicación de AGs a problemas de optimización radica en la adecuada selección de una función de codificación. Es decir, en la propicia definición de una función que asigne a cada cromosoma, un individuo que representa una posible solución al problema [28, 67, 29]. La aplicación de AGs a problemas de optimización discreta es un buen ejemplo de ello. La codificación y la combinación de ella con adecuados operadores de recombinación deciden en buena parte el éxito de la aplicación [68, 69, 70]. Debido a que los individuos en un AG deben representan soluciones, entonces los individuos en el AG aplicado al problema de ruteo debe representar rutas posibles, además de incluir información sobre el tipo de avión y los pasajeros a trasladar a lo largo de la ruta. En nuestro caso nos hemos reducido a representar la ruta y el tipo de avión.

En la aplicación de los AGs a problemas de optimización discreta, el éxito ha dependido en muchos casos de una buena codificación de las posibles soluciones al problema. El concepto de bondad de la codificación está asociado a diferentes cosas. Entre ellas una importante es que el cruce de individuos produzca individuos válidos. La literatura de aplicación de AGs a problemas de optimización discreta abunda en operadores especiales y codificiaciones para secuencias [71, 72, 29]. En nuestro caso, hemos hecho uso la codificación utilizada por Ordoñez y Valenzuela [71, 72] en la cual una secuencia se codifica en un cromosoma binario.

Supongamos que el problema está definido sobre la red dirigida $G = (V, E)$. Por ello, existe una manera de enumerar lo vértices en V, con números enteros 0,2,..$n-1$, donde $n = |V|$, de tal manera que podemos identificar a V por el conjunto $\{0, 1, 2, 3, \cdots, n-1\}$, y adicionalmente que los lados del grafo pueden considerarse elementos del conjunto $\{(i, j)|i < j\}$. Un proceso de enumeración simple podría ser el siguiente. Como G tiene sólo un vértice con índice de entrada 0 este vértice se enumera con la etiqueta 0. Se remueve del grafo dicho vértice junto con los lados que salen de él. Ahora, supongamos etiquetados los vértices hasta el i-ésimo de ellos. y que han sido removidos del grafo G los vértices etiquetados junto con los lados que salen de ellos. Si el conjunto de vértices restantes no queda uno solo, el proceso de enumeración ha concluido. Si quedan vértices sin remover, escojamos uno que tenga índice de entrada 0. Si no hubieran tales vértices, el grafo resultante tendría ciclos, y por consiguiente también el grafo original G, contradiciendo el hecho que G es una red dirigida. Marquemos este vértice con la etiqueta $i + 1$, removamos el vértice del grafo reducido así como los lados que salen de él, repitamos el proceso hasta concluir con la totalidad de los vértices. Esta forma de etiquetar garantiza que si existe un lado dirigido en el grafo del vértice x al vértice y, dicho lado contribuye al índice de entrada del vértice y, haciendo que este vértice no sea seleccionado en el proceso de etiquetación hasta el vértice x sea etiquetado para ser removido del grafo. Lo cual hace que la etiqueta del vértice x sea menor que la

etiqueta del vértice y.

Mediante una etiquetación de los vértices con las características mencionadas, una ruta de longitud m que salga de la ciudad origen hacia la ciudad destino se puede describir por una secuencia ordenada

$$i_0 = 0, i_1, i_2, \ldots, i_m = (n-1) \tag{6.1}$$

Ahora, si suponemos que los arcos que salen de cada ciudad están enumerados, iniciando de la posición 0, entoces una ruta de longitud m como la anterior se puede describir por una sucesión de m enteros como

$$j_1, j_2, \ldots, j_m \tag{6.2}$$

donde la ruta que describe tal sucesión sería

$$i_0 = 0, i_1, \ldots, i_m, \tag{6.3}$$

donde i_1 es el vértice descendiente del vértice 0 utilizando el lado de la posición j_1, i_2 es el vértice descendiente del vértice i_1 utilizando el lado j_2 que sale de él, y así sucesivamente. Por ejemplo, supongamos que la red dirigida G es el grafo con vértices

$$V = \{0, 1, 2, 3, 4\}, \tag{6.4}$$

y con lados

$$E = \{(0,1), (0,2), (1,2), (1,3), (1,4), (2,3), (2,4), (3,4)\} \tag{6.5}$$

. Podemos ordenar los descendientes:

- Descendientes del nodo 0:
  - descendiente de la posición 0: nodo 1,
  - descendiente de la posición 1: nodo 2,
- descendientes del nodo 1:
  - descendiente de la posición 0: nodo 2,
  - descendiente de la posición 1: nodo 3,
  - descendiente de la posición 2: nodo 4,
- descedientes del nodo 2:
  - descendiente de la posición 0: nodo 3.
  - descendiente de la posición 1: nodo 4.
- descendientes del nodo 3:
  - descendiente de la posición 0: nodo 4

Mediante la ordenación anterior se tendría que la secuencia ordenada

- $0, 0, 0, 0$ representa la ruta $0 - 1 - 2 - 3 - 4$
- $0, 0, 1$ representa la ruta $0 - 1 - 2 - 4$
- $0, 1, 0$ representa la ruta $0 - 1 - 3 - 4$
- $0, 2$ representa la ruta $0 - 1 - 3$
- $1, 0, 0$ representa la ruta $0 - 1 - 3 - 4$
- $1, 1$ representa la ruta $0 - 2 - 4$

Así, toda ruta se puede expresar mediante una cadena de números enteros entre 0 y $n-1$ posiblemente repetidos, pero no toda sucesión del tipo de enteros entre 0 y $n-1$ puede interpretarse como una ruta; por ejemplo $2, 2$ no tendría significado pues no existe el sucesor de la posición 2 del nodo 0. Para evitar estas eventualidades y poder construir siempre rutas válidas podemos definir la siguiente interpretación para una sucesión de la forma:

$$j_1, j_2, \ldots, j_m \tag{6.6}$$

El nodo de la posición 0 es 0,$i_0 = 1$. Supongamos construida la ruta hasta el nodo i_k, si este nodo es $n-1$, la ruta está construida. En caso contrario el nodo de la posición $k+1$ es el siguiente. Se determina el sucesor de i_k con posición j_k. Si éste no existe, tomar el de la posición $j_k \bmod s_k$, donde s_k es el número de sucesores del nodo i_k. Designar a este suceso como i_{k+1}. Si N es la profundidad del grafo G, es decir la máxima lóngitud de un camino dirigido que sale de 0 y que llega al nodo $n-1$, y M es el mayor índice de salida de todos los vértices en G entonces una ruta puede ser descrita por una cadena de longitud N conteniendo números sólo entre 1 y M. Todas las sucesiones de este tipo también se pueden interpretar como rutas del nodo 0 al nodo $n-1$. Este tipo de sucesiones se pueden codificar en cromosomas binarios de longitud $\lceil \log_2 (M) \rceil N$. Si además añadimos un segmento de cromosoma adicional, suficiente en longitud para codificar el tipo de avión utilizado entonces la cadena binaria tendrá codificada tanto la ruta como el tipo de avión en una forma compacta.

6.3. Evaluando posibles soluciones

Una vez definida una codificación para las posibles soluciones, el siguiente paso es evaluar los individuos. Siendo un individuo una ruta y un tipo de avión, su evaluación será la máxima ganancia obtenida al utilizarlo para transportar pasajeros a lo largo de la ruta. En la formulación matemática al problema de ruteo dado en el capítulo 2 las variables $x_{i,j,t}$ ya tienen un valor específico de cero o uno. Recordemos sus valores. Tiene el valor uno si el segmento de la ciudad con etiqueta i a la ciudad j está presente en la ruta cubierta y además está cubierta por el avión de tipo t. Es cero en otro caso.

La restricción dada por la ecuación 2.15 se convierte en una identidad. La restricción definida por la ecuación 2.16 se convierte en

$$\sum_{k \in Q(i)} \sum_{m \in R(j)} y_{km} \leq C_t \ \forall (i,j). \tag{6.7}$$

Esta desigualdad se describe diciendo que la cantidad de pasajeros transportados en cada segmento de la ruta no excede la capacidad del avión. La restricción planteada en la ecuación 2.17 queda

$$y_{ij} \leq \sum_{t} \sum_{k \in P(j)} D_{ij} x_{ikt} \forall (i,j) \in M; \ \forall t. \tag{6.8}$$

El problema consiste en maximizar para un ruta fija R cubierta por el avión tipo t

$$\text{máx} \left(\sum_{(i,j) \in R} \rho_{ij} y_{ij} - \sum_{(i,j) \in R} \gamma_{ijt} \right) \tag{6.9}$$

Este problema es uno de programación lineal donde las soluciones y_{ij} buscadas deben de ser enteras. Debido a que todos los coeficientes de las variables de decisión y_{ij} de las restricciones son uno, la solución obtenida al problema donde las variables de decisión se piensan como reales, son también enteras. Por lo tanto, la evaluación de una combinación de una ruta R y un tipo de avión t, se puede calcular primeramente formulando el problema de programación lineal definido por las restricciones 6.7, 6.8, y las restricciones de no negatividad sujeto a maximizar 6.9, y posteriormente resolviendo tal problema como si su naturaleza fuera real.

En resumen, la evaluación de un cromosoma procede primeramente decodificándolo, de lo cual se obtiene la ruta y tipo de avión. Posteriormente se formula el problema de programación lineal asociado que es como se describe en el párrafo anterior. Finalmente se aplica el método simplex para resolverlo. El método simplex [73] fue implementado sobre matrices esparcidas como aparece en la literatura [74].

6.4. Problemas de muestra y experimentación

La experimentación realizada fue limitada. Dos problemas, conocidos como Net8 y Net9 que aparecen descritos en el apéndice 4 y 5 respectivamente, fueron generados aleatoriamente. Ante la ausencia del conocimiento de la solución multimodal a ambos problemas se procedió de la siguiente manera para obtener lo que asumiremos como la solución multimodal a cada uno de los problemas formulados. Haciendo uso de los mismos algoritmos a comparar, CD, SH y memGA se procedió a determinar lo *mejor* que ellos pueden encontrar con *abundantes* recursos. Para ello se procedió de la siguiente manera. Cada una de las estrategias se ejecutó con una población aleatoria de 200 individuos. Dicha población se hizo evolucionar hasta que el criterio de paro utilizado en el capítulo anterior con parámetro $s_o = 0.001$ se activó. La población solución ofrecida por el algoritmo se guardó. Se hicieron 50 simulaciones independientes y en

todos los casos se conservó la solución del algoritmo. Con todas ellas se formó un gran conjunto de soluciones obtenidas y se removieron las soluciones repetidas. Se juntaron los resultados de las tres estrategias y con ellas nuevamente se formó un gran conjunto del cual se removieron las soluciones repetidas; designemos a este conjunto como S. Si

$$f_{max} = \text{Max}_{r \in S}\text{Evaluacion}(r), \tag{6.10}$$

entonces

$$R_s = \{r \in S | \text{Evaluacion}(r) \geq 0.9\, f_{max}\} \tag{6.11}$$

Estas soluciones aparecen incluidas también en el apéndice 2. La idea principal en la definición de R_s es contener aquellas rutas y selección de tipo de avión que proporciona una ganancia que no está por abajo del 90 % de mejor la evaluación obtenida.

6.5. Diseño de experimentos y resultados

El propósito de los experimentos aquí planteados es comparar el desempeño de los algoritmos SH, CD, y memGA ante los problemas Net8 y Net9. Específicamente, lo que deseamos evaluar es cuál estrategia proporciona más óptimos locales haciendo uso de un cierto número fijo de evaluaciones y partiendo de una población de tamaño fijo e igual para cada estrategia. Los experimentos que con tal fin se desarrollaron fueron los siguientes. Se fijó un tamaño de población de 100 individuos para los tres algoritmos. Se utilizó el operador de cruce en un punto con probabilidad de cruce de 1.0. El operador de mutación se utilizó con probabilidad de mutación de 0.1. El valor de σ_{share} utilizado fue de 2.0, el cual coincidió con el valor de radio de nicho r_o utilizado. La distancia utilizada fue una especialmente definida para tratar rutas con tipos de aviones diferentes como rutas *suficientemente* alejadas. Aunado a esto la distancia entre dos rutas recorridas con el mismo tipo de avión se define de la siguiente manera. Primeramente se determina la longitud más pequeña entre las dos rutas que están siendo consideradas. A esta cantidad se le resta el número de ciudades que tienen en común en forma consecutiva partiendo de la ciudad origen. La población proporcionada por la estrategia fue revisada y se contabilizaron cuántas soluciones del conjunto meta R_s no estaban presentes. Adicionalmente se determinó la distancia promedio a los óptimos deseados dividida entre r_o. Se realizaron 50 simulaciones independientes para las estrategias CD, SH, y memGA (memGA$_{2,2}$). Los resultados obtenidos se muestran en las tablas 6.1 y 6.2. De las tablas podemos observar lo siguiente. Que para los problemas planteados SH manifiesta su capacidad para retener puntos en las vecindades de los óptimos locales (8 para Net8 de acuerdo a la tabla D.5, 21 para Net9 de acuerdo a la tabla E.8). Esto lo observamos debido a que en ambos problemas el contador de vecindades vacías decrece conforme avanza el número de evaluaciones. Esto no ocurre en CD; existe un intervalo en la escala de tiempo dado por el número de evaluaciones donde baja el indicador de vecindades vacías, significando con ello la identificación de óptimos locales. Pero conforme el *tiempo avanza* el contador aumenta, es decir los individuos en la población emigran, y al juzgar por la forma como crece la evaluación promedio de la población, emigran hacia los óptimos mejor evaluados dejando vecindades vacías.

Desempeño comparado en Net8						
	SH		CD		memGA	
e	$\bar{f}(\sigma_{\bar{f}})$	$\bar{v}(\sigma_{\bar{v}})$	$\bar{f}(\sigma_{\bar{f}})$	$\bar{v}(\sigma_{\bar{v}})$	$\bar{f}(\sigma_{\bar{f}})$	$\bar{v}(\sigma_{\bar{v}})$
500	356.1(23.46)	5.42(1.326)	588.7(15.90)	4.78(1.234)	475.2(17.73)	2.82(1.438)
1000	383.4(22.55)	4.06(1.609)	686.1(26.50)	2.82(1.945)	495.8(14.81)	1.78(1.112)
1500	383.3(20.31)	3.42(1.655)	744.0(26.93)	2.02(1.436)	510.6(13.39)	0.88(0.773)
2000	388.9(16.82)	2.62(1.354)	779.3(23.00)	3.14(1.539)	510.8(12.20)	0.70(0.863)
3000	388.5(16.46)	1.92(1.209)	788.6(29.43)	4.74(0.899)	518.5(11.80)	0.24(0.431)
4000	389.2(18.95)	1.68(1.186)	803.8(20.58)	5.00(0.350)	516.7(14.35)	0.16(0.370)
5000	384.3(19.78)	1.38(1.028)	809.4(20.55)	5.04(0.198)	513.6(14.92)	0.04(0.198)

Tabla 6.1: Promedio sobre 50 simulaciones de la evaluación promedio ($\bar{f}$), su desviación estándar ($\sigma_{\bar{f}}$), el conteo promedio de vecindades vacías ($\bar{v}$), y su desviación estándar ($\sigma_{\bar{v}}$) para SH, CD, y memGA haciendo uso de un número de evaluaciones e sobre Net8.

memGA se desempeña en esos problemas de forma muy superior a sus oponentes: el contador de vecindades decrece, indicando que retiene puntos en las vecindades de los óptimos locales, y que consumiendo evaluaciones posiciona rápidamente elementos en nichos antes vacíos.

6.6. Resumen y conclusiones

En este capítulo hemos presentado la aplicación de memGA a un problema de optimización combinatoria relevante y complejo; hemos definido una codificación y una evaluación posible. Con esta implementación hemos comparado memGA con el método de las funciones compartidas y con *crowding* determinístico usando un marco experimental muy limitado. A pesar de ello, los resultados experimentales son bastante atractivos. Mucho falta por experimentar; desde otro tipo de codificaciones, otro tipo de definición de una distancia entre rutas, hasta la búsqueda de nuevos y quizá más problemas reales donde la estrategia propuesta tenga una confrontación quizá más retadora con éstos u otras estrategias. De momento parece haber buenas expectativas.

Desempeño comparado en Net9						
	SH		CD		memGA	
e	$\bar{f}(\sigma_{\bar{f}})$	$\bar{v}(\sigma_{\bar{v}})$	$\bar{f}(\sigma_{\bar{f}})$	$\bar{v}(\sigma_{\bar{v}})$	$\bar{f}(\sigma_{\bar{f}})$	$\bar{v}(\sigma_{\bar{v}})$
500	461.4(29.24)	19.6(1.42)	715.1(25.35)	18.6(1.84)	616.7(21.14)	15.1(2.07)
1000	506.5(38.36)	17.9(2.35)	876.6(22.56)	15.7(2.47)	658.7(13.98)	11.5(2.52)
1500	526.2(30.07)	15.9(2.57)	948.1(24.29)	13.6(2.65)	666.2(16.89)	9.8(2.93)
2000	545.2(27.68)	14.6(2.79)	981.8(25.28)	13.5(3.20)	687.7(27.35)	7.2(2.32)
3000	564.0(28.44)	12.2(2.70)	1001.7(24.14)	16.7(1.62)	727.1(27.02)	5.1(2.26)
4000	570.2(25.93)	11.4(2.22)	1002.6(26.51)	17.6(0.77)	760.9(23.23)	3.2(1.98)
5000	567.6(29.44)	11.2(2.09)	1004.9(23.86)	17.9(0.46)	786.1(21.36)	2.2(1.88)

Tabla 6.2: Promedio sobre 50 simulaciones de la evaluación promedio ($\bar{f}$), su desviación estándar ($\sigma_{\bar{f}}$), el conteo promedio de vecindades vacías ($\bar{v}$), y su desviación estándar ($\sigma_{\bar{v}}$) para SH, CD, y memGA haciendo uso de un número de evaluaciones e sobre Net9.

Capítulo 7

Conclusiones

En los capítulos precedentes hemos desarrollado una estrategia basada en algoritmos genéticos para resolver problemas de optimización multimodal. En el presente capítulo se presenta una visión general de la investigación y de las metas alcanzadas así como de algunas fracasos que se presentaron. También se describen algunas posibles extensiones al trabajo de investigación.

7.1. Visión global de la investigación

El presente trabajo ha tenido como propósito desarrollar una estrategia basada en la recombinación de características que resuelva satisfactoriamente problemas multimodales, y que aún ante el cambio de naturaleza o dificultad de los mismos, el algoritmo logre un desempeño aceptable. Los problemas que presentaban las estrategias que consideramos como previas se pueden reducir a la falta de retención de elementos en las vecindades de los óptimos a lo largo de las generaciones, y/o la falta de convergencia a los óptimos locales de la función problema. Ello se probó experimentalmente en el capítulo 2. El algoritmo que pensamos requería desarrollarse debería tener como meta alcanzar la idea que se formula en forma simple como *si se deja el algoritmo tiempo adicional, se debe obtener un mejor resultado.* Para ello, el algoritmo debería efectivamente poseer la capacidad de retención de información ya encontrada y que se considera *valiosa*, y a la vez poseer la habilidad de no viciar la búsqueda para generar información que se considera como más precisa o aceptable. Esto creemos se ha alcanzado mediante el uso de una población memoria, de un indicador del concepto de valioso, el cual es el indicador en el cual se basa el reemplazo, el uso de un mecanismo de selección que indica los elementos que aportan características para una exploración, y del uso de un mecanismo de recombinación de características. En este sentido el algoritmo genético que se aplica sobre la población exploradora hace la veces de un *metaoperador* de recombinación para producir descendientes. El algoritmo propuesto que bautizamos como memGA y cuyas partes hemos descrito se plantea al final del capítulo 2.

El capítulo 3 se dedicó a analizar y ajustar el algoritmo propuesto. Un punto que consideramos a favor ha sido el establecer parámetros de ajuste a la intensidad de los

indicadores, y poder hacer un análisis aunque sea separado de su efecto en la diversidad. Un fracaso de alguna forma esperado era suponer que el ajuste de los parámetros para los indicadores dependía de la *naturaleza* del problema; en los experimentos desarrollados en ese mismo capítulo enfrentamos esta realidad. El ajuste de parámetros propuesto se fijó en un *centroide* más cargado hacia donde los problemas difíciles apuntan. En el capítulo 4 vimos experimentalmente que la decisión de elegir esos parámetros, aún cuando no óptimos al problema, tenía un desempeño satisfactoriamente superior en casi la mayoría de los problemas al desempeño de los algoritmos contra los cuales se comparaba.

Uno de los principales elementos en contra de la estrategia desarrollada es el costo computacional involucrado en el cálculo de los indicadores de reemplazo y selección para la población. Este costo depende directamente y en forma cuadrática del tamaño de las poblaciones; poblaciones moderadamente altas involucran tiempos de cálculo relativamente grandes. Para probar la bondad de la estrategia propuesta debíamos buscar una aplicación novedosa donde este costo computacional extra estuviera por abajo, de ser posible muy por debajo, del ahorro de evaluaciones de la función problema. Adicionalmente, el problema debería tener sentido cuando se formulara multimodalmente. El problema de ruteo de vuelos con múltiples paradas desarrollado en el capítulo 6 cumplió con creces estos dos requisitos. Aunque los experimentos fueron limitados, ellos resultaron realmente alhagadores para el algoritmo propuesto cuando se contrastaba contra sus competidores.

7.2. Trabajo futuro

A lo largo de la presente investigación se tomaron varias decisiones que simplificaron su desarrollo y que es conveniente ahora reconsiderar, así como plantear alternativas que pudieran mejorar el producto. Haremos una lista de ellas y de cómo pueden afrontarse desde otra perspectiva.

El número de óptimos locales en un problema de optimización multimodal es un parámetro intrínseco al propio problema. De esta forma, el conjunto que la presente estrategia, o cualquier otra que resuelva el problema multimodal, entregue pretendiendo ser solución al problema debe tener un tamaño que dependa del problema en si. Es decir, que idóneamente el algoritmo debe modificar el tamaño del conjunto solución, y por consiguiente el tamaño de la población memoria, de acuerdo a parámetros internos de manera que le permitan cubrir adecuadamente el espacio de búsqueda y dar una respuesta adecuada al conjunto de óptimos locales. Una posible implementación o modificación, para contemplar esta característica podría estar basada en el indicador modificado del contador de nichos. Por ejemplo, cuando individuos *altamente* evaluados y con un contador de modificado de nichos reducido estén perdiendo competencias en el reemplazo puede ser el momento de decidir que el tamaño de población es reducido y por consiguiente debe aumentarse. Una alternativa podría ser sencillamente agrandar la población para que incluya a ese individuo momentáneamente reconocido como perdedor. Dos elementos intervienen en esta decisión. El aumento en el costo

computacional en el cálculo de los indicadores de reemplazo y selección, y la duda sobre el valor de la distancia de compartición. Este último puede influir seriamente en este sentido, y lo que resulta conveniente es hacer un análisis de sensibilidad de la estrategia respecto a cambios en el radio de compartición. El primero refrente a costos computacionales puede eludirse si en lugar del cálculo exacto de los indicadores de reemplazo y selección se cambian por estimación de naturaleza incremental como fue calculada por Valenzuela y Uresti [75] para el caso de problemas multiobjetivo. Alternativamente a crecer está el disminuir. Si en la población memoria existen individuos con un contador de nichos modificado alto podría deberse a dos causas: que efectivamente el individuo posea información redundante en la memoria, en cuyo caso podría ser conveniente eliminarlo reduciendo memoria y por consiguiente trabajo computacional, y otra de las causas podría ser un valor inadecuadamente grande de la distancia de compartición: esto podría causar que eliminar tal individuo de la población sea negativo. Nuevamente surge el tema del análisis de la sensibilidad de la respuesta ante cambios en el radio de compartición.

La velocidad de muestreo del espacio de búsqueda está íntimamente relacionada con el tamaño de la población exploradora. Pocos reemplazos en la población memoria generación a generación podrían ser indicadores de que la solución está prácticamente construida o podría indicar que la producción de individuos adecuados es baja. Una estrategia agresiva podría ser que, ante este tipo de situación, se aumente el tamaño de la población. Ello podría ayudar a la identificación de mejores soluciones. Ello lo comprobamos experimentalmente con las funciones engañosas: había un tamaño adecuado de la población exploradora que minimizaba población y número de poblaciones totales. También es cierto que toda exploración está relacionada con una pérdida de esfuerzo computacional. A la par de pensar en incrementar la población exploradora, podemos pensar en reducirla. Reducirla significa reducir la complejidad del proceso de administración. Todo es un balance y lucha por dos elementos que compiten por recursos. La estrategia de CD fue escogida por su bajísima complejidad computacional, a pesar de tener un comportamiento indeseable en algunos problemas, los cuales parecían eliminarse con el proceso de administración. Sin embargo, es interesante el pensar que el mismo proceso de administración pueda seleccionar, dentro de un cierto conjunto de estrategias, una adeuada para un problema. O más aún, que mediante un proceso de experimentación en la misma solución de un problema, decida escoger un AG *hijo* diferente en forma estocástica dependiendo de su comportamiento a lo largo del mismo problema.

El cambio del cálculo de indicadores por estimaciones de ellos es un tema importante. Para problemas con evaluación relativamente sencilla y poco costosa en tiempo computacional, memGA se desempeña más lento que *sharing* o *crowding* determinístico, a pesar de que el número de evaluaciones sea menor. Una estrategia de estimación incremental de los indicadores como la empleada para problemas multiobjetivo desarrollada por Valenzuela y Uresti [75] podría ser utilizada. En la aplicación de la estrategia desarrollada al problema de ruteo de vuelos con múltiples paradas fue una aplicación con muchos elementos que reclaman una revisión. Hay varias codificaciones posibles para secuencias en la literatura y no se establece comparación con la que aquí se presenta.

Este es un punto pendiente. El uso de diferentes operadores de cruce ya desarrollados para secuencias y/o el desarrollo de operadores de cruce específicos puede beneficiar la búsqueda. Este tema está pendiente. La experimentación que se realizó estuvo limitada; la falta de problemas prácticos que se acusa define un área de oportunidad. Así, el valor práctico del ofrecimiento de una solución multimodal a los tomadores de decisiones podría ser un motivo para aumentar el entusiasmo en la continuación de este tipo de enfoques.

7.3. Conclusiones

La presente investigación muestra que es posible construir una estrategia para resolver problemas multimodales que presente un efecto incremental en la calidad de sus soluciones, de manera que a mayor tiempo de cómputo hay mayores expectativas de obtener mejores soluciones. En el desarrollo de la presente estrategia la posibilidad de regular intensidades de búsqueda fue planteada y a partir de esto se pudo formular un ajuste adecuado de parámetros para lograr cubrir con relativa amplitud problemas de diferente naturaleza con ventajas, en lo general, sobre las estrategias de *crowding* determinístico y *sharing*. La propuesta de algoritmo ofrece buenas expectativas de consolidarse sobre otras por lo adecuado de sus resultados experimentales. Los buenos resultados comparativos de experimentos aunque limitados, de la aplicación memGA al problema de ruteo de vuelos con múltiples son prometedores para continuar con este trabajo desde la perspectiva multimodal con problemas de optimización discreta.

Apéndice A

Variantes de memGA sobre funciones similares a M_2

En este apéndice aparecen reportados los resultados de aplicar memGA con diferentes valores para los parámetros α y γ a variaciones del problema M_2 2.8. Estos problemas aparecen definidos en el capítulo 3. En cada caso los resultados corresponden al promedio sobre 50 simulaciones independientes. El tamaño de la población exploradora fue de 4 individuos, mientras que el tamaño inicial de la población memoria fue también de 4. El criterio de paro utilizado en memGA fue el basado en el promedio sobre 4 generaciones consecutivas del promedio de la evaluación de la población, con parámetro $s_o = 0.001$ tal y como es descrito en el capítulo 3. El criterio para aceptar un conjunto de individuos como solución al problema multimodal utilizó el radio de nicho, r_o, como el valor de σ_{share} en cada problema. En el problema M_{2a}, $\sigma_{share} = 0.125$; para M_{2b}, $\sigma_{share} = 0.083$; para M_{2c} a M_{2i}, $\sigma_{share} = 0.1$. Si la población evolucionada no se acepta como solución, se reinicia la aplicación de memGA sobre una población inicial que es del doble en tamaño al de la población recién utilizada: ésta se forma con la población inicial de la simulación previa complementada con individuos generados en forma aleatoria. La probabilidad de mutación se mantuvo en 0.0, y se utilizó siempre el operador de cruce en un punto, es decir la probabilidad de mutación se fijó en 1.0.

A.1. Recursos requeridos por cada variante de memGA

La tabla A.1 contiene, para una selección de los parámetros α y γ definidos vertical y horizontalmente respectivamente, el promedio del tamaño de población utilizado por memGA; entre paréntesis aparece la desviación estándar. La tabla A.2 contiene los promedios del número de evaluaciones utilizadas para el AG. La tabla A.3 contiene el promedio de las evaluaciones totales para una combinación de parámetros α y γ. Recordemos que el término *evaluaciones totales* hace referencia a la suma entre el número de evaluaciones utilizadas por el AG y el número de evaluaciones utilizadas

	Tamaño de población al resolver M_{2a}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	16.8(6.8)	21.8(12.3)	21.4(10.7)	21.8(10.3)	23.4(12.1)
2.0	16.7(8.9)	15.4(7.2)	17.6(10.3)	15.6(7.6)	19.8(10.2)
4.0	13.7(9.5)	14.0(8.2)	12.7(7.0)	13.1(6.3)	13.5(6.8)
8.0	12.0(7.1)	11.3(5.6)	12.4(6.4)	11.0(5.2)	12.0(5.9)
16.0	11.0(5.2)	12.5(7.3)	10.7(5.9)	12.5(6.3)	11.6(4.4)

Tabla A.1: Promedio sobre 50 simulaciones independientes del tamaño de población utilizado por memGA para diferentes valores de los parámetros α y γ al resolver M_{2a}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Conteo de evaluaciones al resolver M_{2a}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	347.5(133.2)	394.1(176.4)	371.5(166.2)	351.5(130.2)	377.5(166.6)
2.0	341.1(126.5)	329.1(138.3)	326.8(129.9)	310.5(110.6)	355.0(131.6)
4.0	309.3(143.0)	324.9(135.7)	281.5(101.1)	296.5(89.4)	273.1(98.5)
8.0	292.2(119.2)	288.7(119.7)	294.5(118.5)	257.5(97.8)	280.8(118.7)
16.0	273.2(100.7)	305.2(137.2)	282.0(111.6)	311.1(125.3)	248.8(113.8)

Tabla A.2: Promedio sobre 50 simulaciones independientes del número del evaluaciones utilizadas por memGA para diferentes valores de los parámetros α y γ al M_{2a}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

por el buscador local PHC aplicado al conjunto que el algoritmo proporciona como solución. El algoritmo PHC es referido en el capítulo 4 y tiene una descripción, para el caso donde el fenotipo es un vector real, en la figura 4.10. Los formatos de las tablas A.1, A.2 y A.3 son similares. Asimismo las correspondientes a los problemas M_{2b}, M_{2c}, M_{2d}, M_{2e}, M_{2f}, M_{2g}, M_{2h}, y M_{2i}.

A.1.1. Listas de modelos no perdedores

Usando los promedios, desviaciones y tamaños de muestra de las tablas de tamaño de población, número de evaluaciones de memGA, y número de evaluaciones totales, en forma independiente, para comparar estadísticamente con un índice de confianza del 95 % cuál o cuáles variantes de memGA hacen uso de una menor cantidad de recursos promedio obtenemos los siguientes hechos.

1. Para M_{2a}:

   - En tamaño de población:

	Conteo de evaluaciones totales al resolver M_{2a}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	627.0(141.8)	664.8(186.5)	627.1(175.3)	609.5(133.5)	634.1(171.3)
2.0	612.4(146.3)	595.9(146.9)	592.4(150.3)	564.3(113.9)	603.6(132.1)
4.0	589.4(151.6)	590.0(138.6)	534.1(105.7)	559.0(91.1)	539.9(104.7)
8.0	567.8(130.0)	557.6(123.8)	557.4(117.1)	519.4(96.0)	550.0(124.6)
16.0	554.0(105.0)	585.9(140.9)	547.9(104.5)	591.9(123.9)	538.3(118.0)

Tabla A.3: Promedio sobre 50 simulaciones de la suma del número de evaluaciones utilizadas por memGA con el número de evaluaciones utilizadas por PHC aplicado a la solución provista por memGA en cada simulación, para diferentes valores de los parámetros α y γ al resolver M_{2a}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Tamaño de población al resolver M_{2b}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	38.5(18.3)	37.5(16.4)	38.5(15.6)	39.7(20.4)	40.6(19.6)
2.0	30.8(18.7)	30.1(14.7)	30.3(14.5)	28.1(11.9)	28.8(15.0)
4.0	26.5(12.5)	26.0(11.2)	21.8(8.2)	24.1(10.0)	24.1(13.2)
8.0	21.4(10.3)	20.8(12.2)	22.3(12.8)	21.2(10.5)	24.3(11.8)
16.0	20.6(9.9)	19.7(10.7)	18.3(8.1)	19.5(7.0)	18.5(10.1)

Tabla A.4: Promedio sobre 50 simulaciones independientes del tamaño de población utilizado por memGA para diferentes valores de los parámetros α y γ al resolver M_{2b}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Conteo de evaluaciones al resolver M_{2b}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	606.8(233.4)	569.2(201.8)	600.0(234.1)	584.9(223.3)	583.8(201.5)
2.0	581.0(269.9)	521.8(215.4)	523.5(217.6)	448.4(180.1)	459.2(208.5)
4.0	460.0(215.3)	510.3(239.9)	418.4(162.4)	417.5(134.3)	389.8(178.6)
8.0	433.4(181.9)	409.3(178.3)	416.0(182.6)	374.0(148.1)	407.9(174.1)
16.0	408.8(167.3)	421.9(188.1)	391.8(155.9)	379.0(151.6)	340.7(167.5)

Tabla A.5: Promedio sobre 50 simulaciones independientes del número del evaluaciones utilizadas por memGA para diferentes valores de los parámetros α y γ al resolver M_{2b}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Conteo de evaluaciones totales al resolver M_{2b}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	1089.8(264.1)	1013.3(219.5)	1017.5(231.3)	982.5(240.4)	988.3(218.3)
2.0	1041.9(277.0)	979.5(220.1)	959.4(229.6)	862.9(186.0)	879.3(215.2)
4.0	943.2(229.6)	948.0(233.4)	824.5(162.4)	840.1(145.5)	823.5(183.4)
8.0	913.1(189.7)	860.0(192.1)	866.7(191.0)	803.8(157.2)	848.8(193.9)
16.0	921.9(158.9)	882.8(198.7)	852.9(173.2)	830.4(156.6)	790.4(171.0)

Tabla A.6: Promedio sobre 50 simulaciones de la suma de las evaluaciones utilizadas por memGA con el número de evaluaciones utilizadas por PHC aplicado a la solución provista por memGA en cada simulación, para diferentes valores de los parámetros α y γ al resolver M_{2b}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Tamaño de población al resolver M_{2c}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	53.5(18.7)	48.6(16.1)	47.4(16.1)	52.3(19.0)	52.0(16.1)
2.0	27.5(12.1)	28.9(13.8)	24.6(8.0)	26.8(10.8)	27.7(13.1)
4.0	22.8(10.8)	20.5(8.0)	22.1(13.2)	20.6(8.7)	22.0(9.4)
8.0	18.8(10.6)	20.1(12.3)	17.2(8.0)	19.4(10.5)	18.8(10.6)
16.0	18.5(8.5)	16.9(7.7)	16.3(10.1)	15.4(5.9)	18.5(10.1)

Tabla A.7: Promedio sobre 50 simulaciones independientes del tamaño de población utilizado por memGA para diferentes valores de los parámetros α y γ al resolver M_{2c}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Conteo de evaluaciones al resolver M_{2c}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	721.4(240.9)	699.2(242.6)	661.1(208.5)	702.5(257.7)	720.7(231.1)
2.0	474.7(186.9)	421.3(154.8)	417.8(155.8)	380.6(147.8)	422.6(173.2)
4.0	349.1(150.5)	329.3(139.2)	352.1(130.1)	355.8(140.1)	347.9(132.5)
8.0	356.4(141.9)	372.0(163.4)	320.4(147.0)	329.1(140.5)	316.5(146.0)
16.0	322.7(118.6)	303.9(136.3)	307.7(166.8)	322.1(134.8)	311.0(149.6)

Tabla A.8: Promedio sobre 50 simulaciones independientes del número del evaluaciones utilizadas por memGA para diferentes valores de los parámetros α y γ al resolver M_{2c}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Conteo de evaluaciones totales al resolver M_{2c}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	1128.2(251.1)	1068.4(230.2)	1000.0(222.7)	1023.7(269.3)	1040.9(227.3)
2.0	879.1(206.5)	813.7(179.2)	765.3(160.2)	715.8(150.4)	767.1(180.9)
4.0	762.2(157.0)	724.2(155.4)	718.2(143.1)	699.3(147.8)	709.1(146.8)
8.0	776.0(165.9)	765.7(201.6)	712.3(153.2)	705.5(161.0)	683.7(154.9)
16.0	752.7(129.0)	726.3(167.7)	716.9(181.5)	705.8(141.7)	702.5(150.6)

Tabla A.9: Promedio sobre 50 simulaciones de la suma de la evaluaciones utilizadas por memGA con el número de evaluaciones utilizadas por el proceso PHC aplicado a la solución provista por memGA en cada simulación, para diferentes valores de los parámetros α y γ al resolver M_{2c}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Tamaño de población al resolver M_{2d}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	28.6(13.9)	30.3(14.5)	32.1(15.6)	31.7(13.6)	28.9(12.7)
2.0	25.2(9.7)	25.7(13.7)	27.5(13.3)	22.8(8.0)	22.0(8.5)
4.0	23.8(13.5)	22.5(10.4)	21.5(11.3)	23.4(11.5)	19.5(8.4)
8.0	18.6(8.3)	16.9(7.2)	16.3(7.6)	18.8(8.2)	18.5(8.0)
16.0	17.5(8.7)	18.8(9.5)	19.1(10.4)	18.6(10.0)	16.9(6.7)

Tabla A.10: Promedio sobre 50 simulaciones independientes del tamaño de población utilizado por memGA para diferentes valores de los parámetros α y γ al resolver M_{2d}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Conteo de evaluaciones al resolver M_{2d}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	494.8(220.0)	546.1(199.9)	540.8(245.7)	504.0(192.1)	446.1(171.4)
2.0	478.0(197.1)	494.5(186.1)	474.3(178.6)	396.5(149.6)	419.3(170.1)
4.0	465.4(173.3)	487.0(179.5)	420.6(182.2)	437.7(208.2)	386.0(165.0)
8.0	415.2(213.7)	413.2(225.0)	372.3(149.1)	398.6(191.2)	360.7(159.2)
16.0	403.0(234.8)	394.4(222.0)	422.9(189.9)	367.1(205.4)	342.5(129.7)

Tabla A.11: Promedio sobre 50 simulaciones independientes del número del evaluaciones utilizadas por memGA para diferentes valores de los parámetros α y γ al resolver M_{2d}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Conteo de evaluaciones totales al resolver M_{2d}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	876.1(230.0)	905.0(211.1)	882.4(234.7)	830.5(191.5)	766.9(169.2)
2.0	861.8(211.8)	846.8(181.8)	807.7(183.1)	714.6(149.0)	748.2(168.4)
4.0	841.8(216.6)	845.9(189.1)	760.8(200.5)	782.3(199.9)	714.1(162.1)
8.0	807.9(223.9)	783.0(217.2)	715.3(138.8)	734.5(196.4)	707.4(151.0)
16.0	807.8(238.9)	777.5(207.7)	780.4(185.2)	740.0(196.2)	699.7(131.5)

Tabla A.12: Promedio sobre 50 simulaciones de la suma del número de evaluaciones utilizadas por memGA con el número de evaluaciones utilizadas por el proceso PHC aplicado a la solución provista por memGA en cada simulación, para diferentes valores de los parámetros α y γ al resolver M_{2d}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Tamaño de población al resolver M_{2e}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	50.1(22.6)	54.1(21.5)	54.8(24.6)	52.6(22.3)	61.8(32.8)
2.0	46.8(31.3)	42.1(21.3)	42.8(20.7)	42.1(20.5)	39.2(17.7)
4.0	38.5(16.5)	34.1(15.2)	34.0(18.1)	31.4(15.8)	34.1(14.2)
8.0	26.0(11.2)	22.5(10.4)	26.5(13.7)	27.2(15.0)	24.9(11.5)
16.0	21.5(11.6)	20.0(7.5)	23.8(13.2)	20.9(7.9)	19.2(7.8)

Tabla A.13: Promedio sobre 50 simulaciones independientes del tamaño de población utilizado por memGA para diferentes valores de los parámetros α y γ al resolver M_{2e}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Conteo de evaluaciones al resolver M_{2e}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	695.9(296.6)	705.7(281.6)	729.8(273.1)	671.7(270.1)	756.4(341.1)
2.0	552.3(264.9)	641.3(304.6)	586.0(250.9)	535.3(230.2)	517.1(208.9)
4.0	538.5(207.7)	465.8(235.9)	514.4(284.4)	457.1(208.8)	492.8(171.1)
8.0	421.1(183.4)	372.7(167.1)	427.1(188.1)	425.9(206.1)	407.8(163.3)
16.0	356.9(181.7)	348.1(143.7)	369.5(198.9)	365.8(136.7)	344.3(139.9)

Tabla A.14: Promedio sobre 50 simulaciones independientes del número del evaluaciones utilizadas por memGA para diferentes valores de los parámetros α y γ al resolver M_{2e}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Conteo de evaluaciones totales al resolver M_{2e}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	1085.9(307.1)	1092.7(282.7)	1071.9(297.3)	993.3(268.4)	1073.1(339.2)
2.0	957.1(290.1)	1016.8(308.1)	929.8(262.0)	874.3(227.9)	843.2(207.3)
4.0	941.6(228.8)	853.6(244.0)	863.0(280.6)	783.1(210.8)	829.7(173.2)
8.0	828.2(187.1)	750.1(172.1)	797.5(197.6)	776.4(200.4)	760.2(146.8)
16.0	784.1(197.2)	737.6(142.1)	754.2(194.9)	753.6(132.3)	714.0(126.6)

Tabla A.15: Promedio sobre 50 simulaciones de la suma del número de evaluaciones utilizadas por memGA y el número de evaluaciones utilizadas por PHC aplicado a la solución provista por memGA en cada simulación, para diferentes valores de los parámetros α y γ al resolver M_{2e}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Tamaño de población al resolver M_{2f}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	19.7(6.8)	19.7(6.8)	20.0(8.9)	20.3(7.2)	20.3(7.2)
2.0	16.9(8.2)	16.3(8.1)	15.5(7.0)	16.5(9.7)	13.8(5.3)
4.0	13.1(5.5)	13.8(6.5)	15.2(7.6)	14.6(7.4)	13.5(5.4)
8.0	15.2(7.1)	14.0(7.1)	14.1(8.9)	14.9(6.7)	16.5(8.4)
16.0	14.0(8.1)	14.6(9.2)	15.8(8.3)	13.4(6.7)	13.2(5.5)

Tabla A.16: Promedio sobre 50 simulaciones independientes del tamaño de población utilizado por memGA para diferentes valores de los parámetros α y γ al resolver M_{2f}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Conteo de evaluaciones al resolver M_{2f}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	371.2(142.6)	384.9(138.8)	353.1(127.9)	351.5(121.4)	341.1(107.7)
2.0	363.9(127.6)	333.9(163.3)	317.6(125.1)	299.5(116.3)	283.5(122.0)
4.0	360.4(202.7)	364.5(181.0)	371.6(178.7)	305.1(165.7)	291.4(141.6)
8.0	349.2(133.3)	364.1(178.6)	337.3(155.3)	333.5(162.6)	305.7(136.9)
16.0	359.5(140.8)	328.1(111.5)	363.5(168.5)	315.2(140.9)	313.7(152.3)

Tabla A.17: Promedio sobre 50 simulaciones independientes del número del evaluaciones utilizadas por memGA para diferentes valores de los parámetros α y γ al resolver M_{2f}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

	Conteo de evaluaciones totales al resolver M_{2f}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	783.8(154.0)	789.0(162.4)	741.8(147.6)	734.1(147.4)	718.9(118.3)
2.0	803.8(173.4)	741.6(185.1)	730.9(159.6)	702.3(161.6)	675.5(134.1)
4.0	791.2(227.1)	789.5(198.4)	797.8(193.6)	710.4(177.5)	710.0(141.5)
8.0	822.1(183.5)	802.2(211.1)	750.6(169.3)	760.5(188.7)	739.5(177.1)
16.0	800.5(185.0)	795.6(150.8)	796.7(175.1)	722.1(157.1)	751.6(159.6)

Tabla A.18: Promedio sobre 50 simulaciones de la suma del número de evaluaciones utilizadas por memGA y el número de evaluaciones utilizadas por PHC aplicado a la solución provista por memGA en cada simulación, para diferentes valores de los parámetros α y γ al resolver M_{2f}. En cada entrada de la tabla se muestra el promedio acompañado de la desviación estándar entre paréntesis.

$\text{memGA}_{8,1}$, $\text{memGA}_{16,1}$, $\text{memGA}_{8,2}$, $\text{memGA}_{16,2}$, $\text{memGA}_{4,4}$, $\text{memGA}_{8,4}$, $\text{memGA}_{16,4}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$

- En evaluaciones de memGA:
  $\text{memGA}_{16,1}$, $\text{memGA}_{4,4}$, $\text{memGA}_{16,4}$, $\text{memGA}_{8,8}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$
- En evaluaciones totales:
  $\text{memGA}_{4,4}$, $\text{memGA}_{16,4}$, $\text{memGA}_{8,8}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$

2. Para M_{2b}:

   - En tamaño de población:
     $\text{memGA}_{16,1}$, $\text{memGA}_{8,2}$, $\text{memGA}_{16,2}$, $\text{memGA}_{16,4}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{16,16}$
   - En evaluaciones de memGA:
     $\text{memGA}_{16,4}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{4,16}$, $\text{memGA}_{16,16}$
   - En evaluaciones totales:
     $\text{memGA}_{4,4}$, $\text{memGA}_{4,8}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$

3. Para M_{2c}:

   - En tamaño de población:
     $\text{memGA}_{16,2}$, $\text{memGA}_{8,4}$, $\text{memGA}_{16,4}$, $\text{memGA}_{16,8}$
   - En evaluaciones de memGA:
     $\text{memGA}_{4,1}$, $\text{memGA}_{16,1}$, $\text{memGA}_{4,2}$, $\text{memGA}_{16,2}$, $\text{memGA}_{8,4}$, $\text{memGA}_{16,4}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$

- En evaluaciones totales:
  $\text{memGA}_{4,2}$, $\text{memGA}_{16,2}$, $\text{memGA}_{4,4}$, $\text{memGA}_{8,4}$, $\text{memGA}_{16,4}$, $\text{memGA}_{2,8}$, $\text{memGA}_{4,8}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$

4. Para M_{2d}:

   - En tamaño de población:
     $\text{memGA}_{8,1}$, $\text{memGA}_{16,1}$, $\text{memGA}_{8,2}$, $\text{memGA}_{16,2}$, $\text{memGA}_{8,4}$, $\text{memGA}_{16,4}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$
   - En evaluaciones de memGA:
     $\text{memGA}_{16,1}$, $\text{memGA}_{16,2}$, $\text{memGA}_{8,4}$, $\text{memGA}_{16,8}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$
   - En evaluaciones totales:
     $\text{memGA}_{8,4}$, $\text{memGA}_{2,8}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{2,16}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$

5. Para M_{2e}:

   - En tamaño de población:
     $\text{memGA}_{16,1}$, $\text{memGA}_{16,2}$, $\text{memGA}_{16,8}$, $\text{memGA}_{16,16}$
   - En evaluaciones de memGA:
     $\text{memGA}_{16,1}$, $\text{memGA}_{8,2}$, $\text{memGA}_{16,2}$, $\text{memGA}_{16,4}$, $\text{memGA}_{16,8}$, $\text{memGA}_{16,16}$
   - En evaluaciones totales:
     $\text{memGA}_{8,2}$, $\text{memGA}_{16,2}$, $\text{memGA}_{16,4}$, $\text{memGA}_{16,8}$, $\text{memGA}_{16,16}$

6. Para M_{2f}:

   - En tamaño de población:
     $\text{memGA}_{4,1}$, $\text{memGA}_{16,1}$, $\text{memGA}_{4,2}$, $\text{memGA}_{8,2}$, $\text{memGA}_{16,2}$, $\text{memGA}_{4,4}$, $\text{memGA}_{8,4}$, $\text{memGA}_{4,8}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{2,16}$, $\text{memGA}_{4,16}$, $\text{memGA}_{16,16}$
   - En evaluaciones de memGA:
     $\text{memGA}_{2,4}$, $\text{memGA}_{2,8}$, $\text{memGA}_{4,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{2,16}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$
   - En evaluaciones totales:
     $\text{memGA}_{2,8}$, $\text{memGA}_{4,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{2,16}$, $\text{memGA}_{4,16}$

7. Para M_{2g}:

   - En tamaño de población:
     $\text{memGA}_{16,1}$, $\text{memGA}_{8,2}$, $\text{memGA}_{16,2}$, $\text{memGA}_{8,4}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$

- En evaluaciones de memGA:
  $\text{memGA}_{16,1}$, $\text{memGA}_{16,2}$, $\text{memGA}_{4,4}$, $\text{memGA}_{4,8}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$
- En evaluaciones totales:
  $\text{memGA}_{4,4}$, $\text{memGA}_{2,8}$, $\text{memGA}_{4,8}$, $\text{memGA}_{8,8}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$

8. Para M_{2h}:

   - En tamaño de población:
     $\text{memGA}_{8,1}$, $\text{memGA}_{16,1}$, $\text{memGA}_{8,2}$, $\text{memGA}_{16,2}$, $\text{memGA}_{8,4}$, $\text{memGA}_{16,4}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$
   - En evaluaciones de memGA:
     $\text{memGA}_{16,1}$, $\text{memGA}_{16,2}$, $\text{memGA}_{4,4}$, $\text{memGA}_{8,4}$, $\text{memGA}_{16,4}$, $\text{memGA}_{4,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$
   - En evaluaciones totales:
     $\text{memGA}_{4,4}$, $\text{memGA}_{8,4}$, $\text{memGA}_{4,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{2,16}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$

9. Para M_{2i}:

   - En tamaño de población:
     $\text{memGA}_{8,1}$, $\text{memGA}_{16,1}$
   - En evaluaciones de memGA:
     $\text{memGA}_{8,1}$, $\text{memGA}_{16,1}$, $\text{memGA}_{4,2}$, $\text{memGA}_{16,2}$, $\text{memGA}_{16,4}$, $\text{memGA}_{4,8}$, $\text{memGA}_{8,8}$, $\text{memGA}_{16,8}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$
   - En evaluaciones totales:
     $\text{memGA}_{8,1}$, $\text{memGA}_{16,1}$, $\text{memGA}_{4,2}$, $\text{memGA}_{16,2}$, $\text{memGA}_{4,4}$, $\text{memGA}_{2,8}$, $\text{memGA}_{4,8}$, $\text{memGA}_{8,8}$, $\text{memGA}_{2,16}$, $\text{memGA}_{4,16}$, $\text{memGA}_{8,16}$, $\text{memGA}_{16,16}$

	Tamaño de población al resolver M_{2g}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	28.3(11.7)	28.9(12.7)	29.1(13.9)	32.0(11.0)	30.1(14.4)
2.0	23.1(8.5)	22.3(11.9)	24.3(12.2)	21.4(9.2)	25.8(13.8)
4.0	19.4(12.2)	18.8(8.2)	19.8(10.6)	19.1(8.8)	19.5(8.4)
8.0	18.8(8.7)	17.7(8.1)	17.7(10.2)	16.0(6.7)	16.0(7.3)
16.0	17.5(11.6)	16.0(7.3)	18.9(9.0)	17.1(9.8)	17.2(10.8)

Tabla A.19: Promedio sobre 50 simulaciones independientes del tamaño de población utilizado por memGA para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_{2g}. En cada entrada de la tabla se muestra el promedio acompañado de la la desviación estándar entre paréntesis.

	Conteo de evaluaciones al resolver M_{2g}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	493.1(181.6)	501.7(155.9)	480.1(199.3)	499.8(167.6)	471.8(177.4)
2.0	441.0(158.8)	409.7(175.5)	421.3(147.9)	363.9(125.0)	415.4(184.1)
4.0	398.4(171.7)	360.9(123.9)	345.3(118.0)	342.8(124.5)	341.2(134.8)
8.0	389.4(142.0)	379.4(172.0)	373.1(153.1)	315.8(121.3)	327.7(128.7)
16.0	346.9(150.8)	339.8(122.6)	389.2(189.8)	353.8(149.9)	335.1(148.8)

Tabla A.20: Promedio sobre 50 simulaciones independientes del número del evaluaciones utilizado por memGA para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_{2g}. En cada entrada de la tabla se muestra el promedio acompañado de la la desviación estándar entre paréntesis.

	Conteo de evaluaciones totales al resolver M_{2g}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	856.6(188.4)	865.7(165.9)	826.9(219.7)	829.3(160.5)	799.8(180.2)
2.0	834.2(163.4)	782.9(175.5)	754.3(152.0)	707.8(123.3)	758.7(199.7)
4.0	762.1(176.9)	723.9(128.5)	705.2(136.3)	697.1(139.8)	682.6(137.3)
8.0	773.6(163.5)	769.7(171.1)	726.0(171.4)	673.9(137.4)	667.6(132.6)
16.0	750.5(161.4)	724.9(149.0)	745.4(192.6)	717.4(157.5)	695.6(157.4)

Tabla A.21: Promedio sobre 50 simulaciones de la suma de el número de evaluaciones utilizadas por memGA y el número de evaluaciones utilizadas por PHC aplicado a la solución provista por memGA en cada simulación, para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_{2g}. En cada entrada de la tabla se muestra el promedio acompañado de la la desviación estándar entre paréntesis.

	Tamaño de población al resolver M_{2h}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	33.8(20.4)	33.5(15.6)	30.0(13.5)	32.6(14.2)	31.5(13.8)
2.0	24.6(11.5)	24.1(10.3)	24.8(9.9)	25.4(11.3)	27.1(10.8)
4.0	20.0(10.4)	22.5(10.4)	19.8(8.1)	19.5(8.0)	20.5(8.0)
8.0	18.5(9.7)	18.3(7.6)	16.8(7.8)	18.6(10.0)	18.8(8.2)
16.0	16.6(7.4)	17.8(8.5)	18.6(10.3)	16.8(8.3)	16.6(9.2)

Tabla A.22: Promedio sobre 50 simulaciones independientes del tamaño de población utilizado por memGA para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_{2h}. En cada entrada de la tabla se muestra el promedio acompañado de la la desviación estándar entre paréntesis.

	Conteo de evaluaciones al resolver M_{2h}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	551.0(234.6)	502.1(203.2)	511.0(198.5)	512.0(193.7)	497.3(201.9)
2.0	460.6(187.1)	447.9(150.1)	425.5(149.8)	410.7(150.4)	405.9(154.9)
4.0	413.7(198.3)	413.7(156.2)	346.2(124.7)	361.5(129.5)	375.4(123.1)
8.0	380.4(149.5)	388.1(153.6)	374.1(139.7)	382.7(142.8)	330.1(130.8)
16.0	347.2(133.8)	376.5(158.3)	375.1(176.1)	354.4(150.5)	340.8(144.6)

Tabla A.23: Promedio sobre 50 simulaciones independientes del número del evaluaciones utilizado por memGA para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_{2h}. En cada entrada de la tabla se muestra el promedio acompañado de la la desviación estándar entre paréntesis.

	Conteo de evaluaciones totales al resolver M_{2h}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	917.1(245.2)	890.5(211.3)	858.3(206.9)	826.6(193.0)	817.6(203.0)
2.0	825.9(186.4)	805.4(170.3)	759.9(155.1)	747.2(172.2)	729.8(146.7)
4.0	782.5(203.2)	791.6(167.8)	696.7(130.3)	705.7(126.5)	715.6(133.9)
8.0	775.8(151.9)	748.8(157.4)	722.1(143.6)	736.6(134.9)	686.8(138.5)
16.0	752.4(137.2)	749.5(180.9)	739.3(169.0)	723.7(150.5)	713.1(143.7)

Tabla A.24: Promedio sobre 50 simulaciones del número de evaluaciones totales utilizadas por memGA complementadas con el número de evaluaciones utilizadas por el proceso PHC aplicado a la solución provista por memGA en cada simulación, para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_{2h}. En cada entrada de la tabla se muestra el promedio acompañado de la la desviación estándar entre paréntesis.

	Tamaño de población al resolver M_{2i}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	28.0(11.8)	28.3(11.7)	27.8(10.7)	28.0(11.8)	27.2(13.4)
2.0	21.2(12.5)	21.4(10.0)	21.7(10.1)	21.4(9.2)	23.1(11.8)
4.0	16.9(7.7)	14.9(6.1)	17.2(8.4)	17.8(8.9)	18.9(8.5)
8.0	13.5(6.0)	16.3(7.1)	17.1(7.6)	15.5(7.0)	17.5(11.0)
16.0	12.0(4.0)	14.3(6.4)	14.9(7.3)	15.1(7.2)	15.5(8.0)

Tabla A.25: Promedio sobre 50 simulaciones independientes del tamaño de población utilizado por memGA para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_{2i}. En cada entrada de la tabla se muestra el promedio acompañado de la la desviación estándar entre paréntesis.

	Conteo de evaluaciones al resolver M_{2i}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	475.5(187.5)	483.5(138.5)	481.3(145.4)	430.5(156.3)	432.1(161.4)
2.0	414.5(166.9)	416.2(141.2)	386.8(155.7)	364.1(127.8)	365.5(150.0)
4.0	405.4(165.4)	314.1(111.8)	356.8(147.0)	340.3(118.1)	330.3(115.4)
8.0	330.5(124.7)	366.8(115.3)	359.1(111.4)	335.5(111.4)	348.0(147.2)
16.0	312.2(106.5)	333.4(132.6)	348.9(131.5)	346.6(132.7)	348.8(152.2)

Tabla A.26: Promedio sobre 50 simulaciones independientes del número del evaluaciones utilizado por memGA para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_{2i}. En cada entrada de la tabla se muestra el promedio acompañado de la la desviación estándar entre paréntesis.

	Conteo de evaluaciones totales al resolver M_{2i}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	849.0(212.6)	843.9(166.0)	807.5(147.6)	761.9(171.1)	762.6(164.8)
2.0	782.3(187.0)	766.6(158.6)	729.2(172.2)	691.1(137.0)	702.8(159.9)
4.0	768.7(181.0)	656.0(129.5)	690.5(158.5)	697.4(134.2)	680.4(133.5)
8.0	697.1(142.4)	724.8(132.1)	718.1(130.3)	695.1(114.9)	692.0(166.8)
16.0	677.0(120.8)	688.4(137.6)	707.5(139.8)	722.6(150.7)	704.9(167.3)

Tabla A.27: Promedio sobre 50 simulaciones del número de evaluaciones totales utilizadas por memGA complementadas con el número de evaluaciones utilizadas por el proceso PHC aplicado a la solución provista por memGA en cada simulación, para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_{2i}. En cada entrada de la tabla se muestra el promedio acompañado de la la desviación estándar entre paréntesis.

Apéndice B

Variantes de memGA sobre M_6

En este apéndice aparecen reportados los resultados de aplicar memGA con diferentes valores para los parámetros α y γ a variaciones del problema M_6 2.12. En cada caso los resultados corresponden al promedio sobre 50 simulaciones independientes. El tamaño de la población exploradora fue de 4 individuos, mientras que el tamaño inicial de la población memoria fue también de 4. El criterio de paro utilizado en memGA fue el basado en el promedio sobre 4 generaciones consecutivas del promedio de la evaluación de la población, con parámetro $s_o = 0.1$ tal y como es descrito en el capítulo 3. El criterio para aceptar un conjunto de individuos como solución al problema multimodal utilizó el radio de nicho, r_o, como el valor de $\sigma_{share} = 8.0$ como reportado en la tesis de Mahfoud [36]. La probabilidad de mutación se mantuvo en 0.0, y se utilizó siempre el operador de cruce en un punto, es decir la probabilidad de mutación se fijó en 1.0.

B.1. Recursos requeridos por cada variante de memGA

La tabla B.1 contiene, para una selección de los parámetros α y γ definidos vertical y horizontalmente respectivamente, el promedio del tamaño de población utilizado por memGA; entre paréntesis aparece la desviación estándar. La tabla B.2 contiene los promedios del número de evaluaciones utilizadas para el AG. La tabla B.3 contiene el promedio de las evaluaciones totales para una combinación de parámetros α y γ. Recordemos que el termino *evaluaciones totales* hace referencia a la suma entre el número de evaluaciones utilizadas por el AG y el número de evaluaciones utilizadas por el buscador local PHC aplicado al conjunto que el algoritmo proporciona como solución. El algoritmo PHC es referido en el capítulo 4 y tiene una descripción, para el caso donde el fenotipo es un vector real, en la figura 4.10.

B.1.1. Listas de modelos no perdedores

Usando los promedios, desviaciones y tamaños de muestra de las tablas de tamaño de población, número de evaluaciones de memGA, y número de evaluaciones totales,

	Tamaño de población al resolver M_6				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	450.5(190.5)	448.0(142.3)	521.8(167.8)	477.5(175.8)	472.6(128.1)
2.0	300.3(131.4)	320.0(155.2)	366.8(158.4)	349.5(132.0)	386.5(132.9)
4.0	292.9(150.7)	332.3(168.6)	324.9(130.4)	322.5(125.4)	352.0(128.9)
8.0	339.7(162.1)	364.3(167.0)	342.1(133.5)	342.1(153.6)	366.8(201.2)
16.0	320.0(135.3)	354.5(140.2)	403.7(242.3)	393.8(208.8)	349.5(170.3)

Tabla B.1: Promedio sobre 50 simulaciones independientes del tamaño de población utilizado por memGA para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_6. En cada entrada de la tabla se muestra el promedio, μ, acompañado de la la desviación estándar entre paréntesis.

	Conteo de evaluaciones al resolver M_6				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	5251.2(2030.3)	5211.2(1487.2)	5450.4(1553.2)	4620.7(1567.4)	4464.4(1173.7)
2.0	4122.8(1621.9)	4106.4(1650.6)	4464.4(1674.5)	3914.0(1351.5)	3989.3(1228.1)
4.0	3467.9(1143.0)	3575.0(1379.1)	3285.8(1202.0)	2913.9(933.2)	3166.1(960.4)
8.0	2831.5(939.4)	2751.2(893.6)	2303.4(648.6)	2301.7(761.5)	2371.5(932.1)
16.0	2130.4(625.6)	2098.1(537.7)	2165.0(814.2)	2063.4(727.3)	2029.3(636.5)

Tabla B.2: Promedio sobre 50 simulaciones independientes del número del evaluaciones utilizado por memGA para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_6. En cada entrada de la tabla se muestra el promedio, μ, acompañado de la la desviación estándar entre paréntesis.

	Conteo de evaluaciones totales al resolver M_6				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	7813.3(2053.2)	7834.4(1527.2)	8199.2(1599.0)	7474.5(1652.9)	7358.3(1292.2)
2.0	7385.6(2192.3)	7652.5(2130.9)	8329.1(2104.2)	7981.4(1712.6)	8250.1(1619.9)
4.0	9303.3(2523.4)	9831.5(3052.1)	10178.1(2892.9)	10198.0(2815.2)	10656.1(2548.2)
8.0	14240.5(3275.1)	14650.3(3310.5)	14494.3(2902.0)	13977.1(4025.0)	13874.1(3774.6)
16.0	14831.9(2408.4)	15126.3(2118.0)	14486.1(2381.6)	14635.6(2369.2)	14592.6(2481.3)

Tabla B.3: Promedio sobre 50 simulaciones de la suma del número de evaluaciones utilizadas por memGA y el número de evaluaciones utilizadas por PHC aplicado a la solución provista por memGA en cada simulación, para diferentes valores de los parámetros α y γ al resolver el problema asociado a M_6. En cada entrada de la tabla se muestra el promedio, μ, acompañado de la la desviación estándar entre paréntesis.

en forma independiente, para comparar estadísticamente con un índice de confianza del 95 % cuál o cuáles variantes de memGA hacen uso de una menor cantidad de recursos promedio obtenemos los siguientes hechos.

- En tamaño:
  $\text{memGA}_{2,1}$, $\text{memGA}_{4,1}$, $\text{memGA}_{8,1}$, $\text{memGA}_{16,1}$, $\text{memGA}_{2,2}$, $\text{memGA}_{4,2}$, $\text{memGA}_{4,4}$, $\text{memGA}_{4,8}$, $\text{memGA}_{8,8}$
- En evaluaciones:
  $\text{memGA}_{16,1}$, $\text{memGA}_{16,2}$, $\text{memGA}_{16,4}$, $\text{memGA}_{16,8}$, $\text{memGA}_{16,16}$
- En evaluaciones totales:
  $\text{memGA}_{1,1}$, $\text{memGA}_{2,1}$, $\text{memGA}_{2,2}$, $\text{memGA}_{1,8}$, $\text{memGA}_{1,16}$

Apéndice C

Variantes de memGA sobre E_2

En este apéndice aparecen reportados los resultados de aplicar memGA con diferentes valores para los parámetros α y γ a variaciones del problema E_2 4.42. En cada caso los resultados corresponden al promedio sobre 50 simulaciones independientes. El tamaño de la población exploradora fue de 8 individuos, mientras que el tamaño inicial de la población memoria fue también de 8. El criterio de paro utilizado en memGA fue el basado en el promedio sobre 4 generaciones consecutivas del promedio de la evaluación de la población, con parámetro $s_o = 0.001$ tal y como se describe en el capítulo 3. El criterio para aceptar un conjunto de individuos como solución al problema multimodal utilizó el radio de nicho, r_o, como el valor de $\sigma_{share} = 2.5$ como se reporta en la tesis de Mahfoud [36]. La probabilidad de mutación se mantuvo en 0.0, y se utilizó siempre el operador de cruce en un punto, es decir la probabilidad de cruce se fijó en 1.0.

C.1. Recursos requeridos por cada variante de memGA

La tabla C.1 contiene, para una selección de los parámetros α y γ definidos vertical y horizontalmente respectivamente, el promedio del tamaño de problación utilizado por memGA; entre paréntesis aparece la desviación estándar. La tabla C.2 contiene los promedios del número de evaluaciones utilizadas para el AG. La tabla B.3 contiene el promedio de las evaluaciones totales para una combinación de parámetros α y γ. Recordemos que el término *evaluaciones totales* hace referencia a la suma entre el número de evaluaciones utilizadas por el AG y el número de evaluaciones utilizadas por el buscador local PHC aplicado al conjunto que el algoritmo proporciona como solución. El algoritmo PHC es referido en el capítulo 4 y tiene una descripción, para el caso donde el fenotipo es un vector real, en la figura 4.10.

C.1.1. Listas de modelos no perdedores

Usando los promedios, desviaciones y tamaños de muestra de las tablas de tamaño de población, número de evaluaciones de memGA, y número de evaluaciones totales,

	Tamaño de población al resolver E_2				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	135.4(91.5)	121.8(71.1)	122.1(66.5)	147.7(88.0)	153.2(102.8)
2.0	101.2(67.0)	104.0(67.5)	97.2(69.5)	132.3(75.2)	134.1(70.6)
4.0	91.7(87.3)	109.2(84.0)	96.6(58.2)	128.0(85.2)	138.5(143.5)
8.0	93.8(64.5)	85.5(70.1)	112.6(66.6)	120.6(75.9)	145.5(101.6)
16.0	109.2(90.9)	79.7(37.7)	101.5(53.7)	108.6(70.7)	138.8(72.6)

Tabla C.1: Promedio sobre 50 simulaciones independientes del tamaño de población utilizado por memGA para diferentes valores de los parámetros α y γ al resolver el problema asociado a E_2. En cada entrada de la tabla se muestra el promedio, μ, acompañado de la la desviación estándar entre paréntesis.

	Conteo de evaluaciones al resolver M_{7_2}				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	3594.1(1303.6)	3101.8(1242.1)	3417.7(1321.1)	3094.5(1270.6)	3273.8(1345.0)
2.0	3481.2(1232.8)	3394.0(1244.4)	3353.8(1271.7)	3302.1(1133.7)	3301.2(1044.5)
4.0	3576.1(1519.1)	3758.6(1305.0)	3600.1(1091.6)	3441.8(1257.8)	3368.9(1599.6)
8.0	3642.1(1212.8)	3247.4(1402.8)	3728.3(1204.7)	3778.8(1407.1)	3537.7(1293.7)
16.0	3755.5(1547.8)	3625.5(1315.1)	4059.7(1171.5)	3583.5(1320.1)	3545.5(1139.5)

Tabla C.2: Promedio sobre 50 simulaciones independientes del número del evaluaciones utilizado por memGA para diferentes valores de los parámetros α y γ al resolver el problema asociado a E_2. En cada entrada de la tabla se muestra el promedio, μ, acompañado de la la desviación estándar entre paréntesis.

	Conteo de evaluaciones totales al resolver E_2				
α / γ	1.0	2.0	4.0	8.0	16.0
1.0	4577.5(1595.0)	4060.2(1529.4)	4380.9(1542.5)	4177.7(1549.7)	4316.7(1619.2)
2.0	4428.8(1515.6)	4358.1(1534.2)	4249.5(1643.1)	4432.0(1383.5)	4478.8(1303.9)
4.0	4473.4(1887.5)	4831.5(1595.1)	4617.6(1343.8)	4608.1(1603.6)	4571.2(1819.1)
8.0	4703.0(1567.2)	4188.5(1793.9)	4949.5(1488.4)	4961.2(1692.9)	4857.0(1534.8)
16.0	4909.4(1929.7)	4647.8(1568.9)	5239.1(1447.5)	4723.8(1670.7)	4942.8(1409.6)

Tabla C.3: Promedio sobre 50 simulaciones de la suma del número de evaluaciones utilizadas por memGA y con el número de evaluaciones utilizadas por el proceso PHC aplicado a la solución provista por memGA en cada simulación, para diferentes valores de los parámetros α y γ al resolver el problema asociado a E_2. En cada entrada de la tabla se muestra el promedio, μ, acompañado de la la desviación estándar entre paréntesis.

en forma independiente, para comparar estadísticamente con un índice de confianza del 95 % cuál o cuáles variantes de memGA hacen uso de una menor cantidad de recursos promedio obtenemos los siguientes hechos.

- En tamaño de población:
  $\text{memGA}_{4,1}$, $\text{memGA}_{8,1}$, $\text{memGA}_{8,2}$, $\text{memGA}_{16,2}$, $\text{memGA}_{2,4}$
- En evaluaciones de memGA:
  $\text{memGA}_{2,1}$, $\text{memGA}_{1,2}$, $\text{memGA}_{2,2}$, $\text{memGA}_{8,2}$, $\text{memGA}_{1,4}$, $\text{memGA}_{2,4}$, $\text{memGA}_{1,8}$, $\text{memGA}_{2,8}$, $\text{memGA}_{4,8}$, $\text{memGA}_{1,16}$, $\text{memGA}_{2,16}$, $\text{memGA}_{4,16}$
- En evaluaciones totales:
  $\text{memGA}_{2,1}$, $\text{memGA}_{4,1}$, $\text{memGA}_{1,2}$, $\text{memGA}_{2,2}$, $\text{memGA}_{8,2}$, $\text{memGA}_{1,4}$, $\text{memGA}_{2,4}$, $\text{memGA}_{1,8}$, $\text{memGA}_{2,8}$, $\text{memGA}_{1,16}$, $\text{memGA}_{2,16}$, $\text{memGA}_{4,16}$

Apéndice D

El problema Net8

El problema de ruteo Net8 es un problema sobre una red de 16 ciudades etiquetadas con enteros de 0 a 15. La etiqueta 0 corresponde a la ciudad de partida u origen, y la etiqueta 15 a la ciudad terminal. Los lados dirigidos aparecen en la tabla D.1. Existen 5 tipos de aviones y están etiquetados por los enteros de 0 a 4. Las capacidades para cada uno de los tipos son 22, 28, 23, 28 y 20 respectivamente. Los costos de cada segmento de la red dependiendo del tipo de avión aparece la tabla D.4. La demanda, o número de pasajeros que desean viajar, de cada nodo de la red a otro nodo en la ruta, no necesariamente inmediato pero sí alcanzable, aparece en la tabla D.4. La ganancia por cada pasajero transportado de una ciudad a otra aparece en la tabla D.2. Las rutas que se consideran deseables de buscar aparecen en la tabla D.5.

	Segmentos en Net8														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	*	*	*												
1				*											
2					*			*			*				
3						*			*			*	*		
4							*	*			*			*	
5										*					
6														*	*
7											*	*			*
8									*				*	*	*
9										*			*	*	
10												*	*	*	*
11												*	*		*
12													*	*	*
13														*	
14															*

Tabla D.1: Trayectos disponibles en la red de ciudades en el problema Net8. Una marca $*$ en la posición (i, j) indica que el trayecto de la ciudad i a la ciudad j está disponible en la red.

	Ganancias en Net8														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	10	4	7	9	8	10	6	7	9	7	10	4	6	6	8
1				6			4	4	5	9	7	7	5	6	4
2					6			5	5	8	8	7	6	9	9
3						7			6	8		9	6	5	7
4							5	9	5	9	5	7	9	8	10
5										6		9	7	7	6
6														8	6
7											5	10	6	7	9
8									4	8		7	7	10	8
9										5		10	6	5	5
10												4	9	9	7
11												7	8	7	10
12													4	8	5
13														6	6
14															4

Tabla D.2: Tabla de ganancias por pasajero en el problema Net8. El número en la localidad (i, j) representa la ganancia por pasajero al moverlos de la ciudad i a la ciudad j.

	Demandas en Net8														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	7	4	7	4	8	7	5	9	4	10	7	10	9	7	10
1				7			8	9	4	6	7	9	4	4	7
2					10			5	6	5	8	10	10	6	6
3						5			4	4		10	7	7	7
4							6	4	9	5	4	5	9	10	8
5										7		8	9	6	6
6														10	8
7											10	9	5	5	4
8									9	4		7	7	8	8
9										8		7	10	4	4
10												8	10	8	9
11												6	7	10	5
12													7	8	10
13														10	4
14															10

Tabla D.3: Demanda de pasaje en el problema Net8. El número en la posición (i, j) es el número de pasajeros en la ciudad i que desean trasladarse a la ciudad j.

Costos de segmentos en el problema Net8														
		Tipos							Tipos					
i	j	0	1	2	3	4		i	j	0	1	2	3	4
0	1	5	7	8	6	10		7	15	9	6	7	5	6
0	2	7	6	10	10	7		8	9	7	8	6	9	10
0	3	8	6	5	5	8		8	13	6	9	5	6	5
1	4	8	10	10	6	10		8	14	7	6	8	10	5
2	5	10	6	8	9	8		8	15	8	10	8	5	10
2	8	7	9	9	9	8		9	10	6	8	6	10	10
2	11	10	9	9	8	5		9	13	8	7	5	7	5
3	6	9	5	9	5	8		9	14	7	5	10	5	8
3	9	5	10	10	10	8		10	12	6	7	10	7	8
3	12	9	8	5	6	5		10	13	7	7	5	7	5
3	13	8	7	5	10	5		10	14	10	8	9	10	10
4	7	7	7	6	5	9		10	15	9	8	8	6	9
4	8	6	8	8	8	5		11	12	5	6	9	7	8
4	11	9	10	9	9	8		11	13	5	10	10	9	7
4	14	10	8	5	6	6		11	15	6	5	5	8	5
5	10	10	10	9	7	8		12	13	7	5	9	8	8
6	14	8	8	6	8	8		12	14	7	6	9	5	10
6	15	8	6	5	5	8		12	15	9	7	9	10	7
7	11	5	8	5	7	8		13	14	5	9	6	7	8
7	12	9	6	10	10	10		14	15	8	5	10	7	9

Tabla D.4: Costos para cada segmento de la red del problema Net8 para cada tipo de avión. La columna i tiene la ciudad de origen y j de llegada. Los 5 tipos de aviones están etiquetados de 0 a 4.

Óptimos deseable para Net8		
Ruta	tipo	Ganancia
$(0, 1, 4, 8, 9, 10, 12, 13, 14, 15)$	1	853.0
$(0, 1, 4, 8, 9, 10, 12, 13, 14, 15)$	3	852.0
$(0, 1, 4, 8, 9, 10, 12, 14, 15)$	3	790.0
$(0, 1, 4, 8, 9, 10, 12, 14, 15)$	1	789.0
$(0, 1, 4, 8, 9, 10, 13, 14, 15)$	3	781.0
$(0, 1, 4, 8, 9, 10, 13, 14, 15)$	1	779.0
$(0, 1, 4, 7, 11, 12, 13, 14, 15)$	3	769.0

Tabla D.5: Lista de los óptimos deseables para el problema Net8. Se incluye la secuencia de la etiqueta de las ciudades, el tipo de avión usado y la ganancia obtenida.

Apéndice E

El problema Net9

El problema de ruteo Net9 es un problema sobre una red de 26 ciudades etiquetadas con enteros de 0 a 25. La etiqueta 0 corresponde a la ciudad de partida u origen, y la etiqueta 25 a la ciudad terminal. Los lados dirigidos aparecen en las tablas E.1 y E.2. Existen 3 tipos de aviones y están etiquetados por los enteros de 0 a 2. Las capacidades para cada uno de los tipos son 22, 21, y 22 respectivamente. Los costos de cada segmento de la red dependiendo del tipo de avión aparece la tabla E.7. La demanda, o número de pasajeros que desean viajar, de cada nodo de la red a otro nodo en la ruta, no necesariamente inmediato pero sí alcanzable, aparece en las tablas E.5 y E.6. La ganancia por cada pasajero transportado de una ciudad a otra aparece en las tablas E.3 y E.4. Las rutas que se consideran deseables de buscar aparecen en la tabla E.8.

	Segmentos en Net9 (Parte 1)												
	1	2	3	4	5	6	7	8	9	10	11	12	13
0	*	*	*	*									
1					*								
2						*			*				
3				*			*						*
4								*					
5										*	*		
6							*					*	
7										*		*	
8													
9													
10													
11													*
12													

Tabla E.1: Trayectos disponibles en la red de ciudades en el problema Net9. Una marca $*$ en la posición (i, j) indica que el trayecto de la ciudad i a la ciudad j está disponible en la red.

	Segmentos en Net9 (Parte 2)											
	14	15	16	17	18	19	20	21	22	23	24	25
0												
1												
2												
3							*					
4											*	
5						*						
6								*			*	
7	*								*			
8		*										
9	*		*				*					
10				*								
11	*				*	*						
12									*	*		
13												*
14					*							
15			*									
16						*			*			
17						*		*	*	*		
18						*			*	*		*
19								*				*
20										*		
21									*	*		*
22										*	*	*
23											*	
24												*

Tabla E.2: Trayectos disponibles en la red de ciudades en el problema Net9 (Continuación).

	Ganancias en Net9												
	1	2	3	4	5	6	7	8	9	10	11	12	13
0	10	6	9	8	7	9	5	5	5	9	10	7	9
1					9					4	8		9
2						8	7		7	4	4	4	
3				7			6	7		8			5
4								4					
5										8	10		9
6							6			8		8	
7										10		6	
8													
9													
10													
11													9
12													
13													
14													

Tabla E.3: Tabla de ganancias por pasajero en el problema Net9. El número en la localidad (i, j) representa la ganancia por pasajero al moverlos de la ciudad i a la ciudad j.

	Ganancias en Net9											
	14	15	16	17	18	19	20	21	22	23	24	25
0	5	7	4	8	5	4	5	10	4	6	9	8
1	9			7	8	4		4	7	8	5	4
2	7		6	4	10	9	7	7	8	8	10	8
3	7	10	5	9	4	4	5	8	6	4	5	8
4		7	5			5		8	6	4	10	10
5	10			9	10	4		5	9	6	10	4
6	8			7	7	5		9	5	4	10	9
7	8			7	10	8		7	5	10	10	8
8		10	4			9		9	10	4	7	9
9	6		8		4	9	9	5	7	7	6	8
10				6		9		10	10	10	4	5
11	9				4	4		4	6	10	6	10
12									10	10	7	4
13												4
14					4	8		6	8	5	10	5
15			7			4		9	7	7	5	4
16						9		7	4	5	4	8
17						4		8	5	7	8	7
18						7		8	4	8	4	10
19								6	8	6	9	10
20										5	8	7
21									10	10	10	7
22										6	9	7
23											6	10
24												8

Tabla E.4: Tabla de ganancias por pasajero en el problema Net9 (Continuación).

	Demandas en Net9												
	1	2	3	4	5	6	7	8	9	10	11	12	13
0	4	9	6	5	7	6	8	6	8	8	8	6	6
1					5					6	8		8
2						7	5		4	10		8	
3				4			8	7		6		6	9
4								9					
5										8	8		8
6							5			6		9	
7										6		9	
8													
9													
10													
11													10
12													

Tabla E.5: Demanda de pasaje en el problema Net9. El número en la posición (i, j) es el número de pasajeros en la ciudad i que desean trasladarse a la ciudad j.

	Demandas en Net9											
	14	15	16	17	18	19	20	21	22	23	24	25
0	8	9	10	10	7	4	7	5	10	10	4	6
1	5			4	7	10		9	4	4	4	10
2	10		7	4	5	8	7	6	8	7	10	7
3	6	6	7	4	4	10	10	5	6	8	6	8
4		5	5			9		9	9	9	4	6
5	7			5	5	8		8	9	4	8	10
6	4			9	8	4		8	8	6	6	4
7	5			5	7	5		7	7	7	6	4
8		6	6			9		8	10	4	6	5
9	5		4		6	6	6	5	7	5	10	4
10				8		6		9	9	10	7	6
11	4				4	9		7	4	10	8	10
12									10	6	8	5
13												10
14					6	7		4	6	10	7	4
15			5			6		10	8	10	6	5
16						8		8	4	6	4	8
17						7		9	7	4	9	7
18						9		6	7	7	9	5
19								5	10	5	6	10
20										5	8	9
21									8	6	5	6
22										9	5	4
23											4	5
24												8

Tabla E.6: Demanda de pasaje en el problema Net9 (Continuación).

Costos de segmentos en el problema Net9										
		Tipos						Tipos		
i	j	0	1	2		i	j	0	1	2
0	1	9	9	6		11	14	8	10	10
0	2	9	7	6		11	18	8	8	5
0	3	9	6	10		11	19	7	5	7
0	4	9	5	7		12	22	9	7	5
1	5	10	5	6		12	23	6	9	9
2	6	6	8	7		13	25	6	9	8
2	9	9	5	9		14	18	5	5	9
3	4	5	5	10		15	16	6	10	8
3	7	8	10	9		16	19	5	8	8
3	13	10	6	6		16	22	5	9	7
3	20	6	7	9		17	19	5	6	10
4	8	8	7	5		17	21	5	7	5
4	24	6	7	8		17	22	10	6	10
5	10	6	7	10		17	23	10	9	10
5	11	8	5	7		18	19	6	9	7
5	19	8	7	5		18	22	9	8	9
6	7	7	5	6		18	23	5	8	7
6	12	6	6	9		18	25	6	8	8
6	21	10	9	9		19	21	8	8	9
6	24	8	7	9		19	25	8	5	9
7	10	8	10	10		20	23	6	5	9
7	12	5	5	8		21	22	10	6	8
7	14	5	5	7		21	23	5	8	8
7	22	8	8	6		21	25	10	6	6
8	15	10	5	6		22	23	8	10	6
9	14	8	6	8		22	24	9	10	8
9	16	5	8	5		22	25	10	7	7
9	20	8	9	9		23	24	9	5	6
10	17	9	8	5		24	25	5	8	5
11	13	6	8	8						

Tabla E.7: Costos para cada segmento de la red del problema Net9 para cada tipo de avión. La columna i tiene la ciudad de origen y j de llegada. Los 5 tipos de aviones están etiquetados de 0 a 4.

Óptimos deseable para Net9		
Ruta	tipo	Ganancia
(0, 2, 6, 7, 10, 17, 19, 21, 22, 23, 24, 25)	2	1031.0
(0, 2, 6, 7, 14, 18, 19, 21, 22, 23, 24, 25)	2	1025.0
(0, 2, 6, 7, 10, 17, 19, 21, 22, 23, 24, 25)	0	1025.0
(0, 2, 6, 7, 14, 18, 19, 21, 22, 23, 24, 25)	0	1023.0
(0, 1, 5, 11, 14, 18, 19, 21, 22, 23, 24, 25)	2	1021.0
(0, 1, 5, 11, 14, 18, 19, 21, 22, 23, 24, 25)	0	1014.0
(0, 2, 6, 7, 14, 18, 19, 21, 22, 23, 24, 25)	1	1002.0
(0, 2, 6, 7, 10, 17, 19, 21, 22, 23, 24, 25)	1	999.0
(0, 3, 4, 8, 15, 16, 19, 21, 22, 23, 24, 25)	2	986.0
(0, 2, 6, 7, 10, 17, 21, 22, 23, 24, 25)	2	985.0
(0, 1, 5, 11, 14, 18, 19, 21, 22, 23, 24, 25)	1	985.0
(0, 3, 4, 8, 15, 16, 19, 21, 22, 23, 24, 25)	0	984.0
(0, 2, 6, 7, 10, 17, 21, 22, 23, 24, 25)	0	973.0
(0, 3, 4, 8, 15, 16, 19, 21, 22, 23, 24, 25)	1	961.0
(0, 2, 6, 7, 10, 17, 21, 22, 23, 24, 25)	1	949.0
(0, 1, 5, 10, 17, 19, 21, 22, 23, 24, 25)	2	945.0
(0, 2, 6, 7, 10, 17, 19, 21, 22, 24, 25)	2	940.0
(0, 2, 6, 7, 10, 17, 19, 21, 22, 24, 25)	0	938.0
(0, 1, 5, 11, 18, 19, 21, 22, 23, 24, 25)	2	937.0
(0, 1, 5, 10, 17, 19, 21, 22, 23, 24, 25)	0	937.0
(0, 3, 7, 10, 17, 19, 21, 22, 23, 24, 25)	2	928.0

Tabla E.8: Lista de los óptimos deseables para el problema Net9. Se incluye la secuencia de la etiqueta de las ciudades, el tipo de avión usado y la ganancia obtenida.

Bibliografía

[1] J. H. Holland, *Adaptation in Natural and Artificial Systems.* Ann Arbor, MI: University of Michigan Press, 1975.

[2] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning.* MA: Addison-Wesley Publishing Company, 1989.

[3] M. Mitchell, *An introduction to Genetic Algorithms.* MA: MIT Press, 1998.

[4] L. Davis, *Handbook of genetic algorithms.* International Thomson Computer Press, 1991.

[5] D. Whitley, "A genetic algorithm tutorial," Tech. Rep. CS-93-103, Colorado State University, November 1994.

[6] C. R. Darwin, *El origen de las especies.* Mexico: UNAM, 1997.

[7] C. R. Darwin, *The descent of man.* Princeton, NY: Princeton University, 1981.

[8] L. J. Eshelman y J. D. Schaffer, "Real-coded genetic algorithms and interval schemata," en *Foundations of Genetic Algorithms, 2,* (San Mateo CA), pp. 187–202, Morgan Kaufman, 1993.

[9] C. Z. Janikow y Z. Michalewicz, "An experimental comparison of binary and floating point representatios in genetic algorithms," en *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 31–36, 1991.

[10] A. J. Mason, "Partition coefficients, static deception and deceptive problems for non-binary alphabets," en *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 210–214, 1991.

[11] J. R. Koza, *Genetic programming: On the programming of computers by means of natural selection.* Cambridge, MA: MIT Press, 1992.

[12] D. E. Goldberg, K. Deb, y B. Korb, "Don't worry, be messy," en *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 24–30, 1991.

[13] J. T. Richardson, M. R. Palmer, G. Liepins, y M. Hilliard, "Some guidelines for genetic algorithms with penalty functions," en *Proceedings of the Third International Conference on Genetic Algorithms*, (San Mateo, California), pp. 191–197, Morgan-Kaufmann, 1989.

[14] D. M. Levine, *A parallel genetic algorithm for the set partition problem.* PhD thesis, Illinois Intitute of Technology, May 1994.

[15] L. Kallel y M. Schoenauer, "Alternative random initialization in genetic algorithm," en *Proceedings of the Seventh International Conference on Genetic Algorithms*, (Morgan-Kaufmann), pp. 268–275, 1997.

[16] J. J. Grefenstette, "Incorporating problem specific knowledge in genetic algorithms," en *Genetic algorithms and Simulated annealing* (L. D. Davis, ed.), pp. 42–60, Morgan Kaufman Publishers, 1987.

[17] A. Shultz y J. J. Grefenstette, "Improving tactical plans with genetic algorithms," en *IEEE conference on tools for IA*, pp. 328–334, Morgan Kaufman, 1990.

[18] K. A. De Jong y J. Sarma, "Generation gaps revisited," en *Foundations of Genetic Algorithms* (D. Whitley, ed.), (San Mateo, CA), pp. 19–28, Morgan Kaufmann, 1993.

[19] G. Syswerda, "A study of reproduction in generational and steady state genetic algorithms," en *Foundations of Genetic Algorithms* (G. J. E. Rawlins, ed.), (San Mateo:Morgan Kaufman), pp. 94–101, 1991.

[20] J. E. Baker, "Reducing bias and inefficency in the selection algorithm," en *Proceedings of the Second International Conference on Genetic Algorithms*, pp. 14–21, 1987.

[21] A. Brindle, *Genetic algorithms for function optimization.* PhD thesis, University of Alberta, 1981.

[22] D. E. Goldberg y K. Deb, "A comparative analysis of selection schemes used in genetic algorithm," en *Foundations of Genetic Algorithms* (G. J. E. Rawlins, ed.), (San Mateo:Morgan Kaufman), pp. 69–93, 1991.

[23] J. E. Baker, "Adaptive selection methods for genetic algorithms," en *Proceedings of the International Conference on Genetic Algorithms*, pp. 100–111, 1985.

[24] C. Bridges y D. Goldberg, "An analysis of reproduction and crossover in a binary-coded genetic algorithm," en *Proceedings of the Third International Conference on Genetic Algorithms*, 1989.

[25] C. Bridges y D. Goldberg, "Biases in the crossover landscape," en *Proceedings of the Third International Conference on Genetic Algorithms*, 1989.

[26] W. M. Spears y K. A. De Jong, "An analysis of multi-point crossover," en *Foundation of Genetic Algorithms* (G. J. E. Rawlings, ed.), (San Mateo, CA), pp. 301–315, 1991.

[27] G. Syswerda, "Uniform crossover in genetic algorithms," en *Proceedings of the Third International Conference on Genetic Algorithms* (J. D. Schaffer, ed.), (San Mateo, California), pp. 2–9, Morgan Kaufmann, 1989.

[28] D. E. Goldberg y R. L. Jr., "Alleles, loci, and the TSP," en *Proceedings of the First International Conference on Genetic Algorithms*, (San Mateo, California), pp. 154–159, Morgan Kaufman, 1985.

[29] D. Whitley, T. Starkweather, y D. Fuquay, "Scheduling Problems and Traveling Salesman:The Genetic Edge Recombination Operator," en *Proceedings of the Third International Conference on Genetic Algorithms* (J. D. Schaffer, ed.), (San Mateo, California), pp. 133–140, Morgan Kaufman, 1989.

[30] D. Whitley, "The Genitor Algorithm and Selection Pressure: Why Ranked-Based Allocation of Reproductive Trail is Best," en *Proceedings of the Third International Conference of Genetic algorithms* (J. D. Schaffer, ed.), pp. 116–121, Morgan Kaufmann, June 1989.

[31] L. J. Eshelman, "The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination," en *Foundations of Genetic Algorithms* (G. J. E. Rawlins, ed.), (San Mateo, CA), pp. 265–283, Morgan Kaufman, 1991.

[32] K. A. De Jong, *An analysis of the behavior of a class of genetic adaptive systems.* PhD thesis, University of Michigan, 1975.

[33] D. E. Goldberg y J. Richarson, "Genetic algorithms with sharing for multimodal function optimization," en *Proceedings of the Second International Conference on Genetic Algorithms*, pp. 41–49, 1987.

[34] K. Deb y D. E. Goldberg, "An investigation of niche and species formation in genetic function optimization," en *Proceedings of the Third International Conference on Genetic Algorithms* (J. D. Schaffer, ed.), pp. 42–50, 1989.

[35] D. Beasley, D. Bull, y R. R. Martin, "A sequential niche technique for multimodal function optimization," *Evolutionary Computation*, vol. 2, no. 1, pp. 101–125, 1993.

[36] S. W. Mahfoud, *Niching methods in genetic algorithms.* PhD thesis, University of Illinois, September 1995.

[37] D. E. Goldberg, K. Deb, y J. Horn, "Massive multimodality deception and genetic algorithms," en *Parallel Problem Solving From Nature, 2*, pp. 37–46, North Holland, 1992.

[38] C. A. Hjorring, *The vehicle routing problem and local search metaheuristics.* PhD thesis, University of Auckland, October 1995.

[39] J. E. Beasley y P. C. Chu, "A genetic algorithm for the set covering problem," en *European Journal of Operational Research*, June 1995.

[40] H.-L. Fang, P. Ross, y D. Corne, "A promising GA approach to job shop scheduling, rescheduling for and open shop scheduling problems," en *Proceedings of the 5th International Conference in Genetic Algorithms* (S. Forrest, ed.), (San Mateo), pp. 375–382, Morgan-Kaufmann, 1993.

[41] S. García-Lumbreras, *The multi-stop aircraft routing problem.* PhD thesis, ITESM, December 1995.

[42] M. Garey y D. Johnson, *Computers and Intractability.* New York: W.H. Freeman and Company, 1979.

[43] C. H. Papadimitriou y K. Steiglitz, *Combinatorial Optimization: Algorithms and Ccomplexity.* Englewood Cliffs, NJ: Prentice-Hall, Inc., 1982.

[44] D. H. Wolpert y W. G. Macready, "No free lunch theorems for search," Tech. Rep. SFI-TR-95-02-010, Santa Fe Institute, 1995.

[45] B. Sareni y L. Krahenbuhl, "Fitness sharing and niching methods revisited," *IEEE Transactions on Evolutionary Computation*, no. 3, pp. 97–106, 1998.

[46] G. R. Walsh, *Methods of optimization.* John Wiley and Sons Ltd., 1975.

[47] S. Kirkpatrick, C. Gelatt, y M. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[48] F. Glover y M. Laguna, *Tabu search.* Boston, NJ: Kluwer, 1997.

[49] D. E. Goldberg, "Sizing population for serial and parallel genetic algorithms," en *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 70–79, 1989.

[50] K. Krishnakumar, "Micro-genetic algorithms for stationary and non-starionary function optimization," en *SPIE Proceedings: Intelligence Control and Adaptive Systems*, pp. 289–296, 1989.

[51] G. Dozier, J. Bowen, y D. Bahler, "Solving small and laarge scale constraint satisfaction problems using a heuristic-based microgenetic algorithm," en *Proceedings of the IEEE Conference on Evolutionary Computation*, pp. 306–311, 1994.

[52] C. Coello-Coello y G. Toscano-Pulido, "A micro-genetic algorithm for multiobjective optimization," en *First International Conference on Evolutionary Multi-Criterion Optimization* (E. Zitzler, K. Deb, L. Thiele, C. Coello, y D. Corne, eds.), pp. 126–140, Springer-Verlag, 2001.

[53] C. Coello-Coello y G. Toscano-Pulido, "Multiobjetive optimization using a micro-genetic algorithm," en *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 274–282, Morgan Kaufman Publishers, 2001.

[54] J. D. Knowles y D. W. Corne, "Approximating the nondominated front using the pareto archived evolution strategy," *Evolutionary Computation*, vol. 8, no. 2, pp. 149–172, 2000.

[55] X. Yin y N. Germay, "A Fast Genetic Algorithm with Sharing Scheme Using Cluster Analysis Methods in Multimodal Function Optimization," en *Artificial Neural Nets and Genetic Algorithms, Proceedings of the International Conference in Inssbruck* (R. F. Albrecht, C. R. Reeves, y N. C. Steele, eds.), (Austria), pp. 450–457, Springer-Verlag, 1993.

[56] X. Yin y N. Germay, "Improving Genetic Algorithms with Sharing through Cluster Analysis," en *Proceedings of the Fifth International Conference on Genetic Algorithm* (S. Forrest, ed.), (San Mateo, CA), pp. 100–100, Morgan Kaufman, 1993.

[57] K. A. De Jong, W. M. Spears, y D. F. Gordon, "Using Markov chains to analize GAFOs," en *Foundations of Genetic Algorithms, 3*, (San Mateo CA), pp. 115–137, 1995.

[58] D. E. Goldberg y P. Segrest, "Finite Markov chain analysis of genetic algorithms," en *Proceedings of the Second International Conference on Genetic Algorithms* (J. J. Grefenstette, ed.), pp. 1–8, 1987.

[59] J. Horn, "Finite Markov chain analysis of genetic algorithms with niching," en *Proceedings of the Fifth International Conference on Genetic Algorithms*, pp. 110–117, 1993.

[60] M. Hulin, "Analysis of schema distributions," en *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 204–209, 1991.

[61] C. A. Ankenbrandt, "An extension to the theory of convergence and a proof of the time complexity of genetic algorithms," en *Foundations of Genetic Algorithms* (G. J. E. Rawlins, ed.), (San Mateo:Morgan Kaufman), pp. 53–68, 1991.

[62] T. Blickle y L. Thiele, "A comparison of selection schemes used in evolutionary algorithms," *Evolutionary Computation*, vol. 4, pp. 361–394, 1997.

[63] H. Muhlenbein, "Evolution in time and space - the parallel genetic algorithm," en *Foundations of Genetic Algorithms* (G. J. E. Rawlins, ed.), (San Mateo:Morgan Kaufman), pp. 316–337, 1991.

[64] W. W. Hines y D. C. Montgomery, *Probabilidad y estadística para ingeniería y administración.* México: CECSA, 1993.

[65] A. Pétrowski, "A clearing procedure as a niching method for genetic algorithms," *Proceedings of the IEEE 3rd International Conference on Evolutionary Computation*, pp. 798–803, 1996.

[66] G. L. Nemhauser y L. A. Wolsey, *Integer and combinatorial optimization.* New York: John Wiley and Sons, 1988.

[67] D. Whitley, T. Starkweather, y D. Fuquay, "Scheduling problems and TSP: The genetic edge recombination operator," en *Proceedings of the Third International Conference in Genetic Algorithms* (J. D. Schaffer, ed.), pp. 133–140, Morgan Kaufmann, 1989.

[68] I. M. Oliver, D. J. Smith, y J. R. C. Holland, "An study of permutation crossover operators on the travelling salesman problem," en *Proceedings of the Second International Conference on Genetic Algorithms*, pp. 224–230, 1987.

[69] T. Starkweather, S. McDaniel, K. Mathias, y D. Whitley, "A comparison of genetic sequencing operators," en *Proceedings of the Forth International Conference on Genetic Algorithm*, pp. 69–76, 1991.

[70] B. R. Fox y M. B. MacMahon, "Genetic operators for sequencing problems," en *Foundations of Genetic Algorithms* (G. J. E. Rawlins, ed.), (San Mateo:Morgan Kaufman), pp. 284–300, 1991.

[71] I. Ordonez y M. Valenzuela, "Solving ordering optimization problems using traditional crossover," en *Working paper. ITESM. Centro de Inteligencia Artificial.*, 1991.

[72] E. Bautista, "Utilización de un algoritmo genético para resolver el problema de programación de tareas en dos máquinas," Master's thesis, ITESM, Campus Monterrey, Diciembre 1991.

[73] G. Dantzig, *Linear Programming and Extensions.* Princeton, N.J.: Princeton University Press, 1963.

[74] F. S. Hillier y G. J. Lieberman, *Introduction to Operations Research.* McGraw-Hill, 2001.

[75] M. Valenzuela-Rendón y E. Uresti-Charre, "A non-generational genetic algorithm for multiobjetive optimization," en *Proceedings of the Seventh International Conference on Genetic Algorithms*, (Morgan-Kaufmann), pp. 658–665, 1997.