Universidad de Granada

Department of Computer Science and Artificial Intelligence

Region-Based Memetic Algorithms for Global and Multimodal Continuous Optimisation

PhD Thesis

Benjamin Lacroix

October 2014

Editor: Universidad de Granada. Tesis Doctorales
Autor: Benjamin Lacroix
ISBN: 978-84-9125-032-6
URI: http://hdl.handle.net/10481/39823

This work, submitted by Benjamin Lacroix to obtain the degree of PhilosophiæDoctor, has been realized within the PhD program "Information and Communication Technologies" of the University of Granada, under the supervision of Prof. Daniel Molina Cabrera and Prof. Francisco Herrera Triguero.

This work has been funded by the European Commission under the MIBISOC project (Grant Agreement: 238819), within the action Marie Curie Initial Training Network of the 7FP.

El doctorando Benjamin Lacroix y los directores de la tesis Daniel Molina Cabrera y Francisco Herrera Triguero garantizamos, al firmar esta tesis doctoral, que el trabajo ha sido realizado por el doctorando bajo la dirección de los directores de la tesis y hasta donde nuestro conocimiento alcanza, en la realización del trabajo, se han respetado los derechos de otros autores a ser citados, cuando se han utilizado sus resultados o publicaciones.

Granada,

PhD Fellow

Benjamin Lacroix

Directors

Daniel Molina Cabrera

Francisco Herrera Triguero

Acknowledgements

This work has been funded by the European Commission under the MIBISOC project (Grant Agreement: 238819) within the action Marie Curie Initial Training Network of the 7FP.

On a more personal level, I would like to thank:

- My supervisors, Francisco Herrera and Daniel Molina, for their precious support and advices.
- My family and especially my parents and sisters for their emotional support and for still loving me despite my repeated expatriations.
- My friends and colleagues. Every members of the MIBISOC ITN for making the meetings such a great, fun and interesting experience. A special thought goes of course to Michela Fazzorali for being so kind hearted, Jeremie Dubois-Lacoste for being such a great buddy. Every *becarios y contratados* of the SCI2S, especially Isaac Triguero Velázquez and Daniel Peralta Cámara for taking care of the clusters and equipments. My friends in Granada for making my stay in this beautiful city that great especially Mathieu Alexandre Oscar, Austin Hazlett and Allana Ross, Tarek Ziad, the Cuesta Gomerez crew... Luis Gonzales Jaime, Candela Menendez Baena and her flatmates for hosting me so kindly in the final stages of my thesis writing. All my friends from abroad who visited me and the even ones who did not. Especially Julien Arnould and Claire Ribeyre for hosting me on my way down to or up from Granada.

Resumen

Esta tesis se centra en el estudio y el diseño de algoritmos meméticos (AMs) para optimización continua. Esta investigación se inicia con el estudio de la cooperación entre los componentes de búsqueda global (BG) y la búsqueda local (BL) del AM, y conduce a la propuesta de una nueva estrategia de nichos denominada estrategia de nichos basada en regiones (region-based niching). A partir de dicha propuesta de nichos, se han desarrollado una nueva familia de AMs, denominados AMs basados en regiones. La originalidad de esta estrategia se basa en dividir el espacio de búsqueda en hipercubos de igual tamaño denominados regiones que definen los límites de cada nicho. Cuando es incluído dentro de un AM, se mantiene de tal modo la diversidad que se garantiza una más adecuada exploración del espacio de búsqueda. El objetivo es ofrecer una más controlada separación entre el componente de BL y el de BG para mantener las tareas de exploración y explotación lo más separadas posibles, y mejorar así la eficiencia de la búsqueda. Cuando se incluye dentro del AM, esta estrategia al ofrecer una separación más clara entre exploración y explotación:

- Se fuerza al algoritmo de búsqueda global a realizar una búsqueda entre regiones para garantizar que el espacio de búsqueda global se explora suficientemente.
- Al forzar al algoritmo de exploración local a realizar la búsqueda intra-region (dentro de cada región) se asegura que las regiones más prometedoras identificadas por la BG sean explotadas de forma adecuada.

AM basado en regiones ha sido aplicado a dos tipos de problemas:

- Problemas de optimización global, en los que se busca identificar un único óptimo global de la función objetivo. Los resultados ofrecidos para este tipo de problemas muestran ser muy competitivos con las técnicas del estado-del-arte.
- Problemas de optimización multimodal, en los que se busca identificar y preserver múltiples óptimos globales de la función objetivo, es decir, obtener un conjunto amplio de soluciones satisfactorias. La definición de nichos mediante regiones permiten incluir en nuestro modelo un archivo compuesto por un índice de regiones excluídas de seguir explorándose, reduciendo el espacio de búsqueda y por tanto mejorando la eficiencia de la misma. El algoritmo resultante ofrece una mejor exploración y resultados con mayor precisión que algoritmos existentes.

Contents

List of Figures

Resumen y Conclusiones en Español

En este apartado se incluye el resumen en castellano de la tesis, para cumplir con los requisitos necesarios para poder acceder al título de doctor en Tecnologías de la Información y la Comunicación de la Universidad de Granada.

Introducción

El objetivo primario de esta memoria es analizar y estudiar una nueva estrategia de nichos que consiste en dividir el espacio de búsqueda en hipercubos de igual tamaño llamados regiones. Estas regiones delimitan los nichos dentro del espacio de búsqueda. Al implementarse este criterio dentro de un Algoritmo Evolutivo, AE, únicamente una solución puede encontrase dentro de una misma región, por lo que en cada región compiten las soluciones de la población, manteniéndose en dicha región únicamente la mejor solución.

Esta estrategia de regiones es muy sencilla y rápida, y se utiliza para dos objetivos distintos:

- Mantener la diversidad de la población del AE, para evitar una excesivamente rápida convergencia y así asegurar una adecuada exploración del espacio de búsqueda, y poder así obtener mejores resultados.
- Permitir identificar múltiples óptimos en problemas multimodales, ya que puede ser de utilidad obtener distintas óptimos distintas.

Para abordar ambos objetivos, aplicamos la estrategia de nichos basada en regiones en dos Algoritmos Meméticos, AMs, distintos, cada uno de ellos orientado específicamente a cada uno de los objetivos anteriores.

El contenido de este apartado está organizado en varias secciones:

- En la sección primera resumimos la problemática que intentamos abordar.
- En la sección segunda se introduce la propuesta de la tesis.
- En la sección tercera se plantean los objetivos propuestos.
- En la sección cuarta se plantea el desarrollo de la tesis, resumiendo cada uno de sus capítulos, y presentando para cada propuesta presentada una breve discusión de los resultados objetivos.
- Finalmente, se resumen las principales conclusiones obtenidas, y posibles líneas de trabajo futuro.

Planteamiento del problema

La optimización es un área de gran interés ya que se presenta en múltiples problemas de ciencia e ingeniería que no pueden ser abordados mediante un proceso de búsqueda exhaustiva. Por tanto, es necesario recurrir a técnicas como las metaheurísticas que permiten abordarlo de forma aproximativa. Estos algoritmos permiten obtener resultados de gran calidad dentro de un número limitado de generación de soluciones.

En los problemas de optimización continua el problema es doble: Por un lado, es necesario realizar una búsqueda por todo el espacio de búsqueda, y por el otro, es necesario obtener precisión en las soluciones para obtener buenos resultados.

Las metaheurísticas pueden clasificarse en dos categorías:

- Algoritmos de Búsqueda Global (BG), normalmente algoritmos basados en población que generan soluciones por todo el espacio de búsqueda. Esta amplia cobertura ofrece una amplia exploración del espacio de búsqueda, y son conocidos por su habilidad de exploración. Son especialmente importantes en problemas multimodales ya que su estructura está diseñada para evitar atascarse en óptimos locales.
- Métodos de Búsqueda Locales, BL, diseñados para alcanzar rápidamente soluciones precisas en el vecindario de una solución dada. Suelen consistir en modificaciones locales a una solución hasta que es encontrada la mejor solución en un entorno cercano. Como consecuencia, su rendimiento es muy bueno en problemas unimodales, pero con el grave inconveniente de que al trabajar en un área restringida se corre el riesgo de atascarse en un óptimo únicamente local.

Los Algoritmos Meméticos son algoritmos evolutivos que combinan un algoritmo de Búsqueda Global, BG, y un algoritmo de Búsqueda Local, BL, para realizar cada una de estas tareas.

El objetivo de los AMs es que cada uno de sus componentes pueda centrarse en la labor encomendada (BG ó BL) ya que es el algoritmo más eficiente para ello, para que el AM puede conseguir un resultado óptimo. Sin embargo, en optimización continua no existe una separación tajante entre un algoritmo de exploración y de explotación. Aunque es fácil limitar el grado de explotación de la BL no lo es en el caso del algoritmo de BG, por lo que dicho algoritmo dedicará parte de su esfuerzo en explotar las solución, redundando en una peor exploración global ya que dicha tarea sería mejor realizada por el método de BL.

El algoritmo BG debería ser forzado a explorar en el espacio de búsqueda y dejar que el método de BL se centre en mejorar las solución encontradas. Las estrategias de nichos son propuestas que se encargan de mantener una adecuada diversidad entre las soluciones del algoritmo poblacional, para mantener la exploración.

Propuesta de Tesis

En esta tesis proponemos, desarrollamos y estudiamos el uso de una nueva estrategia de nichos a la que llamamos estrategia de nichos basada en regiones. Al incorporar este criterio de nichos se fomenta la exploración del algoritmo de BG ya que se evita que explore en las cercanías de las soluciones de la población. Así, la tarea de explorar dentro de cada región le corresponde a la BL.

Aplicamos esta estrategia a un AM, al que denominamos AM basado en regiones, RMA en inglés, y hemos estudiado cómo la estrategia de nichos propuesta mejora su rendimiento para problemas de optimización continua, incluyendo problemas multimodales.

Objetivos de la tesis

Los problemas de optimización continua puede clasificarse en dos categorías:

- Problemas de optimización global: El objetivo es identificar un único óptimo global de la función objetivo. Es decir, se alcanza el éxito con un único óptimo.

- Problemas de optimización multimodal: En muchos problemas del mundo real, es deseable obtener múltiples soluciones distintas, en vez de una única solución. La particularidad de la optimización multimodal reside en la necesidad de identificar y mantener tantas soluciones óptimas como sea posible.

En ambos casos, la exploración adecuada del espacio de búsqueda es la pieza clave, por lo que requiere una especial atención. En optimización global, es muy alto el riesgo de que la búsqueda converja en óptimos locales cuando se trabajan problemas con múltiples óptimos, con lo que empeoraría significativamente el rendimiento del algoritmo de búsqueda. En optimización multimodal, la necesidad de asegurar una fuerte exploración influye de forma muy directa en la habilidad del algoritmo en identicar un número mayor de óptimos globales repartidos por todo el espacio de búsqueda. Además, en la tesis, abordamos otros aspectos cruciales para optimización multimodal, como la preservación de los óptimos encontrados y su uso en la población para guiar la búsqueda.

En esta tesis presentamos que utilizar RMAs es la forma más eficiente de abordar los problemas de optimización tanto globales como multimodales.

Como primer objetivo, mostraremos cómo el uso de la estrategia de nichos puede ayudar a diseñar un algoritmo memético que separe de forma clara los objetivos de exploración y explotación entre el AE y el método de BL para una adecuada optimización global. De esta manera, se mantiene una mayor diversidad dentro de la población que posibilita una mejor exploración del espacio de búsqueda para dejar menos regiones sin explorar. El uso de una estrategia de nichos como la propuesta lo lo posibilita ya que el AE realiza búsqueda *entre-nichos* enfocando sus esfuerzos en identificar los nichos más prometedores mientras el método de BL realiza una búsqueda intra-nicho explotando los nichos identificados por el AE.

El segundo objetivo de la tesis es utilizar las propiedades de nuestro concepto de región para definir nichos en optimización multimodal y mejorar así sus resultados. No sólo, como del objetivo anterior, nos beneficiamos de la estrategia de nichos propuesta para obtener mayor diversidad (y, por tanto, mejor exploración), si no que vamos a utilizar la novedosa definición de nicho como regiones para crear muy fácilmente un índice de las regiones ya exploradas y optimizadas. De esta manera, podemos descartarlas en siguientes exploraciones, mejorar mucho la búsqueda tanto en resultados como en eficiencia.

Desarrollo de la tesis

A continuación, detallo un resumen de cada uno de los capítulos de la tesis, resumiendo las propuestas incluidas en la memoria y presentando para cada una de ellas una breve discusión de los resultados obtenidos.

El primer capítulo presenta la problemática de la optimización real, introduciendo las metaheurísticas para optimización continua, prestando especial atención los algoritmos meméticos, y las técnicas de optimización multimodal, mediante un repaso de la literatura. El resto de capítulos presentan las propuestas de la tesis, por lo que ofrecemos un resumen de cada uno de dichos capítulos.

Conclusiones del Capítulo 2: Algoritmo Memético Basado en Regiones con Encadenamiento de Búsqueda Local para Optimization Global

Con el objetivo de obtener una fuerte separación entre el esfuerzo de exploración del AE y el esfuerzo de explotación que ofrece la Búsqueda Local, BL, de un AM, aplicamos la estrategia de nichos basados en regiones dentro de un AM. Al incorporar este criterio de nichos se fomenta la exploración del AE ya que se evita que explore en las cercanías de las soluciones de la población. En cambio, esa tarea de explorar las zonas cercanas, explotando las soluciones ya alcanzados le corresponde a la BL, que explorará en el vecindario de las soluciones para centrar la exploración dentro de su región.

Aplicamos esta estrategia a un algoritmo previo, el AM con encadenamiento de BL, MA-LS-CMA [MLGMH10], que aplica de forma alterna un algoritmo genético estacionario, Steady-State Genenetic Algorithm, como AE, y el algoritmo CMA-ES como método de BL en el proceso de encadenamiento. El encadenamiento de BL es un mecanismo que permite adaptar la intensidad de la BL en función de la calidad de la solución sobre la que se aplica. Almacena para cada solución los parámetros de la BL, permitiendo que cuando se aplica la BL sobre una solución de nuevo, la BL *continúa* donde se quedó.

Para limitar la influencia del tamaño del niño/región (definido por el número de divisiones por dimensión del espacio de búsqueda), proponemos también un proceso para modificar dinámicamente el tamaño del nicho durante la búsqueda. La idea es simple, se incrementa varias veces durante la búsqueda el número de divisiones por dimensión (reduciendo así el tamaño del nicho).

El algoritmo resultante lo hemos llamado AM basado en regiones con encadenamiento de BL y CMA-ES, en inglés *region-based MA with LS chaining and CMA-ES*, RMA-LSCh-CMA.

Hemos comparado el RMA-LSCh-CMA, usando el benchmark de optimización continua propuesto en la Sesión Especial de Optimización de Parámetros Reales, del Congreso de IEEE de Computación Evolutiva del 2005, IEEE Conference on Evolutionary Competition, CEC'2005 para dimensiones 10, 30, 50; y el benchmark propuesto en Número especial de Soft Computing para optimización de altas dimensiones (Special Issue on Large Scale Optimiation, SOCO'2011), con problemas de dimensión 100. A partir de los experimentos realizados con dichos benchmarks, hemos estudiado la influencia del mecanismo de nichos propuesto, obteniendo las siguientes conclusiones:

- La modificación dinámica del tamaño de nicho otorga más robustez al algoritmo, al hacerle menos dependiente de dicho parámetro, especialmente cuando se considera la distinta dimensionalidad de los problemas abordados.

- El uso del mecanismo de nichos basados en regiones, al mantener mayor diversidad en la población, mejora significativamente el rendimiento del AM.

Posteriormente, hemos comparamos los resultados obtenidos por nuestra propuesta con los resultados obtenidos por un conjunto representativo de algoritmos, IPOP-CMA-ES, MDE-pBX, y 3SOME, con conclusiones interesantes:

- En ambos benchmarks, hemos mejorado de forma estadísticamente significativa los resultados obtenidos por MDE-pBX.

- No detectamos mejoras significativas respecto a IPOP-CMA-ES en el benchmark CEC'2005, pero detectamos que IPOP-CMA-ES se comporta mejor con menor dimensionalidad, obteniendo nuestro algoritmo mejores resultados con mayor dimensionalidad. Esta tendencia se confirma por el hecho de que RMA-LSCH-CMA obtiene mejoras significativas en el SOCO'2011 para dimensión 100.

- Mientras que RMA-LSCh-CMA obtiene resultados estadísticamente mejores que 3SOME en el benchmark CEC'2005, 3SOME domina ligeramente al nuestro en el benchmark del SOCO'2011. La razón puede deberse a que el CMA-ES se comporta de forma menos eficiente al aumentar la dimensionalidad, lo cual motiva el uso de otro método de BL para problemas de optimización de mayor dimensión.

Resumen del Capítulo 3: Algoritmo memético basado en regiones con archivo para optimización multimodal

Motivados por la idea de probar la propuesta de definir nichos mediante regiones al propósito original de las estrategias de nichos, presentamos un modelo basado en el capítulo 2 pero modificado para abordar problemas de optimización multimodal.

En el Capítulo 3 presentamos las distintas modificaciones del modelo anterior para adaptarlo a la nueva categoría de problemas, y tomar ventaja de la definición de nicho introducida en esta tesis:

- Aplicación de la BL: El modelo propuesto no hace más uso de encadenamientos de la BL. Ahora, se mantiene la aplicación de la BL sobre una misma solución hasta que deja de ofrecer una mejora suficiente.
- Uso de un archivo de áreas excluyentes: El uso de un archivo permite la conversación de los óptimos identificados, y evita su pérdida potencial dentro del proceso evolutivo del AE, limitando al mismo tiempo la dependencia del algoritmo respecto al tamaño de la población. La novedad de nuestro modelo es considerar las regiones representadas por las soluciones del archivo como zonas de exclusión, en las que el AE no podrá generar nuevas soluciones. Este proceso se puede hacer de forma muy eficiente en tiempo gracias al uso de índices que permiten recuperar rápidamente las regiones.

Llamamos al algoritmo resultante, Algoritmo Memético basado en Regiones, con memoria, en inglés *region-based memetic algorithm with archive*, RMA-Archive. RMA-Archive fue comparado usando el benchmark propuesto durante la competición y sesión especial de Métodos de Nichos para Optimización de Funciones Multimodales, del IEEE Conference on Evolutionary Computation, CEC'2013.

Al igual que en el capítulo anterior, hemos analizado las mejoras obtenidas por las novedades del modelo. Desde un primer momento, hemos observado que la estrategia basada en regiones implica menor carga en tiempo que el concepto tradicional de nichos basados en distancia. También hemos demostrado que al considerar a las regiones representadas en el archivo como espacios que el AE no puede seguir explorando, se mejora significativamente el grado de exploración del algoritmo. De esta manera, se consigue aumentar sustancialmente la capacidad de identificar óptimos por parte del AM con un costo muy reducido en términos de tiempo computacional.

Por último, hemos comparado el algoritmo con una serie de técnicas existentes, obteniendo un comportamiento general significativamente mejor que todos ellos. Sin embargo, dado que el benchmark CEC'2013 implica evaluar el algoritmo para distintos valores de precisión, es interesante resaltar algunas conclusiones de las comparaciones:

- Con respecto a los algoritmos comparados, RMA-Archive es el que presenta el mejor comportamiento para un alto valor de precisión. De hecho, el grado de mejora que ofrece nuestro algoritmo sobre los otros aumenta conforme se incrementa el nivel de precisión exigido. La razón de ese rendimiento puede ser debido al uso intensivo del CMA-ES para refinar las soluciones más prometedoras.
- Al exigir un menor grado de precisión, que es cuando nuestro algoritmo presenta sus peores valores relativos al resto de algoritmos, no hemos detectado diferencias significativas entre RMA-Achive y el resto de propuestas.

Conclusiones y Trabajos futuros

En esta tesis se ha propuesto un nueva estrategia de nichos basada en regiones, dividiendo el espacio de búsqueda en hipercubos, cuyo tamaño se va reduciendo durante la ejecución del algoritmo. Mediante la incorporación de dicha estrategia hemos definido dos algoritmos meméticos, orientados a optimización global y a optimización multimodal. Tras comparar cn otros algoritmos de referencia hemos llegado a las siguientes conclusiones:

- El uso del mecanismo de nichos basados en regiones, al mantener mayor diversidad en la población, mejora significativamente el rendimiento del AM, tanto en optimización global, como en optimización multimodal.
- La modificación dinámica del tamaño de nicho otorga más robustez al algoritmo, al hacerle menos dependiente de dicho parámetro.
- Al comparar en optimización continua, nuestro resultado es altamente competitivo frente al resto, y escalable (aunque el método de BL puede limitar la escalabilidad del algoritmo).
- Para optimización multimodal es el que alcazaba mayor número de óptimos cuando se pedía una precisión, para menor precisión los resultados eran ligeramente peores, pero sin diferencias significativas detectadas.

Los prometedores resultados de ambos modelos animan la idea de seguir investigando en esta línea con vistas a reducir las debilidades identificadas, siguiente las siguientes líneas de investigación abiertas:

- Aplicar el modelo RMA-LSCh-CMA a problemas de alta dimensionalidad. Para ello deberías de modificar el método de BL empleado debido a las limitaciones detectadas de CMA-ES para los problemas de mayor dimensionalidad.

- El rendimiento de RMA-Archive puede ser mejorado, mediante una mejor exploración e identificación de las soluciones prometedoras que van a ser explotadas por la BL. Puede ser interesante evaluar el uso de otro AE en este específico framework. Por ejemplo, en las comparaciones del Capítulo 3, el algoritmo dADE obtenía mejor rendimiento que RMA-Archive para valores bajos de precisión. Por tanto, el uso de DE/nrand/1/bin como AE en vez del algoritmo genético podría ser una interesante alternativa.

- El uso de la estrategia de nichos basados en regiones se puede extender, mediante estudios adicionales.

  - Uno de los elementos que creemos más interesantes, aunque de mayor dificultad sería un mecanismo más avanzado de adaptación del tamaño del nicho. Esta línea podría incluir estudios sobre el modelo de división, e incluso criterios para determinar de forma auto-adaptativa cuándo aplicarse.
  - Hasta ahora se ha planteado un modelo de hipercubos de igual tamaño, una linea de trabajo futuro, de gran interés, sería la posibilidad de que las no todas las regiones se dividiesen de igual manera en regiones de menor tamaño, permitiendo la existencia de regiones de tamaño distinto. De esta manera, se podrían tratar de forma distinta las zonas más prometedoras.

Introduction

1 Framework: real-parameter optimisation

Continuous optimisation, also referred to as real-parameter optimisation, is a topic of major interest nowadays in the research community. The wide number of real-world applications and the variety in the difficulties and challenges they imply make the search for solutions and to solve them more and more fascinating.

The impossibility of performing an exhaustive search of all the possible solution implies the development and use of approximation algorithms. Metaheurstic algorithms [GK03] have been for the past few decades the most efficient and popular methods for this type of problems.

Metaheuristics can be classified into two categories:

- Global search algorithm (GS) [ES03] are usually population-based algorithms using collaborating agents representing different solutions spread across the search space. Their strength lies in their ability to explore the search space preventing the search to fall into local minima. For that reason, they are known to be very efficient in multimodal problems.
- Local Search methods (LS) [Sch81] are designed to rapidly reach precise solutions in the vicinity of a given solution. They usually consist in applying local modification to a solution until the nearby optimum is reached. As a consequence, their performance are notably good in unimodal problems. Their main drawback is they focus the search on a restricted area creating the risk of being trapped in local optima.

The challenging aspect of continuous optimisation problems lies in the multimodal characteristic posed in real-world problems. A problem is often referred to as multimodal when the landscape of its objective function contains multiple optima. This characteristic implies the necessity of designing optimisation algorithms offering both strong exploration and exploitation ability. The former aims at preventing the fall of the search into local optima while the latter aims at ensuring the most precise approximation of optima.

In order to combine those two aspects of an efficient search, arose memetic algorithms (MA) [Mos89]. They are the hybridisation of a GS algorithm providing its exploration ability and a LS method providing an efficient refinement of the promising solutions identified in the GS process. Such proposal however implies the establishment of an appropriate strategy to combine in the most efficient way those components. Thanks to the promising results they brought, MA quickly became an important topic of investigation.

The design of MA algorithm can be partitioned in the design of three components, the GS algorithm, the LS method and the hybridisation method. This last component is the essence of a MA. Its objective is to combine in the best possible way the strength of the first two components in order to achieve the most efficient search.

We believe in the importance of designing all three components in a way that limits the competition between the two search methods. In other words, the GS algorithm should be forced to explore the search space and offer to the LS method promising solutions for refinement. Niching strategies have been proposed to prevent the genetic drift of an EA's population by maintaining a high diversity between its solutions.

In this thesis, we propose, develop, and study the use of a novel niching strategy called region based niching strategy to enhance the performances of MA for continuous optimisation problems. MA using this strategy are referred to as region-based MA (RMA).

2 Objectives

In continuous optimisation, problems can be classified into two categories :

- Global optimisation problems : They consist in identifying the single global optimum of an objective function meaning that a single solution is considered as satisfactory for a given problem.
- Multimodal optimisation problems : In many real-world problems, multiple solutions can be considered as satisfactory. The particularity of multimodal optimisation lies in the need of identifying and preserving multiple global optima.

In both cases, a proper exploration of the search space is a key issue which requires special attention. In global optimisation, the risk of seeing the search converging towards local optima when dealing with multimodal problems is high and may considerably harm the performances of a search algorithm. In multimodal optimisation, the need of ensuring a strong exploration directly influences the ability of an algorithm to identify more or less global optima spread across the search space. In the latter case, other issues such as the preservation of the found optima are also at stake and are addressed in this thesis.

This thesis aims at using RMAs in the most efficient way in order to tackle both global and multimodal optimisation problems.

First, we wish to demonstrate how the use of a niching strategy can help the design of a memetic framework to clearly separate the exploration and exploitation efforts between the EA and the LS method for global optimisation. By doing so, we maintain a higher diversity in the population to ensure the proper exploration of the search space with the objective of leaving the least possible areas unexplored. The use of a niching strategy such as the one proposed offers such a possibility. Indeed, The EA performs an *inter-niche* search by focusing its effort in identifying the most promising niches while the LS method performs an *intra-niche* search by exploiting niches identified by the EA.

The second objective is to use the properties of the region definition of a niche in MAs for multimodal optimisation. The first objective in that matter is to assess the ability of the region-based niching strategy to ensure a proper exploration of the search space by the EA against classical euclidean of a niche. The second objective is to take advantage of this niche definition to create an index of explored and optimised regions in order to discard them from further exploration and to demonstrate how it can assist the exploration of the search space by the EA.

3 Summary

This thesis is divided into three chapters briefly described here.

In Chapter I, a review on real-parameter optimisation is presented. First by defining the main principles of this research area and then by listing and introducing the models that brought the most interest in the research community along the years. We particularly bring special attention to MAs and various niching strategies proposed to handle multimodal optimisation problems.

In Chapter II, we propose a method to efficiently include the region-based niching strategy in an existing MA for global optimisation. We study in this chapter the influence of the region-based niching strategy on the diversity of the EA's population and on the overall performances of the MA.

Chapter III presents how we took advantage of the definitions of a niche introduced here in a MA environment for multimodal optimisation problems. We present novel mechanisms allowing a more efficient exploration of the search space for the identification of multiple optima.

Finally, we conclude by summarizing the developed models, the experiments performed, and the results obtained in this study. Finalizing this thesis by stating a few lines of research opened after this work.

Chapter I

Real-Parameter Optimisation

1 Introduction

Many real world problems can be seen as optimisation problems over an objective function (also called fitness function) defined in a domain of solutions. The objective in optimisation is to find the solution that minimizes this function. This solution is called global optima and is traditionally noted x^*. Given a search domain D and an objective function $f : D \rightarrow R$, an optimisation problem can then be formulated by:

$$x^* = argmin_{x \in D} f(x) \tag{I.1}$$

In real-parameter optimisation a solution x in a set of D real values noted $x = \{x_1, x_2, ..., x_D\}$ where D is the dimensionality of the problem.

The nature of such problems making the evaluation of every solutions impossible, approximative algorithms have been developed to intelligently explore the search space of solution in the sake of the global optimum.

A wide variety of models have been proposed to tackle these problems. This chapter introduces the notion of meta-heuristics and more specifically Evolutionary and swarm algorithms. These algorithms have brought a major interest of the research community in the past years for their simplicity and efficiency in solving optimisation problems.

In the case of our study, we are particularly interested in MA. These models are the hybridisation of various algorithms aiming with the objective of combining their strength in a single framework. Usually, they combine a GS algorithm meant to explore the search space and a local search algorithm which intensively refines the search around a limited number of promising areas.

The need of combining those two aspects of the search is due to the need of identifying with the most precision the optimum solution of a problem by still avoiding the risk of being trapped in local optima.

The purpose of this chapter is to give the reader an overview of the research in the field of continuous optimisation in order to situate the different components used in this thesis in the research landscape in this area.

In Section 2, we provide an introduction to metaheuristics models that have been the most applied over the years with a special attention to GAs. In Section 3 we then focus our interest on MAs by explaining their strength, the issues paused by their design and the most remarkable proposals. Finally, in Section 4, we give an overview of the existing techniques for multimodal optimisation.

2 Metaheuristics for continuous optimisation

In this section, we review the most relevant metaheurstics proposed to tackle continuous optimisation problems. Their ability to efficiently tackle optimisation problems as black-box, *i.e.* no mathematical information is required, provide the EAs with an adaptability to a wide range of problems. In a first section, we provide an overview of some popular population-based algorithms. As we make use of GAs in the model presented in this thesis, we then focus on this family of algorithms.

2.1 Evolutionary and swarm algorithms

In this section we provide a quick introduction to the most relevant population-based algorithm:

- **Genetic Algorithms** (GA) [Hol75, Gol89]: use principles inspired by natural genetic populations to evolve solutions to problems. A population of chromosomes representing candidate solutions to the given problem evolves towards better solutions. From a population randomly initialised in the search domain, solutions compete to survive and participate in the creation of new solutions. New solutions are generated by means of crossover and mutation operators. More details on GAs are given in the following section.

- **Evolution Strategy** (ES) [Rec65]: The same way as GAs, ES apply evolutionary operators where the fittest solutions participate in the generation of new solutions. They sample new candidate solutions stochastically, most commonly from a multivariate normal probability distribution

From the early stages, evolution strategies were characterised by the use of real encoding as described above, mutation as the primary operator for generating new solutions, and self-adaptation of endogenous strategy parameters (mutation strengths) by embedding them into the encoding of solutions. Nowadays, the similarity between evolution strategies and real-coded genetic algorithms has allowed several researchers to combine their particular features into one unique evolutionary framework [MSV93b].

Hansen and Ostermeier [HO01] applied the idea of cumulation for the adaptation of covariance matrices needed to produce correlated mutations. The result was an evolutionary strategy model with a completely deterministic adaptation of its operators, which is called Covariance Matrix Adaptation-ES (CMA-ES).

Auger and Hansen proposed in [AH05] a CMA-ES variant (IPOP-CMA-ES) that reinitialised the search process, with an increased population size, when the progress had been detected to be minimal. IPOP-CMA-ES was the winner of the real-parameter optimisation competition, organised in the 2005 IEEE congress on evolutionary computation (CEC'2005). CMA-ES and its variants are now considered as state-of-the-art in real-parameter optimisation. A comprehensive overview of ES at that time can be seen in [BS02, HAA13].

- **Particle Swarm Optimisation** (PSO) [KE95, KE01] emulates the swarm behaviour of insects, animals herding, bird flocking, and fish schooling when these swarms search for food in a collaborative manner. Each member in the swarm adapts its search patterns by learning from its own experience and the ones of its neighbours. In particle swarm optimization, a member in the swarm, called a particle, represents a potential solution which is a point in the search space. At each iteration, particles adjust their flying direction according to the best experiences of the swarm and their own ones, and then, moves to the corresponding new position. [BVA07] provides a review of the knowledge of this field.

- **Differential Evolution** (DE) [SP97]: Differential evolution (DE) is a more recent evolutionary algorithm mainly focused on solving real-parameter optimisation problems [SP97]. Similarly to genetic and evolution strategies algorithms, it applies two operators for generation new solutions, mutation and crossover. However, its main characteristic is that mutation considers difference vectors of randomly chosen elements from the population, in contrast to classic mutation operators that took just one element as input. At each generation, mutation is applied for every element in the population producing a mutant vector; subsquently, crossover generates a trial vector by combining parameter values from the mutant vector and the element in the population; then, the best solution between the trial vector and the element in the population is selected for the next generation.

DE are are known to be very sensitive to two parameters, the crossover rate (CR), which defines the probability for a decision variable to me transmitted from the trial vector to the solution in the population, and the scale factor (F) which defines the amplitude of the mutation. To tackle many adaptive methods have been proposed to automatically tune the values of these parameters along the search and proposing new mutation operators [QS05, ZS09, BZB+09]. Das and Suganthan [DS11] provides a recent survey of the state-of-the-art.

Differential is probably the model receiving the greatest interest in the research community nowadays. In the last competition on real parameter optimisation during the IEEE Conference on Evolutionary Computation in 2013, 13 entrants out of 23 were variants of this model.

- Other models: Scatter Search [LM03], Artificial Bee Colony [KB07], Bacterial Foraging Optimization [LP02]

2.2 Genetic Algorithms

As this thesis makes use of GA we will introduce more in details this model. Introduced by Holland [Hol75], GA set the foundation of EA. They are inspired by the evolution theory of Darwin [Dar59]. Here, individuals represent solutions to a given problem. These solutions are represented by a string of variable encoding their genome. Within a given environment (objective function), they compete to survive.

A genetic algorithms starts off with a population of randomly generated solutions that evolves towards better solutions. During successive iterations, called generations, solutions are evaluated and maintained in the population through selection mechanisms to participate in the creation of new solutions by means of crossover and mutation operators.

Many adaptations, new operators and strategies have been proposed over the long life of GAs leading in a multitude of different aspects to take into account when designing a GA for continuous optimisation.

- **Encoding** : Original genetic algorithms for real parameter optimisation encoded the decision variables of the problems into strings of binary variables, by means of a binary or a gray transformation. The problem with this representation is the lack of precision as only a finite number of solutions can be encoded which can be a problem for problems with large search domains. Wright [Wri91] presented one of the first genetic algorithms using a real encoding as described above, which, introduced two new genetic operators: a mutation operator that uniformly sampled a new value for the parameter being mutated from an interval centered on its current value, and a linear crossover that produces the offspring according to particular linear combinations. Experimental results showed that the real-coded genetic algorithm gave superior results to binary-coded genetic algorithms on most of the test problems [HLV98]. Since then, most of the research uses this representation.

- **General strategy** : GAs can be classified into two categories, generational [Gol89] and steady state [Sch89]. Their difference occurs in the selection process. In the first one, new solutions are generated from a set of solutions selected from the population of the previous generation. In steady state GA (SSGA), new solutions are generated and inserted in the population one at a time. Different study shows that SSGA perform better [GD91, VF96]

- **Crossover operator**: To generate new solutions, the main operator is called crossover or recombination. It consists in "mixing" solutions called parents from the current population to obtain one or various new solutions. In [HLV98], the authors propose a comparative study of the different operators. Later in [HLS03], they categorized those operator identifying four families

  - Discrete Crossover Operators: the operators used in binary coding applicable to real-coded GAs (*e.g.* simple [Gol89], two-point [ECS89] and uniform crossover [Sys89])
  - Aggregation-based crossover operators: they use an aggregation function that numerically combines the values of the genes of the parents (*e.g.* Arthmetical [Mic94], geometrical [MNM96] and linear [Wri91] crossover)
  - Neighbourhood-based crossover operators: the genes of the offspring are determined by extracting values from intervals defined by neighbourhoods associated with the genes of the parents (*e.g.* BLX-α [ES93], simulated binary crossover [DA95])
  - Hybrid crossover operators: they combine operators from the previous categories. (*e.g.* Max-Min-arithmetical crossover [HLV97])

To those categories, we can also group crossover operators using multiple parents such as unimodal distribution crossover [OKK03], simplex crossover [TYH99], parent centric crossover [DAJ02], triangular crossover [ESE08] or, more recently, GA-MPC [ESE11] the winner of the CEC 2011 Competition on Testing Evolutionary Algorithms on Real World Optimization Problems.

- **Mutation operator**: Occurring only a limited amount of time consists, mutation randomly altering a solution in order to preserve a certain diversity in the population. In binary coding, the mutation consists in "flipping" a bit, meaning swapping it from "1" to "0" and vice-versa [Hol75]. Strategies proposed for real-coded GAs offer more variety. The same as for crossover operators, [HLV98] lists and compare some of them (boundary, uniform and non-uniform mutation [Mic94], Real Number Creep [Dav91], breeder genetic algorithm mutation (BGA) [MSV93a]).

- **Selection method**: A key aspect of genetic algorithm lies in the selection of the individuals used for recombination. Various strategies have been proposed, we list here the most popular:

  - Roulette wheel selection [DJ75]: The probability of an individual to be selected is calculated according to its fitness evaluation.
  - Linear [GB89] and exponential [BT97] ranking selection: Similarly to the previous the probability for an individual to be selected is calculated from its rank in the population using a linear or exponential function.
  - Tournament selection [Bri81]: Choose randomly t individuals (usually $t = 2$) from the population and keep the best individual of that group.
  - Truncation selection [MSV93a]: Only the T best individuals of the population can be selected with the same probability.
  - Non-random Mating [AFR01]: This family groups the methods that take into account similarity and/or parenthood in the selection process of parents. Usually a first parent is selected through classical strategies as the ones cited previously and the second parent is selected with respect to its relation with the first parent. For instance, in Negative Assortative Mating (NAM), the second parent is least similar individual taken from a subset of the population.

3 Memetic Algorithms

Memetic Algorithms are a specification of Memetic Computing (MC) [OLC10, COLT11, INM+12]. MC is the paradigm that uses the notion of memes. In general terms, memes are problem solvers. In MC, memes are included in a global framework allowing them to cooperate and/or compete in the problem solving.

MA can be considered as a sub class of MC. They are the union of population-based global search and local improvements which are inspired by Darwinian principles of natural evolution and Dawkins' notion of a meme [Daw90], defined as a unit of cultural evolution that is capable of local refinements.

In general terms, MAs are composed by a group of search algorithms cooperating and/or competing for the optimisation purpose. While this is usually accomplished by applying LS strategies to members of an EA's population, the MA paradigm also includes other kind of strategies such as the combinations of EAs with problem dependent heuristics, approximation algorithms, truncated exact methods, and specialised recombination operators [KS05, NC12].

The key issue when designing a MA and more generally speaking a MC is the interaction mechanism between the different search components. In [OLZW06], Ong et al. proposed a classification of adaptive MA:

1. Adaptive hyper-Heuristics : The coordination of the memes is performed by predetermined heursitic rules. For instance, in [KIP08] the evaluation budget is simply divided between each meme. [KSC02] proposes a reward-based approach where the the application of a given meme is reiterated until it stops being successful.

2. Meta-Larmarckian learning : these methods are probabilistic coordinator where the probability of each meme to be applied is based on its successes and failures history. [LOJS09][OK04] [NOL09]

3. Self-Adaptive and Co-Evolutionary : [Smi07] In self-apdaptive MA, solutions are encoded with their own genetic and memetic material. The memes are directly encoded in into the solution. For instance LS methods are associated with each solution and try to improve it.

4. Diversity Adaptation [CCN+07, NTM07b] : a diversity measure is used to to identify the diversity level of a population and to decide which meme to apply. A low diversity in the population will favour the use of an exploratory algorithm while a high diversity will prefer the application of a LS methods.

MA have been a hot topic in the field of optimisation. Many different instantiations of MAs have been reported across a wide variety of application domains that range from scheduling [CF07] and floor-planning problems [TY07], to extending wireless sensor network lifetime [TL10], aerodynamic design [RD00], vehicle routing [FPC08, MCG$^+$10], engineering control problems [NM10] and drug design [NTM07a], to name but a few. This large body of evidence has revealed that MAs not only converge to high-quality solutions, but also search vast, and sometimes noisy, solution spaces more efficiently than their conventional counterparts. Thus, MAs are the preferred methodology for many real-world applications, and nowadays receives more attention [OKI07, NC12].

MA presents many advantages which made this approach popular:

- The first main advantage of using MAs is based on the "*divide to conquer*" idea. MA separates the exploration effort from the exploitation effort in two components, the former being performed by ea EA, the latter by a LS algorithm. Not only this eases the development of each component, while classical EAs try to combine both aspects of an optimisation search in the same framework it also offers a better control on their functioning.

- By allowing an easy inclusion of problem knowledge, MAs are also considered as a guideline for addressing specific problems [NC12, BSEK06, Mos03]

- MAs have arisen as a promising approach for improving the convergence speed to the Pareto front of EAs for multiobjective optimisation problems, which actually concentrate increasing research efforts [IYM03],[LTGH07].

4 Multimodal optimisation

Multimodal optimisation is the discipline which consists in identifying multiple global (or satisfactory) optima in a fitness landscape. Research in this area has been focused in adapting global optimisation algorithms such as the ones introduced above in order to force them in exploring, exploiting and preserving distinct area of the search space.

In this section, we give an overview of the different ways and methods that have been used to alter of model search algorithm for this purpose. The main trend lies in the idea of maintaining a high diversity in the population in order to prevent its convergence toward a single optima. Such techniques are commonly called niching strategies referring to the technique used for the discovery and preservation of distinct niches. This term is a reference to the ecological concept of niches referring to the formation of distinct species exploiting different niches (resources) in an ecosystem.

We classified those techniques into two categories. The first one lists the classical niching strategies which mainly intervened at the replacement level of the EA they are applied to. The second one works with the idea of creating subgroups within the population which optimise in parallel different area of the search space by limiting the cooperation of each individual to its closest neighbours. We refer to them as neighbourhood based techniques.

In this section, we first describe the different elements composing those two categories by giving a general overview of proposal making use of such techniques. In a third section, we briefly introduce proposals combining those techniques with MA which demonstrate that the use of refinement method improved the performances of the original EA for multimodal optimisation.

4.1 Classical niching techniques

The first niching techniques consist in limiting the presence of multiple solutions within the same niche in order to maintain the high diversity in the population. When included in a classical EA, those mechanisms are mainly replacement strategies design with the objective eliminating solutions present in the same vicinity. We describe here the four major trends, With several examples of each one, to achieve this objective: crowding, clearing, fitness sharing and speciation.

Crowding

Crowding is one of the first techniques proposed to tackle multimodal optimisation problems [DJ75]. After the generation of a new solution, a random sample of CF solutions is selected in the population. The new solution competes with the closest solution of the sample to stay in the population. This methods main drawbacks is the definition of the parameter the crowding factor (CF). Small value can lead to the replacement of a distant solution to the offspring and thus a loss of information. CF should thus be choosen very large which however leads to an increased computational cost. The efficiency of this technique has however proven to be limited [Mah95] and advanced versions have been proposed:

- *Deterministic crowding* proposed by [Mah95] tries to limit the problem of replacement errors induced by the crowding technique by eliminating the need of defining the CF parameter. To do so, an offspring competes with its own parents to stay in the population.

- *Probabilistic crowding* [MG99] on the other hand modifies the replacement strategy of the original technique. In this scheme, the offspring and its most similar individual in the crowding sample compete in a probabilistic tournament where the probabilities of winning for each individual is calculated according to their fitness:

$$p(X) = \frac{f(X)}{f(X) + f(Y)} \tag{I.2}$$

  The idea is not to always prefer solutions with higher fitnesses which may lead to the loss of niches.

In [Tho04], Thomsen proposed the popular crowding differential evolution (CDE) applying a classical crowding strategy on a differential evolution (DE) where a new solution created by means of classical DE mutation and crossover scheme compared with its closest solution in the whole population for replacement.

CDE was then extended to multipopulation crowding DE (MCDE) in [Zah04] where multiple subpopulation evolve in parallel using CDE. When all the subpopulations have converged, the optima identified by each of them are stored in an archive and the subpopulations are reinitialised.

More recently, Qu et al. proposed the dynamic grouping of CDE (DGCDE) [QGS10] with ensemble of parameters. The population is divided into three subpopulation to which a set of control parameters is assigned.

In [QGZQ08] , Qing et al. proposed a Crowding Clustering Genetic Algorithm (CCGA) usind a clustering technique to eliminate the genetic drift introduced by the crowding strategy.

Clearing

Clearing techniques [Pet96] lie in the principle of dedicating the limited resources of a niche to its best individuals. The population is sorted according to the individual fitness value. The solutions are then selected one after the other and the solutions with worse fitness falling within their niche radius σ_{clear} are removed. Clearing has a low complexity and shows the best performances amongst the classical technique but is highly sensitive to the niche radius [SK98].

Variations have then been proposed to limit influence of the σ_{clear} parameter. For instance, in [QLSC12], similarly to the previously cited DGCDE, the authors propose an ensemble of clearing DE (ECLDE) in which the population was equaly divided into 3 subpopulations each evolving in parallel using a clearing DE with different values of σ_{clear}.

Some techniques use a redefinition of the niche in order to simply remove the use of the parameter σ_{clear}. In [EO09], the niches are defined through hill-valley detection mechanism instead of using a niche radius. In [SMPS04], the niches are defined by fuzzy clustering of the solutions of the populations.

Fitness sharing

Contrarily to clearing technique which consist in dedicating niche resources to a single solution, fitness sharing [GR87] consists in sharing them to multiple solution. This concept is modelled by reducing the fitness of a individuals present in densely populated regions. As a consequence, the shared fitness of the *ith* individual is:

$$f_{shared}(i) = \frac{f_{original}(i)}{\sum_{j=1}^{NP} sh(d_{ij})} \tag{I.3}$$

where the sharing function is calculated by:

$$sh(d_{ij}) = \begin{cases} 1 - \left(\frac{d_{ij}}{\sigma_{share}}\right)^{\alpha}, & \text{if } d_{ij} < \sigma_{share} \\ 0, & \text{otherwise} \end{cases} \tag{I.4}$$

where d_{ij} is the distance between individual i and j, σ_{share} is the sharing radius and α is a constant called sharing level.

In [Tho04], Thomasen also proposed a DE using sharing where, after each generation, the new shared fitnesses are calculated over the population individuals and the trial vectors, the best half being kept in the population.

Speciation

Proposed in [LBPC02], speciation or species conservation introduces the notion of species by separating into several groups (species) according to their similarity. Those species are identified by a dominating individual called the species seed and a species distance $\sigma_{species}$ defining the maximum distance between two individual of the same species. The set of species seed is build at each generation by iteratively adding individuals from the population that are further from any species seed than $\sigma_{species}/2$. The individuals are kept from generation to another until a better solution is identified within their species while the classical recombination operators are applied.

In [Li04], this concept is applied in a spciation-based PSO (SPSO). In SPSO, the particles are gathered into species to form subpopulations. This proposal was later extended to reduce its dependency to the species distance parameter by using population statistics [BL06b] and a time-based convergence measure [BL06a].

4.2 Neighborhood based technique

Another class of niching strategies can be referred to as neighbourhood-based. Contrarily to the previous section where the niching strategy could be seen as replacement strategy, these methods use the geographical information of the solutions in a population to modify the recombination scheme of a given EA. The main idea is to make solutions solely cooperate with their neighbours in order to emphasize the speciation.

Originally named spatially-structured EAs (SSEA) [Tom05], these algorithms form subpopulations of individuals (called here deme) based on their similarity and perform genetic operation within each deme.

This idea has then been extended and two kinds of neighbourhoods can be identified in the literature:

- *Index-based neighborhood* [Li10] uses the the indices in the population of a PSO to identify the neighborhood of a solution. The velocity of a particle is thus influenced by the local best solution instead of the global best.

- *Distance-based neighborhood* uses the euclidean distance between individuals. In [Li07], the author proposed the FER-PSO algorithm where particles are attracted towards the "fittest-and-closest" neighbours. Similarly, the notion of neighbourhood is applied for DE in [EPV11]. A new mutation strategy, DE/nrand/x is proposed. It uses as base vector the nearest neighbor of each individual. This mutation strategy has then be used for more advanced models like in [ELB13].

Neighbourhood-based strategies have often been coupled with classical niching strategies. For instance in [DW11], Dick proposed to include in a SSEA a fitness sharing and a clearing strategy.

In [QSL12], the authors use of the DE/nrand/x operator with crowding, sharing and species-based niching strategies and obtain better results than the original algorithms.

4.3 Memetic algorithms for multimodal optimisation

As stated in the introduction, MA are the hybridization of an EA and an LS method. This model is particularly adapted to multimodal optimisation problems as, when applied to different solutions, an LS method can offer a strong refinement of the promising solutions discovered by the EA. This offer great accuracy for the identification multiple optima. The use of such model has raised interest in the research community.

For instance, the Sequential Niching Memetic Algorithm (SNMA) proposed by Vitela et al. in [VC08] and then extended in [VC12] is an MA which combines a genetic algorithm (GA) with a gradient-based LS method. Before each generation, the LS is applied to each solution of the population.

In [QLS12], Qu et al. included an LS method to various previoulsy cited PSO for multimodal optimisation (FER-PSO, SPSO, rPSO). The LS method used consisted in generating at each iteration a new solutions in the neighbourhood of the personal best of each particle to explore its surrounding. They demonstrated that the resulting memetic PSO obtained better results than the original algorithms. Similarly, Wang et al. proposed a memetic SPSO [WMYW12] which adaptively uses two different LS methods and came to the same conclusions.

Chapter II

Region based memetic algorithm with local search chaining for real-parameter optimisation

1 Introduction

One of the main issues when designing an evolutionary algorithm (EA) [BFM97] for real-coded parameter optimisation problems is to offer a good exploration of the search space and, at the same time, to exploit the most promising regions to obtain high quality solutions. Memetic algorithms (MA) were proposed [Mos89] to manage these competing objectives. They are a hybridisation between EA and local search (LS) algorithms, mixing in one model the exploration power of EA and the exploitative power of the LS. MAs are characterised by the combination of an exploration algorithm and a local improvement algorithm.

MAs with an appropriate trade-off between the exploration and exploitation can obtain accurate solutions, improving the search [Dav91, GV99]. Therefore, the key issue when designing a MA is to organise both efforts in the most cooperative way.

Niching strategies have been used in EA to either identify various optima in a fitness landscape or to maintain a strong diversity in the EA's population [DMQS11]. In our study, we consider that using niching strategy to maintain a higher diversity in the population leads to a better separation of the effort between the EA and the LS.

In this chapter, we design a niching strategy to limit the competition between the EA and the LS in a MA. The purpose of this method is to let the

EA focus on the exploration task by limiting its exploitation power, this task being more efficiently performed by the LS method. Contrarily to most niching strategies where the niches are defined around the solutions of the population, the niches are predefined as divisions of the search space. The search space is divided into equal hypercubes each of which represent one exclusion region, not allowing more than one solution in each one. Also, the LS method is initialised to explore inside these regions. This way, there is no competition between the EA and the LS method. In order to obtain a more robust strategy, we also propose a version with a dynamic niche size. It consists in decreasing the niche size along the search to have a great diversity in the early stage of the search and reduce it along the process.

To assess the efficiency of this strategy, we implement it in the MA-LSCh-CMA [MLGMH10]. MA-LSCh-CMA is a successful MA which originality lies in its ability to apply various times the LS on the same solution. Although this algorithm obtains good results, it lacks a diversity control mechanism and does not limits the competition between the EA and the LS efforts. The association of the MA-LSCh-CMA algorithm with the niching strategy proposed gives the algorithm called Region-based MA-LSCh-CMA (RMA-LSCh-CMA).

Various studies are performed to demonstrate the influence of the niching strategy on the diversity of the EA's population and the improvements brought to the original model.

The proposed algorithm is then automatically configured using IRACE [LIDLSB11] for comparisons with various state-of-the-art algorithms.

This chapter is structured as follows. Section 2 is dedicated to explain the MA-LSCh-CMA we used as case study here. In Section 3, we describe in detail the new proposal, remarking the differences with the previous model. In Section 4, the experimental framework is designed. In Section 5, we show the results and analysis of the different studies and comparisons carried out. Finally, in Section 6, we present the conclusions and future works.

2 The MA-LSCh-CMA

This section describes the general scheme of the memetic algoritm with local search chaingin and CMA-ES (MA-LSCh-CMA) and its main components. More details can be seen in [MLGMH10].

2.1 General scheme

MA-LSCh-CMA was designed with the idea that the LS should be applied with higher intensity on the most promising regions. By promising regions, we consider the areas/regions where the solutions are maintained the most time in the population for their good fitness.

The MA-LSCh-CMA is a steady state MA which alternatively applies a Steady-State Genetic Algorithm(SSGA) as EA [Sch89], and a CMA-ES [HMK03] as LS method with an I_{str}. This hybridisation model allows the same solution improve several times, creating *LS chain*. Also, it uses a mechanism to store with the solution the final state of the LS parameters after each LS application. This way, the final state of a LS application over a solution will be used as the initial point of a subsequent LS application over the same solution, *continuing* the LS. The general scheme can be seen in Algorithm 1.

Algorithm 1 Pseudocode of MA-LSCh-CMA

1: Generate the initial population
2: **while** not termination-condition **do**
3: Perform the SSGA with n_{frec} evaluations
4: Build the set S_{LS} of individuals which can be refined by LS
5: Pick the best individual c_{LS} in S_{LS}
6: **if** c_{LS} belongs to an existing LS chain **then**
7: Initialise the LS operator with the LS state stored with c_{LS}
8: **else**
9: Initialise the LS operator with the default LS parameters
10: **end if**
11: Apply the LS algorithm to c_{LS} with I_{str} evaluations, giving c^r_{LS}
12: Replace c_{LS} by c^r_{LS}
13: Store the final LS state with c^r_{LS}
14: **end while**

To select the individual c_{LS} to which the LS will be applied, the following process is used (steps 4 and 5 in Algorithm 1):

1. The set S_{LS} is build with the individuals of the population that:

   (a) have never been improved by the LS.

   (b) have been improved by the LS but with an improvement (in fitness) superior to δ_{LS}^{min}.

2. If $|S_{LS}| \neq 0$, the LS is applied on the best individual in S_{LS}. If S_{LS} is empty, the whole population is reinitialised except for the best individual which is maintained in the population.

With this mechanism, if SSGA obtains a next best solution, it should be improved by the LS in the following application of the LS method.

2.2 The EA

The SSGA applied was specifically designed to promote high population diversity levels by means of the combination of the $BLX-\alpha$ crossover operator [ES93] with a high value for its associated parameter ($\alpha = 0.5$) and the *negative assortative mating* strategy (NAM) [AFR01]. Diversity is favoured as well by means of the BGA mutation operator [MSV93a]. The replacement strategy used is *Replacement Worst, RW*. The combination *NAM-RW* produces a high selective pressure. The SSGA is described in Algorithm 2.

Algorithm 2 Pseudo-code for the SSGA

```
Randomly generate the population
while not termination-condition do
    Select two parents in the population using the NAM strategy
    Create an offspring c_n using BLX − α crossover and BGA mutation
    Replace the worst individual c_worst in the population if f(c_worst) > f(c_n)
end while
```

2.3 The LS

The continuous LS algorithm is CMA-ES [HMK03]. This algorithm is the *state-of-the-art* in continuous optimisation. Thanks to the adaptability of its parameters, its convergence is very fast and obtains very good results. CMA-ES is an algorithm that uses a distribution function to obtain new solutions, and adapt the distribution around the best created solutions.

The only required parameters are the initial average of the distribution $\vec{m}$ and the initial standard deviation σ. MA-LSCh-CMA sets the individual to optimise c_{LS} as $\vec{m}$, and as the initial σ value the half of the distance of c_{LS} to its nearest neighbour in the EA's population. In our proposal, this initialisation strategy is modified to focus the LS in the regions.

3 Region based memetic algorithms

This section presents the basic concepts of the novel rigid niching strategy and explains how we included it in the MA-LSCh-CMA.

Niching strategies consist in creating an area around the solutions of an EA's population where no other solution can be present. The main purpose of this tool is to maintain the diversity of the population at a higher level. Maintaining the diversity in a population prevents a fast convergence of the population and allows a better exploration of the search space. This notion is particularly interesting in MA as an EA's first task is to explore, the exploitation of the solutions being done by the LS method. In other words, throw this strategy, we offer a clearer separation between the exploration effort done by the EA and the exploitation task of the LS method.

In Section 3.1, we describe the proposed niching strategy. Including such niching strategy implied two major modifications in the MA-LSCh-CMA, the redefinition of the EA, explained in Section 3.2 and the initial parameters of the LS explained in Section 3.3. Finally, we explain the scheme of the dynamic model in Section 3.4. We named the resulting algorithm RMA-LSCh-CMA.

3.1 Basic concepts

Contrarily to most niching strategies where the niches are defined by the area surrounding solutions of the population, we propose here a strategy in which the niches are predefined as divisions of the search space, divided into hypercubes of equal size called here regions. This definition of a niche is illustrated in Figure 1. Each dimension is divided into ND divisions creating a grid of equal hypercubes, that represent olution.

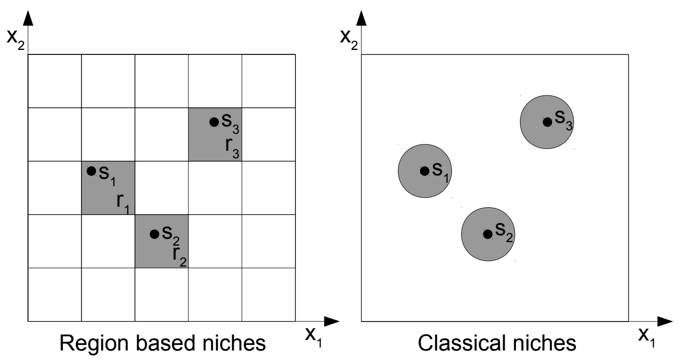

Figure 1: Different niching strategy

3.2 The SSGA in a region-based MA

One of the key issues in niching strategies is to decide what to do with a solution generated in the exclusion area of an other solution. The modifications to the SSGA are described in Algorithm 3. It consists in not allowing the generation of a solution by the SSGA in a region that is already occupied by an optimised solution of the population. By optimised, we refer to the fact that it was previously applied the LS over this solution, and the last LS applied has not brought enough improvements (upper than δ_{LS}^{min}). Then, if a solution is optimised, we consider its neighbourhood (and by consequence the region it lies in) has sufficiently been explored. On the other hand, if the solution is not optimised, the EA can replace it with a solution with a better fitness in that region. That way, we avoid unnecessary LS evaluations within the region to get a higher quality solutions. This way, we ensure that the population does not hold two solutions in the same region.

Algorithm 3 Pseudo-code for the region-based SSGA

Randomly generate the population
while not termination-condition **do**
  Select two parents in the population
  Create an offspring c_n using crossover and mutation
  if c_n falls in a region containing an individual c_o **then**
    if c_o is considered optimised **then**
      Mutate c_n using the BGA mutation and go back to 5
    end if
  end if
  if c_n falls in a region containing an individual c_o **then**
    Replace c_o by c_n if $f(c_o) > f(c_n)$
  else
    Replace the worst individual c_{worst} in the population if $f(c_{worst}) > f(c_n)$
  end if
end while

3.3 The LS in a region-based MA

In order to put the emphasis on dedicating the exploration task to the EA and the exploitation one to the LS, we have also modified the strategy for initialising the parameters of the LS. In the MA-LSCh-CMA, the initial step of the CMA-ES is set between the area limited by its neighbouring solutions. Here the CMA-ES initial step is set according to the size of the region. We want to ensure that the close surrounding of a solution are properly explored by the LS as this task will not be done by the EA. The initial standard deviation is set to half the size of the region. Apart from this modification, in order to allow a proper refinement of the solution, the LS is not influenced by the divisions of the search space. However, if at the end of the LS application, the new solution is in a region occupied, the best solution is kept and the other one is replaced by a randomly generated solution.

3.4 A dynamic number of divisions

One of the main issues when implementing a niching strategy is to define the size of the niche. It is also the case in this model and it represents the most critical parameter. Here, the size of the region is directly dependent on the number of divisions per dimensions ND.

A high number of divisions leads to smaller niches and thus, a poor influence on the search. On the other hand a small number of divisions creates big niches. The diversity will be high but the chances that the local search fails to reach the best solution in its surroundings are higher.

These reasons shows that the niche size can limit the effectiveness of the search. This motivates the use of a dynamic niche size in order to achieve a better robustness of the algorithm. To do so, the number of divisions is increased along the search. With bigger regions at the beginning of the search, a greater diversity is maintained to ensure a strong exploration of the search space. The number of division is then increased in order to allow a better convergence in later stages of the process.

We have decided to use a linear increase of the number of division. At each update, $ND_i = 2 \cdot ND_{i-1}$. If the total number of updates is u, an update occurs every $max\_eval/(u+1)$ where max_eval is the maximum number of fitness evaluation allowed. With this strategy, two parameters appear, ND_0, the initial number of divisions and u, the number of updates.

4 Experimental framework

We have carried out different experiments to assess the performances of the region-based niching strategy. In this section, we describe the test suites used. For its low and median dimensions, the benchmark proposed in the Special Session on Real Parameter Optimisation organised in the 2005 IEEE Congress on Evolutionary Computation (CEC'2005)[SHL+05] is the basis of the study of the model and is used for all the experiments. On the other hand, another benchmark, the Soft Computing Special Issue on Large Scale Continuous Optimisation benchmark (SOCO'2011) [LMH11] is used for offering results in a higher dimension.

Section 4.1 and 4.2 respectively describe the CEC'2005 and SOCO'2011 benchmark. Section 4.3 lists the statistical tests used for the comparisons.

4.1 The 2005 IEEE Congress on Evolutionary Computation benchmark

For the experimental sections, we have used the benchmark proposed in the CEC'2005. The complete description of the functions can be seen in [SHL$^+$05]. Table II.1 lists those functions. The first five are unimodal functions (F1-F5), followed by seven basic multimodal (F6-F12), two expanded (F13-F14) and 11 hybrid composition functions (F15-F25). Those last ones are compositions of the twelve first. Note that every functions have been shifted to ensure that the global optimum is not in the center of search space. In F7 and F25, the optima are out of the ranges of initialisation. These functions have been implemented in dimension $D = 10, 30, 50$.

Table II.1: Test functions of the CEC'2005 benchmark

F1	Sphere function
F2	Schwefel's problem 1.2
F3	Rotated High Conditioned Elliptic Function
F4	Schwefel's Problem 1.2 with Noise in Fitness
F5	Unimodal function
F6	Rosenbrock's function
F7	Griewank's function
F8	Ackley's function
F9	Rastrigin's function
F10	Rotated Rastrigin's function
F11	Rotated Weiestrass' function
F12	Schwefel's Problem 2.13
F13-F14	Expanded functions
F15-F25	Hybrid composition function

In order to be able to compare our results with other algorithms involved in the competition, we followed the requirements described in [SHL$^+$05] :

- Each algorithm is run 25 times for each test function, and the average of error of the best individual of the population is computed. The *function error* value for a solution x is defined as $(f(x) - f(x^*))$, where x^* is the global optimum of the function.

- The study has been made with dimensions $D = 10$, $D = 30$, and $D = 50$.

- The maximum number of fitness evaluations for each run is $10,000 \cdot D$, where D is the dimension of the problem.

- Each run stops either when the error obtained is less than 10^{-8}, or when the maximal number of evaluations is achieved.

4.2 The Soft Computing Special Issue on Large Scale Continuous Optimisation Problems

To assess the scalability of our model against the state-of-the-art, we used the SOCO'2011 benchmark [LMH11]. Table II.2 lists those function. It is composed of 19 shifted functions, 11 basic functions (F1-F11) and 8 hybrid composition functions (F12-F19) which are non-separable functions built by combining two of the 11 first functions.

This benchmark has been implemented in 4 different dimensions 50, 100, 500 and 1000. In our experiment we will only work with dimension 100, as dimension 50 is already assessed in the CEC'2005 benchmark, and higher dimensions are outside the scope of this paper.

As for the CEC'2005 benchmark, each algorithm is run 25 and the mean error is kept. The maximum number of fitness evaluations for each run is $5,000 \cdot D$

4.3 Statistical Tests

Non-parametric tests must be used for comparing the results of different search algorithms for this benchmark [DGMH11]. Given that the non-parametric tests do not require explicit conditions for being conducted, it is recommendable that the sample of results would be obtained following the same criterion to compute the same aggregation (average, mode, etc.) over the same number of runs for each algorithm and problem. We use the program available in http://sci2s.ugr.es/sicidm/

In particular, we have considered two alternative methods based on non parametric tests to analyse the experimental results:

Table II.2: Test functions of the SOCO'2011 benchmark

F1	sphere function
F2	Schwefel's problem 2.21
F3	Rosenbrock's function
F4	Rastrigin's function
F5	Griewank's function
F6	Ackley's function
F7	Schwefel's problem 2.22
F8	Schwefel's problem 1.2
F9	Extended F10
F10	Bohachevsky
F11	Schaffer's function
F12-F19	Hybrid composition function

- Application of the Iman and Davenport's test and the Holm's method as post-hoc procedure. The first test may be used to see whether there are significant statistical differences among the algorithms on a certain group of test algorithms. If differences are detected, then Holm's test is employed to compare the best algorithm (control algorithm) against the remaining ones.

- Application of the Wilcoxon matched-pairs signed-ranks test. With this test, the results of two algorithms may be directly compared.

5 Experimental Results

We have carried out the experiments of RMA-LSCh-CMA using the parameters' values proposed by the authors of MA-LSCh-CMA [MLGMH10], except for the population size which was set to 60 in the previous model:

- the population size is 80
- the pool size for the NAM selection $N_{NAM} = 3$
- the mutation probability $P_{mutation} = 0.125$
- the number of evaluation allocated to each LS $I_{str} = 500$
- the LS/EA ratio $R_{LS} = 0.5$.

In Section 5.1, we first experiment the influence of the number of divisions. From the observation made, we then justify the use of a using a dynamic number of divisions in Section 5.2. In Section 5.3, we compare the results of our model against the original one, MA-LSCh-CMA and illustrate the influence of the niching strategy on the diversity of the population. Finally, in Section 5.4, we tune the parameters using IRACE [LIDLSB11] over the CEC'2005 benchmark functions to compare the performances of the RMA-LSCh-CMA against a sample of representative algorithms in Section 5.5.

5.1 Study of the number of divisions

When implementing a niching strategy, the most critical parameter is the size of the niches. In this section, we assess the influence of the number of divisions on the results and the diversity of the population. We tested three fixed values of ND: 10, 50 and 100.

Results on the CEC'2005 benchmark

We present here the results obtained by the RMA-LSCh-CMA with different values of ND. Detailed results can be seen in Appendix A.1.

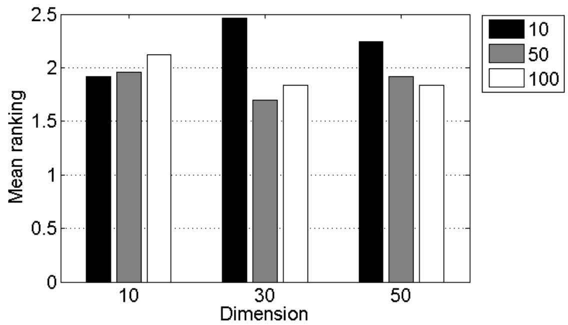

Figure 2: Mean rankings obtained by RMA-LSCh-CMA with different number of divisions over every functions of the CEC'2005 benchmark. The lower columns corresponds to the best algorithms.

Figure 2 shows the average rankings obtained by the RMA-LSCh-CMA instances with different ND values on the 25 test functions with dimensions $D = 10$, 30, and 50. The mean rankings correspond to the average of the ranking of each algorithm on each function. We can note that the influence on the number of divisions depends on the dimension. Indeed, for smaller dimensions, a smaller number of divisions obtains better results while a higher number of divisions performs better on higher dimensions.

Table II.3: Iman-Davenport test for significant difference between the instances of R-MA-LSCh-CMA with $ND = 10$, $ND = 50$ and $ND = 100$

Dimension	p-value	Significant differences?
10	0.763	No
30	0.014	Yes
50	0.333	No

We first applied the Iman-Davenport's test to the results of the three instances of the model to assess any significant differences. Table II.3 shows that there are no significant differences in dimension 10 and 50. In dimension 30, we can observe significant differences, thus we apply the Holm's test using the algorithm with best fitness, ND=50, as the *control algorithm*. Table II.4 show the results. It can be observed that ND=50 gives significantly better results than with ND=10 and they are fairly equivalent with ND=100.

Table II.4: Comparison using Holm's test with $\alpha = 0.05$ of the instance where $ND = 50$ against the other instances

i	ND	$z = (R_0 - R_i)/SE$	p	α/i	Significant difference?
2	10	2.687	0.007	0.025	Yes
1	100	0.495	0.621	0.05	No

Diversity study

This section aims to demonstrate the influence of the number of divisions on the diversity of the population. We analyse the evolution of the diversity along the search of RMA-LSCh-CMA with fixed values of ND using the *distance-to-average-point* measure as described in [Urs02]. It consists in calculating the mean distance of each individual in the population to the average point of the population:

$$diversity(P) = \frac{1}{|L| \cdot |P|} \cdot \sum_{i=1}^{|P|} \sqrt{\sum_{j=1}^{N} (s_{ij} - \overline{s}_j)^2} \tag{II.1}$$

where $|L|$ is the length of the diagonal in the search space $S \subseteq \Re^N$, P the population , $|P|$ the population size, N the dimensionality of the problem, c_{ij} the j^{th} value of the individual i and $\overline{s}_j$ the j^{th} value of the average point of the population $\overline{s}$.

To perform this study, we ran an experiment over 25 runs in each function in dimension 10 measuring after the each pair of SSGA and LS is launched (each of which running through 500 evaluations, the diversity is calculated every approximately 1000 function evaluations).

Figure 3 represents the evolution of the population's diversity for each instance of the RMA-LSCh-CMA.

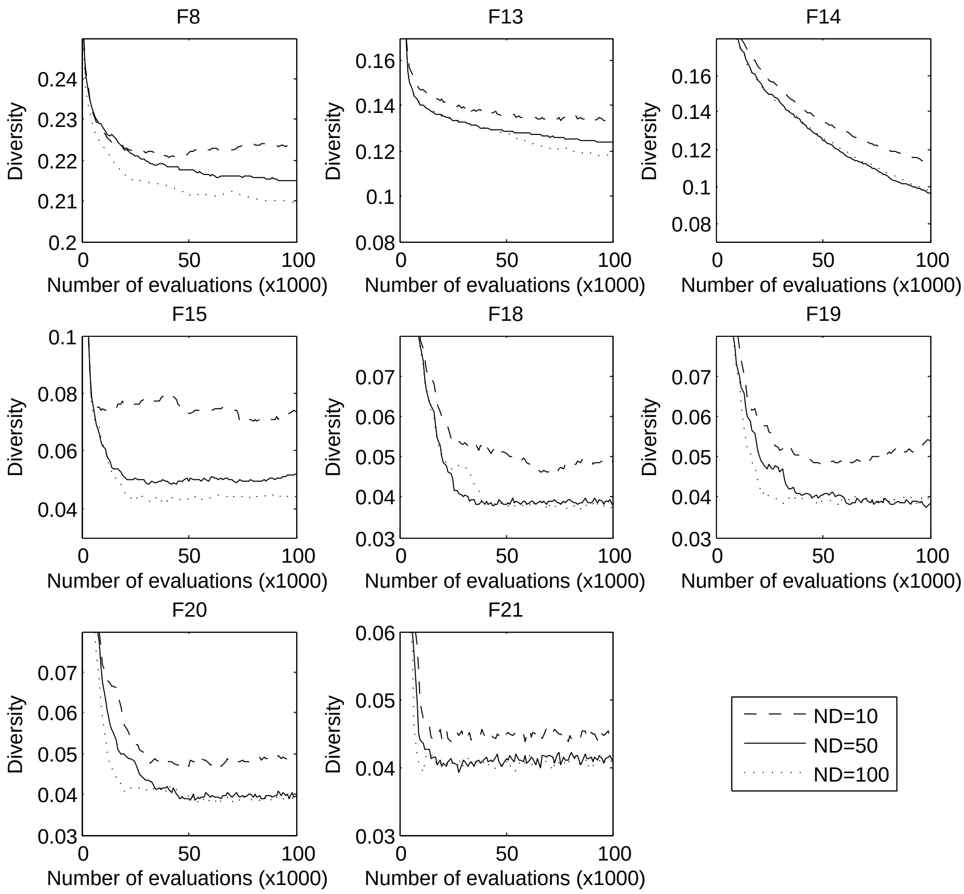

Figure 3: Evolution of the diversity in the population for different number of divisions for different functions

As it was expected, the number of divisions influences the diversity. Indeed, the smaller the number of divisions, the higher is the diversity in the population remains along the search.

5.2 Dynamic vs static number of divisions

The previous section showed that the choice of the number of divisions depended on the dimension. In this section we thus assess the use of a dynamic number of divisions. In this experiment, we chose to perform 3 updates during the search, $u = 3$, with an initial number of divisions $ND_0 = 10$, giving a sequence of divisions number of 10, 20, 40 and 80. In table II.5 we compare the dynamic model with the static one with different values of ND.

We saw in the previous section that setting a static number of divisions influenced the results according to the dimensionality of the problem. This influence is reduced by using a dynamic number of divisions. Indeed, we can note that while, when comparing the dynamic with a static high number of divisions, both strategies are statistically equivalent in higher dimensions ($D = 30$ and $D = 50$), the dynamic model obtains better results in smaller dimensions ($D = 10$). On the other hand, when comparing it with a static small number of divisions ($ND = 10$), we obtain better performances in higher dimensions (statistically better for $D = 30$).

Table II.5: Dynamic region based MA-LSCh-CMA versus various static numbers of divisions using Wilcoxon's test

Dimension	ND	R+ Dynamic	R- Static	$p - value$
10	10	222.5	102.5	0.110
10	50	264.5	60.5	0.005
10	100	250	53.5	0.005
30	10	216.5	86	0.069
30	50	177	148	0.696
30	100	135.5	167	0.679
50	10	181.5	121	0.407
50	50	169.5	155.5	0.851
50	100	152.5	150	1.000

For the following experiments, we will use the dynamic version of the algorithm.

5.3 Comparison with the MA-LSCh-CMA

The original purpose of this work was to improve the promising results of the MA-LSCh-CMA. We analyse in this section the improvements brought by the proposed niching strategy to this algorithm and its influence on the diversity of the population using the same parameters in both algorithms.

Results on the CEC'2005 benchmark

Table II.6: Wilcoxon signed rank test results of RMA-LSCh-CMA vs MA-LSCh-CMA

Dim	$R+$ RMA-LSCh-CMA	$R-$ MA-LSCh-CMA	$p-value$
10	247	78	0.022
30	202.5	99	0.152
50	211.5	90	0.089
All Dim	1927.5	854	0.004

The detailed results can be seen in Appendix A.2. Table II.6 shows the Wilcoxon signed rank obtained when comparing both algorithms. We can see that the niching strategy, and the modifications it implies, improves the results in every dimensions. The results are statistically better in dimension 10, with $\alpha = 0.05$, and in dimension 50 with $\alpha = 0.1$

Diversity study

We demonstrate in this section that the implementation of this niching strategy actually influences the diversity of the population. The evolution of the diversity on various functions is plotted in Figure 4 following the conditions described Section 5.1.

We can see that the diversity remains higher in the population of the RMA-LSCh-CMA.

5.4 Automatic configuration

In the previous section, we demonstrated that the use of the region-based niching strategy in the LS chaining framework significantly improved the performances

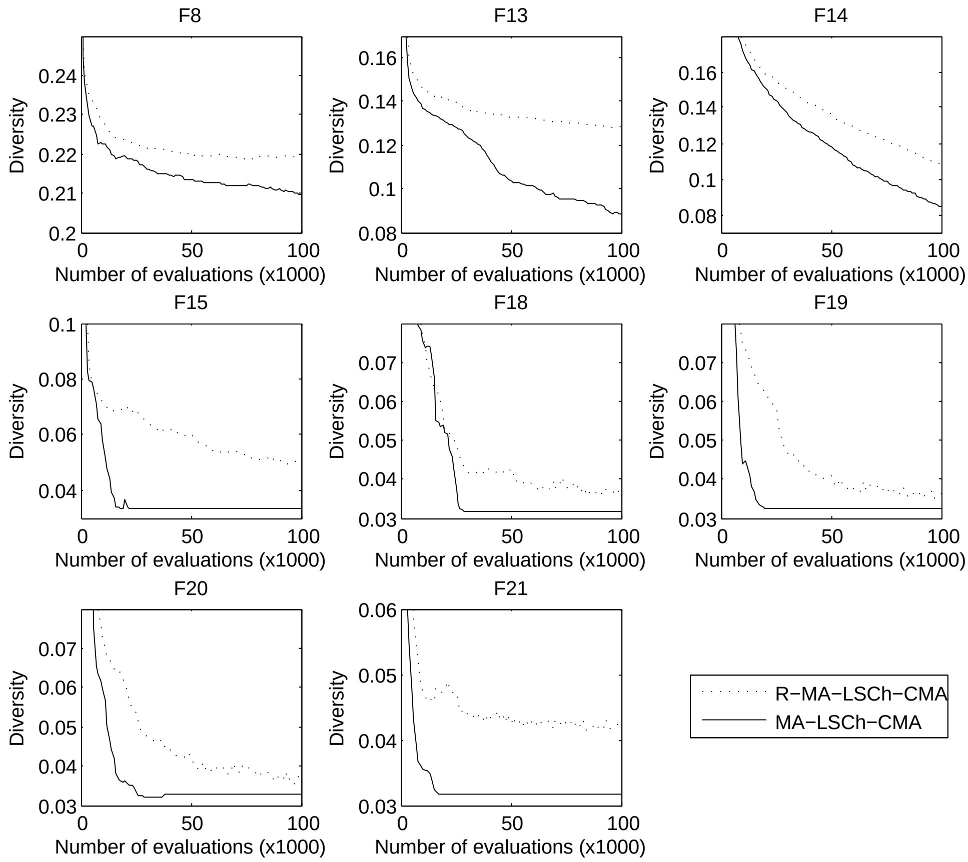

Figure 4: Evolution of the diversity in the population of the EA of the RMA-LSCh-CMA and MA-LSCh-CMA for different functions

of this MA. However, in order to fully adapt the design of this new model to the problems at hand, we applied the automatic tuning of its parameters using the automatic configuration tool IRACE [LIDLSB11].

IRACE package

The automatic configuration tool that we use is the IRACE package. Based on previous works [BBS07, Bir04, Bir09, BSPV02, BYBS10], it implements an automatic configuration approach based on *racing* [MM97]. Statistical tests are used to test for significantly inferior candidate configurations. The IRACE package, implemented as an R [R Development Core Team08] package, implements a general *iterated racing* procedure. For more details on this tool, the reader can refer to [LIDLSB11].

The IRACE package has already been extensively tested in several research projects, leading to successful improvement over the state-of-the-art, see for instance [DLLIS11b, DLLIS11a].

The advantage of this tool is that it handles several parameter types: continuous, integer, categorical, and ordered. Continuous and integer parameters take values within a range specified by the user. Categorical parameters can take any value among a set of possible ones explicitly given by the user. An ordered parameter is a categorical parameter with a pre-defined strict order of its possible values. We also relied on its capability to parallelize the configuration phase in order to reduce considerably the amount of time required for it.

Application to the RMA-LSCh-CMA

We selected a set of parameters that we considered the most critical to be tuned. Those parameters are listed in Table II.7 along with the ranges of search, their default values and their obtained values after tuning. The tuning budget allocated to IRACE is set to 5000. The budget corresponds to number of runs in the conditions defined by the benchmark that irace uses to perform the tuning.

In Table II.8, we compare the results of the RMA-LSCh-CMA with default parameters and the ones obtained by tuning. The automatic configuration brings significant overall improvements to the model and more specifically in dimension 10 and 50.

For the following experiment, we use the parameters obtained by tuning and listed in Table II.7

Table II.7: Parameters tuned and obtained values

Parameters	Descriptions	Ranges	Default	Tuned
I_{str}	LS intensity, number of evaluations allocated to each LS application	$[100, 1000]$	500	950
ND_0	Initial number of divisions, defines the size of the niches/regions	$[2, 10]$	10	6
u	Number of update to be performed	$[2, 5]$	3	2
m_u	Update multiplier	$[1, 5]$	2	4
$r_{EA/LS}$	The repartition of the overall effort between the EA and the LS the higher the value the more evaluations allocated to the LS	$[0.1, 0.9]$	0.5	0.6
NP	Population size of the EA	$[40, 120]$	80	40
λ	Parameter to define the CMA-ES population size $p = 4 + \lambda ln(D)$	$[1, 10]$	3	8
μ	Defines the parent size for the CMA-ES p/μ	$[1, 5]$	2	4
α	Parameter for the $BLX - \alpha$ crossover	$[0.1, 0.9]$	0.5	0.6

Table II.8: Wilcoxon signed rank test between the default version of RMA-LSCh-CMA ($R-$) and its tuned version ($R+$)

Dim	Tuned $R+$	Default $R-$	$p - value$
10	241.5	62	0.011
30	160	142.5	0.830
50	234	68.5	0.019
All Dimensions	1951	832.5	0.003

5.5 Comparison with other algorithms

In this section, we compare the efficiency of our algorithm with IPOP-CMA-ES [AH05], MDE_pBX [IDG+12] and 3SOME [INM+12] :

- IPOP-CMA-ES is the winner of the CEC2005 Real-Parameter Optimisation competition. It is a restart algorithm that uses CMA-ES and detects premature convergence and launches a restart strategy that doubles the population size on each restart. This process allows a more global approach of the search which empowers the operation of the CMA-ES on multimodal functions.

- MDE_pBX is a state-of-the-art differential evolution(DE). It uses a new mutation operator called *DE/current-to-gr_best/1*,a variant of the *DE/current-to-best/1*, which uses the best of a random group of individuals in the population instead of the global best and performs a the recombination with a random individual of the p best individuals of the population. It also adapts its parameters according to the successes and failures of each of them.

- 3SOME is an example from the MC family. It is a simple, in its concept and implementation, memetic optimiser based on the philosophical concept of Ockham's Razor. The search is divided in three stages each of which corresponds to a variations between explorations and exploitation named long middle and short distance exploration.

The experiments on these algorithms have been performed using the original source code provided by the authors or, when available, their published results.

Results on the CEC'2005 benchmark

Table II.9 shows the results of the comparison with those three algorithms applying the Wilcoxon's test. With regards to the detailed results in Appendix Tables A.4, A.5 and A.6, our algorithm obtains signitificantly better results than 3SOME, thus the inconvenience of the greater complexity of our algorithms is justified by its results.

Concerning MDE_pBX, the overall performances of our algorithm is significantly better (with $\alpha = 0.05$). When analysing the results on each dimension individually, we can see that the superiority of our algorithm appears in higher dimensions (30 and 50 with $\alpha = 0.05$).

Table II.9: Wilcoxon signed rank test results between RMA-LSCh-CMA ($R+$) and reference algorithms ($R-$)

Dim	RMA-LSCh-CMA vs	$R+$	$R-$	$p - value$
10	3SOME	291.5	10	5.13E-6
10	MDE_pBX	195.5	108	0.230
10	IPOP-CMA-ES	94.5	209	0.117
30	3SOME	299.5	25.5	5.88E-5
30	MDE_pBX	257.5	67.5	0.01
30	IPOP-CMA-ES	162.5	162.5	1
50	3SOME	282.5	42.5	6.73E-4
50	MDE_pBX	299.5	25.5	5.88E-5
50	IPOP-CMA-ES	212	113	0.191
All Dimensions	3SOME	2560.5	218	2.97E-10
All Dimensions	MDE_pBX	2239.5	541	5.11E-6
All Dimensions	IPOP-CMA-ES	1396	1387.5	1

Finally, IPOP-CMA-ES obtains equivalent results over the whole benchmark. The differences. We note a small tendency of our algorithm to perform better on higher dimensions and worse on smaller ones although no significant difference can be detected.

Results on the SOCO'2011 benchmark

We saw in the previous section that the performance of the RMA-LSCh-CMA algorithm against the IPOP-CMA-ES was improving when increasing the dimensionality of the problems. It is thus interesting to assess the performances of this model on higher dimensions. The CEC'2005 benchmark does not propose problems in dimensions higher than 50. We thus used the Soft Computing Special Issue on Large Scale Continuous Optimisation Problems (SOCO'2011) benchmark to assess the performances of our proposal against IPOP-CMA-ES in a higher dimension. We performed the experiments in dimension 100. The detailed results can be seen in Appendix A.3.

We can see from Table II.10 that we obtain better results with a significant level of $\alpha = 0.05$ in higher dimensions than the IPOP-CMA-ES and MDE_pBX. Against the MA-LSCh-CMA and the 3SOME, the results obtained are statistically equivalent. However, we note that 3SOME obtains slightly better results. This reflects the declining efficiency of CMA-ES in higher dimensions.

In [MLSH10], the original MA with LS chaining is applied to the SOCO'2011 benchmark using a Solis Wets algorithm [SW81] as LS instead of

Table II.10: Wilcoxon signed ranks test results between RMA-LSCh-CMA ($R+$) reference algorithms ($R-$) on the SOCO'2011 benchmark in dimension 100

RMA-LSCh-CMA vs	$R+$	$R-$	$p-value$
MA-LSCh-CMA	119.5	70.5	0.324
3SOME	79	92	0.777
MDE_pBX	180	10	1.64E-4
IPOP-CMAES	153.5	36.5	0.017

CMA-ES. The same way, further experiments can thus be performed in higher dimensions (up to 1000) but would imply modifying the LS.

6 Conclusion

The aim of this chapter is to present a novel niching mechanism for MAs which can improve the solutions by maintaining the diversity of the population. It demonstrates the importance of separating the effort of the global search from the refinement of the solution. To avoid the competition between the EA and the LS, we have decided to divide the search space into rigid niches ensuring each one only contains one solution. The search space is thus divided into equal hypercubes we called regions. In order to assess the efficiency of this strategy, we implemented it in the MA-LSCh-CMA algorithm, creating an algorithm we called RMA-LSCh-CMA. It led to two major modifications. The first one is to ensure that only one solution of the EA's population can be present in a region. This ensures that a certain diversity in the population is maintained and that the close neighbourhood of a solution will not be explored by the EA as this task is meant to be more efficiently performed by the LS method. The second modification is the initialisation of the LS. It is now initialised according to the size the regions to ensure that the region the solution to which the LS is applied is properly explored. Both modifications go in the sense of limiting the competition between the LS and EA by limiting the EA in the performance of the exploitation effort and forcing the LS to focus its search on the close surroundings of solution.

In order to limit the dependence of the model to the niche size, we also proposed a method to automatically update the the number divisions per dimensions. This number is increased along the search to decrease the niche size. As a result, we proved that RMA-LSCh-CMA obtained significantly better results than the original algorithms.

Appendix A

Detailed Results

In this section of the appendices, we list the detailed results of the experiments performed in Chapter II. Tables A.1, A.2 and A.3 in Section 1 contain the results obtained by the static version of RMA-LSCh-CMA. In Section 2, Tables A.4, A.6 and A.6 show the results of the final version of the RMA-LSCh-CMA alog with the the comparison algorithms results. Those two first section contains the results obtained on the CEC'2005 benchmark in dimensions 10, 30 and 50. Finally in Section 3, Table A.7 presents the results obtained for the RMA-LSCh-CMA ant the comparison algorithms on the SOCO'2011 in dimension 100.

1 Results of the static model on the CEC'2005 benchmark

Table A.1: Results in Dimension 10 of the RMA-LSCh-CMA with various values of ND

F/ND	10	50	100
F1	**1.00E-008**	**1.00E-008**	**1.00E-008**
F2	**1.00E-008**	**1.00E-008**	**1.00E-008**
F3	**1.00E-008**	**1.00E-008**	**1.00E-008**
F4	**1.00E-008**	**1.00E-008**	**1.00E-008**
F5	**1.00E-008**	**1.00E-008**	**1.00E-008**
F6	**1.00E-008**	5.68E-003	1.68E-003
F7	**1.00E-008**	**1.00E-008**	**1.00E-008**
F8	**2.04E+001**	**2.04E+001**	**2.04E+001**
F9	8.15E-001	7.96E-002	**1.00E-008**
F10	4.18E+000	2.35E+000	**1.83E+000**
F11	**3.32E-001**	1.29E+000	1.64E+000
F12	1.47E+002	**1.22E+002**	2.19E+002
F13	6.29E-001	5.69E-001	**4.78E-001**
F14	2.84E+000	2.52E+000	**2.15E+000**
F15	**2.13E+002**	2.67E+002	2.72E+002
F16	**8.43E+001**	9.09E+001	9.02E+001
F17	9.72E+001	9.34E+001	**9.28E+001**
F18	**7.79E+002**	8.47E+002	8.57E+002
F19	**7.63E+002**	8.03E+002	8.60E+002
F20	**7.51E+002**	8.21E+002	8.36E+002
F21	**7.47E+002**	7.70E+002	7.70E+002
F22	7.42E+002	7.35E+002	**7.30E+002**
F23	**9.31E+002**	9.47E+002	9.35E+002
F24	2.36E+002	**2.12E+002**	2.76E+002
F25	4.10E+002	**4.06E+002**	4.40E+002

Table A.2: Results in Dimension 30 of the RMA-LSCh-CMA with various values of ND

F/ND	10	50	100
F1	**1.00E-008**	**1.00E-008**	**1.00E-008**
F2	**1.00E-008**	**1.00E-008**	**1.00E-008**
F3	**1.00E-008**	1.06E-008	**1.00E-008**
F4	2.43E+001	4.02E-001	**2.98E-001**
F5	9.27E+001	3.77E+001	**5.70E+000**
F6	2.83E+001	**1.49E+001**	1.57E+001
F7	6.90E-004	**1.00E-008**	**1.00E-008**
F8	2.10E+001	**2.09E+001**	2.09E+001
F9	6.70E+000	2.73E-002	**5.69E-004**
F10	2.46E+001	1.79E+001	**1.71E+001**
F11	**4.04E+000**	1.24E+001	1.60E+001
F12	1.88E+003	**1.64E+003**	2.26E+003
F13	3.46E+000	2.50E+000	**2.17E+000**
F14	1.28E+001	**1.26E+001**	1.27E+001
F15	3.32E+002	3.15E+002	**3.14E+002**
F16	9.69E+001	8.57E+001	**7.55E+001**
F17	9.05E+001	**7.18E+001**	7.36E+001
F18	9.07E+002	**9.02E+002**	**9.02E+002**
F19	9.03E+002	**9.02E+002**	**9.02E+002**
F20	**9.03E+002**	9.06E+002	9.06E+002
F21	**5.00E+002**	**5.00E+002**	**5.00E+002**
F22	8.95E+002	**8.67E+002**	8.76E+002
F23	5.50E+002	**5.34E+002**	5.57E+002
F24	**2.00E+002**	**2.00E+002**	**2.00E+002**
F25	**2.10E+002**	2.11E+002	2.13E+002

Table A.3: Results in Dimension 50 of the RMA-LSCh-CMA with various values of ND

F/ND	10	50	100
F1	**1.00E-008**	**1.00E-008**	**1.00E-008**
F2	**1.00E-008**	1.01E-008	1.03E-008
F3	**1.00E-008**	1.05E-008	**1.00E-008**
F4	3.82E+003	**1.06E+003**	1.83E+003
F5	2.22E+003	1.82E+003	**1.70E+003**
F6	1.87E+001	**9.48E+000**	3.79E+001
F7	**1.00E-008**	1.08E-003	**1.00E-008**
F8	**2.11E+001**	**2.11E+001**	**2.11E+001**
F9	5.69E-001	2.19E-002	**1.00E-003**
F10	6.60E+001	**3.63E+001**	3.81E+001
F11	**1.11E+001**	2.61E+001	3.27E+001
F12	1.47E+004	**1.03E+004**	1.18E+004
F13	6.00E+000	4.52E+000	**4.06E+000**
F14	2.26E+001	2.23E+001	**2.21E+001**
F15	3.13E+002	3.57E+002	**3.01E+002**
F16	5.89E+001	7.12E+001	**5.33E+001**
F17	9.59E+001	8.47E+001	**5.92E+001**
F18	**8.72E+002**	8.97E+002	9.21E+002
F19	**8.21E+002**	9.21E+002	9.20E+002
F20	8.97E+002	**8.93E+002**	9.21E+002
F21	5.24E+002	5.12E+002	**5.00E+002**
F22	9.41E+002	9.35E+002	**9.12E+002**
F23	5.67E+002	**5.39E+002**	5.53E+002
F24	**2.00E+002**	**2.00E+002**	**2.00E+002**
F25	**2.14E+002**	2.15E+002	2.19E+002

2 Results of the dynamic RMA-LSCh-CMA, the MA-LSCh-CMA, IPOP-CMA-ES, MDE_pBX and 3SOME on the CEC'2005 benchmark

Table A.4: Results on the CEC'2005 benchmark in dimension 10

F	RMA-LSCh-CMA	MA-LSCh-CMA	IPOP-CMAES	MDE_pBX	3SOME
F1	1.00E-008	1.00E-008	1.00E-008	1.00E-008	1.00E-008
F2	1.00E-008	1.00E-008	1.00E-008	1.00E-008	1.00E-008
F3	1.00E-008	1.00E-008	1.00E-008	1.00E-008	4.57E+004
F4	1.00E-008	5.54E-003	1.00E-008	1.00E-008	2.00E+002
F5	1.00E-008	6.75E-007	1.00E-008	1.00E-008	1.76E+003
F6	1.00E-008	3.19E-001	1.00E-008	1.59E-001	6.64E+001
F7	1.00E-008	1.43E-001	1.00E-008	1.27E+003	1.27E+003
F8	2.03E+001	2.00E+001	2.00E+001	2.01E+001	2.00E+001
F9	1.00E-008	1.00E-008	2.39E-001	1.00E-008	1.00E-008
F10	2.79E+000	2.67E+000	7.96E-002	4.61E+000	4.27E+001
F11	5.04E-001	2.43E+000	9.34E-001	2.20E+000	7.38E+000
F12	6.31E+001	1.14E+002	2.93E+001	9.23E+002	2.25E+002
F13	4.83E-001	5.45E-001	6.96E-001	5.08E-001	4.72E-001
F14	2.55E+000	2.25E+000	3.01E+000	2.50E+000	4.18E+000
F15	1.95E+002	2.24E+002	2.28E+002	2.67E+002	2.28E+002
F16	9.48E+001	9.18E+001	9.13E+001	9.80E+001	1.99E+002
F17	9.52E+001	1.01E+002	1.23E+002	1.08E+002	2.28E+002
F18	7.42E+002	8.84E+002	3.32E+002	6.30E+002	8.90E+002
F19	7.17E+002	8.78E+002	3.26E+002	6.24E+002	9.28E+002
F20	7.93E+002	8.63E+002	3.00E+002	6.71E+002	9.14E+002
F21	7.03E+002	7.94E+002	5.00E+002	6.54E+002	9.26E+002
F22	6.76E+002	7.53E+002	7.29E+002	7.55E+002	8.60E+002
F23	8.91E+002	8.88E+002	5.59E+002	9.03E+002	9.23E+002
F24	2.36E+002	2.28E+002	2.00E+002	2.36E+002	2.88E+002
F25	4.08E+002	4.55E+002	3.74E+002	8.60E+002	1.80E+003
F26	2.42E+000	2.94E+000	2.12E+000	3.16E+000	4.36E+000

Table A.5: Results on the CEC'2005 benchmark in dimension 30

F	RMA-LSCh-CMA	MA-LSCh-CMA	IPOP-CMAES	MDE_pBX	3SOME
F1	1.00E-008	1.00E-008	1.00E-008	1.00E-008	1.00E-008
F2	1.00E-008	1.00E-008	1.00E-008	1.00E-008	1.00E-008
F3	1.00E-008	2.75E+004	1.00E-008	4.53E+004	1.82E+005
F4	5.94E-001	3.02E+002	1.11E+004	1.84E-007	7.88E+003
F5	1.00E-008	1.26E+003	1.00E-008	4.65E+000	1.25E+004
F6	1.00E-008	1.12E+000	1.00E-008	1.28E+000	8.40E+001
F7	4.93E-004	1.75E-002	1.00E-008	4.70E+003	4.70E+003
F8	2.10E+001	2.00E+001	2.01E+001	2.03E+001	2.00E+001
F9	1.89E-004	1.00E-008	9.38E-001	1.67E+001	1.00E-008
F10	1.95E+001	2.25E+001	1.65E+000	2.67E+001	3.39E+002
F11	5.65E+000	2.15E+001	5.48E+000	1.46E+001	3.25E+001
F12	1.59E+003	1.67E+003	4.43E+004	1.75E+005	2.84E+003
F13	2.07E+000	2.03E+000	2.49E+000	3.65E+000	1.73E+000
F14	1.26E+001	1.25E+001	1.29E+001	1.25E+001	1.37E+001
F15	3.19E+002	3.00E+002	2.08E+002	2.87E+002	2.09E+002
F16	1.41E+002	1.26E+002	3.50E+001	1.54E+002	4.20E+002
F17	2.06E+002	1.83E+002	2.91E+002	1.70E+002	4.09E+002
F18	9.05E+002	8.98E+002	9.04E+002	9.05E+002	9.68E+002
F19	9.05E+002	9.01E+002	9.04E+002	8.96E+002	9.79E+002
F20	9.01E+002	8.96E+002	9.04E+002	9.06E+002	9.56E+002
F21	5.00E+002	5.12E+002	5.00E+002	5.24E+002	1.02E+003
F22	8.41E+002	8.80E+002	8.03E+002	8.52E+002	1.13E+003
F23	5.49E+002	5.34E+002	5.34E+002	5.84E+002	8.95E+002
F24	2.00E+002	2.00E+002	9.10E+002	2.30E+002	4.13E+002
F25	2.10E+002	2.14E+002	2.11E+002	9.68E+002	1.72E+003

Table A.6: Results on the CEC'2005 benchmark in dimension 50

F	RMA-LSCh-CMA	MA-LSCh-CMA	IPOP-CMAES	MDE_pBX	3SOME
F1	1.00E-008	1.00E-008	1.00E-008	1.00E-008	1.00E-008
F2	1.00E-008	3.06E-002	1.00E-008	1.00E-008	1.00E-008
F3	1.00E-008	3.21E+004	1.00E-008	8.37E+004	1.51E+005
F4	4.73E+002	3.23E+003	4.68E+005	2.97E+001	3.37E+004
F5	4.92E+002	2.69E+003	2.85E+000	2.31E+003	1.90E+004
F6	5.01E+000	4.10E+000	1.00E-008	9.55E+000	1.01E+002
F7	1.00E-008	5.40E-003	1.00E-008	6.20E+003	6.20E+003
F8	2.11E+001	2.00E+001	2.01E+001	2.02E+001	2.00E+001
F9	1.07E-003	1.00E-008	1.39E+000	5.24E+001	1.00E-008
F10	4.58E+001	5.01E+001	1.72E+000	6.07E+001	8.61E+002
F11	1.22E+001	4.13E+001	1.17E+001	3.75E+001	6.23E+001
F12	1.51E+004	1.39E+004	2.27E+005	9.47E+005	6.01E+003
F13	3.66E+000	3.15E+000	4.59E+000	9.11E+000	3.01E+000
F14	2.23E+001	2.22E+001	2.29E+001	2.24E+001	2.35E+001
F15	2.90E+002	3.72E+002	2.04E+002	3.45E+002	2.72E+002
F16	6.96E+001	6.90E+001	3.09E+001	9.78E+001	5.13E+002
F17	1.22E+002	1.47E+002	2.34E+002	1.38E+002	5.18E+002
F18	8.45E+002	9.41E+002	9.13E+002	9.33E+002	1.12E+003
F19	8.72E+002	9.38E+002	9.12E+002	9.32E+002	1.10E+003
F20	8.70E+002	9.28E+002	9.12E+002	9.35E+002	1.13E+003
F21	5.24E+002	5.00E+002	1.00E+003	5.67E+002	8.50E+002
F22	8.63E+002	9.14E+002	8.05E+002	9.00E+002	1.17E+003
F23	5.39E+002	5.39E+002	1.01E+003	5.87E+002	8.58E+002
F24	2.00E+002	2.00E+002	9.55E+002	3.61E+002	1.16E+003
F25	2.14E+002	2.21E+002	2.15E+002	1.23E+003	1.78E+003

3 Results on the SOCO'2011 benchmark in dimension 100

Table A.7: Results on the SOCO'2011 benchmark in dimension 100

F	RMA-LSCh-CMA	MA-LSCh-CMA	IPOP-CMAES	MDE_pBX	3SOME
F1	0.00E+000	0.00E+000	0.00E+000	1.41E-013	0.00E+000
F2	8.27E-011	1.26E-001	1.51E-010	6.66E+001	1.39E-008
F3	2.03E+002	1.15E+001	3.88E+000	1.70E+002	5.00E+001
F4	0.00E+000	1.47E+000	2.50E+002	2.11E+002	8.30E-001
F5	3.65E-003	0.00E+000	1.58E-003	3.65E-002	0.00E+000
F6	1.09E-012	8.07E-014	2.12E+001	3.09E+000	0.00E+000
F7	4.64E-014	0.00E+000	4.22E-004	0.00E+000	4.79E-003
F8	1.21E-004	2.26E+003	0.00E+000	2.40E-001	0.00E+000
F9	5.60E+002	5.64E+002	1.02E+002	5.57E+002	5.81E+002
F10	0.00E+000	0.00E+000	1.66E+001	3.40E+001	1.09E-002
F11	6.55E+000	6.82E-001	1.64E+002	9.92E+001	9.66E+000
F12	4.53E+000	1.25E+000	4.17E+002	1.61E+002	5.55E-002
F13	7.10E+001	1.04E+002	4.21E+002	3.47E+002	9.17E+001
F14	1.66E-001	1.00E+000	2.55E+002	1.69E+002	6.72E-001
F15	3.57E-014	4.12E-007	6.30E-001	5.11E+000	3.17E-002
F16	3.75E+000	1.29E-001	8.59E+002	2.62E+002	8.42E-002
F17	4.14E+001	2.32E+002	1.51E+003	4.65E+002	3.78E+001
F18	1.26E+000	1.68E-001	3.07E+002	1.05E+002	3.15E-002
F19	1.42E-014	0.00E+000	2.02E+001	2.22E+001	7.97E-003

Chapter III

Region-based memetic algorithm with archive for multimodal optimisation

1 Introduction

Many real world problems offer various solutions considered as global optima. The identification of multiple solution has thus gained popularity in the research community. It is referred to multimodal optimisation as the objective is to retrieve more than one optima. While classical evolutionary algorithms (EA) were designed to identify a single optimum, modification have to be proposed to prevent their convergence and maintain the diversity in their population in order to ensure the exploration and exploitation of distinct areas of the fitness landscape. Such techniques, known as niching strategies [DMQS11], are meant to maintain subgroups of individuals in a single population in order to locate multiple optima.

Most existing techniques' efficiency rely on two problem dependent parameters, the niche radius and the population size [DJ75, GR87, Pet96]. The first one should be defined according to the distance between optima in the fitness landscape and the second one according to the number of optima to locate. Both information are however usually unknown in real world problems. Nowadays, research interest focuses in designing EA less dependent on those parameters.

In addition, the challenges when designing an EA for multimodal optimisation are to create models offering great exploration power to identify with the highest accuracy several global optima and being the least dependent to those parameters.

In the previous chapter, we demonstrated how the use of a region-based niching strategy could be used to maintain a high level of diversity in the population of a MA. By maintaining the diversity in the population, the resulting algorithm, RMA-LSCh-CMA was provided with a higher exploration power leading to an significant improvement in the results achieved compared to the original MA.

In this chapter, we enhance the exploration ability of this model for the identification of multiple optima by providing it with an external archive serving two purposes:

- As it is usually done in the literature [ELB13, Zah04, ZL11], the archive stores solutions considered as optima in order to limit the dependence of the algorithm to the population size parameter.

- The archive is used to eliminate from the search space already explored regions. To do so, the archive maintains an index of the regions represented by the solutions it contains and prevents the EA to generate new solutions in those regions. This mechanism is made possible by the representation of a niche by regions as it makes an indexation and retrieval of the regions in the archive very straightforward and efficient. We demonstrate in this paper that the use of the archive for such purpose increases the efficiency of the exploration of the search space by the EA.

As it is done for RMA-LSCh-CMA, the dependency to the niche size (here defined by the number of divisions of the search space) is reduced by increasing along the search the number of divisions. The resulting algorithm is there referred to as Region-based Memetic Algorithm with Archive (RMAwA).

Different studies are performed to demonstrate that the use of the region based niching strategy coupled with an archive provide interesting improvements to the memetic framework and that the RMAwA is a very competitive algorithm against existing ones.

This chapter is organised as follows. In Section 2, we present the RMAwA and detail each components. In Section 3, we explain the experimental framework, the benchmark used, the evaluation and the parameter setting of the algorithm. Section 4 presents the studies performed on the proposals made in this paper a comparison with the results obtained by other algorithms. Finally, some concluding remarks are pointed out in Section 6.

2 Region-based memetic algorithm with archive

Following the general scheme of the RMA-LSCh-CMA presented in the previous chapter, we present here a region-based MA with archive (RMAwA) which takes full advantage the region definition of a niche to enhance the exploration ability of the algorithm.

Usually, the use of an archive in an algorithm designed for multimodal optimisation is ensure to the preservation of found optima and as a consequence limit the dependency of the algorithm to the population size when using the solely the solutions of the population as memory of the found optima [ZL11]. Indeed, as the number of optima present in a the fitness landscape of a problem is usually unknown, having a fixed population size can lead to the loss of optima. Here, we introduce the use of an archive as a storing facility for the preservation of optima but also as an index to restricted regions in order to subtract from the search space the regions already explored.

In this section, we detail the use of the archive, explaining its structure and the solutions it is meant to store followed by a description of the modification brought to the previous model in order to include the archive component in the search.

2.1 The archive

As described previously, this algorithm implements an archive aiming at storing optimised solutions and creating an index of regions of the search space considered undesirable for further exploration.

We describe in this section here the structure of the archive allowing such mechanisms and solutions that are inserted in the archive to serve later for the definition of such undesirable solutions.

Structure

The archive is composed by two collections and its size is not limited. The first one is a simple list of solutions. It contains every real-valued solutions stored in the archive. The second one is a sorted index of the regions represented in the archive. This index allows the efficient retrieval of the regions represented by solutions in the archive. Thanks to this structure, the regions listed in the index are considered as forbidden areas for the generation of future solutions by the EA. This prevents the EA to explore undesirable regions. The index is a self-balancing binary search tree which offers an insertion and search complexity of $O(\log n)$. This low complexity allows a large amount of solutions to be stored in the archive with a limited computational cost. Moreover, it only allows unique elements. In Figure 1, we show an example of the archive structures. We can see how a new solution, composed by the actual real-value solution s_n and the region it belongs to r_n, are used. The former is stored in the archive while the latter is added to the index. Thanks to this property, if a region is represented by multiple solutions in the archive, there will be only one entry in the index for that region. The following section describes what regions are considered as restricted to further exploration.

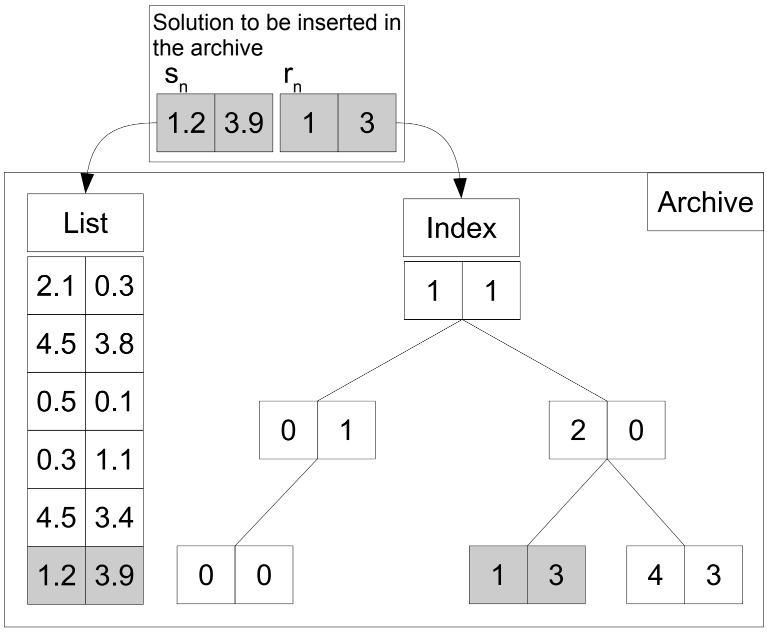

Figure 1: Example of the representation of the archive and its index for a two-dimensional problems

Solutions added in the archive

First of all, the archive's purpose is to store optima identified during the search. Knowing when an optima is found can however be complicated if the fitness value of the optima is unknown. Thanks to the use of an LS method, we consider a solution as an optima (local or global) when the last LS application does not bring sufficient improvements. An insufficient improvement occurs when the difference between the fitness of the the starting point of the LS and the fitness of the obtained solution is below δ_{LS}^{min}.

This method of selection also allows local optima to be included in the archive restricting their region, as well as global optima, from further exploration.

Apart from storing the optima found by means of LS, the archive also saves the solution that serves as starting point of each LS application. The idea behind that is to also eliminate from the search space regions that lead to already identified optima.

To summarize, the archive stores forbidden regions in order to diminish the search space with areas already explored and for which we do not want to dedicate more effort. These regions are the ones containing a local optima, global optima and solutions leading to an optima. The structure of the archive depicted in the previous section allows the presence of multiple solutions belong to the same region in the archive without increasing the retrieval cost as they are represented by only one entry in the index.

2.2 General Scheme

The RMAwA uses a similar scheme as RMA-LSCh-CMA. It alternatively applies an EA and an LS method. At the end of each EA application, the best solution of the population s_{best} is selected for local improvements by the LS.

The same region size update mechanism is used as described in Section 3.4. However, considering that the EA is forbidden to generate solutions in the regions represented in the archive, an update of the number of divisions ND also occurs when every regions are already represented in the archive. That way, we prevent the search to stall. For each update, the corresponding regions of each solution in the population are recalculated and the index of the archive is updated according to the solutions present in the archive. The general scheme of the algorithm can be seen in Algorithm 4.

The following two section describe the EA and the LS method used with the way they are incorporated in the RMAwA.

2.3 The EA

The EA used here, as in the RMA-LSCh-CMA, is a steady-state genetic algorithm (SSGA). It uses the same genetic operators (crossover and mutation). The major difference lies in the mechanisms following the creation of a new solution. When a new solution s_n is generated via the operators described above, it goes through different processes before validation. First the region r_n it belongs to is calculated. Then, r_n is looked for in the archive index. If this region is already represented by another solution in the archive, s_n is discarded and thus not evaluated. Otherwise, if r_n is not in the archive, then s_n is evaluated and compared to the set of solutions

Algorithm 4 Pseudo-code for general scheme of the RMAwA

```
Initialize population
while MaxFEs is not reached do
    Apply SSGA with i_EA evaluations following Algorithm 5
    Select the best individual in the population s_best
    Apply LS method following Algorithm 6 on s_best
    if conditions for number of divisions update then
        Update number of divisions: ND_i = m_u · ND_{i-1}
        Update index of the archive
    end if
end while
```

from the population present in the same region r_n. The worst solution is then removed and replaced by s_n. If r_n is not yet represented in the population, then s_n competes with the worst solution of the whole population to replace it. The SSGA in the RMAwA is described in Algorithm 5.

Algorithm 5 Pseudo-code for the SSGA in RMAwA

```
i = 0
while i < i_EA do
    Select two parents in the population
    repeat
        Create an offspring s_n using crossover and mutation
        Calculate the region r_n to which s_n belongs
    until r_n is not represented in the archive
    Evaluate s_n, i = i + 1
    Retrieve from the population the set of solutions S_{r_n} of solutions belonging
    in the region r_n
    if S_{r_n} ≠ ∅ then
        set S_{r_n} = S_{r_n} ∪ s_n
        Remove worst individual from S_{r_n}
    else
        Replace the worst individual s_worst in the population if f(s_worst) > f(s_n)
    end if
end while
```

2.4 The LS method

Here again, the LS algorithm used remains the same (CMA-ES). Contrarily to RMA-LSCh-CMA, RMAwA does not implement a LS chaining mechanism because the local search here is applied to same solution until it can not be improved anymore. This modification is due to the fact that this algorithm considers as optima solutions which cannot be improved by LS application.

As stated before, the best solution s_{best} of the population is selected for local refinement. To ensure that this solution will not take part to further exploration, it is removed from the population, placed in the archive and replaced by a random solution. The LS is applied multiple times with i_{LS} evaluations until the last application does not bring any sufficient improvements. The obtained solution is then stored in the archive. The application of the LS is described in Algorithm 6.

Algorithm 6 Pseudo-code for the application of the LS in RMAwA

```
Add s_best to the archive
Replace s_best by a random solution in the population
repeat
    Apply the LS method to s_best with i_LS evaluations, giving s_LS
until |f(s_best) − f(s_LS)| < δ_LS^min
Add s_LS to archive
```

3 Experimental framework

The experiments in this chapter were carried out using the benchmark proposed for the special session and competition on niching methods for multimodal function optimization of the IEEE Congress on Evolutionary Computation in 2013 (CEC'2013) [LEE13]. In this section, we describe the framework used to perform these experiments, first by describing the benchmark used and the evaluation method. Finally, we explain the parameter tuning used for the final version of the algorithm.

3.1 The CEC'2013 benchamark

The CEC'2013 benchmark is composed of 12 bounded functions :

- f_1 : Five-Uneven-Peak Trap , $f_1(x)$ where $x \in [0, 30]$, $D = 1$
- f_2 : Equal Maxima , $f_2(x)$ where $x \in [0, 1]$, $D = 1$
- f_3 : Uneven Decreasing Maxima , $f_3(x)$ where $x \in [0, 1]$, $D = 1$

- f_4 : Himmelblau , $f_4(\vec{x})$ where $\vec{x} \in [-6, 6]^D$, $D = 2$
- f_5 : Six-Hump Camel Back , $f_5(x_1, x_2)$ where $x_1 \in [-1.9, 1.9]$ and $x_2 \in [-1.1, 1.1]$, $D = 2$
- f_6 : Shubert , $f_6(\vec{x})$ where $\vec{x} \in [-10, 10]^D$, $D = \{2, 3\}$
- f_7 : Vincent , $f_7(\vec{x})$ where $\vec{x} \in [0.25, 10]^D$, $D = \{2, 3\}$
- f_8 : Modified Rastrigin - All Global Optima , $f_8(\vec{x})$ where $\vec{x} \in [0, 1]^D$, $D = 2$
- f_9 : Composition Function 1 , $f_9(\vec{x})$ where $\vec{x} \in [-5, 5]^D$, $D = 2$
- f_{10} : Composition Function 2 , $f_{10}(\vec{x})$ where $\vec{x} \in [-5, 5]^D$, $D = 2$
- f_{11} : Composition Function 3 , $f_{11}(\vec{x})$ where $\vec{x} \in [-5, 5]^D$, $D = \{2, 3, 5, 10\}$
- f_{12} : Composition Function 4 , $f_{12}(\vec{x})$ where $\vec{x} \in [-5, 5]^D$, $D = \{3, 5, 10, 20\}$

Each function is declined in various dimensionality creating a total of 20 problems.

Table III.1 details the 20 problems characteristics. We are only interested here in identifying the global optima. The number of global optima is known and finite. This information however cannot be used in the optimisation process. In this paper, we refer by f_i to i-th function and F_j to the j-th problem, a problem consisting in the pair $\{f_i, D\}$ where D is the dimensionality of the problem.

This benchmark is very heterogeneous. It proposes problems with a high number of optima (F_8, F_9), dimensionality ranging from 1 to 20. More details on the properties of each functions can be seen in [LEE13].

3.2 Evaluation

For the evaluation of an algorithm's performance over multiple run (50 runs to be executed following the competition requirements), we use the now commonly used *peak ratio* (PR). The PR measures the average percentage of all known global optima found within the Max_{FEs} evalautions given for each problem:

$$PR = \frac{\sum_{i=1}^{NR} NPF_i}{NKP * NR} \tag{III.1}$$

where NPF_i is the number of global optima found in the *i-th* run, NKP is the number of known global optima and NR is the number of runs.

Table III.1: CEC'2013 benchmark problems

Problem	Function	D	Number of optima	MaxFEs
F_1	f_1	1	2	$5 \cdot 10^4$
F_2	f_2	1	5	$5 \cdot 10^4$
F_3	f_3	1	1	$5 \cdot 10^4$
F_4	f_4	2	4	$5 \cdot 10^4$
F_5	f_5	2	2	$5 \cdot 10^4$
F_6	f_6	2	18	$2 \cdot 10^5$
F_7	f_7	2	36	$2 \cdot 10^5$
F_8	f_6	3	81	$4 \cdot 10^5$
F_9	f_7	3	216	$4 \cdot 10^5$
F_{10}	f_8	2	12	$2 \cdot 10^5$
F_{11}	f_9	2	6	$2 \cdot 10^5$
F_{12}	f_{10}	2	8	$2 \cdot 10^5$
F_{13}	f_{11}	2	6	$2 \cdot 10^5$
F_{14}	f_{11}	3	6	$4 \cdot 10^5$
F_{15}	f_{12}	3	8	$4 \cdot 10^5$
F_{16}	f_{11}	5	6	$4 \cdot 10^5$
F_{17}	f_{12}	5	8	$4 \cdot 10^5$
F_{18}	f_{11}	10	6	$4 \cdot 10^5$
F_{19}	f_{12}	10	6	$4 \cdot 10^5$
F_{20}	f_{12}	20	8	$4 \cdot 10^5$

The PR are calculated according to five different accuracy level $\epsilon = \{1E{-}1, 1E{-}2, 1E{-}3, 1E{-}4, 1E{-}5\}$. The accuracy level corresponds to the threshold that determines if the fitness of a given solution is close enough to that of the global optima.

Comparison between algorithms have been performed for each accuracy level independently. For the comparison of two algorithms we considered non-parametric statistical tests [DGMH11]. More specifically, we used the Wilcoxon matched-pairs signed ranks tests for the direct comparison of two algorithms.

3.3 Automatic configuration

In the same fashion as for RMA-LSCh-CMA in the previous chapter, we use IRACE to assist us in the design of the algorithm by automatically tuning the parameters to an optimal setting for this benchmark.

We selected a set of parameters that we considered the most critical and tuned them over the 20 problems of the CEC'2013 benchmark. The list can be seen in Table III.2.

Table III.2: Parameters tuned and obtained values

Parameters	Descriptions	Ranges	Tuned
i_{EA}	EA intensity, number of evaluations allocated to each EA application	$[100, 1000]$	550
i_{LS}	LS intensity, number of evaluations allocated to each LS application	$[100, 1000]$	150
ND_0	Initial number of divisions, defines the size of the niches/regions	$[2, 10]$	2
u	Number of update to be performed	$[2, 5]$	4
m_u	Update multiplier	$[1, 5]$	1.7
NP	Population size of the EA	$[40, 120]$	70
α	Parameter for the $BLX-\alpha$ crossover	$[0.1, 0.9]$	0.9

We can note that the EA intensity is almost four times the LS intensity. This is due to the fact that the LS is applied multiple times (until the improvements brought not significant enough) in each cycle. Concerning the number of division, we can see that the smallest number of divisions have been preferred ($ND_0 = 2$) along with a slow increase along the search by multiplying five times by 1.7. The number of divisions sequence is then $[2, 4, 7, 13, 23, 40]$. Finally an important thing to note is the value of the α parameter for the BLX-α. Set to a high value ($\alpha = 0.9$), it gives the EA a great exploration range.

The other parameters listed in Table III.3 were left to their default values taken from the corresponding papers. δ_{LS}^{min} defines the accuracy required for the search and is set to $1E{-}6$ as the highest accuracy level required can is $1E{-}5$.

Table III.3: Other parameters

Parameters	Descriptions	Value
λ	Parameter to define the CMA-ES population size $p = 4 + \lambda ln(D)$	3 [HMK03]
μ	Defines the parent size for the CMA-ES p/μ	2 [HMK03]
α	Parameter for the $BLX - \alpha$ crossover	0.5
NAM_{size}	Size of the NAM selection method	3
δ_{LS}^{min}	Threshold for the LS stopping criterion	$1E{-}6$

The parameters presented in Table III.2 and III.3 are the ones used in the every experiments performed on every function and dimension of the benchmark. When modifying certain parts of the model to assess their performances in the following section, these parameters remain the same.

4 Experimental results

Based on the experimental framework (benchmark, evaluation and parameters) explained in the previous section, we assess here the performances of the different proposals made in this chapter as well as the performances of the algorithm against existing models. First we start by proving that using the region definition of a niche compared to the euclidean is more efficient in terms of computational time and exploration. We then demonstrate that using the solutions in the archive as excluding regions enhance the performances of the model. We also analyse the memory and computational cost of the archive and the different components of the algorithm. Finally, we compare the proposed algorithm RMAwA with existing algorithms.

4.1 Region niches versus classical niches

Here, we assess the efficiency in terms of computation time and performances of the region definition of niches against the classical definition which implies calculating the euclidean distance between solutions. To do so, we consider the model presented without the use of the archive.

The resulting algorithm here simply referred to as region based memetic algorithm (Region-MA) is opposed to an equivalent algorithm which uses the euclidean distance based definition of a niche as it is used in the classical clearing algorithm. This version is referred to as euclidean-distance based memetic algorithm (Euclidean-MA). On the generation of a new solution by the EA, the offspring created competes with the solutions falling within its niche radius σ, which is set half the size of a region.

In order to simplify the display of the results, we will only focus on the highest level of accuracy ($\epsilon = 1e^{-5}$). Indeed, the definition of a niche only affects the ability of the algorithm to explore the search space and not the precision of the solutions obtained.

Table III.4: PRs (for $\epsilon = 1e^{-5}$) obtained by Region-MA and Euclidean-MA and execution time difference (in percentage)

Problem	F_1	F_2	F_3	F_4	F_5
Region-MA	**0.81**	**0.42**	**1**	**0.97**	**0.99**
Euclidean-MA	0.77	0.56	**1**	0.36	0.87
Time difference (%)	-35.88	-26.10	-28.36	-45.20	-43.57
Problem	F_6	F_7	F_8	F_9	F_{10}
Region-MA	**0**	**0.7**	**0.06**	**0.22**	**0.94**
Euclidean-MA	**0**	0.05	**0.06**	0.01	0.13
Time difference (%)	-30.26	-39.05	-42.13	-38.96	-24.89
Problem	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
Region-MA	**0.68**	**0.86**	**0.63**	**0.64**	**0.15**
Euclidean-MA	0.27	0.14	0.2	0.18	0.14
Time difference (%)	-19.42	-20.90	-28.38	-19.20	-21.11
Problem	F_{16}	F_{17}	F_{18}	F_{19}	F_{20}
Region-MA	**0.36**	**0.16**	**0.17**	**0.13**	**0.13**
Euclidean-MA	0.19	0.13	**0.17**	**0.13**	**0.13**
Time difference (%)	-15.93	-1.56	-25.74	-21.19	-7.42

In Table III.4, we show the PRs obtained by both version along with the execution time difference in percentage. We can see that not only the results obtained are significantly better (see Table III.5 for Wilcoxon comparison) but the execution time is much smaller. Over the whole benchmark, using the region-based niches saves up to 17.4% of time.

Table III.5: Wilcoxon comparison of the PR obtained by Region-MA and Classic-MA (for $\epsilon = 1e^{-5}$)

R+ Region-MA	R- Classic-MA	p-value
189	21	0.0008

4.2 Using the archive to reduce the search space

Usually, the archive is used to store solutions considered as optima to prevent their loss in the evolutionary process and to remove the dependence of the algorithm's performance to the population size. We are not interested here in evaluating the ability of using an archive as a storage facility as it would would imply running experiments with different parameters (especially the population size) making comparisons by definition unfair and thus not reliable. In this section, we are rather interested in assessing how using the regions represented in the archive as excluding areas for the exploration process of the EA improves the exploration of the search space and thus the discovery of more optima.

In order to perform this comparison, we ran two versions of the algorithm. The first one is as presented in Section 2. The second one is the same algorithm without verifying that each solution created by the EA is present or not in the archive (*i.e.* step 6 in Algorithm 2 is ignored). We thus compare here the proposed algorithm which uses an excluding archive (RMAwA) against one with a simple archive called RMA with Simple Archive (RMAwSA).

The same way as the previous experiment, we will only focus on the highest level of accuracy ($\epsilon = 1e^{-5}$). Indeed, the specific use of the archive only affects the algorithm's ability to explore the search space and not the precision of the solutions obtained.

In Table III.6, we show the PRs obtained by both versions of the algorithm. Thanks to the excluding property of the archive, the performances of the algorithm are significantly improved (see Table III.7 for Wilcoxon comparison). We also display in this table the CPU time increase caused by the use of the archive in the search. As we could expect, this property implies more computational effort. However, the percentage increase in the computational time is reduced with the complexity and the dimensionality of the problem. This can be easily explained by the fact that in higher dimensions, the computational time of the evaluation is increased while the time cost of the archive remains steady regardless the dimensionality. Over the whole benchmark, the time cost of RMAwA

Table III.6: PRs of the RMA using an excluding archive (RMAwA) and a simple archive (RMAwSA) for $\epsilon = 1e^{-5}$ and computational time difference between the two versions.

Problem	F_1	F_2	F_3	F_4	F_5
RMAwA	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
RMAwSA	**1.000**	0.312	**1.000**	**1.000**	**1.000**
Time difference (%)	22.6	23.3	7.7	15.5	3.1
Problem	F_6	F_7	F_8	F_9	F_{10}
RMAwA	0.000	**0.917**	0.824	**0.513**	**1.000**
RMAwSA	0.000	0.658	**0.908**	0.343	0.983
Time difference (%)	46.3	34.8	50.8	43.4	4.1
Problem	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
RMAwA	**1.000**	**1.000**	**0.997**	**0.813**	**0.703**
RMAwSA	0.667	0.930	0.667	0.667	0.648
Time difference (%)	5.8	1.5	2.2	21.8	15.3
Problem	F_{16}	F_{17}	F_{18}	F_{19}	F_{20}
RMAwA	**0.670**	**0.660**	**0.233**	**0.128**	**0.125**
RMAwSA	0.667	0.323	0.183	0.125	0.125
Time difference (%)	5.0	14.1	2.4	0.7	1.2

is 8.2% higher than RMAwSA.

Table III.7: Wilcoxon comparison of the PR of the RMA with and without archive (for $\epsilon = 1e^{-5}$)

R+ RMAwA	R- RMAwSA	p-value
186.5	23.5	0.00132

4.3 Time and memory cost of RMAwA

In this section, we study the time and memory cost of RMAwA and more precisely of the proposals made in this paper. First, we assess the memory used by the archive. Then we study the computational cost implied by the exclusive property of the archive and the different components of the algorithm.

Memory cost

We present in this section the memory cost implied by the archive. As explained in Section 2.1, the archive list stores two kind of solutions, the starting and final points of LS applications. In order to evaluate the memory cost of the archive in both cases, we retrieved the number of solutions stored in the archive's list and the number of their corresponding regions represented in the index at the end of each run. From these data, we estimate the total memory size of the archive. The archive's list is a collection of real-value vectors and the index is a collection of integer vectors. In our implementation, real values are represented by "double", coded on eight bytes and integers are represented by "int" coded on four bytes, the space used by the archive is thus calculated by:

$$ArchiveSize = |S| \cdot D \cdot 8 + |R| \cdot D \cdot 4 \tag{III.2}$$

where $|S|$ is the number of solutions in the archive's list, $|R|$ is the number of regions in the index and D is the dimensionality of the problem. The final size is thus proportionate to the dimensionality. It is also dependant on the maximum number of evaluations allowed by the problem. Indeed, an increase in the number of evaluations is increases the number of LS applications and thus the number of solutions stored in the archive. In Table III.8, we present the average of 50 runs of these data along with the dimensionality and the maximum number of evaluation for each function of the CEC'2013 benchmark.

As expected, we can observe a strong increase of the physical size used by the archive for the most complex problems. However, the memory used remains reasonable for nowadays machines. In the most extreme problem, F_{20} where $D = 20$, the archive only uses 64.88 kB of memory.

Table III.8: Average number of elements in the archive's list ($|S|$), the index ($|R|$) and total memory used by the archive (in kB) at the end of each run

Problem	D	Max_{FEs}	$\|S\|$	$\|R\|$	ArchiveSize
F_1	1	$5.00 \cdot 10^4$	135.92	4.58	1.08
F_2	1	$5.00 \cdot 10^4$	130.24	9.96	1.06
F_3	1	$5.00 \cdot 10^4$	129.32	10.52	1.05
F_4	2	$5.00 \cdot 10^4$	106	22.76	1.83
F_5	2	$5.00 \cdot 10^4$	112.76	14.5	1.88
F_6	2	$2.00 \cdot 10^5$	425.52	112.64	7.53
F_7	2	$2.00 \cdot 10^5$	448.28	100.18	7.79
F_8	3	$4.00 \cdot 10^5$	681.84	398.62	20.65
F_9	3	$4.00 \cdot 10^5$	811.64	389.08	23.58
F_{10}	2	$2.00 \cdot 10^5$	431.28	100.68	7.53
F_{11}	2	$2.00 \cdot 10^5$	372.72	106.42	6.66
F_{12}	2	$2.00 \cdot 10^5$	326.04	104.42	5.91
F_{13}	2	$2.00 \cdot 10^5$	349.52	121.84	6.41
F_{14}	3	$4.00 \cdot 10^5$	583	283.48	16.99
F_{15}	3	$4.00 \cdot 10^5$	581.6	278.68	16.90
F_{16}	5	$4.00 \cdot 10^5$	524	259.42	25.54
F_{17}	5	$4.00 \cdot 10^5$	516.64	270.26	25.46
F_{18}	10	$4.00 \cdot 10^5$	446.84	187.36	42.23
F_{19}	10	$4.00 \cdot 10^5$	338.52	168.28	33.02
F_{20}	20	$4.00 \cdot 10^5$	343.8	142.92	64.88

Computational time of the different components of RMAwA

Now, we wish to analyse the amount of time taken by the different components of RMAwA over a whole run namely:

- LS operations: the operations performed by CMA-ES during its search process.
- EA operations: the operations performed by the SSGA to evolve the population.

- Niching: the time it takes to a new solution to go through the niching process (retrieval and comparison of the solutions present in the same region in the population).
- Archive: the time implied by the excluding property of the archive (assessing the presence of the solution's region in the archive's index.

First, we selected in the CEC'13 benchmark function f_{12} which is implemented in this benchmarck in 4 dimensions, $D = \{3, 5, 10, 20\}$, giving four problems ($F_{\{15,17,19,20\}}$). For those four problems, we calculated the CPU time used by each components to assess their scalability. The search effort is unequally divided between the LS and the EA (the number of evaluation at each EA application is fixed while the number of evaluation for each LS application is not limited). Thus, to perform a fair comparison, we only select the average time per evaluation. We plot the results in Figure 2.

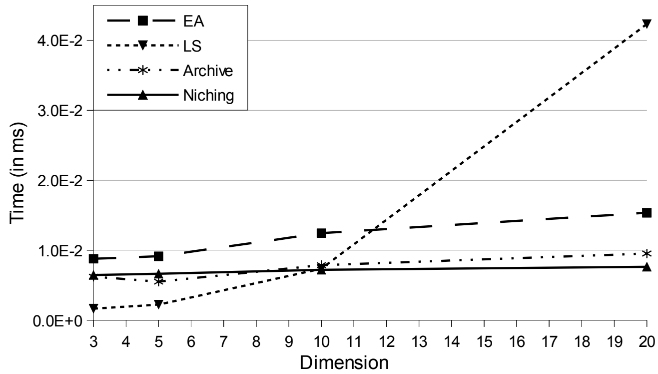

Figure 2: CPU time (in ms) of each component per evaluations for problem f_{12} for different dimensions

As we can see, the complexity of the niching strategy and the use of the archive are barely affected by an increase on the dimensionality. The same way, the simplistic operations of the SSGA algorithms shows interesting scalable properties. The main weakness lies in the use of CMA-ES as LS method. Although it offers a low complexity in the lowest dimensions, passed ten variables, CMA-ES shows poor scalability in terms of complexity.

In order to counter balance the importance of this drawback, we show in table III.9 the CPU time of each components along with the evaluation time. Here, we remind the reader the notation used in this paper, we grouped the problems F_j by function f_i in order to ease the reading and see the relations between the different dimensions of each function.

Table III.9: CPU time (in seconds) details of RMAwA for each problem $F_j = \{f_i, D\}$ with the percentage in the whole optimisatioc process

Problem	$F_1 = \{f_1, 1\}$	$F_2 = \{f_2, 1\}$	$F_3 = \{f_3, 1\}$	$F_4 = \{f_4, 2\}$	$F_5 = \{f_5, 2\}$
Archive	0.150 (18.41%)	0.355 (40.84%)	0.148 (18.99%)	0.099 (14.96%)	0.127 (9.96%)
Niching	0.236 (28.91%)	0.221 (25.39%)	0.254 (32.43%)	0.180 (27.17%)	0.204 (16.01%)
EA	0.260 (31.86%)	0.257 (29.59%)	0.282 (36.08%)	0.207 (31.18%)	0.233 (18.27%)
LS	0.164 (20.08%)	0.027 (3.07%)	0.064 (8.23%)	0.172 (25.93%)	0.696 (54.67%)
Evaluations	0.006 (0.73%)	0.010 (1.11%)	0.033 (4.27%)	0.005 (0.76%)	0.014 (1.10%)
total	0.816	0.869	0.782	0.664	1.273
Problem	$F_6 = \{f_6, 2\}$	$F_8 = \{f_6, 3\}$	$F_7 = \{f_7, 2\}$	$F_9 = \{f_7, 3\}$	$F_{10} = \{f_8, 2\}$
Archive	1.085 (35.82%)	1.512 (29.74%)	1.354 (40.56%)	2.210 (35.27%)	0.740 (28.09%)
Niching	0.759 (25.05%)	1.188 (23.37%)	0.787 (23.58%)	1.403 (22.38%)	0.759 (28.80%)
EA	0.884 (29.18%)	1.513 (29.75%)	0.942 (28.23%)	1.793 (28.61%)	0.934 (35.41%)
LS	0.159 (5.24%)	0.494 (9.72%)	0.173 (5.18%)	0.661 (10.55%)	0.153 (5.82%)
Evaluations	0.143 (4.70%)	0.377 (7.41%)	0.081 (2.44%)	0.200 (3.19%)	0.050 (1.88%)
total	3.029	5.085	3.338	6.267	2.636
Problem	$F_{11} = \{f_9, 2\}$	$F_{12} = \{f_{10}, 2\}$	$F_{13} = \{f_{11}, 2\}$	$F_{14} = \{f_{11}, 3\}$	$F_{16} = \{f_{11}, 5\}$
Archive	0.615 (6.15%)	0.521 (5.38%)	0.604 (6.15%)	0.910 (3.88%)	0.853 (2.60%)
Niching	0.675 (6.74%)	0.586 (6.05%)	0.630 (6.42%)	1.028 (4.39%)	0.951 (2.90%)
EA	0.903 (9.02%)	0.802 (8.28%)	0.852 (8.69%)	1.441 (6.15%)	1.300 (3.96%)
LS	0.332 (3.32%)	0.207 (2.13%)	0.210 (2.14%)	0.412 (1.76%)	0.671 (2.04%)
Evaluations	7.479 (74.77%)	7.571 (78.16%)	7.515 (76.59%)	19.641 (83.82%)	29.054 (88.50%)
total	10.003	9.687	9.812	23.431	32.829
Problem	$F_{18} = \{f_{11}, 10\}$	$F_{15} = \{f_{12}, 3\}$	$F_{17} = \{f_{12}, 5\}$	$F_{19} = \{f_{12}, 10\}$	$F_{20} = \{f_{12}, 20\}$
Archive	1.065 (1.81%)	0.982 (4.15%)	0.773 (2.33%)	0.723 (1.22%)	0.891 (0.71%)
Niching	0.868 (1.48%)	1.020 (4.31%)	0.932 (2.81%)	0.663 (1.12%)	0.714 (0.57%)
EA	1.267 (2.16%)	1.387 (5.86%)	1.282 (3.87%)	1.144 (1.93%)	1.435 (1.15%)
LS	2.099 (3.58%)	0.399 (1.69%)	0.580 (1.75%)	2.267 (3.81%)	12.974 (10.37%)
Evaluations	53.400 (90.97%)	19.867 (83.98%)	29.560 (89.23%)	54.636 (91.93%)	109.063 (87.20%)
total	58.698	23.655	33.127	59.434	125.077

From this table, when increasing the dimensionality, even if the proportion of the LS (*i.e.* CMA-ES) operations increases, the total CPU time is particularly affected by the computational time of the evaluation which is independant of the algorithm. However, as the complexity of CMA-ES increases explonetially with the dimension, larger scale problems may require the use of another LS method.

4.4 Comparison with existing algorithms

In this section we compare the results obtained by our algorithm, RMAwA. We selected a number of algorithm from the literature along with algorithms presented for the CEC'2013 competition:

- PNA-NSGAII [BD13] proposed for the competition, this algorithm is an improvement of A-NSGAII [DS12]. Those algorithms tackles the multimodal optimisation problem by turning them into bi-objective problems. The first objective is the minimisation of the original function and the second one is the maximisation of the diversity brought by the evaluated individual.

- dADE/nrand/1/bin [ELB13] : a DE using a neighbourhood based mutation strategy and a dynamically updated archive.

- CrowdingDE [Tho04] : a DE using a crowding method to prevent premature convergence.

- DE/nrand/1/bin [EPV11] : a DE using the neighbourhood based mutation strategy.

- NCDE [QSL12] : a differential evolution using a neighbourhood based mutation and a crowding mechanism.

- r3pso [Li10] : a PSO using a ring neighbourhood topology.

The two first algorithms (PNA-NSGAII and dADE/nrand/1) took part of the CEC'2013 competition. The two following ones (CrowdingDE and DE/nrand/1/bin) were given as base algorithms by the organisers and the two last ones (NCDE and r3pso) are additional niching algorithms. The detailed results of each algorithms can be seen in the Appendix. We first analyse the overall performance of each algorithm on the benchmark and how their performances are affected by the increase of accuracy. Then we study in details their behaviour according to the problem characteristics.

Accuracy level analysis

We analyse here the general performances of these algorithms on the CEC'2013 benchmark for each accuracy level. To support this analysis, we show in Table III.10 the mean rankings of each algorithm according to the different accuracy levels and in Table III.11 the Wilcoxon comparison of RMAwA against the other algorithms.

Table III.10: Mean rankings obtained by different algorithms over every functions CEC'2013 benchmark for each accuracy level

Accuracy level	$1E{-}1$	$1E{-}2$	$1E{-}3$	$1E{-}4$	$1E{-}5$
CrowdingDE	4.25	4.28	4.45	4.68	4.7
DE/nrand/1/bin	5.08	4.18	4.05	3.63	3.43
r3pso	5.00	5.78	5.95	6.03	6.15
NCDE	3.95	4.38	4.33	4.38	4.48
PNA-NSGAII	3.4	3.73	3.83	4.13	3.9
dADE/nrand/1	**3.05**	3.00	3.08	3.00	3.13
RMAwA	3.28	**2.68**	**2.33**	**2.18**	**2.23**

We can see that RMAwA is second best for the smallest accuracy level ($\epsilon = 1E{-}1$) behind dADE/nrand/1/bin although no statistical difference can be observed from Table III.11. The superiority of RMAwA gets notable when increasing the accuracy level. This is confirmed by reaching significantly better results than every other algorithm, for $\epsilon = \{1E{-}4, 1E{-}5\}$.

As we can see in the detailed result tables in the Appendix, our algorithm is less affected by the accuracy requirements of a problem compared to other algorithms. The PRs obtained remain constant when increasing the accuracy level while it tends to considerably decrease for other algorithms. This is mainly due to the specific care implied by the LS mechanism in the MA framework to strongly refine every promising solutions. It is also well known that CMA-ES is a powerful method to achieve this objective ensuring that each solutions it refines will be very accurately close to the targeted optima.

Problem specific performance analysis

Let us now consider every problem individually. As it is the most challenging for this benchmark, we will consider here only the highest accuracy level ($\epsilon = 1E{-}5$). Table III.12 lists the PRs obtained by each algorithm for this accuracy level.

Table III.11: Wilcoxon comparison of the PRs of the RMAwA ($R+$) with other algorithms ($R-$) (for $\epsilon = \{1E{-}1, 1E{-}2, 1E{-}3, 1E{-}4, 1E{-}5\}$)

$\epsilon = 1E{-}1$			
RMAwA vs	$R+$	$R-$	p-value
CrowdingDE	150.5	59.5	**0.0935**
DE/nrand/1/bin	179.5	30.5	**0.0039**
r3pso	192.5	17.5	**0.0004**
NCDE	128.5	65	0.2273
PNA-NSGAII	97	95.5	0.9839
dADE/nrand/1	68.5	125	0.2862
$\epsilon = 1E{-}2$			
RMAwA vs	$R+$	$R-$	p-value
CrowdingDE	172	20.5	**0.0016**
DE/nrand/1/bin	173.5	36.5	**0.0089**
r3pso	205	5	**1.91E−5**
NCDE	199.5	10.5	**0.0001**
PNA-NSGAII	135.5	74.5	0.2549
dADE/nrand/1	125.5	84.5	0.4441
$\epsilon = 1E{-}3$			
RMAwA vs	$R+$	$R-$	p-value
CrowdingDE	185	7.5	**0.0001**
DE/nrand/1/bin	162	30.5	**0.0077**
r3pso	188.5	3	**0.0000**
NCDE	185	7.5	**0.0001**
PNA-NSGAII	141	51.5	**0.0837**
dADE/nrand/1	146.5	63.5	0.1279
$\epsilon = 1E{-}4$			
RMAwA vs	$R+$	$R-$	p-value
CrowdingDE	205	5	**1.91E−5**
DE/nrand/1/bin	162.5	31	**0.0082**
r3pso	188.5	3	**1.91E−5**
NCDE	185	7.5	**0.0001**
PNA-NSGAII	158	52	**0.0484**
dADE/nrand/1	165.5	44.5	**0.0227**
$\epsilon = 1E{-}5$			
RMAwA vs	$R+$	$R-$	p-value
CrowdingDE	185	7.5	**0.0001**
DE/nrand/1/bin	160.5	33	**0.0108**
r3pso	188.5	3	**1.91E−5**
NCDE	199.5	10.5	**0.0001**
PNA-NSGAII	152	40.5	**0.0274**
dADE/nrand/1	151.5	42	**0.0323**
$\epsilon = *$			
RMAwA vs	$R+$	$R-$	p-value
CrowdingDE	4531	431.5	**0**
DE/nrand/1/bin	4197.5	768	**2.55E−9**
r3pso	4844.5	115	**0**
NCDE	4502	462.5	**0**
PNA-NSGAII	3427	1535.5	**0.0010**
dADE/nrand/1	3288	1762	**0.0087**

In this analysis we will focus on the problems offering the major differences between the results obtained by the compared algorithms. Concerning problems with highly multimodal fitness landscape, F_7 to F_9 where the number of optima ranges from 36 to 216, RMAwA obtains the second best performances after PNA-NSGAII. For problems with combination functions F_{11} to F_{20}, RMAwA obtains the best results for lower dimensions ($D = \{2, 3, 5\}$, F_{11} to F_{17}). When increasing the dimensionality ($D = 10$), RMAwA still performs reasonably ranking fourth for function F_{18}, second for F_{19}. Finally, we can note that for F_{20}, where $D = 20$, while most of the algorithms including the ones with the best rankings over the whole benchmark (dADE, PNA-NSGAII) fail in identifying any optimum, RMAwA still locates at least a few optima, obtaining the best performances on this problem.

As we can see in Table III.12 and according the the No Free Lunch Theorem, designing an algorithm with fixed parameters for the an heterogeneous test bed of problems is very challenging. We note that algorithms could perform well on problems with certain characteristics and poorly on others. The proposed algorithm of this paper, RMAwA, offers an overall performance significantly superior to the other algorithms by obtaining competitive if not better results in most problems proposed in the CEC'2013 benchmark.

5 Conclusion

In this chapter, we presented an application of a RMA to multimodal optimisation problems. The aim was to assess the efficiency and the possibilities offered by this original representation of a niche to identify multiple optima in a fitness landscape.

We demonstrated that the use of the region based niching strategy is more efficient than using a classical euclidean definition of a niche in this model. Such definition also allows an efficient indexation and retrieval from the archive of the regions already explored. Subtracting those regions from the search space significantly improves the ability of the model to identify multiple optima in a fitness landscape.

Finally, we compared the proposed algorithm with existing techniques using the benchmark issued for the special session and competition on niching methods for multimodal function optimization of the IEEE Congress on Evolutionary Computation in 2013. We noted that our algorithm was fairly independent to the different accuracy levels tested in this benchmark compared to the other algorithms obtaining significantly better results.

Table III.12: PRs obtained by each algorithm for $\epsilon = 1E{-}5$ on the CEC'2013 benchmark. Values in the parenthesis represent the standard competition ranking of each algorithm for each problem

Problem	CrowdingDE	DE/nrand/1/bin	r3pso	NCDE	PNA-NSGAII	dADE/nrand/1/bin	RMAwA
F_1	0.000 (6)	1.000 (1)	0.000 (6)	1.000 (1)	1.000 (1)	1.000 (1)	1.000 (1)
F_2	1.000 (1)	1.000 (1)	1.000 (1)	1.000 (1)	1.000 (1)	1.000 (1)	1.000 (1)
F_3	1.000 (1)	1.000 (1)	1.000 (1)	1.000 (1)	1.000 (1)	1.000 (1)	1.000 (1)
F_4	0.420 (6)	1.000 (1)	0.075 (7)	0.805 (4)	0.605 (5)	1.000 (1)	1.000 (1)
F_5	1.000 (1)	1.000 (1)	0.000 (6)	1.000 (1)	0.000 (6)	1.000 (1)	1.000 (1)
F_6	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
F_7	0.716 (4)	0.333 (7)	0.497 (6)	0.683 (5)	0.927 (1)	0.732 (3)	0.917 (2)
F_8	0.038 (6)	0.113 (5)	0.000 (7)	0.252 (4)	0.986 (1)	0.947 (2)	0.824 (3)
F_9	0.270 (5)	0.094 (6)	0.040 (7)	0.276 (4)	0.549 (1)	0.356 (3)	0.513 (2)
F_{10}	1.000 (1)	1.000 (1)	0.998 (7)	1.000 (1)	1.000 (1)	1.000 (1)	1.000 (1)
F_{11}	0.667 (3)	0.670 (2)	0.657 (7)	0.663 (6)	0.667 (5)	0.667 (3)	1.000 (1)
F_{12}	0.002 (7)	0.777 (2)	0.330 (5)	0.573 (4)	0.028 (6)	0.728 (3)	1.000 (1)
F_{13}	0.667 (2)	0.667 (2)	0.603 (7)	0.623 (6)	0.660 (5)	0.667 (2)	0.997 (1)
F_{14}	0.667 (2)	0.667 (2)	0.013 (7)	0.610 (6)	0.667 (5)	0.667 (2)	0.813 (1)
F_{15}	0.375 (5)	0.507 (3)	0.005 (7)	0.443 (4)	0.368 (6)	0.620 (2)	0.703 (1)
F_{16}	0.667 (2)	0.657 (5)	0.000 (7)	0.323 (6)	0.667 (4)	0.667 (2)	0.670 (1)
F_{17}	0.000 (6)	0.287 (3)	0.000 (6)	0.245 (5)	0.250 (4)	0.410 (2)	0.660 (1)
F_{18}	0.000 (6)	0.250 (3)	0.000 (6)	0.093 (5)	0.330 (2)	0.627 (1)	0.233 (4)
F_{19}	0.000 (5)	0.170 (1)	0.000 (5)	0.010 (4)	0.027 (3)	0.000 (5)	0.128 (2)
F_{20}	0.000 (3)	0.123 (2)	0.000 (3)	0.000 (3)	0.000 (3)	0.000 (3)	0.125 (1)

Appendix B

Detailed results

This section shows the PR obtained on the CEC'2013 benchmark in the 5 accuracy levels by:

- RMA-Archive (Table B.1)
- CrowdingDE (Table B.3)
- DE/nrand/1/bin (Table B.2)
- r3PSO (Table B.7)
- NCDE (Table B.6)
- PNA-NSGAII (Table B.5)
- dADE/nrand/1/bin (Table B.4).

Table B.1: RMA-Archive

Problem	Function	Dimension	Accuracy level				
			$1e^{-1}$	$1e^{-2}$	$1e^{-3}$	$1e^{-4}$	$1e^{-5}$
F_1	f_1	1	1	1	1	1	1
F_2	f_2	1	1	1	1	1	1
F_3	f_3	1	1	1	1	1	1
F_4	f_4	2	1	1	1	1	1
F_5	f_5	2	1	1	1	1	1
F_6	f_6	2	0.99	0.99	0.99	0.99	0
F_7	f_7	2	1	0.92	0.92	0.92	0.92
F_8	f_6	3	0.82	0.82	0.82	0.82	0.82
F_9	f_7	3	1	0.52	0.52	0.51	0.51
F_{10}	f_8	2	1	1	1	1	1
F_{11}	f_9	2	1	1	1	1	1
F_{12}	f_{10}	2	1	1	1	1	1
F_{13}	f_{11}	2	1	1	1	1	1
F_{14}	f_{11}	3	0.82	0.81	0.81	0.81	0.81
F_{15}	f_{12}	3	0.71	0.7	0.7	0.7	0.7
F_{16}	f_{11}	5	0.68	0.67	0.67	0.67	0.67
F_{17}	f_{12}	5	0.67	0.66	0.66	0.66	0.66
F_{18}	f_{11}	10	0.38	0.24	0.24	0.23	0.23
F_{19}	f_{12}	10	0.13	0.13	0.13	0.13	0.13
F_{20}	f_{12}	20	0.25	0.13	0.13	0.13	0.13

Table B.2: DE/nrand/1/bin

Problem	Function	Dimension	Accuracy level				
			$1e^{-1}$	$1e^{-2}$	$1e^{-3}$	$1e^{-4}$	$1e^{-5}$
F_1	f_1	1	1	1	1	1	1
F_2	f_2	1	1	1	1	1	1
F_3	f_3	1	1	1	1	1	1
F_4	f_4	2	1	1	1	1	1
F_5	f_5	2	1	1	1	1	1
F_6	f_6	2	0.45	0.44	0.44	0.43	0
F_7	f_7	2	0.35	0.35	0.35	0.34	0.33
F_8	f_6	3	0.11	0.11	0.11	0.11	0.11
F_9	f_7	3	0.1	0.1	0.1	0.1	0.09
F_{10}	f_8	2	1	1	1	1	1
F_{11}	f_9	2	0.68	0.67	0.68	0.67	0.67
F_{12}	f_{10}	2	0.86	0.84	0.82	0.82	0.78
F_{13}	f_{11}	2	0.67	0.67	0.67	0.67	0.67
F_{14}	f_{11}	3	0.67	0.67	0.67	0.67	0.67
F_{15}	f_{12}	3	0.52	0.54	0.51	0.5	0.51
F_{16}	f_{11}	5	0.68	0.66	0.66	0.66	0.66
F_{17}	f_{12}	5	0.35	0.33	0.3	0.29	0.29
F_{18}	f_{11}	10	0.4	0.34	0.32	0.27	0.25
F_{19}	f_{12}	10	0.3	0.22	0.2	0.17	0.17
F_{20}	f_{12}	20	0.13	0.13	0.13	0.13	0.12

Table B.3: Crowding DE/RAND/bin

Problem	Function	Dimension	Accuracy level				
			$1e^{-1}$	$1e^{-2}$	$1e^{-3}$	$1e^{-4}$	$1e^{-5}$
F_1	f_1	1	1	0.71	0.09	0.02	0
F_2	f_2	1	1	1	1	1	1
F_3	f_3	1	1	1	1	1	1
F_4	f_4	2	1	1	1	1	0.42
F_5	f_5	2	1	1	1	1	1
F_6	f_6	2	1	1	0.97	0.11	0
F_7	f_7	2	0.7	0.72	0.72	0.71	0.72
F_8	f_6	3	0.85	0.84	0.72	0.29	0.04
F_9	f_7	3	0.27	0.27	0.27	0.27	0.27
F_{10}	f_8	2	1	1	1	1	1
F_{11}	f_9	2	0.94	0.69	0.67	0.67	0.67
F_{12}	f_{10}	2	0.38	0.06	0.01	0.01	0
F_{13}	f_{11}	2	0.84	0.68	0.67	0.67	0.67
F_{14}	f_{11}	3	0.68	0.67	0.67	0.67	0.67
F_{15}	f_{12}	3	0.73	0.69	0.63	0.49	0.38
F_{16}	f_{11}	5	0.7	0.67	0.67	0.67	0.67
F_{17}	f_{12}	5	0.08	0	0	0	0
F_{18}	f_{11}	10	0.08	0	0	0	0
F_{19}	f_{12}	10	0	0	0	0	0
F_{20}	f_{12}	20	0.5	0.01	0	0	0

Table B.4: dADE/nrand/1/bin

Problem	Function	Dimension	Accuracy level				
			$1e^{-1}$	$1e^{-2}$	$1e^{-3}$	$1e^{-4}$	$1e^{-5}$
F_1	f_1	1	1	1	1	1	1
F_2	f_2	1	1	1	1	1	1
F_3	f_3	1	1	1	1	1	1
F_4	f_4	2	1	1	1	1	1
F_5	f_5	2	1	1	1	1	1
F_6	f_6	2	1	1	1	0.98	0
F_7	f_7	2	1	0.96	0.89	0.82	0.73
F_8	f_6	3	0.99	0.98	0.98	0.97	0.95
F_9	f_7	3	0.84	0.6	0.55	0.43	0.36
F_{10}	f_8	2	1	1	1	1	1
F_{11}	f_9	2	0.89	0.67	0.67	0.67	0.67
F_{12}	f_{10}	2	1	0.89	0.75	0.74	0.73
F_{13}	f_{11}	2	0.74	0.67	0.67	0.67	0.67
F_{14}	f_{11}	3	0.92	0.67	0.67	0.67	0.67
F_{15}	f_{12}	3	1	0.62	0.62	0.63	0.62
F_{16}	f_{11}	5	0.87	0.67	0.67	0.67	0.67
F_{17}	f_{12}	5	0.94	0.47	0.42	0.4	0.41
F_{18}	f_{11}	10	0.68	0.66	0.63	0.63	0.63
F_{19}	f_{12}	10	0.42	0.14	0.06	0.02	0
F_{20}	f_{12}	20	0	0	0	0	0

Table B.5: PNA-NSGA

Problem	Function	Dimension	Accuracy level				
			$1e^{-1}$	$1e^{-2}$	$1e^{-3}$	$1e^{-4}$	$1e^{-5}$
F_1	f_1	1	1	1	1	1	1
F_2	f_2	1	1	1	1	1	1
F_3	f_3	1	1	1	1	1	1
F_4	f_4	2	1	1	1	0.99	0.81
F_5	f_5	2	1	1	1	1	1
F_6	f_6	2	0.56	0.54	0.52	0.47	0
F_7	f_7	2	1	0.74	0.73	0.71	0.68
F_8	f_6	3	0.35	0.33	0.31	0.28	0.25
F_9	f_7	3	0.48	0.33	0.32	0.3	0.28
F_{10}	f_8	2	1	1	1	1	1
F_{11}	f_9	2	0.88	0.68	0.67	0.68	0.66
F_{12}	f_{10}	2	0.75	0.72	0.67	0.64	0.57
F_{13}	f_{11}	2	0.7	0.67	0.67	0.66	0.62
F_{14}	f_{11}	3	0.93	0.67	0.67	0.66	0.61
F_{15}	f_{12}	3	0.67	0.5	0.49	0.47	0.44
F_{16}	f_{11}	5	1	0.52	0.52	0.42	0.32
F_{17}	f_{12}	5	0.92	0.35	0.34	0.3	0.25
F_{18}	f_{11}	10	0.64	0.12	0.11	0.11	0.09
F_{19}	f_{12}	10	0.02	0.02	0.04	0.02	0.01
F_{20}	f_{12}	20	0	0	0	0	0

Table B.6: NCDE

Problem	Function	Dimension	Accuracy level				
			$1e^{-1}$	$1e^{-2}$	$1e^{-3}$	$1e^{-4}$	$1e^{-5}$
F_1	f_1	1	1	1	1	1	1
F_2	f_2	1	1	1	1	1	1
F_3	f_3	1	1	1	1	1	1
F_4	f_4	1	1	1	1	0.61	0.61
F_5	f_5	0	0	0	0	0	0
F_6	f_6	1	0.99	0.8	0.06	0	0
F_7	f_7	1	0.94	0.94	0.94	0.93	0.93
F_8	f_6	1	1	1	1	0.99	0.99
F_9	f_7	0.55	0.55	0.55	0.55	0.55	0.55
F_{10}	f_8	1	1	1	1	1	1
F_{11}	f_9	0.97	0.72	0.68	0.67	0.67	0.67
F_{12}	f_{10}	0.72	0.43	0.23	0.09	0.03	0.03
F_{13}	f_{11}	0.74	0.67	0.67	0.67	0.66	0.66
F_{14}	f_{11}	0.83	0.67	0.67	0.67	0.67	0.67
F_{15}	f_{12}	0.64	0.38	0.38	0.37	0.37	0.37
F_{16}	f_{11}	1	0.67	0.67	0.67	0.67	0.67
F_{17}	f_{12}	0.66	0.25	0.25	0.25	0.25	0.25
F_{18}	f_{11}	1	0.45	0.39	0.35	0.33	0.33
F_{19}	f_{12}	0.58	0.24	0.2	0.17	0.03	0.03
F_{20}	f_{12}	0.35	0.13	0	0	0	0

Table B.7: r3PSO

Problem	Function	Dimension	Accuracy level				
			$1e^{-1}$	$1e^{-2}$	$1e^{-3}$	$1e^{-4}$	$1e^{-5}$
F_1	f_1	1	0.95	0.27	0	0	0
F_2	f_2	1	1	1	1	1	1
F_3	f_3	1	1	1	1	1	1
F_4	f_4	2	1	1	0.97	0.43	0.08
F_5	f_5	2	0	0	0	0	0
F_6	f_6	2	1	0.98	0.88	0.4	0
F_7	f_7	2	1	0.74	0.65	0.56	0.5
F_8	f_6	3	0.02	0	0	0	0
F_9	f_7	3	0.94	0.3	0.19	0.1	0.04
F_{10}	f_8	2	1	1	1	1	1
F_{11}	f_9	2	1	0.67	0.67	0.67	0.66
F_{12}	f_{10}	2	0.67	0.59	0.5	0.4	0.33
F_{13}	f_{11}	2	0.96	0.67	0.67	0.65	0.6
F_{14}	f_{11}	3	0.81	0.26	0.11	0.04	0.01
F_{15}	f_{12}	3	0.29	0.07	0.02	0.01	0.01
F_{16}	f_{11}	5	0	0	0	0	0
F_{17}	f_{12}	5	0	0	0	0	0
F_{18}	f_{11}	10	0	0	0	0	0
F_{19}	f_{12}	10	0	0	0	0	0
F_{20}	f_{12}	20	0	0	0	0	0

Chapter IV

Final remarks

This chapter is dedicated to a summary of the works presented in this thesis. We will remind the different proposals developed along with the conclusions on the different analysis. We state the publications associated with this thesis and the research lines opened by the research conducted here.

1 Summary and conclusion

This thesis is based on the development and study of a novel niching strategy which consists in dividing the search space into equal hypercubes called regions. These regions predefines the niches across the search space. When implemented in an EA, the solutions of the population compete to remain in the region they belong to in order to favour diversity in the population.

Niching strategies are commonly used to serve two purposes:

- Maintaining the diversity in an EA's population in order to prevent fast convergence and ensure the proper exploration of the search space.
- Identifying multiple optima in a fitness landscape in multimodal optimisation problems.

We thus applied the region-based niching strategy in two MA to take advantage of both properties.

1.1 Region-based memetic algorithm with local search chaining for global optimisation

With the objective of creating a strong separation between the exploration effort of the EA and the exploitation effort performed by the LS method of a MA, we applied the region-based niching strategy to a MA. By doing so, we force the EA to explore by preventing it from exploring the close surroundings of the solutions present in the population. Consequently, the LS method is forced to exploit the neighbourhood of the solution it is applied to by forcing it to primarily explore its region.

We applied this strategy to a successful MA with LS chaining (MA-LSCh-CMA) which alternatively applies a SSGA as EA and CMA-ES as LS method in a LS chaining framework. LS chaining is a mechanism that adapt the LS intensity according to the quality of the solution it is applied to by storing alongside with the solution the LS parameters, allowing it to be carried on in future applications.

In order to limit the dependency of the niche size (here defined by the number of divisions per dimensions of the search space) we also proposed a mechanism which dynamically updates the niche size along the search. The idea is simply increase the number of divisions per dimension (and thus reduce the niche size) various times during the search.

We tested the created model called region-based MA with LS chaining and CMA-ES (RMA-LSCh-CMA) on the Special Session on Real Parameter optimisation of the IEE Conference on Evolutionary Competition 2005 (CEC'2005) benchmark for small dimensions (10, 30 and 50) and on the Soft Computing Special Issue on Large Scale optimisation Optimisation (SOCO'2011) benchmark for dimension 100. Using those benchmarks, we performed various experiments to study the influence of the concept introduced here and drew the following conclusions:

- The dynamic update of the niche size offers more robustness to the algorithm by making it less dependent to this parameter, especially when considering the various dimensionality of a problem.
- The use of region-based niching framework, by maintaining a higher diversity in the population, significantly improves the performances of the MA.

Finally, we compared the results obtained with a set of representative algorithm, IPOP-CMA-ES, MDE_pBX and 3SOME.

- We obtain significantly better results than MDE_pBX on both benchmarks.

- IPOP-CMA-ES obtains statistically equivalent results on the CEC'2005 benchmark. However, we can note that IPOP-CMA-ES appears to perform better on lower dimensions and our algorithm better in higher dimensions. This tendency is confirmed by the fact RMA-LSCh-CMA obtains significantly better on the SOCO'2011 benchmark for dimension 100.

- While RMA-LSCh-CMA obtains significantly better results than 3SOME on the CEC'2005 benchmark, we note a slight domination of 3SOME on the SOCO'2011 benchmark over our algorithm. This can be interpreted by the fact that CMA-ES performs with less efficiency when increasing the dimension which should motivate the use of other LS method for larger scale problems.

1.2 Region-based memetic algorithm with archive for multimodal optimisation

Heaven Is for Real Motivated by the idea of assessing the proposed region definition of niches to the original purpose of niching strategies, we developed a model based on the one presented in Chapter II to handle multimodal optimisation problems.

In Chapter III, we present the various modifications brought to that model in order to adapt it to the problems at hand and take advantage of the definition of a niche introduced in this thesis:

- Application of the LS: the LS is applied to a given solution multiple times until it does not bring sufficient improvements.

- Use of an archive as excluding area: The use of an archive allows the conservation of identified optima and prevents their potential loss in the evolutionary process of the EA this also limits the dependency of the algorithm's performance to the population size parameter. The novelty of our model is to consider the regions represented by solutions of the archive as excluding areas, forbidding the EA to generate solutions in them. This process is made possible and little time consuming by the possibility of efficiently index and thus retrieve those regions when a new solution is created.

The resulting algorithm we called region-based memetic algorithm with archive (RMAwA) was tested on the benchmark proposed during the Special Session and Competition on Niching Methods for Multimodal Function Optimisation of the IEEE Conference on Evolutionary Computation (CEC'2013).

Here again, we analysed the improvements brought by the novelties introduced in this model. We first remarked that the region-based niching strategy was less time consuming than using the traditional distance-based notion of niche. We also demonstrated that considering the regions represented in the archive as forbidden areas for the EA to explore significantly improved the exploration power, and thus the optima identification, of the MA at little cost in term of computational time.

Finally, we compared the algorithm with a set of existing techniques and obtained an overall performance significantly better than any of them. However, providing that the CEC'2013 benchmark requires the evaluation of an algorithm for different level of accuracy, it is interesting to add a few remark on the comparison performed here:

- With respect to the compared algorithms, the RMAwA is less sensitive to high level of accuracy. We indeed note that the dominance of our algorithm over the other ones is significantly emphasised when increasing the level of accuracy required. The credit for such performance can be given to the use intensive use of CMA-ES for the refinement of the promising solutions.
- When considering the lowest level of accuracy, RMAwA remains statistically equivalent to other existing algorithms. This shows that our algorithm performances remain equivalent in terms of exploration capabilities to other algorithms.

2 Publications associated with this thesis

In the following, we present a list including the publications associated to this thesis:

- International journals:
  - B. Lacroix, D. Molina, F. Herrera, *Region based memetic algorithm for real-parameter optimisation*. Information Sciences, 262 (2014) 15-31.
  - B. Lacroix, D. Molina, F. Herrera, *Region-based Memetic Algorithm with Archive for multimodal optimisation*. Submited, 2014.
- International conferences:
  - B. Lacroix, D. Molina, F. Herrera, *Region Based Memetic Algorithm with LS chaining*, IEEE World Congress on Computational Intelligence (WCCI), 2012, pp. 1474-1479.

– B. Lacroix, D. Molina, F. Herrera, *Dynamically updated region based memetic algorithm for the 2013 CEC Special Session and Competition on Real Parameter Single Objective Optimization*. IEEE Congress on Evolutionary Computation (CEC), 2013, pp 1945-1951. Note: The proposed algorithm raked third of 23 algorithms in this competition.

3 Future Works

On both models presented in this thesis, we identified strengths and weaknesses. The promising results of both methods encourage the idea of carrying this research line in order to improve the weaknesses identified. The following research line remain opened:

- The models presented in this thesis can be extended to large scale optimisation problems. This would however imply the modification of the LS method. CMA-ES is indeed not adapted to high dimensional problems in terms of computation costs and perform.

- In multimodal optimisation an appropriate trade-off between accuracy and exploration is essential. A specific application may require more global optima to be identified but with less accuracy while on the other hand the user may need fewer optima but with great accuracy. We saw that other models offered better performances (dADE) when less accuracy was required. Identifying and adjusting the parameters of RMAwA (accuracy threshold for the LS, region size adaptation, number of evaluations allocated to EA's application...) to study its ability to adapt to such requirements could be an interesting line of research. This could also be achieved by using different search operators within this framework.

- Choosing to linearly increase the number of divisions of the search space to update the region size has shown to increase the robustness of the model. It mainly makes sense for global optimisation when one might find interesting to let the population to converge in the latter stages of the search. However niching models for multimodal optimisation work more efficiently if the niche size is adapted to the basin of attraction size and the distance between optima. A more advanced niche size (in our case number of divisions) adaptation mechanism could be offer RMAwA significant improvements.

- Models of GAs (*e.g.* Hierchical Genetic Search [SK02]) using binary coding propose a population hierarchy where different levels work with different levels of accuracy (different number of bits in the solutions encoding).

The same way, region-based models could recreate a similar hierarchy with different number of divisions. This would allow different levels of exploration/exploitation to be active using the same optimisers.

- Parallel computation is currently an important research line to increase the computational performances of existing algorithm. By nature, MA algorithms offer the possibility of parallelism by their separation of the effort between different search components. To link it to the previous point, different hierarchical population can be executed in parallel as it is done with Island models [WRH97].

Bibliography

[AFR01] Agostinho C. F., Fernandes C., and Rosa A. (2001) A Study on Non-random Mating and Varying Population Size in Genetic Algorithms Using a Royal Road Function. In *Proceedings of the 2001 IEEE Congress on Evolutionary Computation*, pp. 60–66. IEEE Press.

[AH05] Auger A. and Hansen N. (2005) A Restart CMA Evolution Strategy with Increasing Population Size. In *2005 IEEE Congress on Evolutionary Computation*, pp. 1769–1776.

[BBS07] Balaprakash P., Birattari M., and Stützle T. (2007) Improvement strategies for the F-race algorithm: Sampling design and iterative refinement. In Bartz-Beielstein T., Blesa M. J., Blum C., Naujoks B., Roli A., Rudolph G., and Sampels M. (Eds.) *Hybrid Metaheuristics*, volume 4771 of *Lecture Notes in Computer Science*, pp. 108–122. Springer, Heidelberg, Germany.

[BD13] Bandaru S. and Deb K. (2013) A parameterless-niching-assisted bi-objective approach to multimodal optimization. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pp. 95–102.

[BFM97] Back T., Fogel D. B., and Michalewicz Z. (Eds.) (1997) *Handbook of Evolutionary Computation.* IOP Publishing Ltd., Bristol, UK.

[Bir04] Birattari M. (2004) *The Problem of Tuning Metaheuristics as Seen from a Machine Learning Per-*

spective. PhD thesis, Université Libre de Bruxelles, Brussels, Belgium.

[Bir09] Birattari M. (2009) *Tuning Metaheuristics: A Machine Learning Perspective*, volume 197 of *Studies in Computational Intelligence*. Springer, Berlin/Heidelberg, Germany.

[BL06a] Bird S. and Li X. (2006) Enhancing the robustness of a speciation-based pso. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pp. 843–850.

[BL06b] Bird S. and Li X. (2006) Adaptively choosing niching parameters in a pso. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, GECCO '06, pp. 3–10. ACM, New York, NY, USA.

[Bri81] Brindle A. (1981) *Genetic Algorithms for Function Optimization*. Phd thesis, University of Alberta.

[BS02] Beyer H.-G. and Schwefel H.-P. (2002) Evolution strategies – a comprehensive introduction. *Natural Computing* 1(1): 3–52.

[BSEK06] Bonissone P., Subbu R., Eklund N., and Kiehl T. (june 2006) Evolutionary algorithms + domain knowledge = real-world evolutionary computation. *IEEE Transactions on Evolutionary Computation* 10(3): 256 – 280.

[BSPV02] Birattari M., Stützle T., Paquete L., and Varrentrapp K. (2002) A racing algorithm for configuring metaheuristics. In Langdon W. B. *et al.* (Eds.) *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2002*, pp. 11–18. Morgan Kaufmann Publishers, San Francisco, CA.

[BT97] Blickle T. and Thiele L. (1997) A comparison of selection schemes used in evolutionary algorithms. *Evolutionary Computation* 4: 361–394.

[BVA07] Banks A., Vincent J., and Anyakoha C. (2007) A review of particle swarm optimization. part i: background and development. *Natural Computing* 6(4): 467–484.

[BYBS10] Birattari M., Yuan Z., Balaprakash P., and Stützle T. (2010) F-race and iterated F-race: An overview. In Bartz-Beielstein T., Chiarandini M., Paquete L., and Preuss M. (Eds.) *Experimental Methods for the Analysis of Optimization Algorithms*, pp. 311–336. Springer, Berlin, Germany.

[BZB+09] Brest J., Zamuda A., Boskovic B., Maucec M., and Zumer V. (2009) Dynamic optimization using self-adaptive differential evolution. In *Evolutionary Computation, 2009. CEC '09. IEEE Congress on*, pp. 415–422.

[CCN+07] Caponio A., Cascella G., Neri F., Salvatore N., and Sumner M. (2007) A fast adaptive memetic algorithm for online and offline control design of pmsm drives. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 37(1): 28–41.

[CF07] Cotta C. and Fernàndez A. (2007) Memetic algorithms in planning, scheduling, and timetabling. In Dahal K., Tan K., and Cowling P. (Eds.) *Evolutionary Scheduling*, volume 49 of *Studies in Computational Intelligence*, pp. 1–30. Springer Berlin / Heidelberg.

[COLT11] Chen X., Ong Y.-S., Lim M.-H., and Tan K. C. (oct. 2011) A multi-facet survey on memetic computation. *Evolutionary Computation, IEEE Transactions on* 15(5): 591 –607.

[DA95] Deb K. and Agrawal R. B. (1995) Simulated Binary Crossover for Continuous Search Space. *Complex Systems* 9: 115–148.

[DAJ02] Deb K., Anand A., and Joshi D. (Diciembre 2002) A computationally efficient evolutionary algorithm for real-parameter optimization. *Evolutionary Computation* 10(4): 371–395.

[Dar59] Darwin C. (1859) *On the Origin of Species by Means of Natural Selection*. Murray, London. or the Preservation of Favored Races in the Struggle for Life.

[Dav91] Davis L. (1991) *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York.

[Daw90] Dawkins R. (Septiembre 1990) *The Selfish Gene*. Oxford University Press.

[DGMH11] Derrac J., García S., Molina D., and Herrera F. (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation* 1(1): 3–18.

[DJ75] De Jong K. A. (1975) *An analysis of the behavior of a class of genetic adaptive systems.* PhD thesis, Ann Arbor, MI, USA. AAI7609381.

[DLLIS11a] Dubois-Lacoste J., López-Ibáñez M., and Stützle T. (2011) Automatic configuration of state-of-the-art multi-objective optimizers using the TP+PLS framework. In Krasnogor N. and Lanzi P. L. (Eds.) *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2011*, pp. 2019–2026. ACM Press, New York, NY.

[DLLIS11b] Dubois-Lacoste J., López-Ibáñez M., and Stützle T. (2011) A hybrid TP+PLS algorithm for bi-objective flow-shop scheduling problems. *Computers & Operations Research* 38(8): 1219–1236.

[DMQS11] Das S., Maity S., Qu B.-Y., and Suganthan P. (2011) Real-parameter evolutionary multimodal optimization — a survey of the state-of-the-art. *Swarm and Evolutionary Computation* 1(2): 71 – 88.

[DS11] Das S. and Suganthan P. (2011) Differential evolution: A survey of the state-of-the-art. *Evolutionary Computation, IEEE Transactions on* 15(1): 4–31.

[DS12] Deb K. and Saha A. (2012) Multimodal optimization using a bi-objective evolutionary algorithm. *Evolutionary Computation* 20(1): 27–62.

[DW11] Dick G. and Whigham P. (2011) Weighted local sharing and local clearing for multimodal optimisation. *Soft Computing* 15(9): 1707–1721.

[ECS89] Eshelman L. J., Caruana R. A., and Schaffer J. D. (1989) Biases in the crossover landscape. In *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 10–19. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[ELB13] Epitropakis M., Li X., and Burke E. (2013) A dynamic archive niching differential evolution algorithm for multimodal optimization. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pp. 79–86.

[EO09] Ellabaan M. and Ong Y.-S. (2009) Valley-adaptive clearing scheme for multimodal optimization evolutionary search. In *Intelligent Systems Design and Applications, 2009. ISDA '09. Ninth International Conference on*, pp. 1–6.

[EPV11] Epitropakis M., Plagianakos V., and Vrahatis M. (2011) Finding multiple global optima exploiting differential evolution's niching capability. In *Differential Evolution (SDE), 2011 IEEE Symposium on*, pp. 1–8.

[ES93] Eshelman L. J. and Schaffer J. D. (1993) Real-coded Genetic Algorithms in Genetic Algorithms by Preventing Incest. *Foundation of Genetic Algorithms 2* pp. 187–202.

[ES03] Eiben A. E. and Smith J. E. (2003) *Introduction to evolutionary computing*. springer.

[ESE08] Elfeky E., Sarker R., and Essam D. (2008) Analyzing the simple ranking and selection process for constrained evolutionary optimization. *Journal of Computer Science and Technology* 23(1): 19–34.

[ESE11] Elsayed S., Sarker R., and Essam D. (June 2011) Ga with a new multi-parent crossover for solving ieee-cec2011 competition problems. In *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pp. 1034–1040.

[FPC08] Fallahi A. E., Prins C., and Calvo R. W. (2008) A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. *Computers & Operations Research* 35(5): 1725 – 1741. Part Special Issue: Algorithms and Computational Methods in Feasibility and Infeasibility.

[GB89] Grefenstette J. J. and Baker J. E. (1989) How genetic algorithms work: A critical look at implicit parallelism. In *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 20–27. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[GD91] Goldberg D. E. and Deb K. (1991) A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of Genetic Algorithms*, pp. 69–93. Morgan Kaufmann.

[GK03] Glover F. and Kochenberger G. A. (2003) *Handbook of metaheuristics*. Springer.

[Gol89] Goldberg D. E. (1989) *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition.

[GR87] Goldberg D. E. and Richardson J. (1987) Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms on*

Genetic algorithms and their application, pp. 41–49. L. Erlbaum Associates Inc., Hillsdale, NJ, USA.

[GV99] Goldberg D. E. and Voessner S. (1999) Optimizing global-local search hybrids. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 220–228. Morgan Kaufmann.

[HAA13] Hansen N., Arnold D. V., and Auger A. (2013) Evolution Strategies. Technical report.

[HLS03] Herrera F., Lozano M., and Sánchez A. M. (2003) A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study. *International Journal of Intelligent Systems* 18(3): 309–338.

[HLV97] Herrera F., Lozano M., and Verdegay J. (1997) Fuzzy connectives based crossover operators to model genetic algorithms population diversity. *Fuzzy Sets and Systems* 92(1): 21 – 30.

[HLV98] Herrera F., Lozano M., and Verdegay J. (1998) Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis. *Artificial Intelligence Review* 12(4): 265–319.

[HMK03] Hansen N., Müller S. D., and Koumoutsakos P. (2003) Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). *Evolutionary Computation* 1(11): 1–18.

[HO01] Hansen N. and Ostermeier A. (Junio 2001) Completely derandomized self-adaptation in evolution strategies. *Evol. Comput.* 9(2): 159–195.

[Hol75] Holland J. (1975) *Adaptation in Natural and Artificial Systems.* The University of Michigan Press.

[IDG+12] Islam S. M., Das S., Ghosh S., Roy S., and Suganthan P. N. (Abril 2012) An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society* 42(2): 482–500.

[INM+12] Iacca G., Neri F., Mininno E., Ong Y.-S., and Lim M.-H. (Abril 2012) Ockham's Razor in memetic computing: Three stage optimal memetic exploration. *Information Sciences* 188: 17–43.

[IYM03] Ishibuchi H., Yoshida T., and Murata T. (april 2003) Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. *IEEE Transactions on Evolutionary Computation* 7(2): 204 – 223.

[KB07] Karaboga D. and Basturk B. (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. *Journal of Global Optimization* 39(3): 459–471.

[KE95] Kennedy J. and Eberhart R. (1995) Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 4, pp. 1942–1948 vol.4.

[KE01] Kennedy J. and Eberhart R. (2001) *Swarm Intelligence*. Morgan Kaufmann.

[KIP08] Kononova A., Ingham D., and Pourkashanian M. (2008) Simple scheduled memetic algorithm for inverse problems in higher dimensions: Application to chemical kinetics. In *Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on*, pp. 3905–3912.

[KS05] Krasnogor N. and Smith J. (oct. 2005) A tutorial for competent memetic algorithms: model, taxonomy, and design issues. *IEEE Transactions on Evolutionary Computation* 9(5): 474 – 488.

[KSC02] Kendall G., Soubeiga E., and Cowling P. (2002) Choice function and random hyperheuristics. In *Proceedings of the fourth Asia-Pacific Conference on Simulated Evolution And Learning, SEAL*, pp. 667–671. Springer.

[LBPC02] Li J.-P., Balazs M. E., Parks G. T., and Clarkson P. J. (Septiembre 2002) A species conserving genetic algorithm for multimodal function optimization. *Evol. Comput.* 10(3): 207–234.

[LEE13] Li X., Engelbrecht A., and Epitropakis M. (2013) Benchmark functions for cec 2013 special session and competition on niching methods for multimodal function optimization. Technical report, Royal Melbourne Institute of Technology.

[Li04] Li X. (2004) Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization. In Deb K. (Ed.) *Genetic and Evolutionary Computation – GECCO 2004*, volume 3102 of *Lecture Notes in Computer Science*, pp. 105–116. Springer Berlin Heidelberg.

[Li07] Li X. (2007) A multimodal particle swarm optimizer based on fitness euclidean-distance ratio. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, GECCO '07, pp. 78–85. ACM, New York, NY, USA.

[Li10] Li X. (2010) Niching without niching parameters: Particle swarm optimization using a ring topology. *Evolutionary Computation, IEEE Transactions on* 14(1): 150–169.

[LIDLSB11] López-Ibáñez M., Dubois-Lacoste J., Stützle T., and Birattari M. (2011) The irace package, iterated race for automatic algorithm configuration. Technical Report TR/IRIDIA/2011-004, IRIDIA, Université Libre de Bruxelles, Belgium.

[LM03] Laguna M. and Martí R. (2003) *Scatter Search – Methodology and Implementations in C.* Kluwer Academic Publishers, Boston.

[LMH11] Lozano M., Molina D., and Herrera F. (2011) Editorial scalability of evolutionary algorithms and other metaheuristics for large-scale continuous optimization problems. *Soft Computing* 15(11): 2085–2087.

[LOJS09] Le M., Ong Y.-S., Jin Y., and Sendhoff B. (2009) Lamarckian memetic algorithms: local optimum and connectivity structure analysis. *Memetic Computing* 1(3): 175–190.

[LP02] Liu Y. and Passino K. (2002) Biomimicry of social foraging bacteria for distributed optimization: Models, principles, and emergent behaviors. *Journal of Optimization Theory and Applications* 115(3): 603–628.

[LTGH07] Liu D., Tan K., Goh C., and Ho W. (feb. 2007) A multiobjective memetic algorithm based on particle swarm optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 37(1): 42 –50.

[Mah95] Mahfoud S. W. (1995) *Niching methods for genetic algorithms.* PhD thesis, Champaign, IL, USA. UMI Order No. GAX95-43663.

[MCG+10] Mendoza J. E., Castanier B., Guéret C., Medaglia A. L., and Velasco N. (2010) A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. *Computers & Operations Research* 37(11): 1886–1898.

[MG99] Mengshoel O. J. and Goldberg D. E. (July 13–17 1999) Probabilistic crowding: Deterministic crowding with probabilistic replacement. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*, pp. 409–416.

[Mic94] Michalewicz Z. (1994) *Genetic Algorithms + Data Structures = Evolution Programs (2Nd, Extended Ed.)*. Springer-Verlag New York, Inc., New York, NY, USA.

[MLGMH10] Molina D., Lozano M., García-Martínez C., and Herrera F. (Enero 2010) Memetic Algorithms for Continuous Optimisation Based on Local Search Chains. *Evolutionary Computation* 18(1): 27–63.

[MLSH10] Molina D., Lozano M., Sánchez A. M., and Herrera F. (Septiembre 2010) Memetic algorithms based on local search chains for large scale continuous optimisation problems: MA-SSW-Chains. *Soft Computing* 15(11): 2201–2220.

[MM97] Maron O. and Moore A. W. (1997) The racing algorithm: Model selection for lazy learners. *Artificial Intelligence Research* 11(1–5): 193–225.

[MNM96] Michalewicz Z., Nazhiyath G., and Michalewicz M. (1996) A note on usefulness of geometrical crossover for numerical optimization problems. In *Proc. 5th Annu. Conf. Evolutionary Programming*, pp. 305–312.

[Mos89] Moscato P. (1989) On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Toward Memetic Algorithms. Technical report.

[Mos03] Moscato P. (2003) *A gentle introduction to memetic algorithms*. Kluwer Academic Publishers.

[MSV93a] Mühlenbein H. and Schlierkamp-Voosen D. (1993) Predictive models for the breeding genetic algorithm in continuous parameter optimization. *Evolutionary Computation* 1: 25–49.

[MSV93b] Mühlenbein H. and Schlierkamp-Voosen D. (1993) Predictive models for the breeder genetic algorithm – i. continuous parameter optimization. *EVOLUTIONARY COMPUTATION* 1: 25–49.

[NC12] Neri F. and Cotta C. (2012) Memetic algorithms and memetic computing optimization: A literature review. *Swarm and Evolutionary Computation* 2(0): 1 – 14.

[NM10] Neri F. and Mininno E. (may 2010) Memetic compact differential evolution for cartesian robot control. *Computational Intelligence Magazine, IEEE* 5(2): 54 –65.

[NOL09] Nguyen Q. H., Ong Y.-S., and Lim M.-H. (2009) A probabilistic memetic framework. *Evolutionary Computation, IEEE Transactions on* 13(3): 604–623.

[NTM07a] Neri F., Toivanen J., and Mäkinen R. A. (December 2007) An adaptive evolutionary algorithm with intelligent mutation local searchers for designing multidrug therapies for hiv. *Applied Intelligence* 27: 219–235.

[NTM07b] Neri F., Toivanen J., and Mäkinen R. (2007) An adaptive evolutionary algorithm with intelligent mutation local searchers for designing multidrug therapies for hiv. *Applied Intelligence* 27(3): 219–235.

[OK04] Ong Y.-S. and Keane A. (2004) Meta-lamarckian learning in memetic algorithms. *Evolutionary Computation, IEEE Transactions on* 8(2): 99–110.

[OKI07] Ong Y.-S., Krasnogor N., and Ishibuchi H. (feb. 2007) Special issue on memetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 37(1): 2 –5.

[OKK03] Ono I., Kita H., and Kobayashi S. (2003) A real-coded genetic algorithm using the unimodal normal distribution crossover. In Ghosh A. and Tsutsui S. (Eds.) *Advances in Evolutionary Computing*, Natural Computing Series, pp. 213–237. Springer Berlin Heidelberg.

[OLC10] Ong Y.-S., Lim M., and Chen X. (may 2010) Research frontier: Memetic computation - past, present & future. *Computational Intelligence Magazine, IEEE* 5(2): 24 –31.

[OLZW06] Ong Y.-S., Lim M.-H., Zhu N., and Wong K.-W. (2006) Classification of adaptive memetic algorithms: a comparative study. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 36(1): 141–152.

[Pet96] Petrowski A. (1996) A clearing procedure as a niching method for genetic algorithms. In *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, pp. 798–803.

[QGS10] Qu B., Gouthanan P., and Suganthan P. (2010) Dynamic grouping crowding differential evolution with ensemble of parameters for multi-modal optimization. In Panigrahi B., Das S., Suganthan P., and Dash S. (Eds.) *Swarm, Evolutionary, and Memetic Computing*, volume 6466 of *Lecture Notes in Computer Science*, pp. 19–28. Springer Berlin Heidelberg.

[QGZQ08] Qing L., Gang W., Zaiyue Y., and Qiuping W. (2008) Crowding clustering genetic algorithm for multimodal function optimization. *Applied Soft Computing* 8(1): 88 – 95.

[QLS12] Qu B., Liang J., and Suganthan P. (2012) Niching particle swarm optimization with local search for multi-modal optimization. *Information Sciences* 197(0): 131 – 143.

[QLSC12] Qu B., Liang J., Suganthan P., and Chen T. (2012) Ensemble of clearing differential evolution for multi-modal optimization. In Tan Y., Shi Y., and Ji Z. (Eds.) *Advances in Swarm Intelligence*, volume 7331 of *Lecture Notes in Computer Science*, pp. 350–357. Springer Berlin Heidelberg.

[QS05] Qin A. K. and Suganthan P. (2005) Self-adaptive differential evolution algorithm for numerical optimization. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 2, pp. 1785–1791 Vol. 2.

[QSL12] Qu B. Y., Suganthan P., and Liang J. (2012) Differential evolution with neighborhood mutation for multimodal optimization. *Evolutionary Computation, IEEE Transactions on* 16(5): 601–614.

[RD00] Rogalsky T. and Derksen R. W. (2000) Hybridization of differential evolution for aerodynamic design. In *Proceedings of the 8th Annual Conference of the Computational Fluid Dynamics Society of Canada*, pp. 729–736.

[R Development Core Team08] R Development Core Team (2008) *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria.

[Rec65] Rechenberg I. (1965) Cybernetic solution path of an experimental problem. Technical report, Royal Air Force Establishment.

[Sch81] Schwefel H.-P. (1981) *Numerical Optimization of Computer Models.* John Wiley & Sons, Inc., New York, NY, USA.

[Sch89] Schaffer J. D. (Ed.) (1989) *The GENITOR Algorithm and Selection Pressure: Why Rank-Based Allocation of Reproductive Trials is Best.* Morgan Kaufmann Publishers, San Mateo.

[SHL+05] Suganthan P. N., Hansen N., Liang J. J., Deb K., Chen Y. P., Auger A., and Tiwari S. (2005) Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization. Technical report.

[SK98] Sareni B. and Krahenbuhl L. (1998) Fitness sharing and niching methods revisited. *Evolutionary Computation, IEEE Transactions on* 2(3): 97–106.

[SK02] Schaefer R. and Kolodziej J. (2002) Genetic search reinforced by the population hierarchy. In *FOGA*, volume 7, pp. 383–401.

[Smi07] Smith J. (2007) Coevolving memetic algorithms: A review and progress report. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 37(1): 6–17.

[SMPS04] Sacco W. F., Machado M. D., Pereira C. M., and Schirru R. (2004) The fuzzy clearing approach for a niching genetic algorithm applied to a nuclear reactor core design optimization problem. *Annals of Nuclear Energy* 31(1): 55 – 69.

[SP97] Storn R. and Price K. (1997) Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization* 11(4): 341–359.

[SW81] Solis F. and Wets R. (1981) Minimization by random search techniques. *Mathematics of operations research* 6(1): 19–30.

[Sys89] Syswerda G. (1989) Uniform crossover in genetic algorithms. In *Proceedings of the 3rd International Conference on Genetic Algorithms*, pp. 2–9. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[Tho04] Thomsen R. (2004) Multimodal optimization using crowding-based differential evolution. In *Evolutionary Computation, 2004. CEC2004. Congress on*, volume 2, pp. 1382–1389 Vol.2.

[TL10] Ting C.-K. and Liao C.-C. (2010) A memetic algorithm for extending wireless sensor network lifetime. *Information Sciences* 180(24): 4818 – 4833.

[Tom05] Tomassini M. (2005) *Spatially Structured Evolutionary Algorithms*. Springer.

[TY07] Tang M. and Yao X. (February 2007) A memetic algorithm for vlsi floorplanning. *IEEE Transactions On Systems, Man, And Cybernetics—Part B: Cybernetics* 37(1): 62–69.

[TYH99] Tsutsui S., Yamamura M., and Higuchi T. (13-17 July 1999) Multi-parent recombination with simplex crossover in real coded genetic algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 1, pp. 657–664. Morgan Kaufmann.

[Urs02] Ursem R. K. (2002) Diversity-Guided Evolutionary Algorithms. In *Proceedings of the Congress on Evolutionary Computation*, pp. 1633–1640. IEEE Press.

[VC08] Vitela J. E. and Castanos O. (2008) A real-coded niching memetic algorithm for continuous multimodal function optimization. In *Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on*, pp. 2170–2177.

[VC12] Vitela J. E. and Castaños O. (2012) A sequential niching memetic algorithm for continuous multimodal function optimization. *Applied Mathematics and Computation* 218(17): 8242 – 8259.

[VF96] Vavak F. and Fogarty T. (1996) A comparative study of steady state and generational genetic algorithms for use in nonstationary environments. In Fogarty T. (Ed.) *Evolutionary Computing*, volume 1143 of *Lecture Notes in Computer Science*, pp. 297–304. Springer Berlin Heidelberg.

[WMYW12] Wang H., Moon I., Yang S., and Wang D. (2012) A memetic particle swarm optimization algorithm for multimodal optimization problems. *Information Sciences* 197(0): 38 – 52.

[WRH97] Whitley D., Rana S., and Heckendorn R. B. (1997) Island model genetic algorithms and linearly separable problems. In *Evolutionary computing*, pp. 109–125. Springer.

[Wri91] Wright A. H. (1991) Genetic algorithms for real parameter optimization. In *Foundations of Genetic Algorithms*, pp. 205–218. Morgan Kaufmann.

[Zah04] Zaharie D. (2004) Extensions of differential evolution algorithms for multimodal optimization. In *Proceedings of SYNASC*, pp. 523–534.

[ZL11] Zhai Z. and Li X. (2011) A dynamic archive based niching particle swarm optimizer using a small population size. In *Proceedings of the Thirty-Fourth Australasian Computer Science Conference - Volume 113*, ACSC '11, pp. 83–90. Australian Computer Society, Inc., Darlinghurst, Australia, Australia.

[ZS09] Zhang J. and Sanderson A. (2009) Jade: Adaptive differential evolution with optional external archive. *Evolutionary Computation, IEEE Transactions on* 13(5): 945–958.