compAS
Grupo de capacitación e investigación pedagogica

Metaheurísticas de optimización multiobjetivo aplicadas a la inferencia filogenética y al alineamiento múltiple de secuencias

Cristian Gabriel Zambrano Vega
Antonio Nebro Urbaneja
José Francisco Aldana Montes

Metaheurísticas de optimización multiobjetivo aplicadas a la inferencia filogenética y al alineamiento múltiple de secuencias

Cristian Gabriel Zambrano Vega
Antonio Nebro Urbaneja
José Francisco Aldana Montes

Metaheurísticas de optimización multiobjetivo aplicadas a la inferencia filogenética y al alineamiento múltiple de secuencias

compAs
Grupo de capacitación e investigación pedagógica

Título original: Metaheurísticas de optimización multiobjetivo aplicadas a la inferencia filogenética y al alineamiento múltiple de secuencias

Editado en Guayaquil - Ecuador

ISBN: 978-9942-33-268-4

compÁs
Grupo de capacitación e investigación pedagógica

Cita.
Zambrano. C, Nebro. A, Aldana. J. (2020) Metaheurísticas de optimización multiobjetivo aplicadas a la inferencia filogenética y al alineamiento múltiple de secuencias, Editorial Compás, Guayaquil Ecuador, 188 pag

Índice de Tablas

Índice de figuras

Capítulo 1

Introducción

Este capítulo inicial presenta los principales temas que han motivado la investigación de esta tesis doctoral y describe los principales objetivos y las fases que han sido llevadas a cabo a lo largo de su desarrollo. También incluye las principales contribuciones y la organización de todo este documento.

1.1. Motivación

El interés de la investigación en resolver problemas computacionalmente exigentes y de dominio biológico ha crecido significativamente en la última década. Los avances en la secuenciación molecular y el surgimiento de tecnologías como The Next-Generation Sequencing (NGS) (Shendure and Ji, 2008) han llevado a la generación de cantidades cada vez mayores de datos biológicos cuyo procesamiento representa un desafío significante tanto desde la perspectiva biológica como computacional.

El campo de investigación de la Bioinformática incluye una amplia gama de aplicaciones basadas en el modelado de procesos biológicos de alta complejidad. Tales procesos a menudo implican procedimientos de optimización que buscan encontrar los resultados más satisfactorios de acuerdo con uno o más criterios de calidad que evalúan su relevancia biológica (Pal et al, 2006). Muchos de estos problemas biológicos muestran una naturaleza de complejidad NP-Completa, lo que hace muy difícil la generación de soluciones óptimas o de alta calidad bajo técnicas algorítmicas clásicas. Esta tesis doctoral está centrada en dos problemas de este tipo: la Inferencia Filogenética y el Alineamiento Múltiple de Secuencias o MSA (según sus siglas en inglés: *Multiple Sequence Alignment*).

La historia evolutiva de las especies vivientes y extinguidas en la tierra es una cuestión que ha sido intrigante para la humanidad durante siglos y, la construcción del "árbol de la vida" que incluya a todos ellas ha sido una idea fascinante y desafiante desde el surgimiento de la teoría de la evolución propuesta por Charles Darwin (Darwin et al, 1872). Actualmente la inferencia de este árbol de la vida es considerado como uno de los grandes retos de la Bioinformática (Stamatakis, 2004). La Filogenética es el estudio de las relaciones evolutivas entre organismos, las cuales son representadas en un árbol filogenético que indica sus descendencias. La inferencia filogenética consiste en encontrar el árbol evolutivo que describa lo más exactamente posible las relaciones genealógicas o la historia evolutiva de un conjunto de especies a partir de sus secuencias moleculares.

Actualmente es ampliamente aceptado entre los biólogos que la comprensión de la evolución de un grupo de organismos requiere una comprensión de sus relaciones filogenéticas. Además de su importancia en los estudios sistemáticos, los métodos filogenéticos presentan otras aplicaciones,

como la predicción de la evolución de las enfermedades infecciosas (Bush et al, 1999), el descubrimiento de nuevos fármacos (Brown and Warren, 1998), la planificación de la conservación de prioridades (Faith, 1994), e incluso forenses (Ou et al, 1992).

Varias investigaciones (Huelsenbeck, 1995; Kuhner and Felsenstein, 1994; Tateno et al, 1994) han mostrado importantes diferencias en sus resultados aplicando distintos métodos de reconstrucción a los mismos conjunto de secuencias. Otras dificultades se dan por el hecho de que diferentes criterios modelan diferentes principios sobre la forma en que las especies evolucionan en la naturaleza. Es decir, diferentes métodos hacen sus propios supuestos sobre el proceso evolutivo y, por lo tanto, a menudo dan lugar a explicaciones conflictivas sobre la evolución de las especies bajo estudio, según lo informado por múltiples trabajos en la literatura (Burleigh and Mathews, 2007; Chippindale et al, 2004; Macey, 2005; Schmidt et al, 2010). Por consecuencia, la aplicación de la optimización monoobjetivo a menudo no es adecuada para hacer frente a estos problemas, ya que diferentes criterios pueden conducir a soluciones conflictivas para un mismo conjunto de datos biológicos.

Un requisito primordial para la inferencia de un árbol filogenético es que las secuencias de entrada deben estar alineadas, y la calidad de tal alineamiento influye directamente en la calidad de árbol inferido. Por este motivo el segundo tema abordado en nuestra investigación es el Alineamiento Múltiple de Secuencias o MSA, que, como se ha mencionado anteriormente, es también uno de los principales tópicos de interés dentro del campo de la BioInformática (Pei, 2008).

El objetivo principal del alineamiento múltiple de secuencias es la de representar y comparar dos o más secuencias biológicas (ADN, ARN o proteínas), para resaltar la mayor cantidad de zonas de similitud o zonas conservadas entre ellas, las cuales podrían indicar relaciones evolutivas o funcionales entre los genes o proteínas consultadas. Además de la filogenética, los resultados de la calidad de los alineamientos influyen en otros procesos bioinformáticos como son el modelado estructural y la predicción funcionalidad de proteínas.

A pesar de la existencia de numerosas herramientas para el alineamiento múltiple de secuencias, todavía no se dispone de un estándar apropiado para construir los alineamientos. Como consecuencia, cada herramienta genera un alineamiento que puede diferir notablemente del generado por otra, debido a la aplicación de sus propios criterios. La evaluación de los alineamientos también genera un problema adicional. Se han propuesto una serie de métricas diferentes para medir la precisión y calidad de los alineamientos, tales como el porcentaje de columnas totalmente alineadas (TC), el porcentaje de caracteres -No Gaps- (NonGapsP), la Suma de pares (*Sum-of-Pairs*, SOP), la suma ponderada de pares con penalidad de gaps afines (*weighted Sum-of-Pairs*, wSOP), Strike (Kemena et al, 2011), Entropy (Soto and Becerra, 2014), BAliScore (Soto and Becerra, 2014) o MetAl (Blackburne and Whelan, 2012a). Sin embargo, todavía no existe un consenso acerca de qué métrica es la mas apropiada o la mas precisa para medir la calidad de los alineamientos. Por este motivo se tiende a evaluarlos aplicando los sistemas clásicos de evaluación lo que puede conllevar a alineamientos no suficientemente precisos. Así, la mejora de estos sistemas de evaluación mediante la incorporación de información complementaria podría contribuir a la mejora del análisis de calidad y a la obtención de herramientas de alineamiento más eficientes.

Tanto la inferencia filogenética como el alineamiento múltiple de secuencias son problemas NP-Completos. En el caso de la inferencia filogenética, el espacio de árboles filogenéticos ("tree-space") crece de forma exponencial por cada una de las especies del conjunto de secuencias a considerar (Poladian and Jermiin, 2005); en el caso del alineamiento, el espacio de búsqueda se incrementa exponencialmente según el número de secuencias a alinear y la longitud de las mismas (Waterman et al, 1976). Para abordar esta problemática, en esta tesis se ha hecho uso de una familia de técnicas conocidas como metaheurísticas, que constituyen una familia de algoritmos de optimización no exactos que son capaces en general de proporcionar soluciones de alta calidad a problemas de optimización complejos. Una metaheurística se puede definir como una estrategia de alto nivel que combina una serie de técnicas heurísticas más simples enfocadas en buscar el óptimo

de un problema (Blum and Roli, 2003).

Para tratar la problemática de los diferentes criterios de calidad que se pueden aplicar tanto a la inferencia filogenética como al alineamiento múltiple de secuencias se ha optado por formular ambos problemas de acuerdo a un enfoque multiobjetivo. De esta forma, en lugar de guiar las búsquedas en base a un único criterio u objetivo se consideran dos o más al mismo tiempo, lo que permite producir un conjunto de soluciones de compromiso entre los distintos objetivos, de forma que al experto en el problema se le ofrece un abanico de alternativas que puede analizar para a partir de ahí seleccionar la solución o soluciones que estime más oportunas.

1.2. Objetivos y Fases

La motivación principal de esta tesis es estudiar la aplicación de metaheurísticas de optimización multiobjetivo para abordar la resolución de problemas de optimización tanto de inferencia filogenética como de MSA. En el estudio realizado en Handl et al (2007) se señalaron ambos problemas biológicos como casos en los que la aplicación de metaheurísticas multiobjetivo era particularmente prometedora. Se pretende no sólo estudiar los resultados que algoritmos multiobjetivo representativos del estado del arte pueden producir al aplicarlos a ambos problemas, sino también proponer mejoras algorítmicas. Un aspecto muy importante en esta tesis es el ofrecer herramientas software de código abierto que incluyan todos los algoritmos y técnicas utilizadas, que se alojarán en repositorios públicos con el fin de facilitar el acceso a las mismas a los investigadores interesados en usarlas.

En esta tesis se han seguido un conjunto de pasos bien definidos para llevar a cabo una investigación científica, que se enumeran a continuación:

1. **Estudio de fundamentos de los problemas de Inferencia Filogenética y de Alineamiento Múltiple de Secuencias.** Aquí se incluyen los conceptos básicos relacionados a ambos problemas, sus bases biológicas y los aspectos a tener en cuenta para resolverlos como problemas de optimización. En esta etapa se aborda también el estudio y análisis de las estrategias a tener en cuenta para poder resolver el problema desde un enfoque de optimización multiobjetivo.

2. **Estudio del estado del arte.** En esta fase nos centramos a realizar un estudio bibliográfico de las técnicas de optimización aplicadas a la resolución de los problemas de la Inferencia Filogenética y al Alineamiento Múltiple de Secuencias, pero principalmente con un especial interés en aquellas que abordaron un enfoque multiobjetivo a los problemas.

3. **Análisis, diseño e implementación de herramientas de optimización multiobjetivo para la Inferencia Filogenética y al Alineamiento Múltiple de Secuencias.** El paso anterior mostrará que apenas existen implementaciones públicamente disponibles de las técnicas y algoritmos presentados en la literatura, lo que dificulta la utilización práctica de los mismos. Conscientes de la importancia de que el trabajo de investigación a realizar en esta tesis pueda ser no sólo replicado sino también reutilizado por otros investigadores, se plantea como siguiente paso el diseño e implementación de herramientas software con todo el código que se vaya a desarrollar y utilizar. Se crearán para tal fin proyectos de código abierto que estarán alojados en GitHub (`https://github.com`), que incluirán no sólo el software sino la documentación para usarlo así como los conjuntos de datos que se analizarán.

4. **Análisis experimental del rendimiento de metaheurísticas multiobjetivo aplicadas a la Inferencia Filogenética y al Alineamiento Múltiple de Secuencias.** Con el objetivo de conocer el rendimiento de las principales y recientes referencias metaheurísticas

multiobjetivose plantea realizar estudios basados en metodologías experimentales rigurosas, que incluirán la aplicación de indicadores calidad multiobjetivo y la aplicación de técnicas estadísticas que han de permitir proporcionar conclusiones confiables.

5. **Implementación de nuevas propuestas algorítmicas multiobjetivo aplicadas a la Inferencia Filogenética y al Alineamiento Múltiple de Secuencias.** El estudio experimental del paso anterior dará pie a la presentación de nuevas propuestas algorítmicas, para lo que se tendrán en cuenta tanto la inclusión de técnicas que permitan mejorar las capacidades de búsqueda de los algoritmos como el uso de paralelismo para aprovechar los recursos computacionales que ofrecen hoy día los procesadores multi-núcleo.

1.3. Contribuciones de la Tesis

Las contribuciones de la tesis giran en torno a la aplicación de técnicas metaheurísticas multiobjetivo a dos de los principales problemas de optimización en el área de la BioInformática, la Inferencia Filogenética y al Alineamiento Múltiple de Secuencias. Entre las contribuciones podemos indicar las siguientes:

1. Análisis comparativo de metaheurísticas multiobjetivo representativas del estado del arte aplicadas al Alineamiento Múltiple de Secuencias, bajo formulaciones de dos y tres objetivos de calidad. Los resultados llevaron a tomar decisiones sobre qué metaheurística escoger para plantear una nueva propuesta algorítmica al problema.

2. Desarrollo de la herramienta informática MO-Phylogenetics, un software de optimización multiobjetivo para la Inferencia Filogenética. Incluye un conjunto de algoritmos multiobjetivos referentes y recientes del estado del arte adaptados al problema. Está basado en las siguientes herramientas: jMetalCpp (Optimización MultiObjetivo), Bio++ (Funciones BioInformáticas) y PLL (Librería Filogenética). La adaptación de los algoritmos contienen tecnologías recientes y estrategias de exploración rápidas y eficaces del espacio de búsqueda de árboles filogenéticos aplicando movimientos topológicos y con evaluaciones parciales, provistas por 'core' de dos los principales software filogenéticos *RAxML* para el criterio de la Máxima Verosimilitud e *Hydra* para el criterio de la Máxima Parsimonia. El software se encuentra publicado en un repositorio público cuyo código fuente es de libre acceso para la comunidad científica.

3. Desarrollo del software de optimización multiobjetivo para el Alineamiento Múltiple de Secuencias, jMetalMSA, que está basado en el framework de optimización multiobjetivo jMetal y provee un conjunto diverso de sus algoritmos adaptados al problema. Incluye siete métricas de calidad implementadas como objetivos, algunas basadas en información estructural de las secuencias y otras basadas en medir la conservación y calidad de los alineamientos

4. Implementación de una propuesta algorítmica multiobjetivo para resolver el problema de la Inferencia Filogenética, denominada MORPHY. Está desarrollado usando las funcionalidades del framework MO-Phylogenetics y basado en la clásica referencia multiobjetivo NSGA-II. A diferencia de todas las propuestas publicadas en el estado del arte, MORPHY trabaja con secuencias ADN (nucléotidos) y Proteínas (Amino-ácidos) implementado además del Modelo de Sustitución GTR $\cdot\ \gamma$ un conjunto amplio de Modelos para proteínas (LG, WAG, JTT y otros). Los objetivos a optimizar del algoritmo son la Máxima Parsimonia y la Máxima Verosimilitud.

5. Implementación de una propuesta algorítmica multiobjetivo para resolver el problema del Alineamiento Múltiple de Secuencias, denominada M2Align. Está desarrollado usando las funcionalidades del framework jMetalMSA y basado en la clásica referencia multiobjetivo NSGA-II, debido a que los resultados obtenidos en el análisis de rendimiento indican que es la que mejor rendimiento provee tanto bajo una formulación de dos como de tres objetivos. Implementa una representación rápida de las secuencias alineadas basadas solo en la información de los *gaps*. Los objetivos a optimizar de M2Align son tres, uno basado en la información estructural de las secuencias el cual nos garantiza la obtención de alineamientos más precisos en los casos de secuencias menos relacionadas evolutivamente y dos basados en la conservación de la estructura de las secuencias. Considerando el alto coste computacional requerido para el proceso de inferencia de un alineamiento óptimo para un grandes conjuntos de secuencias y de gran tamaño, se ha incluído funcionales de paralelismo que permiten aprovechar las ventajas que brindan los sistemas de hardware multi-core, permitiendo un importante incremento de los speed-ups del algoritmo usando un conjunto de 4, 10, y 20 cores.

Los resultados científicos de esta tesis han sido publicados en tres artículos de revista indexadas en el JRC de primer cuartil, (Zambrano-Vega et al, 2016) (Q1), (Zambrano-Vega et al, 2017d) (Q1) y (Zambrano-Vega et al, 2017a) (Q1), una revista no JCR (Zambrano-Vega et al, 2017b), dos artículos de congreso ESCIM 2015 (Nebro et al, 2015b) e iWBBIO 2017 (Zambrano-Vega et al, 2017c) (ver Apéndice A).

1.4. Organización de la Tesis

Esta memoria se estructura en los siguientes capítulos que se enumeran a continuación. El presente capítulo contiene una introducción al trabajo realizado, introduciendo la motivación para llevarlo a cabo, los objetivos que se han pretendido alcanzar, las fases que se han seguido para conseguir dichos objetivos y las contribuciones de la tesis.

El capítulo 2 se introducen los fundamentos tanto de las técnicas metaheurísticas como de optimización multiobjetivo. Se incluye una clasificación de las primeras y se detalla el procedimiento de evaluación estadística que se ha seguido en los estudios experimentales.

El capítulo 3 está dedicado al problema de la Inferencia Filogenética, incluyendo la definición del mismo, su formulación multiobjetivo, el estudio experimental con técnicas del estado del arte, la herramienta software que se ha desarrollado y una propuesta algorítmica.

El capítulo 4 está centrado en el problema del Alineamiento Múltiple de Secuencias, y su estructura es idéntica al capítulo anterior.

El capítulo 5 recoge las conclusiones de la tesis y las líneas que han quedado abiertas para posteriores trabajos.

El apéndice A lista las publicaciones que avalan la tesis doctoral.

El apéndice B contiene tablas de resultados que no se han incluido en los capítulos precedentes.

Los apéndices C y D enumeran los parámetros de las herramientas software desarrolladas.

Capítulo 2

Metaheurísticas

En esta sección dedicamos a establecer los fundamentos necesarios sobre los algoritmos de optimización utilizados para abordar los problemas del campo de las ingeniería civil correspondientes a tres de estructuras de distintas complejidades. Partiremos de un planteamiento clásico de optimización para definir el concepto de metaheurística y establecer su clasificación. A continuación se introducen conceptos de optimización multiobjetivo, puesto que tratamos con problemas de ingeniería donde existen varias funciones que se han de optimizar a la vez. Estos problemas no sólo proceden del mundo real, sino que además estamos considerando instancias reales (de gran tamaño) que suponen tareas con enormes necesidades de cómputo. Para uno de los problemas hemos utilizado técnicas paralelas para afrontar este inconveniente. Esto nos lleva al siguiente bloque de este capítulo, los modelos paralelos para metaheurísticas como medio para reducir los tiempos de ejecución. Por último, terminamos con el procedimiento estadístico seguido para evaluar metaheurísticas, donde se presentan las principales medidas de rendimiento así como los indicadores de calidad utilizados tanto en problemas de optimización monoobjetivo como multiobjetivo.

2.1. Definición de metaheurística

La optimización en el sentido de encontrar la mejor solución, o al menos una solución lo suficientemente buena, para un problema es un campo de vital importancia en el mundo real y, en particular, en ingeniería. Constantemente estamos resolviendo pequeños problemas de optimización, como el camino más corto para ir de un lugar a otro, la organización de una agenda, etc. En general, estos problemas son lo suficientemente pequeños y podemos resolverlos sin ayuda adicional, pero conforme se hacen más grandes y complejos, el uso de los ordenadores para su resolución es inevitable.

Comenzaremos este capítulo dando una definición formal del concepto de optimización. Asumiendo, sin perdida de generalidad, el caso de la minimización, podemos definir un *problema de optimización* como sigue:

Definición 1 (Problema de optimización). *Un problema de optimización se formaliza como un par* (S, f)*, donde* $S \neq \emptyset$ *representa el espacio de soluciones (o de búsqueda) del problema, mientras que* f *es una función denominada* función objetivo *o* función de fitness*, que se define como:*

$$f : S \rightarrow \mathbb{R} \ . \tag{2.1}$$

Así, resolver un problema de optimización consiste en encontrar una solución, $i^* \in S$*, que satisfaga la siguiente desigualdad:*

$$f(i^*) \leq f(i), \quad \forall \ i \in S \ . \tag{2.2}$$

Asumir el caso de maximización o minimización no restringe la generalidad de los resultados, puesto que se puede establecer una igualdad entre tipos de problemas de maximización y minimización de la siguiente forma (Bäck, 1996; Goldberg, 1989):

$$\text{máx}\{f(i)|i \in S\} \equiv \text{mín}\{-f(i)|i \in S\} \ . \tag{2.3}$$

En función del dominio al que pertenezca S, podemos definir problemas de *optimización binaria* ($S \subseteq \mathbb{B}^*$), *entera* ($S \subseteq \mathbb{N}^*$), *continua*($S \subseteq \mathbb{R}^*$), o *heterogénea* ($S \subseteq (\mathbb{B} \cup \mathbb{N} \cup \mathbb{R})^*$).

Debido a la gran importancia de los problemas de optimización, a lo largo de la historia de la Informática se han desarrollado múltiples métodos para tratar de resolverlos. Una clasificación muy simple de estos métodos se muestra en la Figura 2.1. Inicialmente, las técnicas las podemos clasificar en exactas (o enumerativas, exhaustivas, etc.) y aproximadas. Las técnicas exactas garantizan encontrar la solución óptima para cualquier instancia de cualquier problema en un tiempo acotado. El inconveniente de estos métodos es que el tiempo y/o memoria que se necesitan, aunque acotados, crecen exponencialmente con el tamaño del problema, ya que la mayoría de éstos son NP-duros. Esto supone en muchos casos que el uso de estas técnicas sea inviable, ya que se requiere mucho tiempo (posiblemente miles de años) y/o una cantidad desorbitada de memoria para la resolución del problema. Por lo tanto, los algoritmos aproximados para resolver estos problemas están recibiendo una atención cada vez mayor por parte de la comunidad internacional desde hace unas décadas. Estos métodos sacrifican la garantía de encontrar el óptimo a cambio de encontrar una solución satisfactoria en un tiempo razonable.

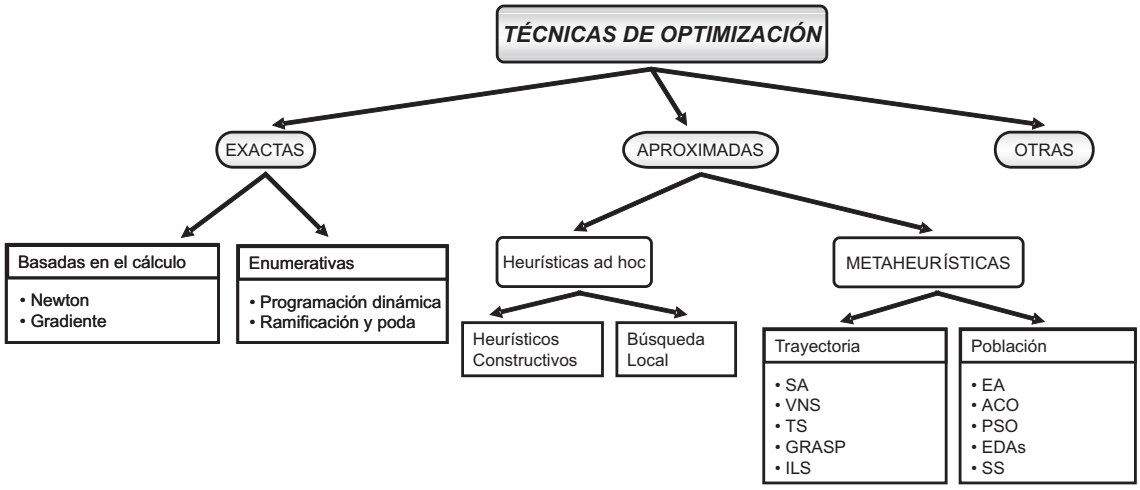


Figura 2.1: Clasificación de las técnicas de optimización.

Dentro de los algoritmos aproximados se pueden encontrar dos tipos: los heurísticos *ad hoc* y las metaheurísticas (en las que nos centramos en este capítulo). Los heurísticos *ad hoc*, a su vez, pueden dividirse en *heurísticos constructivos* y *métodos de búsqueda local*.

Los heurísticos constructivos suelen ser los métodos más rápidos. Construyen una solución desde cero mediante la incorporación de componentes hasta obtener una solución completa, que es el resultado del algoritmo. Aunque en muchos casos encontrar un heurístico constructivo es relativamente sencillo, las soluciones ofrecidas suelen ser de muy baja calidad. Encontrar métodos

de esta clase que produzca buenas soluciones es muy difícil, ya que dependen mucho del problema, y para su planteamiento se debe tener un conocimiento muy extenso del mismo. Por ejemplo, en problemas con muchas restricciones puede que la mayoría de las soluciones parciales sólo conduzcan a soluciones no factibles.

Los métodos de búsqueda local o seguimiento del gradiente parten de una solución ya completa y, usando el concepto de *vecindario*, recorren parte del espacio de búsqueda hasta encontrar un *óptimo local*. El vecindario de una solución s, que denotamos con $N(s)$, es el conjunto de soluciones que se pueden construir a partir de s aplicando un operador específico de modificación (generalmente denominado *movimiento*). Un óptimo local es una solución mejor o igual que cualquier otra solución de su vecindario. Estos métodos, partiendo de una solución inicial, examinan su vecindario y se quedan con el mejor vecino, continuando el proceso hasta que encuentran un óptimo local. En muchos casos, la exploración completa del vecindario es inabordable y se siguen diversas estrategias, dando lugar a diferentes variaciones del esquema genérico. Según el operador de movimiento elegido, el vecindario cambia y el modo de explorar el espacio de búsqueda también, pudiendo simplificarse o complicarse el proceso de búsqueda.

Finalmente, en los años setenta surgió una nueva clase de algoritmos aproximados, cuya idea básica era combinar diferentes métodos heurísticos a un nivel más alto para conseguir una exploración del espacio de búsqueda de forma eficiente y efectiva. Estas técnicas se han denominado *metaheurísticas*. Este término fue introducido por primera vez por Glover (Glover, 1986). Antes de que el término fuese aceptado completamente por la comunidad científica, estas técnicas eran denominadas *heurísticas modernas* (Reeves, 1993). Esta clase de algoritmos incluye técnicas como colonias de hormigas, algoritmos evolutivos, búsqueda local iterada, enfriamiento simulado y búsqueda tabú. Se pueden encontrar revisiones de metaheurísticas en (Blum and Roli, 2003; Glover and Kochenberger, 2003). De las diferentes descripciones de metaheurísticas que se encuentran en la literatura se pueden destacar ciertas propiedades fundamentales que caracterizan a este tipo de métodos:

- Las metaheurísticas son estrategias o plantillas generales que guían el proceso de búsqueda.
- El objetivo es una exploración eficiente del espacio de búsqueda para encontrar soluciones (casi) óptimas.
- Las metaheurísticas son algoritmos no exactos y generalmente son no deterministas.
- Pueden incorporar mecanismos para evitar regiones no prometedoras del espacio de búsqueda.
- El esquema básico de cualquier metaheurística tiene una estructura predefinida.
- Las metaheurísticas pueden hacer uso de conocimiento del problema que se trata de resolver en forma de heurísticos específicos que son controlados por la estrategia de más alto nivel.

Resumiendo estos puntos, se puede acordar que una metaheurística es una estrategia de alto nivel que usa diferentes métodos para explorar el espacio de búsqueda. En otras palabras, una metaheurística es una plantilla general no determinista que debe ser rellenada con datos específicos del problema (representación de las soluciones, operadores para manipularlas, etc.) y que permiten abordar problemas con espacios de búsqueda de gran tamaño. En este tipo de técnicas es especialmente importante el correcto equilibrio (generalmente dinámico) que haya entre *diversificación* e *intensificación*. El término diversificación se refiere a la evaluación de soluciones en regiones distantes del espacio de búsqueda (de acuerdo a una distancia previamente definida entre soluciones); también se conoce como *exploración* del espacio de búsqueda. El término intensificación, por otro lado, se refiere a la evaluación de soluciones en regiones acotadas y pequeñas con respecto al espacio de búsqueda centradas en el vecindario de soluciones concretas (*explotación* del espacio de

búsqueda). El equilibrio entre estos dos aspectos contrapuestos es de gran importancia, ya que por un lado deben identificarse rápidamente las regiones prometedoras del espacio de búsqueda global y por otro lado no se debe malgastar tiempo en las regiones que ya han sido exploradas o que no contienen soluciones de alta calidad.

Dentro de las metaheurísticas podemos distinguir dos tipos de estrategias de búsqueda. Por un lado, tenemos las extensiones "inteligentes" de los métodos de búsqueda local (metaheurísticas basadas en trayectoria en la Figura 2.1). La meta de estas estrategias es evitar de alguna forma los mínimos locales y moverse a otras regiones prometedoras del espacio de búsqueda. Este tipo de estrategia es el seguido por la búsqueda tabú, la búsqueda local iterada, la búsqueda con vecindario variable y el enfriamiento simulado. Estas metaheurísticas trabajan sobre una o varias estructuras de vecindario impuestas por el espacio de búsqueda. Otro tipo de estrategia es el seguido por las colonias de hormigas o los algoritmos evolutivos. Éstos incorporan un componente de aprendizaje en el sentido de que, de forma implícita o explícita, intentan aprender la correlación entre las variables del problema para identificar las regiones del espacio de búsqueda con soluciones de alta calidad (en la Figura 2.1, metaheurísticas basadas en población). Estos métodos realizan, en este sentido, un muestreo sesgado del espacio de búsqueda.

Más formalmente, una metaheurística se define como una tupla de elementos que, dependiendo de cómo se definan, dan lugar a una técnica concreta u otra. Esta definición formal ha sido desarrollada en (Luque, 2006) y posteriormente extendida en
citepchicano06metaheuristica.

Definición 2 (Metaheurística). *Una metaheurística $\mathcal{M}$ es una tupla formada por los siguientes ocho componentes:*

$$\mathcal{M} = \langle \mathcal{T}, \Xi, \mu, \lambda, \Phi, \sigma, \mathcal{U}, \tau \rangle \ , \tag{2.4}$$

donde:

- *$\mathcal{T}$ es el conjunto de elementos que manipula la metaheurística. Este conjunto contiene al espacio de búsqueda y en la mayoría de los casos coincide con él.*
- *$\Xi = \{(\xi_1, D_1), (\xi_2, D_2), \ldots, (\xi_v, D_v)\}$ es un conjunto de v pares. Cada par está formado por una variable de estado de la metaheurística y el dominio de dicha variable.*
- *μ es el número de soluciones con las que trabaja $\mathcal{M}$ en un paso.*
- *λ es el número de nuevas soluciones generadas en cada iteración de $\mathcal{M}$.*
- *$\Phi : \mathcal{T}^\mu \times \prod_{i=1}^{v} D_i \times \mathcal{T}^\lambda \to [0,1]$ representa el operador que genera nuevas soluciones a partir de las existentes. Esta función debe cumplir para todo $x \in \mathcal{T}^\mu$ y para todo $t \in \prod_{i=1}^{v} D_i$,*

$$\sum_{y \in \mathcal{T}^\lambda} \Phi(x,t,y) = 1 \ . \tag{2.5}$$

- *$\sigma : \mathcal{T}^\mu \times \mathcal{T}^\lambda \times \prod_{i=1}^{v} D_i \times \mathcal{T}^\mu \to [0,1]$ es una función que permite seleccionar las soluciones que serán manipuladas en la siguiente iteración de $\mathcal{M}$. Esta función debe cumplir para todo $x \in \mathcal{T}^\mu$, $z \in \mathcal{T}^\lambda$ y $t \in \prod_{i=1}^{v} D_i$,*

$$\sum_{y \in \mathcal{T}^\mu} \sigma(x,z,t,y) = 1 \ , \tag{2.6}$$

$$\begin{aligned} &\forall y \in \mathcal{T}^\mu, \sigma(x,z,t,y) = 0 \vee \\ &\quad \vee \sigma(x,z,t,y) > 0 \wedge \\ &(\forall i \in \{1,\ldots,\mu\} \bullet (\exists j \in \{1,\ldots,\mu\}, y_i = x_j) \vee (\exists j \in \{1,\ldots,\lambda\}, y_i = z_j)) \ . \end{aligned} \tag{2.7}$$

- $\mathcal{U} : \mathcal{T}^{\mu} \times \mathcal{T}^{\lambda} \times \prod_{i=1}^{v} D_i \times \prod_{i=1}^{v} D_i \rightarrow [0,1]$ *representa el procedimiento de actualización de las variables de estado de la metaheurística. Esta función debe cumplir para todo* $x \in \mathcal{T}^{\mu}$, $z \in \mathcal{T}^{\lambda}$ *y* $t \in \prod_{i=1}^{v} D_i$,

$$\sum_{u \in \prod_{i=1}^{v} D_i} \mathcal{U}(x,z,t,u) = 1 \ . \tag{2.8}$$

- $\tau : \mathcal{T}^{\mu} \times \prod_{i=1}^{v} D_i \rightarrow \{falso, cierto\}$ *es una función que decide la terminación del algoritmo.*

La definición anterior recoge el comportamiento estocástico típico de las técnicas metaheurísticas. En concreto, las funciones Φ, σ y $\mathcal{U}$ deben interpretarse como probabilidades condicionadas. Por ejemplo, el valor de $\Phi(x,t,y)$ se interpreta como la probabilidad de que se genere el vector de hijos $y \in \mathcal{T}^{\lambda}$ dado que actualmente el conjunto de individuos con el que la metaheurística trabaja es $x \in \mathcal{T}^{\mu}$ y su estado interno viene definido por las variables de estado $t \in \prod_{i=1}^{v} D_i$. Puede observarse que las restricciones que se le imponen a las funciones Φ, σ y $\mathcal{U}$ permite considerarlas como funciones que devuelven estas probabilidades condicionadas.

Definición 3 (Estado de una metaheurística)**.** *Sea* $\mathcal{M} = \langle \mathcal{T}, \Xi, \mu, \lambda, \Phi, \sigma, \mathcal{U}, \tau \rangle$ *una metaheurística y* $\Theta = \{\theta_1, \theta_2, \ldots, \theta_{\mu}\}$ *el conjunto de variables que almacenarán las soluciones con las que trabaja la metaheurística. Utilizaremos la notación* $first(\Xi)$ *para referirnos al conjunto de las variables de estado de la metaheurística,* $\{\xi_1, \xi_2, \ldots, \xi_v\}$*. Un* estado s *de la metaheurística es un par de funciones* $s = (s_1, s_2)$ *con*

$$s_1 : \Theta \rightarrow \mathcal{T}, \tag{2.9}$$

$$s_2 : first(\Xi) \rightarrow \bigcup_{i=1}^{v} D_i \ , \tag{2.10}$$

donde s_2 *cumple*

$$s_2(\xi_i) \in D_i \ \ \forall \xi_i \in first(\Xi) \ . \tag{2.11}$$

Denotaremos con $\mathcal{S}_{\mathcal{M}}$ *el conjunto de todos los estados de una metaheurística* $\mathcal{M}$*.*

Por último, una vez definido el estado de la metaheurística, podemos definir su dinámica.

Definición 4 (Dinámica de una metaheurística)**.** *Sea* $\mathcal{M} = \langle \mathcal{T}, \Xi, \mu, \lambda, \Phi, \sigma, \mathcal{U}, \tau \rangle$ *una metaheurística y* $\Theta = \{\theta_1, \theta_2, \ldots, \theta_{\mu}\}$ *el conjunto de variables que almacenarán las soluciones con las que trabaja la metaheurística. Denotaremos con* $\overline{\Theta}$ *a la tupla* $(\theta_1, \theta_2, \ldots, \theta_{\mu})$ *y con* $\overline{\Xi}$ *a la tupla* $(\xi_1, \xi_2, \ldots, \xi_v)$*. Extenderemos la definición de estado para que pueda aplicarse a tuplas de elementos, esto es, definimos* $\overline{s} = (\overline{s}_1, \overline{s}_2)$ *donde*

$$\overline{s}_1 : \Theta^n \rightarrow \mathcal{T}^n \ , \tag{2.12}$$

$$\overline{s}_2 : first(\Xi)^n \rightarrow \left(\bigcup_{i=1}^{v} D_i \right)^n \ , \tag{2.13}$$

y además

$$\overline{s}_1(\theta_{i_1}, \theta_{i_2}, \ldots, \theta_{i_n}) = (s_1(\theta_{i_1}), s_1(\theta_{i_2}), \ldots, s_1(\theta_{i_n})) \ , \tag{2.14}$$

$$\overline{s}_2(\xi_{j_1}, \xi_{j_2}, \ldots, \xi_{j_n}) = (s_2(\xi_{j_1}), s_2(\xi_{j_2}), \ldots, s_2(\xi_{j_n})) \ , \tag{2.15}$$

para $n \geq 2$. *Diremos que* r *es un estado sucesor de* s *si existe* $t \in \mathcal{T}^{\lambda}$ *tal que* $\Phi(\overline{s}_1(\overline{\Theta}), \overline{s}_2(\overline{\Xi}), t) > 0$ *y además*

$$\sigma(\overline{s}_1(\overline{\Theta}), t, \overline{s}_2(\overline{\Xi}), \overline{r}_1(\overline{\Theta})) > 0 \ y \tag{2.16}$$

$$\mathcal{U}(\overline{s}_1(\overline{\Theta}), t, \overline{s}_2(\overline{\Xi}), \overline{r}_2(\overline{\Xi})) > 0 \ . \tag{2.17}$$

Denotaremos con $\mathcal{F}_{\mathcal{M}}$ *la relación binaria "ser sucesor de" definida en el conjunto de estados de una metaheurística* $\mathcal{M}$. *Es decir,* $\mathcal{F}_{\mathcal{M}} \subseteq \mathcal{S}_{\mathcal{M}} \times \mathcal{S}_{\mathcal{M}}$, *y* $\mathcal{F}_{\mathcal{M}}(s, r)$ *si* r *es un estado sucesor de* s.

Definición 5 (Ejecución de una metaheurística). *Una ejecución de una metaheurística* $\mathcal{M}$ *es una secuencia finita o infinita de estados,* $s_0, s_1, \ldots$ *en la que* $\mathcal{F}_{\mathcal{M}}(s_i, s_{i+1})$ *para todo* $i \geq 0$ *y además:*

- *si la secuencia es infinita se cumple* $\tau(s_i(\overline{\Theta}), s_i(\overline{\Xi})) = falso$ *para todo* $i \geq 0$ *y*
- *si la secuencia es finita se cumple* $\tau(s_k(\overline{\Theta}), s_k(\overline{\Xi})) = cierto$ *para el último estado* s_k *y, además,* $\tau(s_i(\overline{\Theta}), s_i(\overline{\Xi})) = falso$ *para todo* $i \geq 0$ *tal que* $i < k$.

En las próximas secciones tendremos la oportunidad de comprobar cómo esta formulación general se puede adaptar a las técnicas concretas (obviando aquellos parámetros no fijados por la metaheurística o que dependen de otros aspectos como el problema o la implementación concreta).

2.2. Clasificación de las metaheurísticas

Hay diferentes formas de clasificar y describir las técnicas metaheurísticas (Blum and Roli, 2003). Dependiendo de las características que se seleccionen se pueden obtener diferentes taxonomías: basadas en la naturaleza y no basadas en la naturaleza, con memoria o sin ella, con una o varias estructuras de vecindario, etc. Una de las clasificaciones más populares las divide en metaheurísticas *basadas en trayectoria* y *basadas en población*. Las primeras manipulan en cada paso un único elemento del espacio de búsqueda, mientras que las segundas trabajan sobre un conjunto de ellos (población). Esta taxonomía se muestra de forma gráfica en la Figura 2.2, que además incluye las técnicas más representativas. Estas metaheurísticas se describen en las dos secciones siguientes.

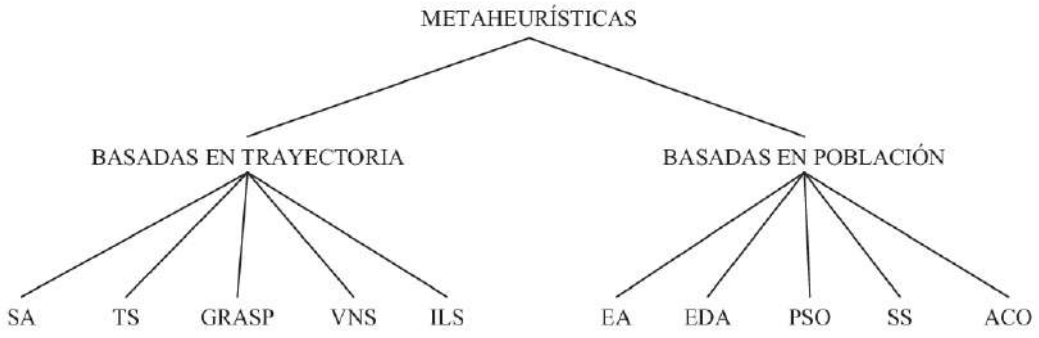


Figura 2.2: Clasificación de las metaheurísticas.

2.2.1. Metaheurísticas basadas en trayectoria

En esta sección repasaremos brevemente algunas metaheurísticas basadas en trayectoria. La principal característica de estos métodos es que parten de una solución y, mediante la exploración

del vecindario, van actualizando la solución actual, formando una trayectoria. Según la notación de la Definición 2, esto se formaliza con $\mu = 1$. La mayoría de estos algoritmos surgen como extensiones de los métodos simples de búsqueda local a los que se les añade algún mecanismo para escapar de los mínimos locales. Esto implica la necesidad de una condición de parada más elaborada que la de encontrar un mínimo local. Normalmente se termina la búsqueda cuando se alcanza un número máximo predefinido de iteraciones, se encuentra una solución con una calidad aceptable, o se detecta un estancamiento del proceso.

2.2.1.0.1. Enfriamiento simulado (SA) El *enfriamiento simulado* o *Simulated Annealing* (SA) es una de las técnicas más antiguas entre las metaheurísticas y posiblemente es el primer algoritmo con una estrategia explícita para escapar de los mínimos locales. Los orígenes del algoritmo se encuentran en un mecanismo estadístico, denominado *metropolis* (Metropolis et al, 1953). La idea del SA es simular el proceso de enfriamiento del metal y del cristal. El SA fue inicialmente presentado en (Kirkpatrick et al, 1983). Para evitar quedar atrapado en un mínimo local, el algoritmo permite elegir con cierta probabilidad una solución cuyo valor de *fitness* sea peor que el de la solución actual. En cada iteración se elige, a partir de la solución actual s, una solución s' del vecindario $N(s)$. Si s' es mejor que s (es decir, tiene un mejor valor en la función de *fitness*), se sustituye s por s' como solución actual. Si la solución s' es peor, entonces es aceptada con una determinada probabilidad que depende de la temperatura actual T y de la diferencia de *fitness* entre ambas soluciones, $f(s') - f(s)$ (caso de minimización).

2.2.1.0.2. Búsqueda tabú (TS) La *búsqueda tabú* o *Tabu Search* (TS) es una de las metaheurísticas que se han aplicado con más éxito a la hora de resolver problemas de optimización combinatoria. Los fundamentos de este método fueron introducidos en (Glover, 1986), y están basados en las ideas formuladas en (Glover, 1977). Un buen resumen de esta técnica y sus componentes se puede encontrar en (Glover, 1977).

La idea básica de la búsqueda tabú es el uso explícito de un historial de la búsqueda (una memoria a corto plazo), tanto para escapar de los mínimos locales como para implementar su estrategia de exploración y evitar buscar varias veces en la misma región. Esta memoria a corto plazo se implementa con una lista tabú, donde se mantienen las soluciones visitadas más recientemente para excluirlas de los próximos movimientos. En cada iteración se elige la mejor solución entre las permitidas y la solución es añadida a la lista tabú.

Desde el punto de vista de la implementación, mantener una lista de soluciones completas no suele ser práctico debido a su ineficiencia. Por lo tanto, en general, se suelen almacenar los movimientos que ha llevado al algoritmo a generar esa solución o los componentes principales que definen la solución. En cualquier caso, los elementos de esta lista permiten filtrar el vecindario, generando un conjunto reducido de soluciones elegibles denominado $N_a(s)$. El almacenamiento de los movimientos en vez de las soluciones completas es bastante más eficiente, pero introduce una perdida de información. Para evitar este problema, se define un criterio de aspiración que permite incluir una solución en $N_a(s)$ incluso si está prohibida debido a la lista tabú. El criterio de aspiración más ampliamente usado es permitir soluciones cuyo *fitness* sea mejor que el de la mejor solución encontrada hasta el momento.

2.2.1.0.3. GRASP El *procedimiento de búsqueda miope aleatorizado y adaptativo* o *Greedy Randomized Adaptive Search Procedure* (GRASP) (Feo and Resende, 1999) es una metaheurística simple que combina heurísticos constructivos con búsqueda local. GRASP es un procedimiento iterativo, compuesto de dos fases: primero la construcción de una solución y después un proceso de mejora. La solución mejorada es el resultado del proceso de búsqueda. El mecanismo de construcción de soluciones es un heurístico constructivo aleatorio. Va añadiendo paso a paso diferentes

componentes c a la solución parcial s^p, que inicialmente está vacía. Los componentes que se añaden en cada paso son elegidos aleatoriamente de una lista restringida de candidatos (RCL). Esta lista es un subconjunto de $N(s^p)$, el conjunto de componentes permitidos para la solución parcial s^p. Para generar esta lista, los componentes de la solución en $N(s^p)$ se ordenan de acuerdo a alguna función dependiente del problema (η).

La lista RCL está compuesta por los α mejores componentes de ese conjunto. En el caso extremo de $\alpha = 1$, siempre se añade el mejor componente encontrado de manera determinista, con lo que el método de construcción es equivalente a un algoritmo voraz. En el otro extremo, con $\alpha = |N(s^p)|$ el componente a añadir se elige de forma totalmente aleatoria de entre todos los disponibles. Por lo tanto, α es un parámetro clave que influye en cómo se va a muestrear el espacio de búsqueda. La segunda fase del algoritmo consiste en aplicar un método de búsqueda local para mejorar la solución generada. Este mecanismo de mejora puede ser una técnica de mejora simple o algoritmos más complejos como SA o TS.

2.2.1.0.4. Búsqueda con vecindario variable (VNS) La *búsqueda con vecindario variable* o *Variable Neighborhood Search* (VNS) es una metaheurística propuesta en (Mladenovic and Hansen, 1997) que aplica explícitamente una estrategia para cambiar entre diferentes vecindarios durante la búsqueda. Este algoritmo es muy general y con muchos grados de libertad a la hora de diseñar variaciones e instanciaciones particulares.

El primer paso a realizar es definir un conjunto de vecindarios. Esta elección puede hacerse de muchas formas: desde ser elegidos aleatoriamente hasta utilizar complejas ecuaciones deducidas del problema. Cada iteración consiste en tres fases: la elección del candidato, una fase de mejora y, finalmente, el movimiento. En la primera fase, se elige aleatoriamente un vecino s' de s usando el k-ésimo vecindario. Esta solución s' es utilizada como punto de partida de la búsqueda local de la segunda fase. Cuando termina el proceso de mejora, se compara la nueva solución s'' con la original s. Si es mejor, s'' se convierte en la solución actual y se inicializa el contador de vecindarios ($k \leftarrow 1$); si no es mejor, se repite el proceso pero utilizando el siguiente vecindario ($k \leftarrow k+1$). La búsqueda local es el paso de intensificación del método y el cambio de vecindario puede considerarse como el paso de diversificación.

2.2.1.0.5. Búsqueda local iterada (ILS) La *búsqueda local iterada* o *Iterated Local Search* (ILS) (Lourenço et al, 2002; Stützle, 1999) es una metaheurística basada en un concepto simple pero muy efectivo. En cada iteración, la solución actual es perturbada y, a esta nueva solución, se le aplica un método de búsqueda local para mejorarla. El mínimo local obtenido por el método de mejora puede ser aceptado como nueva solución actual si pasa un test de aceptación. La importancia del proceso de perturbación es obvia: si es demasiado pequeña puede que el algoritmo no sea capaz de escapar del mínimo local; por otro lado, si es demasiado grande, la perturbación puede hacer que el algoritmo sea como un método de búsqueda local con un reinicio aleatorio. Por lo tanto, el método de perturbación debe generar una nueva solución que sirva como inicio a la búsqueda local, pero que no debe estar muy lejos del actual para que no sea una solución aleatoria. El criterio de aceptación actúa como contra-balance, ya que filtra la aceptación de nuevas soluciones dependiendo de la historia de búsqueda y de las características del nuevo mínimo local.

2.2.2. Metaheurísticas basadas en población

Los métodos basados en población se caracterizan por trabajar con un conjunto de soluciones, usualmente denominado población, en cada iteración (es decir, generalmente $\mu > 1$ y/o $\lambda > 1$), a diferencia de los métodos basados en trayectoria, que únicamente manipulan una solución del espacio de búsqueda por iteración.

2.2.2.0.6. Algoritmos evolutivos (EA) Los *algoritmos evolutivos* o *Evolutionary Algorithms* (EAs) están inspirados en la teoría de la evolución natural. Esta familia de técnicas sigue un proceso iterativo y estocástico que opera sobre una población de soluciones, denominadas en este contexto *individuos*. Inicialmente, la población es generada típicamente de forma aleatoria (quizás con ayuda de un heurístico de construcción).

El esquema general de un algoritmo evolutivo comprende tres fases principales: selección, reproducción y reemplaz. El proceso completo es repetido hasta que se cumpla un cierto criterio de terminación (normalmente después de un número dado de iteraciones). En la fase de selección se escogen generalmente los individuos más aptos de la población actual para ser posteriormente recombinados en la fase de reproducción. Los individuos resultantes de la recombinación se alteran mediante un operador de mutación. Finalmente, a partir de la población actual y/o los mejores individuos generados (de acuerdo a su valor de *fitness*) se forma la nueva población, dando paso a la siguiente generación del algoritmo.

2.2.2.0.7. Algoritmos de estimación de la distribución (EDA) Los *algoritmos de estimación de la distribución* o *Estimation of Distribution Algorithms* (EDAs) (Mühlenbein, 1998) muestran un comportamiento similar a los algoritmos evolutivos presentados en la sección anterior y, de hecho, muchos autores consideran los EDAs como otro tipo de EA. Los EDAs operan sobre una población de soluciones tentativas como los algoritmos evolutivos pero, a diferencia de estos últimos, que utilizan operadores de recombinación y mutación para mejorar las soluciones, los EDAs infieren la distribución de probabilidad del conjunto seleccionado y, a partir de ésta, generan nuevas soluciones que formarán parte de la población.

Los modelos gráficos probabilísticos son herramientas comúnmente usadas en el contexto de los EDAs para representar eficientemente la distribución de probabilidad. Algunos autores (Larrañaga et al, 1999; Pelikan et al, 1999; Soto et al, 1999) han propuesto las redes bayesianas para representar la distribución de probabilidad en dominios discretos, mientras que las redes gaussianas se emplean usualmente en los dominios continuos (Whittaker, 1990).

2.2.2.0.8. Búsqueda dispersa (SS) La *búsqueda dispersa* o *Scatter Search* (SS) (Glover, 1998) es una metaheurística cuyos principios fueron presentados en (Glover, 1977) y que actualmente está recibiendo una gran atención por parte de la comunidad científica (Laguna and Martí, 2003). El algoritmo se basa en mantener un conjunto relativamente pequeño de soluciones tentativas (llamado conjunto de referencia o *RefSet*) que se caracteriza por contener soluciones de calidad y diversas (distantes en el espacio de búsqueda). Para la definición completa de SS hay que concretar cinco componentes: creación de la población inicial, generación del conjunto de referencia, generación de subconjuntos de soluciones, método de combinación de soluciones y método de mejora.

2.2.2.0.9. Optimización basada en colonias de hormigas (ACO) Los *algoritmos de optimización basados en colonias de hormigas* o *Ant Colony Optimization* (ACO) (Dorigo, 1992; Dorigo and Stützle, 2003) están inspirados en el comportamiento de las hormigas reales cuando buscan comida. Este comportamiento es el siguiente: inicialmente, las hormigas exploran el área cercana a su nido de forma aleatoria. Tan pronto como una hormiga encuentra comida, la lleva al nido. Mientras que realiza este camino, la hormiga va depositando una sustancia química denominada feromona. Esta sustancia ayudará al resto de las hormigas a encontrar la comida. La comunicación indirecta entre las hormigas mediante el rastro de feromona las capacita para encontrar el camino más corto entre el nido y la comida. Este comportamiento es el que intenta simular este método para resolver problemas de optimización. La técnica se basa en dos pasos principales: construcción de una solución basada en el comportamiento de una hormiga y actualización de los rastros de

feromona artificiales. El algoritmo no fija ninguna planificación o sincronización *a priori* entre las fases, pudiendo ser incluso realizadas simultáneamente.

2.2.2.0.10. Optimización basada en cúmulos de partículas (PSO) Los *algoritmos de optimización basados en cúmulos de partículas* o *Particle Swarm Optimization* (PSO) (Kennedy, 1999) están inspirados en el comportamiento social del vuelo de las bandadas de aves o el movimiento de los bancos de peces. El algoritmo PSO mantiene un conjunto de soluciones, también llamadas *partículas*, que son inicializadas aleatoriamente en el espacio de búsqueda. Cada partícula posee una posición y velocidad que cambia conforme avanza la búsqueda. En el movimiento de una partícula influye su velocidad y las posiciones donde la propia partícula y las demás de su vecindario encontraron buenas soluciones. En el contexto de PSO, el *vecindario de una partícula* se define como un conjunto de partículas del cúmulo. No debe confundirse con el concepto de vecindario de una solución utilizado previamente en este capítulo. El vecindario de una partícula puede ser *global*, en el cual todas las partículas del cúmulo se consideran vecinas, o *local*, en el que sólo las partículas más cercanas se consideran vecinas.

2.2.3. Metaheurísticas para optimización multiobjetivo

La mayoría de los problemas de optimización del mundo real son de naturaleza multiobjetivo, lo que supone que hay que minimizar o maximizar varias funciones a la vez que están normalmente en conflicto entre sí (problemas multiobjetivo o MOPs, *Multiobjective Optimization Problems*). Debido a la falta de soluciones metodológicas adecuadas, los problemas multiobjetivo se han resuelto en el pasado como problemas monoobjetivo. Sin embargo, existen diferencias fundamentales en los principios de funcionamiento de los algoritmos para optimización mono y multiobjetivo. Así, las técnicas utilizadas para resolver MOPs no se restringen normalmente a encontrar una solución única, sino un conjunto de soluciones de compromiso entre los múltiples objetivos contrapuestos, ya que no suele existir una solución que optimice simultáneamente todos los objetivos. Se pueden distinguir, por tanto, dos etapas cuando se aborda este tipo de problemas: por un lado, la optimización de varias funciones objetivo involucradas y, por otro, el proceso de toma de decisiones sobre qué solución de compromiso es la más adecuada (Coello et al, 2007). Atendiendo a cómo manejan estas dos etapas, las técnicas para resolver MOPs se pueden clasificar en (Cohon and Marks, 1975):

- *A priori*: cuando las decisiones se toman antes de buscar soluciones.
- *Progresivas*: cuando se integran la búsqueda de soluciones y la toma de decisiones.
- *A posteriori*: cuando se busca antes de tomar decisiones.

Cada una de ellas tiene ciertas ventajas e inconvenientes que las hacen más adecuadas para determinados escenarios concretos (Coello et al, 2007; Deb, 2001). No obstante, en las dos primeras clases la búsqueda está muy influenciada por la decisión de un experto (*decision maker*) que determina la importancia de un objetivo sobre otro y que puede limitar arbitrariamente el espacio de búsqueda, impidiendo una resolución óptima del problema. En las técnicas *a posteriori*, por el contrario, se realiza una exploración lo más amplia posible para generar tantas soluciones de compromiso como sea posible. Es, entonces, cuando tiene lugar el proceso de toma de decisiones por parte del experto. Precisamente por la aproximación que siguen, estas técnicas *a posteriori* están siendo muy utilizadas dentro del campo de las metaheurísticas y, particularmente, en el campo de la computación evolutiva (Coello et al, 2007; Deb, 2001). Más concretamente, los algoritmos más avanzados aplican técnicas *a posteriori* basadas en el concepto de *Optimalidad de Pareto* (Pareto, 1896) y es el enfoque seguido en esta tesis. Así, hemos estructurado esta sección en tres apartados. El primero de ellos presenta formalmente los conceptos básicos relacionados con esta optimalidad

de Pareto. El siguiente apartado presenta las metas que debe perseguir todo algoritmo que utiliza estas técnicas cuando aborda un MOP. Finalmente, el tercer apartado discute algunos aspectos de diseño que se deben adoptar en los algoritmos que resuelven problemas siguiendo la aproximación anterior.

2.2.4. Conceptos básicos

En esta sección se presentan algunos conceptos básicos de la optimización multiobjetivo para familiarizar al lector con este campo. Comenzaremos dando unas nociones de lo que entendemos por problema de optimización multiobjetivo o MOP. Informalmente, un MOP se puede definir como el problema de encontrar un vector de variables de decisión que satisface un conjunto de restricciones y que optimiza un conjunto de funciones objetivo. Estas funciones forman una descripción matemática de criterios de rendimiento que están normalmente en conflicto entre sí. Por tanto, el término "optimización" se refiere a la búsqueda de una solución tal que contenga valores aceptables para todas las funciones objetivo (Osyczka, 1895).

Matemáticamente, la formulación de un MOP extiende la definición clásica de optimización monoobjetivo (Definición 1) para considerar la existencia de varias funciones objetivo. No hay, por tanto, una única solución al problema, sino un conjunto de soluciones. Este conjunto de soluciones se encuentra mediante la utilización de la Teoría de Optimalidad de Pareto (Ehrgott, 2005). Formalmente (Van Veldhuizen, 1999):

Definición 6 (MOP). *Encontrar un vector* $\vec{x}^* = [x_1^*, x_2^*, \ldots, x_n^*]$ *que satisfaga las* m *restricciones de desigualdad* $g_i(\vec{x}) \geq 0, i = 1, 2, \ldots, m$, *las* p *restricciones de igualdad* $h_i(\vec{x}) = 0, i = 1, 2, \ldots, p$, *y que minimice la función vector* $\vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})]^T$, *donde* $\vec{x} = [x_1, x_2, \ldots, x_n]^T$ *es el vector de decisión de variables.*

El conjunto de todos los valores que satisfacen las restricciones define la *región de soluciones factibles* Ω y cualquier punto en $\vec{x} \in \Omega$ es una *solución factible*.

Teniendo varias funciones objetivo, la noción de "óptimo" cambia, ya que el objetivo para cualquier MOP es encontrar buenos compromisos (*trade-offs*) entre estas funciones. La noción de "óptimo" más utilizada es la propuesta por Francis Ysidro Edgeworth (Edgeworth, 1881), generalizada posteriormente por Vilfredo Pareto (Pareto, 1896). Aunque algunos autores lo denominan óptimo de Edgeworth-Pareto, el término óptimo de Pareto es comúnmente aceptado. Su definición formal se da a continuación:

Definición 7 (Optimalidad de Pareto). *Un punto* $\vec{x}^* \in \Omega$ *es un óptimo de Pareto si para cada* $\vec{x} \in \Omega$ *y* $I = \{1, 2, \ldots, k\}$, *o bien* $\forall_{i \in I}(f_i(\vec{x}) = f_i(\vec{x}^*))$ *o bien hay al menos un* $i \in I \mid f_i(\vec{x}) > f_i(\vec{x}^*)$.

Esta definición dice que $\vec{x}^*$ es un óptimo de Pareto si no existe ningún vector factible $\vec{x}$ que mejore algún criterio sin causar simultáneamente un empeoramiento en al menos otro criterio (asumiendo minimización). El concepto de optimalidad de Pareto es integral tanto a la teoría como a la resolución de MOPs. Existen un algunas definiciones adicionales que son también básicas en optimización multiobjetivo (Van Veldhuizen, 1999):

Definición 8 (Dominancia de Pareto). *Un vector* $\vec{u} = (u_1, \ldots, u_k)$ *se dice que domina a otro vector* $\vec{v}=(v_1, \ldots, v_k)$ *(representado por* $\vec{u} \prec \vec{v}$*) si y sólo si* $\vec{u}$ *es parcialmente menor que* $\vec{v}$, *es decir,* $\forall i \in \{1, \ldots, k\},\ u_i \leq v_i \ \wedge\ \exists\, i \in \{1, \ldots, k\} :\ u_i < v_i$.

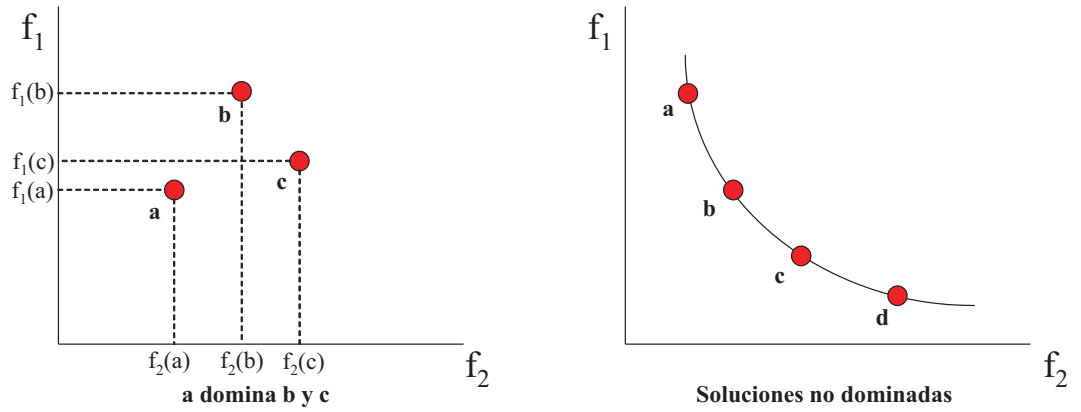


Figura 2.3: Ejemplo del concepto de dominancia de Pareto.

Ilustremos gráficamente este concepto. La Figura 2.3 incluye dos conjuntos de soluciones para un problema multiobjetivo con dos funciones f_1 y f_2, que han de ser minimizadas. Siendo ambos objetivos igual de importantes, no resulta trivial distinguir qué solución es mejor que otra. Podemos utilizar la definición anterior para esto. Así, si nos fijamos en la parte izquierda de la figura, podemos decir que a es mejor que b puesto que $f_1(a) < f_1(b)$ y $f_2(a) < f_2(b)$, es decir, es mejor en ambos objetivos y, por tanto, se dice que a domina a b ($a \prec b$). Lo mismo ocurre si comparamos a y c, en ambos objetivos $f_1(a) < f_1(c)$ y $f_2(a) < f_2(c)$, por lo que $a \prec c$. Comparemos ahora las soluciones b y c entre ellas. Se puede observar que c es mejor que b en f_1 ($f_1(c) < f_1(b)$, pero b es mejor que c para f_2 ($f_2(b) < f_2(b)$). Según la Definición 8, no podemos decir que b domina a c ni que c domina a b, es decir, no podemos concluir que una solución es mejor que la otra. En este caso se dice que ambas soluciones son no dominadas. En la parte derecha de la Figura 2.3 se muestran 4 soluciones de este tipo, en las que ninguna es mejor que las demás.

Resolver un MOP consiste, por tanto, en encontrar el conjunto de soluciones que dominan a cualquier otra solución del espacio de soluciones, lo que significa que son las mejores para el problema y, por tanto, conforman su solución óptima. Formalmente:

Definición 9 (Conjunto Óptimo de Pareto). *Para un MOP dado $\vec{f}(\vec{x})$, el conjunto óptimo de Pareto se define como* $\mathcal{P}^* = \{\vec{x} \in \Omega | \neg\exists \vec{x'} \in \Omega, \vec{f}(\vec{x'}) \preccurlyeq \vec{f}(\vec{x})\}$.

No hay que olvidar que las soluciones Pareto-óptimas que están en $\mathcal{P}^*$ están en el espacio de las variables (genotípico), pero cuyos componentes del vector en el espacio de objetivos (fenotípico) no se pueden mejorar simultáneamente. Estas soluciones también suelen llamarse *no inferiores*, *admisibles* o *eficientes*. El frente de Pareto se define, entonces, como:

Definición 10 (Frente de Pareto). *Para un MOP dado $\vec{f}(\vec{x})$ y su conjunto óptimo de Pareto $\mathcal{P}^*$, el frente de Pareto se define como* $\mathcal{PF}^* = \{\vec{f}(\vec{x}), \vec{x} \in \mathcal{P}^*\}$.

Es decir, el frente de Pareto está compuesto por los valores en el espacio de objetivos del conjunto óptimo de Pareto. En general, no es fácil encontrar una expresión analítica de la línea o superficie que contiene estos puntos. De hecho,en la mayoría de los casos es imposible. Como ejemplo, las Figuras 2.4 y 2.5 muestran la formulación y su correspondiente frente de Pareto de los problemas Binh2 y DTLZ4 (Coello et al, 2007). En el primer caso se trata de un problema biobjetivo, f_1 y f_2, con dos variables de decisión x_1 y x_2, que tiene definidas, además, dos restricciones $g1$ y $g2$. El problema DTLZ4, por su parti, tiene tres objetivos y ninguna restricción (la función $g()$ aquí es únicamente una notación usada para su formulación).

$$
\begin{array}{lcll}
\text{Min } F & = & (f_1(\vec{x}), f_2(\vec{x})) & \\
f_1(\vec{x}) & = & 4x_1^2 + 4x_2^2 & \\
f_2(\vec{x}) & = & (x_1 - 5)^2 + (x_2 - 5)^2 & \\
\text{Subject to:} & & & \\
g1(\vec{x}) & = & (x_1 - 5)^2 + x_2^2 - 25 & \leq 0 \\
g2(\vec{x}) & = & -(x_1 - 8)^2 - (x_2 + 3)^2 + 7{,}7 & \leq 0 \\
0 & \leq & x_1 & \leq 5 \\
0 & \leq & x_2 & \leq 3
\end{array}
$$

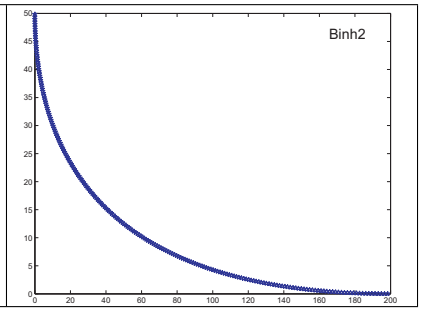


Figura 2.4: Formulación y frente de Pareto del problema Bihn2.

$$
\begin{array}{lcl}
f_1(x) & = & (1 + g(x_M)) \cos(x_1^{\alpha} \frac{\pi}{2}) \cos(x_2^{\alpha} \frac{\pi}{2}) \\
f_2(x) & = & (1 + g(x_M)) \cos(x_1^{\alpha} \frac{\pi}{2}) \sin(x_2^{\alpha} \frac{\pi}{2}) \\
f_3(x) & = & (1 + g(x_M)) \sin(x_1^{\alpha} \frac{\pi}{2}) \\
 & & 0 \leq x_i \leq 1, \quad i = 1, 2, \ldots, n \\
 & & \\
g(x_M) & = & \sum_{x_i \in x_M} (x_i - 0{,}5)^2 \\
 & & \\
n & = & 12 \\
x_M & = & x_3, ..., x_{12} \\
\alpha & = & 100
\end{array}
$$

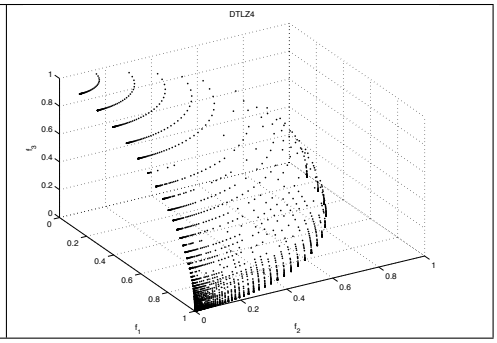


Figura 2.5: Formulación y frente de Pareto del problema DTLZ4.

2.3. Objetivos en la resolución de MOPs

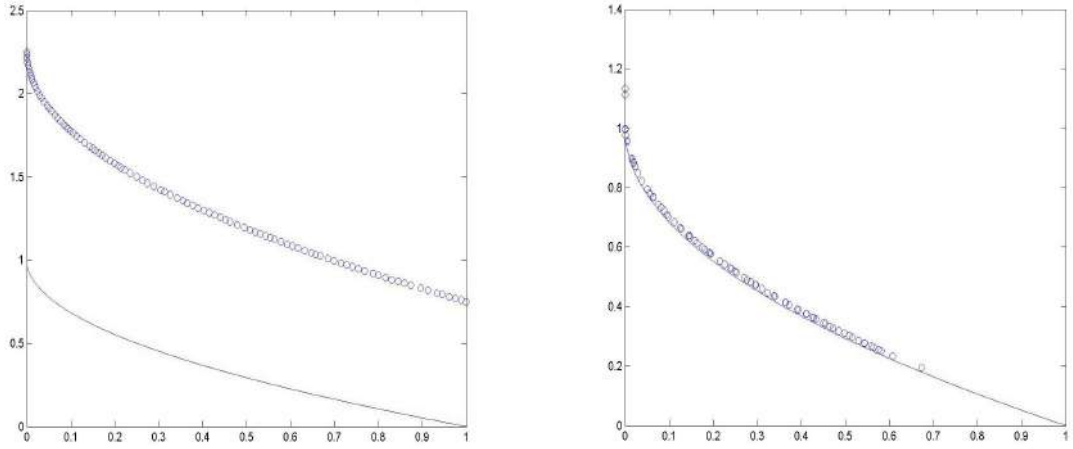
Cuando se aborda la resolución de un problema de optimización multiobjetivo, la principal meta de todo algoritmo de optimización que utiliza los conceptos y técnicas descritas en la sección anterior es encontrar su frente de Pareto (o, lo que es lo mismo, su conjunto óptimo de Pareto). No obstante, la presencia de múltiples soluciones Pareto-óptimas hace que sea difícil elegir una solución sobre otra sin información adicional sobre el problema, ya que todas estas soluciones son igualmente importantes. Dado un MOP, por tanto, se busca idealmente un número de soluciones no dominadas que persiguen dos metas:

1. Encontrar un conjunto de soluciones lo más cercano posible al frente de Pareto óptimo.

2. Encontrar un conjunto de soluciones tan uniformemente diverso como sea posible.

Mientras que la primera meta, converger hacia la solución óptima, es obligatoria en toda tarea de optimización mono o multiobjetivo, la segunda es completamente específica para optimización multiobjetivo. Además de converger hacia el frente óptimo, las soluciones deben estar uniformemente repartidas a lo largo de todo el frente. Sólo con un conjunto diverso de soluciones se puede asegurar, por una parte, un buen conjunto de soluciones de compromiso entre los diferentes objetivos para la posterior toma de decisiones por parte del experto y, por otra, que se ha realizado una buena exploración del espacio de búsqueda. La Figura 2.6 muestra dos ejemplos de frentes que fallan, cada uno, en una de las metas anteriores. En la parte (a) podemos ver una aproximación al

frente en el que las soluciones no dominadas se distribuyen perfectamente. No obstante, se trata de un MOP diseñado de forma que contiene múltiples frentes engañosos y, en realidad, las soluciones obtenidas no son Pareto-óptimas, aunque su diversidad es excelente. Por el contrario, en la parte (b) de la misma figura, tenemos un conjunto de soluciones que han convergido hacia el frente de Pareto óptimo pero, sin embargo, deja regiones de éste sin cubrir. Aunque ninguno de los dos casos es deseable, la primera situación es claramente peor: ninguna de las soluciones obtenidas es Pareto-óptima.



(a) (b)

Figura 2.6: Ejemplos de mala convergencia (a) y diversidad (b) en frentes de Pareto.

2.3.1. Aspectos de diseño

Adoptar técnicas basadas en optimalidad de Pareto dentro de algoritmos metaheurísticos supone, por un lado, trabajar con soluciones no dominadas que hacen necesario incorporar mecanismos específicos para manejarlas y, por otro, encontrar no una solución única, sino un conjunto de soluciones Pareto-óptimas que, además, ha de tener la diversidad suficiente para cubrir el todo el frente. Si bien existen muchos aspectos a considerar dependiendo de cada algoritmo concreto, los siguientes se pueden considerar comunes a todos ellos: función de aptitud (*función de fitness*) de las soluciones, mantenimiento de la diversidad, y manejo de restricciones. Cada uno de ellos se discute a continuación.

2.3.1.0.11. Función de aptitud En el ciclo de funcionamiento de toda metaheurística siempre existe alguna fase en la que hay que ordenar las soluciones con las que trabaja según su función de fitness para seleccionar alguna de ellas. Hablamos, por ejemplo, de los operadores de selección y reemplazo en algoritmos evolutivos o el método de actualización del conjunto de referencia en búsqueda dispersa. En el caso de optimización monoobjetivo, el fitness de una solución es un valor único y la ordenación de soluciones es trivial de acuerdo con este valor. No obstante, en nuestra aproximación para resolver MOPs, el fitness es un vector de valores (un valor por cada objetivo) por lo que la ordenación no es tan directa.

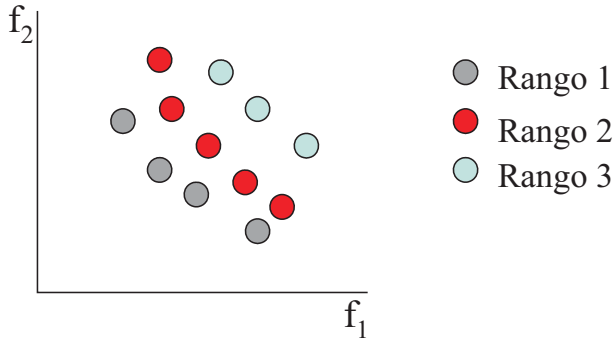


Figura 2.7: Ejemplo de ordenación (*ranking*) de soluciones en un MOP con dos objetivos.

La relación de dominancia (Ecuación 8) es la clave en este tipo de técnicas basadas en optimalidad de Pareto, ya que nos va a permitir establecer una ordenación de las soluciones. De hecho, esta relación es una relación de orden parcial estricto, puesto que no es reflexiva, ni simétrica, ni antisimétrica, pero sí transitiva. Así, se han propuesto diferentes métodos en la literatura (Coello et al, 2007; Deb, 2001) que, básicamente, transforman el vector de fitness en un valor único utilizando esta relación. Esta estrategia fue originalmente propuesta por Goldberg en (Goldberg, 1989) para guiar la población de un GA hacia el frente de Pareto de un MOP. La idea básica consiste en encontrar las soluciones de la población que no están dominadas por ninguna otra. A estas soluciones se le asigna el mayor orden (las mejores en la ordenación establecida por la relación de dominancia). A continuación, se consideran las soluciones no dominadas que quedan si se eliminan todas las anteriores, a las que se asigna el siguiente rango. El proceso continúa hasta que se le asigna un rango a todas las soluciones. La Figura 2.7 muestra un ejemplo del funcionamiento de este método de ordenación (f_1 y f_2 son funciones que han de minimizarse). Esta ordenación basada en dominancia es la más básica. Otra más avanzadas, como la fuerza o *strength* de SPEA2 (Zitzler et al, 2001) tienen en cuenta, además, el número de soluciones a los que domina cada solución.

2.3.1.0.12. Diversidad Si bien la función de fitness basada en dominancia ya dirige la búsqueda hacia el frente de Pareto dando una mayor aptitud a las soluciones no dominadas, esta aproximación por sí sola no es suficiente cuando se aborda un MOP. Si recordamos la Sección 2.3, además de converger al frente óptimo, las soluciones han de distribuirse lo mejor posible sobre este frente para poder ofrecer al experto el abanico más amplio de soluciones al problema multiobjetivo.

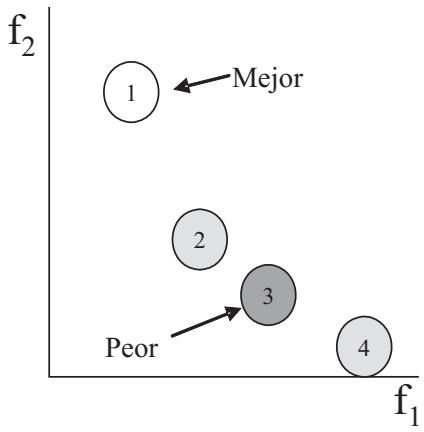


Figura 2.8: Ejemplo de estimador de densidad para soluciones no dominadas en un MOP con dos objetivos.

Aunque existen diferentes aproximaciones en la literatura (Coello et al, 2007), las más utilizadas en los algoritmos del estado del arte están basadas en complementar la función de fitness basada en dominancia (sección anterior) con un estimador que mide la densidad, en el espacio de objetivos, de soluciones alrededor de una solución dada. Así, dada dos soluciones con el mismo fitness (*ranking*, *strength*), el estimador de densidad discrimina entre las mejores y peores soluciones atendiendo a la diversidad de las mismas. Consideremos, por ejemplo, el conjunto de soluciones no dominadas de la Figura 2.8. Según la densidad de las mismas, la solución 1 se puede considerar como la mejor puesto es la que está en la zona menos "poblada". La solución 3, por el contrario, sería la peor al estar en una zona del frente donde ya existen soluciones cercanas. Algunos de los estimadores de densidad propuestos por los algoritmos multiobjetivo más conocidos son: *niching* en MOGA (Fonseca and Fleming, 1993) y NSGA (Srinivas and Deb, 1994), el grid adaptativo de PAES (Knowles and Corne, 1999a), *crowding* en NSGA-II (Deb et al, 2002a) y la distancia al k-ésimo vecino de SPEA2 (Zitzler et al, 2001).

2.3.1.0.13. Manejo de restricciones La definición de problema multiobjetivo (Ecuación 6) incluida en la Sección 2.2.4 incluye explícitamente restricciones ya que, principalmente, es la situación típica cuando se consideran problemas de corte real, como los considerados en esta tesis. Las restricciones se pueden considerar como duras o débiles. Una restricción es dura cuando ha de satisfacerse para que una solución dada sea aceptable. Por el contrario, una restricción débil es aquella que se puede relajar de alguna forma para aceptar una solución.

La aproximación más utilizada en las metaheurísticas multiobjetivo del estado del arte para tratar con restricciones están basadas en un esquema en el que las soluciones factibles son superiores a las no factibles (Deb, 2000, 2001). Así, dadas dos soluciones que se han de comparar, se pueden dar tres casos:

1. Si ambas soluciones son factibles, se utiliza la función de fitness basada en dominancia de Pareto explicada en la Sección 2.3.1.0.11. En el caso de que ambas sean no dominadas (mismo fitness), se utiliza un estimador de densidad (Sección 2.3.1.0.12) para discriminar entre ellas.

2. Si una solución es factible y la otra no, la factible se considera como mejor.

3. Si ambas soluciones son no factibles, entonces se selecciona la que menos viola las restricciones.

Queda por determinar cómo se cuantifica la cantidad de violación de restricciones de una solución dada. Para esto, la estrategia más utilizada consiste en transformar todas las restricciones para que sean de tipo *mayor-o-igual-que* cero: $g_i(\vec{x}) \geq 0$, según la definición de MOP (Ecuación 6) (Deb, 2001). Se puede considerar como un tipo de normalización, de forma que el propio valor $g_i(\vec{x})$ se usa para medir cuánto se viola la restricción. El mayor inconveniente para esta estrategia viene dado por la restricciones de igualdad $h_i(\vec{x}) = 0$. Si se trata de una restricción débil, se puede relajar directamente a $h_i(\vec{x}) \geq 0$. Sin embargo, si $h_i(\vec{x}) = 0$ es una restricción dura, la transformación no es directa (especialmente cuando es una restricción no lineal). Según un resultado obtenido en (Deb, 1995), es posible convertir estas restricciones duras de igualdad en restricciones débiles con pérdidas de precisión, lo que permite considerar ya todas las restricciones del mismo tipo. Existen otras muchas estrategias para tratar con restricciones en optimización multiobjetivo (Coello et al, 2007; Deb, 2001) pero sólo hemos detallado la que se utilizará en esta tesis.

2.4. Metaheurísticas paralelas

Aunque el uso de metaheurísticas permite reducir significativamente la complejidad temporal del proceso de búsqueda, este tiempo puede seguir siendo muy elevado en algunos problemas de interés real. Con la proliferación de plataformas paralelas de cómputo eficientes, la implementación paralela de estas metaheurísticas surge de forma natural como una alternativa para acelerar la obtención de soluciones precisas a estos problemas. La literatura es muy extensa en cuanto a la paralelización de metaheurísticas se refiere (Alba, 2005; Crainic and Toulouse, 2003; Cung et al, 2003) ya que se trata de una aproximación que puede ayudar no sólo a reducir el tiempo de cómputo, sino a producir también una mejora en la calidad de las soluciones encontradas. Esta mejora está basada en un nuevo modelo de búsqueda que alcanza un mejor balance entre intensificación y diversificación. De hecho, muchos investigadores no utilizan plataformas paralelas de cómputo para ejecutar estos modelos paralelos y, aún así, siguen obteniendo mejores resultados que con los algoritmos secuenciales tradicionales. Tanto para las metaheurísticas basadas en trayectoria como para las basadas en población se han propuesto modelos paralelos acorde a sus características. Las siguientes secciones 2.4.1 y 2.4.2 presentan, respectivamente, generalidades relacionadas con la paralelización de cada tipo de metaheurística.

En nuestro contexto, donde vamos a considerar problemas del mundo real, la utilización no sólo de modelos paralelos, sino también de plataformas paralelas es casi obligatorio, ya que una simple evaluación del problema puede llevar minutos o incluso horas si se consideran problemas en los que intervienen, por ejemplo, complejas simulaciones. En estos últimos casos, ni siquiera las arquitecturas paralelas más usuales como los clusters de máquinas o los multiprocesadores de memoria compartida tienen la capacidad suficiente para abordar, en un tiempo razonable, estos problemas de optimización que implican tareas tan costosas computacionalmente. La aparición de los sistemas de computación grid (Berman et al, 2003; Foster and Kesselman, 1999) puede permitir solventar estos inconvenientes en cierta medida, ya que son capaces de agrupar, como si de un elemento único se tratase, la potencia computacional de una gran cantidad de recursos geográficamente distribuidos. No obstante, todos los modelos paralelos de metaheurísticas no se adecúan a este tipo de plataformas y explotan toda su capacidad de cómputo.

2.4.1. Modelos paralelos para métodos basados en trayectoria

Los modelos paralelos de metaheurísticas basadas en trayectoria encontrados en la literatura se pueden clasificar, generalmente, dentro de tres posibles esquemas: ejecución en paralelo de varios métodos (*modelo de múltiples ejecuciones*), exploración en paralelo del vecindario (*modelo*

de movimientos paralelos), y cálculo en paralelo de la función de *fitness* (*modelo de aceleración del movimiento*). A continuación detallamos cada uno de ellos.

- **Modelo de múltiples ejecuciones:** este modelo consiste en ejecutar en paralelo varios subalgoritmos ya sean homogéneos o heterogéneos. En general, cada subalgoritmo comienza con una solución inicial diferente. Se pueden distinguir diferentes casos dependiendo de si los subalgoritmos colaboran entre sí o no. El caso en el que las ejecuciones son totalmente independientes se usa ampliamente porque es simple de utilizar y muy natural. En este caso, la semántica del modelo es la misma que la de la ejecución secuencial, ya que no existe cooperación. El único beneficio al utilizar este modelo respecto a realizar las ejecuciones en una única máquina es la reducción del tiempo de ejecución total.

  Por otro lado, en el caso cooperativo (véase el ejemplo de la derecha de la Figura 2.9), los diferentes subalgoritmos intercambian información durante la ejecución. En este caso el comportamiento global del algoritmo paralelo es diferente al secuencial y su rendimiento se ve afectado por cómo esté configurado este intercambio. El usuario debe fijar ciertos parámetros para completar el modelo: qué información se intercambian, cada cuánto se pasan la información y cómo se realiza este intercambio. La información intercambiada suele ser la mejor solución encontrada, los movimientos realizados o algún tipo de información sobre la trayectoria realizada. En cualquier caso, esta información no debe ser abundante para que el coste de la comunicación no sea excesivo e influya negativamente en la eficiencia. También se debe fijar cada cuántos pasos del algoritmo se intercambia la información. Para elegir este valor hay que tener en cuenta que el intercambio no sea muy frecuente, para que el coste de la comunicación no sea perjudicial, ni muy poco frecuente, para que el intercambio tenga algún efecto en el comportamiento global. Por último, se debe indicar si las comunicaciones se realizarán de forma asíncrona o síncrona. En el caso síncrono, los subalgoritmos, cuando llegan a la fase de comunicación, se detienen hasta que todos ellos llegan a este paso y sólo entonces se realiza la comunicación. En el caso asíncrono, que es el más usado, cada subalgoritmo realiza el intercambio sin esperar al resto cuando llega al paso de comunicación.

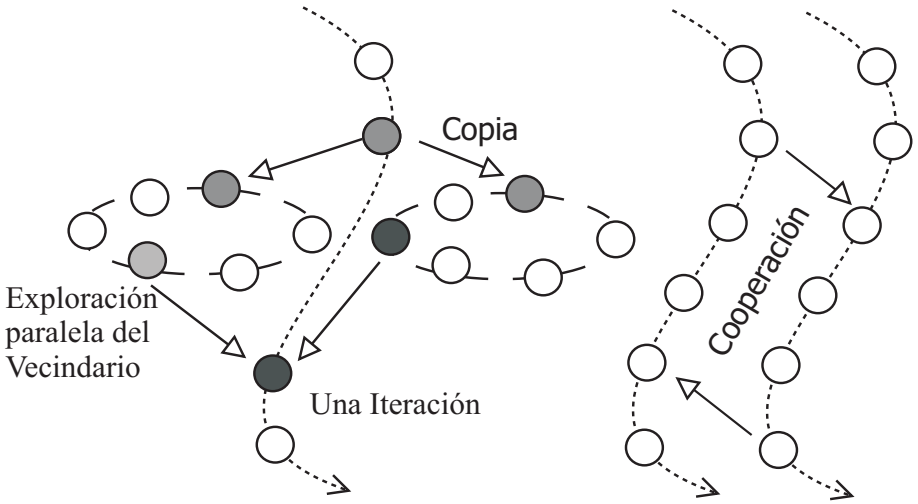


Figura 2.9: Modelos paralelos más usados en los métodos basados en trayectoria. A la izquierda se muestra el modelo de movimientos paralelos, donde se hace una exploración paralela del vecindario. A la derecha, se detalla el modelo de ejecuciones múltiples con cooperación, donde hay varios métodos ejecutándose en paralelo y cooperando entre ellos.

- **Modelo de movimientos paralelos:** los métodos basados en trayectoria en cada paso examinan parte de su vecindario y, de él, eligen la siguiente solución a considerar. Este paso suele ser computacionalmente costoso, ya que examinar el vecindario implica múltiples cálculos de la función de *fitness*. El modelo de movimientos paralelos tiene como objetivo acelerar dicho proceso mediante la exploración en paralelo del vecindario (véase el esquema de la izquierda de la Figura 2.9). Siguiendo un modelo maestro-esclavo, el maestro (el que ejecuta el algoritmo) pasa a cada esclavo la solución actual. Cada esclavo explora parte del vecindario de esta solución devolviendo la más prometedora. Entre todas estas soluciones devueltas el maestro elige una para continuar el proceso. Este modelo no cambia la semántica del algoritmo, sino que simplemente acelera su ejecución en caso de ser lanzado en una plataforma paralela. Este modelo es bastante popular debido a su simplicidad.

- **Modelo de aceleración del movimiento:** en muchos casos, el proceso más costoso del algoritmo es el cálculo de la función de *fitness*. Este cálculo, en muchos problemas, se puede descomponer en varios cómputos independientes más simples que, una vez llevados a cabo, se pueden combinar para obtener el valor final de la función de *fitness*. En este modelo, cada uno de esos cálculos más simples se asignan a los diferentes procesadores y se realizan en paralelo, acelerando el cálculo total. Al igual que el anterior, este modelo tampoco modifica la semántica del algoritmo respecto a su ejecución secuencial.

2.4.2. Modelos paralelos para métodos basados en población

El paralelismo surge de manera natural cuando se trabaja con poblaciones, ya que cada individuo puede manejarse de forma independiente. Debido a esto, el rendimiento de los algoritmos basados en población suele mejorar bastante cuando se ejecutan en paralelo. A alto nivel podemos dividir las estrategias de paralelización de este tipo de métodos en dos categorías: (1) paralelización del cómputo, donde las operaciones que se llevan a cabo sobre los individuos son ejecutadas en paralelo, y (2) paralelización de la población, donde se procede a la estructuración de la población.

Uno de los modelos más utilizado que sigue la primera de las estrategias es el denominado *maestro-esclavo* (también conocido como *paralelización global*). En este esquema, un proceso central realiza las operaciones que afectan a toda la población (como, por ejemplo, la selección en los algoritmos evolutivos) mientras que los procesos esclavos se encargan de las operaciones que afectan a los individuos independientemente (como la evaluación de la función de *fitness*, la mutación e incluso, en algunos casos, la recombinación). Con este modelo, la semántica del algoritmo paralelo no cambia respecto al secuencial pero el tiempo global de cómputo es reducido. Este tipo de estrategias son muy utilizadas en las situaciones donde el cálculo de la función de *fitness* es un proceso muy costoso en tiempo. Otra estrategia muy popular es la de acelerar el cómputo mediante la realización de múltiples ejecuciones independientes (sin ninguna interacción entre ellas) usando múltiples máquinas para, finalmente, quedarse con la mejor solución encontrada entre todas las ejecuciones. Al igual que ocurría con el modelo de múltiples ejecuciones sin cooperación de las metaheurísticas paralelas basadas en trayectoria, este esquema no cambia el comportamiento del algoritmo, pero permite reducir de forma importante el tiempo total de cómputo.

Al margen del modelo maestro-esclavo, la mayoría de los algoritmos paralelos basados en población encontrados en la literatura utilizan alguna clase de estructuración de los individuos de la población. Este esquema es ampliamente utilizado especialmente en el campo de los algoritmos evolutivos y es el que mejor ilustra esta categorización. Entre los esquemas más populares para estructurar la población encontramos el modelo *distribuido* (o de grano grueso) y el modelo *celular* (o de grano fino) (Alba and Tomassini, 2002).

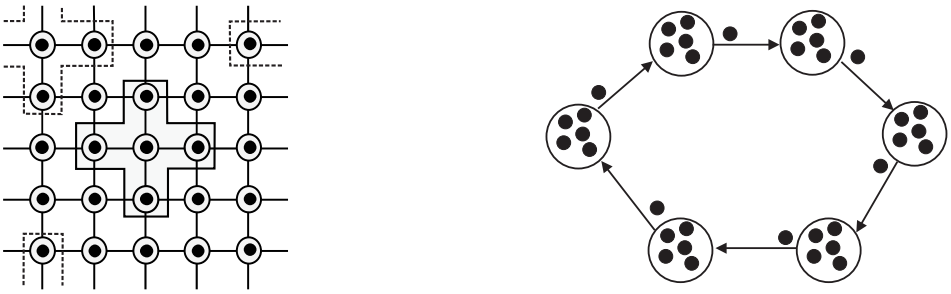


Figura 2.10: Los dos modelos más populares para estructurar la población: a la izquierda el modelo celular y a la derecha el modelo distribuido.

En el caso de los algoritmos distribuidos (Alba, 1999) (véase el esquema de la derecha en la Figura 2.10), la población se divide entre un conjunto de islas que ejecutan una metaheurística secuencial. Las islas cooperan entre sí mediante el intercambio de información (generalmente individuos, aunque nada impide intercambiar otro tipo de información). Esta cooperación permite introducir diversidad en las subpoblaciones, evitando caer así en los óptimos locales. Para terminar de definir este esquema el usuario debe dar una serie de parámetros como: la topología, que indica a dónde se envían los individuos de cada isla y de dónde se pueden recibir; el periodo de migración, que es el número de iteraciones entre dos intercambios de información; la tasa de migración, que es el número de individuos emigrados; el criterio de selección de los individuos a migrar y criterio de reemplazo, que indica si se reemplazan algunos individuos de la población actual para introducir a los inmigrantes y determina qué individuos se reemplazarán. Finalmente, se debe decidir si estos intercambios se realizan de forma síncrona o asíncrona.

Por otro lado, las metaheurísticas celulares (Dorronsoro, 2007) (véase el esquema de la izquierda en la Figura 2.10) se basan en el concepto de vecindario[1]. Cada individuo tiene a su alrededor un conjunto de individuos vecinos donde se lleva a cabo la explotación de las soluciones. La exploración y la difusión de las soluciones al resto de la población se produce debido a que los vecindarios están solapados, lo que produce que las buenas soluciones se extiendan lentamente por toda la población.

A parte de estos modelos básicos, en la literatura también se han propuesto modelos híbridos donde se implementan esquemas de dos niveles. Por ejemplo, una estrategia bastante común en la literatura es aquélla donde en el nivel más alto tenemos un esquema de grano grueso, mientras que cada subpoblación se organiza siguiendo un esquema celular.

2.5. Evaluación estadística de resultados

Como ya se ha comentado en varias ocasiones a lo largo de este documento, las metaheurísticas son técnicas no deterministas. Esto implica que diferentes ejecuciones del mismo algoritmo sobre un problema dado no tienen por qué encontrar la misma solución. Esta propiedad característica de las metaheurísticas supone un problema importante para los investigadores a la hora de evaluar sus resultados y, por tanto, a la hora de comparar su algoritmo con otros algoritmos existentes.

Existen algunos trabajos que abordan el análisis teórico para un gran número de heurísticas y problemas (Graham, 1969; Karp, 1977), pero dada la dificultad que entraña este tipo de análisis teórico, tradicionalmente se analiza el comportamiento de los algoritmos mediante comparaciones empíricas. Para ello es necesario definir indicadores que permitan estas comparaciones. Podemos encontrar, en general, dos tipos diferentes de indicadores. Por un lado, tenemos aquellos que miden

[1]De nuevo aquí, la palabra *vecindario* se usa en el sentido de definir un conjunto de individuos que serán vecinos de uno dado, como en PSO. No debe confundirse con el concepto de vecindario en el espacio de soluciones que se usa en la búsqueda local o en metaheurísticas como SA o VNS.

la calidad de las soluciones obtenidas. Dado que a lo largo del desarrollo de esta tesis se han abordado problemas de optimización tanto mono como multiobjetivo, hay que considerar indicadores de calidad diferentes para cada tipo ya que, si bien el resultado en el primer caso es una única solución (el óptimo global), en el segundo caso tenemos un conjunto de soluciones, el conjunto óptimo de Pareto (Ecuación 9). Por otro lado, están los indicadores que miden el rendimiento de los algoritmos y que hacen referencia a los tiempos de ejecución o a la cantidad de recursos computacionales utilizados. Aunque la discusión que incluimos en las secciones siguientes trata los dos tipos de indicadores por separado, ambos están íntimamente ligados y suelen utilizarse conjuntamente para la evaluación de metaheurísticas, ya que el objetivo de este tipo de algoritmos es encontrar soluciones de alta calidad en un tiempo razonable.

Una vez definidos los indicadores, hay que realizar un mínimo de ejecuciones independientes del algoritmo para obtener resultados estadísticamente consistentes. Un valor de 30 suele considerarse el mínimo aceptable, aunque valores de 100 son recomendables. No vale la mera inclusión de medias y desviaciones típicas (algo usual, e incorrecto, en la literatura), ya que se pueden obtener conclusiones erróneas. Así, es necesario realizar un análisis estadístico global para asegurar que estas conclusiones son significativas y no son provocadas por variaciones aleatorias. Este tema se aborda con más detenimiento en la Sección 2.5.5.

2.5.1. Indicadores de calidad

Estos indicadores son los más importantes a la hora de evaluar una metaheurística. Son distintos dependiendo de si se conoce o no la solución óptima del problema en cuestión (algo común para problema clásicos de la literatura, pero poco usual en problemas del mundo real). Como ya se ha mencionado anteriormente, hay que distinguir además entre indicadores para problemas monoobjetivo y multiobjetivo.

2.5.2. Indicadores para optimización monoobjetivo

Para instancias de problemas donde la solución óptima es conocida, es fácil definir un indicador para controlar la calidad de la metaheurística: el número de veces que aquella se alcanza (*hit rate*). Esta medida generalmente se define como el porcentaje de veces que se alcanza la solución óptima respecto al total de ejecuciones realizadas. Desgraciadamente, conocer la solución óptima no es un caso habitual para problemas realistas o, aunque se conozca, su cálculo puede ser tan pesado computacionalmente que lo que interesa es encontrar una buena aproximación en un tiempo menor. De hecho, es común que los experimentos con metaheurísticas estén limitados a realizar a lo sumo un esfuerzo computacional definido de antemano (visitar un máximo número de puntos del espacio de búsqueda o un tiempo máximo de ejecución).

En estos casos, cuando el óptimo no es conocido, se suelen usar medidas estadísticas del indicador correspondiente. Las más populares son la media y la mediana del mejor valor de fitness encontrado en cada ejecución independiente. En general, es necesario ofrecer otros datos estadísticos como la varianza o la desviación estándar, además del correspondiente análisis estadístico, para dar confianza estadística a los resultados.

En los problemas donde el óptimo es conocido se pueden usar ambas métricas, tanto el número de éxitos como la media/mediana del fitness final (o también del esfuerzo). Es más, al usar ambas se obtiene más información: por ejemplo, un bajo número de éxitos pero una alta precisión indica que raramente encuentra el óptimo pero que es un método robusto.

2.5.3. Indicadores para optimización multiobjetivo

Si bien el procedimiento para medir la calidad de las soluciones en problemas monoobjetivo está clara, dentro del campo multiobjetivo esto es un tema de investigación muy activo (Knowles et al, 2006; Zitzler et al, 2003), ya que el resultado de estos algoritmos es un conjunto de soluciones no dominadas y no una solución única. Hay que definir, por tanto, indicadores de calidad para aproximaciones al frente de Pareto. Hay normalmente dos aspectos a considerar para medir la calidad de un frente: convergencia y diversidad. La primera hace referencia a la distancia existente entre la aproximación y el frente de Pareto óptimo del problema, mientras que la segunda mide la uniformidad de la distribución de soluciones sobre el frente. Como ocurre para el caso monoobjetivo, existen indicadores atendiendo a si se conoce o no el frente óptimo. A continuación se muestran los indicadores de calidad utilizados en esta tesis (el lector interesado puede ver (Coello et al, 2007; Deb, 2001) para otros indicadores de calidad definidos en la literatura):

- **Número de óptimos de Pareto.** En la resolución de algunos problemas multiobjetivo muy complejos, encontrar un número alto de soluciones no dominadas puede ser una tarea realmente dura para el algoritmo. En este sentido, el número de óptimos de Pareto encontrados se puede utilizar como una medida de la capacidad del algoritmo para explorar los espacios de búsqueda definidos por el problema multiobjetivo.

- **Cubrimiento de conjuntos – C(A, B).** El *cubrimiento de conjuntos* $C(A, B)$ calcula la proporción de soluciones en el conjunto B que son dominadas por soluciones del conjunto A:

$$C(A,B) = \frac{|\{b \in B | \exists a \in A : a \preceq b\}|}{|B|} \quad . \tag{2.18}$$

  Un valor de la métrica $C(A, B) = 1$ significa que todos los miembros de B son dominados por A, mientras que $C(A, B) = 0$ significa que ningún miembro de B es dominado por A. De esta forma, cuanto mayor sea $C(A, B)$, mejor es el frente de Pareto A con respecto a B. Ya que el operador de dominancia no es simétrico, $C(A, B)$ no es necesariamente igual a $1 - C(B, A)$, y tanto $C(A, B)$ como $C(B, A)$ deben ser calculados para entender cuántas soluciones de A son cubiertas por B y viceversa.

- **Distancia generacional – GD.** Esta métrica fue introducida por Van Veldhuizen y Lamont (Van Veldhuizen and Lamont, 1998) para medir cómo de lejos están los elementos del conjunto de soluciones no dominadas encontradas respecto del conjunto óptimo de Pareto. Se define como:

$$GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \quad , \tag{2.19}$$

  donde n es el número de soluciones no dominadas, y d_i es la distancia Euclídea (medida en el espacio objetivo) entre cada una estas soluciones y el miembro más cercano del conjunto óptimo de Pareto. Según esta definición, esta claro que un valor de $GD = 0$ significa que todos los elementos generados están en el conjunto óptimo de Pareto.

- **Dispersión – Δ.** La *dispersión* o Δ (Deb et al, 2002a) es un indicador de diversidad que mide la distribución de las soluciones obtenidas. Esta métrica se define como:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} |d_i - \bar{d}|}{d_f + d_l + (N-1)\bar{d}} \quad , \tag{2.20}$$

donde d_i es la distancia Euclídea entre dos soluciones consecutivas, $\bar{d}$ es la media de estas distancias, y d_f y d_l son las distancias euclídeas a las soluciones *extremas* (límite) del frente óptimo en el espacio objetivo (para más detalles, consulte (Deb et al, 2002a)). Esta medida toma el valor cero para una distribución ideal, cuando hay una dispersión perfecta de las soluciones del frente de Pareto.

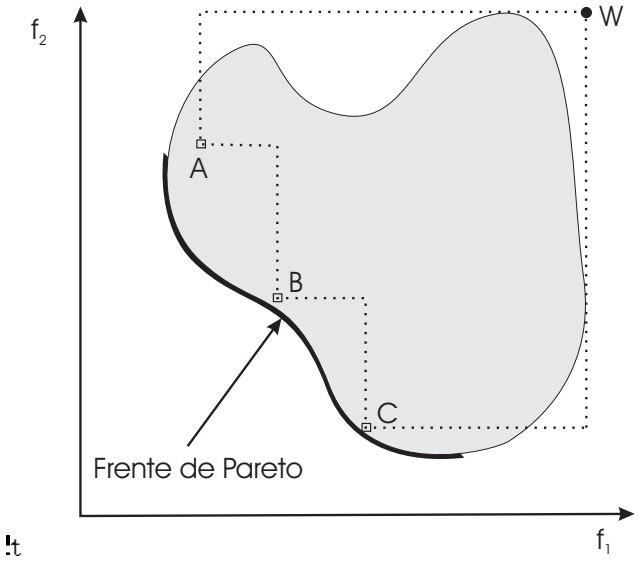


Figura 2.11: El hipervolumen cubierto por las soluciones no dominadas.

- **Hipervolumen – HV.** La métrica *hipervolumen* (Zitzler and Thiele, 1999a) es un indicador combinado de convergencia y diversidad que calcula el volumen, en el espacio de objetivos, cubierto por los miembros de un conjunto Q de soluciones no dominadas (la región acotada por la línea discontinua en la Figura 2.11, $Q = \{A, B, C\}$) para problemas en los que todos los objetivos deben ser minimizados. Matemáticamente, para cada solución $i \in Q$, un hipercubo v_i se construye utilizando un punto de referencia W (que puede estar compuesto por la peor solución para cada objetivo, por ejemplo) y la solución i como las esquinas de la diagonal del hipercubo. El punto de referencia se puede obtener simplemente construyendo un vector de los peores valores para las funciones. Así, HV se calcula como el volumen de la unión de todos los hipercubos:

$$HV = volumen\left(\bigcup_{i=1}^{|Q|} v_i\right) \quad . \tag{2.21}$$

Hay que indicar, finalmente, que para las tres últimas métricas presentadas es necesario normalizar de alguna forma los frentes obtenidos para no obtener resultados engañosos. Para el caso de GD y Δ el proceso está claro: se normaliza respecto al frente óptimo ya que hay que disponer de él obligatoriamente para calcular el valor de estos indicadores de calidad. Calcular HV no requiere frente de referencia alguno, lo que permite medir calidad de frentes sin necesidad de conocer el conjunto óptimo de Pareto. No obstante, esta métrica es muy dependiente de la escala de los objetivos y la normalización es casi imprescindible para conseguir valores fiables de este indicador. Así, si se dispone del frente óptimo, se normaliza respecto a éste. Si, por el contrario, el frente óptimo no es conocido, cuando se quieren comparar varios algoritmos para el problema, el procedimiento

que seguimos consiste en buscar los valores máximos y mínimos de todos los frentes del problema concreto en todos los algoritmos y normalizar respecto a esos valores. Así garantizamos que el punto de referencia para calcular los volúmenes v_i de HV es el mismo en todos los casos.

2.5.4. Indicadores de rendimiento

Entendemos por medidas de rendimiento aquellas que hacen referencia al tiempo o la cantidad de recursos computacionales utilizados por las metaheurísticas, que se suelen medir en base al número de soluciones visitadas del espacio de búsqueda (esfuerzo computacional) o al tiempo de ejecución.

Muchos investigadores prefieren el número de evaluaciones como manera de medir el esfuerzo computacional, ya que elimina los efectos particulares de la implementación, del software y del hardware, haciendo así que las comparaciones sean independientes de esos factores. Pero esta medida puede ser engañosa en algunos casos, ya que puede ocurrir que algunas evaluaciones tarden más que otras (algo muy común en programación genética (Koza, 1992)) o incluso que los operadores que manipulan las soluciones sean más costosos en una técnica u otra. En general, es recomendable usar las dos métricas (evaluaciones y tiempo) para obtener una medida realista del esfuerzo computacional.

Dado que vamos a evaluar algoritmos que se pueden ejecutar sobre plataformas de cómputo paralela, a continuación presentamos los indicadores que se han utilizado. En este sentido, el indicador más importante para los algoritmos paralelos es sin ninguna duda el *speedup*, que compara el tiempo de la ejecución secuencial con el tiempo correspondiente para el caso paralelo a la hora de resolver un problema. Si notamos por T_m el tiempo de ejecución para un algoritmo usando m procesadores, el speedup es el ratio entre la ejecución más rápida en un sistema mono-procesador T_1 y el tiempo de ejecución en m procesadores T_m:

$$s_m = \frac{T_1}{T_m} \tag{2.22}$$

Para algoritmos no deterministas no podemos usar esta métrica directamente. Para esa clase de métodos, se debería comparar el tiempo *medio* secuencial con el tiempo *medio* paralelo:

$$s_m = \frac{E[T_1]}{E[T_m]} \ . \tag{2.23}$$

La principal dificultad con esta medida es que los investigadores no se ponen de acuerdo en el significado de T_1 y T_m. En un estudio realizado por (Alba and Tomassini, 2002) se distingue entre diferentes definiciones de speedup dependiendo del significado de esos valores (véase la Tabla 2.1).

Tabla 2.1: Taxonomía de las medidas de speedup propuesta por (Alba and Tomassini, 2002).

I. Speedup Fuerte
II. Speedup Débil
A. Speedup con parada por calidad de soluciones
1. Versus panmixia
2. Ortodoxo
B. Speedup con un esfuerzo predefinido

El *speedup fuerte* (tipo I) compara el tiempo de ejecución paralelo respecto al mejor algoritmo secuencial. Esta es la definición más exacta de speedup, pero debido a lo complicado que es conseguir el algoritmo más eficiente actual la mayoría de los diseñadores de algoritmos paralelos no la usan. El *speedup débil* (tipo II) compara el algoritmo paralelo desarrollado por el investigador con su propia

versión secuencial. En este caso se puede usar dos criterios de parada: por la calidad de soluciones y por máximo esfuerzo. El autor de esta taxonomía descarta esta última definición debido a que compara algoritmos que no producen soluciones de similar calidad, lo que va en contra del espíritu de esta métrica. Para el speedup débil con parada basada en la calidad de soluciones se proponen dos variantes: comparar el algoritmo paralelo con la versión secuencial canónica (tipo II.A.1) o comparar el tiempo de ejecución del algoritmo paralelo en un procesador con el tiempo que tarda el mismo algoritmo pero en m procesadores (tipo II.A.2). En el primero es claro que se están comparando dos algoritmos diferentes.

Aunque el speedup es la medida más utilizada, también se han definido otras métricas que pueden ser útiles para medir el comportamiento del algoritmo paralelo. Presentamos aquí las dos que hemos utilizado en esta tesis, la eficiencia paralela y la fracción serie.

La *eficiencia* (Ecuación 2.24) es una normalización del speedup que muestra un valor entre 0 y 1 indicando el grado de aprovechamiento de los m procesadores:

$$e_m = \frac{s_m}{m} \ . \tag{2.24}$$

Finalmente, Karp y Flatt (Karp and Flatt, 1990) desarrollaron una interesante métrica para medir el rendimiento de cualquier algoritmo paralelo que puede ayudarnos a identificar factores más sutiles que los proporcionados por el speedup por sí sólo. Esta métrica se denomina *fracción serie* de un algoritmo (Ecuación 2.25).

$$f_m = \frac{1/s_m - 1/m}{1 - 1/m} \ . \tag{2.25}$$

Idealmente, la fracción serie debería permanecer constante para un algoritmo aunque se cambie la potencia de cálculo de la plataforma donde se ejecuta. Si el speedup es pequeño pero el valor de la fracción serie se mantiene constante para diferentes números de procesadores (m), entonces podemos concluir que la perdida de eficiencia es debida al limitado paralelismo del programa. Por otro lado, un incremento ligero de f_m puede indicar que la granularidad de las tareas es demasiado fina. Se puede dar un tercer escenario en el que se produce una reducción significativa en el valor de f_m, lo que indica que alguna clase de speedup superlineal.

2.5.5. Análisis estadístico de los resultados

Una vez definidos los indicadores de calidad y rendimiento, y realizadas un mínimo de ejecuciones independientes, se tiene un conjunto de datos por cada indicador, algoritmo y problema. Desde el punto de vista estadístico, estos datos se pueden considerar como muestras de una función de densidad de probabilidad y, para poder sacar conclusiones correctas, hay que realizar un análisis de los mismos para determinar si las diferencias observadas son significativas.

En los estudios que se han realizado en esta tesis se han usado varios tests estadísticos para la comparación múltiple de algoritmos, entre ellos los procedimientos no paramétricos: la suma de rangos de Wilcoxon y Friedman, y el procedimiento post-hoc de Holm.

2.5.5.1. Test de Wilcoxon

El test de suma de rangos de Wilcoxon, un test no paramétrico que permite hacer comparaciones dos a dos entre los resultados de pares de algoritmos para analizar la significancia estadística de los mismos (Demšar, 2006).

2.5.5.2. Test de Friedman

El test de Friedman es una prueba equivalente a la prueba ANOVA para medidas repetidas en la versión no paramétrica. Se trata de una extensión de la prueba de los rangos de Wilcoxon para más de dos grupos (basada en suma de rangos). Asumiendo ciertas simplificaciones, puede considerarse como una comparación entre las medianas de varios grupos. El método consiste en ordenar los datos por filas o bloques, reemplazándolos por su respectivo orden. Al ordenarlos, debemos considerar la existencia de datos idénticos.

2.5.5.3. Test de Holm

El test de Holm es un procedimiento incremental que define diferentes niveles de rechazo por hipótesis (Holm, 1979). Sea $p_1, ..., p_m$ los valores p ordenados (del más pequeño al más grande) y $H_i, ..., H_m$ las correspondientes hipótesis, el procedimiento de Holm rechaza H_1 hasta $H_{(i?1)}$ si i es el entero más pequeño tal que $p_i \succ \alpha/(m - i + 1)$ (Derrac et al, 2011).

En todos los trabajos se ha aplicado un nivel de confianza del 95 % (un nivel de significancia del 5 % o p-value menor de 0.05), lo que implica que la probabilidad de que las diferencias observadas no se deban al azar es del 95 %.

Capítulo 3

Inferencia Filogenética

3.1. Introducción

La Filogenética es el estudio de las relaciones evolutivas entre los organismos que comparten antepasados comunes. La estructura más común utilizada para representar estas relaciones es un árbol filogenético, que representa las relaciones ancestro-descendientes. Los datos de entrada utilizados para inferir los árboles pueden ser secuencias morfológicas o moleculares. En esta tesis nos hemos enfocado a trabajar con las secuencias moleculares (ADN, ARN o proteínas). Una secuencia molecular puede representarse como una cadena de caracteres, donde cada carácter es un elemento de un alfabeto finito. En el caso de los datos de ADN los cuatro nucleótidos (también llamados bases) son A, C, G, T. Para el caso de ARN, son A, C, G, U y para el caso de aminoácidos (AA), el alfabeto comprende un conjunto 20 caracteres que también se denominan residuos. Cada elemento del alfabeto representa un posible estado de carácter para un índice de secuencia dado (posición o sitio).

3.2. Árbol filogenético

En la teoría de grafos, una grafo es una representación de un conjunto de vértices donde algunos pares de vértices están conectados por bordes. En un grafo no dirigido los bordes no tienen orientación. Un camino es una secuencia de aristas que conectan una secuencia de vértices. Un árbol es un grafo no dirigido en el que cualquier par de vértices del árbol están conectados por exactamente un camino. Por lo tanto, en un árbol no tiene ciclos, es decir, no existen dos vértices conectados por más de un camino.

En Filogenética se habla de árboles filogenéticos, en los que la ramificación define la topología. Los vértices o nodos se conocen también como TUs (*Taxonomic Units* - Unidades Taxonómicas). Los bordes se denominan ramas. Los nodos terminales del árbol se llaman a menudo puntas, taxones u OTUs (*Operational Taxonomic Unit* - Unidad Taxonómica Operacional). Un extremo se llama taxón u OTU. Los taxones representan los organismos vivos (existentes) para los cuales se dispone de sus datos moleculares y se pueden secuenciar. Los nodos internos representan hipotéticos antepasados comunes extintos.

En general, un árbol filogenético es representado como un árbol binario no enraizado, donde todos los nodos tienen grado uno (hojas) o grado tres (nodos internos). Así que, dado N taxones, existen $n-2$ nodos internos (nodos ancestrales) y $2n-3$ bordes (Ramas).

Tabla 3.1: Número de posibles árboles para filogenias entre 5 y 50 especies.

Número Especies	Número de árboles alternativos
3	1
4	3
5	15
6	105
7	945
10	2.027.025
15	7.905.853.580.625
20	$2.21 * 10^{20}$
50	$2.84 * 10^{76}$

3.3. Complejidad del problema

El principal problema computacional de la inferencia filogenética consiste en el gran número de posibles topologías en el espacio de búsqueda, el cual crece exponencialmente con el número de especies a analizar. Dado n organismos, la cantidad de posibles árboles binarios sin raíz es (Edwards et al, 1964):

$$|SS| = \prod_{i=1}^{n}(2_i - 5) = \frac{(2n-5)!}{2^{n-3}(n-3)!} \tag{3.1}$$

Algunas cifras ejemplares de esta fórmula se describen en la Tabla 3.1. Hay que tener en cuenta que para 50 organismos existen casi tantas topologías de árboles alternativas como átomos en el universo ($\approx 10^{80}$).

Debido al enorme número de posibles combinaciones, los enfoques exhaustivos se vuelvan totalmente inviables desde un punto de vista computacional, al tratar de inferir filogenias con un número mayor a diez especies. Por esta esta explosión combinatoria, el problema de la inferencia filogenética es considerada como un problema de complejidad NP-Completo, lo cual ha sido demostrado formalmente tanto bajo un enfoque de Parsimonia (Day et al, 1986) como de Verosimilitud (Chor and Tuller, 2005).

3.4. Clasificación de los métodos

Uno de los criterios de clasificación más conocidos para el estudio de las historias evolutivas son los métodos filogenéticos. Bajo este criterio, los procedimientos filogenéticos se agrupan de la siguiente forma: métodos basados en caracteres y métodos basados en la distancia (Lemey, 2009). Por un lado, los métodos basados en caracteres operan directamente sobre los estados de los caracteres discretos descritos en el conjunto de secuencias de entrada. Esta clase de métodos generalmente procede generando un árbol inicial, por ejemplo, mediante la agrupación de los organismos en el orden en el que aparecen en los datos. Posteriormente, realizan una búsqueda basada en el estudio de la divergencia y similitudes observadas a nivel de los estados de los caracteres, modificando el árbol de arranque hasta que se logre una topología filogenética satisfactoria. Con el objetivo de evaluar la calidad de las filogenias, los criterios de optimización de calidad basados en caracteres suelen ser usados para guiar las búsquedas topológicas hacia árboles filogenéticos de alta calidad, considerando algunos principios biológicos y estadísticos. Ejemplos de enfoques basados en los caracteres están los métodos muy conocidos la Máxima Parsimonia (Fitch, 1971) y la Máxima Verosimilitud (Felsenstein, 1981) (más detalles de ambos criterios de optimización se detallan en las subsecciones 3.5.1 y 3.5.2).

Por otra parte, los métodos basados en distancias se centran en la obtención de historias filogenéticas a través del procesamiento de una matriz simétrica de distancias evolutivas NxN, donde

N es el número de especies en el conjuntos de secuencias de entrada. Cada entrada i, j en la matriz de distancia define una valor de tipo coma flotante de la distancia evolutiva o de pares genética observada entre dos OTUs i y j en el alineamiento múltiple de secuencias. Un primer enfoque para medir las distancias genéticas consiste en contar la fracción de posiciones en las que se encuentran diferencias entre cada par de secuencias, llamada p-distancia. Esta distancia suele ser complementada con mediciones probabilísticas y modelos matemáticos de la evolución molecular (Bos and Posada, 2005), generando así las distancias evolutivas. Una vez definidas estas distancias evolutivas, los métodos basados en distancias, construyen la filogenia agrupando las especies de acuerdo a su distancia. Ejemplos de enfoques basados en distancias son el Método de grupo de pares no ponderado con medias aritmética (*Unweighted Pair Group Method with Arithmetic Means* - UPGMA) (Sokal, 1958), el método del grupo de pares ponderado con medias aritméticos (*Weighted Pair Group Method with Arithmetic means* - WPGMA) (Sneath et al, 1973) y el método *Neighbour-joining* (NJ) (Saitou and Nei, 1987a).

3.5. Criterios de optimización

Los métodos filogenéticos se basan en la definición de un criterio de optimización, el cual especifica una medición de la calidad biológica de la filogenias, para guiar el proceso de inferencia. El objetivo principal es encontrar la solución más apta según el criterio de optimización implementado. Bajo esta perspectiva, la inferencia filogenética se aborda como un problema de optimización, que consiste en explorar en el espacio de búsqueda de árboles SS con el objetivo de encontrar la filogenia τ tal que $\tau \in SS$ y que optimiza una función objetivo $f(\tau)$, es decir, la hipótesis evolutiva más cercana a la filogenia perfecta. El alcance de esta tesis considera dos de los criterios de optimización más empleados el área de la Filogenética: La Máxima Parsimonia y Máxima Verosimilitud.

3.5.1. Máxima parsimonia

Entre las diferentes hipótesis que explican la naturaleza de un sistema, el razonamiento de Occam sugiere que la hipótesis más simple relativa a un fenómeno debe ser siempre la preferida. Esta declaración es muy aplicada en una amplia gama de dominios científicos, incluyendo la Bioinformática. El principio de la parsimonia es un análisis inspirado en este razonamiento.

Los métodos de máxima parsimonia tienen como objetivo encontrar un árbol que minimice el número de cambios de estado de caracteres (o pasos evolutivos) que se precisan para explicar los datos. Se prefiere aquel árbol cuya topología implique una menor cantidad de transformaciones a nivel molecular, constituyendo la descripción más simple de la historia evolutiva de los organismos de entrada (Felsenstein, 2004). El problema de la máxima parsimonia se describe de la siguiente manera: Sea D un conjunto de datos que contenga n especies. Cada especie tiene N sitios, donde d_{ij} es el carácter estado de la especie i en el sitio j. Dado el árbol τ con el conjunto de nodos $V(\tau)$ y el conjunto de ramas $E(\tau)$, el valor de parsimonia del árbol τ se define como (Swofford et al, 1996).

$$PS(\tau) = \sum_{j=1}^{N} \sum_{(v,u)\in E(\tau)} w_j C(v_j, u_j) \tag{3.2}$$

donde w_j se refiere al peso del sitio j, v_j y u_j son los estados de carácter de los nodos v y u en el sitio j para cada rama (u, v) en τ, respectivamente, y C es la matriz de costo, tal que $C(v_j, u_j)$ es el costo de cambiar de estado v_j a estado u_j. Las hojas de τ están etiquetadas por estados de carácter de especie de D, es decir, una hoja que representa la k-ésima especie tiene un estado de carácter d_{kj} para la posición j. Las siguientes propiedades se pueden observar a partir de la ecuación (3.2):

1. El criterio de parsimonia asume la independencia de los sitios, es decir, cada sitio es evaluado separadamente.

2. El cálculo del valor de parsimonia sólo considera la topología del árbol, por lo que no es necesaria otra información, como por ejemplo las longitudes de las ramas.

Existen varios métodos para computar la parsimonia de un árbol en la literatura, siendo uno de los más usados el conocido como Algoritmo de Fitch (Fitch, 1971), el cual asume la matriz de costo unitario de la siguiente manera:

$$C(v_j, u_j) = \begin{cases} 1 & Si\, v_j \neq u_j \\ 0 & caso\, contrario \end{cases} \tag{3.3}$$

Teniendo definida el algoritmo que minimice $PS(\tau)$ para un árbol τ, se debe encontrar el árbol τ^* tal que $PS(\tau^*)$ es el valor con el menor valor de parsimonia en todo el espacio de árboles.

3.5.2. Máxima verosimilutud

La verosimilitud es una función estadística que, aplicada a la Filogenética, indica la probabilidad de que la hipótesis evolutiva que plantea una topología de árbol filogenético y un modelo de evolución molecular Φ diera lugar al conjunto de organismos observados en los datos de entrada D (conjunto de secuencias alineadas). En otras palabras, mide la probabilidad de que la historia evolutiva sugerida por un método basado en un modelo de evolución de a lugar a la diversidad observada en el conjunto de secuencias alineadas (Swofford et al, 1996).

El análisis por máxima verosimilitud busca dar lugar a aquel árbol que represente la historia evolutiva más probable de los organismos de entrada. Puede definirse de la siguiente manera: La verosimilitud de un árbol filogenético, denotado por $L = P(D|\tau, \Phi)$, es la probabilidad condicional que se cumpla el conjunto de secuencias alineadas D dado un árbol τ y un modelo evolutivo Φ (Felsenstein, 2004).

Se debe tener en cuenta lo siguiente para calcular la verosimilitud:

1. El criterio de parsimonia asume la independencia de los sitios, es decir, cada sitio es evaluado separadamente.

2. La evolución de diferentes linajes es independiente, es decir, cada sub-árbol evoluciona separadamente.

Dado un árbol τ, $L = (\tau)$ se calcula a partir del producto de probabilidades parciales de todos los sitios:

$$L(\tau) = \prod_{j=1}^{N} L_j(\tau) \tag{3.4}$$

donde $L_j(\tau) = P(D_j|\tau, \Phi)$ es la verosimilitud en el sitio j, la cual puede se denota de la siguiente manera:

$$L_j(\tau) = \sum_{r_j} C_j(r_j, r).\pi_{r_j} \tag{3.5}$$

donde r es el nodo raíz de τ, r_j se refiere a cualquier posible estado de r en el sitio j, π_{r_j} es la frecuencia de estado r_j y $C_j(r_j, r)$ es la verosimilitud condicional del subárbol enraizado por r. Más específicamente,$C_j(r_j, r)$ es la probabilidad de que todo lo que se observa desde el nodo raíz r a las hojas del árbol τ, en el sitio j y dado r, tiene estado r_j. Sean u y v los nodos descendientes inmediatos de r, entonces $C_j(r_j, r)$ puede formularse como:

$$C_j(r_j, r) = \left[\sum_{u_j} C_j(u_j, u).P(r_j, u_j, t_{ru})\right]\left[\sum_{v_j} C_j(v_j, v).P(r_j, v_j, t_{rv})\right] \tag{3.6}$$

donde u_j y v_j se refieren a cualquier posible estado de los nodos u y v, respectivamente. t_{ru} y t_{rv} son las longitudes de las rama que conectan el nodo r con los nodos v y u, respectivamente. $P(r_j, u_j, t_{ru})$ es la probabilidad de cambiar del estado r_j al estado u_j durante el tiempo evolutivo t_{ru}. De forma similar, $P(r_j, v_j, t_{rv})$ es la probabilidad de cambiar del estado r_j al estado v_j en el tiempo t_{rv}. Ambas probabilidades son proporcionadas por el modelo evolutivo Φ.

Un método eficiente para calcular L fue propuesto por Felsenstein (Felsenstein, 2004) usando programación dinámica, donde L se obtiene por un recorrido post-orden en τ. Usualmente, es conveniente trabajar con valores logarítmicos de L, por lo que la ecuación (3.4) se puede definir como:

$$\ln L(\tau) = \prod_{j=1}^{N} \ln L_j(\tau) \tag{3.7}$$

El cálculo de la verosimilitud presentado en esta sección supone que los sitios evolucionan a tasas iguales. Sin embargo, esta suposición se infringe a menudo en datos con secuencias reales (Yang, 2006). Varios enfoques *among site-rate variations* (ASRV) pueden ser incorporados en el modelo evolutivo Φ. Uno de los más empleados el modelo discreto-gamma (Yang, 1994) donde las tasas variables en los sitios siguen una distribución discretizada Γ en una serie de categorías.

Varios estudios (Huelsenbeck, 1995; Tateno et al, 1994) han señalado que el uso de modelos ASRV puede mejorar los resultados en la inferencia de la verosimilitud. Sin embargo, hay que tener en cuenta que la inclusión de modelos ASRV puede aumentar el coste computacional de los cálculos. Con el fin de maximizar L para un árbol dado τ, es necesario optimizar los parámetros del modelo evolutivo Φ (es decir, las longitudes de las ramas y los parámetros del modelo de sustitución elegido), mediante métodos clásicos de optimización como Newton-Raphson o Gradiente (Felsenstein, 2004). En cuanto a los modelos de sustitución, se opta por llevar a cabo la configuración más apropiada en función del modelo elegido y las secuencias de datos a analizar.

3.6. Modelos evolutivos

Una de las principales ventajas de la máxima verosimilitud sobre otros métodos consiste en que permite explícitamente la especificación de un modelo evolutivo de secuencias. Un modelo de evolución describe las probabilidades con que se puede producir un cambio que implique la sustitución de aminoácidos o nucleótidos entre generaciones, esto es, la probabilidad de que se de la modificación de un carácter en las secuencias de los nodos del árbol. Los modelos de evolución utilizados típicamente en los análisis filogenéticos son clasificados como modelos de Markov reversibles en el tiempo; no están dirigidos con respecto al tiempo, es decir, la probabilidad de cualquier sustitución no depende de su historia sino sólo del predecesor inmediato.

Varios modelos han sido desarrollados para diversos tipos de secuencias (nucleótidos, aminoácidos y codones) (JC69,F84, HKY85, GTR...) (Felsenstein, 2004), y de su elección dependerá en gran medida el valor de verosimilitud obtenido. A continuación definiremos los modelos desde su forma más general a su forma más específica, es decir de arriba a abajo.

El modelo más general reversible en el tiempo se denomina, *General Time-Reversible* - GTR (Lanave et al, 1984). Los parámetros del modelo GTR incluyen las frecuencias de equilibrio de los cuatro nucleótidos $(\pi_A, \pi_C, \pi_G, \pi_T)$ y seis parámetros que representan las tasas relativas de sustitución entre cada par de nucleótidos (a, b, c, d, e, f). Hay que tener en cuenta que aunque

son diez parámetros, sólo ocho de ellos pueden ser definidos libremente, ya que las frecuencias de equilibrio deben sumar entre todas 1 y la tasa f se fija típicamente para asegurar la identificabilidad de las demás tasas relativas.

Estos parámetros son utilizados para definir una matriz Q de tamaño $4x4$, que representa al modelo concreto, donde Q_{ij} representa la tasa instantánea de sustitución de la base i a la base j. Usando la convención de ordenar las bases en orden alfabético (A, C, G, T):

$$Q = \begin{bmatrix} - & \mu a \pi_C & \mu b \pi_G & \mu c \pi_T \\ \mu a \pi_A & - & \mu d \pi_G & \mu e \pi_T \\ \mu b \pi_A & \mu d \pi_C & - & \mu f \pi_T \\ \mu c \pi_A & \mu e \pi_C & \mu f \pi_G & - \end{bmatrix} \tag{3.8}$$

El factor μ es la tasa de sustitución instantánea media. Los elementos de la diagonal principal de Q representan la suma negativa de los otros elementos en cada fila, y se omiten por razones de claridad. La matriz Q debe ser re-escala de manera que la tasa de sustitución media sea igual a uno, para mantener la definición adecuada de las longitudes de las ramas como el número esperado de sustituciones por sitio.

Una representación abstracta y más legible de los tipos de transición en el modelo GTR se ilustra en la Figura 3.1

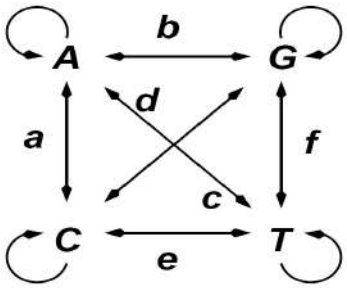


Figura 3.1: Representación esquemática de los parámetros del modelo GTR .

Otro de los modelos más empleados es el modelo evolutivo HKY85 (Hasegawa et al, 1985) (Hasegawa, Kishino, Yano, 1985), un modelo exacto y de mejor velocidad. HKY85 emplea las cuatro frecuencias de equilibrio $(\pi_A, \pi_C, \pi_G, \pi_T)$, pero sólo para dos clases de sustituciones de nucleótidos: Transiciones y Transversiones. La razón de esto es que las Transiciones ocurren entre bases que están químicamente más estrechamente relacionados : $A|G <->A|G$ y $C|T <->C|T$, y las Transversiones entre $A|G <->C|T$. En vista que, que las transiciones ocurren a una tasa diferente que las transversiones, esta relación de transición/transversión se expresa como la tasa κ.

El modelo HKY85 como se muestra a continuación y se deriva del modelo GTR estableciendo $\pi_R = \pi_A + \pi_G$, $\pi_Y = \pi_C + \pi_T$, $a = c = d = f = 1$ y $b = e = \kappa$

$$Q = \begin{bmatrix} -\mu(\kappa\pi_G + \pi_Y) & \mu\pi_C & \mu\kappa\pi_G & \mu\pi_T \\ \mu\pi_A & -\mu(\kappa\pi_T + \pi_R) & \mu\pi_G & \mu\kappa\pi_T \\ \mu\kappa\pi_A & \mu\pi_C & -\mu(\kappa\pi_A + \pi_Y) & \mu\pi_T \\ \mu\pi_A & \mu\kappa\pi_C & \mu\pi_G & -\mu(\kappa\pi_C + \pi_R) \end{bmatrix} \tag{3.9}$$

Dos modelos más simples se pueden derivar de HKY85 por cualquiera de los ajustes $\pi_A = \pi_C = \pi_G = \pi_T = 0{,}25$ para obtener el modelo de Kimura-2-Parameter - K2P (Kimura, 1980), representado de la siguiente manera:

$$Q = \begin{bmatrix} -\mu(\kappa+2) & \mu\pi_C & \mu\kappa\pi_G & \mu\pi_T \\ \mu\pi_A & -\mu(\kappa+2) & \mu\pi_G & \mu\kappa\pi_T \\ \mu\kappa\pi_A & \mu\pi_C & -\mu(\kappa+2) & \mu\pi_T \\ \mu\pi_A & \mu\kappa\pi_C & \mu\pi_G & -\mu(\kappa+2) \end{bmatrix} \tag{3.10}$$

O permitir un sólo un tipo de tasa de sustitución, es decir $a = b = c = d = e = f = 1$ y convertirse en el modelo Felsenstein 81 - F81 (Kimura, 1980), el cual está representado de la siguiente manera:

$$Q = \begin{bmatrix} -\mu(\pi_G+\pi_Y) & \mu\pi_C & \mu\kappa\pi_G & \mu\pi_T \\ \mu\pi_A & -\mu(\pi_T+\pi_R) & \mu\pi_G & \mu\kappa\pi_T \\ \mu\kappa\pi_A & \mu\pi_C & -\mu(\pi_A+\pi_Y) & \mu\pi_T \\ \mu\pi_A & \mu\kappa\pi_C & \mu\pi_G & -\mu(\pi_C+\pi_R) \end{bmatrix} \tag{3.11}$$

Finalmente, el modelo más simple y antiguo se conoce como Jukes-Cantor JC69 (Jukes and Cantor, 1969), el cual tiene frecuencias de base iguales, es decir $\pi_A = \pi_C = \pi_G = \pi_T = 0{,}25$, y sólo un tipo de sustitución, es decir $a = b = c = d = e = f = 1$. Este modelo está representado de la siguiente manera:

$$Q = \begin{bmatrix} -\mu\frac{3}{4} & \mu\frac{1}{4} & \mu\frac{1}{4} & \mu\frac{1}{4} \\ -\mu\frac{1}{4} & -\mu\frac{3}{4} & \mu\frac{1}{4} & \mu\frac{1}{4} \\ -\mu\frac{1}{4} & \mu\frac{1}{4} & -\mu\frac{3}{4} & \mu\frac{1}{4} \\ -\mu\frac{1}{4} & \mu\frac{1}{4} & \mu\frac{1}{4} & -\mu\frac{3}{4} \end{bmatrix} \tag{3.12}$$

3.7. Estado del arte

3.7.1. Algoritmos evolutivos monoobjetivo aplicados a la inferencia filogenética

A continuación detallaremos varias propuestas basadas en algoritmos evolutivos (EAs), (algoritmos genéticos (GAs) en su mayor parte), optimizando un solo criterio de reconstrucción (formulación monoobjetivo).

(Matsuda, 1995) realizó la primera aplicación de EAs para inferencia filogenética utilizando el criterio de máxima verosimilitud. (Lewis, 1998) propuso GAML, un algoritmo genético para la optimización de la máxima verosimilitud, que introduce un operador de cruce de intercambios de sub-árboles y un operador de mutación basado en SPR (Sub-tree Pruning and Regrafting). En su estudio, Lewis utilizó el método evolutivo HKY85 (Hasegawa et al, 1985), cuyos parámetros (κ, πA, πC, πG y πT) se incluyen en la codificación de los individuos de la población. Así, GAML optimiza la topología de árbol, las longitudes de las ramas y los parámetros del modelo HKY85, simultáneamente.

(Katoh et al, 2001) propusieron GA-mt, un algoritmo genético para la máxima verosimilitud que genera múltiples árboles en la población final. Estos árboles incluyen el árbol con mejor score de máxima verosimilitud y múltiples alternativas que no son significativamente peores en comparación con el mejor. GA-mt también considera los modelos ASRV (among site-rate variations) en el cálculo de la verosimilitud. El operador de cruce es un operador de intercambio de árboles y la mutación se basa en los movimientos topológicos TBR (Tree Bisection and Reconnection). La población inicial de GA-mt es generada a partir de árboles iniciales tomados de análisis bootstrap (Felsenstein, 2004).

Lemmon y Milinkovitch desarrollaron METAPIGA (Lemmon and Milinkovitch, 2002), un GA Metapoblacional para la inferencia filogenética utilizando la máxima verosimilitud. En METAPIGA varias poblaciones evolucionan simultáneamente y cooperan en la búsqueda de las soluciones óptimas. METAPIGA combina ventajas como la rápida búsqueda de árboles, identificación de múltiples óptimos, control sobre la velocidad y exactitud del algoritmo y una interfaz fácil de usar. Otro elemento clave propuesto por los autores es el mecanismo de poda por consenso. Este procedimiento identifica las regiones comunes (particiones) que son compartidas por los árboles en las poblaciones. Estas regiones son protegidas contra posibles cambios sobre ellas por alguna modificación topológica. Por lo tanto, la búsqueda sólo se enfoca en las regiones desprotegidas hasta que no se permitan más cambios. METAPIGA incluye un operador de cruce de intercambio de sub-árboles y varios operadores de mutación basado en SPR, NNI (Nearest Neighbor Interchange), intercambio de hojas e intercambios de sub-árboles. Estos operadores sólo se aplican si no destruyen cualquier región protegida de consenso.

(Zwickl, 2006) propuso un nuevo enfoque basado en algoritmos genéticos llamado GARLI (Genetic Algorithm for Rapid Likelihood). GARLI fue desarrollado para encontrar el árbol de máxima verosimilitud para pequeñas y grandes cantidades de secuencia de datos (nucleótidos, aminoácidos y codones). El autor introduce varias mejoras en la búsqueda topológicas y la optimización de las longitudes de las rama, reduciendo significativamente el tiempo computacional necesario para realizar ambas tareas. Por ejemplo, en lugar de optimizar todas las ramas del árbol, GARLI optimiza una rama si la mejora de la verosimilitud de árbol es mayor a un valor predeterminado. Por lo tanto, sólo las ramas que conducen a un aumento significativo de la verosimilitud son considerados para la optimización. Además el autor propuso varias versiones paralelas de GARLI.

GAs y una búsqueda local fueron combinados por (Moilanen, 2001) en PARSIGAL, un algoritmo genético híbrido para la Inferencia filogenética utilizando el criterio de máxima parsimonia. PARSIGAL utiliza un operador de cruce de intercambio de sub-árboles y, en lugar de mutación, utiliza un enfoque de búsqueda local basado en movimientos topológicos NNI y TBR. Usando este algoritmo híbrido, el GA define regiones prometedoras que podrían contener un óptimo global, mientras que la búsqueda local alcanza rápidamente una solución. PARSIGAL también incluye heurísticas para un rápido y parcial calculo de la parsimonia después de las modificaciones topológicas realizadas por la búsqueda local.

(Congdon, 2002) propuso un GA, llamada GAPHYL, que utiliza el criterio de parsimonia para la inferencia de árboles filogenéticos. GAPHYL utiliza varias sub-poblaciones para evitar prematuras convergencias, un operador de cruce de intercambio de sub-árboles y un operador de mutación de intercambio de hojas.

(Cotta and Moscato, 2002a) propusieron otra aplicación de GAs para la inferencia filogenética basadas en criterios de matrices de distancia.

Los resultados experimentales de las investigaciones descritas anteriormente han demostrado que los GAs tienen un mejor rendimiento y una mejor precisión en comparación con las heurísticas implementadas en software filogenéticos como PHYLIP y PAUP*. Por otra parte, los GAs también son adecuados para su uso con varios criterios de calidad para resolver problemas de optimización multiobjetivo; la siguiente sección describe varios ejemplos de su implementación.

3.7.2. Algoritmos evolutivos multiobjetivo aplicados a la inferencia filogenética

En los últimos años varios autores han propuesto enfoques multiobjetivo para abordar las principales fuentes de incongruencia en el proceso de inferencia filogenética basada en un solo criterio de optimización.

El primer algoritmo multiobjetivo aplicado a la filogenética fue propuesto por (Poladian and Jermiin, 2005), con el objetivo de tratar con diferentes fuentes de datos que proporcionan infor-

mación conflictiva sobre el proceso evolutivo. Por otra parte, otros investigadores se centraron en la necesidad de inferir historias evolutivas de acuerdo a múltiple criterios simultáneamente. (Coelho et al, 2010) propusieron un algoritmo multiobjetivo inmune inspirado empleando dos criterios basados en distancias: evolución mínima y error medio cuadrático. (Cancino and Delbem, 2007a) propusieron su algoritmo PhyloMOEA, el cual está basado en la referencia clásica NSGA-II y empleado como objetivos a optimizar los criterios de reconstrucción máxima parsimonia y máxima verosimilitud. Luego su propuesta se amplió en (Cancino and Delbem, 2010), considerando ASRV (*among-site rate variation*) sobre su modelo evolutivo HKY85 · Γ para mejorar los resultados de verosimilitud.

Por último, Santander-Jiménez *et. al* publicaron la adaptación y desarrollo de varias metaheurísticas multiobjetivo bioinspiradas, las cuales podemos agrupar según dos aspectos algorítmicos de la optimización multiobjetivo: dominancia de Pareto y técnicas basadas en indicadores de calidad. Entre los primeros están: Multiobjective Artificial Bee Colony Algorithm (MOABC) (Santander-Jiménez and Vega-Rodríguez, 2013a), un algoritmo de inteligencia de enjambre inspirado en el comportamiento colectivo de las abejas melíferas en la naturaleza; Multiobjective Firefly Algorithm (MO-FA) (Santander-Jiménez and Vega-Rodríguez, 2013b), una metaheurística de inteligencia de enjambre bioinspirada en la Bioluminiscencia de las luciérnagas; e Hybrid OpenMP/MPI NSGA-II (MO-Phyl) (Santander-Jiménez and Vega-Rodríguez, 2015), un enfoque híbrido entre OpenMP/MPI para paralelizar la principal referencia metaheurística multiobjetivo NSGA-II. Entre algoritmos basados en indicadores de calidad multiobjetivo se citan: Indicator-Based Multiobjective Bat Algorithm (IMOBA), un algoritmo de inteligencia de enjambre bioinspiradas en las capacidades de ecolocalización de los murciélagos, e Indicator-Based Evolutionary Algorithm (IBEA), una implementación del framework propuesto por Zitzler y Künzli en (Zitzler and Künzli, 2004) para integrar el cálculo de indicadores de calidad multiobjetivo en motores de búsqueda algorítmicos.

3.7.3. Codificaciones de Árboles

Una de las primeras representaciones propuestas es el código Prüfer (Prüfer, 1918), una secuencia única de representación de los árboles con n vértices $\{1, ..., n\}$ con una lista de $n-2$ posiciones de esos símbolos. Estas secuencias fueron usadas por primera vez por Heinz Prüfer para probar la fórmula de Cayley en 1918. Esta representación ha demostrado tener un bajo rendimiento precisamente empleada en algoritmos evolutivos. Uno de los primeros trabajos que empleó ésta representación fue presentada en (Reijmers et al, 1999), un modelo que presentó deficiencias con respecto a la jerarquía de los árboles filogenéticos. Además (Gen and Li, 1999; Gottlieb et al, 2001) realizaron implementaciones de esta representación. Gottlieb *et al.* sugieren que podría limitar la eficacia de la exploración en el espacio de búsqueda en algunos casos. (Cotta and Moscato, 2002b) proporcionan un estudio comparativo de la representación directa e indirecta. Han sugerido que la elección de la representación (y sus operadores correspondientes) tiene un impacto significativo en la calidad de las soluciones obtenidas. Entre los casos particulares está el algoritmo evolutivo multiobjetivo PhyMOEA (Hassan et al, 2008) aplicado a la inferencia filogenética, en el que se emplea el método indirecto de Prüfer para codificar los árboles filogenéticos representados en formato Newick.

(Picciotto, 1999) identificó tres mapeos entre los códigos Prüfer y los árboles de expansión: Dandelion code, Blob code y Happy code. En 2001, Julstrom presentó una implementación del Código Blob (Julstrom, 2001), una alternativa identificada y adaptada para su uso en algoritmos genéticos. Julstrom demostró que esta representación exhibía un rendimiento significativamente más alto que las representaciones del código Prüfer. Así mismo (Thompson et al, 2007) presentaron una implementación de los códigos Dandelion Code que, a pesar que no logra mejorar los rendimientos de las representaciones directas o NetKeys, presenta ser una alternativa fuerte, en particular para los redes de nodos muy grandes.

Entre las mas recientes y mejores representaciones están: Network Random Keys (NetKeys) presentada por (Rothlauf et al, 2002) y Edge sets presentada por (Raidl and Julstrom, 2003). En la representación de NetKeys un cromosoma asigna a cada borde en la red una calificación de su importancia, definida como su peso, un número real en el rango [0,1]. Un árbol de expansión es descodificado desde el cromosoma mediante la adición de bordes de la red a un gráfico inicialmente vacío en orden de importancia, ignorando los bordes que introducen ciclos. Una vez que se han añadido los $n-1$ bordes, ya puede ser identificado el árbol.

En la presente capítulo de nuestra tesis hemos usada la codificación de representación directa de los árboles filogenéticos. En esta representación, la identidad de todos los $n-1$ bordes en el árbol se pueden identificar directamente en los cromosomas de los individuos. Un nodo r se designa como el nodo raíz del árbol y, para cada nodo i, el predecesor inmediato p_i en la trayectoria desde i a r se almacena. Un árbol de expansión $T = (V, E)$ se codifica como el vector $P = \{p_1, p_2, ..., p_{n-1}\}$, donde $(i, p_i) \in E$ y n es designado como el nodo raíz.

3.8. Herramienta software: MO-Phylogenetics

Comúnmente, los árboles filogenéticos inferidos por los biólogos se obtienen al optimizar un solo criterio de calidad. Estas filogenias pueden ser diferentes según el método utilizado: Máxima Parsimonia, Máxima Verosimilitud o métodos basados en matrices de distancia. Pueden además inferir discordantes relaciones genealógicas que conducen a diferentes enfoques según los datos observados. La optimización multiobjetivo, aplicada a este problema, permite obtener un conjunto de árboles filogenéticos que representan soluciones de compromiso entre la parsimonia y la verosimilitud, introduciendo hipótesis alternativas que pueden ser útiles desde un punto de vista biológico.

Con el objetivo de hacer frente al problema de la inferencia filogenética con técnicas multiobjetivo representativas del estado del arte hemos desarrollado el software de optimización aplicado a la inferencia filogenética llamado MO-Phylogenetics, integrando las técnicas multiobjetivo del framework jMetalCpp (López-Camacho et al, 2014), con las funcionalidades bioinformáticas del framework BIO++ (Dutheil et al, 2006) y las funciones filogenéticas de la librería PLL (Phylogenetic Likelihood Library) (Flouri et al, 2015). Este trabajo ha sido publicado por la revista internacional *Methods in Ecology and Evolution* (Zambrano-Vega et al, 2016). El proyecto software se encuentra alojado en el repositorio GitHub[1][2], y su código fuente puede ser accedido libremente.

Hasta donde sabemos, nuestra propuesta es el primer software de código abierto que proporciona un framework de optimización multiobjetivo dirigido a la inferencia filogenética.

3.8.1. Componentes del framework

MO-Phylogenetics ha sido desarrollado integrando las funcionalidades de tres frameworks software multidisciplinares, jMetalCpp (López-Camacho et al, 2014) un framework de optimización multiobjetivo, el conjunto de librerías bioinformáticas BIO++ (Dutheil et al, 2006) y la librería filogenética PLL (Phylogenetic Likelihood Library) (Flouri et al, 2015).

Entre la lista de los algoritmos que dispone en jMetalCpp y que han sido incluidos y adaptados en el software están: dos técnicas clásicas, pero aun comúnmente usadas, *Non-dominated Sorting Genetic Algorithm-II* (NSGA-II) (Deb et al, 2002b) y Pareto Archived Evolution Strategy (PAES) (Knowles and Corne, 1999b), y dos técnicas más recientes, como son Multiobjective Evolutionary Algorithm Based on Decomposition (MOEA/D) (Zhang and Li, 2007) y Multiobjective selection based on dominated hypervolume (SMS-EMOA) (Beume et al, 2007).

[1]Proyecto MO-Phylogenetics en GitHub: https://github.com/cristianzambrano/MO-Phylogenetics
[2]Página Web: http://khaos.uma.es/mophylogenetics/

Bio · · es un framework bioinformático que provee un conjunto de librerías para el desarrollo de software para varias áreas de la BioInformática: análisis de secuencias biológicas, inferencia filogenética, evolución molecular y genética poblacional. Entre sus librerías podemos citar a *SeqLib* (*Sequence Library*), la cual incluye funciones para manipular y analizar secuencias biológicas de ADN, ARN, proteínas y secuencias de codones. Su librería filogenética *PhylLib* (*Phylogenetics Library*) proporciona varias funcionalidades para la lectura, almacenamiento y manipulación de árboles filogenéticos, además de funciones para inferir filogenias bajos los métodos basados en caracteres como la máxima parsimonia, máxima verosimilitud y métodos basados en distancias como *Neighbour-joining*. Para el desarrollo de *MO-Phylogenetics* hemos utilizado ambas librerías (PhylLib y SeqLib); para la lectura y manejo de las secuencias de entrada (alineamientos múltiples de secuencias) usamos SeqLib, la cual provee soporte formatos Phylip y Fasta; y para la representación de las soluciones de los algoritmos (árboles filogenéticos) e implementación de los operadores genéticos usamos la clase *TreeTemplate* de PhylLib. Para evaluar el criterio de la Máxima Parsimonia de los árboles filogenéticos se usaron también las funcionalidades de la librería filogenética PhyLib.

La librería filogenética (PLL) es una librería altamente optimizada que provee funcionalidades para la estimación computacional de la máxima verosimilitud sobre árboles filogenéticos (*PLF Phylogenetic Likelihood Function*) proporcionando algunas características que apuntan a reducir los altos costos computacionales requeridos por esta función. Además provee métodos para la optimización de la longitudes de ramas de los árboles filogenéticos y una función adaptada a un esquema de vectorización que permite una rápida y eficiente exploración del espacio de búsqueda de árboles filogenéticos mediante modificaciones topológicas y mejoramiento de las longitudes de las ramas afectas por los cambios, lo que permite inferir rápidamente filogenias bajo el criterio de la máxima verosimilitud. Para el desarrollo de MO-Phylogenetics se han incorporado las funcionalidades de la función PLF para evaluar el objetivo a optimizar de la Máxima Verosimilitud con el modelo evolutivo GTR · Γ y la función de exploración del espacio de búsqueda como una búsqueda local incorporada a los algoritmos

3.8.2. Adaptación filogenética de las metaheurísticas multiobjetivo

Todas las metaheurísticas del software han sido adaptadas al problema de la inferencia filogenética con las siguientes fases en la estructura estándar de un algoritmo genético:

3.8.2.1. Población inicial

El punto de partida de los algoritmos pueden ser definido de tres formas:

- **Aleatoria**: se crean árboles cuya topología es generada de forma totalmente aleatoria con todas las longitudes de sus ramas establecidas en 0.05, normalmente estos árboles suelen estar muy alejados de los óptimos globales tanto en parsimonia como verosimilitud por lo que la convergencia de sus algoritmos podría verse seriamente afectada.

- **Definidos por usuario**: árboles filogenéticos en formato newick previamente inferidos mediante técnicas de bootstrap (Felsenstein, 2004) bajo el criterio de la máxima parsimonia, máxima verosimilitud o combinados; esta estrategia es usada por otros GA aplicados a la inferencia filogenética (Lemmon and Milinkovitch, 2002; Katoh et al, 2001; Cancino and Delbem, 2007a; Santander-Jiménez and Vega-Rodríguez, 2013a).

- **Árboles generados mediante el método Stepwise-addition** (Felsenstein, 2004): (Stamatakis, 2004) sugiere empezar con este tipo de árboles parsimoniosos por dos razones: la

parsimonia está relacionada directamente con la verosimilitud, al menos con modelos de sustitución simples (Tuffley and Steel, 1997) y, a diferencia de los métodos basados en distancias como BIONJ, Stepwise-addition es un método no determinista lo que permite obtener diferentes topologías de inicio en cada ejecución de independiente de los algoritmos y poder alcanzar óptimo globales entre varios ejecuciones (Zwickl, 2006).

Durante la inicialización de la población se pueden optimizar los parámetros del modelo evolutivo y las longitudes de las ramas de cada topología usando técnicas de optimización como Newton-Raphson (Press et al, 1992), Gradient o Brent (Brent, 1973).

3.8.2.2. Operadores evolutivos

De los varios operadores de cruce usados en la literatura (Matsuda, 1995; Congdon, 2002; Lewis, 1998), MO-Phylogenetics proporciona el operador *Prune-Delete-Graft* (PDG), el cual ha demostrado generar mejores resultados sobre los diferentes criterios de optimización (Cotta and Moscato, 2002a; Gallardo et al, 2007). Este operador combina un subárbol de dos árboles padres y crea dos nuevos árboles descendientes. Dado los árboles T_1 y T_2, este operador realiza los siguientes pasos:

1. Poda un subárbol s escogido al azar desde T_1;
2. Quita todas las hojas de T_2 que también están en s;
3. Crea un nuevo descendiente T_1' insertando el subárbol s a un borde aleatorio de T_2.

El segundo descendiente, denominado T_2', se crea de manera similar: poda un subárbol de T_2 y lo inserta en T_1. La figura 3.2 ilustra este operador.

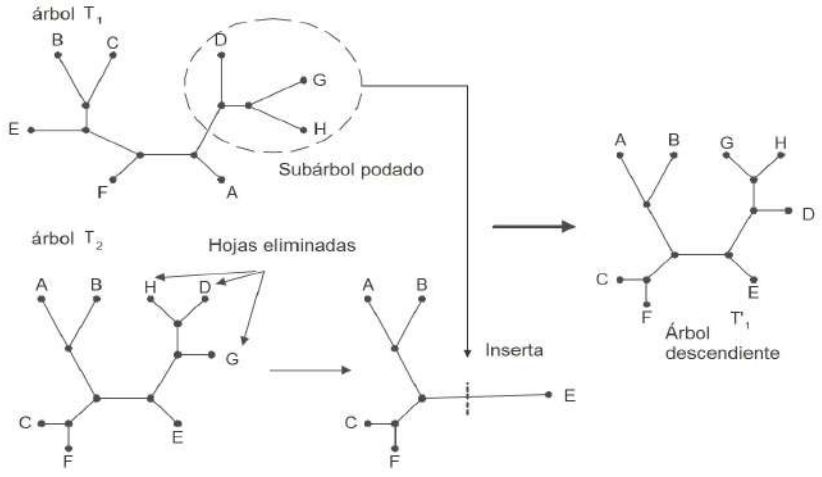
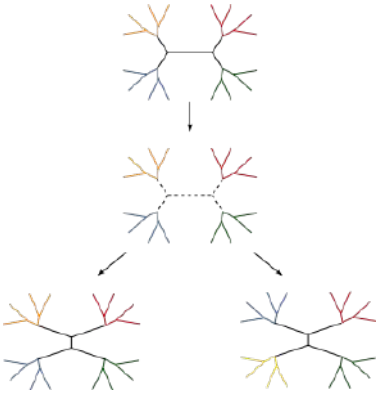
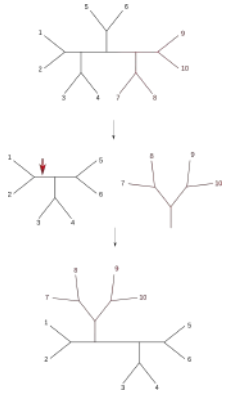


Figura 3.2: Ejemplo del operador de cruce PDG

En MO-Phylogenetics se disponen tres tipos de operadores de mutación de modificaciones topológicas: *Nearest Neighbour Interchange* (NNI), y *Subtree Pruning and Regrafting* (SPR) (Felsenstein, 2004). NNI selecciona aleatoriamente una rama del árbol y realiza un intercambio entre dos nodos de la rama. Por otro lado, SPR selecciona un subárbol del árbol, lo quita del árbol y lo reinserta en un lugar diferente para generar un nuevo árbol. El funcionamiento de estos dos operadores se ilustra en la imagen 3.3



(a) Nearest Neighbour Interchange



(b) Subtree Pruning and Regrafting

Figura 3.3: Ejemplos de los operadores de mutación (a)NNI y (b) SPR.

3.8.2.3. Técnicas de optimización filogenética

El software dispone de dos técnicas específicas para la inferencia de árboles filogenéticos optimizando los criterios de máxima parsimonia y máxima verosimilitud de forma simultánea. La primera técnica está basada en la alta relación que existe entre la parsimonia y la verosimilitud, en la que bajo una perspectiva teórica definida por (Steel and Penny, 2000) se establece que minimizar la parsimonia es equivalente a maximizar la verosimilitud bajo ciertos supuestos (ideal en nuestro enfoque), por lo que al igual a una de las técnicas de búsquedas implementadas en el software PhyML basadas en este principio (Guindon et al, 2010), MO-Phylogenetics explora el espacio de árboles encontrando soluciones multiobjetivo aplicando movimientos topológicos que minimizan la parsimonia y ajustando las longitudes de las ramas luego de cambios realizados. Para la optimización de la parsimonia se emplea la técnica *Parametric Progressive Tree Neighbourhood* (PPN) propuesta por (Goëffon et al, 2008), la cual está definida como un conjunto de movimientos topológicos SPR en los que las distancia de la rama del subárbol podado y la rama en la que se realizará el injerto está basada por d, cuyo valor inicial es la distancia máxima que hay entre el nodo raíz y las hojas del árbol, permitiendo empezar con movimientos globales, reduciendo progresivamente su valor hasta llegar a un valor mínimo de 1, con el cual los movimientos SPR representan un movimiento local similar a los que se obtienen con la técnica NNI. Solo aquellos movimientos que mejoren la parsimonia son aplicados definitivamente sobre la topología con el fin de ir optimizándola.

La segunda técnica es una combinación parametrizada de dos técnicas enfocadas a mejorar los criterios de optimización por separado: para la parsimonia usa PPN, cuyo funcionamiento es el mismo definido en la técnica 1, y por el lado de la verosimilitud una técnica de exploración basada en reordenamiento de topologías de la librería *Phylogenetic Likelihood Library* (PLL) llamada *pllRearrangeSearch*, la cual realiza movimientos topológicos de tipo NNI o SPR desde un nodo específico con todos los demás nodos a su alrededor dentro de un radio de cobertura. Gracias al esquema genérico de vectorización en el que está desarrollada PLL, se puede recalcular de forma rápida y óptima la verosimilitud de los árboles después de cada movimiento topológico probado realizando simplemente evaluaciones parciales de acuerdo sobre los cambios realizados.

3.8.2.4. Intervalo de actualización de parámetros

Una misma topología posee un solo valor de parsimonia pero también posee diferentes valores de verosimilitud según las longitudes de sus ramas y los parámetros del modelo de sustitución que se use (Stamatakis, 2004). Es por esto que el software incluye varias técnicas para la optimización de estos parámetros, como Newton-Rapson (Press et al, 1992) y Gradient y Brent (Brent, 1973), las cuales pueden ser parametrizadas para ser usadas antes, durante y al final de la ejecución de los algoritmos.

Con la finalidad de ir ajustando todos parámetros requeridos en la estimación de la verosimilitud de los árboles filogenéticos, el software tiene definido un intervalo de actualización cada cierto número de evaluaciones de los algoritmos, en el que específicamente se optimiza las longitudes de las ramas y los parámetros del Modelo de Sustitución GTR $\cdot$ Γ: Tasas de Sustitución de nucleótidos $\pi_A, \pi_C, \pi_G, \pi_T$ y la tasa de sustitución entre sitios alpha (α).

3.8.2.5. Formato de los resultados

Los resultados de MO-Phylogenetics están basados en el formato de los resultados de salida de jMetalCpp, en el que crea dos archivos planos (denominados por defecto FUN y VAR); el primero contiene los datos de los frentes de Pareto aproximados y el segundo archivo contiene los árboles filogenéticos optimizados bi-objetivos en formato newick.

3.8.3. Ejemplos de uso

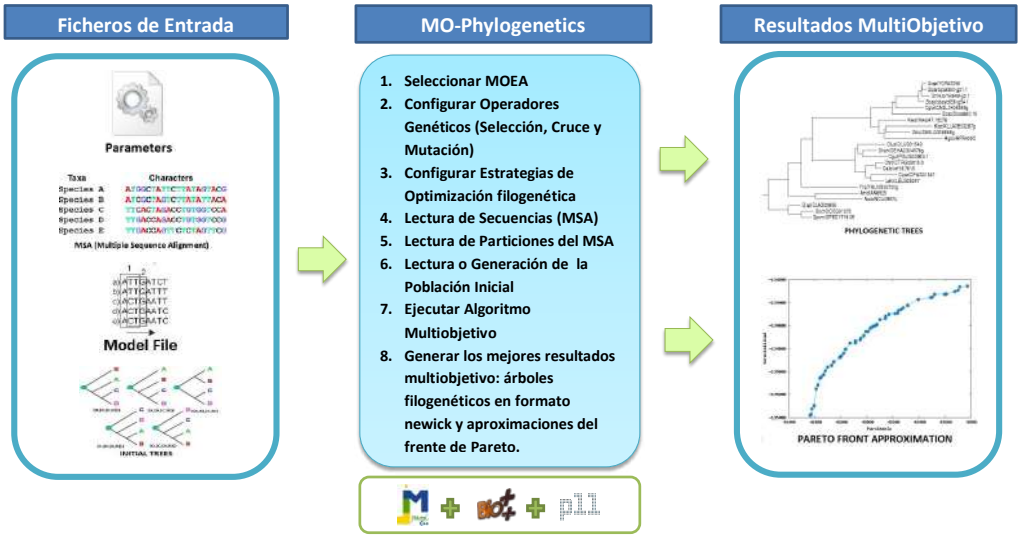


Figura 3.4: Esquema de MO-Phylogenetics. MOEA hace referencia a Multi-Objective Evolutionary Algorithm.

MO-Phylogenetics puede ser fácilmente configurado a través de un archivo de configuraciones (ver Figura 3.4). La lista de los parámetros se detallan en la Tabla D.1 del apéndice D.

Para su ejecución se debe crear este archivo de texto plano con los parámetros obligatorios (marcados con * en la Tabla D.1 del apéndice D) y con los demás parámetros según sea necesario. Para ejecutarlo, debemos ejecutar el siguiente comando:

```
$ MOPhylogenetics param = parametersfile
```

donde *parametersfile* es el nombre del archivo que contiene todos los parámetros requeridos para ejecutar el programa.

A continuación se describen tres ejemplos de cómo configurar MO-Phylogenetics y llevar a cabo experimentos sobre tres conjuntos de datos de nucleótidos y utilizando diferentes algoritmos y técnicas de optimización.

3.8.3.1. Conjunto de problemas de pruebas

El conjunto de secuencias utilizadas para los diferentes ejemplos está formado de cuatro conjuntos de datos de nucleótidos:

1. rbcL_55: comprende 55 secuencias (cada secuencia tiene 1314 sitios) del gen rbcL Cloroplasto de plantas verdes (Lewis, 1998);

2. mtDNA_186: Contiene 186 secuencias de ADN mitocondrial humano (cada secuencia con 16608 sitios) obtenidos de la base de datos del genoma humano mitocondrial (MtDB) (Ingman and Gyllensten, 2006);

3. RDPII_218: comprende 218 secuencias procarióticas de ARN (cada secuencia contiene 4182 sitios) tomado del Proyecto de Base de Datos Ribosomal II (Cole et al, 2009);

4. ZILLA_500: incluye 500 secuencias del gen rbcL (cada secuencia tiene 1428) de plaguicidas vegetales Guindon and Gascuel (2003).

Ejemplo 1. Estrategia basada en parsimonia para una rápida exploración del espacio búsqueda (tree-space)

Este ejemplo utiliza el algoritmo MOEA/D, y teniendo en cuenta la perspectiva teórica de la fuerte relación entre la parsimonia y la verosimilitud, detallada en la sección 3.8.2.3, minimizar el score de parsimonia es equivalente a maximizar la verosimilitud bajo algunas suposiciones (Steel and Penny, 2000); esta estrategia se basa en movimientos topológicos utilizando la técnica *Parametric Progressive Neighborhood* (PPN) (Goëffon et al, 2008). Con el objetivo de mejorar la verosimilitud, todas las longitudes de rama afectadas por los movimientos topológicos se optimizan mediante el uso de los métodos de optimización numérica Newton-Raphson y Gradiente. Para llevar a cabo este experimento en la Tabla 3.2 se detalla la configuración de los parámetros.

Parámetro Abreviado	**Valor**	**Descripción**
algorithm	MOEAD	Nombre del algoritmo
populationsize	100	Tamaño de la población
seq.format	Phylip(split=spaces, order=interleaved, type=extended)	El formato del alineamiento. Las opciones son: **Order**=*sequential* o *interleaved* y **type**=*extended* o *classic*.
opt.method	hl	Técnica *Parametric Progressive Tree Neighbourhood* (PPN)
opt.ppn.numit	500	Número de iteraciones de la técnica PPN
opt.ppn.maxSprMvs	100	Máximo número de movimientos *buenos* a ser aplicados en la técnica PPN
bl_opt	true	true o false
bl_opt_method	newton	Métodos disponibles Newton-Raphson y Gradient
bl_opt_maxit	1000	Máximo número de iteraciones en la función de optimización
bl_opt_tolerance	0.001	Tolerancia de error.

Tabla 3.2: Valores de los parámetros para configurar MO-Phylogenetics para el Ejemplo 1

Para ilustrar los resultados de esta configuración se han realizado algunos experimentos sobre el conjunto de datos *rbcL_55* (55 secuencias del gen del cloroplasto rbcL de plantas verdes, de un tamaño de con 1314 nucleótidos por secuencia).

La Figura 3.5 muestra una aproximación típica del frente de Pareto lograda después de hacer 20 ejecuciones independientes, donde los resultados biológicos obtenidos de los puntos extremos nos sugieren una mejora en comparación con otros resultados de estado del arte (Cancino and Delbem, 2007b; Santander-Jiménez and Vega-Rodríguez, 2013a, 2014) en términos de un mejor valor de verosimilitud. Con respecto a los valores de parsimonia, los resultados siguen siendo los mismos. Estos árboles filogenéticos, junto con el de la mitad del frente, se ilustran en la Figura 3.6.

Ejemplo 2. Generación de soluciones iniciales para alcanzar una rápida convergencia multiobjetivo

(Stamatakis, 2004) sugiere iniciar la exploración del espacio de árboles ("tree-space") con ár-

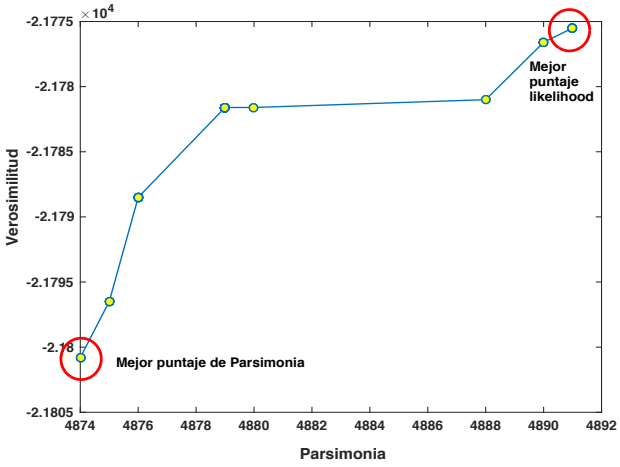


Figura 3.5: Aproximación del frente de Pareto generado por MO-Phylogenetics utilizando la configuración del ejemplo 1 sobre el conjunto de datos rbcL _55. Los puntos extremos representan los mejores árboles filogenéticos inferidos considerando los criterios de parsimonia y verosimilitud.

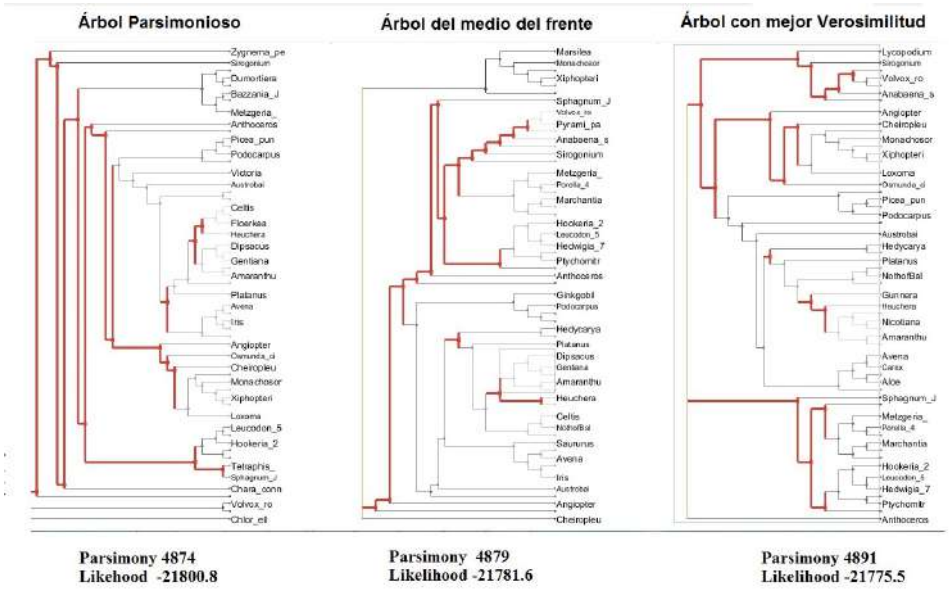


Figura 3.6: Comparación de los árboles de los extremos (mejor parsimonia - izquierda, mejor verosimilitud - derecha) y del árbol del medio (centro) del conjunto de datos *rbcL_55*. Las diferencias entre los árboles se resaltan en color rojo. Los árboles se han obtenido con la herramienta Tree-Juxtaposer.

boles filogenéticos generados por el método *Stepwise-Addition* por dos razones: en primer lugar, además de empezar con un árbol inicial con un buen valor de parsimonia, también podemos esperar obtener que el mismo árbol posea un valor de verosimilitud relativamente bueno; en segundo lugar, porque es un método no determinista que genera diferentes topologías iniciales para cada ejecución

independiente de los algoritmos que se realice (Zwickl, 2006). Un ejemplo de esta configuración se incluye en la Tabla 3.3.

Parámetro Abreviado	**Valor**	**Descripción**
algorithm	SMSEMOA	Nombre del algoritmo
populationsize	100	Tamaño de la población
init.pop	stepwise	Método *Stepwise Addition*
bl_opt_start	true	Optimización de las longitudes de las ramas a las topologís auto-generadas inicialmente
bl_opt_start_method	newton	Métodos disponibles Newton-Raphson y Gradient

Tabla 3.3: Valores de los parámetros para configurar MO-Phylogenetics para el Ejemplo 2.

Hemos escogido el algoritmo SMS-EMOA, el cual es configurado con el parámetro *init.pop* igual a *stepwise*. Con el objetivo de mejorar la verosimilitud de los árboles iniciales, podemos optimizar todas las longitudes de las ramas estableciendo el parámetro *bl_opt_start* en *true*.

Hemos llevado a cabo experimentos utilizando estos parámetros sobre un conjunto de datos reales de nucleótidos del gen ARN ribosomal 16S (16S rRNA) de 19 diferentes especies de bacterias analizadas en (Canchignia et al, 2015). La figura 3.7 muestra una aproximación típica del frente de Pareto.

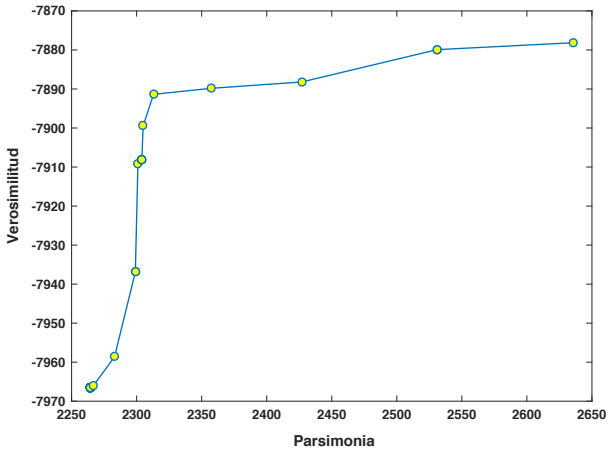


Figura 3.7: Aproximación del frente de Pareto generada por MO-Phylogenetics sobre un conjunto de secuencias de nucleotidos reales (16S_rRNA) usando el método Stepwise Addition para generar los árboles filogenéticos iniciales y el método numércio Newton-Raphson para optimizar las longitudes de las ramas de estas topologías de arranque. El algoritmo usado es SMS-EMOA.

Para comparar las diferencias entre las soluciones generadas, hemos incluido en la figura 3.8 los árboles de los extremos (mejor árbol parsimonioso y árbol con mejor verosimilitud) y el del medio del frente.

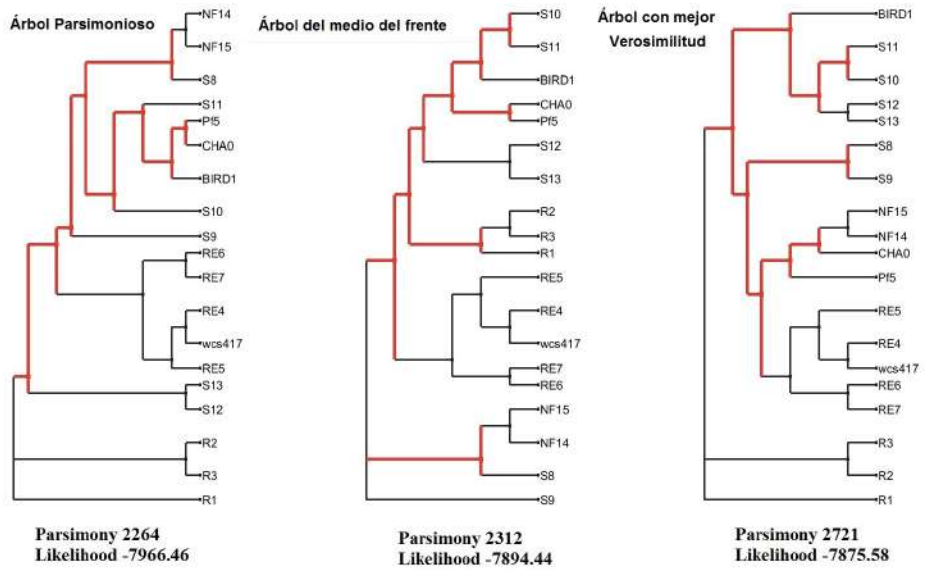


Figura 3.8: Comparación de los árboles de los extremos (mejor parsimonia - izquierda, mejor verosimilitud - derecha) y del árbol de la mitad (centro) del conjunto de datos *16S_rRNA* generados por MO-Phylogenetics en el ejemplo 2. Las diferencias entre los árboles se resaltan en color rojo. Los árboles se han obtenido con la herramienta TreeJuxtaposer.

Ejemplo 3. Uso de la búsqueda local basada en la técnica híbrida parsimonia&verosimilitud para mejorar la exploración del espacio de árboles.

En este ejemplo, describimos cómo configurar el algoritmo NSGA-II usando nuestra búsqueda local que combina dos técnicas altamente optimizadas para la exploración del espacio de árboles ("tree-space"), *PPN* (Goëffon et al, 2008) y *pllRearrangeSearch* (Flouri et al, 2015), con el objetivo de optimizar los objetivos de la máxima parsimonia y la máxima verosimilitud, respectivamente. Esta estrategia combinada permite que los algoritmos evolutivos encuentren regiones no descubiertas durante la exploración del espacio de búsqueda, generando nuevos árboles filogenéticos prometedores. Los parámetros de este ejemplo se incluyen en la Tabla 3.4.

Parámetros Abreviados	Valor	Descripción
algorithm	NSGAII	Nombre del Algoritmo
maxevaluations	6000	Número de evaluaciones del algoritmo evolutivo multiobjetivo
opt.method	h2	Función Híbrida conformada entre *Likelihood-based Rearrange Search* (PLL) *and Parsimony-based technique PNN*
opt.method.perc	0.5	Porcentaje de aplicación entre las técnicas *pllRearrangeSearch* y *PPN*
opt.pll.percnodes	0.3	Porcentaje de nodos a ser usados como nodo raíz en la función *pllRearrangeSearch*
opt.pll.mintranv	1	Define el rango de los movimientos topológicos de la función.

opt.pll.maxtranv	20	
opt.pll.newton3SPRbranch	false	Optimización de las longitudes de las ramas afectadas por los movimientos topológicos.
bl_opt	true	Optimización de todas las longitudes de las ramas de las topologías

Tabla 3.4: Valores de los parámetros para configurar MO-Phylogenetics para el Ejemplo 3.

Hemos aplicado MO-Phylogenetics con esta configuración al conjunto de datos *RDPII_218* el cual contiene 218 secuencias de Prokaryotic RNA de 4182 nucleótidos por secuencia). Después de haber realizado 20 ejecuciones independientes, una aproximación típica del frente de Pareto se muestra en la Figura 3.9. Podemos observar que este frente tiene una buena dispersión de soluciones, generando a un amplio conjunto de hipótesis filogenéticas alternativas que podrían ser útiles desde un punto de vista biológico.

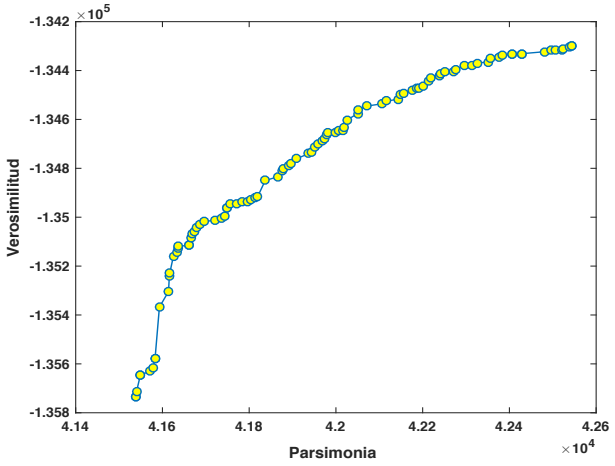


Figura 3.9: Aproximación del frente de Pareto generada por MO-Phylogenetics usando NSGAII con una Búsqueda Local híbrida basada en una técnica combina entre Verosimilitud&Parsimonia para mejorar la exploracíón del espacio de búsqueda sobre el conjunto de secuencias *RDPII_218*.

El árbol con el mejor score de parsimonia y el árbol con el mejor score de verosimilitud son ilustrados en la figura 3.10. Además para resaltar las diferencias de las hipótesis filogenéticas inferidas, se ilustra también el árbol del medio, el cual contiene una mezcla de ambos métodos. Las diferencias entre los tres están resaltadas con rojos.

3.8.4. Análisis de rendimiento frente a herramientas de inferencia filogenética

Además de conocer el funcionamiento de MO-Phylogenetics, con el objetivo de conocer su rendimiento hemos realizado una comparación entre MO-Phylogenetics y otras dos herramientas software de referencia en el área de la máxima verosimilitud: RAxML Stamatakis (2006) e IQ-Tree (Nguyen et al, 2015). Los parámetros del modelo evolutivo GTR + Γ se han establecido en los mismos valores para hacer una comparación equitativa. Algunos de los parámetros de control de

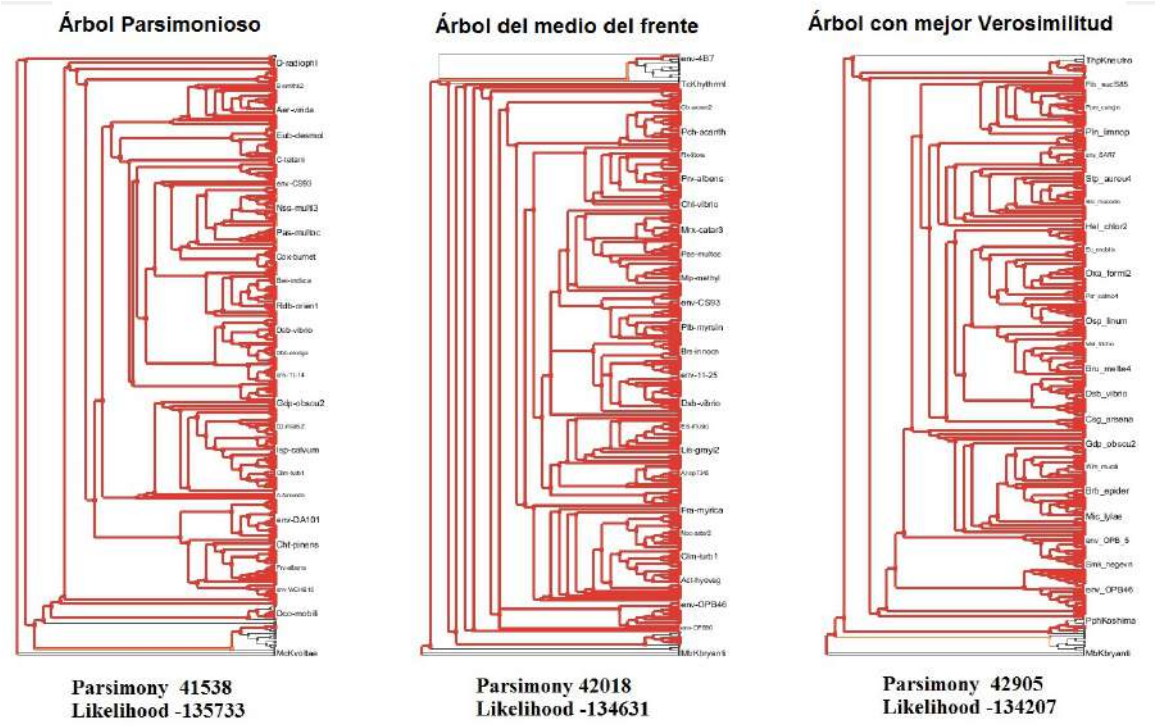


Figura 3.10: Comparación de los árboles de los extremos (mejor parsimonia - izquierda, mejor verosimilitud - derecha) y del árbol de la mitad (centro) del conjunto de datos *RDPII_218* generados por MO-Phylogenetics en el ejemplo 3. Las diferencias entre los árboles se resaltan en color rojo. Los árboles se han obtenido con la herramienta TreeJuxtaposer.

RAxML y IQ-Tree han sido ajustados para mejorar su rendimiento. Los siguientes comandos han sido usados para ejecutar RAxML y IQ-Tree:

```
// RAxML version 8.2.4
$raxmlHPC -s 55.msa -m GTRGAMMA -c 4 -p 12345 -# 20 -n TEST55

// IQ-Tree version 1.3.10
$IQTree -s 55.msa -st DNA -spp 55.model -m GTR+G4{0.36}+FO
```

Los resultados obtenidos se incluyen en la Tabla 3.5. Estos resultados demuestran que MO-Phylogenetics logra el mejor valor de verosimilitud en el problema de rcbL_55 y IQ-Tree produce los mejores resultados en los otros dos problemas. Tenemos que señalar que, aunque hemos utilizado la última versión de IQ-Tree con parámetros ajustados, en MO-Phyogenetics hemos configurado los algoritmos con configuraciones estándar, por lo que hemos considerado realizar un análisis de sensibilidad de la configuración de parámetros para refinar el rendimiento de los algoritmos.

3.8.5. Medición de las diferencias entre los árboles de un frente

Con el objetivo de conocer cuán diferentes o similares son los árboles filogenéticos que componen las aproximaciones del frente de Pareto generados por MO-Phylogenetics, hemos calculado la

Tabla 3.5: Valores de máxima verosimilitud obtenidos por MO-Phylogenetics, RAxML e IQ-Tree en las tres instancias de los ejemplos (los valores en **negritas** indica los mejores resultados).

Dataset	MO-Phylogenetics	RAxML	IQ-Tree
rcbL_55	**-21769.20**	-21824.90	-21769.22
16SrRNA_19	-7875.58	-7877.22	**-7875.17**
RDPII_218	-134248.30	-134162.45	**-134095.13**

distancia de Robinson-Foulds del conjunto de árboles de los frentes usando el software RAxML. El comando que hemos utilizado es:

```
$ raxmlHPC -m GTRGAMMA -z treesPF500 -f r -n 500
```

siendo *treesPF500* el dataset con el conjunto de hipótesis filogenéticas generadas pro MO-Phylogentics. La Tabla 3.6 resume los resultados obtenidos. Los porcentajes de árboles únicos para las instancias rcbL_55 y 16SrRNA_19 están por debajo del 12%, pero el porcentaje de diversidad es mayor (93%) entre todas las soluciones en las aproximaciones del frente Pareto del conjunto de datos RDPII_218.

Tabla 3.6: Distancias de Robinson-Foulds (RF) entre los árboles de las aproximaciones del frente de Pareto generados por MO-Phylogenetics para los conjunto de datos rcbL_55, 16SrRNA_19 y RDPII_218.

Conjunto de Datos	Promedio de las distancias RF normalizada	% de árboles únicos en el frente
rcbL_55	0.059	8%
16SrRNA_19	0.06	12%
RDPII_218	0.38	93%

Con el objetivo de visualizar el espacio de árboles (tree-space) explorado por las soluciones finales de MO-Phylogenetics para los tres ejemplos, hemos ploteado la matriz de distancia Robinson-Foulds usando un Análisis de Coordenadas Principales (Principal Coordinates Analysis - PCoA) (Figura 3.11) para visualizar la exploración.

Los gráficos muestran que las soluciones en los frentes no pertenecen a una sola "isla" (o terraza), sino a un conjunto de "islas" diferentes que están muy separadas. El número de islas es pequeño en el conjunto de datos rcbL_55 y 16SrRNA (3 y 2, respectivamente), mientras que en el caso del conjunto de datos RDPII_218 el número de islas aumenta a 7. Esto sugiere que cuanto más complejo es el problema, mayor es el número de islas.

3.9. Propuesta algorítmica: MORPHY

Como evolución del sistema MO-Phylogenetics se ha iniciado un nuevo proyecto denominado **MORPHY** (*Multi-Objective softwaRe for PHYlogenetic inference*). Aunque es un proyecto que está en sus fases iniciales, en esta sección describiremos sus objetivos y los resultados preliminares que se han obtenido hasta el momento.

La motivación del proyecto MORPHY es, por un lado, incorporar funcionalidad para optimizar problemas de inferencia que estén descritos en términos de proteínas, además de ADN. Esta funcionalidad, por lo que sabemos, no existe en ninguna propuesta de optimización multiobjetivo con metaheurísticas aplicada a inferencia filogenética. Por otro lado, se busca diseñar un nuevo

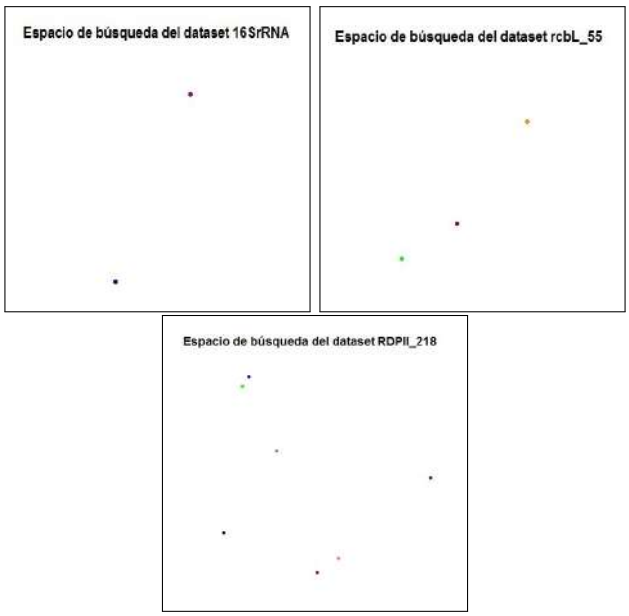


Figura 3.11: Visualización del espacio de árboles (tree-space) según las distancias Robinson-Foulds en los tres conjunto de datos.

algoritmo diseñado específicamente para inferencia filogenética multiobjetivo, que también ha denominado en principio igual que el proyecto. La propuesta inicial de dicho algoritmo se describe a continuación.

3.9.1. MORPHY: descripción del algoritmo

MORPHY es un algoritmo evolutivo basado en la principal referencia algorítmica de optimización multiobjetivo NSGA-II (Deb et al, 2002b). La representación interna de las soluciones está basada en la plantilla *TreeTemplate* que proveé BIO++ para el manejo de árboles filogenéticos.

Las soluciones de inicio del algoritmo son árboles parsimoniosos generados por el método Stepwise-Addtion (Felsenstein, 2004) con una optimización exhaustiva de las longitudes de sus ramas mediante el método de optimización numérica Newton-Raphson (Press et al, 1992), logrando así obtener una amplia muestra de árboles iniciales.

Durante el ciclo de vida del algoritmo, con el objetivo de encontrar nuevos y prometedores árboles filogenéticos, MORPHY genera una población de soluciones alternativas de la población inicial, llamada *offSpring* o descendencia, aplicando a cada una de las soluciones, una perturbación aleatoria mediante la ejecución del operador de mutación NNI (Felsenstein, 2004) (véase 3.8.2.2) y la estrategia híbrida de optimización de MO-Phylogenetics para mejorar la calidad de las nuevas soluciones (véase 3.8.2.3). Además, con el objetivo de mejorar la verosimilitud de los árboles, cada intervalo de N evaluaciones del algoritmo, se optimizan las longitudes de las ramas y los parámetros del modelo evolutivo de cada solución. Cada solución del algoritmo provee sus propios parámetros por lo que son ajustados de acuerdo a su topología.

Con el objetivo de evitar óptimos locales, si el número de nuevas soluciones de la población resultante es menor que un valor umbral mínimo, a las soluciones dominadas se les aplica de forma exhaustiva la técnica de optimización híbrida de MO-Phylogenetics (ver 3.8.2.3) para modificar su topología y obtener nuevas y mejores soluciones de ellas. Esto significa que los árboles filogenéticos

con peores valores de parsimonia y/o verosimilitud que otros tienen que ser reemplazados/o mejorados para generar una nueva topología que permita el acceso a zonas inexploradas del espacio de búsqueda.

3.9.2. Funcionalidades del algoritmo

MORPHY tiene las siguientes funcionalides:

- Soporte a secuencias de ADN (nucleótidos) y proteínas (aminoácidos).
- Provee una implementación completa del modelo evolutivo GTR (General Time Reversible) (véase 3.6) para secuencias de ADN y los modelos evolutivos más comunes para las secuencias de proteínas, todos proporcionados por la librería PLL.
- Un análisis particionado del conjunto de secuencias, asignando diferentes modelos de sustitución a cada partición del alineamiento, similar a la funcionalidad RAxML.
- Provee los modelos Rate heterogeneity across sites: modelo Γ discreto con 4 categorías y CAT con N categorías.
- Técnicas de optimización númericas de los parámetros del modelo de sustitución definido.
- Creación del árbol consenso del conjunto final de soluciones no dominadas (árboles filogenéticos) generado por el algoritmo, definido a según el número de ocurrencias de las biparticiones.
- Ejecución rápida y sencilla a través de línea de comandos, hay parámetros que han sido establecidos con valores por defecto que no son necesarios definirlos para la ejecución del programa.
- Es un proyecto de código abierto y de libre distribución, que está alojado en GitHub: https://github.com/KhaosResearch/MORPHY.

Al igual que MO-Phylogenetics 3.8, MORPHY puede ser fácilmente configurado mediante la definición de sus parámetros en un archivo de texto plano, el cual es enviado como único parámetro de entrada al algoritmo. En la tabla E.1 del apéndice E se detallan todos los parámetros de MORPHY.

Para ejecutar MORPHY hay que usar la siguiente línea de comando:

```
$ MORPHY param = parametersfile
```

La configuración inicial de los parámetros del modelo de sustitución puede ser obtenido por otros software como jMoetlTest (Darriba et al, 2012) o IQ-Tree (Nguyen et al, 2015), y especificado en el archivo de configuración, or directly best-fit estimated by MORPHY.

3.9.3. Ejemplos del rendimiento filogenético con dataset de proteínas (M2926)

Con el objetivo de ilustrar un ejemplo del desempeño de nuestro algoritmo, hemos realizado algunos experimentos sobre el conjunto de proteínas M2926 extraídos del repositorio TreeBase, utilizados igualmente por los autores de IQ-TREE para ilustrar su rendimiento (Nguyen et al, 2015).

La Figura 3.12 ilustra una aproximación del frente de Pareto generada a partir del conjunto de soluciones (árboles filogenéticos) inferidos por MORPHY, en el que los puntos de los extremos representan las mejores valoraciones de parsimonia (izquierda) y verosimilitud (derecha), el cual precisamente representa una mejora al score de verosimilitud promedio publicado por IQ-TREE (Nguyen et al, 2015).

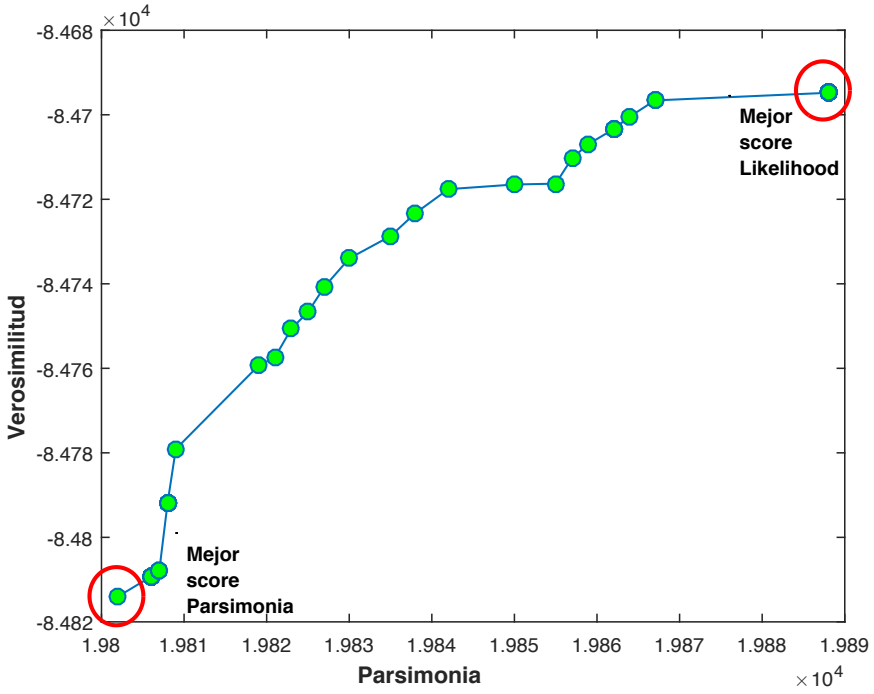


Figura 3.12: Aproximación del frente de Pareto para el conjunto de proteínas M2926 generado por MORPHY, donde cada solución es considerada como una solución no dominada. Los puntos de los extremos representan a los mejores árboles filogenéticos generados bajo la máxima parsimonia (izquierda) y la máxima verosimilitud(derecha).

3.9.4. Análisis comparativo multiobjetivo frente a otras metaheurísticas

Con el objetivo de conocer el rendimiento multiobjetivo de nuestro algoritmo MORPHY, hemos realizado un análisis comparativo preliminar con otras metaheurísticas de optimización implementadas en el framework MO-Phylogenetics: NSGA-II (Deb et al, 2002b) y dos técnicas más modernas del estado del arte: Multiobjective Evolutionary Algorithm Based on Decomposition (MOEA/D) (Zhang and Li, 2007) y Multiobjective selection based on dominated hypervolume (SMS-EMOA) (Beume et al, 2007). Los problemas de proteínas a resolver han sido obtenidos desde el repositorio público TreeBASE (Sanderson et al, 1994) los cuales son detallados en la Tabla 3.7.

Tabla 3.7: Lista de problemas de proteínas extraídas de TreeBASE

Matriz	Taxas	Length
M510	57	430
M3807	82	591
M3810	55	271
M9973	60	327

Para hacer una comparativa exacta entre ellos, todos los algoritmos fueron configurados con los mismos valores. Cada ejecución independiente de cada uno de los algoritmos dura hasta 10000 evaluaciones computadas, el tamaño de la población es de 100 individuos, la probabilidad de Cruce es del 80 % y la de mutación del 20 %. El modelo de sustitución y los parámetros ajustados a cada

problema fueron estimados por el software w-IQTRee (Trifinopoulos et al, 2016) y definitivos de forma similar para todas las ejecuciones de todos los algoritmos. Se llevaron a cabo 10 ejecuciones independientes de cada algoritmo resolviendo cada uno de los problemas de prueba.

Hemos definido como métricas de evaluación los siguientes indicadores de calidad multiobjetivo: **Épsilon ($\mathbf{I}_{E^+}$)**, que mide la convergencia al determinar la distancia mínima (en cualquier objetivo) que habría que desplazar cada solución para ser no dominada con respecto a otro frente (cuanto más pequeño mejor); el indicador Dispersión (**Spread ($\mathbf{I}_\Delta$)**), que mide la distribución de las soluciones y la distancia con los extremos del verdadero frente de Pareto (su valor ideal es cero) y el indicador de calidad IGD^+, que tiene en cuenta tanto la convergencia como la distribución de los frentes. La mediana y los rangos intercuartílicos IQRs de los resultados obtenidos de estos indicadores de calidad se ilustran en las tablas 3.8, 3.9 y 3.10 respectivamente.

En las tablas de resultados, los mejores de cada experimento son marcados con un fondo gris más oscuro para ser resaltados y los segundos mejores resultados son marcados con tono de gris más claro. Como no se conoce el frente de pareto óptimo de los problemas, se ha generado un frente de Pareto de referencia para cada problema, uniendo todos los frentes de Pareto aproximados generados por todas las ejecuciones independientes de los tres algoritmos, descartando las soluciones dominadas.

Para comprobar si las diferencias obtenidas entre los algoritmos son estadísticamente significativas, se ha aplicado el test de Wilcoxon con un nivel de confianza del 95% para cada par de algoritmos. Para ilustrar estos resultados, en las tablas se usa la siguiente simbología: el carácter '-' indica que no hay diferencias significativas entre los algoritmos de la fila y columna, el símbolo ▲ significa que el algoritmo de la fila ha producido mejores resultados que el algoritmo de la columna con significancia estadística, y el símbolo ▽ se utiliza cuando el algoritmo en la columna es estadísticamente mejor que el de la fila en el problema considerado. Los resultados se detallan en las tablas 3.11, 3.12 y 3.13 para los indicadores Épsilon (I_{E^+}), Spread (I_Δ) e IGD · respectivamente.

Tabla 3.8: Mediana y rango intercuartílico del indicador EPSILON.

	NSGAII	MORPHY	MOEAD	SMSEMOA
M510	$1{,}00e+00_{7{,}8e-01}$	$1{,}00e+00_{0{,}0e+00}$	$2{,}78e+00_{4{,}2e+00}$	$1{,}00e+00_{1{,}0e+00}$
M3807	$2{,}58e+00_{1{,}1e+01}$	$9{,}92e+00_{9{,}1e+00}$	$1{,}69e+01_{1{,}3e+01}$	$1{,}08e+01_{1{,}1e+01}$
M3810	$1{,}25e-01_{6{,}3e-02}$	$1{,}25e-01_{7{,}8e-02}$	$1{,}56e-01_{9{,}4e-02}$	$1{,}56e-01_{6{,}3e-02}$
M9973	$1{,}10e+00_{1{,}3e+00}$	$2{,}27e+00_{0{,}0e+00}$	$2{,}28e+00_{1{,}6e-01}$	$2{,}27e+00_{7{,}5e-01}$

Tabla 3.9: Mediana y rango intercuartílico del indicador SPREAD.

	NSGAII	MORPHY	MOEAD	SMSEMOA
M510	$1{,}16e+00_{3{,}1e-01}$	$1{,}36e+00_{6{,}2e-02}$	$1{,}33e+00_{9{,}1e-02}$	$1{,}17e+00_{2{,}3e-01}$
M3807	$1{,}32e+00_{6{,}0e-02}$	$1{,}30e+00_{2{,}1e-01}$	$1{,}21e+00_{1{,}1e-01}$	$1{,}25e+00_{2{,}2e-01}$
M3810	$1{,}32e+00_{9{,}6e-02}$	$1{,}36e+00_{6{,}3e-02}$	$1{,}29e+00_{6{,}5e-02}$	$1{,}21e+00_{2{,}0e-01}$
M9973	$1{,}41e+00_{7{,}4e-01}$	$9{,}50e-01_{2{,}4e-02}$	$1{,}17e+00_{7{,}7e-02}$	$1{,}22e+00_{1{,}4e-01}$

Tabla 3.10: Mediana y rango intercuartílico del indicador IGD · .

	NSGAII	MORPHY	MOEAD	SMSEMOA
M510	$8{,}18e-01_{8{,}4e-01}$	$8{,}18e-01_{8{,}0e-02}$	$2{,}33e+00_{3{,}9e+00}$	$7{,}95e-01_{1{,}2e+00}$
M3807	$2{,}24e+00_{1{,}1e+01}$	$9{,}49e+00_{9{,}1e+00}$	$1{,}65e+01_{1{,}3e+01}$	$1{,}03e+01_{1{,}1e+01}$
M3810	$3{,}35e-02_{2{,}0e-02}$	$3{,}86e-02_{2{,}7e-02}$	$5{,}53e-02_{5{,}0e-02}$	$5{,}12e-02_{2{,}9e-02}$
M9973	$7{,}84e-01_{1{,}3e+00}$	$2{,}22e+00_{0{,}0e+00}$	$2{,}22e+00_{3{,}0e-01}$	$2{,}13e+00_{9{,}6e-01}$

En la Tabla 3.14 se muestra una comparativa analizando la Máxima Verosimilitud de los árboles inferidos entre MORPHY y otros software de referencia en el objetivo de la Máxima Verosimilitud como IQ-Tree, RAxML y PhyML.

Tabla 3.11: Resultado del test de wilcoxon rank-sum del indicador EPSILON I_{E^+} M510 - M3807 - M3810 - M9973.

	MORPHY				MOEAD				SMSEMOA			
NSGAII	–	▲	–	▲	▲	▲	▲	▲	–	–	▲	▲
MORPHY					▲	–	–	–	–	–	–	–
MOEAD									–	–	–	–

Tabla 3.12: Resultado del test de wilcoxon rank-sum del indicador SPREAD I_{Δ} M510 - M3807 - M3810 - M9973.

	MORPHY				MOEAD				SMSEMOA			
NSGAII	▲	–	–	▽	▲	▽	–	▽	–	–	–	▽
MORPHY					▽	▽	▽	▲	▽	–	▽	▲
MOEAD									▽	–	–	–

Tabla 3.13: Resultado del test de wilcoxon rank-sum del indicador IGD $\cdot$. M510 - M3807 - M3810 - M9973.

	MORPHY				MOEAD				SMSEMOA			
NSGAII	–	▲	–	▲	▲	▲	▲	▲	–	–	▲	▲
MORPHY					▲	–	▲	–	–	–	–	–
MOEAD									▽	–	–	–

Tabla 3.14: Resultados de los scores de la Máxima Verosimilitud entre MORPHY con otros softwares de referencia ML.

Dataset	**IQ-TREE**	**RAxML**	**PhyML**	**MORPHY**
M510	-8164.14	-8,165.75	-8,175.26	**-8077.26**
M3807	-38,706.56	-38,709.34	-38,722.00	**-38,467.10**
M3810	**-11,061.54**	-11,061.76	-11,061.55	-11,074.10
M9973	-13,077.17	-13,077.98	-13,077.18	**-13,042.50**

Los resultados muestran que NSGA-II es el algoritmo que mejor rendimiento ofrece si tenemos en cuenta los indicadores I_{E^+} e IGD $\cdot$, ya que obtiene los mejores valores en tres de los cuatro problemas seleccionados. Nuestra propuesta queda por detrás de este algoritmo y por delante de las técnicas MOEA/D y SMS-EMOA. Sin embargo, estos algoritmos son los que mejor rinden según el indicador de calidad que mide la diversidad.

Estos primeros resultados indican que la propuesta inicial de MORPHY obtiene resultados competitivos pero no como para mejorar a los de su algoritmo base, NSGA-II, por lo que hay que continuar investigando en esta línea. También hay que reseñar que sólo se han usado cuatro problemas en este primer estudio, lo que tampoco permite obtener resultados concluyentes.

Capítulo 4

Alineamiento Múltiple de Secuencias

4.1. Introducción

En biología una secuencia se define como una molécula continua unidimensional compuesta de monómeros covalentemente unidos dentro de un biopolímero. Estas secuencias son relacionadas con el ADN, el ARN o proteínas de acuerdo al tipo de molécula subyacente. Las moléculas de ADN son construidas por cadenas de nucleótidos que están representados por un alfabeto de cuatro letras {A, G, C, T} ({A, G, C, U} para moléculas de ARN). Asimismo, las proteínas se construyen a partir de secuencias de 20 aminoácidos diferentes, cada uno representado por una letra {A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y} cada elemento de estas secuencias es normalmente denominada como **residuo**. Por lo tanto, los alineamientos de secuencias se definen como comparaciones formales de estas cadenas moleculares. Además cualquier alineamiento no trivial contiene espacios vacíos denominados **gaps** (sitios para los que no se ha asignado ningún estado), los cuales son representados normalmente con los siguientes caracteres '-' o '*'. Los gaps pueden producirse por algunas razones biológicas (inserciones o eliminaciones de residuos), por razones técnicas (los datos no se leen correctamente durante la secuenciación), o porque falta información (un gen específico para un taxón específico nunca ha sido secuenciado). Por todos estos aspectos los caracteres que faltan y que son indeterminados son tratados como gaps.

El objetivo principal de alinear las secuencias es identificar el mayor número posible de regiones conservadas a través del conjunto de secuencias consideradas. Mientras mayor sean las regiones conservadas identificadas mayor la probabilidad de identificar la existencia de un ancestral compartido entre las especies (en biología, la semejanza obtenida de antepasados comunes se denomina homología) (Doolittle, 1981). Esta comparativa representa un papel muy importante en el descubrimiento de relaciones evolutivas entre las secuencias de ADN, ARN y proteínas (Fitch, 1966). Para realizar estas comparaciones las secuencias se representan en alineamientos por sus correspondientes alfabetos. Así, los residuos que coincidan exactamente en el alineamiento se los denominan "identities" mientras que diferentes residuos que comparten características físico-químicas se los denominan "similarities". Adicionalmente, aquellos residuos alineados que son totalmente diferentes se consideran "mismatches". Gaps observados en una columna del alineamiento se denominan "indels" (inserciones o eliminaciones en la secuencia). Las columnas que incluyen "mismatches" pueden interpretarse como sitios de la secuencia donde una o varias especies han sido objeto de mutaciones puntuales. Sobre secuencias de ADN, las mutaciones puntuales se denominan SNPs aleatorios (Single nucleotide Polymorphism - Polimorfismo de un sólo nucleótido), que habitualmente tienen lugar durante la replicación del ADN.

Si el codón (triplete de nucleótidos) resultante por la mutación se traduce en el mismo ami-

noácido, se llama una sustitución silenciosa o sinónima (*silent* o *synonymous*). Por otra parte, si la mutación induce un cambio en un aminoácido diferente, se denomina sustitución no-sinónima (*nonsynonymous*) y, dependiendo de cuán diferente (en términos de las propiedades bioquímicas) sea el nuevo aminoácido, la función de la proteína puede ser afectada.

El alineamiento de secuencias está basado en la noción de que regiones similares comparten un historia evolutiva común. La interpretación es que las regiones con alta similitud no muestran cambios porque corresponden a una importante, conserva evolutiva, propiedad o función estructural.

Los alineamientos se pueden clasificar según la manera en que son alineadas las secuencias. En este caso, se definen dos tipos de alineamientos: *alíncamientos globales* y *alíncamientos locales*. En los alineamientos globales se intenta alinear completamente todo el contenido de la secuencias, y en los alineamientos locales, las secuencias son divididas en bloques más pequeños para encontrar alineamientos óptimos por cada una estas regiones. Finalmente, los alineamientos también pueden dividirse según el número de secuencias a alinear: Alineamientos de pares de secuencias (Pairwise Sequence Alignment) cuando se alinean dos secuencias y en Alineamiento Múltiple de Secuencias (Multiple Sequence Alignment - MSA) cuando se consideran mas de dos secuencias.

4.2. Definición del problema MSA

Esta sección define el dominio del problema del Alineamiento Múltiple de Secuencia en términos formales.

Sea Σ un alfabeto finito, por ejemplo un conjunto finito de caracteres, y $\Sigma \neq 0$, y un conjunto de k secuencias biológicas $S = (s_1, s_2, ..., s_k)$ de longitudes finitas y variables denotadas como l_1 a l_k y compuestas de caracteres $s_i = s_{i1}s_{i2}, ..., s_{il_i}(1 \leq i \leq k)$, S' es una matriz que representa el alineamiento óptimo de S, la cual está definida formalmente por la siguiente ecuación 4.1:

$$S' = (s'_{ij}),\ con\ 1 \leq i \leq k, 1 \leq j \leq l, max(l_i) \leq l \leq \sum_{i=1}^{k} l_i \tag{4.1}$$

Y cumple con:

1. $s'_{ij} \in \Sigma \cup \{-\}$, donde "-" denota el carácter de espacios o "gaps";
2. cada fila $s'_i = s'_{i1}s'_{i2}, ..., s'_{il}(1 \leq i \leq k)$ de S' es exactamente igual a la secuencia correspondiente s_i si eliminanos todos los gaps;
3. La longitud de todas las k secuencias es exactamente la misma;
4. S' no tiene columnas conformada solo por gaps.

En biología molecular, para las secuencias de ADN, el alfabeto Σ consiste de cuatro nuclétidos representados por los caracteres {A, T, G, C} y para las secuencias de proteinas, el alfabeto Σ consiste de 20 amino ácidos representados por los caracteres {A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y};

Un ejemplo de alineamiento se muestra a continuación, en el se representan cuatro secuencias con seis columnas alineadas las cuales están marcadas con un asterisco (*).

```
APPSVFAEVPJQKTM-AQPVMKLJ
AKRS-V-E-PJFKTMR-IKMK---
-LISKRA-YPJ-KTM-I---MALP
-SASTIGVEPJCK-M-RA-P--KL
*        ** ***
```

4.2.1. Complejidad del problema MSA

El problema MSA es considerado como un problema de Complejidad NP-Completo (NP-Hard), ya que la exploración del espacio de búsqueda se incrementa exponencialmente; según el número de secuencias a alinear k y a su longitud máxima L, definida como $O(k2^k L^k)$ (Waterman et al, 1976). Para tenerlo un poco más claro, en un grupo de solo 5 secuencias con un máximo de 10 residuos (amino-ácidos o nucleótidos) existen 1038 posibles combinaciones de alineamientos que se pueden generar. Inicialmente el alineamiento de un par de secuencias se realizaba mediante el uso de técnicas de Programación Dinámica (Needleman and Wunsch, 1970). Aunque el uso de estas estrategias garantizan alineamientos matemáticamente óptimos, no pueden ser aplicadas cuando se consideran más de dos secuencias en el proceso, debido a la complejidad antes mencionada. Por estas razones, cada vez más, se considera importante y necesario el uso de metaheurísticas de optimización en la resolución del problema.

4.2.2. Funciones objetivos

Una función objetivo mide la calidad del alineamiento y refleja cuan cerca está dicho del alineamiento óptimo biológico. En esta sección, definimos algunas de las funciones objetivo que fueron consideradas en esta investigación, todas ellas están destinadas a ser maximizadas. Para la formulación de estas funciones, hemos considerado al alineamiento a evaluar como S, con un conjunto de k secuencias alineadas representadas como $S = s_1, s_2, ..., s_k$ todas ellas con la misma longitud L.

4.2.2.1. Suma de pares (sum-of-pairs SOP)

La suma de pares (SOP) de un alineamiento, presentada en la ecuación 4.2, se calcula sumando todos los puntajes de las comparaciones de pares entre cada residuo en cada columna del alineamiento.

$$SOP(S) = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \sum_{c=1}^{L} ScoringMatrix(s_{ic}, s_{jc}) \tag{4.2}$$

donde $ScoringMatrix$ representa la matriz que determina el costo de sustituir un residuo por otro. Esta matriz incluye también el valor de penalización de un gap que determina el costo de alinear un residuo con un gap. Esta penalización es sólo cuando se alinean un residuo con gap o viceversa, no cuando se alinean dos o más gaps.

4.2.2.2. Suma de pares ponderada con afinidad entre gaps (weighted sum of pairs with affine gaps wSOP)

La suma de pares ponderada con afinidad entre gaps (wSOP) se calcula restando el puntaje de suma de pares (comparaciones entre pares de cada uno de los caracteres amino-ácidos o nucleótidos) de cada una de las columnas del alineamiento menos el puntaje de penalización a los gaps afines de cada una de las secuencias. La wSOP está representada por la ecuación 4.3:

$$wSOP(S) = \sum_{l=1}^{L} SP(l) - \sum_{i=1}^{k} AGP(s_i) \tag{4.3}$$

donde $SP(l)$ representa el puntaje de la suma de pares de la columna l el cuál está denifido como (ecuación 4.4):

$$SP(l) = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} W_{i,j}\ x\ \delta(s_{i,l}, s_{j,l}) \tag{4.4}$$

En la ecuación 4.4, δ representa la matriz de sustitución usada (como pueden ser *Pointed Accepted Mutation*, - PAM (Dayhoff et al, 1978) o *Block Substitution Matrix*, - BLOSUM (Henikoff and Henikoff, 1992)), la cual proporciona los costos de alineamientos de pares para cada uno de los aminoácidos y el valor de penalidad que se tiene al alinear un caracter con un gap. $W_{i,j}$ representa la ponderación (pesos) entre las sequencias s'_i y s'_j, definida en la siguiente ecuación 4.5:

$$W_{i,j} = 1 - \frac{LD(s'_i, s'_j)}{max(|s'_i|, |s'_j|)} \tag{4.5}$$

donde LD representa la distancia de Levenshtein entre dos secuencias no alineadas (s'_i y s'_j) (el mínimo número de inserciones, eliminaciones o sustituciones de caracteres requeridas para convertir una secuencia en otra).

Finalmente, en la Ecuación 4.3, $AGP(s'_i)$ representa la penalización por gaps afines de la secuencia s_i, la cual está definida en la siguiente ecuación 4.6:

$$AGP(s_i) = (g_{open}\ x\ \#gaps) + (g_{extend}\ x\ \#spaces) \tag{4.6}$$

en la que g_{open} es el peso por empezar con un gap y g_{extend} es el peso por extender el gap con uno o mas espacios.

4.2.2.3. Bali-score (SP y TC)

Baliscore es un software proporcionado por el benchmark BAliBASE v3.0 (Thompson et al, 1999a, 2005), que incluye dos funciones de calidad la Suma de Pares (SP) y Columna Total (TC), las cuales estiman la precisión de calidad de los alineamientos de entrada en comparación a alineamientos de referencias generados por el benchmark. Por una parte, la métrica SP se calcula como la razón de la suma de las puntuaciones p para todos los pares de residuos en cada columna del alineamiento de entrada por la suma de las puntuaciones en el alineamiento de referencia; $p = 1$ si el par de residuos comparados se ajusta de forma idéntica al alineamiento de referencia, de lo contrario $p = 0$. Así, el puntaje SP aumenta con el número de secuencias alineadas correctamente. Por otra parte, la métrica TC se calcula considerando la relación de la suma de las puntuaciones c por el número de columnas en el alineamiento de entrada, siendo $c = 1$ si todos los residuos en la columna se alinean de forma idéntica al alineamiento de referencia, de lo contrario $c = 0$.

4.2.2.4. Single structure induced evaluation (STRIKE)

STRIKE (Kemena et al, 2011) representa una nueva métrica para evaluar la calidad de los alineamientos basada en información estructural, de al menos, una de las secuencias del alineamiento. La información estructural de las secuencias de proteínas es comúnmente obtenida desde el sitio web del Protein Data Bank (PDB) (Berman et al, 2000).

STRIKE calcula los contactos de la secuencia conteniendo información estructural mediante las distancias entre sus aminoácidos. Específicamente, se dice que dos átomos están en contacto intramolecular cuando una solvente molécula no puede ser insertada entre sus superficies moleculares (Connolly, 1983). La distancia entre dos aminoácidos para determinar si están en contacto se calcula a partir de la posición espacial de los átomos en los aminoácidos información que viene proporcionada en la estructura del archivo PDB. Con el fin de evitar contactos producidos por la estructura secundaria, STRIKE sólo considera aquellos contactos que involucran aminoácidos

separados por al menos cinco aminoácidos en la secuencia. Después de estimar los contactos de una secuencia, de acuerdo con su estructura terciaria, los pares de aminoácidos alineados en las mismas posiciones de las demás secuencias, son también retribuidos como contactos. Tales pares de aminoácidos son puntuados de acuerdo a una nueva matriz de puntuación denominada STRIKE Matrix, provista por los mismos autores

Esta matriz STRIKE está basada en información estructural según los contactos encontrados en un conjuntos de alineamientos extraídos en varias base de datos. Específicamente se genera calculando el radio entre la frecuencias de cada posible contacto y de su expansión, dado la frecuencia de fondo de cada uno de los aminoácidos. Dado cualquier par de aminoácidos i y j, el puntaje de sus contactos es estimada según la siguiente Ecuación 4.7:

$$M_{ij} = 10 \times \ln(\frac{f_{ij}}{f_i f_j}) \tag{4.7}$$

donde f_{ij} es la frecuencia de los contactos que implican los aminoácidos i y j a través de todos los contactos $residue - residue$ observados, f_i y f_j son las frecuencias de aminoácidos únicas en el conjunto de datos considerado. Entonces, dado una secuencia s con su información estructural, la puntuación STRIKE del alineamiento calculada según la Ecuación 4.8:

$$\sum_{i=1;i\neq s}^{k} \sum_{j,c=1}^{L} M(x_{ij}, x_{ic}) \times EsContacto(x_{ij}, x_{ic}) \tag{4.8}$$

en la que x_{ij} y x_{ic} representan a cada par de aminoácidos en a i-ésima secuencia a excepción de la secuencia s. La función $EsContacto$ es definida según la Ecuación 4.9

$$EsContacto(x_{ij}, x_{ic}) = \begin{cases} 1 & Si\ x_{ij}\ y\ x_{ic}\ son\ contactos \\ 0 & caso\ contrario \end{cases} \tag{4.9}$$

En el caso de existir varias estructuras proporcionadas, el puntaje STRIKE es calculado de forma separada para cada estructura y finalmente promediado.

Esta métrica de evaluación permite identificar de mejor manera la exactitud en los alineamientos mejor que otras puntuaciones clásicas como BLOSUM62 (Henikoff and Henikoff, 1992) and PAM250 (Dayhoff et al, 1978). Además, supera claramente a las otras métricas clásicas cuando las secuencias son evolutivamente más distantes (Kemena et al, 2011). STRIKE también muestra un fuerte efecto de correlación no-paramétrico con los valores BAliscore (subsección 4.2.2.3). Es decir, en una comparativa entre dos diferentes alineamientos, tanto BAliscore como STRIKE, generalmente identifican al mismo alineamiento como el mejor (alrededor del 79 % de los casos) (Kemena et al, 2011)].

4.2.2.5. El porcentaje de columnas totalmente alineadas

El porcentaje de columnas totalmente alineadas (TC) se refiere al número de columnas que están compuestas totalmente del mismo caracter en cada una de sus filas (amino ácidos o nucleótidos). Esta función objetivo necesita ser maximizada para asegurar la mayor cantidad de regiones conservadas dentro del alineamiento. TC puede ser definida como (Ecuacióo 4.10):

$$TC(S) = 100 \sum_{l=1}^{L} \frac{ColumnaAlineada(S_l)}{L} \tag{4.10}$$

donde S_l representa la l-ésima columna del alineamiento S, tal que $S_l = s_{il} \forall i = 1, .., k$, y la función $ColumnaAlineada(S_l)$ está definida como (Ecuación 4.11):

$$ColumnaAlineada(S_l) = \begin{cases} 1 & Si\ s_{il} = s_{1l}\ \forall i = 2, ..., k \\ 0 & caso\ contrario \end{cases} \tag{4.11}$$

4.2.2.6. Porcentaje de no-gaps

El porcentaej de no-gaps mide el número de residuos con respecto al número de gaps dentro del alineamiento.está definido en la Ecuación4.12:

$$NonGaps(S) = 100 \sum_{i=1}^{k} \sum_{j=1}^{L} \frac{EsNonGap(s_{ij})}{k * L} \tag{4.12}$$

donde s_{ij} representa el símbolo en la j-ésima posición de la i-ésima secuencia en el alineamiento S. La función $EsNonGap$ para un determinado residuo del alineamiento está definido en la siguiente Ecuación 4.13:

$$EsNonGap(residuo) = \begin{cases} 1 & si\ residuo = \text{“}-\text{”}(gap) \\ 0 & caso\ contrario \end{cases} \tag{4.13}$$

4.3. Estado del arte

En esta sección abordaremos las principales técnicas metaheurísticas monoobjetivo y multiobjetivo aplicadas a resolver el problema del alineamiento múltiple de secuencias. Debido a la complejidad del problema MSA se han propuesto diversos métodos heurísticos y metaheurísticos en la literatura, y podemos clasificarlos en cuatro grupos principales: métodos progresivos, métodos basados en consistencia, algoritmos evolutivos mono y multiobjetivo, y métodos basados en información estructural.

4.3.1. Métodos progresivos

Los métodos progresivos representan no sólo la primera metodología a ser aplicada al Alineamiento Múltiple de Secuencias sino también la más usada por los softwares MSA (Hogeweg and Hesper, 1984). Su funcionamiento básicamente comienza calculando una matriz de distancia para cada uno de los pares de secuencias, cuya fórmula se está representada en la Ecuación 4.14.

$$D_{pair}(s_1, s_2) = 1 - \frac{M_{s_1,s_2}}{min(l_1, l_2)} \tag{4.14}$$

En esta ecuación, M_{s_1,s_2} indica el número de coincidencias entre las secuencias s_1 y s_2 mientras que l_1 y l_2 definen la longitud de ambas secuencias. A continuación se construye un árbol guía utilizando cualquier algoritmo de agrupación jerárquica, para combinar los alineamientos por pares de secuencias hasta obtener el alineamiento múltiple final (Blackburne and Whelan, 2012b).

Su amplio uso radica a su flexibilidad para complementarse en cualquier nueva técnica MSA. Luego, cada herramienta permite configurar el mecanismo de cómo generar el árbol guía o definir las matrices de puntuación a utilizar. Además, cabe indicar que estos métodos progresivos alcanzan una alta calidad cuando las secuencias están altamente relacionadas. Sin embargo, están propensos a generar alineamientos imprecisos o incorrectos debido a errores producidos en los alineamientos pairwise iniciales con secuencias no tan relacionadas o errores en los datos. Además errores en la generación de un árbol guía pueden causar soluciones no óptimas.

Con el fin de minimizar estos problemas, las estrategias actuales suelen funcionar en dos ciclos: (i) los alineamientos de pares de secuencias son generados empleando el procedimiento estándar; y (ii) el alineamiento final es optimizado gracias a la reconstrucción de un árbol guía nuevo.

Entre las principales técnicas que emplean los métodos progresivos está el software Clustal W (Thompson et al, 1994), el cual es el programa de alineamiento de secuencias múltiples más popular. Utiliza un método de alineamiento progresivo global. Los pasos de alineamiento empiezan por alinear por pares de todas las secuencias usando la programación dinámica o un método k-tuple rápido aproximado y a continuación los *scores* de los alineamientos de pares de secuencias se utilizan para construir una matriz de distancia de las distancias genéticas. Inicialmente se obtiene el número de posiciones coincidentes dividido por el número total de residuos sin gaps; estos scores se dividen por 100 y se restan de 1,0 para obtener las distancias reales. Estas distancias se usan para construir el árbol guía usando el método neighbor-joining (Saitou and Nei, 1987b). Por último, se utiliza programación dinámica para alinear las secuencias de las más estrechamente relacionadas con las menos estrechamente relacionadas guiadas por las distancias desde el árbol.

Además de ClustaW, podemos citar otros softwares que usan esta metodología como son: Clustal Ω (Sievers et al, 2011), PRANK (Lötynoja and Goldman, 2005), Reticular Alignment (RetAlign) (Szabó et al, 2010), MUltiple Sequence Comparison by Log-Expectation (MUSCLE) (Edgar, 2004a), Multiple Alignment using Fast Fourier Transform (MAFFT) (Katoh et al, 2002) or Kalign (Lassmann and Sonnhammer, 2005).

4.3.2. Métodos basados en consistencia

Los métodos basados en consistencia fueron diseñados para considerar simultáneamente más secuencias que los métodos progresivos y en un tiempo de ejecución razonable (Gotoh, 1990). El objetivo principal de estos métodos es construir una base de datos de alineamientos locales y globales para con ello lograr encontrar una alineamiento final.

Estas técnicas aprovechan la información contenida dentro de las regiones que están alineadas consistentemente entre un conjunto de pares de superposiciones con el fin de realinear pares de secuencias a través de métodos de refinamiento global y local (Ebert and Brutlag, 2006). Los métodos basados en consistencia más importantes son: Tree-based Consistency Objective Function For alignment Evaluation (T-Coffee) (Notredame et al, 2000), PROBabilistic CONSistency-based MSA (ProbCons) (Do et al, 2005), Fast Statistical Alignment (FSA) (Bradley et al, 2009), ProbAlign (Roshan and Livesay, 2006) y MSAProbs (Liu and Schmidt, 2014).

4.3.3. Algoritmos evolutivos monoobjetivo

Los algoritmos evolutivos son particulamente adecuados para problemas de esta naturaleza. Un algoritmo evolutivo no puede competir en términos de velocidad con métodos de alineamiento progresivo, pero tiene la ventaja de poder corregir secuencias inicialmente desalineadas, lo que no es posible con los métodos progresivos.

En la literatura, encontramos diversos algoritmos evolutivos (EA) propuestos al problema del MSA. Una de los primeras propuestas fue Sequence Alignment by Genetic Algorithm (SAGA) presentado por (Notredame and Higgins, 1996), el cual es un algoritmo genético (GA) que implementa una población inicial aleatoria y varios operadores genéticos disponibles, que se programan dinámicamente durante la ejecución y se aplican con el objetivo de mejorar el fitness de los alineamientos.

Multiple Sequence Alignment Genetic Algorithm (MSA-GA) es otro algoritmo genético propuesto por (Gondro and Kinghorn, 2007). Este GA utiliza alineamientos previamente computados por Clustal-W (Thompson et al, 1994) para generar la población inicial, mejorando el resultado inicial. Emplean operadores de cruce horizontal y vertical; en el primero se seleccionan aleatoriamente de qué padre escoger cada una de las secuencias que van a ir generando el alineamiento,

y en el segundo se corta verticalmente en el mismo punto a ambos padres y se escoge de forma aleatoria (si del padre o de la madre) las dos partes del nuevo alineamiento descendiente. Además, con el fin de evitar estancamientos en mínimos locales, si el *score* del mejor alineamiento no mejora en 1000 evaluaciones aplica una algoritmo *greedy hill climbing* con el objetivo de mejorar aún mas el score de los alineamientos. La función objetivo es Suma de Pares ponderada empleando como Matriz de Distancia BLOSUM62. La matriz de ponderaciones puede ser definida por el usuario o puede ser obtenida de las puntuaciones de los alineamientos de pares de secuencias utilizados en la generación de la población inicial con el método neighbor-joining (Saitou and Nei, 1987b).

(Taheri and Zomaya, 2009) propusieron una metodología de solución híbrida que combina la técnica Rubber Band Technique (RBT) (inspirada en el comportamiento de una banda elástica de caucho) y un algoritmo genético, llamada RBT-GA, que es un algoritmo de optimización basado en la población diseñado para encontrar un alineamiento óptimo para un conjunto de secuencias de proteínas. Cada respuesta de alineamiento se modela como un cromosoma que consta de varios polos en el RBT framework. Estos polos se asemejan a secciones en las secuencias de entrada que tienen más probabilidades de estar correlacionados y/o biológicamente relacionados. Un proceso de optimización basado en GA mejora estos cromosomas dando gradualmente un conjunto de respuestas en su mayoría óptimas para el problema MSA. En esta propuesta cada respuesta de alineación se modela como un cromosoma que consta de varios polos en el marco RBT. El proceso de optimización basado en GA mejora estos cromosomas dando gradualmente un conjunto de respuestas en su mayoría óptimas para el problema MSA. La Suma de Pares con penalización de gaps de entrada (Sum-of-Pairs Score (SPS) with Penalized Gap Opening) es la función objetivo a optimizar. Su propuesta fue probada usando el benchmark BALiBASE 2.0 (Thompson et al, 1999b).

Naznin *et al.* presentaron en (Naznin et al, 2011) el algoritmo Vertical Decomposition Genetic Algorithm (VD-GA). Este GA divide las secuencias verticalmente en dos o más subsecuencias para resolverlas individualmente usando un enfoque de árbol guía. Luego, todas las subsecuencias se unen para generar el alineamiento óptimo final. Esta técnica se aplica sobre las soluciones de la población inicial y sobre cada nueva generación de descendientes dentro del algoritmo. VDGA utiliza dos mecanismos para generar la población inicial, la primera es generar árboles guía con secuencias seleccionadas al azar y la segunda es mezclar las secuencias dentro de estos árboles. Para probar el rendimiento de su algoritmo, se usaron el conjuntos de secuencias del benchmark BAliBase 2.0 (Thompson et al, 1999b).

Un método de alineamiento progresivo utilizando un algoritmo genético, llamado GAPAM se presentó en (Naznin et al, 2012). Para evaluar la calidad de los alineamientos usan como función la Suma de Pares ponderadas con la matriz de distancia PAM250 y el valor de penalidad de los gaps establecido por defecto en ClustaW. Para la generación de nuevos descendientes utiliza tres operadores genéticos aplicados a MSA: Cruce de un sólo punto, Cruce de Varios puntos y Mutación, este útlimo basado en distancias y un árbol guía. GAPAM introduce dos mecanismos para generar población inicial: el primero basado en árboles de guía aleatorios y el segundo basado en la mezcla de secuencias en tales árboles. Su rendimiento fue evaluado resolviendo un número de datos del benchmark BAliBASE 2.0 (Thompson et al, 1999b).

4.3.4. Algoritmos evolutivos multiobjetivo

Recientemente ha habido un creciente interés en la formulación multiobjetivo de los problemas de optimización que surgen en el campo de la Bioinformática. Handl *et al.* muestra en (Handl et al, 2007) los beneficios de la Optimización Multiobjetivo aplicada específicamente en el campo de la Bioinformática en comparación con los enfoques monoobjetivo. A continuación se detallan algunos trabajos:

En los últimos tiempos, se han publicado varias propuestas multiobjetivo para resolver el problema del MSA usando técnicas metaheurísticas. La primera aproximación multiobjetivo fue presentada por Seeluangsawat *et al.* quienes publicaron MOMSA (Multiple Objective Multiple Sequence Alignment) un algoritmo evolutivo que optimiza dos objetivos de calidad implementados en una sola función, con el fin de mejorar las soluciones obtenidas desde el software Clustal X (Thompson et al, 1994) en (Seeluangsawat and Chongstitvatana, 2005). La población inicial del algoritmo se genera a partir de los resultados generados por el software Clustal X extendiendo el tamaño de los alineamientos un 10 % mas. Considera dos objetivos a optimizar, la Suma de Pares y La Penalidad de Gaps, empleando como matriz de distancia Blosum45. MOMSA implementa un operador de cruce de dos puntos y tres operadores de mutación (Movimientos de columnas, Cambio de posición de Gaps e Intercambio aleatorio entre residuo y grupos de gaps). Este algoritmo propuesto fue probado con nueve conjuntos de datos del benchmark BAliBASE 2.0 (Thompson et al, 1999b).

Ortuño *et al.* presentaron en (Ortuño et al, 2013) MO-SAStrE (Multiobjective Optimizer for Sequence Alignments based on Structural Evaluations), cuyo algoritmo está basado en la clásica metaheurística NSGA-II y trata de optimizar tres objetivos de calidad, uno basado en información estructural STRIKE, el porcentaje de no-Gaps y el porcentaje de columnas totalmente conservadas. En MO-SAStrE, la población inicial se genera mediante la estrategia basada en alineamientos precomputados generados por ocho enfoques representativos del estado del arte, tales como Muscle, ClustalW, Mafft, T-Coffee, Kalign, RetAlign, ProbCons y FSA. Emplea el operador de cruce de un solo punto y como operador de mutación desplazamiento aleatorio de una región de gaps dentro de una secuencia. El rendimiento del algoritmo fue evaluado resolviendo los 218 problemas del benchmark BAliBASE (v3.0) (Thompson et al, 2005). Los resultados multiobjetivo de MOSAStrE fueron evaluados usando el indicador de calidad multiobjetivo Hypervolumen (Zitzler and Thiele, 1999a).

Soto y Becerra propusieron en (Soto and Becerra, 2014) un algoritmo evolutivo multiobjetivo, también inspirado en NSGA-II, para optimizar alineamientos múltiples de secuencias previamente alineadas. Para evaluar la calidad de los individuos utilizan dos métricas de calidad MetAl (Blackburne and Whelan, 2012c) y Entropy, esta última mide la variabilidad de un MSA definiendo las frecuencias de la ocurrencia de cada letra en cada columna. Siendo estas dos los objetivos a optimizar dentro del algoritmo. Los resultados fueron validados por cuatro métricas de calidad estándar: La suma de pares (SOP), Número de Columnas Totalmente Alineadas (TC), MetAl e Hypervolume (Zitzler and Thiele, 1999a). SP y TC fueron computadas usando el script de *Balis-core* (Thompson et al, 2005). Similar a MO-SAStrE, esta propuesta construye la población inicial usando los alineamientos producidas por otros técnicas MSA de última generación, más algunas modificaciones de operadores genéticos. Como operadores de variación aplicaron cruzamiento de dos puntos y mutación de inserción aleatoria y desplazamiento. El método propuesto fue validado resolviendo los 218 instancias del benchmark BAliBASE (3.0).

Kaya *et al.* presentaron MSAGMOGA (Kaya et al, 2014), basado también en NSGA-II, considera tres objetivos conflictivos a optimizar: minimización de la penalidad de gaps afín y de Similitud y la maximización de la Compatibilidad. MSAGMOGA aplica operadores de cruce de un solo punto y dos puntos y tres operadores de mutación (cambio aleatorio de gaps, desplazamiento hacia derecha e izquierda de bloques degaps). Los resultados se obtuvieron del benchmark BAliBASE 2.0 (Thompson et al, 1999b).

da Silva *et al.* presentaron Parallel Niche Pareto AlineaGA (PNPAlineaGA) una versión multiobjetivo de su algoritmo Parallel AlineaGA en (da Silva et al, 2011). Usa dos objetivos a optimizar: La Suma de pares (SOP) con la matriz de distancia PAM350 y el número de columnas totalmente alineadas en el alineamiento. PNPAlineaGA implementa tres operadores de cruce y seis versiones de operadores de mutación. Los resultados fueron validadps sobre 8 datasets del benchmark BAliBASE 2.0 (Thompson et al, 1999b).

Abbasi *et al.* publicaron en su trabajo (Abbasi et al, 2015) varios técnicas de Búsqueda Local

aplicadas al Alineamiento Múltiple de Secuencias Multiobjetivo, sus objetivos a optimizar fueron maximizar la Suma de Pares y minimizar el número de gaps. A pesar que ilustran buenos resultados obtenidos gracias a su técnica propuesta, Pareto Local Search, indican que deben mejorarla, ya que en algunos casos se estanca en óptimos locales. Para ello , sugieren perturbar el conjunto de alineamientos en un archivo de soluciones y reiniciar la búsqueda local cada cierto número de iteraciones. El rendimiento de los algoritmos de búsqueda local fueron probados resolviendo 38 instancias del benchmark BAliBASE 3.0.

Zhu *et al.* presentaron en (Zhu et al, 2016) una propuesta basada en el algoritmo evolutivo multiobjetivo basado en la descomposición (MOEA/D) aplicado a resolver el problema del MSA, llamado MOMSA. Zhu *et al.* resaltan dos nuevas aportaciones dentro de su algoritmo: la generación de la población inicial y un nuevo operador de mutación. La población inicial es generada mediante una nueva técnica basada en inserción de gaps similar al funcionamiento del algoritmo en SAGA (Notredame and Higgins, 1996). A partir de alineamientos previamente obtenidos de alguna técnica como ClustalW, las secuencias son divididas al azar en dos grupos, se insertan un número gaps en posiciones aleatorias, y luego estos grupos de secuencias son unificados para conformar el alineamiento final. El rendimiento de esta técnica se comparó con varios métodos de alineamientos basados en algoritmos evolutivos, y también con técnicas basadas en métodos progresivos, usaron el conjunto de datos del BAliBASE 2.0 y BAliBASE 3.0 para evaluar su rendimiento.

Recientemente, Rubio-Largo *et al.* propusieron dos nuevas técnicas para resolver el problema MSA basadas, el algoritmo Hybrid Multiobjective Artificial Bee Colony (HMOABC) (Rubio-Largo et al, 2016) y el algoritmo hybrid multiobjective memetic metaheuristic (H4MSA) (Rubio-Largo et al, 2015) Ambos basados en algoritmos bioinspirados, HMOABC inspirado en el comportamiento natural de las colonias de abejas y, H4MSA inspirado en la metaheurística memética Shuffled Frog-Leaping Algorithm (SFLA) (Eusuff et al, 2006). Con el objetivo de preservar la calidad y consistencia de sus alineamientos, ambos consideran dos funciones objetivo: La Suma Ponderada de Pares con penalidad de gaps afines(WSP) y el número de columnas totalmente conservadas (TC), respectivamente. La metodología híbrida de ambos algoritmos, está basada en el uso de la técnica progresiva KAlign (Lassmann and Sonnhammer, 2005), unos de los software más rápidos y precisos del estado del arte. En HMOABC, esta técnica adicional se utiliza en la fase de exploración del algoritmo ABC (Artificial Bee Colony) y, en H4MSA ésta se emplea como un procedimiento de búsqueda local que busca alinear mejormente pequeñas porciones de los alineamientos. Ambos algoritmos, generan aleatoriamente la población inicial. El desempeño de H4MSA fue probado en tres benchmarks de referencia: BAliBASE (Thompson et al, 2005), Protein REFerence Alignment Benchmark (PREFAB) (Edgar, 2004b) y el Sequence Alignment Benchmark (SABmark) (Van Walle et al, 2005) y, el desempeño de HMOABC fue probado únicamente utilizando el benchmark BAliBASE (v3.0).

Y por último, Ranjani Rani *et al.* propusieron en (Rani and Ramyachitra, 2016) dos algoritmos: Hybrid Genetic Algorithm with Artificial Bee Colony Algorithm (GA-ABC) y Bacterial Foraging Optimization Algorithm (MO-BFO), pero su trabajo se centra principalmente en el rendimiento del algoritmo MO-BFO ya que obtiene un mejor rendimiento y porque identifica mayormente bloques conservados dentro de los alineamientos. Ranjani Rani *et al.* incorporaron en su trabajo cuatro objetivos a optimizar: la maximización de la Similitud, el porcentaje de no-gaps y Bloques Conservados, y la Minimización de la penalidad por gaps. Los algoritmos propuestos fueron evaluados resolviendo el benchmark BAliBASE v3.0 y comparados con otros métodos MSA clásicosmuy usados como: ClustalW y Clustalω, KAlign, MUSCLE, MAFFT y con varios algoritmos genéticos
.

4.3.4.1. Métodos basados en información estructural

En los últimos años se han desarrollado algunos métodos basados en información estructural para obtener alineamientos precisos. Básicamente, estos enfoques utilizan la información estructural del Protein Data Bank (PDB) Berman et al (2000) como plantilla para guiar el alineamiento de un conjunto de secuencias no alineadas. Dos ejemplos de estas metodologías son: el software 3D-Coffee (Armougom et al, 2006), una versión especial del método progresivo T-Coffee (Notredame et al, 2000) que utiliza alineamientos estructural sobre las secuencias y el algoritmo evolutivo multiobjetivo MO-SAStrE (Ortuño et al, 2013), que entre una de sus funciones objetivo a optimizar está el score basado en información estructural STRIKE (Kemena et al, 2011), un índice para calcular las exactitudes de los alineamientos utilizando al menos la información estructural 3-D de una de las proteínas del conjunto de secuencias a alinear. Para más detalles del funcionamiento del score, ver la subsección 4.2.2.4. El principal inconveniente de estos métodos es la disponibilidad limitada de las estructuras PDB.

4.4. Problemas de pruebas

Actualmente se han diseñado una gran cantidad de técnicas y problemas de pruebas para estandarizar el análisis de calidad y comparativa de resultados de los métodos propuestos para el Alineamiento de Múltiple de Secuencias. Estos problemas de prueba consisten en un conjunto heterogéneo de secuencias no alineadas extraídas de varias bases de datos públicas. Además de las secuencias sin alinear, incluyen los alineamientos óptimos que deben obtenerse para cada uno de los problemas proporcionados, los cuales se conocen generalmente como alineamientos de referencia. Los nuevos métodos propuestos para optimizar el problema del Alineamiento Múltiple de Secuencias pueden comparar sus resultados con estos alineamientos de referencia, con el objetivo de determinar la calidad sus alineamientos. Algunos de los problemas de prueba más utilizados en el estado del arte son: OXBench (Raghava et al, 2003), HOMSTRAD (de Bakker et al, 2001), Prefab (Edgar, 2004a) y BAliBASE (Thompson et al, 1999a, 2005). A continuación se explican en detalle.

4.4.1. Benchmark alignment database (BAliBASE)

BAliBASE (Thompson et al, 1999a, 2005) es una colección de datos de alineamientos de referencias publicadas para validar la calidad biológica de los alineamientos de secuencias generados por las metodologías propuestas para el problema. Representa una de las principales referencias en el estado del arte del MSA. Actualmente existen 4 versiones publicadas (v1.0, V2.0, V3.0 y v4.0). En esta tesis doctoral hemos usado en todos las pruebas la versión BAliBASE 3.0 (Thompson et al, 2005), la cual está compuesta de 6255 secuencias agrupadas en 218 instancias de problemas, extraídas manualmente de la base de datos del Protein Data Bank (Berman et al, 2000). Todas estas secuencias se organizan en seis subconjuntos o familias clasificados por su similitud, su estructura PDB y las familias evolutivas:

- **Familia/Grupo RV11**: Está compuesta por un conjunto de 38 instancias y está formada por secuencias de proteínas PDB equidistantes que comparten $\prec 20\,\%$ de identidad entre cualquier par de secuencias y con un número menos a 35 inserciones.

- **Familia/Grupo RV12**: Está compuesta de un conjunto de 44 instancias e incluye familias que no fueron consideradas en el grupos RV11 con al menos 4 secuencias PDB equidistantes. Comparten un porcentaje de identidad entre el $20\,\%$ y $40\,\%$.

- **Familia/Grupo RV20**: Está compuesta de un conjunto de 41 instancias y está formada por secuencias de proteínas PDB huérfanas que comparten una identidad del mas del 40 %.
- **Familia/Grupo RV30**: Está compuesta de un conjunto de 30 instancias, y contiene secuencias de diferentes subfamilias que comparten mas del 40 % de identidad entre todas ellas.
- **Familia/Grupo RV40**: Este grupo está formado por 49 instancias que comparten mas del 20 % de identidad pero contienen una gran cantidad de inserciones.
- **Familia/Grupo RV50**: Está compuesta de un conjunto de 16 instancias que comparten más del 20 % de identidad y una gran cantidad de inserciones internas asociadas con diferentes secuencias.

Esta clasificación de subconjuntos es un ítem muy importante para estudiar las diferencias en términos de calidad para las herramientas de MSA, especialmente cuando secuencias evolutivamente menos relacionadas deben ser alineadas. Además, BAliBASE proporciona la métrica específica BAliscore (comentada en la sección 4.2.2.3) para evaluar los alineamientos obtenidos de las metodologías MSA. Esta métrica representa el número de coincidencias entre el alineamiento proporcionado y el alineamiento óptimo de referencia propuesto por BAliBASE (Suma de Pares - SP). Por todas estas razones, el benchmark BAliBASE es empleado ampliamente en esta tesis doctoral.

4.4.2. OXBench

Oxbench (Raghava et al, 2003) es una base de datos con varias familias de secuencias junto a su alineamiento óptimo de referencia. Oxbench incorpora además un software de evaluación para medir la calidad de los alineamientos múltiples de secuencias generados por otras metodologías. El conjunto de datos ha sido generado de acuerdo a las definiciones de dominio estructural extraídas de la base de datos 3Dee (Siddiqui et al, 2001) El principal conjunto de datos Oxbench, también llamado conjunto de datos maestro, está compuesto de 672 familias de secuencias. Todas las secuencias en el conjunto de datos maestro contienen conocido estructuras tridimensionales conocidas. Estas familias también se dividen en subconjuntos dependiendo del tipo de análisis de alineación:

- subconjunto para alineamientos pairwise compuesto por 273 familias
- subconjunto para alineamientos múltiples compuesto por 399 familias, y
- subconjunto para alineamientos cortos compuesto por 590 familias.

Hay que considerar que varios de estos subconjuntos pueden contener familias comunes.

4.4.3. Protein reference alignment benchmark (Prefab)

Prefab (Protein REFerence Alignment Benchmark) (Edgar, 2004a) incluye una colección de más de 1900 alineamientos. En este banco de problemas dos secuencias de proteínas no relacionadas se alinean primero mediante un método estructural. Posteriormente, cada una de estas secuencias se utiliza para consultar secuencias altamente relacionadas en una base de datos mediante el uso del Position-Specific Iterative Basic Local Alignment Search Tool (PSI-BLAST). Los dos resultados y las secuencias recuperadas de la base de datos se combinan para armar un MSA para construir así los alineamientos de referencia. Prefab también contiene un software de evaluación llamado Q score para medir la precisión de los alineamientos generados por metodologías MSA contra sus alineamientos de referencia.

4.5. Herramienta software: jMetalMSA

Con el objetivo de ofrecer a la comunidad científica de la biología computacional una plataforma libre que incluya algoritmos de optimización de última generación dirigidos a resolver diferentes formulaciones del MSA, en esta sección presentamos jMetalMSA, una herramienta software de código abierto para el alineamiento múltiple de secuencias con metaheurísticas de optimización multiobjetivo.

A diferencia de los demás trabajos del estado del arte publicados, con algunas excepciones (por ejemplo, (Ortuño et al, 2013)), el código fuente de sus algoritmos propuestos no está disponible públicamente para su uso y beneficio, lo que impide a los investigadores aplicar los algoritmos para resolver sus problemas. Nuestro objetivo es llenar ese vacío y por ende el código fuente del proyecto se encuentra disponible públicamente en GitHub[1].

jMetalMSA está escrito en el lenguaje de programación Java y provee un conjunto de algoritmos referentes, clásicos y recientes del estado del arte multiobjetivo aplicados al problema MSA. Incluye un conjunto de métricas de calidad MSA usadas comúnmente para evaluar la calidad de los alineamientos, las cuales pueden ser usadas como funciones objetivos a optimizar simultáneamente dentro de los algoritmos. Además incluye varios operados genéticos de Cruce y Mutación usados por varios algoritmos evolutivos adaptados al problema.

A continuación describimos la arquitectura de software y sus principales características, incluyendo dos casos de uso práctico de la herramienta.

4.5.1. Arquitectura de jMetalMSA

jMetalMSA está basado en el framework de optimización multiobjetivo jMetal (Durillo and Nebro, 2011)(Nebro et al, 2015a), del cual toma la mayoría de las clases centrales. La arquitectura orientada a objetos de jMetalMSA se muestra en la Figura 4.1, en la que podemos observar que está compuesta de cuatro clases principales (interfaces Java).

Tres de ellos (*MSAProblem*, *MSAAlgorithm*, y *MSASolution*) heredan de sus equivalentes en jMetal (algunas relaciones de herencia se han omitido en el diagrama con el fin de simplificarlo y facilitar su comprensión), y además hay una clase *Score* para representar las métricas de calidad MSA .

Muchas propuestas para resolver el problema del MSA con metaheurísticas incluyen un método de inicialización basado en la toma de un conjunto de alineamientos pre-calculadas obtenidas por otras metodologías MSA no metaheurísticas (tales como Clustal-W, MAFFT, MUSCLE, etc.), por lo que la clase *MSAProblem* incluye el método *createInitialPopulationMethod ()* la cual está destinada para incorporar este tipo de estrategias.

Una cuestión importante en los actuales estudios de MSA es que no hay un consenso formal acerca de qué métricas son las más adecuadas para evaluar la calidad de los alineamientos. Los enfoques multi-objetivos en jMetalMSA tratan de superar esta cuestión en cierta medida seleccionando más de una métrica simultáneamente, pero una revisión de las propuestas existentes revela que la mayoría de ellos utilizan diferentes formulaciones de problemas. Para proporcionar un alto grado de flexibilidad en la definición de un problema MSA con las funciones objetivos particulares (métricas MSA), nuestro enfoque es tener una clase *Score* que represente las métricas de calidad MSA (funciones de evaluación) individuales, de modo que una formulación concreta de un problema MSA esté compuesta por una serie de puntuaciones (objetivos a optimizar) que se puede especificar fácilmente al configurar los algoritmos.

Al aplicarse los operadores genéticos a los alineamientos, existe la posibilidad de obtener columnas completamente llenas de gaps. Estas columnas son inútiles y normalmente son eliminadas,

[1]Weg del Proyecto jMetalMSA en GitHub: http://github.com/jmetal/jmetalmsa

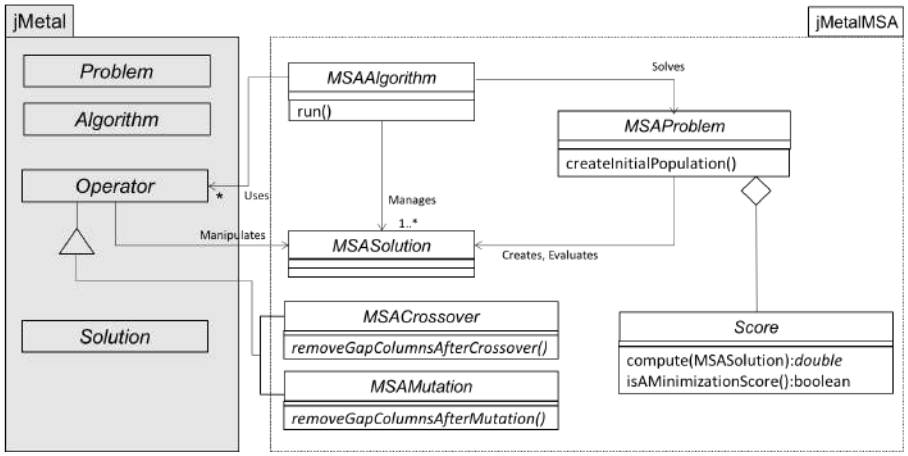


Figura 4.1: Arquitectura de jMetalMSA (algunas relaciones de herencia no han sido incluidas para simplificar el diagrama).

pero encontrarlas dentro de los MSA codificados es un proceso que consume mucho tiempo.El cruce y la mutación son típicamente operaciones consecutivas, y también existe la posibilidad de aplicar más de una operación de mutación, por lo que eliminar los gaps al final de cada operador puede ser muy ineficiente. Los operadores de cruce y mutación en jMetalMSA están dotados de un método para indicar si la remoción debe llevarse a cabo después de los operadores o no.

4.5.2. Algoritmos incluidos en jMetalMSA

Como jMetalMSA se basa en jMetal, la mayoría de los algoritmos incluidos en el último se puede utilizar en el primero. La codificación de los alineamientos (ver sección 4.5.3) está basada en la representación de los grupos de gaps, por lo que los algoritmos de optimización continua, tales como la optimización de enjambre de partículas y la evolución diferencial no pueden ser utilizados en sus versiones clásicas.

A continuación se describen brevemente los algoritmos multiobjetivos disponibles en jMetalMSA: NSGA-II (Deb et al, 2002b), NSGA-III (Deb and Jain, 2014), SMS-EMOA (Beume et al, 2007), SPEA2 (Zitzler et al, 2001), MOEA/D (Zhang and Li, 2007), MOCell (Nebro et al, 2009), y GWASF-GA (R. Saborido and Luque., 2016).

- **NSGA-II** (Deb et al, 2002b) es el algoritmo multiobjetivo de mayor referencia dentro del estado del arte. Aunque se definió en 2002 todavía se considera como una técnica importante y en muchos casos su rendimiento es notable cuando se trata de problemas bi-objetivos. Es un algoritmo generacional genético basado en la obtención de un nuevo individuo de la población original mediante los operadores genéticos típicos (selección, cruce y mutación). Un procedimiento de clasificación se aplica para promover la convergencia, mientras que un estimador de densidad (la distancia de crowding) se utiliza para mejorar la diversidad del conjunto de soluciones encontradas.

- **SPEA2** (Zitzler et al, 2001) es, como NSGA-II, un algoritmo clásico y muy utilizado. Se caracteriza por el uso de una población y un archivo, y la fuerza del fitness y la distancia al k-ésimo vecino más cercano se aplican para fomentar la convergencia y la diversidad, respectivamente.

- **MOCell** (Nebro et al. 2009) es un algoritmo evolutivo celular multiobjetivo que utiliza un archivo externo para almacenar las soluciones no dominadas encontradas durante la búsqueda. Como el archivo está limitado en tamaño, se aplica un estimador de densidad (la misma distancia de crowding utilizada en NSGA-II) para seleccionar qué solución debe eliminarse cuando se excede el tamaño del archivo.
- **MOEA/D** (Zhang and Li, 2007) está basado en la descomposición de un problema de optimización multiobjetivo en una serie de subproblemas de optimización escalares, que se optimizan simultáneamente, usando únicamente información de sus subproblemas vecinos. Este algoritmo también aplica un operador de mutación a las soluciones.
- **SMS-EMOA** (Beume et al. 2007) es un algoritmo evolutivo en estado estacionario que utiliza un operador de selección basado en la indicador de calidad multiobjetivo hipervolumen combinada con el concepto de ordenamiento no-dominado.
- **GWASF-GA** (R. Saborido and Luque., 2016) es un algoritmo evolutivo basado en preferencias que puede encontrar una aproximación del frente de Pareto usando una función de escalarización de logro ponderado.
- NSGA-III (Deb and Jain, 2014) Es un algoritmo enfocado a la optimización de muchos objetivos. Está basado en NSGA-II, pero reemplazando la distancia de crowding por un enfoque basado en puntos de referencia.

Estos algoritmos constituyen el conjunto de algoritmos evolutivos multiobjetivos representativos del estado del arte: están los referencias clásicas (NSGA-II, SPEA2), celular (MOCell), basada en la descomposición (MOEA/D), basada en indicadores (SMS-EMOA) y basado las preferencias (GWASF-GA).

4.5.3. Codificación de las soluciones (MSA)

La elección del esquema para representar la soluciones de un problema es un aspecto muy importante en la implementación y uso de metaheurísticas, ya que influye en gran medida en el rendimiento y ejecución de los operadores genéticos de variación, típicamente cruce y mutación

Con el objetivo de reducir el alto costo de memoria y el tiempo de ejecución que requieren las codificaciones clásicas de los alineamientos, cadenas de caracteres o matrices numéricas como en el caso de MO-SAStrE (Ortuño et al. 2013), hemos implementado una codificación, rápida y de bajo costo de memoria, basada en los grupos de gaps dentro de las secuencias, similar a la propuesta por Rubio-Largo et al (2015). Esta representación MSA almacena únicamente las posiciones (inicio y fin) de los grupos de gaps dentro de las secuencias alienadas. En la Figura 4.2 se ilustra un ejemplo.

Figura 4.2: Ejemplo de un alineamiento (izquierda) y como es codificada en jMetalMSA (derecha).

Por lo que, dada una secuencia S, esta es codificada a S' de la siguiente manera:
$s' : [(Igg_1, Fgg_1), (Igg_2, Fgg_2), ..., (Igg_n, Fgg_n)]$
donde n es el número de grupos de gaps de la secuencia S e Igg_x y Fgg_x representan la posición inicial y la posición final del grupo de gaps x dentro de la secuencia S, respectivamente. Esta

codificación reduce el tiempo de ejecución de los operadores genéticos de cruce y mutación, ya que únicamente se ejecutan operaciones numéricas sobre las grupos gaps y no se tienen que realizar operaciones sobre grandes secuencias de caracteres.

4.5.4. Operadores evolutivos

Como se ha comentado en la sección anterior, muchas metaheurísticas (en particular los algoritmos evolutivos) se basan en los operadores de cruce y mutación para buscar las soluciones del problema. En jMetal se ha implementado los mismos operadores de cruce y mutación empleados para el algoritmo MO-SAStrE (Ortuño et al, 2013).

El operador de cruce es el operador de Cruce de un sólo punto que ha sido adaptado a los alineamientos (da Silva et al, 2010). Este operador selecciona aleatoriamente una posición del padre 1 y lo divide en dos bloques ($P1_a$ y $P1_b$), luego el padre 2 es cortado en dos bloques ($P2_a$ y $P2_b$) de tal manera que la parte de derecha ($P2_b$) pueda encajar a la pieza izquierda del primer padre ($P1_a$) y viceversa. Entonces, los bloques seleccionados se cruzan entre estos dos padres, generando dos nuevos individuos con la combinación de los bloques: $[P1_a \cdot P2_b]$ y $[P2_a \cdot P1_b]$, en la Figura 4.3 se ilustra un ejemplo. Finalmente, con el objetivo de reducir el número de gaps en el alineamiento, luego de la ejecución de ambos operadores genéticos se eliminan las columnas que contienen solo gaps.

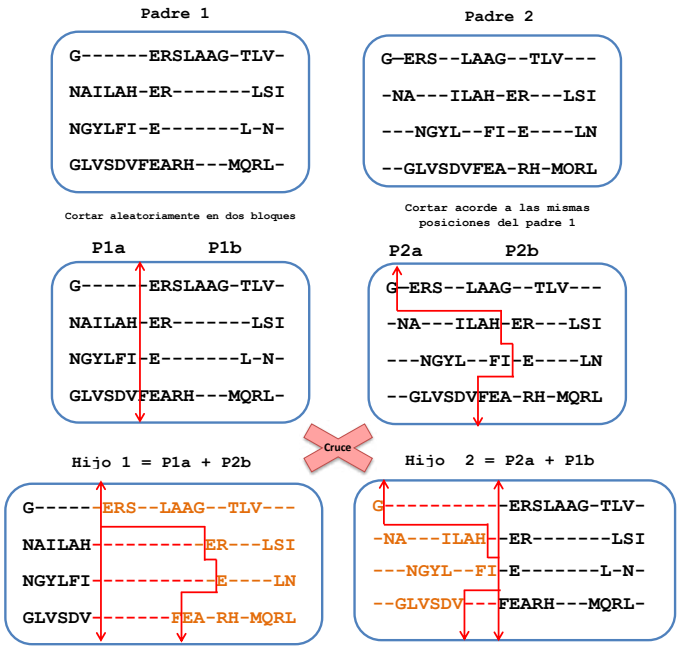


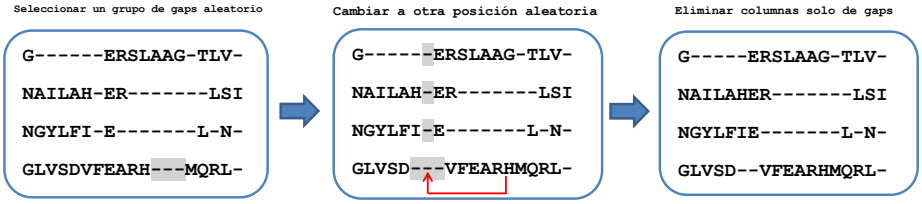
Figura 4.3: Operador de cruce de un solo punto: El primer padre es cortado en dos bloques en una posición elegida al azar. El segundo es cortado en dos bloques, pero de la forma que se adapte el bloque derecho con el bloque izquierdo del primer padre y viceversa.

La lista de operadores de mutación incluida en jMetalMSA son:

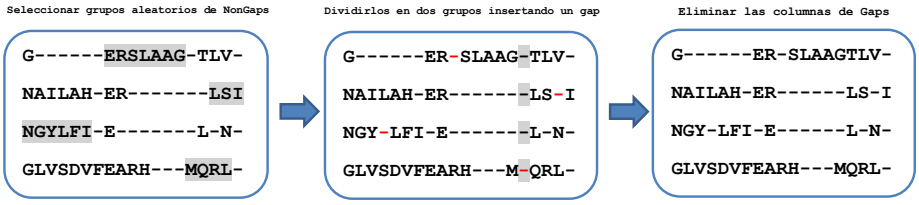
- Desplazamiento de grupo de gaps: selecciona una secuencia aleatoriamente del MSA; Y del grupo de huecos que contiene, se escoge uno al azar y se lo desplaza a otra posición aleatoria dentro de la misma secuencia (véase Figura 4.4 a).

- Separación de grupos No-gaps: se selecciona aleatoriamente un grupo de residuos (no-gaps) y se lo divide en dos grupos insertando un gap en una posición aleatoria entre ellos. (véase Figura 4.4 b).
- Inserción de un gap: Inserta un gap en una posición aleatoria para cada secuencia del alineamiento (véase Figura 4.4 c).
- Fusión de dos grupos de gaps adyacentes: Selecciona un grupo aleatorio de gaps y éste se fusiona con su grupo de gaps más cercano (véase Figura 4.4 d).
- Mutación múltiple: Es una combinación del resto de operadores en uno solo.

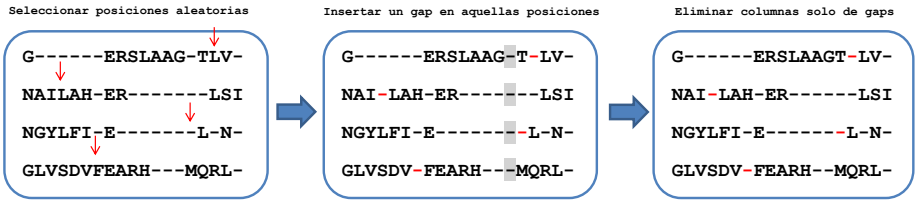
En todo los operadores de mutación se incluye la opción, como se comentó anteriormente, de detectar columnas llenas únicamente de gaps y la eliminación de las mismas.



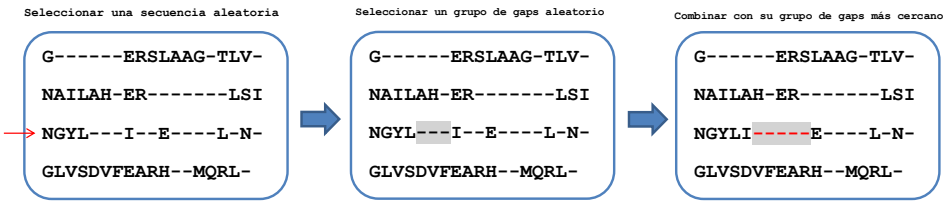
(a) Desplazamiento de grupo de gaps



(b) Separación de grupos No-gaps



(c) Inserción de un gap



(d) Fusión de dos grupos de gaps adyacentes

Figura 4.4: Lista de operadores de mutación disponibles en jMetalMSA.

Tabla 4.1: Métodos utilizados para generar la población inicial de los algoritmos. Estas ocho herramientas se aplican para construir alineaciones de secuencias múltiples iniciales para los conjuntos de datos BAliBASE.

Herramienta	**Versión**	**Tipo de Metodología**
ClustalW (Thompson et al, 1994)	2.1	Progresivo
MUSCLE (Edgar, 2004a)	3.8.31	Progresivo
Kalign (Lassmann and Sonnhammer, 2005)	2.04	Progresivo
Mafft (Katoh et al, 2002)	7.245	Progresivo
RetAlign (Szabó et al, 2010)	1.0	Progresivo
TCOFFEE (Notredame et al, 2000)	11.00	Basado en Consistencia
ProbCons (Do et al, 2005)	1.12	Basado en Consistencia
FSA (Bradley et al, 2009)	1.15.9	Basado en Consistencia

La Figura 4.4 muestra un ejemplo de los cuatro primeros operadores de mutación. El último, Mutación múltiple, permite combinar cualquier la ejecución de los otros operadores en uno solo.

4.5.5. Estrategia para generar la población inicial

El mecanismo habitual para generar la población inicial en algoritmos evolutivos es crearla con soluciones generadas de manera aleatoria, pero en MO-SAStrE se utiliza un conjunto de alineamientos pre-computados para su generación. En jMetlMSA, seguimos una estrategia similar: primero un conjunto de 8 alineamientos, previamente generados por varias técnicas clásicas MSA, son insertados directamente a la población inicial, luego a partir de estos alineamientos, se procede a completar el resto de la población, mediante la creación de nuevas soluciones generadas por el operador de cruce genético de un solo punto y la selección aleatoria de dos alineamientos de la población.

Este proceso se repite hasta que la población esté completamente llena. Las versiones y características específicas de las metodologías MSA (ClustalW, MUSCLE, Kalign, Mafft, RetAlign, TCOFFEE, ProbCons y FSA) usadas para generar los alineamientos pre-computados empleados en esta estrategia de generación de la población inicial se detallan en la Tabla 4.1.

4.5.6. Funciones objetivo

Las métricas de calidad MSA están implementadas como funciones objetivo dentro de jMetalMSA, se ha empleado la clase abstracta *Score* para la implementación de cada una de ellas. Todas las funciones objetivo buscan ser son maximizadas.

La lista de métricas de evaluación MSA disponibles incluida en jMetalMSA incluyen la métrica basada en información estructural STRIKE, las dos métricas basadas en información de matrices de distancias y penalidad de gaps Suma de Pares (SOP) y Suma de Pares ponderadas con penalidad de gaps afines (wSOP); y las métricas basada en la conservación de la consistencia Porcentaje de Columnas Totalmente Alineadas (TC) y el Porcentaje de Non-Gaps (Non-Gaps). Más detalles de la formulación de cada una de estas métricas de calidad MSA se encuentran en el subsección 4.2.2.

4.5.6.1. Matrices de distancia

Las matrices de distancia empleadas en la formulación de las métricas Sumas de Pares y Suma de Pares ponderadas con penalidad de gaps afines ($ScoringMatrix$) representa la matriz que

determina el costo de sustituir un residuo por otro, además el valor de penalidad de los gaps para determinar el costo de alinear un residuo con un gap y viceversa. Esta penalización sólo se emplea al alinear un residuo con un gap y viceversa. La alineación de dos o más gaps no es penalizado. JMetalMSA incluye las dos matrices de sustitución ampliamente utilizadas en la literatura BLOSUM62 (Henikoff and Henikoff, 1992) y PAM250 (Dayhoff et al, 1978).

4.5.7. Ejemplo: optimización de tres objetivos

Para ilustrar el uso de jMetalMSA se va a partir de una formulación con tres objetivos tomando como base la instancia BB11001 del BAliBASE 3.0. En el ejemplo 1 se usa el algoritmo MOCell empleando las métricas de calidad Suma de Pares, TC y %Non-Gaps como objetivos a optimizar y en el ejemplo 2 se usa el algoritmo NSGA-II empleando las métricas de calidad STRIKE, TC y %Non-Gaps como objetivos a optimizar.

Para configurar y ejecutar los algoritmos se debe implementar una clase *Runner* (*MOCellRunner* y *NSGAIIRunner*) con las instrucciones necesarias para instanciar la clase *AlgorithmBuilder* (*MOCellMSABuilder* y *NSGAIIMSABuilder*) las cuales a su vez están encargadas de configurar e instanciar los algoritmos *MOCellMSA* y *NSGGAIIMSA*. Además se debe establecer los operadores genéticos a usar y los objetivos a optimizarse. En primer lugar, los operadores se indican de esta manera:

```
crossover = new SPXMSACrossover(0.8);
mutation = new ShiftClosedGapsMSAMutation(0.2);
selection = new BinaryTournamentSelection();
```

En segundo lugar, se crea una lista con los scores seleccionados; la cual es enviada como uno de los argumentos de entrada para la creación del Problema MSA. En este ejemplo, para resolver los problemas del BAliBASE usamos la clase *BAliBASE_MSAProblem*, caso contrario para cualquier otro problema MSA estándar usamos la clase *Standard_MSAProblem*.

Para el caso de optimizar los objetivos Suma de Pares, TC y %Non-Gaps, las instrucciones serían:

```
List<Objective> scoreList = new ArrayList<>();
scoreList.add(new SumOfPairsObjective(new PAM250()));
scoreList.add(new PercentageOfAlignedColumnsObjective());
scoreList.add(new PercentageOfNonGapsObjective());
problem = new BAliBASE_MSAProblem(balibaseInstanceName, dataDirectory, scoreList);
```

Y, para el caso de optimizar los objetivos STRIKE, TC y %Non-Gaps, habría que hacer lo siguiente:

```
List<Score> scoreList = new ArrayList<>();
StrikeScore objStrike = new StrikeScore();
scoreList.add(objStrike);
scoreList.add(new PercentageOfAlignedColumnsScore());
scoreList.add(new PercentageOfNonGapsScore());
problem = new BAliBASE_MSAProblem(balibaseInstanceName, dataDirectory, scoreList);
objStrike.initializeParameters(problem.PDBPath, problem.getListOfSequenceNames());
```

Una vez configurado y ejecutado el algoritmo, jMetalMSA genera dos ficheros como resultado de la ejecución (*VAR* y *FUN*), el primero contiene todas las soluciones no-dominadas (alineamientos en formato FASTA) y el segundo los valores de las funciones objetivo que forman la aproximación

al frente de Pareto. La Figura 4.5 muestra las aproximaciones a los frentes de Pareto generados por los algoritmos MOCell y NSGAII con la formulación de tres objetivos, en la que cada uno de los puntos corresponden a los alineamientos generadas por los algoritmos. De allí los biólogos tienen la opción de seleccionar, de este conjunto de soluciones no-dominadas, el alineamiento que mejor satisfaga sus criterios.

Las Figuras 4.6 y 4.7 muestran los alineamientos correspondientes a puntos de los extremos de los frentes (el mejor alineamiento por cada objetivo) de las figura 4.5 . Podemos observar que el alineamiento con la mejor puntuación de TC contiene el mayor número de columnas totalmente alineadas, mientras que el alineamiento con la mejor puntuación de Non-Gaps contiene un menor número de gaps en las secuencias que otros y por ende pose un menor tamaño.

4.5.8. Ejemplo: optimización de dos objetivos

Con el fin de complementar la sección anterior, aquí se muestra el uso de jMetalMSA resolviendo una formulación bi-objetivo del problema MSA. En la figura 4.8 se ilustran los resultados del uso de los algoritmos GWASF-GA y NSGA-II al alinear las secuencias de las instancias BB12001 y BB12003 del BAliBASE 3.0, optimizando dos objetivos: La Suma de Pares (SOP) y el porcentaje de Columnas Totalmente Alineadas (TC).

4.6. Análisis comparativo

En la sección anterior se han ofrecido algunas muestras de los tipos de frentes que se pueden obtener al resolver instancias del problema MSA usando formulaciones con dos y tres objetivos. El objetivo de esta sección es llevar a cabo un estudio comparativo exhaustivo de varios algoritmos referentes, clásicos y recientes del campo e la optimización multiobjetivo aplicados a resolución del problema MSA. Se han llevado a cabos dos análisis por separado, uno bajo una perspectiva de bi-objetivo y la otra basada en una formulación de tres objetivos.

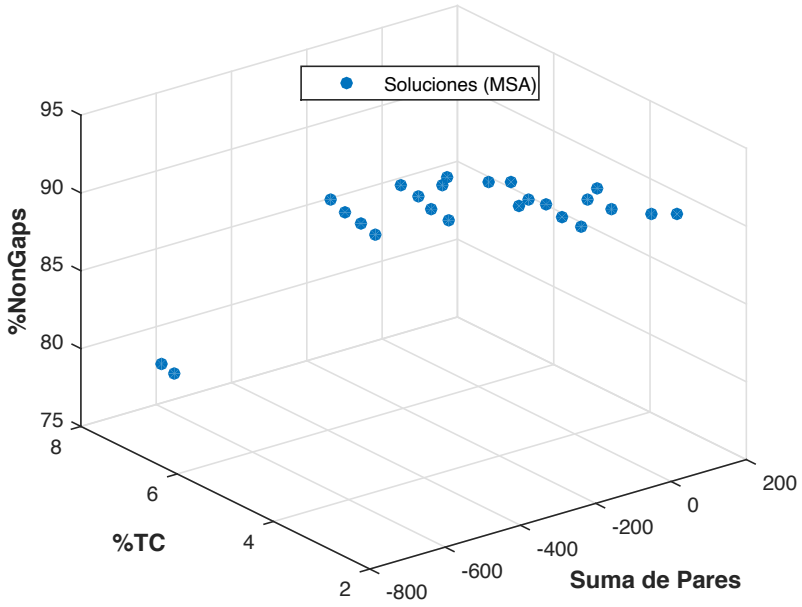
A continuación detallaremos la metodología aplicada, los algoritmos empleados, los objetivos a optimizar, el banco de problemas de pruebas y los resultados obtenidos para ambas comparativas.

4.6.1. Formulación biobjetivo

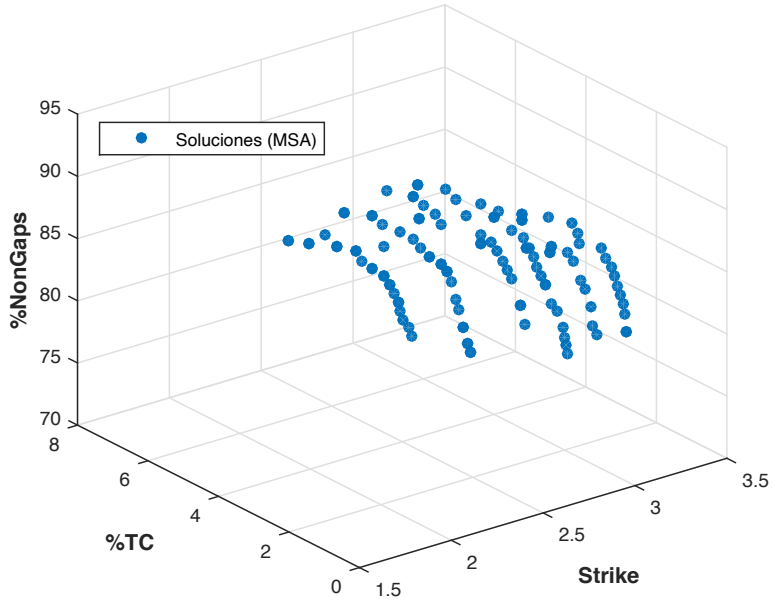
Para este primer análisis se ha considerado una formulación de dos objetivos del problema MSA, siendo las métricas de calidad la Suma de Pares (SOP) y el Porcentaje de Columnas Totalmente Alineadas (TC) las funciones a optimizar. Los algoritmos elegidos son NSGA-II, SPEA2, MOCell, MOEA/D, SMS-EMOA y GWASF-GA, descritos en la sección 4.5.2. El banco de pruebas es BAliBASE v3.0.

4.6.1.1. Configuración de parámetros

Con el objetivo de realizar una comparativa equitativa hemos configurado todos los algoritmos con la misma parametrización. Todos usan el mismo operador de cruce de un solo punto y el operador de mutación de desplazamiento de grupos de gaps del jMetalMSA descritos en la subsección 4.5.4. Ambos operadores son aplicados con probabilidades de 0,8 y 0,2. El tamaño de la población es 100 y la condición de parada se establece para calcular un número total de 50.000 evaluaciones. El algoritmo MOCell utiliza un archivo externo de tamaño 100; el tamaño del vecindario, el número de soluciones sustituidas y la probabilidad de selección del vecindario en MOEA/D son 20, 2 y 0.9, respectivamente.

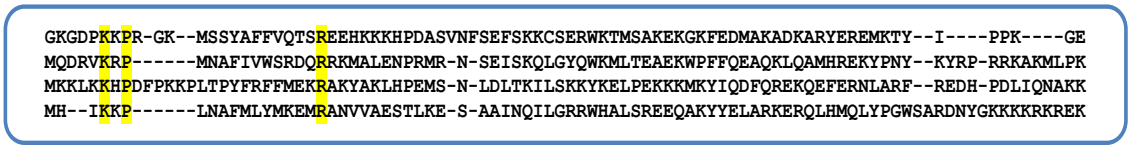


(a) MOCell con SOP. TC y %Non-Gaps

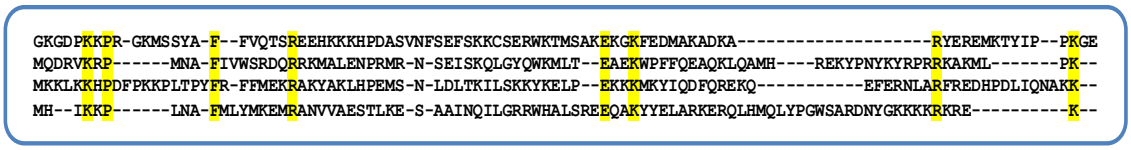


(b) NSGAII con STRIKE. TC y %Non-Gaps

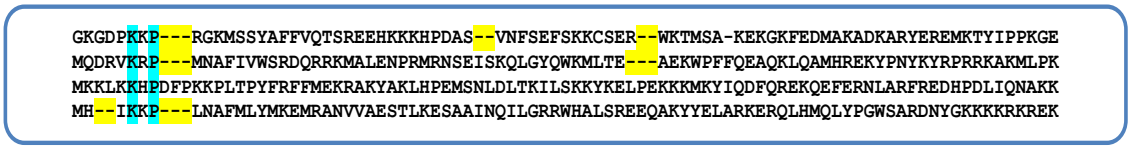
Figura 4.5: Aproximaciones del frente de Pareto obtenida por los algoritmos MOCell (a) y NSGAII (b) resolviendo la instancia BB11001 del BAliBASE 3.0 con una formulación de tres objectivos son: Suma de Pares, TC y Porcentaje de Non-gaps con MOCell y STRIKE, TC y Porcentaje de Non-gaps con NSGAII.



(a) MSA con el mejor Suma de Pares

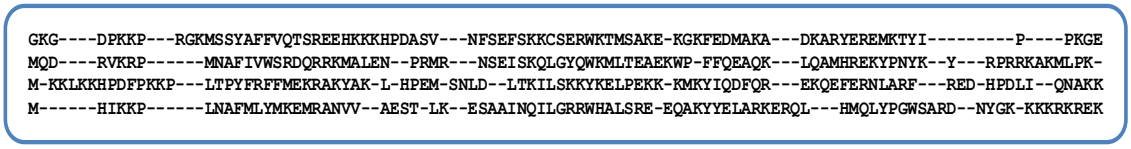


(b) MSA con el mejor TC

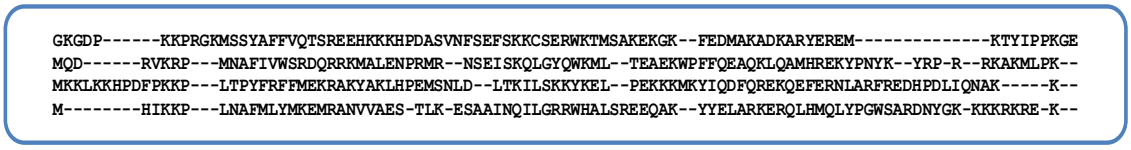


(c) MSA con el mejor %Non-Gaps

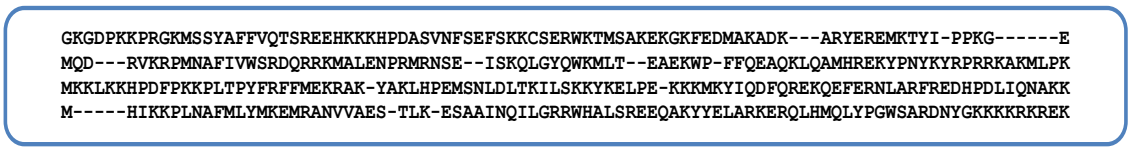
Figura 4.6: Las soluciones de los extremos del frente obtenido (en Figura. 4.5 a) del algoritmo MOCell resolviendo la instancia BB11001 del BAliBASE 3.0.



(a) MSA con el mejor STRIKE

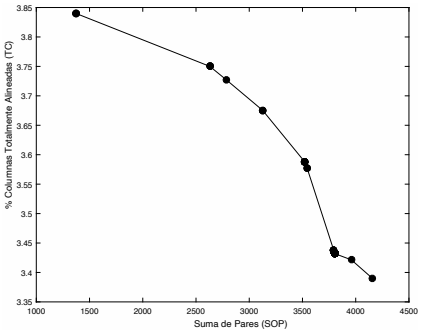


(b) MSA con el mejor TC

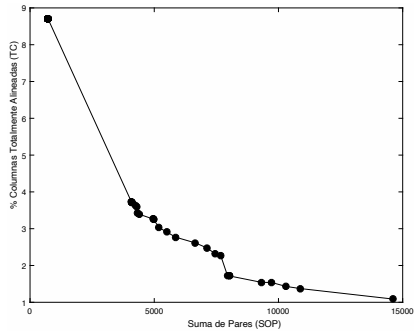


(c) MSA con el mejor %Non-Gaps

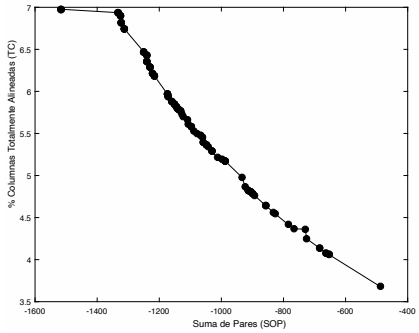
Figura 4.7: Las soluciones de los extremos del frente obtenido (en Figura. 4.5 b) del algoritmo NSGAII resolviendo la instancia BB11001 del BAliBASE 3.0.



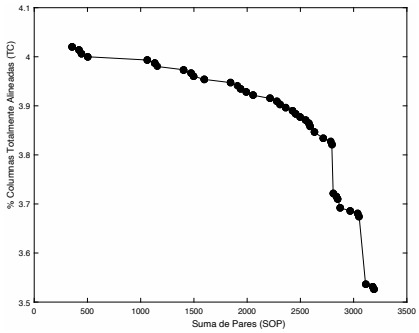
(a) GWASFGA resolviendo instancia BB12001



(b) GWASFGA resolviendo instancia BB12003



(c) NSGAII resolviendo instancia BB11021



(d) NSGAII resolviendo instancia BB12001

Figura 4.8: Aproximaciones del frente de Pareto generadas por los algoritmos GWASFGA y NSGAII resolviendo dos instancias del BAliBASE 3.0. Los objectivos son: La Suma de Pares (SOP) y el porcentaje de Columnas Totalmente Alineadas (TC).

4.6.1.2. Metodología aplicada al estudio

Para este análisis comparativo hemos utilizado tres indicadores de calidad multiobjetivo para evaluar el rendimiento de los algoritmos. El primero es el Hypervolume (I_{HV}) (Zitzler and Thiele, 1999b), que indicador que mide tanto la convergencia como la diversidad de las aproximaciones de los frentes de Pareto, y los indicadores Epsilon aditivo ($I_{\epsilon+}$)) (Zitzler et al, 2003) y Spread Δ (I_Δ) usados como complemento para medir el grado de convergencia y diversidad, respectivamente.

Para cada combinación de los algoritmos y las 218 instancias del BAliBASE hemos realizado 20 ejecuciones independientes, y se reportan la mediana, $\tilde{x}$, y el intervalo intercuartílico, IQR, como medidas de localización (o tendencia central) y dispersión estadística, respectivamente, para cada indicador de calidad considerado. Al presentar los valores obtenidos en las tablas de resultados, destacamos con un fondo gris oscuro el mejor resultado para cada problema y se utiliza un fondo gris claro para indicar el segundo mejor resultado; de esta manera, podemos identificar rápidamente los algoritmos mejor puntuados.

Con el fin de proporcionar resultados estadísticamente confiables, se han aplicado una serie de pruebas estadísticas no paramétricas, ya que en varios casos las distribuciones de resultados no siguieron las condiciones de normalidad y homocedasticidad (Sheskin, 2007). Se ha utilizado un nivel de confianza de 95 % (es decir, nivel de significación de 5 % o p-value inferior a $0,05$) en todos los casos, lo que significa que es improbable que las diferencias hayan ocurrido por casualidad con una probabilidad de 95 %. Por lo tanto, los análisis y comparaciones se centran en toda la distribución de las energías obligatorias, aunque prestan especial atención a los valores de las medianas, para los 218 casos abordados. En particular, se han aplicado el ranking de Friedman y las pruebas multicomparativas post-hoc de Holm (Sheskin, 2007) para conocer qué algoritmos son estadísticamente peores que el algoritmo control (el algoritmo con el mejor ranking).

Para calcular los indicadores de calidad, en particular $I_{\epsilon+}$ e I_Δ, es necesario conocer el frente de Pareto de los problemas. Como son desconocidos para el caso de los problemas de BAliBASE que estamos considerando, hemos adoptado la estrategia de generar un frente de referencia, el cual es el resultado de combinar en un solo frente todas las soluciones no dominadas producidas por todos los algoritmos en todas las ejecuciones independientes para cada instancia de problema BAliBASE. Esta estrategia permite hacer una evaluación relativa del desempeño de las metaheurísticas, porque si el comportamiento de todas las técnicas comparadas es pobre podemos detarminar cuál de ellas produce los mejores frentes, pero no sabemos si están cerca o lejos del verdadero frente de Pareto.

4.6.1.3. Resultados

Los resultados de los experimentos realizados se detallan en esta subsección. Primero analizaremos los resultados de los tres indicadores de calidad multiobjetivo. Las tablas B.1, B.2, B.3, B.4, B.5 y B.6 del apéndice B.1 contienen las medianas y los rangos intercuartílicos del indicador hypervolumen I_{HV} obtenidos por los algoritmos NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA,GWASFGA para la familias RV11, RV12, RV20, RV30, RV40, RV50, respectivamente. Las Tablas B.7, B.8, B.9, B.10, B.11 y B.12 del apéndice B.1 contienen las medianas y los rangos intercuartílicos del indicador Epsilon $I_{\epsilon+}$ obtenidos por los algoritmos NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA,GWASFGA para la familias RV11, RV12, RV20, RV30, RV40, RV50, respectivamente, y finalmente las Tablas B.13, B.14, B.15, B.16, B.17 y B.18 del apéndice B.1 contienen las medianas y los rangos intercuartílicos del indicador Spread I_Δ obtenidos por los algoritmos NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA,GWASFGA para la familias RV11, RV12, RV20, RV30, RV40, RV50, respectivamente. Tenemos que notar que en el caso de I_{HV}, cuanto mayor sea el valor mejor, y lo contrario se aplica en los otros indicadores.

Las cifras I_{HV} de las Tablas B.1, B.2, B.3, B.4, B.5 y B.6 muestran que NSGA-II es la metaheurística que proporciona el mejor rendimiento general, ya que obtiene los mejores valores para

la mayoría de las instancias del BAliBASE. MOCell y GWASFGA son las técnicas que obtienen las segunda mejores aproximaciones del frente de Pareto.

Acerca de los resultados del indicador de calidad I_{E^+}, el cual mide la convergencia de los frentes obtenidos. Los valores, incluidos en las Tablas B.7, B.8, B.9, B.10, B.11 y B.12, muestran que hay tres algoritmos (NSGA-II, MOCell y GWASFGA) con un rendimiento similar si consideramos el número de los mejores y segundo mejores resultados. Como en el caso de los resultados del indicador I_{HV}, SMS-EMO y MOEA/D han mostrado un mal comportamiento frente al resto de algoritmos.

El último indicador de calidad que hemos utilizado, Spread I_Δ el cual mide el grado de diversidad de los frentes, sus resultados se ilustran en las Tablas B.13, B.14, B.15, B.16, B.17 y B.18. En este caso MOCell es claramente el algoritmo que da los mejores valores en la mayoría de los problemas.

A primera vista, puede parecer que los resultados proporcionados por los tres indicadores de calidad son contradictorios, ya que NSGA-II y MOCell serían los primeros clasificados de acuerdo con los indicadores I_{HV} e I_{Delta}, respectivamente, e I_{E+} no permiten declarar un algoritmo ganador. Esto se explica porque hay que considerar que los indicadores utilizan diferentes esquemas para medir la convergencia y la diversidad. I_{HV} se basa en la adición de volúmenes, I_{E+} mide las distancias entre los frentes obtenidos y el frente de Pareto (o el frente de referencia) e I_Δ necesita conocer los puntos extremos del frente de Pareto.

Reportando únicamente los valores de dispersión y tendencia central podríamos estar definiendo conclusiones erróneas, por lo que los resultados han sido estadísticamente probados por los test de Friedman y Holm (detallados en la sub-sección 2.5.5), de los que podemos extraer comentarios ligeramente diferentes. Como se muestra en la tabla 4.2, NSGA-II alcanza el mejor valor de rankeo (Friedman) con Ĩ.65 para el indicador I_{HV}, seguido de MOCell, SPEA2, SMS-EMOA, GWASFGA y MOEA/D. Por lo tanto, NSGA-II se establece como el algoritmo de control para I_{HV} en el test *post-hoc* de Holm, en el que se compara con los demás algoritmos.

Los valores p ajustados ($Holm_{Ap}$ en la Tabla 4.2) resultado la comparativa realizada, nos indican que el algoritmo (MOEA/D) obtiene valores menores al nivel de confianza, lo que significa que NSGA-II es estadísticamente mejor que dicho algoritmo. Sin embargo, esto no afirma que NSGA-II tenga un rendimiento estadísticamente mejor que los demás algoritmos MOCell, SPEA2, SMS-EMOA y GWASFGA.

Para el caso del indicador $I_{\epsilon+}$, se realizó el mismo análisis, el cual indica que NSGA-II es el algoritmo mejor rankeado, convirtiéndose en el algoritmo control con un valor de 2.19, seguido por MOCell, GWASFGA, SPEA2, SMS-EMOA y MOEA/D. Nuevamente, sólo el algoritmo MOEA/D genera un rendimiento estadísticamente peor que el algoritmo de control (NSGA-II), ya que sus valores p ajustados ($Holm_{Ap}$) resultaron ser menores a 0.05 (el nivel de confianza).

Para el tercer indicador I_Δ, el algoritmo mejor rankeado es MOCell (algoritmo control) con un valor de 1.00, seguido por SMS-EMOA, SPEA2, NSGA-II, MOEA/D y GWASFGA. En este caso, MOEA/D y GWASFGA, generan un rendimiento estadísticamente peor que el algoritmo control (MOCell), ya que sus valores p ajustados ($Holm_{Ap}$) resultaron ser menores a 0.05 (el nivel de confianza).

Resumiendo todas las posiciones de ranking (como se muestra en la última fila de la Tabla 4.2), podemos observar que MOCell y NSGA-II muestran el mejor balance global de los tres indicadores de calidad, seguidos por SPEA2, SMS-EMOA, GWASFGA, y MOEA/D. En general, se puede observar que los enfoques clásicos multiobjetivos obtienen mejores resultados que los modernos.

4.6.1.4. Resultados multiobjetivo - Aproximaciones de los frentes de Pareto

Estos resultados son apoyados gráficamente proporcionando algunos ejemplos de los frentes de Pareto aproximados producidos por los algoritmos. Incluimos los frentes con los mejores valores de

Tabla 4.2: Promedio de los rankings de Friedman con los valores p ajustados de Holm ($\alpha = 0{,}05$) de algoritmos comparados para el conjunto de pruebas de instancias RV11 y RV12. El símbolo * indica el algoritmo de control y la columna de la derecha contiene la clasificación general de las posiciones con respecto a los indicadores de calidad I_{HV}, $I_{\epsilon+}$ y I_{Δ}

Hypervolume (I_{HV})			Epsilon ($I_{\epsilon+}$)			Spread I_{Δ}		
Algorithm	Fri_{Rank}	$Holm_{Ap}$	Algorithm	Fri_{Rank}	$Holm_{Ap}$	Algorithm	Fri_{Rank}	$Holm_{Ap}$
*NSGA-II	1.65	-	*NSGA-II	2.19	-	*MOCell	1.00	-
MOCell	3.07	3.71e-01	MOCell	2.75	8.61e-01	SMS-EMOA	2.95	1.45e-01
SPEA2	3.15	3.71e-01	GWASFGA	2.92	8.61e-01	SPEA2	3.00	1.45e-01
SMS-EMOA	3.37	3.71e-01	SPEA2	3.75	3.87e-01	NSGA-II	3.05	1.45e-01
GWASFGA	3.92	7.84e-02	SMS-EMOA	4.05	2.29e-01	MOEA/D	5.17	3.31e-04
MOEA/D	5.82	3.41e-04	MOEA/D	5.32	2.66e-02	GWASFGA	5.82	3.43e-05
Global Ranking: **MOCell (5)**, NSGA-II (6), SPEA2 (10), SMS-EMOA (11), GWASFGA (14), MOEA/D (17)								

I_{HV} en seis instancias de problemas BAliBASE BB12010, BB12024, BB20001, BB30007, BB40010 y BB50013 en las Figuras 4.9, 4.10, 4.11, 4.12, 4.13, 4.14,

4.6.1.5. Conclusiones de la comparativa

Nuestro estudio revela que en el contexto de los algoritmos elegidos, los parámetros adoptados, la metodología de experimentación y los problemas resueltos, MOCell y NSGA-II son las metaheurísticas que proporcionan el mejor desempeño global, mientras que las metaheurísticas modernas como SMS-EMOA, MOEA/D, y GWASF-GA han encontrado dificultades para generar aproximaciones precisas del frente de Pareto.

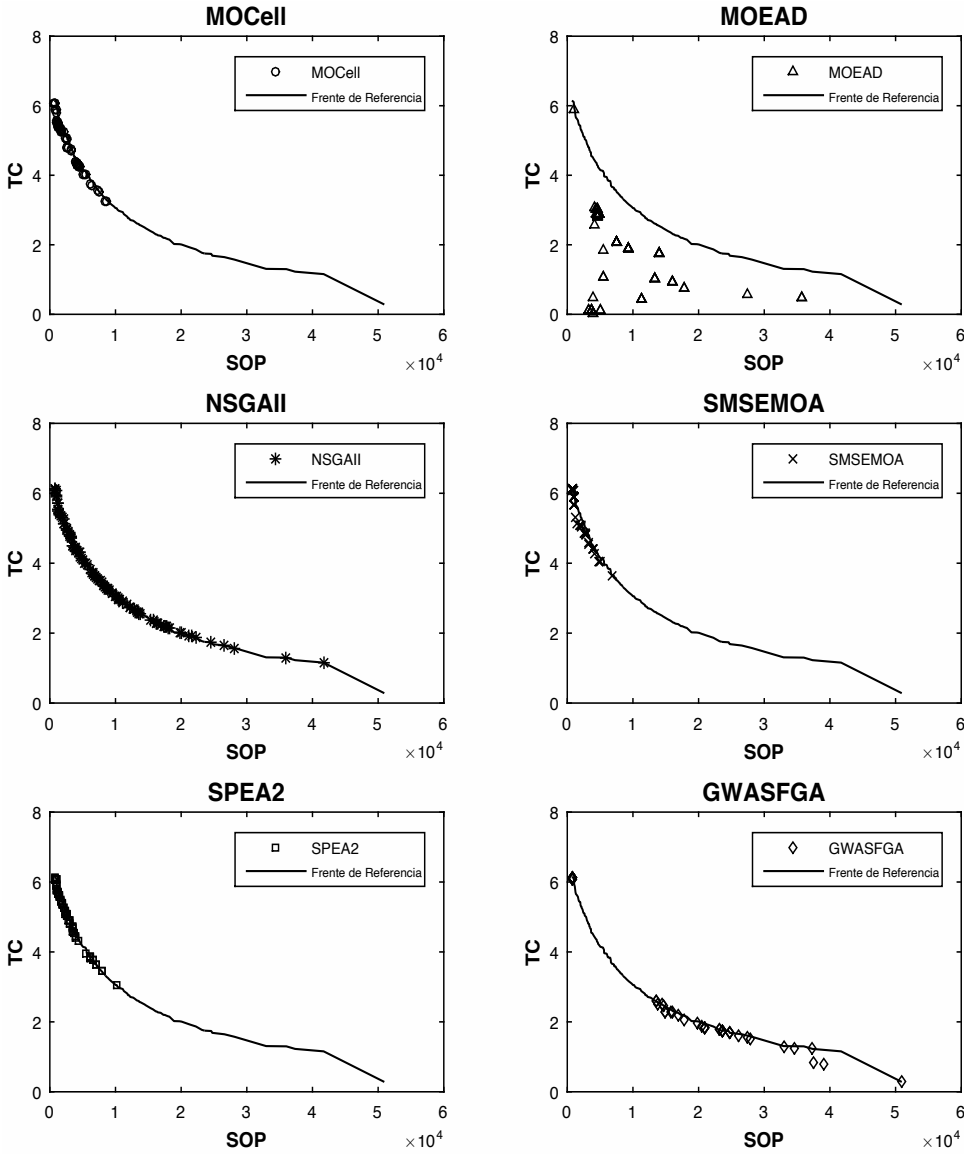


Figura 4.9: Aproximaciones del frente de Pareto con el mejor I_{HV} obtenido de todos los algoritmos (MOCell, MOEA/D, NSGAII, SMSEMOA, SPEA2 y GWASFGA) sobre la instancia BAliBASE BB12010 en 20 ejecuciones independentes.

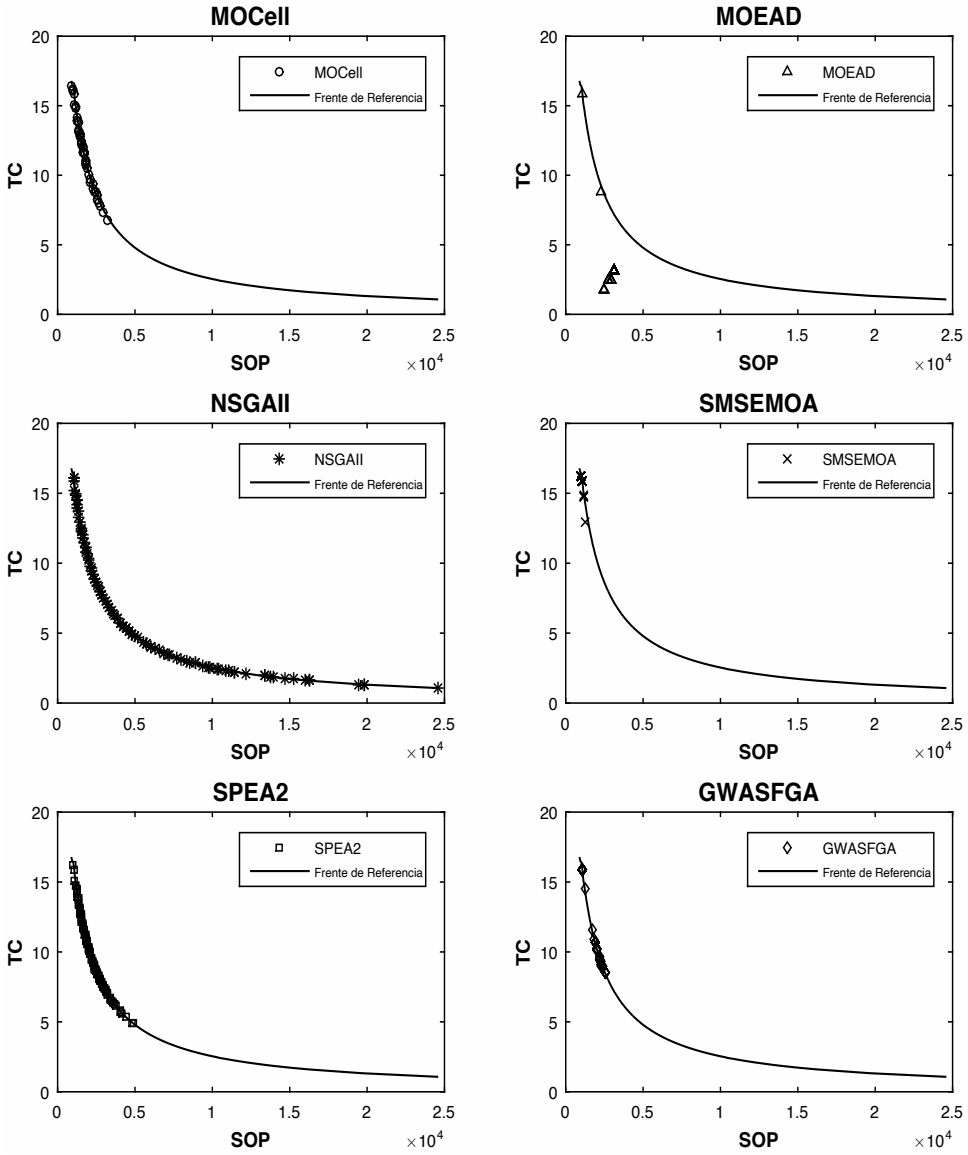


Figura 4.10: Aproximaciones del frente de Pareto con el mejor I_{HV} obtenido de todos los algoritmos (MOCell, MOEA/D, NSGAII, SMSEMOA, SPEA2 y GWASFGA) sobre la instancia BAliBASE BB12024 en 20 ejecuciones independentes.

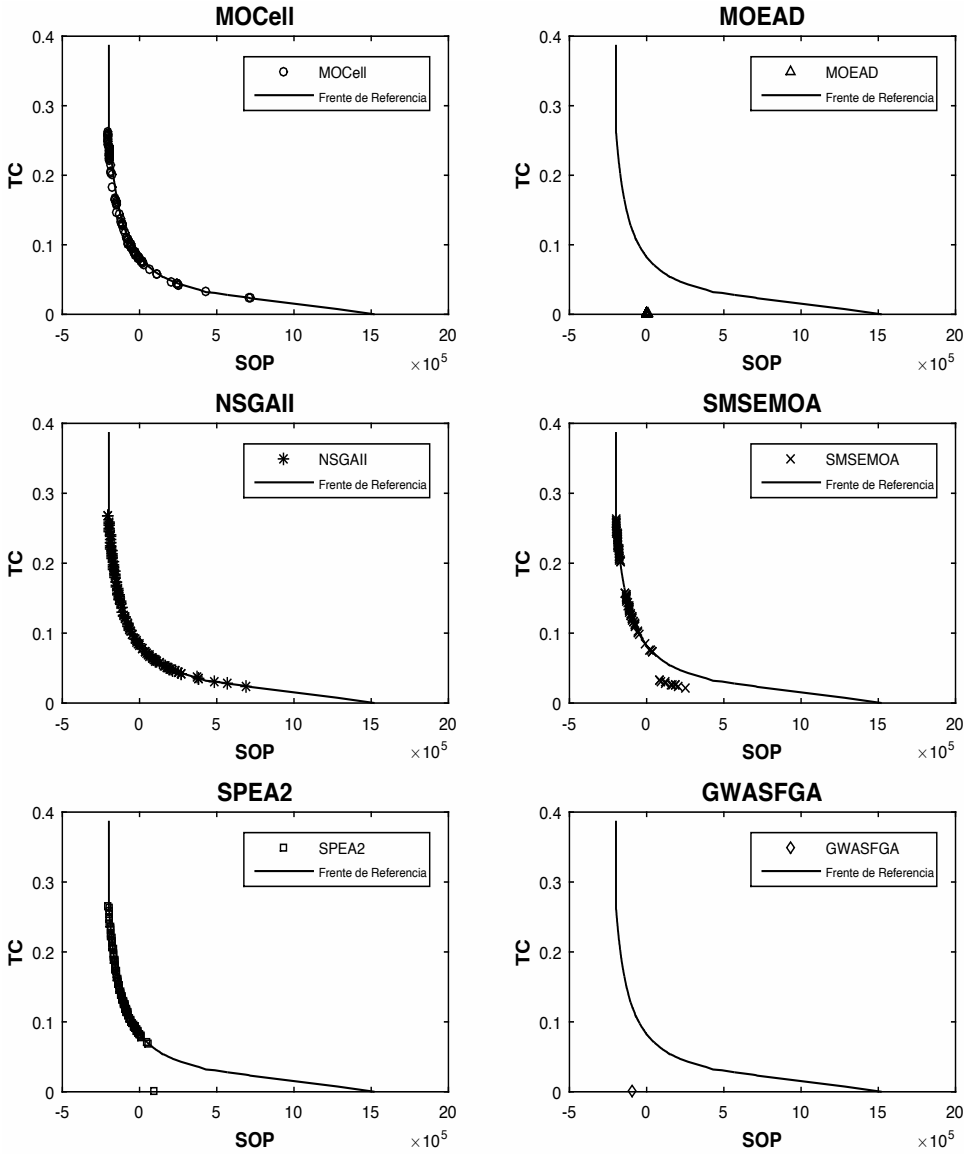


Figura 4.11: Aproximaciones del frente de Pareto con el mejor I_{HV} obtenido de todos los algoritmos (MOCell, MOEA/D, NSGAII, SMSEMOA, SPEA2 y GWASFGA) sobre la instancia BAliBASE BB20001 en 20 ejecuciones independentes.

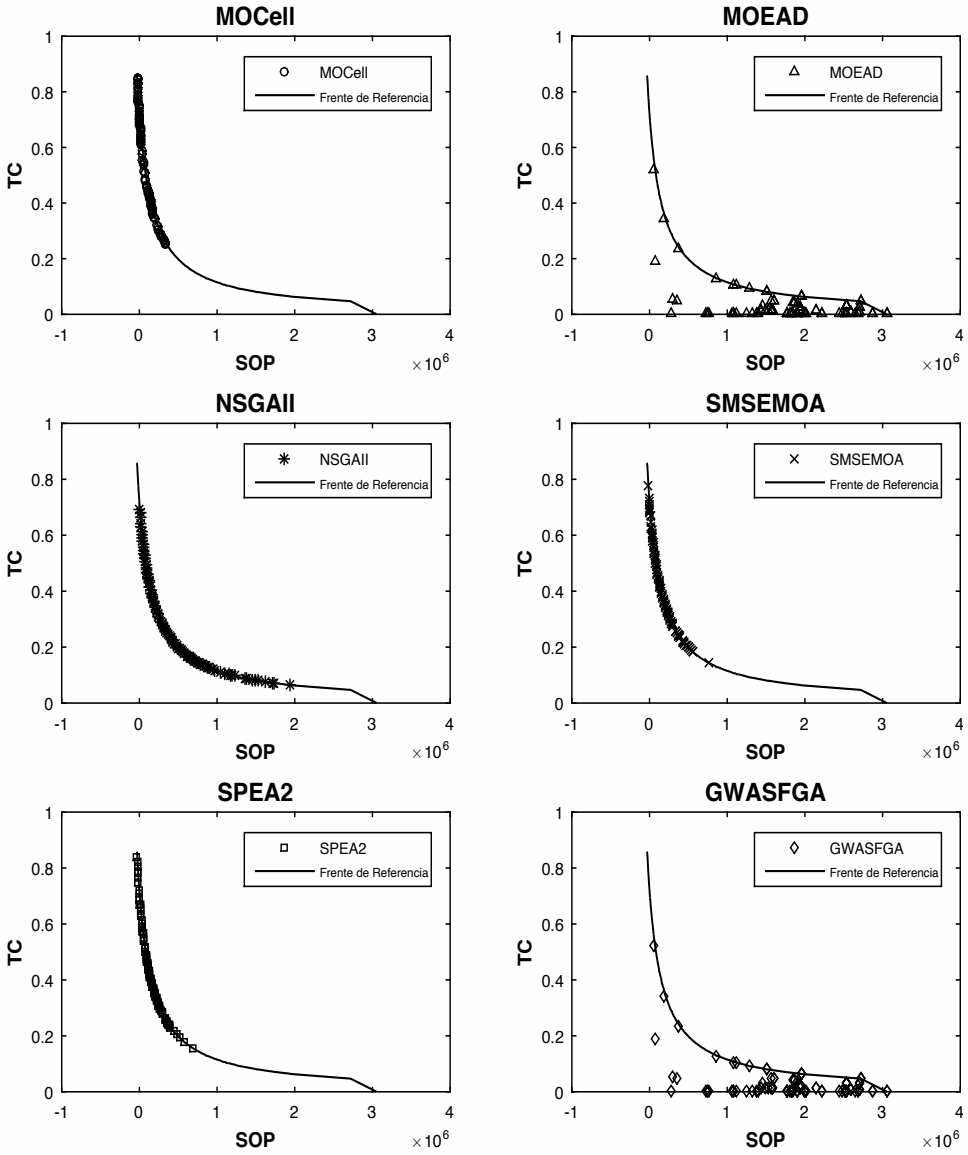


Figura 4.12: Aproximaciones del frente de Pareto con el mejor I_{HV} obtenido de todos los algoritmos (MOCell, MOEA/D, NSGAII, SMSEMOA, SPEA2 y GWASFGA) sobre la instancia BAliBASE BB30007 en 20 ejecuciones independentes.

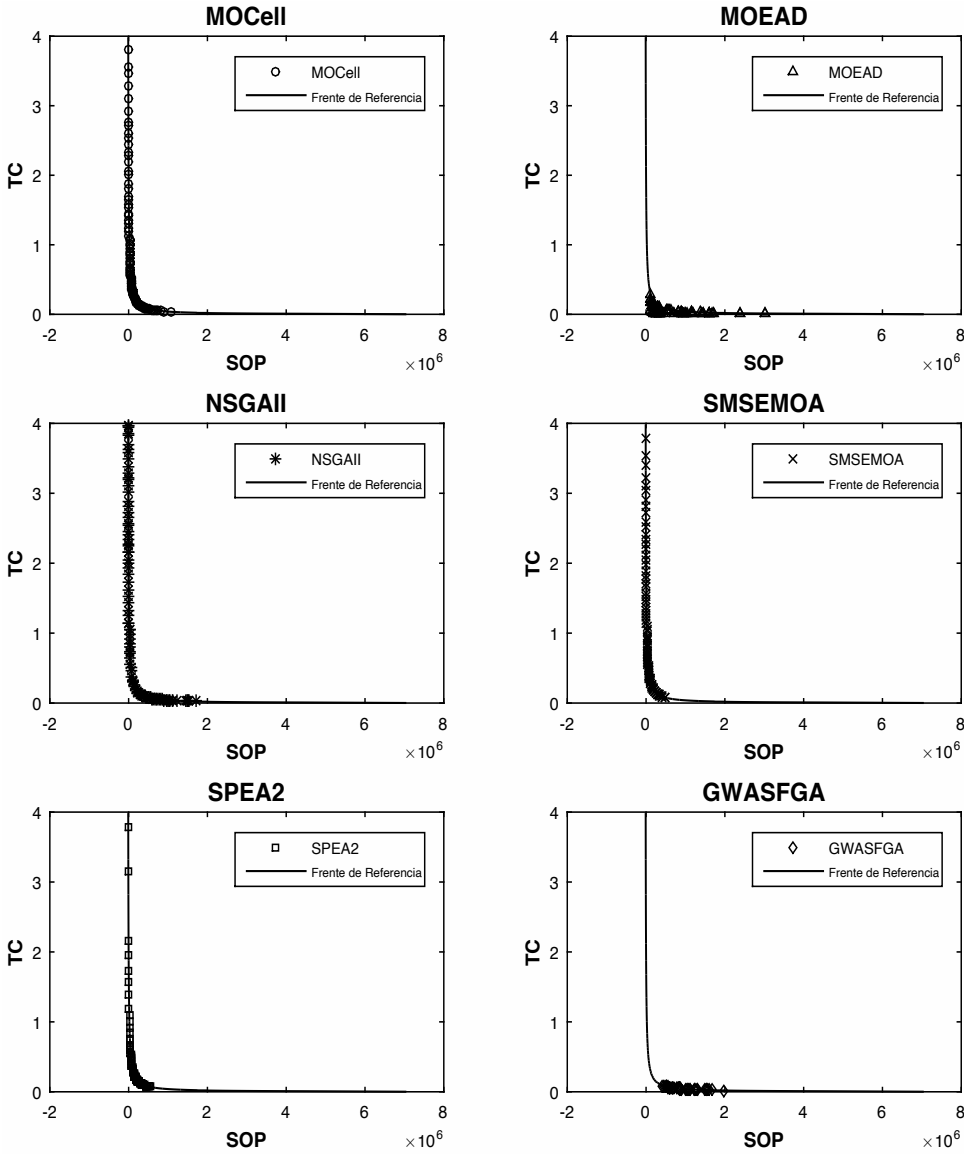


Figura 4.13: Aproximaciones del frente de Pareto con el mejor I_{HV} obtenido de todos los algoritmos (MOCell, MOEA/D, NSGAII, SMSEMOA, SPEA2 y GWASFGA) sobre la instancia BAliBASE BB40010 en 20 ejecuciones independentes.

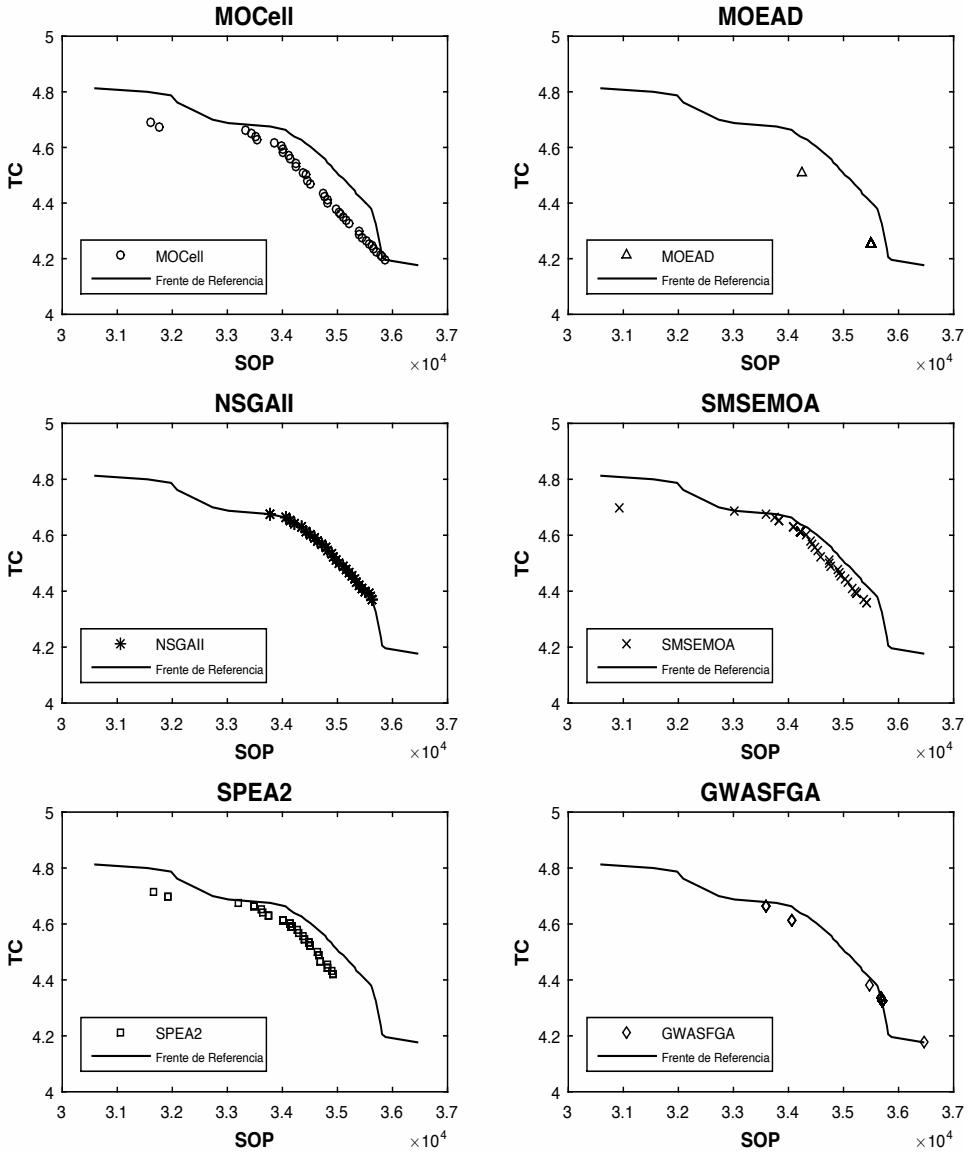


Figura 4.14: Aproximaciones del frente de Pareto con el mejor I_{HV} obtenido de todos los algoritmos (MOCell, MOEA/D, NSGAII, SMSEMOA, SPEA2 y GWASFGA) sobre la instancia BAliBASE BB50013 en 20 ejecuciones independentes.

```
s1: G------ERSLAA--TLV-     s1: --G----ERSLAA--TLV-
s2: NAILAH-ER-------LSI     s2: NAI-LAHER-------LSI
s3: NGYLFI-E---Q----L-N     s3: -NGYLFIE----Q---LN-
s4: GLVSDVFEARH--MQRL--     s4: GLVSDVFE-ARH-MQRL--
```

Figura 4.15: Ejemplo de dos soluciones MSA que tienen la misma puntuación de TC y Non-gaps. Ambas soluciones tienen la misma longitud y dos columnas totalmente alineadas (E and L).

4.6.2. Formulación de tres objetivos

Para este segundo análisis hemos considerado una formulación de tres objetivos al problema (como en Ortuño et al (2013)). Los objetivos a optimizar son: El porcentaje de columnas totalmente alineadas (TC), el porcentaje de no-gaps (Non-Gaps), y la métrica STRIKE. Todos ellos deben ser maximizados, y son descritos en la subsección 4.2.2 de Métricas de Calidad MSA.

Tenemos que resaltar que los objetivos TC y non-Gaps se caracterizan por tener un espacio de búsqueda con un elevado número de "plateaus" (diferentes soluciones con los mismos valores objetivos), lo que introducirá un nivel de complejidad adicional para los software de alineamiento multiobjetivo. Ilustramos este problema con un ejemplo. En la Figura. 4.15 incluimos dos alineamientos que tienen la misma longitud (19 columnas) y dos columnas totalmente alineadas, por lo que tienen el mismo valor TC (10.5 %). Además, las soluciones MSA de igual longitud tienen la misma puntuación non-Gaps.

Para este análisis hemos considerado cuatro optimizadores multi-objetivo: NSGA-II, MOCell, NSGA-III y GWASF-GA. Hemos elegido NSGA-II ya que es el algoritmo de mayor referencia. MOCell ha demostrado un desempeño notable en aplicaciones bi-objetivo, y estamos interesados en evaluar sus capacidades de búsqueda con un problema de tres objetivos como MSA usando aquí una variante con un archivo basado en el hipervolumen. Y finalmente escogimos a los algoritmos NSGA-III y GWASF-GA porque son capaces de resolver problemas con muchos objetivos, por lo que también estamos interesados en evaluar sus resultados en el contexto de MSA de tres objetivos.

4.6.2.1. Configuración de los parámetros

Todas las metaheurísticas consideradas en esta comparativa son algoritmos evolutivos, y comparten operadores de codificación y variación similares. Los operadores de cruce y mutación son los mismos que los aplicados en la comparativa bi-objetivo (véase Subsección 4.6.1.1): Cruce de un solo punto y Mutación de desplazamiento de grupo de gaps.

El método para generar la población inicial también ha sido tomado del software jMetalMSA 4.5. En lugar de crear soluciones aleatorias, se han obtenido varias soluciones pre-computadas con un conjunto de herramientas externas de MSA. Entonces, estas soluciones son insertadas en la población y luego se recombinan aplicando el operador de cruce para completar el resto de soluciones. En la subsección 4.5.5 se incluyen mas detalles.

El tamaño de la población de todos los algoritmos, es de 100. En NSGA-III, el tamaño de la población no se puede establecer libremente, por lo que en el caso de un problema con tres objetivos el tamaño es de 91 soluciones. El archivo externo en MOCell tiene también una capacidad de 100. La condición de parada en todos los casos es computar 50.000 evaluaciones.

4.6.2.2. Metodología experimental

Para evaluar el rendimiento de los algoritmos multiobjetivo hemos usado los indicadores Epsilon $I_{\epsilon+}$) y la distancia generacional invertida - (*Inverted Generational Distance* / o I_{IGD+}), que considera tanto convergencia como diversidad. Hemos elegido este último en lugar del hipervolumen

porque el tamaño de población diferente de NSGA-III haría una comparación injusta con el resto de algoritmos, ya que el hipervolumen está compuesto por la suma de los volúmenes de todas las soluciones encontradas con respecto a un punto de referencia, de modo que cuanto mayor sea el número de soluciones, mayor será la probabilidad de obtener un valor de Hypervolume más alto. Al igual que en casos anteriores, se ha generado un frente de referencia combinando todas las soluciones no dominadas computadas en todas las ejecuciones de todos los algoritmos.

Para cada combinación de algoritmo y para cada instancia de problema BAliBASE, hemos realizado 25 ejecuciones independientes. A partir de estas ejecuciones hemos calculado la mediana y el intervalo intercuartílico (IQR) como medidas de tendencia central y dispersión estadística, respectivamente. Se han abordado un total de 140 casos de MSA de 5 familias de problemas diferentes (RV11, RV12, RV30, RV40 y RV50), con el fin de asegurar conclusiones generales y no sesgadas.

4.6.2.3. Resultados multiobjetivo - indicadores de calidad

Las tablas B.19, B.21, B.23, B.25, B.27 y B.20, B.22, B.24, B.26, B.28 del Apéndice B.2 contienen las medianas y el rango intercuartil de las distribuciones resultantes de los indicadores I_{e+} e I_{IGD+} (de 25 ejecuciones independientes), para cada algoritmo y para cada familia de instancias BAliBASE, respectivamente. Se han abordado un total de 140 casos diferentes de MSA de 5 familias problemáticas (RV-11, RV-12, RV-30, RV-40 y RV-50), lo que nos lleva a sugerir que las conclusiones reportadas son imparciales y suficientemente generales.

Con el fin de proporcionar resultados estadísticamente confiables (en este estudio $\alpha = 0,05$), hemos aplicado una serie de pruebas estadísticas no paramétricas, ya que en varios casos las distribuciones de resultados no siguieron las condiciones de normalidad y Homoscedasticidad (Sheskin, 2007). Por lo tanto, los análisis y las comparaciones se centran en la distribución completa de cada una de las dos métricas estudiadas (I_{e+} e I_{IGD+}). Específicamente, hemos aplicado el ranking de Friedman y las pruebas multicomparativas post-hoc de Holm (Derrac et al, 2011) para saber qué algoritmos son estadísticamente peores que el control (con la mejor clasificación).

En este sentido, la Tabla 4.3 resume los resultados de la aplicación de las pruebas estadísticas por separado para cada familia de problemas BAliBASE y para los dos indicadores de calidad considerados en la experimentación. La última columna contiene una clasificación general que suma las posiciones de los algoritmos para cada categoría de problema. Como se muestra en esta tabla, NSGA-II es la técnica mejor clasificada seguida por MOCell y NSGA-III para prácticamente todas las familias de problemas BAliBASE, con excepción de RV30, para lo cual GWASF-GA es el algoritmo segundo mejor clasificado. Por lo tanto NSGA-II se establece como el algoritmo de control en la prueba post-hoc de Holm para ser comparado con las técnicas restantes.

Vale la pena señalar que, en la mayoría de estas comparaciones, los valores p ajustados con respecto al procedimiento $Holm_{Ap}$ son inferiores al nivel de confianza (0.05), lo que significa que el NSGA-II es estadísticamente mejor que MOCell, NSGA-III, y GWASF-GA, en el contexto del procedimiento experimental seguido aquí. Existen varias excepciones en el caso de problemas de la familia RV50, para los cuales no se calculan diferencias estadísticas para los algoritmos NSGA-II y MOCell, sino para GWASF-GA. Probablemente, el corto número de instancias de problemas MSA de la familia RV50, con respecto a las otras familias, hace que la prueba estadística pierda la precisión cuando se calculan diferencias significativas de MOCell y NSGA-III en comparación con NSGA-II, rechazando así la hipótesis nula.

Una segunda observación, aún interesante, es que para la mayoría de las instancias BAliBASE, NSGA-III muestra las soluciones segunda mejor ubicadas para el indicador I_{IGD+}, mientras que MOCell hace lo mismo para el indicador I_{e+}. Esto nos lleva a sugerir que NSGA-III es capaz de proceder con la convergencia y la diversidad precisa, aunque MOCell es un buen candidato para tratar con el problema del MSA en términos de convergencia. En el caso de GWASF-GA,

Tabla 4.3: Promedio de los rankings de Friedman con los valores p ajustados de Holm (0,05) de los algoritmos comparados para el conjunto de pruebas de instancias BAliBASE. El símbolo * indica el algoritmo de control y la columna de la derecha contiene la clasificación general de las posiciones con respecto a I_{IGD+} y $I_{\epsilon+}$.

Problem	Inverted Generational Distance (I_{IGD+})			Epsilon ($I_{\epsilon+}$)			Overall	
Family	Algorithm	Fri_{Rank}	$Holm_{Ap}$	Algorithm	Fri_{Rank}	$Holm_{Ap}$	Algorithm	Rank
RV11	***NSGA-II**	**1.09**	-	***NSGA-II**	**1.23**	-	**NSGA-II**	**2**
	NSGA-III	2.56	5.34e-06	MOCell	2.20	2.68e-03	MOCell	5
	MOCell	2.78	3.41e-07	NSGA-III	2.77	3.28e-06	NSGAIII	5
	GWASF-GA	3.36	6.06e-14	GWASF-GA	3.78	8.98e-15	GWASF-GA	8
RV12	***NSGA-II**	**1.54**	-	***NSGA-II**	**1.12**	-	**NSGA-II**	**2**
	MOCell	2.09	6.49e-02	MOCell	2.51	2.52e-06	MOCell	2
	NSGA-III	2.52	2.02e-03	NSGA-III	2.63	9.18e-07	NSGA-III	6
	GWASF-GA	3.83	5.85e-14	GWASF-GA	3.72	1.09e-17	GWASF-GA	8
RV30	***NSGA-II**	**1.47**	-	***NSGA-II**	**1.59**	-	**NSGA-II**	**2**
	GWASF-GA	2.63	2.91e-03	MOCell	2.43	3.07e-02	GWASF-GA	5
	NSGA-III	2.90	4.69e-04	GWASF-GA	2.74	5.80e-03	MOCell	6
	MOCell	2.97	3.49e-04	NSGA-III	3.22	7.87e-05	NSGA-III	7
RV40	***NSGA-II**	**1.08**	-	***NSGA-II**	**1.45**	-	**NSGA-II**	**2**
	NSGA-III	2.54	2.86e-07	MOCell	2.12	1.86e-02	MOCell	5
	MOCell	2.65	6.88e-08	NSGA-III	2.54	2.36e-04	NSGA-III	5
	GWASF-GA	3.70	1.11e-19	GWASF-GA	3.87	5.16e-17	GWASF-GA	8
RV50	***NSGA-II**	**1.30**	-	***NSGA-II**	**1.57**	-	**NSGA-II**	**2**
	NSGA-III	2.30	6.68e-02	MOCell	1.80	6.48e-01	MOCell	5
	MOCell	2.38	6.68e-02	NSGA-III	2.69	5.52e-02	NSGA-III	5
	GWASF-GA	3.99	3.16e-07	GWASF-GA	3.92	1.07e-05	GWASF-GA	8

muestra un comportamiento deficiente para casi todas las familias de problemas BAliBASE, pero para RV30, se clasifica como el segundo mejor algoritmo con respecto al indicador I_{IGD+}. En resumen, NSGA-II muestra el mejor balance global para los dos indicadores de calidad, seguido por NSGA-III, MOCell y GWASF-GA.

4.6.2.4. Resultados multiobjetivo - frentes de pareto

Para ilustrar los resultados obtenidos en los experimentos en las Figuras 4.16, 4.17, 4.18 se muestra la contribución de las metaheurísticas comparadas del frente de referencia de las instancias BB11001, BB12024 y BB50001 del BAliBASE, respectivamente.

4.6.2.5. Discusión de los resultados

Las Figuras 4.16, 4.17, y 4.18 muestran la contribución de cada uno de los algoritmos en estudio al frente de Pareto de referencia para tres instancias BAliBASE seleccionadas: BB11001 (RV11), BB12024 (RV12) y BB50001 (RV50). Si observamos la Figura 4.16, podemos observar que las soluciones de NSGA-II (los puntos con el símbolo +) se concentran principalmente en la región de la mano derecha, lo que corresponde a valores altos de la función STRIKE, mientras que NSGA-III y MOCell proporcionan soluciones con mejores valores TC. Esto trae una discusión interesante en el sentido de que la evaluación del desempeño mediante el uso de los indicadores de calidad multiobjetivos clásicos podría no coincidir con lo acordado por los biólogos, quienes deciden qué algoritmo escoger; esto podría suceder, por ejemplo, si ellos prefieren alineamientos con puntuaciones más altas de TC.

Para hacer la discusión más complicada, podemos ver que los puntos de la Figura. 4.17 están claramente divididos en dos partes: las de GWASF-GA (representadas por cuadrados) se agrupan en la esquina donde el % de Non-Gaps y STRIKE tienen los valores más altos, donde el resto producido por los otros algoritmos, incluyendo NSGA-II, se dispersan hacia la región del mejor TC. Sin embargo, las soluciones de NSGA-II en la Figura. 4.18 se concentran en una estrecha

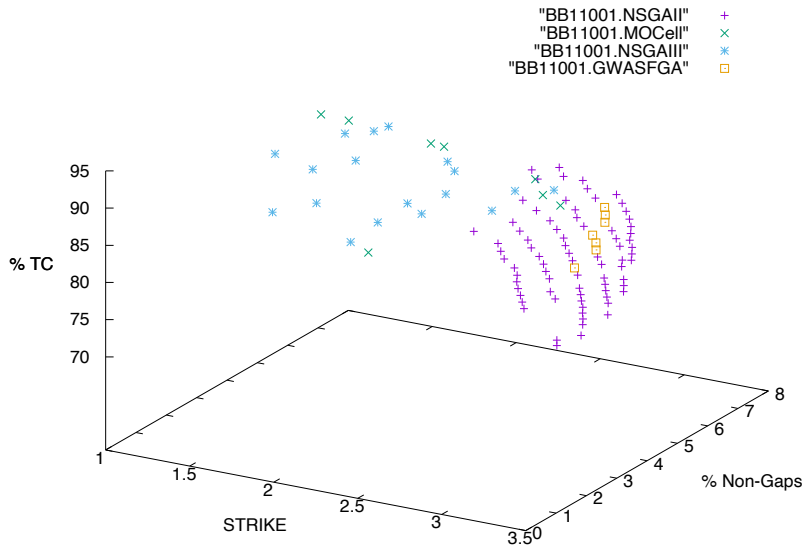


Figura 4.16: Contribución de las metaheurísticas comparadas del frente de referencia del problema BAliBASE BB11001.

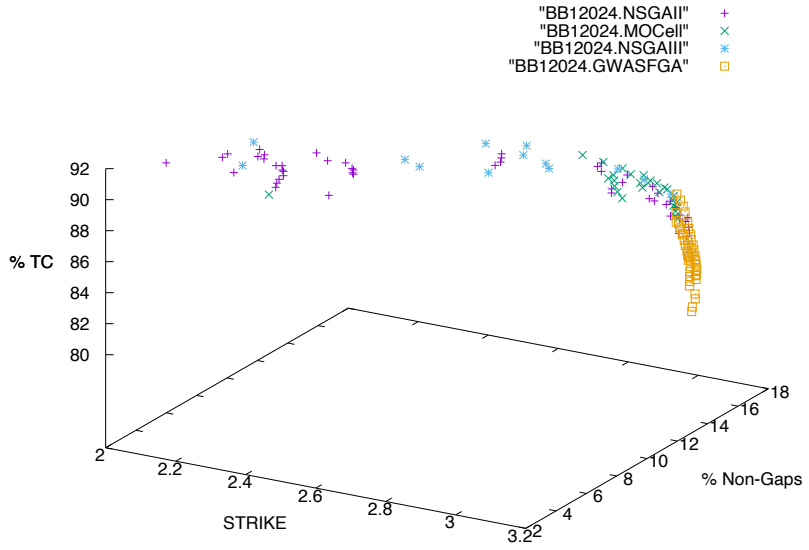


Figura 4.17: Contribución de las metaheurísticas comparadas del frente de referencia del problema BAliBASE BB12024.

región del frente de Pareto de referencia. Tenemos que recordar que NSGA-II logra los mejores

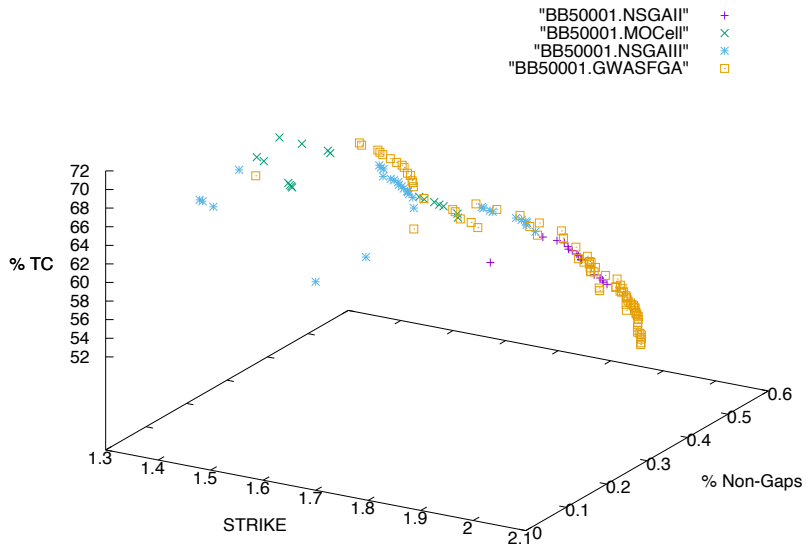


Figura 4.18: Contribución de las metaheurísticas comparadas del frente de referencia del problema BAliBASE BB50001.

indicadores de calidad en estos problemas.

En nuestro estudio hemos aplicado la misma metodología de experimentación utilizada en trabajos anteriores, donde la condición de parada es la misma en todos los problemas. El análisis de la contribución de los algoritmos de los frentes de referencia indica que los 'fitness landscapes' de los problemas del BAliBASE pueden ser muy diferentes, por lo que el ajuste de parámetros de los algoritmos tal vez debería ajustarse por problema particular en lugar de utilizar una configuración común en todas las situaciones.

4.6.2.6. Conclusiones de la comparativa 3D

Las principales conclusiones de este análisis comparativo de la formulación de tres objetivos al problema MSA, se pueden esbozar de la siguiente manera:

1. Incluir la puntuación STRIKE como un tercer objetivo para resolver el MSA es útil para evitar "plateaus" habituales en el espacio de búsqueda cuando solo se usa una formulación de dos objetivos, como Porcentaje de Columnas totalmente alineadas y Non-Gaps.

2. NSGA-II proporciona el mejor rendimiento general según los dos indicadores de calidad utilizados en este estudio y para las instancias BAliBASE.

3. NSGA-III y MOCell son capaces de mostrar un comportamiento exitoso al tratar con los problemas de la familia RV50. MOCell también es un optimizador preciso en el ámbito de las instancias de la familia RV12.

4. El análisis de la contribución de los algoritmos a los frentes de Pareto de referencia de tres problemas indica que el desempeño cualitativo puede ser difícil de lograr sólo observando las cifras del indicador de calidad.

4.7. Propuesta algorítmica: M2Align

El alineamiento múltiple de secuencias, aun a pesar de usar metaheurísticas, puede requerir una cantidad considerable de tiempo de cómputo, sobre todo cuando hay que hacer miles de evaluaciones y cada una de éstas requiere un tiempo mayor de unos pocos segundos. Dado a que los biólogos frecuentemente necesitan calcular alineamientos para conjuntos de secuencias cada vez a mayor escala, la reducción de los tiempos de ejecución de las técnicas es de gran importancia en la actualidad, por lo que uno de los enfoques planteados en esta Tesis Doctoral es aplicar paralelismo para aprovechar las ventajas que proveen modernos equipos computacionales multi-núcleo y así lograr obtener resultados de igual o mejor calidad pero en un menor tiempo posible.

En esta sección hacemos una propuesta que radica en este contexto. En concreto, hemos implementado el software **M2Align**, una versión más rápida y eficiente del algoritmo MO-SAStrE (Ortuño et al, 2013), capaz de resolver el problema de MSA en paralelo.

4.7.1. Funcionalidades de M2Align

Entre las funcionalidades compartidas entre M2Align y MO-SAStrE están las siguientes características:

- Está basado en el algoritmo evolutivo NSGA-II (Non-dominated Sorting Evolutionary Algorithm) (Deb et al, 2002b).
- Los operadores genéticos empleados son Cruce de un sólo punto y Mutación por desplazamiento de grupos de gaps.
- Los objetivos a optimizar son: la Métrica basada en información estructural STRIKE, el porcentaje de columnas totalmente alineadas y el porcentaje de Non-Gaps.
- La población inicial del algoritmo NSGA-II se genera a partir de alineamientos pre-computados con herramientas representativas del estado del arte MSA: ClustalW, MUSCLE, Kalign, Mafft, RetAlign, TCOFFEE, ProbCons y FSA.

Pero además de ellas nuestra propuesta algorítmica incorpora las siguientes características:

- Escrito en el lenguaje de programación Java (MO-SAStrE está implementado en Matlab), por lo que puede ejecutarse en cualquier computadora que tenga instalado un Java JDK.
- Puede ejecutar en paralelo en sistemas multi-core.
- La codificación de las soluciones, en lugar de almacenar matrices de números enteros, está basada en la representación presentada en (Rubio-Largo et al, 2015), en la que sólo se almacena la información de los gaps. Como consecuencia, las exigencias de memoria se reducen significativamente y los ejecuciones de los operadores genéticos (cruce y mutación) son muchos más rápidas y e implementadas de manera más eficiente.
- Si la información estructural provistas por las base de datos PDB's no están disponibles, M2Align proporciona métricas como la Suma de Pares (SOP) y la Suma de Pares ponderadas con penalidad de gaps afines(wSOP) como alternativas a STRIKE.
- M2Align es un proyecto Open Source alojado en el repositorio público GitHub[2], facilitando así su uso a usuarios interesados. La información sobre cómo descargarlo, compilarlo y ejecutarlo con problemas de pruebas se incluye en el sitio del proyecto.

[2]https://github.com/KhaosResearch/M2Align

4.7.2. Detalles de la implementación

M2Align ha sido desarrollado utilizando el software jMetalMSA (véase sección 4.7). La arquitectura orientada a objetos de jMetalMSA ha permitido reutilizar la implementación del algoritimo NSGA-II, base de MO-SAStrE y de M2Align. Emplea el esquema de codificación de las soluciones MSA basadas en la especificación de los grupos de gaps (subsección 4.5.3), los operadores genéticos de cruce de un sólo punto y mutación de desplazamiento de grupo de gaps (detallados en la subsección 4.5.4). Emplea las métricas de calidad implementadas en jMetalMSA usando las clases *StrikeScore*, *PercentageOfAlignedColumnsScore* y *PercentageOfNonGapsScore*;

4.7.3. Enfoque paralelo

Como se comentó anteriormente, una estrategia natural para paralelizar NSGA-II es ejecutar al mismo tiempo todas las funciones de evaluación de las nuevas soluciones creadas. Sin embargo, el resto de los pasos se seguirán ejecutando secuencialmente, por lo que la relación entre las ejecuciones paralelas y secuenciales debe ser claramente favorable a la primera para obtener reducciones de tiempo significativas.

Por esta razón, hemos realizado un estudio previo para identificar aquellas operaciones que contribuyen de manera significativa al tiempo de ejecución global del algoritmo. Para este propósito, la Figura 4.19 muestra los perfiles de tiempo serial promedio para el algoritmo NSGA-II resolviendo la instancia BAliBASE BB20001. En esta figura se detallan los porcentajes de tiempo utilizados en las siguientes operaciones del algoritmo: evaluación de la solución, reemplazo, reproducción (dado un conjunto de padres seleccionados, operadores de cruce y mutación se aplican para generar nuevos individuos), creación de la población inicial, y el operador de selección.

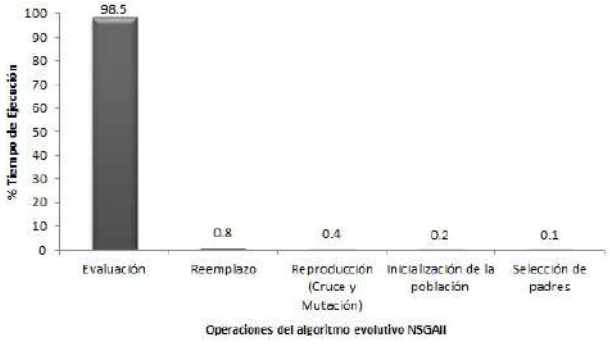


Figura 4.19: Perfil de Tiempo de ejecución secuencial

Estas cifras indican que el enfoque paralelo utilizado en M2Align, evaluando las soluciones en paralelo, es totalmente justificado, ya que la función de evaluación es la tarea que más tiempo consume (98.5 %) de la ejecución global del algoritmo.

El esquema de paralelización también está basado en el de jMetal. La evaluación de las soluciones de la población de una metaheurística es un procedimiento autónomo en la mayoría de las metaheurísticas de jMetal (es decir, tienen un método `evaluationPopulation (List <Solution>)`, que incluye una instancia de una clase llamada `Evaluator`, siendo este objeto el responsable de evaluar todas las soluciones. La idea es utilizar un evaluador de múltiples hilos que sea capaz de realizar todas las evaluaciones de las soluciones en paralelo en equipos basados en varios núcleos (multi-core).

Además las ventajas de este esquema son: primero, no requieren ningún cambio en el algoritmo original y, segundo, el comportamiento de NSGA-II permanece sin cambios.

4.7.4. Resultados y discusión del rendimiento

Para evaluar el rendimiento de M2Align hemos elegido una vez más el Benchmark Alignment dataBASE (BALiBASE v3.0) Los experimentos se han llevado a cabo a través de un sistema multi-núcleo compuesto de 20 núcleos. M2Align ha sido configurado de forma similar a MO-SAStrE, con un tamaño de población de 100 soluciones y un total de 50.000 evaluaciones. El operador de mutación y crossover se definen con una probabilidad de ejecución de 0.80 y 0.20, respectivamente. La población inicial se ha generado utilizando los conjuntos de alineamientos pre-computados por las técnicas MSA especificadas en la Tabla 4.1. La arquitectura seleccionada para realizar estos experimentos fue un CPU Intel Xeon de 2 procesadores E5-2650 v3 de 10 núcleos a 2.3GHz y 25 MB de caché ejecutando CentOS Linux 7.

4.7.4.1. Rendimiento paralelo

Para evaluar el rendimiento paralelo de M2Align, hemos utilizado dos métricas del tema bien conocidas: Speed-Up y la Eficiencia. El Speed-Up está definido por la división entre el tiempo secuencial para el tiempo paralelo, por lo que si el número de unidades de procesamiento (por ejemplo, núcleos) es P, este valor debería ser la reducción de tiempo ideal a lograr. Relacionado al Speed-Up, la métrica de eficiencia se calcula dividiendo el Speed-Up por el número de núcleos utilizados; Por lo tanto, una eficiencia del 100 % indica una aceleración de P.

La Tabla 4.4 muestra el tiempo de ejecución total, el Speed-Up y la eficiencia de M2Align resolviendo los seis grupos de familias contenidas en el BAliBASE v3.0, utilizando 1 (ejecución secuencial), 4, 10 y 20 núcleos.

Tabla 4.4: Evaluación del rendimiento paralelo de M2Align (los tiempos están expresado en horas) sobre las 218 instancias de problemas del BAliBASE v3.0 agrupadas en familias RV11, Rv12, RV20, Rv30, Rv40 y RV50. SR: Tiempo de ejecución sequencial, SU: Speed-up (SR dividido para el número de núcleos), Eff: Efficiencia (SU dividido por el número de núcleos).

Familia	SR	4 Núcleos			10 Núcleos			20 Núcleos		
		Tiempo	SU	Eff	Tiempo	SU	Eff	Tiempo	SU	Eff
RV11	9.05	2.54	3.56	89 %	1.27	7.15	72 %	0.99	9.12	46 %
RV12	4.58	1.34	3.43	86 %	0.73	6.29	63 %	0.63	7.22	36 %
RV20	7.78	2.30	3.38	84 %	1.20	6.49	65 %	0.94	8.25	41 %
RV30	6.55	1.88	3.48	87 %	0.99	6.61	66 %	0.79	8.28	41 %
RV40	9.14	2.70	3.39	85 %	1.50	6.11	61 %	1.25	7.33	37 %
RV50	5.19	1.46	3.54	89 %	0.74	7.03	70 %	0.59	8.77	44 %

Podemos observar que se pueden obtener valores significativos de speed-up y eficiencia. Por ejemplo, en el caso del grupo RV11, el tiempo secuencial de 9.05 horas se reduce a 2.54 horas, 1.27 horas y 0.99 horas, con 4, 10 y 20 núcleos, respectivamente. En términos de speed-up, los valores pueden ser hasta 3.56 con 4 núcleos (eficiencia de 89 %), 7.15 con 10 núcleos (eficiencia del 72 %) y 9.12 (eficiencia del 46 %), y resultados similares se obtienen para el resto de las familias. Se incluye información más detallada en el apéndice C.1, que contiene los resultados paralelos para cada una de las 218 instancias del benchmark BAliBASE v3.0.

Con el objetivo de conocer el rendimiento de la versión secuencial de M2Align sobre la versión original de MO-SAStrE, hemos realizado algunas ejecuciones de ambas técnicas resolviendo un número seleccionado de instancias del BAliBASE, estableciendo las mismas condiciones para ambas técnicas con el fin de poder realizar una comparación equitativa. Las pruebas se han realizado sobre el mismo sistema informático, utilizando los mismos parámetros y con los mismos conjuntos de

alineamientos pre-computados para la generación de la población inicial. La tabla 4.5 muestra el tiempo de ejecución (en minutos) de ambas técnicas.

Tabla 4.5: Tiempo de ejecución (en minutos) de M2Align frente a la versión original MO-SAStrE resolviendo nueve instancias del BAliBASE v3.0.

Instancia	Tiempo de Ejecución Secuencial (Mins)	
	MOSAStrE	M2Align
BB11001	24.02	0.44
BB11009	37.50	4.09
BB11011	80.96	3.49
BB11013	31.41	0.41
BB12002	85.14	0.76
BB12004	172.18	2.73
BB12010	183.04	3.92
BB12015	142.65	1.80
BB30009	294.24	0.55

Como podemos ver, M2Align realiza una ejecución más rápida alineando cada uno de estos conjuntos de datos. Por ejemplo, en el caso de la instancia BB30009, MO-SAStrE necesita 294.24 minutos, pero la versión secuencial de M2Align sólo requiere 0.55 minutos. La diferencia de tiempo se debe principalmente a la codificación basada en grupos de gaps de los alineamientos, lo que permite una implementación eficiente y rápida ejecución de los operadores de cruce y mutación.

4.7.4.2. Comparativa con otras metodologías MSA

Con el fin de determinar la calidad de los resultados de M2Align, hemos comparado sus resultados con los de otras técnicas clásicas del estado del arte para el Alineamiento Múltiple de Secuencias, las cuales están detalladas en la Tabla 4.1 y la técnica MOSAStrE. Los hemos puesto a prueba alineando todas las 218 instancias del BAliBASE v3.0. La Tabla 4.6 muestra los promedios de las puntuaciones de cada uno de los tres objetivos (STRIKE, TC y Non-Gaps) de los alineamientos optimizados por M2Align y generados por las otras técnicas de MSA y MOSAStrE.

Tabla 4.6: Promedio de las puntuaciones de las 218 instancias del BAliBASE v3.0 optimizadas por M2Align y las 8 técnicas clásicas MSA.

Método	Objetivos		
	STRIKE	TC (%)	Non-Gaps (%)
ClustalW	1.54	1.70	55.39
Muscle	1.76	1.90	52.48
Kalign	1.75	1.85	48.04
RetAlign	1.72	2.10	49.38
Tcoffee	1.75	1.87	45.35
ProbCons	1.74	1.85	44.21
Mafft	1.80	1.97	49.70
FSA	1.37	1.40	31.36
BAliBASE Ref	1.79	1.94	52.16
MOSAStrE	2.37	2.44	**58.51**
M2Align	**2.44**	**2.45**	58.13

Podemos ver que M2Align genera alineamientos con mejor calidad de acuerdo con los scores STRIKE y TC que el resto de las técnicas; y MOSAStrE en cambio provee soluciones mejor puntuadas para la métrica Non-Gaps.

Una comparativa más detallada se muestra en el apéndice C.2, donde hemos añadido dos tablas con las puntuaciones generadas por M2Align MOSAStrE y las técnicas MSA clásicas al alinear un conjunto de ocho instancias seleccionadas de BAliBASE v3.0 (BB11001, BB11009, BB11011, BB11013, BB12002, BB12004, BB12010, BB12015 y BB30009). En ella se puede observar que M2Align genera alineamientos con los mejores scores (STRIKE, TC y Non-Gaps) para todas las instancias de prueba, excepto en la última en la que ClustalW genera un alineamiento con mejor porcentaje de Non-Gaps.

Los resultados obtenidos revelan que nuestra propuesta paralela del algoritmo MOSAStrE, M2Align logra reducciones de tiempo significativas utilizando hasta 20 núcleos al resolver los conjuntos de instancias del benchmark BAliBASE 3.0. Nuestros experimentos también indican que nuestra implementación de MO-SAStrE es claramente más eficiente que la original. Finalmente, la comparación con un conjunto de técnicas MSA clásicas revela que M2Align proporciona los mejores resultados globales en las puntuaciones de STRIKE y TC.

Capítulo 5

Conclusiones y trabajos futuro

Este capítulo detalla las principales conclusiones alcanzadas como resultado de la investigación realizada en esta Tesis. Más concretamente, se destacan los hallazgos más significativos obtenidos en atención a los objetivos multiobjetivo y biológicos. Además, se definen futuras líneas de trabajo.

5.1. Resumen del trabajo realizado

La temática sobre la que ha girado la tesis ha sido la optimización de dos problemas del campo de la Bioinformática, la Inferencia Filogenética y al Alineamiento Múltiple de Secuencias usando metaheurísticas multiobjetivo. Se ha partido de una revisión inicial de los trabajos publicados sobre ambas temáticas, que nos ha permitido introducirnos en los temas biólogicos específicos de cada problema. Una vez estudiado los detalles de ambos problemas, se han desarrollado dos herramientas software de optimización para hacer frente a ambos problemas: MO-Phylogenetics para la Inferencia Filogenética y jMetalMSA para el Alineamiento Múltiple de Secuencias. Con ayuda de sus funcionalidades se realizaron estudios comparativos entre metaheurísticas multiobjetivo clásicas y modernas del estado del arte sobre formulaciones de dos y tres objetivos de ambos problemas, con el objetivo de conocer su rendimiento y capacidad de desarrollo. A partir de estos resultados se logró definir dos propuestas algorítmicas para cada problema, las cuales fueron implementados en ambos frameworks, M2Align para el Alineamiento Múltiple de Secuencias y MORPHY para la Inferencia Filogenética.

Todos estos trabajos han dado lugar a las siguientes publicaciones: tres artículos en revistas internacionales indexadas en el JCR ubicadas en el primer cuartil (Q1), la primera, Methods in Ecology and Evolution, en la que se publicó el framework MO-Phylogenetics, la segunda, International Journal of Intelligent Systems, en la que se publicó el análisis comparativo biobjetivo de algoritmos sobre el Alineamiento Múltiple de Secuencias, y la tercera en la revista Bioinformatics, en la que se publicó la propuesta algorítmica M2Align; un artículo en una revista internacional no indexada en el JCR llamada Progress in Artificial Intelligence, en el que se publicó el análisis algorítmico de una formulación de tres objetivos al problema del Alineamiento Múltiple de Secuencias y finalmente dos participaciones en congresos internacionales, en el 5th International Work-Conference on Bioinformatics and Biomedical Engineering iWBBIO 2017 en la que se presentó el framework jMetalMSA y en el 7th European Symposium on Computational Intelligence and Mathematics ESCIM 2015 donde se presentó un estudio inicial de metaheurísticas multiobjetivo aplicadas al alineamiento múltiple de secuencias.

5.2. Resumen de contribuciones

Las principales contribuciones de esta tesis se comentan a continuación:

- **Desarrollo del framework de optimización multiobjetivo MO-Phylogenetics para la Inferencia Filogenética:** El software MO-Phylogenetics puede ser muy útil para la comunidad bioinformática interesada en optimizar los problemas de inferencia filogenética gracias a la inclusión de modernas técnicas de optimización y a las características avanzadas proporcionadas por los paquetes Bio · · y PLL. Dispone de cuatro algoritmos multiobjetivo (NSGAII, SMS-EMOA, MOEA/D y PAES) adaptados a la inferencia de árboles filogenéticos y estrategias avanzadas de exploración del espacio de árboles. La configuración de sus parámetros está basada en un archivo de texto plano para su fácil implementación. Además, los investigadores que trabajan en algoritmos de optimización pueden usar *MO-Phylogenetics* como una herramienta para implementar nuevas técnicas para resolver problemas filogenéticos.

- **Desarrollo del framework de optimización multiobjetivo jMetalMSA para el Alineamiento Múltiple de Secuencias:** jMetalMSA esta basado en jMetal, provee un conjunto de algoritmos multiobjetivo (NSGAII, NSGAIII, MOCell, MOEA/D, SMS-EMOA y GWASF-GA) adaptados al problema MSA, varias funciones objetivos definidas como *score* (SOP, wSOP, TC, Non-Gaps, y STRIKE) que pueden ser fácilmente establecidos en una plantilla del problema MSA dentro del framework. Al igual provee varios operadores genéticos para su uso. Para jMetalMSA, se incluyeron varias métricas de evaluación (SOP, wSOP, TC, %Non-Gaps, STRIKE) y varios algoritmos disponibles gracias a jMetal. Las clases del problema pueden ser definidas fácilmente para poder incluir sin mayor cambio, una combinación de los objetivos a preferencias de los usuarios.

- **Análisis comparativo de rendimiento de metaheurísticas multiobjetivo aplicadas al MSA bajo formulaciones de dos y tres objetivos:** Con el objetivo de conocer el rendimiento de los algoritmos multiobjetivo aplicados al problema MSA, realizamos dos estudios comparativos utilizando metaheurísticas multiobjetivos clásicas y recientes del estado del arte, bajo formulaciones de dos y tres objetivos a optimizar. En el primer caso, analizando NSGA-II, SPEA2, AbYSS, MOCell y SMS-EMOA con los objetivos Suma de Pares (SOP) y Porcentaje de columnas totalmente alineadas y en el segundo analizando NSGA-II, MOCell, GWASF-GA y NSGA-III con los objetivos STRIKE, Porcentaje de columnas totalmente alineadas y Porcentaje de No-Gaps. En ambos casos NSGA-II concluyó demostrar un mejor rendimiento que el resto de los algoritmos para la mayoría de los problemas del BAliBASE 3.0.

- **Implementación de la propuesta algorítmica MORPHY a la Inferencia Filogenética:** En este caso utilizamos las funcionalidades del framework MO-Phylogenetics, para implementar el algoritmo MORPHY, una propuesta algorítmica al problema de la Inferencia Filogenética, que a diferencia del resto de trabajos expuestos en el estado arte, éste provee funcionalidades para trabajar con secuencias de proteínas incluyendo modelos de sustitución específicos para proteínas.

- **Implementación de la propuesta algorítmica M2Align al Alineamiento Múltiple de Secuencias:** Luego de obtener los resultados de la comparativa algorítmica sobre el problema MSA, en el que se define NSGAII como el algoritmo que provee los mejores resultados en ambas formulaciones, esto nos motivó a implementar M2Align nuestra propia versión del algoritmo MO-SAStrE el cual está basado en NSGAII, pero con una serie de mejoras significativas, incluyendo una nueva versión más rápida y eficiente de representar los alineamientos,

únicamente almacenando las posiciones de los grupos de gaps, con ello se logró acelerar mucho la ejecución de los operadores genéticos y de todo el algoritmo. M2Align es un proyecto Open Source alojado en el repositorio GitHub. Además se incluyeron funcionalidades de paralelismo, a la función de evaluación, la cual previo a un análisis realizado demostró ser la función que requería mas del 90 % del tiempo de ejecución. Con una versión paralela los resultados de ejecución mostraron un alto porcentaje de ganancia speed-up e incluso de la versión secuencial de M2Align, frente a la versión original MO-SAStrE. Los resultados obtenidos revelan que se pueden lograr reducciones de tiempo significativas usando hasta 20 núcleos al resolver los conjuntos de datos incluidos en el benchmark BAliBASE 3.0. Nuestros experimentos también indican que nuestra implementación de MO-SAStrE es claramente más eficiente que la original. Finalmente, una comparación con un conjunto de técnicas de alineación revela que M2Align proporciona los mejores resultados globales en las puntuaciones de STRIKE y TC.

5.3. Análisis Cualitativo de los resultados de la Tesis

Para la Inferencia Filogenética se implementó el algoritmo MORPHY, el cual provee funcionalidades únicas en el estado del arte, ya que además de inferir árboles filogenéticos multiobjetivo a partir de secuencias de ADN (nucléotidos), además provee soporte para secuencias de proteínas (amino-ácidos). Para el problema del Alineamiento Múltiple de Secuencias se implementó el algoritmo M2Align una versión mas eficiente y rápida de un software de alineamiento de secuencias que optimiza tres objetivos de calidad, gracias a la codificación de los alineamientos basados únicamente en información de los gaps y a la explotación de las capacidades que ofrecen las arquitecturas modernas basadas en clúster de procesadores multi-núcleo. A diferencia de las propuestas algorítmicas presentadas para la Inferencia Filogenética y algunas para el Alineamiento Múltiple de Secuencias, todas nuestras implementaciones realizadas en esta investigación se encuentran disponibles en el repositorio público Github para su libre acceso y distribución.

5.4. Líneas de trabajo futuro

De la investigación llevada a cabo en esta tesis doctoral se pueden derivar varias líneas de trabajos correspondientes aspectos abiertos que se han identificado así como a temas que se han iniciado y que todavía están por desarrollar.

En el aspecto de herramientas software, las características de MORPHY para inferir problemas de proteínas han de incorporarse en MO-Phylogenetics, de forma que ambos proyectos deberán quedar unificados en un solo en un futuro cercano. Por otro lado, existe una nueva versión de la librería PLL que incluye nuevas funcionalidades. Un trabajo prometedor es usar estar versión en MO-Phylogenetics para englobar tanto a la antigua versión como a BIO · ·, por lo que esta última dejaría de usarse. Con ello se mejorarían los tiempos de ejecución necesarios para estimar los scores de máxima parsimonia y máxima verosimilitud, primero porque se evitarían las transformaciones internas entre los tipos de estructuras de Bio · · y PLL y segundo porque PLL provee funcionalidades eficientes para estimar la máxima verosimilitud con evaluaciones parciales.

Con respecto a jMetalMSA, nuestro plan es extenderlo incluyendo otras características, tales como nuevas metaheurísticas que incorporan la articulación de preferencias, para que el biólogo pueda indicar algunas metas (propiedades deseadas) de los alineamientos a priori.

Desde un punto de vista algorítmico y la inferencia filogenética, hay que continuar con la idea de incluir en MORPHY características particulares del problema para conseguir que sea más competitivo. Estrategias basadas en búsquedas locales y el diseño de operadores específicos son

líneas claras a explorar, pero dado el alto coste computacional de las ejecuciones estas tareas deberán posponerse a la espera de la integración de la nueva versión de PLL.

Tomando como base el algoritmo M2Align, se pueden plantear varios trabajos, que se enumeran a continuación:

- Una primera idea es incorporar estrategias de búsqueda local que, al margen de intentar mejorar los resultados cualitativos del algoritmo, permitan ajustar el tiempo de cómputo para intentar tener ocupados todos los núcleos de los procesadores el cien por ciento del tiempo, lo que permitiría mejorar el rendimiento paralelo.

- La experiencia obtenida con M2Align muestra que el modelo de paralelismo síncrono que se ha usado siempre va a estar limitado por la parte secuencial del algoritmo; en este sentido, una mejora sería rediseñar el algoritmo con un modelo asíncrono, que permita simultanear las evaluaciones de la soluciones con el código secuencial. Esto es algo que ya se ha hecho con anterioridad y el reto está en cómo llevarlo a cabo de forma lo más simple posible, con un mínimo de cambios en el algoritmo original.

- M2Align está diseñado para funcionar en sistemas paralelos de memoria compartida, mientras que ya existen en el mercado tecnologías como Apache Spark que tienen modelos de programación paralela y distribuida de muy alto nivel, que permiten explotar el paralelismo de forma implícita, al contrario de lo que ocurriría si se usasen tecnologías tradicionales como PVM o MPI. Explorar la combinación de este tipo de tecnologías con M2Align daría pie a poder usar clústeres de cientos de máquinas, con lo que se podrían resolver problemas de Alineamiento Múltiple de Secuencias de gran tamaño. No obstante, tal como se ha comentado en el punto anterior, el modelo de paralelismo síncrono de M2Align impone limitaciones al rendimiento paralelo, por lo que el uso eficiente de Spark estaría sujeto a las mismas restricciones que los sistemas multi-núcleo usados con M2Align.

- Los operadores de cruce y mutación usados en M2Align son los mismos que los del algoritmo en que se basa, MO-SAStrE. Sin embargo, se podrían definir más operadores de variación, por lo que se podrían plantear estrategias dinámicas de selección o combinación de operadores con el fin de adaptar el algoritmo en tiempo de ejecución.

- M2Align ha sido configurado con ajustes convencionales, por lo que se podría hacer un análisis de sensibilidad de los parámetros para ajustar su rendimiento. Para ello se podrían usar herramientas de configuración automática, como irace o ParamILS.

En ambos contextos, tanto en el campo de la Filogenética y del Alineamiento Múltiple de Secuencias, se plantea la aplicación de las técnicas y software desarrollados en problemas reales en colaboración con investigadores del área de la biología.

Bibliografía

Abbasi M., Paquete L., Pereira P. B. (2015) Local search for multiobjective multiple sequence alignment. In: Bioinformatics and Biomedical Engineering, Lecture Notes in Computer Science, vol 9044, Springer International Publishing, pp. 175–182

Alba E. (1999) Análisis y diseño de algoritmos genéticos paralelos distribuidos. PhD thesis, University of Málaga

Alba E. (ed) (2005) Parallel Metaheuristics: A New Class of Algorithms. Wiley

Alba E., Tomassini M. (2002) Parallelism and evolutionary algorithms. IEEE Transactions on Evolutionary Computation 6(5):443 – 462

Armougom F., Moretti S., Poirot O., Audic S., Dumas P., Schaeli B., Keduas V., Notredame C. (2006) Expresso: automatic incorporation of structural information in multiple sequence alignments using 3d-coffee. Nucleic Acids Research 34(suppl 2):W604–W608, DOI 10.1093/nar/gkl092, URL http://nar.oxfordjournals.org/content/34/suppl_2/W604.abstract, http://nar.oxfordjournals.org/content/34/suppl_2/W604.full.pdf+html

Bäck T. (1996) Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. Oxford University Press

de Bakker P. I. W., Bateman A., Burke D. F., Miguel R. N., Mizuguchi K., Shi J., Shirai H., Blundell T. L. (2001) Homstrad: adding sequence information to structure-based alignments of homologous protein families. Bioinformatics 17(8):748–749, DOI 10.1093/bioinformatics/17.8.748, URL http://bioinformatics.oxfordjournals.org/content/17/8/748.abstract, http://bioinformatics.oxfordjournals.org/content/17/8/748.full.pdf+html

Berman F., Fox G., Hey A. (2003) Grid Computing. Making the Global Infrastructure a Reality. Communications Networking and Distributed Systems, Wiley

Berman H., Westbrook J., Feng Z., Gilliland G., Bhat T., Weissig H., Shindyalov I., Bourne P. (2000) The protein data bank. Nucleic Acids Research 28(1):235–242, URL http://nar.oxfordjournals.org/content/28/1/235, http://nar.oxfordjournals.org/content/28/1/235.full.pdf+html

Beume N., Naujoks B., Emmerich M. (2007) Sms-emoa: Multiobjective selection based on dominated hypervolume. European Journal of Operational Research 181(3):1653–1669

Blackburne B., Whelan S. (2012a) Measuring the distance between multiple sequence alignments. Bioinformatics 28(4):495–502, URL http://bioinformatics.oxfordjournals.org/content/28/4/495.abstract, http://bioinformatics.oxfordjournals.org/content/28/4/495.full.pdf+html

Blackburne B. P., Whelan S. (2012b) Class of multiple sequence alignment algorithm affects genomic analysis. Molecular biology and evolution p. mss256

Blackburne B. P., Whelan S. (2012c) Measuring the distance between multiple sequence alignments. Bioinformatics 28(4):495–502

Blum C., Roli A. (2003) Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys 35(3):268 – 308

Bos D. H., Posada D. (2005) Using models of nucleotide evolution to build phylogenetic trees. Developmental & Comparative Immunology 29(3):211–227

Bradley R. K., Roberts A., Smoot M., Juvekar S., Do J., Dewey C., Holmes I., Pachter L. (2009) Fast statistical alignment. PLOS Computational Biology 5(5):1–15, DOI 10.1371/journal.pcbi.1000392

Brent R. P. (1973) Algorithms for minimization without derivatives. Prentice-Hall series in automatic computation, Englewood Cliffs, N.J. Prentice-Hall, URL http://opac.inria.fr/record=b1082765

Brown J. R., Warren P. V. (1998) Antibiotic discovery: is it all in the genes? Drug Discovery Today 3(12):564–566

Burleigh J. G., Mathews S. (2007) Assessing systematic error in the inference of seed plant phylogeny. International Journal of Plant Sciences 168(2):125–135, DOI 10.1086/509588, URL http://dx.doi.org/10.1086/509588, http://dx.doi.org/10.1086/509588

Bush R. M., Bender C. A., Subbarao K., Cox N. J., Fitch W. M. (1999) Predicting the evolution of human influenza a. Science 286(5446):1921–1925

Canchignia H., Altimiram F., Sánchez E., Tapia E., Miccono M., Montes C., Espinoza D., Aguirre C., Valdés J., Tapia Ramírez P., González C., Seeger M., Prieto H. (2015) Isolation, characterization and genome analysis of a new *Pseudomonas veronii* isolate with antagonistic properties against Xiphinema index. Submitted to Microbial Ecology

Cancino W., Delbem A. (2010) A multi-criterion evolutionary approach applied to phylogenetic reconstruction. INTECH Open Access Publisher

Cancino W., Delbem A. C. (2007a) A Multi-objective Evolutionary Approach for Phylogenetic Inference. In: Obayashi S., Deb K., Poloni C., Hiroyasu T., Murata T. (eds) Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science, vol 4403, Springer Berlin Heidelberg, pp. 428–442, DOI 10.1007/978-3-540-70928-2_34, URL http://dx.doi.org/10.1007/978-3-540-70928-2_34

Cancino W., Delbem A. C. (2007b) A Multi-objective Evolutionary Approach for Phylogenetic Inference. In: Obayashi S., Deb K., Poloni C., Hiroyasu T., Murata T. (eds) Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science, vol 4403, Springer Berlin Heidelberg, pp. 428–442, DOI 10.1007/978-3-540-70928-2_34, URL http://dx.doi.org/10.1007/978-3-540-70928-2_34

Chippindale P. T., Bonett R. M., Baldwin A. S., Wiens J. J. (2004) Phylogenetic evidence for a major reversal of life-history evolution in plethodontid salamanders. Evolution 58(12):2809–2822

Chor B., Tuller T. (2005) Maximum likelihood of evolutionary trees: hardness and approximation. Bioinformatics 21(suppl 1):i97–i106

Coelho G. P., Da Silva A. E. A., Von Zuben F. J. (2010) An immune-inspired multi-objective approach to the reconstruction of phylogenetic trees. Neural Computing and Applications 19(8):1103–1132

Coello C. A., Lamont G. B., Veldhuizen D. A. V. (2007) Evolutionary Algorithms for Solving Multi-Objective Problems, 2nd edn. Genetic and Evolutionary Computation Series, Springer

Cohon J. L., Marks D. H. (1975) A review and evaluation of multiobjective programming techniques. Water Resources Research 11(2):208 – 220

Cole J. R., Wang Q., Cardenas E., Fish J., Chai B., Farris R. J., Kulam-Syed-Mohideen A., McGarrell D. M., Marsh T., Garrity G. M., et al (2009) The ribosomal database project: improved alignments and new tools for rrna analysis. Nucleic acids research 37(suppl 1):D141–D145

Congdon C. B. (2002) Gaphyl: An Evolutionary Algorithms Approach For The Study Of Natural Evolution. In: GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1057–1064

Connolly M. L. (1983) Solvent-accessible surfaces of proteins and nucleic acids. Science 221(4612):709–713

Cotta C., Moscato P. (2002a) Inferring phylogenetic trees using evolutionary algorithms. In: International Conference on Parallel Problem Solving from Nature, Springer, pp. 720–729

Cotta C., Moscato P. (2002b) Inferring phylogenetic trees using evolutionary algorithms. In: International Conference on Parallel Problem Solving from Nature, Springer, pp. 720–729

Crainic T. G., Toulouse M. (2003) Parallel strategies for metaheuristics. In: Glover F. W., Kochenberger G. A. (eds) Handbook of Metaheuristics, Kluwer Academic Publishers, Norwell, MA, USA

Cung V.-D., Martins S. L., Ribeiro C. C., Roucairol C. (2003) Strategies for the Parallel Implementation of Metaheuristics. In: Ribeiro C., Hansen P. (eds) Essays and Surveys in Metaheuristics, Kluwer Academic Publishers, Norwell, MA, USA, pp. 263–308

Darriba D., Taboada G. L., Doallo R., Posada D. (2012) jmodeltest 2: more models, new heuristics and parallel computing. Nature methods 9(8):772–772

Darwin C., Adler M. J., Hutchins R. M. (1872) The origin of species by means of natural selection. J. Murray

Day W. H., Johnson D. S., Sankoff D. (1986) The computational complexity of inferring rooted phylogenies by parsimony. Mathematical biosciences 81(1):33–42

Dayhoff M., Schwartz R., B.C. Orcutt B. (1978) A model of evolutionary change in proteins. In Atlas of Protein Sequences and Structure 5:345–352

Deb K. (1995) Optimization for Engineering Design. Prentice-Hall, New Delhi

Deb K. (2000) An efficient constraint handling mechanism method for genetic algorithms. Computer Methods in Applied Mechanics and Engineering 186(2/4):311 – 338

Deb K. (2001) Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley & Sons

Deb K., Jain H. (2014) An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. IEEE Transactions on Evolutionary Computation 18(4):577–601

Deb K., Pratap A., Agarwal S., Meyarivan T. (2002a) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2):182 – 197

Deb K., Pratap A., Agarwal S., Meyarivan T. (2002b) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2):182–197

Demšar J. (2006) Statistical comparison of classifiers over multiple data sets. Journal of Machine Learning Research 7:1 – 30

Derrac J., García S., Molina D., Herrera F. (2011) A practical tutorial on the use of non-parametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation 1(1):3 – 18, DOI http://dx.doi.org/10.1016/j.swevo.2011.02.002, URL http://www.sciencedirect.com/science/article/pii/S2210650211000034

Do C., Mahabhashyam M. S., Brudno M., Batzoglou S. (2005) Probcons: Probabilistic consistency-based multiple sequence alignment. Genome Research 15(2):330–340, DOI 10.1101/gr.2821705, URL http://genome.cshlp.org/content/15/2/330.abstract, http://genome.cshlp.org/content/15/2/330.full.pdf+html

Doolittle R. F. (1981) Similar amino acid sequences: chance or common ancestry. Science 214(4517):149–159

Dorigo M. (1992) Optimization, learning and natural algorithms. PhD thesis, Dipartamento di Elettronica, Politecnico di Milano

Dorigo M., Stützle T. (2003) Handbook of Metaheuristics, International Series In Operations Research and Management Science, vol 57, Kluwer Academic Publisher, chap The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances, pp. 251 – 285

Dorronsoro B. (2007) Diseño e implementación de algoritmos genéticos celulares para problemas complejos. PhD thesis, University of Málaga

Durillo J. J., Nebro A. J. (2011) jmetal: A java framework for multi-objective optimization. Advances in Engineering Software 42(10):760–771

Dutheil J., Gaillard S., Bazin E., Glémin S., Ranwez V., Galtier N., Belkhir K. (2006) Bio++: a set of C++ libraries for sequence analysis, phylogenetics, molecular evolution and population genetics. BMC bioinformatics 7:188, DOI 10.1186/1471-2105-7-188, URL http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=1501049\&tool=pmcentrez\&rendertype=abstract

Ebert J., Brutlag D. (2006) Development and validation of a consistency based multiple structure alignment algorithm. Bioinformatics 22(9):1080–1087, DOI 10.1093/bioinformatics/btl046, URL http://bioinformatics.oxfordjournals.org/content/22/9/1080.abstract, http://bioinformatics.oxfordjournals.org/content/22/9/1080.full.pdf+html

Edgar R. (2004a) Muscle: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Research 32(5):1792–1797, DOI 10.1093/nar/gkh340, URL http://nar.oxfordjournals.org/content/32/5/1792.abstract, http://nar.oxfordjournals.org/content/32/5/1792.full.pdf+html

Edgar R. C. (2004b) Muscle: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Research 32(5):1792–1797, DOI 10.1093/nar/gkh340, URL http://nar.oxfordjournals.org/content/32/5/1792.abstract, http://nar.oxfordjournals.org/content/32/5/1792.full.pdf+html

Edgeworth F. Y. (1881) Mathematical Psychics. P. Keagan, London

Edwards A., Cavalli-Sforza L., Heywood V., et al (1964) Phenetic and phylogenetic classification. Systematic Association Publication No 6 pp. 67–76

Ehrgott M. (2005) Multicriteria Optimization, 2nd edn. Springer

Eusuff M., Lansey K., Pasha F. (2006) Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. Engineering Optimization 38(2):129–154, DOI 10.1080/03052150500384759, URL http://dx.doi.org/10.1080/03052150500384759, http://dx.doi.org/10.1080/03052150500384759

Faith D. P. (1994) Genetic diversity and taxonomic priorities for conservation. Biological Conservation 68(1):69–74

Felsenstein J. (1981) Evolutionary trees from dna sequences: A maximum likelihood approach. Journal of Molecular Evolution 17(6):368–376, DOI 10.1007/BF01734359, URL http://dx.doi.org/10.1007/BF01734359

Felsenstein J. (2004) Inferring Phylogenies. Palgrave Macmillan, URL http://books.google.fr/books?id=GI6PQgAACAAJ

Feo T., Resende M. (1999) Greedy randomized adaptive search procedures. Journal of Global Optimization 6:109 – 133

Fitch W. M. (1966) An improved method of testing for evolutionary homology. Journal of molecular biology 16(1):9–16

Fitch W. M. (1971) Toward defining the course of evolution: Minimum change for a specific tree topology. Systematic Biology 20(4):406–416, DOI 10.1093/sysbio/20.4.406, URL http://sysbio.oxfordjournals.org/content/20/4/406.abstract, http://sysbio.oxfordjournals.org/content/20/4/406.full.pdf+html

Flouri T., Izquierdo-Carrasco F., Darriba D., Aberer A., Nguyen L.-T., Minh B., Von Haeseler A., Stamatakis A. (2015) The Phylogenetic Likelihood Library. Systematic Biology 64(2):356–362, DOI 10.1093/sysbio/syu084, http://sysbio.oxfordjournals.org/content/64/2/356.full.pdf+html

Fonseca C. M., Fleming P. J. (1993) Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In: Proc. of the Fifth Int. Conference on Genetic Algorithms, pp. 416 – 423

Foster I., Kesselman C. (1999) The Grid: Blueprint for a New Computing Infrastructure. Morgan-Kaufmann

Gallardo J. E., Cotta C., Fernández A. J. (2007) Reconstructing phylogenies with memetic algorithms and branch-and-bound.

Gen M., Li Y. (1999) Spanning tree-based genetic algorithm for the bicriteria fixed charge transportation problem. In: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), vol 3, p. 2271 Vol. 3, DOI 10.1109/CEC.1999.785556

Glover F. (1977) Heuristics for integer programming using surrogate constraints. Decision Sciences 8:156 – 166

Glover F. (1986) Future paths for integer programming and links to artificial intelligence. Computers & Operations Research 13:533 – 549

Glover F. (1998) A template for Scatter Search and Path Relinking. In: et al. J.-K. H. (ed) Artificial Evolution, Springer, no. 1363 in LNCS, pp. 13–54

Glover F. W., Kochenberger G. A. (2003) Handbook of Metaheuristics. Kluwer

Goëffon A., Richer J.-M., Hao J.-K. (2008) Progressive tree neighborhood applied to the maximum parsimony problem. IEEE/ACM transactions on computational biology and bioinformatics / IEEE, ACM 5(1):136–45, DOI 10.1109/TCBB.2007.1065, URL http://www.ncbi.nlm.nih.gov/pubmed/18245882

Goldberg D. E. (1989) Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley

Gondro C., Kinghorn B. (2007) A simple genetic algorithm for multiple sequence alignment. Genetics and Molecular Research 6(4):964–982

Gotoh O. (1990) Consistency of optimal sequence alignments. Bulletin of Mathematical Biology 52(4):509 – 525, DOI http://dx.doi.org/10.1016/S0092-8240(05)80359-3, URL http://www.sciencedirect.com/science/article/pii/S0092824005803593

Gottlieb J., Julstrom B. A., Raidl G. R., Rothlauf F. (2001) Prüfer numbers: A poor representation of spanning trees for evolutionary search. In: Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation, Morgan Kaufmann Publishers Inc., pp. 343–350

Graham R. L. (1969) Bounds on multiprocessor timing anomalies. SIAM Journal of Applied Mathematics 17:416 – 429

Guindon S., Gascuel O. (2003) A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. Systematic biology 52(5):696–704

Guindon S., Dufayard J.-F., Lefort V., Anisimova M., Hordijk W., Gascuel O. (2010) New Algorithms and Methods to Estimate Maximum-Likelihood Phylogenies: Assessing the Performance of PhyML 3.0. Systematic Biology 59(3):307–321, DOI 10.1093/sysbio/syq010, URL http://sysbio.oxfordjournals.org/content/59/3/307.abstract, http://sysbio.oxfordjournals.org/content/59/3/307.full.pdf+html

Handl J., Kell D. B., Knowles J. (2007) Multiobjective optimization in bioinformatics and computational biology. IEEE/ACM transactions on computational biology and bioinformatics / IEEE, ACM 4(2):279–92, DOI 10.1109/TCBB.2007.070203, URL http://www.ncbi.nlm.nih.gov/pubmed/17473320

Hasegawa M., Kishino H., Yano T.-a. (1985) Dating of the human-ape splitting by a molecular clock of mitochondrial dna. Journal of molecular evolution 22(2):160–174

Hassan M. R., Hossain M. M., Karmakar C. K., Kirley M. (2008) Phylogeny Inference Using a Multi-objective Evolutionary Algorithm with Indirect Representation, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 41–50, DOI 10.1007/978-3-540-89694-4_5, URL http://dx.doi.org/10.1007/978-3-540-89694-4_5

Henikoff S., Henikoff J. (1992) Amino acid substitution matrices from protein blocks. Proceedings of the National Academy of Sciences 89(22):10,915–10,919

Hogeweg P., Hesper B. (1984) The alignment of sets of sequences and the construction of phyletic trees: An integrated method. Journal of Molecular Evolution 20(2):175–186, DOI 10.1007/BF02257378, URL http://dx.doi.org/10.1007/BF02257378

Holm S. (1979) A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics 6(2):65–70, URL http://www.jstor.org/stable/4615733

Huelsenbeck J. P. (1995) Performance of phylogenetic methods in simulation. Systematic biology 44(1):17–48

Ingman M., Gyllensten U. (2006) mtdb: Human mitochondrial genome database, a resource for population genetics and medical sciences. Nucleic acids research 34(suppl 1):D749–D751

Jukes T. H., Cantor C. R. (1969) Evolution of protein molecules. Mammalian protein metabolism 3(21):132

Julstrom B. A. (2001) The blob code: A better string coding of spanning trees for evolutionary search. In: Genetic and Evolutionary Computation Conference Workshop Program, Morgan Kaufmann, pp. 256–261

Karp A. H., Flatt H. P. (1990) Measuring parallel processor performance. Communications of the ACM 33(5):539 – 543

Karp R. M. (1977) Probabilistic analysis of partitioning algorithms for the traveling salesman problem in the plane. Mathematics of Operations Research 2:209 – 224

Katoh K., Kuma K.-i., Miyata T. (2001) Genetic Algorithm-Based Maximum-Likelihood Analysis for Molecular Phylogeny. Journal of Molecular Evolution 53(4-5):477–484, DOI 10.1007/s002390010238, URL http://dx.doi.org/10.1007/s002390010238

Katoh K., Misawa K., Kuma K., Miyata T. (2002) Mafft: a novel method for rapid multiple sequence alignment based on fast fourier transform. Nucleic Acids Research 30(14):3059–3066, DOI 10.1093/nar/gkf436, URL http://nar.oxfordjournals.org/content/30/14/3059.abstract, http://nar.oxfordjournals.org/content/30/14/3059.full.pdf+html

Kaya M., Sarhan A., Abdullah R. (2014) Multiple sequence alignment with affine gap by using multi-objective genetic algorithm. Computer methods and programs in biomedicine 114(1):38–49, DOI 10.1016/j.cmpb.2014.01.013, URL http://www.ncbi.nlm.nih.gov/pubmed/24534604

Kemena C., Taly J., Kleinjung J., Notredame C. (2011) Strike: evaluation of protein msas using a single 3d structure. Bioinformatics 27(24):3385–3391

Kennedy J. (1999) Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance. In: Proceedings of IEEE Congress on Evolutionary Computation (CEC 1999), pp. 1931 – 1938

Kimura M. (1980) A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. Journal of molecular evolution 16(2):111–120

Kirkpatrick S., Gelatt C. D., Vecchi M. P. (1983) Optimization by simulated annealing. Science 220:671 – 680

Knowles J., Corne D. (1999a) The pareto archived evolution strategy: A new baseline algorithm for multiobjective optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation, IEEE Press, pp. 9 – 105

Knowles J., Corne D. (1999b) The Pareto archived evolution strategy: a new baseline algorithm for Pareto multiobjective optimisation. In: Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on, vol 1, pp. –105 Vol. 1, DOI 10.1109/CEC.1999.781913

Knowles J. D., Thiele L., Zitzler E. (2006) A tutorial on the performance assessment of stochastic multiobjective optimizers. Tech. Rep. TIK-Report 214, Computer Engineering and Networks Laboratory, ETHC Zurich

Koza J. R. (1992) Genetic Programming. On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge, Massachusetts

Kuhner M. K., Felsenstein J. (1994) A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. Molecular Biology and Evolution 11(3):459–468

Laguna M., Martí R. (2003) Scatter Search. Methodology and Implementations in C. Kluwer

Lanave C., Preparata G., Sacone C., Serio G. (1984) A new method for calculating evolutionary substitution rates. Journal of molecular evolution 20(1):86–93

Larrañaga P., Etxeberria R., Lozano J. A., Peña J. M. (1999) Optimization by learning and simulation of Bayesian and Gaussian networks. Tech. Rep. KZZA-IK-4-99, Department of Computer Science and Artificial Intelligence, University of the Basque Country

Lassmann T., Sonnhammer E. L. (2005) Kalign – an accurate and fast multiple sequence alignment algorithm. BMC Bioinformatics 6(1):1–9, DOI 10.1186/1471-2105-6-298, URL http://dx.doi.org/10.1186/1471-2105-6-298

Lemey P. (2009) The phylogenetic handbook: a practical approach to phylogenetic analysis and hypothesis testing. Cambridge University Press

Lemmon A. R., Milinkovitch M. C. (2002) The metapopulation genetic algorithm: an efficient solution for the problem of large phylogeny estimation. Proceedings of the National Academy of Sciences 99(16):10,516–10,521

Lewis P. O. (1998) A genetic algorithm for maximum-likelihood phylogeny inference using nucleotide sequence data. Molecular biology and evolution 15(3):277–83, URL http://www.ncbi.nlm.nih.gov/pubmed/9501494

Liu Y., Schmidt B. (2014) Multiple protein sequence alignment with msaprobs. Multiple Sequence Alignment Methods pp. 211–218

López-Camacho E., García Godoy M. J., Nebro A. J., Aldana-Montes J. F. (2014) jMetalCpp: optimizing molecular docking problems with a C++ metaheuristic framework. Bioinformatics (Oxford, England) 30(3):437–8, DOI 10.1093/bioinformatics/btt679, URL http://www.ncbi.nlm.nih.gov/pubmed/24273242

Lötynoja A., Goldman N. (2005) An algorithm for progressive multiple alignment of sequences with insertions. Proceedings of the National Academy of Sciences of the United States of America 102(30):10,557–10,562, DOI 10.1073/pnas.0409137102, URL http://www.pnas.org/content/102/30/10557.abstract, http://www.pnas.org/content/102/30/10557.full.pdf

Lourenço H. R., Martin O., Stützle T. (2002) Handbook of Metaheuristics, Kluwer Academic Publishers, chap Iterated local search, pp. 321 – 353

Luque G. (2006) Resolución de problemas combinatorios con aplicación real en sistemas distribuidos. PhD thesis, University of Málaga

Macey J. R. (2005) Plethodontid salamander mitochondrial genomics: A parsimony evaluation of character conflict and implications for historical biogeography. Cladistics 21(2):194–202, DOI 10.1111/j.1096-0031.2005.00054.x, URL http://dx.doi.org/10.1111/j.1096-0031.2005.00054.x

Matsuda H. (1995) Construction of Phylogenetic Trees from Amino Acid Sequences using a Genetic Algorithm. Sciences, Computer p. 560

Metropolis N., Rosenbluth A., Rosenbluth M., Teller A., Teller E. (1953) Equation of state calculations by fast computing machines. Journal of Chemical Physics 21:1087 – 1092

Mladenovic N., Hansen P. (1997) Variable neighborhood search. Com Oper Res 24:1097 – 1100

Moilanen A. (2001) Simulated evolutionary optimization and local search: Introduction and application to tree search. Cladistics 17(1):S12–S25

Mühlenbein H. (1998) The equation for response to selection and its use for prediction. Evolutionary Computation 5:303 – 346

Naznin F., Sarker R., Essam D. (2011) Vertical decomposition with genetic algorithm for multiple sequence alignment. BMC Bioinformatics 12:353, DOI 10.1186/1471-2105-12-353, URL http://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-12-353

Naznin F., Sarker R., Essam D. (2012) Progressive alignment method using genetic algorithm for multiple sequence alignment. IEEE Transactions on Evolutionary Computation 16(5):615–631, DOI 10.1109/TEVC.2011.2162849

Nebro A., Durillo J. J., Vergne M. (2015a) Redesigning the jmetal multi-objective optimization framework. In: Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation, ACM, New York, NY, USA, GECCO Companion '15, pp. 1093–1100

Nebro A. J., Durillo J. J., Luna F., Dorronsoro B., Alba E. (2009) Mocell: A cellular genetic algorithm for multiobjective optimization. International Journal of Intelligent Systems 24(7):723 – 725

Nebro A. J., Zambrano-Vega C., Durillo J., Aldana-Montes J. (2015b) A study of multiple sequence alignment with multi-objective metaheuristics. In: 7th European Symposium on Computational Intelligence and Mathematics, László Kóczy, Jesús Medina, pp. 156–161

Needleman S., Wunsch C. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of Molecular Biology 48(3):443 – 453, DOI http://dx.doi.org/10.1016/0022-2836(70)90057-4, URL http://www.sciencedirect.com/science/article/pii/0022283670900574

Nguyen L.-T., Schmidt H. A., von Haeseler A., Minh B. Q. (2015) Iq-tree: A fast and effective stochastic algorithm for estimating maximum-likelihood phylogenies. Molecular Biology and Evolution 32(1):268–274, DOI 10.1093/molbev/msu300, URL http://mbe.oxfordjournals.org/content/32/1/268.abstract, http://mbe.oxfordjournals.org/content/32/1/268.full.pdf+html

Notredame C., Higgins D. G. (1996) Saga: Sequence alignment by genetic algorithm. Nucleic Acids Research 24(8):1515–1524, DOI 10.1093/nar/24.8.1515, URL http://nar.oxfordjournals.org/content/24/8/1515.abstract, http://nar.oxfordjournals.org/content/24/8/1515.full.pdf+html

Notredame C., Higgins D., Heringa J. (2000) T-coffee: a novel method for fast and accurate multiple sequence alignment. Journal of Molecular Biology 302(1):205 – 217, DOI http://dx.doi.org/10.1006/jmbi.2000.4042, URL http://www.sciencedirect.com/science/article/pii/S0022283600940427

Ortuño F., Valenzuela O., Rojas F., Pomares H., Florido J., Urquiza J., Rojas I. (2013) Optimizing multiple sequence alignments using a genetic algorithm based on three objectives: structural information, non-gaps percentage and totally conserved columns. Bioinformatics (Oxford, England) 29(17):2112–21, DOI 10.1093/bioinformatics/btt360, URL http://www.ncbi.nlm.nih.gov/pubmed/23793754

Osyczka A. (1895) Multicriteria optimization for engineering design. In: Gero J. S. (ed) Design Optimization, Academic Press, pp. 193 – 227

Ou C.-Y., Myers G., Mullins J. I., et al (1992) Molecular epidemiology of hiv transmission in a dental practice. Science 256(5060):1165

Pal S. K., Bandyopadhyay S., Ray S. S. (2006) Evolutionary computation in bioinformatics: a review. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 36(5):601–615

Pareto V. (1896) Cours D'Economie Politique, vol I and II. F. Rouge, Lausanne

Pei J. (2008) Multiple protein sequence alignment. Current Opinion in Structural Biology 18(3):382 – 386, URL http://www.sciencedirect.com/science/article/pii/S0959440X08000407, nucleic acids / Sequences and topology

Pelikan M., Goldberg D. E., Cantú-Paz E. (1999) BOA: The Bayesian optimization algorithm. In: Banzhaf W., Daida J., Eiben A. E., Garzon M. H., Honavar V., Jakiela M., Smith R. E. (eds) Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99, Morgan Kaufmann Publishers, San Francisco, CA, vol 1, pp. 525 – 532

Picciotto S. (1999) How to encode a tree. PhD thesis, University of California, San Diego

Poladian L., Jermiin L. (2005) Multi-objective evolutionary algorithms and phylogenetic inference with multiple data sets. Soft Computing 10(4):359–368, DOI 10.1007/s00500-005-0495-7, URL http://link.springer.com/10.1007/s00500-005-0495-7

Press W. H., Teukolsky S. A., Vetterling W. T., Flannery B. P. (1992) Numerical Recipes in C: The Art of Scientific Computing. Second Edition. Cambridge University Press

Prüfer H. (1918) Neuer beweis eines satzes über permutationen. Arch Math Phys 27(1918):742–744

R. Saborido A. R., Luque. M. (2016) Global wasf-ga: An evolutionary algorithm in multiobjective optimization to approximate the whole pareto optimal front., evolutionary Computation. In Press

Raghava G., Searle S. M., Audley P. C., Barber J. D., Barton G. J. (2003) Oxbench: A benchmark for evaluation of protein multiple sequence alignment accuracy. BMC Bioinformatics 4(1):1–23, DOI 10.1186/1471-2105-4-47, URL http://dx.doi.org/10.1186/1471-2105-4-47

Raidl G. R., Julstrom B. A. (2003) Edge sets: an effective evolutionary coding of spanning trees. IEEE Transactions on evolutionary computation 7(3):225–239

Rani R. R., Ramyachitra D. (2016) Multiple sequence alignment using multi-objective based bacterial foraging optimization algorithm. Biosystems 150:177 – 189, DOI http://dx.doi.org/10.1016/j.biosystems.2016.10.005, URL http://www.sciencedirect.com/science/article/pii/S0303264716302611

Reeves C. (1993) Modern Heuristic Techniques for Combinatorial Problems. Blackwell Scientific Publishing, Oxford, UK

Reijmers T. H., Wehrens R., Buydens L. M. (1999) Quality criteria of genetic algorithms for construction of phylogenetic trees. Journal of Computational Chemistry 20(8):867–876

Roshan U., Livesay D. R. (2006) Probalign: multiple sequence alignment using partition function posterior probabilities. Bioinformatics 22(22):2715–2721, DOI 10.1093/bioinformatics/btl472, URL http://bioinformatics.oxfordjournals.org/content/22/22/2715.abstract, http://bioinformatics.oxfordjournals.org/content/22/22/2715.full.pdf+html

Rothlauf F., Goldberg D. E., Heinzl A. (2002) Network random keys: a tree representation scheme for genetic and evolutionary algorithms. Evolutionary Computation 10(1):75–97

Rubio-Largo A., Vega-Rodriguez M., Gonzalez-Alvarez D. (2015) A hybrid multiobjective memetic metaheuristic for multiple sequence alignment. Evolutionary Computation, IEEE Transactions on PP(99):1–16, DOI 10.1109/TEVC.2015.2469546

Rubio-Largo A., Vega-Rodríguez M., González-Álvarez D. (2016) Hybrid multiobjective artificial bee colony for multiple sequence alignment. Applied Soft Computing 41:157 – 168, DOI http://dx.doi.org/10.1016/j.asoc.2015.12.034, URL http://www.sciencedirect.com/science/article/pii/S1568494615008133

Saitou N., Nei M. (1987a) The neighbor-joining method: a new method for reconstructing phylogenetic trees. Molecular biology and evolution 4(4):406–425

Saitou N., Nei M. (1987b) The neighbor-joining method: a new method for reconstructing phylogenetic trees. Molecular biology and evolution 4(4):406–425

Sanderson M., Donoghue M., Piel W., Eriksson T. (1994) Treebase: a prototype database of phylogenetic analyses and an interactive tool for browsing the phylogeny of life. American Journal of Botany 81(6):183

Santander-Jiménez S., Vega-Rodríguez M. A. (2013a) Applying a multiobjective metaheuristic inspired by honey bees to phylogenetic inference. Biosystems 114(1):39–55, DOI http://dx.doi.org/10.1016/j.biosystems.2013.07.001, URL http://www.sciencedirect.com/science/article/pii/S0303264713001615

Santander-Jiménez S., Vega-Rodríguez M. A. (2013b) A multiobjective proposal based on the firefly algorithm for inferring phylogenies. In: European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics, Springer, pp. 141–152

Santander-Jiménez S., Vega-Rodríguez M. A. (2014) A hybrid approach to parallelize a fast non-dominated sorting genetic algorithm for phylogenetic inference. Concurrency and Computation: Practice and Experience 27(3):702–734, URL http://doi.wiley.com/10.1002/cpe.3269

Santander-Jiménez S., Vega-Rodríguez M. A. (2015) A hybrid approach to parallelize a fast non-dominated sorting genetic algorithm for phylogenetic inference. Concurrency and Computation: Practice and Experience 27(3):702–734

Schmidt O., Drake H. L., Horn M. A. (2010) Hitherto unknown [fe-fe]-hydrogenase gene diversity in anaerobes and anoxic enrichments from a moderately acidic fen. Applied and Environmental Microbiology 76(6):2027–2031

Seeluangsawat P., Chongstitvatana P. (2005) A multiple objective evolutionary algorithm for multiple sequence alignment. In: Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation, ACM, New York, NY, USA, GECCO '05, pp. 477–478, DOI 10.1145/1068009.1068088, URL http://doi.acm.org/10.1145/1068009.1068088

Shendure J., Ji H. (2008) Next-generation dna sequencing. Nature Biotechnology 26(10):1135–1145, DOI 10.1038/nbt1486, URL http://dx.doi.org/10.1038/nbt1486

Sheskin D. J. (2007) Handbook of Parametric and Nonparametric Statistical Procedures. Chapman & Hall/CRC

Siddiqui A. S., Dengler U., Barton G. J. (2001) 3dee: a database of protein structural domains. Bioinformatics 17(2):200–201

Sievers F., Wilm A., Dineen D., Gibson T. J., Karplus K., Li W., Lopez R., McWilliam H., Remmert M., Söding J., Thompson J. D., Higgins D. G. (2011) Fast, scalable generation of high-quality protein multiple sequence alignments using clustal omega. Molecular Systems Biology 7(1), DOI 10.1038/msb.2011.75, URL http://msb.embopress.org/content/7/1/539, http://msb.embopress.org/content/7/1/539.full.pdf

da Silva F., Pérez J. S., Pulido J. G., Rodríguez M. V. (2010) Alineaga?a genetic algorithm with local search optimization for multiple sequence alignment. Applied Intelligence 32(2):164–172, DOI 10.1007/s10489-009-0189-4, URL http://dx.doi.org/10.1007/s10489-009-0189-4

da Silva F. J. M., Pérez J. M. S., Pulido J. A. G., Rodríguez M. a. V. (2011) Parallel Niche Pareto AlineaGA–an evolutionary multiobjective approach on multiple sequence alignment. Journal of integrative bioinformatics 8(3):174, DOI 10.2390/biecoll-jib-2011-174, URL http://www.ncbi.nlm.nih.gov/pubmed/21926437

Sneath P. H., Sokal R. R., et al (1973) Numerical taxonomy: the principles and practice of numerical classification. WH Freeman, San Francisco

Sokal R. R. (1958) A statistical method for evaluating systematic relationships. Univ Kans Sci Bull 38:1409–1438

Soto M., Ochoa A., Acid S., de Campos L. M. (1999) Introducing the polytree aproximation of distribution algorithm. In: Second Symposium on Artificial Intelligence. Adaptive Systems. CIMAF 99, pp. 360 – 367

Soto W., Becerra D. (2014) A multi-objective evolutionary algorithm for improving multiple sequence alignments. In: Campos S. (ed) Advances in Bioinformatics and Computational Biology, Lecture Notes in Computer Science, vol 8826, Springer International Publishing, pp. 73–82

Srinivas N., Deb K. (1994) Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation 2(3):221 – 248

Stamatakis A. (2004) Distributed and Parallel Algorithms and Systems for Inference of Huge Phylogenetic Trees based on the Distributed and Parallel Algorithms and Systems for Inference of Huge Phylogenetic Trees based on the Maximum Likelihood Method. PhD thesis, Technische Universität München, Germany

Stamatakis A. (2006) Raxml-vi-hpc: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. Bioinformatics 22(21):2688–2690, DOI 10.1093/bioinformatics/btl446, URL http://bioinformatics.oxfordjournals.org/content/22/21/2688.abstract, http://bioinformatics.oxfordjournals.org/content/22/21/2688.full.pdf+html

Steel M., Penny D. (2000) Parsimony, Likelihood, and the Role of Models in Molecular Phylogenetics. Molecular biology and evolution pp. 839–850

Stützle T. (1999) Local search algorithms for combinatorial problems analysis, algorithms and new applications. Tech. rep., DISKI Dissertationen zur Künstliken Intelligenz. Sankt Augustin, Germany

Swofford D., Olsen G., Waddell P., Hillis D. (1996) Phylogeny reconstruction. In: Molecular Systematics, 3rd edn, Sinauer, chap 11, pp. 407–514

Szabó A., Novák
'A., Miklós I., Hein J. (2010) Reticular alignment: A progressive corner-cutting method for multiple sequence alignment. BMC bioinformatics 11(1):570

Taheri J., Zomaya A. (2009) Rbt-ga: a novel metaheuristic for solving the multiple sequence alignment problem. BMC Genomics 10(1):1–11, URL http://bmcgenomics.biomedcentral.com/articles/10.1186/1471-2164-10-S1-S10

Tateno Y., Takezaki N., Nei M. (1994) Relative efficiencies of the maximum-likelihood, neighbor-joining, and maximum-parsimony methods when substitution rate varies with site. Molecular Biology and Evolution 11(2):261–277

Thompson E., Paulden T., Smith D. K. (2007) The dandelion code: A new coding of spanning trees for genetic algorithms. IEEE Transactions on Evolutionary Computation 11(1):91–100, DOI 10.1109/TEVC.2006.880730

Thompson J., D.G.Higgins, Gibson T. (1994) Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic Acids Research 22(22):4673–4680, DOI 10.1093/nar/22.22.4673, URL http://nar.oxfordjournals.org/content/22/22/4673.abstract, http://nar.oxfordjournals.org/content/22/22/4673.full.pdf+html

Thompson J., Koehl P., Poch O. (2005) Balibase 3.0: latest developments of the multiple sequence alignment benchmark. Proteins 61:127–136

Thompson J. D., Plewniak F., Poch O. (1999a) Balibase: a benchmark alignment database for the evaluation of multiple alignment programs. Bioinformatics 15(1):87–88, DOI 10.1093/bioinformatics/15.1.87, URL http://bioinformatics.oxfordjournals.org/content/15/1/87.abstract, http://bioinformatics.oxfordjournals.org/content/15/1/87.full.pdf+html

Thompson J. D., Plewniak F., Poch O. (1999b) A comprehensive comparison of multiple sequence alignment programs. Nucleic acids research 27(13):2682–2690

Trifinopoulos J., Nguyen L.-T., vonÂ Haeseler A., Minh B. Q. (2016) W-iq-tree: a fast online phylogenetic tool for maximum likelihood analysis. Nucleic Acids Research 44(W1):W232, DOI 10.1093/nar/gkw256, URL +http://dx.doi.org/10.1093/nar/gkw256, /oup/backfile/Content_public/Journal/nar/44/W1/10.1093_nar_gkw256/3/gkw256.pdf

Tuffley C., Steel M. (1997) Links between maximum likelihood and maximum parsimony under a simple model of site substitution. Bulletin of Mathematical Biology 59(3):581–607, DOI 10.1007/BF02459467, URL http://dx.doi.org/10.1007/BF02459467

Van Veldhuizen D. A. (1999) Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations. PhD thesis, DTIC Document, Wright-Patterson, AFB, OH

Van Veldhuizen D. A., Lamont G. B. (1998) Multiobjective evolutionary algorithm research: A history and analysis. Tech. Rep. TR-98-03, Dept. Elec. Comput. Eng., Graduate School of Eng., Air Force Inst. Technol., Wright-Patterson, AFB, OH

Van Walle I., Lasters I., Wyns L. (2005) Sabmark?a benchmark for sequence alignment that covers the entire known fold space. Bioinformatics 21(7):1267–1268, DOI 10.1093/bioinformatics/bth493, URL http://bioinformatics.oxfordjournals.org/content/21/7/1267.abstract, http://bioinformatics.oxfordjournals.org/content/21/7/1267.full.pdf+html

Waterman M., Smith T., Beyer W. (1976) Some biological sequence metrics. Advances in Mathematics 20(3):367 – 387

Whittaker J. (1990) Graphical models in applied multivariate statistics. John Wiley & Sons, Inc.

Yang Z. (1994) Maximum likelihood phylogenetic estimation from dna sequences with variable rates over sites: approximate methods. Journal of Molecular evolution 39(3):306–314

Yang Z. (2006) Computational molecular evolution. Oxford University Press

Zambrano-Vega C., Nebro A. J., Aldana-Montes J. (2016) Mo-phylogenetics: a phylogenetic inference software tool with multi-objective evolutionary metaheuristics. Methods in Ecology and Evolution 7(7):800–805, DOI 10.1111/2041-210X.12529, URL http://dx.doi.org/10.1111/2041-210X.12529

Zambrano-Vega C., Nebro A. J., Durillo J. J., García-Nieto J., Aldana-Montes J. (2017a) Multiple sequence alignment with multiobjective metaheuristics. a comparative study. International Journal of Intelligent Systems 32(8):843–861, DOI 10.1002/int.21892, URL http://dx.doi.org/10.1002/int.21892

Zambrano-Vega C., Nebro A. J., García-Nieto J., Aldana-Montes J. (2017b) Comparing multiobjective metaheuristics for solving a three-objective formulation of multiple sequence alignment. Progress in Artificial Intelligence pp. 1–16, DOI 10.1007/s13748-017-0116-6, URL http://dx.doi.org/10.1007/s13748-017-0116-6

Zambrano-Vega C., Nebro A. J., García-Nieto J., Aldana-Montes J. (2017c) A Multi-objective Optimization Framework for Multiple Sequence Alignment with Metaheuristics, Springer International Publishing, Cham, pp. 245–256. DOI 10.1007/978-3-319-56154-7_23, URL http://dx.doi.org/10.1007/978-3-319-56154-7_23

Zambrano-Vega C., Nebro A. J., García-Nieto J., Aldana-Montes J. F. (2017d) M2align: parallel multiple sequence alignment with a multi-objective metaheuristic. Bioinformatics DOI 10.1093/bioinformatics/btx338, URL https://doi.org/10.1093/bioinformatics/btx338

Zhang Q., Li H. (2007) MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Trans Evolutionary Computation 11(6):712–731

Zhu H., He Z., Jia Y. (2016) A novel approach to multiple sequence alignment using multiobjective evolutionary algorithm based on decomposition. IEEE Journal of Biomedical and Health Informatics 20(2):717–727, DOI 10.1109/JBHI.2015.2403397

Zitzler E., Künzli S. (2004) Indicator-based selection in multiobjective search. In: International Conference on Parallel Problem Solving from Nature, Springer, pp. 832–842

Zitzler E., Thiele L. (1999a) Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. IEEE Transactions on Evolutionary Computation 3(4):257 – 271

Zitzler E., Thiele L. (1999b) Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Trans on Evol Comp 3(4):257–271

Zitzler E., Laumanns M., Thiele L. (2001) SPEA2: Improving the strength pareto evolutionary algorithm. Tech. Rep. 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland

Zitzler E., Thiele L., Laummans M., Fonseca C. M., da Fonseca V. G. (2003) Performance assessment of multiobjective optimizers: an analysis and review. IEEE Transactions on Evolutionary Computation 7(2):114 – 132

Zwickl D. J. (2006) Genetic algorithm approaches for the phylogenetic analysis of large biological sequence datasets under the maximum likelihood criterion. PhD thesis, Biological Sciences at University of Texas

Apéndices

Apéndice A

Contribuciones que avalan la tesis

En este apéndice se enumeran las publicaciones científicas que avalan esta tesis doctoral. Se incluyen en los artículos publicados en revistas internacionales indexadas en el JCR y no indexadas y, los artículos presentados en congresos internacionales.

A.1. Artículos publicados en revistas internacionales indexadas en el JCR

1. C. Zambrano-Vega, A.J. Nebro, J. García Nieto, J.F. Aldana Montes. M2Align: parallel multiple sequence alignment with a multi-objective metaheuristic. Bioinformatics. Published: 24 May 2017. DOI: https://doi.org/10.1093/bioinformatics/btx338. (FACTOR DE IMPACTO: 7.307, 2/57, MATHEMATICAL & COMPUTATIONAL BIOLOGY). ISSN: 1367-4803. OXFORD UNIV PRESS.

2. C. Zambrano-Vega, A.J. Nebro, J.J. Durillo, J. García Nieto, J.F. Aldana Montes. Multiple Sequence Alignment with Multiobjective Metaheuristics. A Comparative Study. International Journal of Intelligent Systems. Volume 32, Issue 8, August 2017. Pages 843-861. Online 16 February 2017. DOI: http://dx.doi.org/10.1002/int.21892. (FACTOR DE IMPACTO: 2.929, 31/133, COMPUTER SCIENCE, ARTIFICIAL INTELLIGENCE). ISSN: 0884-8173. WILEY-BLACKWELL.

3. C. Zambrano-Vega, A. J. Nebro and J. F. Aldana-Montes. MO-Phylogenetics: a phylogenetic inference software tool with multi-objective evolutionary metaheuristics. Methods in Ecology and Evolution. Volume 7, Issue 7 July 2016. Pages 800-805. DOI: http://dx.doi.org/10.1111/2041-210X.12529. (FACTOR DE IMPACTO: 5.708, 13/153, ECOLOGY-SCI). ISSN: 2041-210X. WILEY-BLACKWELL.

A.2. Artículos publicados en revistas internacionales

1. Zambrano-Vega C., Nebro A. J., García-Nieto J., Aldana-Montes J. F. *Comparing Multi-Objective Metaheuristics For Solving a Three-Objective Formulation of Multiple Sequence Alignment.* Progress in Artificial Intelligence. doi: 10.1007/s13748-017-0116-6, 2017.

A.3. Publicaciones en congresos internacionales

1. Zambrano-Vega C., Nebro A. J., García-Nieto J., Aldana-Montes J.F. *A Multi-objective Optimization Framework for Multiple Sequence Alignment with Metaheuristics*, pages 245-256. Springer International Publishing. Bioinformatics and Biomedical Engineering: 5th International Work-Conference, IWBBIO 2017.

2. Nebro A.J., Zambrano-Vega C., Durillo J.J. and Aldana Montes J.F. (2015) *A Study of Multiple Sequence Alignment With Multi-Objective Metaheuristics*. 7th European Symposium on Computational Intelligence and Mathematics (ESCIM 2015). Octubre 2015. Cádiz.

Apéndice B

Tablas de resultados

B.1. Comparativa biobjetivo - indicadores de calidad

Tabla B.1: Mediana y rango intercuartílico del indicador I_{HV} para los problemas de la familia RV11 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB11001	$3{,}50e-01_{1{,}6e-01}$	$4{,}12e-01_{1{,}9e-01}$	$3{,}58e-01_{1{,}9e-01}$	$1{,}82e-01_{2{,}2e-01}$	$2{,}58e-01_{1{,}4e-01}$	$3{,}05e-01_{1{,}7e-01}$
BB11002	$2{,}89e-01_{7{,}9e-02}$	$2{,}11e-01_{9{,}9e-02}$	$2{,}64e-01_{3{,}3e-02}$	$6{,}78e-02_{8{,}3e-02}$	$2{,}77e-01_{2{,}5e-02}$	$2{,}90e-01_{2{,}1e-02}$
BB11003	$3{,}28e-01_{1{,}8e-01}$	$2{,}71e-01_{9{,}9e-02}$	$2{,}90e-01_{1{,}2e-01}$	$4{,}28e-02_{3{,}5e-02}$	$3{,}80e-01_{1{,}6e-01}$	$2{,}71e-01_{2{,}5e-01}$
BB11004	$3{,}95e-01_{8{,}4e-02}$	$3{,}09e-01_{8{,}2e-02}$	$3{,}78e-01_{9{,}8e-02}$	$0{,}00e+00_{0{,}0e+00}$	$3{,}65e-01_{1{,}5e-01}$	$2{,}44e-01_{2{,}1e-01}$
BB11005	$1{,}65e-01_{3{,}6e-02}$	$1{,}53e-01_{1{,}0e-02}$	$1{,}59e-01_{6{,}7e-03}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}02e-01_{4{,}2e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB11006	$3{,}61e-01_{1{,}9e-02}$	$3{,}34e-01_{2{,}4e-02}$	$3{,}22e-01_{2{,}0e-02}$	$0{,}00e+00_{0{,}0e+00}$	$3{,}08e-01_{7{,}5e-02}$	$1{,}00e-01_{4{,}7e-02}$
BB11007	$2{,}66e-01_{3{,}1e-02}$	$2{,}44e-01_{2{,}3e-02}$	$2{,}45e-01_{3{,}0e-02}$	$7{,}33e-02_{3{,}8e-02}$	$2{,}36e-01_{3{,}5e-02}$	$1{,}26e-01_{4{,}8e-02}$
BB11008	$5{,}29e-01_{3{,}4e-02}$	$5{,}06e-01_{3{,}5e-02}$	$6{,}01e-01_{4{,}7e-02}$	$3{,}27e-01_{1{,}3e-01}$	$6{,}14e-01_{1{,}2e-01}$	$4{,}30e-01_{8{,}5e-02}$
BB11009	$4{,}08e-01_{1{,}9e-01}$	$3{,}36e-01_{1{,}3e-01}$	$2{,}81e-01_{1{,}3e-01}$	$5{,}24e-03_{1{,}1e-01}$	$3{,}54e-01_{1{,}2e-01}$	$2{,}31e-01_{1{,}3e-01}$
BB11010	$3{,}71e-01_{1{,}0e-01}$	$2{,}49e-01_{1{,}0e-01}$	$3{,}76e-01_{9{,}9e-02}$	$3{,}76e-02_{6{,}4e-02}$	$2{,}99e-01_{9{,}5e-02}$	$3{,}90e-01_{9{,}2e-02}$
BB11011	$3{,}96e-01_{6{,}3e-02}$	$3{,}13e-01_{7{,}4e-02}$	$3{,}73e-01_{5{,}4e-02}$	$3{,}11e-02_{3{,}9e-02}$	$3{,}64e-01_{9{,}4e-02}$	$8{,}70e-02_{6{,}3e-02}$
BB11012	$2{,}28e-01_{1{,}9e-01}$	$1{,}24e-01_{8{,}3e-02}$	$1{,}36e-01_{1{,}4e-01}$	$4{,}23e-03_{1{,}2e-02}$	$9{,}81e-02_{1{,}7e-01}$	$1{,}97e-01_{1{,}5e-01}$
BB11013	$3{,}32e-01_{1{,}5e-01}$	$2{,}57e-01_{1{,}5e-01}$	$2{,}45e-01_{1{,}5e-01}$	$0{,}00e+00_{0{,}0e+00}$	$2{,}37e-01_{1{,}4e-01}$	$3{,}84e-02_{5{,}1e-02}$
BB11014	$3{,}63e-01_{1{,}4e-01}$	$1{,}74e-01_{9{,}4e-02}$	$2{,}25e-01_{6{,}4e-02}$	$0{,}00e+00_{0{,}0e+00}$	$3{,}15e-01_{4{,}9e-02}$	$3{,}49e-02_{9{,}1e-02}$
BB11015	$4{,}00e-01_{1{,}6e-01}$	$3{,}44e-01_{1{,}3e-01}$	$3{,}60e-01_{1{,}3e-01}$	$0{,}00e+00_{0{,}0e+00}$	$3{,}94e-01_{1{,}6e-01}$	$3{,}84e-01_{3{,}3e-01}$
BB11016	$3{,}95e-01_{4{,}6e-02}$	$3{,}52e-01_{5{,}4e-02}$	$3{,}47e-01_{5{,}9e-02}$	$2{,}52e-02_{4{,}8e-03}$	$3{,}52e-01_{5{,}7e-02}$	$3{,}79e-02_{5{,}0e-02}$
BB11017	$2{,}57e-01_{9{,}1e-02}$	$2{,}27e-01_{6{,}4e-02}$	$3{,}60e-01_{9{,}1e-02}$	$4{,}42e-02_{9{,}0e-02}$	$3{,}19e-01_{9{,}8e-02}$	$3{,}67e-01_{1{,}6e-01}$
BB11018	$3{,}85e-01_{4{,}5e-02}$	$3{,}71e-01_{6{,}0e-03}$	$3{,}27e-01_{8{,}0e-03}$	$2{,}93e-02_{9{,}9e-02}$	$2{,}25e-01_{2{,}6e-02}$	$4{,}95e-02_{8{,}2e-02}$
BB11019	$3{,}28e-01_{5{,}7e-02}$	$3{,}54e-01_{6{,}3e-02}$	$2{,}93e-01_{6{,}5e-02}$	$7{,}59e-02_{3{,}4e-02}$	$3{,}43e-01_{3{,}3e-02}$	$1{,}63e-01_{3{,}7e-02}$
BB11020	$3{,}22e-01_{1{,}3e-02}$	$3{,}06e-01_{1{,}5e-02}$	$2{,}93e-01_{1{,}2e-02}$	$4{,}87e-02_{3{,}5e-02}$	$3{,}03e-01_{1{,}3e-02}$	$1{,}66e-01_{7{,}6e-02}$
BB11021	$4{,}58e-01_{8{,}7e-02}$	$3{,}53e-01_{7{,}3e-02}$	$4{,}57e-01_{1{,}0e-01}$	$2{,}31e-01_{6{,}1e-02}$	$4{,}62e-01_{1{,}0e-01}$	$4{,}86e-01_{6{,}6e-02}$
BB11022	$3{,}94e-01_{9{,}6e-02}$	$2{,}23e-01_{4{,}4e-02}$	$3{,}14e-01_{1{,}0e-01}$	$0{,}00e+00_{0{,}0e+00}$	$2{,}72e-01_{7{,}7e-02}$	$1{,}36e-01_{2{,}0e-02}$
BB11023	$3{,}54e-01_{4{,}6e-02}$	$2{,}95e-01_{3{,}3e-02}$	$2{,}97e-01_{2{,}4e-02}$	$2{,}83e-02_{2{,}9e-02}$	$3{,}19e-01_{4{,}3e-02}$	$4{,}05e-02_{2{,}0e-02}$
BB11024	$4{,}21e-01_{1{,}1e-01}$	$2{,}80e-01_{1{,}5e-01}$	$3{,}41e-01_{6{,}6e-02}$	$0{,}00e+00_{2{,}8e-03}$	$2{,}99e-01_{1{,}1e-01}$	$2{,}10e-01_{8{,}1e-02}$
BB11025	$5{,}29e-01_{1{,}6e-01}$	$2{,}92e-01_{1{,}4e-01}$	$4{,}29e-01_{1{,}2e-01}$	$0{,}00e+00_{1{,}5e-02}$	$3{,}26e-01_{1{,}2e-01}$	$2{,}29e-01_{1{,}6e-01}$
BB11027	$3{,}19e-01_{3{,}0e-02}$	$3{,}28e-01_{2{,}6e-02}$	$2{,}85e-01_{2{,}5e-02}$	$7{,}37e-02_{4{,}7e-03}$	$2{,}96e-01_{5{,}3e-02}$	$2{,}01e-01_{3{,}0e-02}$
BB11028	$1{,}72e-01_{1{,}3e-02}$	$1{,}64e-01_{1{,}0e-02}$	$1{,}64e-01_{1{,}4e-01}$	$0{,}00e+00_{0{,}0e+00}$	$2{,}68e-01_{1{,}4e-01}$	$9{,}07e-02_{9{,}4e-02}$
BB11029	$4{,}37e-01_{7{,}5e-02}$	$3{,}82e-01_{4{,}0e-02}$	$4{,}22e-01_{3{,}5e-02}$	$1{,}82e-01_{1{,}2e-01}$	$3{,}85e-01_{5{,}3e-02}$	$4{,}34e-01_{7{,}3e-02}$
BB11031	$3{,}74e-01_{7{,}0e-02}$	$3{,}17e-01_{5{,}3e-02}$	$3{,}23e-01_{9{,}8e-03}$	$0{,}00e+00_{0{,}0e+00}$	$2{,}82e-01_{1{,}9e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB11032	$4{,}31e-01_{2{,}1e-02}$	$4{,}06e-01_{2{,}7e-02}$	$4{,}05e-01_{2{,}5e-02}$	$1{,}84e-01_{5{,}0e-02}$	$4{,}03e-01_{2{,}7e-02}$	$3{,}56e-01_{1{,}7e-02}$
BB11033	$3{,}38e-01_{1{,}2e-02}$	$3{,}12e-01_{1{,}4e-02}$	$3{,}06e-01_{1{,}2e-02}$	$0{,}00e+00_{0{,}0e+00}$	$3{,}05e-01_{8{,}6e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB11034	$3{,}49e-01_{1{,}4e-02}$	$3{,}27e-01_{1{,}8e-02}$	$3{,}12e-01_{2{,}8e-02}$	$1{,}13e-02_{3{,}3e-02}$	$3{,}17e-01_{2{,}3e-02}$	$8{,}96e-02_{8{,}7e-02}$
BB11035	$4{,}99e-01_{6{,}6e-02}$	$4{,}36e-01_{4{,}4e-02}$	$4{,}26e-01_{3{,}2e-02}$	$9{,}05e-02_{1{,}1e-01}$	$4{,}30e-01_{4{,}1e-02}$	$3{,}72e-01_{6{,}5e-02}$
BB11036	$3{,}28e-01_{6{,}0e-02}$	$2{,}78e-01_{3{,}9e-02}$	$2{,}80e-01_{2{,}3e-02}$	$7{,}65e-02_{3{,}8e-02}$	$2{,}86e-01_{3{,}0e-02}$	$1{,}08e-01_{4{,}7e-02}$
BB11037	$5{,}10e-01_{9{,}7e-02}$	$4{,}45e-01_{7{,}6e-02}$	$3{,}81e-01_{1{,}0e-01}$	$9{,}51e-02_{1{,}5e-01}$	$4{,}03e-01_{9{,}6e-02}$	$4{,}08e-01_{9{,}3e-02}$
BB11038	$4{,}75e-01_{2{,}6e-02}$	$4{,}49e-01_{1{,}8e-02}$	$4{,}27e-01_{1{,}9e-02}$	$2{,}53e-01_{7{,}6e-02}$	$4{,}43e-01_{2{,}0e-02}$	$3{,}17e-01_{4{,}4e-02}$

Tabla B.2: Mediana y rango intercuartílico del indicador I_{HV} para los problemas de la familia RV12 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB12001	$4{,}61e-01_{4{,}3e-02}$	$4{,}27e-01_{3{,}3e-02}$	$4{,}17e-01_{3{,}8e-02}$	$2{,}14e-01_{3{,}7e-02}$	$4{,}31e-01_{4{,}8e-02}$	$2{,}10e-01_{3{,}8e-02}$
BB12002	$1{,}67e-01_{1{,}6e-01}$	$1{,}25e-01_{1{,}8e-01}$	$7{,}67e-02_{1{,}4e-01}$	$0{,}00e+00_{0{,}0e+00}$	$8{,}05e-02_{1{,}6e-01}$	$2{,}40e-01_{2{,}0e-01}$
BB12003	$2{,}95e-01_{5{,}8e-02}$	$2{,}97e-01_{4{,}5e-02}$	$3{,}03e-01_{4{,}6e-02}$	$2{,}14e-01_{3{,}8e-02}$	$2{,}87e-01_{6{,}4e-02}$	$3{,}25e-01_{4{,}8e-02}$
BB12004	$4{,}42e-01_{6{,}4e-02}$	$4{,}24e-01_{8{,}7e-02}$	$4{,}26e-01_{4{,}7e-02}$	$2{,}84e-01_{1{,}4e-02}$	$4{,}09e-01_{3{,}9e-02}$	$3{,}09e-01_{5{,}3e-02}$
BB12005	$4{,}23e-01_{1{,}1e-01}$	$4{,}11e-01_{1{,}1e-01}$	$3{,}67e-01_{1{,}1e-01}$	$1{,}04e-01_{6{,}1e-02}$	$3{,}48e-01_{1{,}1e-01}$	$1{,}80e-01_{1{,}2e-01}$
BB12006	$0{,}00e+00_{9{,}8e-03}$	$0{,}00e+00_{0{,}0e+00}$	$3{,}27e-03_{5{,}1e-03}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}12e-03_{5{,}0e-03}$	$1{,}19e-03_{2{,}0e-02}$
BB12007	$5{,}97e-01_{1{,}7e-02}$	$5{,}63e-01_{1{,}2e-02}$	$5{,}72e-01_{1{,}4e-02}$	$2{,}54e-01_{4{,}3e-02}$	$5{,}91e-01_{2{,}1e-02}$	$3{,}09e-01_{1{,}0e-01}$
BB12008	$1{,}20e-01_{9{,}0e-03}$	$1{,}10e-01_{1{,}6e-02}$	$9{,}91e-02_{8{,}5e-03}$	$8{,}36e-02_{5{,}0e-02}$	$1{,}03e-01_{5{,}3e-03}$	$1{,}05e-01_{1{,}0e-02}$
BB12009	$5{,}24e-01_{4{,}2e-02}$	$4{,}58e-01_{5{,}0e-02}$	$4{,}47e-01_{5{,}6e-02}$	$1{,}68e-01_{6{,}0e-02}$	$4{,}54e-01_{5{,}8e-02}$	$4{,}14e-01_{3{,}3e-02}$
BB12010	$4{,}47e-01_{4{,}2e-02}$	$3{,}84e-01_{2{,}7e-02}$	$4{,}29e-01_{3{,}2e-02}$	$1{,}32e-01_{5{,}5e-02}$	$4{,}16e-01_{4{,}2e-02}$	$2{,}38e-01_{8{,}9e-02}$
BB12011	$3{,}37e-01_{2{,}1e-02}$	$3{,}31e-01_{4{,}4e-02}$	$3{,}14e-01_{1{,}6e-02}$	$1{,}95e-01_{2{,}1e-02}$	$3{,}09e-01_{2{,}0e-02}$	$1{,}92e-01_{1{,}0e-01}$
BB12012	$5{,}45e-01_{5{,}5e-02}$	$4{,}58e-01_{6{,}3e-02}$	$5{,}12e-01_{4{,}5e-02}$	$1{,}65e-01_{1{,}3e-01}$	$4{,}82e-01_{4{,}5e-02}$	$5{,}27e-01_{6{,}8e-02}$
BB12013	$5{,}29e-01_{5{,}0e-02}$	$5{,}12e-01_{3{,}9e-02}$	$5{,}06e-01_{3{,}8e-02}$	$1{,}95e-01_{7{,}8e-02}$	$5{,}19e-01_{5{,}6e-02}$	$3{,}37e-01_{7{,}4e-02}$
BB12014	$5{,}04e-01_{7{,}9e-02}$	$4{,}98e-01_{8{,}1e-02}$	$4{,}44e-01_{1{,}1e-01}$	$2{,}17e-01_{1{,}4e-01}$	$4{,}61e-01_{9{,}2e-02}$	$3{,}52e-01_{9{,}3e-02}$
BB12015	$2{,}21e-01_{3{,}5e-02}$	$2{,}38e-01_{3{,}4e-02}$	$2{,}07e-01_{2{,}6e-02}$	$4{,}50e-02_{1{,}4e-01}$	$1{,}94e-01_{1{,}7e-02}$	$1{,}65e-01_{2{,}9e-02}$
BB12016	$4{,}60e-01_{3{,}0e-02}$	$5{,}00e-01_{3{,}4e-02}$	$4{,}27e-01_{6{,}5e-02}$	$3{,}30e-01_{4{,}8e-03}$	$4{,}45e-01_{5{,}1e-02}$	$3{,}65e-01_{8{,}8e-03}$
BB12017	$4{,}51e-01_{8{,}1e-02}$	$4{,}18e-01_{7{,}1e-02}$	$4{,}24e-01_{5{,}7e-02}$	$2{,}24e-01_{5{,}3e-02}$	$4{,}07e-01_{7{,}3e-02}$	$3{,}40e-01_{9{,}6e-02}$
BB12018	$4{,}96e-01_{1{,}1e-01}$	$4{,}43e-01_{6{,}7e-02}$	$4{,}89e-01_{8{,}2e-02}$	$3{,}62e-01_{5{,}6e-02}$	$4{,}83e-01_{5{,}9e-02}$	$4{,}64e-01_{8{,}9e-02}$
BB12019	$6{,}41e-01_{2{,}5e-02}$	$5{,}80e-01_{6{,}1e-02}$	$6{,}26e-01_{3{,}2e-02}$	$1{,}72e-01_{1{,}7e-01}$	$6{,}36e-01_{1{,}8e-02}$	$5{,}57e-01_{5{,}6e-02}$
BB12020	$4{,}38e-01_{1{,}6e-01}$	$2{,}67e-01_{1{,}8e-01}$	$4{,}01e-01_{8{,}5e-02}$	$2{,}13e-01_{1{,}8e-01}$	$3{,}69e-01_{1{,}0e-01}$	$4{,}38e-01_{3{,}6e-02}$
BB12021	$3{,}75e-01_{3{,}3e-02}$	$3{,}60e-01_{4{,}1e-02}$	$3{,}58e-01_{5{,}0e-02}$	$1{,}10e-01_{1{,}0e-01}$	$3{,}61e-01_{4{,}0e-02}$	$3{,}63e-01_{1{,}1e-01}$
BB12022	$4{,}22e-01_{1{,}4e-02}$	$4{,}14e-01_{4{,}3e-02}$	$4{,}03e-01_{1{,}1e-02}$	$2{,}71e-01_{2{,}7e-02}$	$4{,}12e-01_{1{,}6e-02}$	$3{,}46e-01_{3{,}6e-02}$
BB12023	$4{,}09e-01_{7{,}0e-02}$	$3{,}81e-01_{4{,}5e-02}$	$4{,}56e-01_{6{,}8e-02}$	$0{,}00e+00_{3{,}3e-02}$	$4{,}39e-01_{6{,}9e-02}$	$4{,}45e-01_{5{,}5e-02}$
BB12024	$3{,}02e-01_{9{,}5e-02}$	$3{,}09e-01_{1{,}6e-01}$	$3{,}51e-01_{1{,}2e-01}$	$2{,}50e-01_{3{,}4e-02}$	$3{,}37e-01_{1{,}1e-01}$	$3{,}37e-01_{1{,}6e-01}$
BB12025	$5{,}32e-01_{6{,}1e-02}$	$4{,}90e-01_{7{,}5e-02}$	$5{,}26e-01_{1{,}1e-01}$	$2{,}25e-01_{7{,}9e-02}$	$5{,}14e-01_{5{,}2e-02}$	$4{,}25e-01_{1{,}2e-01}$
BB12026	$1{,}43e-01_{2{,}5e-02}$	$1{,}35e-01_{2{,}0e-02}$	$1{,}18e-01_{7{,}5e-03}$	$1{,}32e-01_{3{,}4e-02}$	$1{,}24e-01_{1{,}2e-02}$	$1{,}64e-01_{4{,}5e-02}$
BB12027	$2{,}38e-01_{2{,}1e-02}$	$3{,}11e-01_{7{,}9e-02}$	$2{,}21e-01_{1{,}8e-02}$	$1{,}52e-01_{7{,}7e-02}$	$2{,}25e-01_{2{,}1e-02}$	$2{,}43e-01_{2{,}5e-02}$
BB12028	$4{,}32e-01_{6{,}3e-03}$	$4{,}19e-01_{7{,}2e-03}$	$4{,}16e-01_{8{,}0e-03}$	$9{,}50e-02_{7{,}0e-02}$	$4{,}22e-01_{1{,}1e-02}$	$2{,}03e-01_{3{,}5e-02}$
BB12029	$2{,}43e-01_{1{,}4e-02}$	$2{,}19e-01_{2{,}1e-02}$	$2{,}22e-01_{1{,}1e-02}$	$1{,}84e-01_{1{,}3e-02}$	$2{,}21e-01_{1{,}1e-02}$	$2{,}04e-01_{1{,}8e-02}$
BB12030	$5{,}43e-01_{3{,}1e-02}$	$5{,}37e-01_{5{,}6e-02}$	$5{,}23e-01_{2{,}8e-02}$	$3{,}10e-01_{7{,}7e-02}$	$5{,}04e-01_{3{,}8e-02}$	$4{,}26e-01_{3{,}7e-02}$
BB12031	$4{,}19e-01_{7{,}3e-03}$	$3{,}92e-01_{9{,}2e-03}$	$3{,}93e-01_{8{,}4e-03}$	$1{,}98e-01_{3{,}3e-02}$	$3{,}72e-01_{1{,}0e-02}$	$1{,}84e-01_{2{,}9e-02}$
BB12032	$3{,}79e-01_{4{,}6e-02}$	$3{,}98e-01_{4{,}1e-02}$	$3{,}55e-01_{3{,}7e-02}$	$2{,}09e-01_{1{,}2e-01}$	$3{,}49e-01_{3{,}5e-02}$	$3{,}02e-01_{1{,}7e-02}$
BB12033	$4{,}69e-01_{1{,}6e-02}$	$4{,}52e-01_{1{,}8e-02}$	$4{,}51e-01_{9{,}4e-03}$	$1{,}68e-01_{7{,}7e-02}$	$4{,}54e-01_{1{,}8e-02}$	$3{,}25e-01_{1{,}9e-02}$
BB12034	$1{,}13e-01_{1{,}3e-01}$	$1{,}36e-01_{1{,}1e-01}$	$1{,}08e-01_{6{,}6e-02}$	$0{,}00e+00_{0{,}0e+00}$	$9{,}34e-02_{1{,}2e-01}$	$1{,}43e-01_{6{,}9e-02}$
BB12035	$5{,}27e-02_{3{,}0e-03}$	$5{,}10e-02_{1{,}3e-02}$	$4{,}61e-02_{2{,}1e-03}$	$1{,}92e-02_{4{,}0e-02}$	$4{,}37e-02_{2{,}9e-03}$	$3{,}62e-02_{4{,}7e-02}$
BB12036	$3{,}70e-01_{7{,}6e-02}$	$3{,}08e-01_{6{,}6e-02}$	$3{,}81e-01_{8{,}9e-02}$	$5{,}88e-02_{1{,}4e-01}$	$3{,}48e-01_{1{,}0e-01}$	$3{,}55e-01_{1{,}5e-01}$
BB12037	$1{,}18e-01_{1{,}1e-02}$	$1{,}09e-01_{8{,}2e-03}$	$1{,}11e-01_{2{,}8e-02}$	$7{,}03e-02_{6{,}0e-02}$	$1{,}04e-01_{1{,}8e-02}$	$8{,}66e-02_{9{,}2e-02}$
BB12038	$3{,}18e-01_{7{,}9e-02}$	$3{,}54e-01_{1{,}9e-01}$	$2{,}74e-01_{3{,}7e-02}$	$2{,}67e-01_{8{,}2e-02}$	$2{,}67e-01_{4{,}8e-02}$	$4{,}39e-01_{1{,}4e-01}$
BB12039	$3{,}98e-01_{3{,}7e-02}$	$3{,}93e-01_{3{,}8e-02}$	$3{,}75e-01_{3{,}4e-02}$	$2{,}10e-01_{8{,}1e-02}$	$3{,}69e-01_{2{,}9e-02}$	$2{,}36e-01_{3{,}7e-02}$
BB12040	$5{,}03e-01_{4{,}4e-02}$	$4{,}43e-01_{7{,}2e-02}$	$4{,}74e-01_{1{,}5e-01}$	$2{,}45e-01_{1{,}5e-01}$	$4{,}35e-01_{8{,}3e-02}$	$5{,}27e-01_{3{,}1e-02}$
BB12041	$3{,}40e-01_{8{,}3e-02}$	$3{,}25e-01_{7{,}1e-02}$	$3{,}40e-01_{4{,}2e-02}$	$2{,}16e-01_{6{,}1e-02}$	$3{,}52e-01_{6{,}8e-02}$	$2{,}98e-01_{1{,}6e-02}$
BB12042	$2{,}42e-01_{1{,}5e-01}$	$1{,}67e-01_{9{,}8e-02}$	$1{,}99e-01_{1{,}7e-01}$	$0{,}00e+00_{0{,}0e+00}$	$2{,}46e-01_{1{,}7e-01}$	$2{,}97e-01_{2{,}6e-01}$
BB12043	$1{,}55e-01_{9{,}1e-03}$	$1{,}50e-01_{6{,}9e-03}$	$1{,}58e-01_{9{,}2e-03}$	$1{,}10e-02_{1{,}1e-01}$	$1{,}07e-01_{6{,}4e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB12044	$4{,}24e-01_{7{,}7e-02}$	$4{,}16e-01_{6{,}0e-02}$	$4{,}33e-01_{7{,}3e-02}$	$3{,}79e-01_{8{,}8e-02}$	$3{,}96e-01_{3{,}6e-02}$	$3{,}26e-01_{6{,}8e-02}$

Tabla B.3: Mediana y rango intercuartílico del indicador I_{HV} para los problemas de la familia RV20 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB20001	$1{,}18e-01_{2{,}8e-02}$	$1{,}57e-01_{1{,}4e-01}$	$1{,}02e-01_{8{,}1e-02}$	$0{,}00e+00_{0{,}0e+00}$	$8{,}31e-02_{7{,}8e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB20002	$3{,}53e-01_{1{,}1e-02}$	$3{,}04e-01_{3{,}9e-02}$	$3{,}16e-01_{4{,}5e-02}$	$2{,}11e-01_{1{,}1e-02}$	$2{,}11e-01_{1{,}1e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB20003	$1{,}58e-01_{2{,}9e-02}$	$1{,}38e-01_{3{,}3e-02}$	$1{,}22e-01_{1{,}0e-01}$	$8{,}53e-02_{9{,}1e-02}$	$7{,}81e-02_{1{,}6e-02}$	$8{,}53e-02_{9{,}1e-02}$
BB20004	$3{,}92e-01_{6{,}0e-02}$	$3{,}24e-01_{3{,}6e-02}$	$3{,}37e-01_{5{,}1e-02}$	$2{,}37e-01_{2{,}0e-02}$	$2{,}37e-01_{2{,}0e-02}$	$2{,}89e-01_{1{,}0e-01}$
BB20005	$8{,}09e-02_{3{,}1e-03}$	$7{,}46e-02_{3{,}8e-03}$	$7{,}37e-02_{2{,}9e-03}$	$7{,}11e-02_{3{,}6e-02}$	$3{,}90e-02_{1{,}2e-03}$	$7{,}11e-02_{3{,}6e-02}$
BB20006	$3{,}08e-02_{3{,}1e-03}$	$2{,}89e-02_{2{,}8e-03}$	$2{,}87e-02_{2{,}5e-03}$	$2{,}70e-02_{1{,}3e-03}$	$2{,}70e-02_{1{,}3e-03}$	$0{,}00e+00_{2{,}7e-02}$
BB20007	$1{,}30e-01_{4{,}7e-03}$	$1{,}24e-01_{1{,}2e-02}$	$1{,}17e-01_{5{,}8e-03}$	$9{,}02e-02_{5{,}8e-03}$	$9{,}02e-02_{5{,}8e-03}$	$1{,}83e-02_{4{,}9e-02}$
BB20008	$3{,}76e-01_{1{,}3e-02}$	$3{,}73e-01_{1{,}7e-02}$	$3{,}73e-01_{1{,}7e-02}$	$3{,}73e-01_{1{,}7e-02}$	$3{,}73e-01_{1{,}7e-02}$	$3{,}73e-01_{1{,}7e-02}$
BB20009	$5{,}25e-02_{1{,}1e-02}$	$4{,}41e-02_{8{,}8e-03}$	$4{,}63e-02_{5{,}7e-03}$	$4{,}17e-02_{4{,}5e-03}$	$4{,}17e-02_{4{,}5e-03}$	$0{,}00e+00_{8{,}9e-03}$
BB20010	$9{,}35e-02_{1{,}2e-02}$	$9{,}05e-02_{1{,}3e-02}$	$8{,}66e-02_{6{,}9e-03}$	$2{,}82e-02_{3{,}5e-02}$	$6{,}99e-02_{2{,}2e-03}$	$2{,}82e-02_{3{,}5e-02}$
BB20011	$2{,}14e-01_{1{,}9e-02}$	$2{,}10e-01_{1{,}0e-02}$	$2{,}14e-01_{1{,}6e-02}$	$1{,}85e-01_{1{,}6e-02}$	$1{,}85e-01_{1{,}6e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB20012	$1{,}43e-01_{1{,}5e-02}$	$1{,}76e-01_{2{,}9e-02}$	$1{,}26e-01_{1{,}4e-02}$	$1{,}76e-01_{2{,}9e-02}$	$8{,}26e-02_{1{,}4e-02}$	$7{,}76e-02_{1{,}1e-01}$
BB20013	$1{,}23e-01_{1{,}7e-02}$	$1{,}44e-01_{2{,}1e-02}$	$1{,}08e-01_{1{,}5e-02}$	$7{,}67e-02_{6{,}7e-02}$	$7{,}51e-02_{3{,}1e-03}$	$7{,}67e-02_{6{,}7e-02}$
BB20014	$1{,}49e-01_{9{,}9e-03}$	$1{,}29e-01_{9{,}9e-03}$	$1{,}40e-01_{7{,}8e-03}$	$1{,}29e-01_{9{,}9e-03}$	$8{,}33e-02_{3{,}6e-03}$	$8{,}53e-02_{5{,}9e-02}$
BB20015	$1{,}25e-01_{6{,}3e-03}$	$1{,}11e-01_{5{,}8e-03}$	$1{,}17e-01_{5{,}2e-03}$	$1{,}11e-01_{5{,}8e-03}$	$6{,}27e-02_{1{,}2e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB20016	$9{,}50e-02_{8{,}3e-02}$	$9{,}30e-02_{9{,}6e-03}$	$8{,}79e-02_{4{,}1e-02}$	$9{,}30e-02_{9{,}6e-03}$	$6{,}55e-02_{5{,}3e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB20017	$6{,}17e-02_{4{,}9e-03}$	$7{,}05e-02_{1{,}2e-02}$	$5{,}65e-02_{8{,}9e-03}$	$4{,}20e-02_{1{,}6e-02}$	$4{,}05e-02_{3{,}3e-03}$	$4{,}20e-02_{1{,}6e-02}$
BB20018	$2{,}58e-02_{3{,}7e-03}$	$2{,}12e-02_{3{,}8e-03}$	$2{,}09e-02_{2{,}7e-03}$	$2{,}12e-02_{3{,}8e-03}$	$2{,}11e-02_{2{,}4e-03}$	$1{,}91e-02_{2{,}1e-02}$
BB20019	$8{,}96e-02_{4{,}7e-03}$	$8{,}15e-02_{2{,}1e-02}$	$8{,}02e-02_{6{,}9e-03}$	$8{,}15e-02_{2{,}1e-02}$	$6{,}41e-02_{4{,}7e-03}$	$5{,}92e-02_{7{,}9e-02}$
BB20020	$9{,}84e-02_{5{,}4e-03}$	$1{,}08e-01_{2{,}7e-02}$	$8{,}95e-02_{1{,}0e-02}$	$8{,}95e-02_{1{,}0e-02}$	$8{,}53e-02_{9{,}2e-03}$	$7{,}63e-02_{1{,}4e-02}$
BB20021	$2{,}73e-01_{2{,}2e-02}$	$2{,}59e-01_{2{,}6e-02}$	$2{,}43e-01_{2{,}7e-02}$	$2{,}43e-01_{2{,}7e-02}$	$1{,}69e-01_{4{,}2e-03}$	$2{,}05e-01_{7{,}3e-02}$
BB20022	$4{,}84e-02_{5{,}0e-03}$	$3{,}87e-02_{3{,}2e-03}$	$3{,}59e-02_{4{,}2e-03}$	$3{,}40e-02_{8{,}7e-03}$	$3{,}06e-02_{2{,}5e-03}$	$3{,}40e-02_{8{,}7e-03}$
BB20023	$7{,}16e-02_{2{,}0e-03}$	$6{,}84e-02_{2{,}9e-03}$	$6{,}67e-02_{1{,}7e-03}$	$6{,}67e-02_{1{,}7e-03}$	$4{,}83e-02_{2{,}2e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB20024	$1{,}79e-01_{5{,}9e-03}$	$1{,}60e-01_{2{,}5e-02}$	$1{,}64e-01_{2{,}1e-02}$	$9{,}45e-02_{1{,}4e-01}$	$9{,}45e-02_{5{,}2e-03}$	$9{,}45e-02_{1{,}4e-01}$
BB20025	$2{,}25e-01_{3{,}9e-02}$	$2{,}36e-01_{5{,}9e-02}$	$2{,}06e-01_{4{,}2e-02}$	$2{,}06e-01_{4{,}2e-02}$	$1{,}30e-01_{7{,}3e-03}$	$1{,}86e-01_{8{,}1e-02}$
BB20026	$1{,}30e-01_{1{,}4e-02}$	$1{,}25e-01_{1{,}7e-02}$	$1{,}24e-01_{1{,}0e-02}$	$1{,}24e-01_{1{,}0e-02}$	$9{,}01e-02_{6{,}7e-03}$	$9{,}34e-02_{1{,}2e-01}$
BB20027	$1{,}79e-01_{5{,}9e-03}$	$1{,}68e-01_{5{,}3e-03}$	$1{,}67e-01_{4{,}9e-03}$	$1{,}67e-01_{4{,}9e-03}$	$1{,}11e-01_{5{,}1e-03}$	$1{,}15e-01_{1{,}1e-01}$
BB20028	$7{,}72e-02_{9{,}1e-03}$	$6{,}48e-02_{1{,}0e-02}$	$6{,}90e-02_{2{,}6e-02}$	$6{,}90e-02_{2{,}6e-02}$	$6{,}15e-02_{9{,}4e-03}$	$6{,}24e-02_{2{,}1e-02}$
BB20030	$4{,}61e-02_{3{,}8e-02}$	$4{,}34e-02_{3{,}8e-02}$	$3{,}97e-02_{2{,}1e-02}$	$4{,}34e-02_{3{,}8e-02}$	$2{,}79e-02_{1{,}5e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB20031	$4{,}02e-02_{5{,}0e-03}$	$3{,}89e-02_{6{,}5e-03}$	$3{,}43e-02_{6{,}7e-03}$	$0{,}00e+00_{0{,}0e+00}$	$2{,}39e-02_{1{,}1e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB20032	$2{,}61e-02_{3{,}7e-03}$	$2{,}62e-02_{1{,}1e-02}$	$2{,}41e-02_{8{,}7e-03}$	$2{,}62e-02_{1{,}1e-02}$	$1{,}75e-02_{7{,}9e-04}$	$0{,}00e+00_{0{,}0e+00}$
BB20033	$4{,}73e-02_{3{,}1e-03}$	$4{,}14e-02_{6{,}9e-03}$	$4{,}20e-02_{4{,}2e-03}$	$4{,}14e-02_{6{,}9e-03}$	$3{,}86e-02_{9{,}5e-04}$	$0{,}00e+00_{0{,}0e+00}$
BB20034	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$
BB20035	$6{,}27e-02_{3{,}6e-03}$	$5{,}96e-02_{4{,}2e-03}$	$5{,}79e-02_{3{,}2e-03}$	$5{,}96e-02_{4{,}2e-03}$	$4{,}04e-02_{2{,}3e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB20036	$7{,}47e-02_{5{,}8e-03}$	$6{,}83e-02_{4{,}2e-03}$	$6{,}82e-02_{4{,}9e-03}$	$0{,}00e+00_{0{,}0e+00}$	$3{,}92e-02_{1{,}3e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB20037	$3{,}63e-01_{5{,}2e-02}$	$3{,}23e-01_{3{,}3e-02}$	$3{,}10e-01_{1{,}6e-02}$	$3{,}23e-01_{3{,}3e-02}$	$2{,}17e-01_{1{,}5e-02}$	$2{,}17e-01_{1{,}5e-02}$
BB20038	$6{,}70e-02_{6{,}4e-03}$	$5{,}96e-02_{4{,}5e-03}$	$6{,}16e-02_{4{,}6e-03}$	$5{,}96e-02_{4{,}5e-03}$	$5{,}30e-02_{3{,}9e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB20039	$1{,}53e-01_{3{,}7e-03}$	$1{,}40e-01_{8{,}7e-03}$	$1{,}39e-01_{8{,}4e-03}$	$1{,}31e-01_{7{,}9e-02}$	$6{,}09e-02_{7{,}0e-03}$	$1{,}31e-01_{7{,}9e-02}$
BB20040	$3{,}17e-01_{1{,}4e-02}$	$3{,}03e-01_{1{,}3e-02}$	$3{,}07e-01_{1{,}3e-02}$	$3{,}03e-01_{1{,}3e-02}$	$1{,}17e-01_{1{,}7e-02}$	$2{,}97e-01_{2{,}0e-01}$
BB20041	$2{,}13e-01_{2{,}7e-03}$	$2{,}12e-01_{3{,}3e-03}$	$2{,}11e-01_{5{,}2e-03}$	$2{,}12e-01_{3{,}3e-03}$	$2{,}08e-01_{1{,}8e-01}$	$2{,}11e-01_{5{,}2e-03}$

Tabla B.4: Mediana y rango intercuartílico del indicador I_{HV} para los problemas de la familia RV30 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB30001	$1{,}76e-01_{2{,}2e-02}$	$1{,}56e-01_{3{,}6e-02}$	$2{,}01e-01_{5{,}1e-03}$	$6{,}62e-02_{3{,}2e-02}$	$6{,}62e-02_{3{,}2e-02}$	$1{,}27e-01_{3{,}3e-02}$
BB30002	$7{,}09e-02_{6{,}8e-03}$	$7{,}11e-02_{7{,}9e-03}$	$6{,}94e-02_{9{,}8e-03}$	$0{,}00e+00_{0{,}0e+00}$	$5{,}54e-02_{3{,}3e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB30003	$1{,}37e-01_{5{,}3e-03}$	$1{,}30e-01_{7{,}8e-03}$	$1{,}34e-01_{7{,}5e-03}$	$1{,}34e-01_{7{,}5e-03}$	$2{,}50e-02_{1{,}4e-02}$	$1{,}28e-01_{5{,}6e-03}$
BB30004	$5{,}25e-02_{9{,}8e-03}$	$5{,}99e-02_{1{,}2e-02}$	$5{,}18e-02_{7{,}6e-03}$	$3{,}71e-02_{6{,}7e-03}$	$3{,}78e-02_{4{,}4e-03}$	$3{,}71e-02_{6{,}7e-03}$
BB30005	$1{,}99e-01_{2{,}0e-02}$	$2{,}06e-01_{2{,}7e-02}$	$1{,}83e-01_{1{,}1e-02}$	$2{,}06e-01_{2{,}7e-02}$	$1{,}04e-01_{2{,}3e-03}$	$2{,}01e-01_{1{,}8e-02}$
BB30006	$9{,}32e-02_{1{,}6e-02}$	$1{,}17e-01_{2{,}8e-02}$	$7{,}91e-02_{8{,}3e-03}$	$0{,}00e+00_{0{,}0e+00}$	$8{,}11e-02_{4{,}8e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB30007	$9{,}10e-02_{1{,}7e-02}$	$9{,}82e-02_{2{,}3e-02}$	$8{,}51e-02_{2{,}3e-02}$	$0{,}00e+00_{0{,}0e+00}$	$8{,}56e-02_{1{,}6e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB30008	$6{,}74e-02_{1{,}3e-02}$	$6{,}99e-02_{1{,}6e-02}$	$6{,}47e-02_{1{,}5e-02}$	$0{,}00e+00_{0{,}0e+00}$	$5{,}47e-02_{8{,}8e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB30010	$5{,}65e-02_{7{,}7e-03}$	$5{,}36e-02_{6{,}5e-03}$	$5{,}30e-02_{4{,}5e-03}$	$4{,}90e-02_{1{,}6e-02}$	$3{,}81e-02_{2{,}2e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB30011	$3{,}73e-01_{2{,}9e-02}$	$3{,}35e-01_{2{,}0e-02}$	$3{,}44e-01_{3{,}3e-02}$	$3{,}15e-01_{9{,}4e-02}$	$2{,}50e-01_{3{,}9e-03}$	$2{,}51e-01_{4{,}0e-03}$
BB30012	$2{,}50e-01_{4{,}0e-02}$	$2{,}69e-01_{8{,}1e-02}$	$2{,}25e-01_{2{,}4e-02}$	$1{,}95e-01_{6{,}2e-02}$	$1{,}65e-01_{1{,}5e-02}$	$1{,}95e-01_{6{,}2e-02}$
BB30013	$2{,}14e-01_{1{,}1e-02}$	$2{,}08e-01_{1{,}7e-02}$	$2{,}03e-01_{6{,}6e-03}$	$1{,}94e-01_{1{,}3e-01}$	$6{,}67e-02_{3{,}6e-02}$	$1{,}94e-01_{1{,}3e-01}$
BB30014	$5{,}64e-02_{3{,}5e-03}$	$5{,}73e-02_{8{,}1e-03}$	$5{,}03e-02_{2{,}9e-03}$	$0{,}00e+00_{0{,}0e+00}$	$3{,}94e-02_{1{,}6e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB30015	$1{,}07e-01_{2{,}7e-02}$	$1{,}15e-01_{4{,}3e-02}$	$9{,}19e-02_{1{,}7e-02}$	$0{,}00e+00_{0{,}0e+00}$	$8{,}13e-02_{1{,}0e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB30017	$1{,}42e-01_{2{,}2e-02}$	$1{,}16e-01_{4{,}9e-02}$	$1{,}27e-01_{2{,}3e-02}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}05e-01_{3{,}5e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB30018	$1{,}60e-01_{2{,}6e-02}$	$1{,}86e-01_{4{,}3e-02}$	$1{,}55e-01_{2{,}6e-02}$	$1{,}26e-01_{4{,}5e-02}$	$1{,}11e-01_{1{,}2e-02}$	$1{,}26e-01_{4{,}5e-02}$
BB30019	$1{,}40e-01_{9{,}8e-03}$	$1{,}50e-01_{2{,}4e-02}$	$1{,}23e-01_{1{,}2e-02}$	$0{,}00e+00_{0{,}0e+00}$	$9{,}18e-02_{9{,}3e-03}$	$0{,}00e+00_{9{,}4e-02}$
BB30021	$1{,}58e-01_{8{,}8e-03}$	$1{,}57e-01_{8{,}8e-03}$	$1{,}43e-01_{1{,}2e-02}$	$1{,}36e-01_{9{,}3e-02}$	$4{,}75e-02_{2{,}9e-02}$	$1{,}36e-01_{9{,}3e-02}$
BB30022	$5{,}34e-02_{3{,}5e-03}$	$4{,}99e-02_{8{,}7e-03}$	$4{,}87e-02_{4{,}7e-03}$	$0{,}00e+00_{0{,}0e+00}$	$3{,}70e-02_{1{,}2e-03}$	$0{,}00e+00_{3{,}6e-02}$
BB30024	$1{,}71e-01_{8{,}7e-03}$	$1{,}60e-01_{1{,}3e-02}$	$1{,}74e-01_{5{,}8e-03}$	$1{,}62e-01_{1{,}3e-01}$	$4{,}85e-02_{1{,}6e-02}$	$1{,}62e-01_{1{,}3e-01}$
BB30025	$5{,}99e-02_{9{,}2e-03}$	$4{,}37e-02_{8{,}1e-03}$	$4{,}69e-02_{1{,}1e-02}$	$4{,}37e-02_{8{,}1e-03}$	$4{,}10e-02_{3{,}5e-03}$	$4{,}37e-02_{8{,}1e-03}$
BB30026	$2{,}65e-01_{4{,}0e-02}$	$2{,}11e-01_{9{,}5e-02}$	$2{,}34e-01_{3{,}7e-02}$	$2{,}11e-01_{9{,}5e-02}$	$1{,}53e-01_{8{,}3e-03}$	$2{,}11e-01_{9{,}5e-02}$
BB30027	$8{,}50e-02_{1{,}1e-02}$	$8{,}50e-02_{1{,}1e-02}$	$7{,}43e-02_{6{,}4e-03}$	$0{,}00e+00_{0{,}0e+00}$	$7{,}21e-02_{6{,}3e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB30028	$1{,}94e-01_{2{,}3e-02}$	$6{,}61e-02_{2{,}5e-02}$	$1{,}70e-01_{3{,}5e-02}$	$1{,}36e-01_{9{,}6e-02}$	$6{,}61e-02_{2{,}5e-02}$	$1{,}36e-01_{9{,}6e-02}$
BB30029	$1{,}74e-01_{1{,}8e-02}$	$1{,}57e-01_{8{,}1e-03}$	$1{,}57e-01_{8{,}1e-03}$	$1{,}40e-01_{8{,}2e-02}$	$7{,}80e-02_{1{,}9e-02}$	$1{,}40e-01_{8{,}2e-02}$
BB30030	$3{,}17e-01_{1{,}0e-02}$	$2{,}99e-01_{2{,}1e-02}$	$2{,}96e-01_{8{,}3e-03}$	$2{,}99e-01_{2{,}1e-02}$	$2{,}99e-01_{2{,}1e-02}$	$2{,}99e-01_{2{,}1e-02}$

Tabla B.5: Mediana y rango intercuartílico del indicador I_{HV} para los problemas de la familia RV40 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB40001	$2{,}67e-01_{5{,}1e-03}$	$2{,}56e-01_{6{,}9e-03}$	$2{,}56e-01_{4{,}1e-03}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}51e-01_{3{,}6e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40002	$2{,}96e-01_{1{,}3e-02}$	$2{,}91e-01_{1{,}1e-02}$	$2{,}91e-01_{1{,}1e-02}$	$1{,}55e-01_{3{,}9e-03}$	$1{,}55e-01_{3{,}9e-03}$	$1{,}55e-01_{3{,}9e-03}$
BB40003	$5{,}57e-01_{2{,}9e-02}$	$4{,}92e-01_{3{,}6e-02}$	$4{,}95e-01_{3{,}4e-02}$	$4{,}01e-01_{7{,}3e-02}$	$4{,}79e-01_{2{,}2e-02}$	$4{,}71e-01_{3{,}6e-02}$
BB40004	$6{,}32e-02_{9{,}9e-03}$	$5{,}59e-02_{1{,}6e-02}$	$5{,}55e-02_{1{,}1e-02}$	$4{,}43e-02_{2{,}9e-02}$	$4{,}67e-02_{6{,}4e-03}$	$4{,}43e-02_{2{,}9e-02}$
BB40005	$1{,}56e-01_{1{,}2e-02}$	$1{,}58e-01_{2{,}1e-02}$	$1{,}47e-01_{1{,}3e-02}$	$1{,}05e-01_{2{,}1e-02}$	$1{,}38e-01_{1{,}2e-02}$	$1{,}18e-01_{2{,}6e-02}$
BB40006	$1{,}43e-01_{1{,}1e-02}$	$1{,}21e-01_{7{,}1e-03}$	$1{,}32e-01_{1{,}1e-02}$	$7{,}08e-02_{5{,}2e-02}$	$1{,}24e-01_{2{,}0e-02}$	$8{,}87e-02_{2{,}4e-02}$
BB40007	$4{,}18e-01_{2{,}6e-02}$	$4{,}28e-01_{2{,}1e-02}$	$3{,}76e-01_{3{,}0e-02}$	$1{,}56e-01_{1{,}0e-02}$	$3{,}50e-01_{2{,}5e-02}$	$1{,}40e-01_{2{,}0e-02}$
BB40008	$1{,}72e-01_{1{,}9e-02}$	$1{,}72e-01_{3{,}7e-02}$	$1{,}56e-01_{1{,}3e-02}$	$5{,}81e-02_{8{,}2e-02}$	$1{,}42e-01_{1{,}6e-02}$	$6{,}06e-02_{9{,}3e-02}$
BB40009	$8{,}49e-02_{5{,}3e-03}$	$9{,}50e-02_{2{,}0e-02}$	$8{,}04e-02_{1{,}1e-02}$	$0{,}00e+00_{0{,}0e+00}$	$6{,}94e-02_{5{,}1e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40010	$3{,}97e-01_{3{,}6e-02}$	$4{,}20e-01_{4{,}4e-02}$	$3{,}70e-01_{3{,}4e-02}$	$1{,}85e-01_{7{,}3e-02}$	$3{,}83e-01_{2{,}3e-02}$	$3{,}00e-01_{1{,}3e-02}$
BB40011	$1{,}58e-01_{6{,}1e-03}$	$1{,}45e-01_{7{,}9e-03}$	$1{,}48e-01_{6{,}3e-03}$	$0{,}00e+00_{0{,}0e+00}$	$9{,}80e-02_{4{,}1e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40012	$9{,}57e-02_{1{,}6e-02}$	$8{,}53e-02_{2{,}1e-02}$	$9{,}11e-02_{1{,}3e-02}$	$0{,}00e+00_{0{,}0e+00}$	$7{,}59e-02_{9{,}9e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40013	$1{,}63e-01_{1{,}0e-02}$	$1{,}51e-01_{1{,}8e-02}$	$1{,}64e-01_{2{,}0e-02}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}27e-01_{7{,}2e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40014	$3{,}50e-01_{2{,}4e-02}$	$3{,}38e-01_{2{,}1e-02}$	$3{,}26e-01_{1{,}7e-02}$	$1{,}26e-01_{3{,}9e-02}$	$3{,}34e-01_{2{,}1e-02}$	$1{,}29e-01_{4{,}8e-02}$
BB40015	$4{,}37e-01_{8{,}2e-03}$	$4{,}20e-01_{6{,}4e-03}$	$4{,}15e-01_{8{,}2e-03}$	$0{,}00e+00_{0{,}0e+00}$	$2{,}62e-01_{6{,}2e-03}$	$0{,}00e+00_{2{,}6e-01}$
BB40016	$0{,}00e+00_{7{,}7e-02}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{7{,}3e-02}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$
BB40017	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{7{,}8e-02}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$
BB40018	$2{,}61e-01_{2{,}1e-02}$	$2{,}20e-01_{1{,}3e-02}$	$2{,}24e-01_{1{,}6e-02}$	$0{,}00e+00_{0{,}0e+00}$	$2{,}15e-01_{6{,}1e-03}$	$2{,}96e-01_{8{,}2e-02}$
BB40019	$6{,}11e-01_{5{,}2e-02}$	$5{,}86e-01_{4{,}0e-02}$	$5{,}22e-01_{1{,}8e-02}$	$2{,}54e-01_{2{,}1e-02}$	$5{,}48e-01_{2{,}9e-02}$	$5{,}36e-01_{2{,}4e-02}$
BB40020	$8{,}93e-02_{9{,}4e-03}$	$9{,}03e-02_{2{,}9e-02}$	$7{,}73e-02_{1{,}0e-02}$	$0{,}00e+00_{0{,}0e+00}$	$8{,}04e-02_{7{,}0e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40021	$2{,}01e-01_{1{,}2e-02}$	$1{,}81e-01_{1{,}3e-02}$	$1{,}92e-01_{1{,}1e-02}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}42e-01_{1{,}0e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB40022	$3{,}70e-01_{1{,}7e-02}$	$3{,}69e-01_{2{,}7e-02}$	$3{,}54e-01_{2{,}2e-02}$	$0{,}00e+00_{0{,}0e+00}$	$2{,}73e-01_{2{,}0e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB40023	$2{,}50e-01_{9{,}9e-03}$	$2{,}61e-01_{4{,}9e-02}$	$2{,}34e-01_{1{,}8e-02}$	$2{,}09e-01_{2{,}2e-01}$	$2{,}18e-01_{1{,}8e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB40024	$1{,}30e-01_{9{,}9e-03}$	$1{,}18e-01_{1{,}1e-02}$	$1{,}20e-01_{9{,}1e-03}$	$0{,}00e+00_{0{,}0e+00}$	$7{,}64e-02_{4{,}5e-03}$	$0{,}00e+00_{7{,}5e-02}$
BB40025	$4{,}23e-01_{3{,}0e-02}$	$4{,}38e-01_{6{,}1e-02}$	$3{,}53e-01_{4{,}3e-02}$	$3{,}00e-01_{2{,}0e-02}$	$3{,}34e-01_{3{,}1e-02}$	$3{,}30e-01_{1{,}0e-02}$
BB40026	$1{,}49e-01_{8{,}2e-03}$	$1{,}40e-01_{4{,}8e-03}$	$1{,}43e-01_{5{,}1e-03}$	$0{,}00e+00_{0{,}0e+00}$	$8{,}96e-02_{3{,}8e-03}$	$0{,}00e+00_{4{,}4e-02}$
BB40027	$1{,}37e-01_{2{,}5e-02}$	$1{,}07e-01_{2{,}3e-02}$	$1{,}43e-01_{3{,}5e-02}$	$0{,}00e+00_{0{,}0e+00}$	$9{,}84e-02_{9{,}2e-03}$	$0{,}00e+00_{9{,}1e-02}$
BB40028	$9{,}29e-02_{7{,}2e-03}$	$1{,}05e-01_{2{,}1e-02}$	$8{,}14e-02_{7{,}1e-03}$	$0{,}00e+00_{0{,}0e+00}$	$6{,}47e-02_{4{,}6e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40029	$2{,}35e-01_{2{,}5e-02}$	$2{,}17e-01_{4{,}6e-02}$	$1{,}76e-01_{4{,}5e-02}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}80e-01_{3{,}1e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB40030	$1{,}02e-01_{4{,}4e-03}$	$9{,}29e-02_{7{,}0e-03}$	$9{,}39e-02_{5{,}3e-03}$	$0{,}00e+00_{0{,}0e+00}$	$7{,}22e-02_{4{,}4e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40031	$1{,}88e-01_{5{,}4e-03}$	$1{,}79e-01_{3{,}6e-03}$	$1{,}78e-01_{5{,}3e-03}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}02e-01_{3{,}6e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40032	$5{,}59e-01_{2{,}9e-02}$	$5{,}40e-01_{1{,}3e-02}$	$5{,}51e-01_{8{,}2e-03}$	$2{,}20e-01_{3{,}8e-02}$	$5{,}58e-01_{8{,}5e-03}$	$2{,}65e-01_{4{,}3e-02}$
BB40033	$1{,}17e-01_{5{,}4e-03}$	$1{,}08e-01_{9{,}9e-03}$	$1{,}14e-01_{9{,}4e-03}$	$0{,}00e+00_{0{,}0e+00}$	$9{,}18e-02_{7{,}2e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40034	$1{,}37e-01_{1{,}0e-02}$	$1{,}58e-01_{1{,}1e-02}$	$1{,}20e-01_{7{,}8e-03}$	$0{,}00e+00_{0{,}0e+00}$	$9{,}06e-02_{6{,}8e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40035	$2{,}16e-01_{3{,}9e-03}$	$2{,}10e-01_{8{,}7e-03}$	$2{,}08e-01_{4{,}2e-03}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}28e-01_{6{,}0e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40036	$1{,}14e-01_{1{,}1e-02}$	$9{,}86e-02_{7{,}8e-03}$	$9{,}90e-02_{1{,}5e-02}$	$0{,}00e+00_{0{,}0e+00}$	$9{,}09e-02_{7{,}9e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40037	$1{,}28e-01_{1{,}6e-02}$	$1{,}25e-01_{2{,}8e-02}$	$1{,}39e-01_{1{,}6e-02}$	$0{,}00e+00_{0{,}0e+00}$	$9{,}02e-02_{8{,}3e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40038	$4{,}03e-01_{4{,}0e-02}$	$3{,}58e-01_{2{,}2e-02}$	$3{,}35e-01_{7{,}8e-03}$	$0{,}00e+00_{0{,}0e+00}$	$3{,}28e-01_{2{,}1e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB40039	$8{,}89e-02_{5{,}0e-03}$	$7{,}73e-02_{8{,}0e-03}$	$8{,}20e-02_{5{,}2e-03}$	$0{,}00e+00_{0{,}0e+00}$	$7{,}06e-02_{4{,}3e-03}$	$0{,}00e+00_{7{,}0e-02}$
BB40040	$2{,}09e-01_{1{,}6e-02}$	$2{,}10e-01_{1{,}1e-02}$	$1{,}94e-01_{1{,}3e-02}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}75e-01_{1{,}6e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB40041	$2{,}58e-01_{1{,}2e-02}$	$2{,}52e-01_{1{,}0e-02}$	$2{,}39e-01_{8{,}7e-02}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}54e-01_{6{,}0e-03}$	$0{,}00e+00_{1{,}5e-01}$
BB40042	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$
BB40043	$1{,}99e-01_{2{,}1e-02}$	$1{,}96e-01_{2{,}4e-02}$	$1{,}79e-01_{1{,}3e-02}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}57e-01_{6{,}1e-03}$	$0{,}00e+00_{2{,}1e-02}$
BB40044	$1{,}31e-01_{5{,}5e-03}$	$1{,}24e-01_{5{,}4e-03}$	$1{,}22e-01_{4{,}8e-03}$	$0{,}00e+00_{0{,}0e+00}$	$7{,}03e-02_{3{,}6e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB40045	$1{,}55e-01_{6{,}2e-02}$	$1{,}27e-01_{3{,}0e-02}$	$1{,}35e-01_{2{,}5e-02}$	$1{,}30e-01_{6{,}2e-02}$	$1{,}28e-01_{2{,}7e-02}$	$1{,}63e-01_{2{,}9e-02}$
BB40046	$1{,}79e-01_{2{,}8e-02}$	$1{,}68e-01_{1{,}6e-02}$	$1{,}61e-01_{2{,}1e-02}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}05e-01_{1{,}2e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB40047	$3{,}57e-01_{9{,}6e-03}$	$3{,}21e-01_{2{,}2e-02}$	$3{,}21e-01_{2{,}2e-02}$	$3{,}21e-01_{2{,}2e-02}$	$3{,}12e-01_{7{,}6e-02}$	$3{,}21e-01_{2{,}2e-02}$
BB40048	$1{,}93e-01_{2{,}0e-02}$	$2{,}26e-01_{7{,}5e-02}$	$1{,}77e-01_{1{,}8e-02}$	$1{,}77e-01_{1{,}8e-02}$	$1{,}75e-01_{2{,}0e-02}$	$1{,}55e-01_{1{,}7e-02}$
BB40049	$3{,}66e-01_{0{,}0e+00}$	$3{,}66e-01_{0{,}0e+00}$	$3{,}66e-01_{0{,}0e+00}$	$3{,}66e-01_{0{,}0e+00}$	$3{,}66e-01_{0{,}0e+00}$	$3{,}66e-01_{0{,}0e+00}$

Tabla B.6: Mediana y rango intercuartílico del indicador I_{HV} para los problemas de la familia RV50 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB50001	$6{,}16e-02_{1{,}6e-02}$	$7{,}29e-02_{3{,}0e-02}$	$6{,}55e-02_{1{,}6e-02}$	$0{,}00e+00_{0{,}0e+00}$	$4{,}65e-02_{8{,}7e-03}$	$0{,}00e+00_{4{,}3e-02}$
BB50002	$2{,}74e-01_{1{,}4e-02}$	$2{,}56e-01_{2{,}7e-02}$	$2{,}51e-01_{1{,}2e-02}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}94e-01_{1{,}3e-02}$	$0{,}00e+00_{0{,}0e+00}$
BB50003	$1{,}35e-01_{1{,}7e-02}$	$1{,}77e-01_{1{,}4e-02}$	$1{,}28e-01_{2{,}6e-02}$	$7{,}73e-02_{8{,}3e-02}$	$8{,}33e-02_{6{,}7e-03}$	$7{,}73e-02_{8{,}3e-02}$
BB50004	$4{,}41e-01_{5{,}3e-02}$	$4{,}20e-01_{5{,}0e-02}$	$4{,}39e-01_{6{,}1e-02}$	$2{,}58e-01_{1{,}2e-01}$	$4{,}39e-01_{3{,}0e-02}$	$4{,}39e-01_{5{,}4e-02}$
BB50005	$5{,}01e-01_{5{,}5e-02}$	$4{,}52e-01_{3{,}1e-02}$	$4{,}48e-01_{2{,}7e-02}$	$1{,}30e-01_{1{,}0e-01}$	$4{,}72e-01_{6{,}9e-02}$	$2{,}82e-01_{6{,}5e-02}$
BB50006	$2{,}30e-01_{7{,}0e-03}$	$2{,}13e-01_{5{,}0e-03}$	$2{,}20e-01_{9{,}6e-03}$	$2{,}13e-01_{5{,}0e-03}$	$2{,}09e-01_{1{,}3e-01}$	$2{,}13e-01_{5{,}0e-03}$
BB50007	$1{,}65e-01_{6{,}9e-03}$	$1{,}54e-01_{9{,}0e-03}$	$1{,}51e-01_{5{,}3e-03}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}10e-01_{5{,}9e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB50008	$9{,}61e-02_{1{,}5e-02}$	$9{,}13e-02_{2{,}0e-02}$	$8{,}74e-02_{1{,}4e-02}$	$0{,}00e+00_{2{,}9e-02}$	$7{,}73e-02_{1{,}2e-02}$	$0{,}00e+00_{5{,}1e-02}$
BB50009	$8{,}75e-02_{1{,}6e-02}$	$7{,}77e-02_{2{,}0e-02}$	$7{,}65e-02_{1{,}3e-02}$	$0{,}00e+00_{0{,}0e+00}$	$5{,}63e-02_{4{,}1e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB50010	$1{,}71e-01_{1{,}1e-02}$	$1{,}53e-01_{9{,}0e-03}$	$1{,}53e-01_{1{,}1e-02}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}27e-01_{4{,}5e-03}$	$0{,}00e+00_{0{,}0e+00}$
BB50011	$1{,}63e-01_{1{,}2e-02}$	$1{,}61e-01_{2{,}3e-02}$	$1{,}63e-01_{1{,}4e-02}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}10e-01_{5{,}7e-03}$	$0{,}00e+00_{1{,}1e-01}$
BB50012	$3{,}50e-01_{8{,}5e-03}$	$3{,}35e-01_{8{,}6e-03}$	$1{,}77e-01_{3{,}6e-03}$	$1{,}77e-01_{3{,}6e-03}$	$1{,}77e-01_{3{,}6e-03}$	$1{,}77e-01_{3{,}6e-03}$
BB50013	$3{,}62e-01_{1{,}6e-02}$	$3{,}51e-01_{4{,}1e-02}$	$3{,}55e-01_{2{,}8e-02}$	$2{,}38e-01_{3{,}7e-02}$	$3{,}38e-01_{2{,}8e-02}$	$2{,}58e-01_{2{,}8e-02}$
BB50014	$1{,}18e-01_{4{,}5e-03}$	$1{,}09e-01_{1{,}2e-02}$	$1{,}08e-01_{5{,}7e-03}$	$5{,}65e-02_{4{,}2e-02}$	$9{,}10e-02_{4{,}6e-03}$	$5{,}38e-03_{1{,}6e-02}$
BB50015	$1{,}74e-01_{4{,}1e-02}$	$1{,}58e-01_{1{,}3e-02}$	$1{,}68e-01_{1{,}4e-02}$	$0{,}00e+00_{0{,}0e+00}$	$1{,}11e-01_{6{,}6e-03}$	$0{,}00e+00_{1{,}1e-01}$
BB50016	$3{,}22e-01_{2{,}7e-03}$	$3{,}07e-01_{4{,}9e-03}$	$3{,}05e-01_{6{,}2e-03}$	$0{,}00e+00_{0{,}0e+00}$	$2{,}26e-01_{7{,}4e-03}$	$0{,}00e+00_{0{,}0e+00}$

Tabla B.7: Mediana y rango intercuartílico del indicador $I_{\epsilon+}$ para los problemas de la familia RV11 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB11001	$4{,}21e-01_{1{,}1e-01}$	$3{,}83e-01_{1{,}4e-01}$	$4{,}45e-01_{1{,}4e-01}$	$6{,}75e-01_{2{,}7e-01}$	$4{,}86e-01_{1{,}5e-01}$	$4{,}26e-01_{1{,}7e-01}$
BB11002	$5{,}03e-01_{5{,}3e-02}$	$4{,}83e-01_{4{,}8e-02}$	$5{,}49e-01_{5{,}1e-02}$	$6{,}61e-01_{1{,}7e-01}$	$5{,}54e-01_{4{,}7e-02}$	$3{,}24e-01_{6{,}3e-02}$
BB11003	$4{,}64e-01_{1{,}7e-01}$	$4{,}15e-01_{1{,}3e-01}$	$4{,}73e-01_{1{,}2e-01}$	$8{,}27e-01_{9{,}4e-02}$	$4{,}26e-01_{9{,}0e-02}$	$4{,}56e-01_{2{,}4e-01}$
BB11004	$3{,}87e-01_{1{,}4e-01}$	$4{,}08e-01_{1{,}4e-01}$	$4{,}05e-01_{9{,}0e-02}$	$1{,}10e+00_{1{,}3e-01}$	$4{,}50e-01_{1{,}2e-01}$	$3{,}80e-01_{2{,}0e-01}$
BB11005	$4{,}68e-01_{3{,}1e-02}$	$5{,}23e-01_{4{,}8e-02}$	$4{,}98e-01_{3{,}8e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}72e-01_{3{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB11006	$4{,}05e-01_{4{,}6e-02}$	$3{,}36e-01_{6{,}7e-02}$	$4{,}31e-01_{6{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$	$4{,}47e-01_{4{,}3e-02}$	$4{,}24e-01_{3{,}8e-02}$
BB11007	$6{,}13e-01_{3{,}5e-02}$	$6{,}17e-01_{5{,}9e-02}$	$6{,}65e-01_{2{,}8e-02}$	$7{,}64e-01_{1{,}1e-01}$	$6{,}71e-01_{3{,}0e-02}$	$4{,}30e-01_{1{,}1e-01}$
BB11008	$2{,}09e-01_{1{,}0e-02}$	$2{,}09e-01_{6{,}3e-03}$	$1{,}49e-01_{8{,}2e-02}$	$4{,}87e-01_{1{,}7e-01}$	$1{,}49e-01_{9{,}4e-02}$	$3{,}55e-01_{4{,}5e-02}$
BB11009	$3{,}58e-01_{2{,}2e-01}$	$4{,}30e-01_{2{,}1e-01}$	$5{,}04e-01_{1{,}4e-01}$	$9{,}90e-01_{1{,}9e-01}$	$4{,}68e-01_{1{,}3e-01}$	$4{,}72e-01_{1{,}0e-01}$
BB11010	$4{,}87e-01_{1{,}2e-01}$	$7{,}04e-01_{2{,}1e-01}$	$4{,}76e-01_{1{,}4e-01}$	$6{,}88e-01_{3{,}2e-01}$	$5{,}99e-01_{1{,}3e-01}$	$3{,}01e-01_{1{,}3e-01}$
BB11011	$3{,}51e-01_{1{,}0e-01}$	$3{,}14e-01_{1{,}4e-01}$	$3{,}44e-01_{1{,}2e-01}$	$8{,}56e-01_{8{,}2e-02}$	$3{,}90e-01_{1{,}1e-01}$	$4{,}88e-01_{5{,}6e-02}$
BB11012	$5{,}52e-01_{2{,}1e-01}$	$6{,}10e-01_{1{,}6e-01}$	$7{,}21e-01_{2{,}3e-01}$	$9{,}16e-01_{1{,}0e-01}$	$7{,}92e-01_{2{,}7e-01}$	$6{,}10e-01_{9{,}2e-02}$
BB11013	$4{,}19e-01_{2{,}7e-01}$	$3{,}57e-01_{3{,}2e-01}$	$5{,}16e-01_{2{,}4e-01}$	$1{,}36e+00_{2{,}5e-01}$	$5{,}99e-01_{2{,}2e-01}$	$6{,}46e-01_{4{,}0e-02}$
BB11014	$3{,}76e-01_{7{,}3e-02}$	$4{,}43e-01_{6{,}8e-02}$	$4{,}69e-01_{4{,}5e-02}$	$8{,}70e-01_{1{,}8e-01}$	$4{,}66e-01_{7{,}2e-02}$	$5{,}78e-01_{1{,}3e-01}$
BB11015	$3{,}99e-01_{1{,}6e-01}$	$4{,}03e-01_{1{,}5e-01}$	$4{,}61e-01_{1{,}4e-01}$	$1{,}34e+00_{2{,}7e-01}$	$3{,}96e-01_{1{,}6e-01}$	$4{,}03e-01_{2{,}9e-01}$
BB11016	$3{,}26e-01_{5{,}9e-02}$	$3{,}38e-01_{1{,}0e-01}$	$4{,}11e-01_{7{,}8e-02}$	$9{,}51e-01_{6{,}4e-03}$	$4{,}26e-01_{4{,}2e-02}$	$5{,}74e-01_{5{,}0e-02}$
BB11017	$4{,}48e-01_{8{,}8e-02}$	$4{,}99e-01_{1{,}1e-01}$	$3{,}54e-01_{8{,}3e-02}$	$6{,}43e-01_{1{,}7e-01}$	$4{,}26e-01_{1{,}1e-01}$	$2{,}76e-01_{1{,}1e-01}$
BB11018	$1{,}74e-01_{4{,}6e-02}$	$2{,}08e-01_{2{,}9e-02}$	$2{,}61e-01_{3{,}1e-02}$	$9{,}65e-01_{1{,}6e-01}$	$7{,}10e-01_{3{,}8e-02}$	$9{,}46e-01_{9{,}8e-02}$
BB11019	$3{,}49e-01_{4{,}3e-02}$	$3{,}48e-01_{6{,}9e-02}$	$4{,}26e-01_{4{,}2e-02}$	$8{,}45e-01_{9{,}1e-02}$	$4{,}27e-01_{2{,}8e-02}$	$4{,}55e-01_{5{,}0e-02}$
BB11020	$4{,}14e-01_{4{,}5e-02}$	$4{,}27e-01_{5{,}9e-02}$	$4{,}85e-01_{3{,}6e-02}$	$7{,}91e-01_{1{,}2e-01}$	$4{,}73e-01_{4{,}5e-02}$	$3{,}50e-01_{6{,}1e-02}$
BB11021	$3{,}34e-01_{8{,}8e-02}$	$4{,}07e-01_{9{,}4e-02}$	$3{,}16e-01_{5{,}6e-02}$	$5{,}14e-01_{1{,}4e-01}$	$3{,}32e-01_{7{,}7e-02}$	$2{,}66e-01_{3{,}5e-02}$
BB11022	$3{,}18e-01_{9{,}5e-02}$	$4{,}23e-01_{2{,}1e-02}$	$4{,}12e-01_{9{,}6e-02}$	$1{,}57e+00_{2{,}5e-01}$	$4{,}11e-01_{7{,}8e-02}$	$5{,}25e-01_{2{,}1e-02}$
BB11023	$3{,}85e-01_{6{,}4e-02}$	$2{,}94e-01_{6{,}4e-02}$	$4{,}63e-01_{6{,}6e-02}$	$8{,}69e-01_{6{,}9e-02}$	$4{,}62e-01_{3{,}7e-02}$	$4{,}43e-01_{2{,}7e-02}$
BB11024	$3{,}36e-01_{1{,}7e-01}$	$3{,}83e-01_{1{,}9e-01}$	$4{,}58e-01_{2{,}0e-01}$	$1{,}12e+00_{1{,}8e-01}$	$5{,}38e-01_{2{,}2e-01}$	$5{,}38e-01_{1{,}1e-01}$
BB11025	$2{,}09e-01_{1{,}2e-01}$	$2{,}87e-01_{4{,}2e-02}$	$3{,}18e-01_{1{,}1e-01}$	$1{,}25e+00_{5{,}6e-01}$	$3{,}18e-01_{6{,}3e-02}$	$3{,}33e-01_{1{,}1e-01}$
BB11027	$4{,}36e-01_{6{,}4e-02}$	$3{,}64e-01_{1{,}1e-01}$	$4{,}88e-01_{7{,}0e-02}$	$8{,}75e-01_{2{,}3e-02}$	$4{,}91e-01_{6{,}9e-02}$	$3{,}82e-01_{2{,}5e-02}$
BB11028	$4{,}32e-01_{1{,}3e-03}$	$4{,}30e-01_{3{,}8e-03}$	$4{,}53e-01_{6{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$	$5{,}04e-01_{5{,}6e-02}$	$8{,}95e-01_{1{,}1e-01}$
BB11029	$3{,}55e-01_{6{,}2e-02}$	$3{,}59e-01_{8{,}7e-02}$	$3{,}36e-01_{5{,}7e-02}$	$5{,}15e-01_{1{,}9e-01}$	$3{,}79e-01_{5{,}0e-02}$	$2{,}77e-01_{2{,}7e-02}$
BB11031	$3{,}20e-01_{2{,}8e-02}$	$3{,}65e-01_{3{,}5e-02}$	$3{,}82e-01_{4{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$	$4{,}36e-01_{5{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB11032	$3{,}59e-01_{4{,}4e-02}$	$3{,}80e-01_{5{,}1e-02}$	$3{,}90e-01_{5{,}0e-02}$	$6{,}44e-01_{1{,}0e-01}$	$4{,}00e-01_{3{,}9e-02}$	$3{,}08e-01_{1{,}3e-02}$
BB11033	$2{,}95e-01_{5{,}3e-02}$	$3{,}80e-01_{5{,}5e-02}$	$3{,}98e-01_{3{,}6e-02}$	$1{,}00e+00_{0{,}0e+00}$	$4{,}00e-01_{3{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB11034	$4{,}64e-01_{3{,}1e-02}$	$4{,}60e-01_{5{,}6e-02}$	$5{,}21e-01_{5{,}9e-02}$	$9{,}78e-01_{1{,}2e-01}$	$5{,}12e-01_{3{,}7e-02}$	$4{,}41e-01_{4{,}9e-02}$
BB11035	$2{,}55e-01_{1{,}4e-01}$	$2{,}25e-01_{8{,}9e-02}$	$3{,}07e-01_{1{,}0e-01}$	$5{,}88e-01_{1{,}3e-01}$	$2{,}88e-01_{1{,}3e-01}$	$3{,}33e-01_{5{,}1e-02}$
BB11036	$4{,}52e-01_{2{,}3e-02}$	$4{,}39e-01_{7{,}9e-02}$	$5{,}16e-01_{3{,}3e-02}$	$8{,}32e-01_{1{,}0e-01}$	$5{,}09e-01_{2{,}6e-02}$	$4{,}06e-01_{5{,}5e-02}$
BB11037	$3{,}39e-01_{9{,}7e-02}$	$3{,}51e-01_{5{,}9e-02}$	$4{,}02e-01_{7{,}2e-02}$	$6{,}44e-01_{1{,}5e-01}$	$3{,}90e-01_{7{,}4e-02}$	$3{,}73e-01_{6{,}7e-02}$
BB11038	$3{,}58e-01_{4{,}4e-02}$	$3{,}88e-01_{3{,}9e-02}$	$4{,}34e-01_{3{,}5e-02}$	$4{,}71e-01_{6{,}4e-02}$	$4{,}12e-01_{3{,}7e-02}$	$3{,}27e-01_{6{,}1e-02}$

Tabla B.8: Mediana y rango intercuartílico del indicador $I_{\epsilon+}$ para los problemas de la familia RV12 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB12001	$2{,}57e-01_{3{,}2e-02}$	$3{,}37e-01_{5{,}2e-02}$	$3{,}53e-01_{4{,}5e-02}$	$5{,}92e-01_{4{,}1e-02}$	$3{,}53e-01_{3{,}7e-02}$	$3{,}98e-01_{2{,}6e-02}$
BB12002	$7{,}20e-01_{2{,}6e-01}$	$7{,}34e-01_{2{,}5e-01}$	$8{,}12e-01_{1{,}8e-01}$	$1{,}08e+00_{1{,}3e-01}$	$7{,}98e-01_{2{,}3e-01}$	$5{,}81e-01_{2{,}5e-01}$
BB12003	$6{,}51e-01_{7{,}8e-02}$	$6{,}24e-01_{7{,}0e-02}$	$6{,}46e-01_{6{,}2e-02}$	$4{,}62e-01_{8{,}4e-02}$	$6{,}62e-01_{8{,}8e-02}$	$3{,}93e-01_{7{,}1e-02}$
BB12004	$4{,}23e-01_{1{,}2e-01}$	$4{,}47e-01_{1{,}1e-01}$	$4{,}64e-01_{6{,}5e-02}$	$4{,}15e-01_{5{,}4e-02}$	$4{,}79e-01_{5{,}9e-02}$	$3{,}30e-01_{5{,}5e-02}$
BB12005	$2{,}99e-01_{9{,}1e-02}$	$3{,}38e-01_{8{,}2e-02}$	$3{,}48e-01_{1{,}2e-01}$	$5{,}74e-01_{6{,}6e-02}$	$3{,}81e-01_{7{,}0e-02}$	$5{,}01e-01_{8{,}8e-02}$
BB12006	$1{,}03e+00_{6{,}4e-01}$	$1{,}35e+00_{3{,}8e-01}$	$7{,}42e-01_{3{,}1e-01}$	$1{,}51e+00_{1{,}1e-01}$	$8{,}28e-01_{6{,}9e-01}$	$7{,}42e-01_{3{,}4e-01}$
BB12007	$1{,}29e-01_{3{,}9e-02}$	$1{,}63e-01_{4{,}2e-02}$	$1{,}98e-01_{4{,}8e-02}$	$4{,}03e-01_{2{,}9e-02}$	$1{,}78e-01_{2{,}9e-02}$	$4{,}15e-01_{8{,}1e-02}$
BB12008	$8{,}47e-01_{1{,}6e-02}$	$8{,}60e-01_{2{,}6e-02}$	$8{,}77e-01_{1{,}2e-02}$	$8{,}63e-01_{9{,}2e-02}$	$8{,}72e-01_{8{,}6e-03}$	$8{,}16e-01_{4{,}2e-02}$
BB12009	$3{,}23e-01_{8{,}8e-02}$	$3{,}45e-01_{1{,}1e-01}$	$4{,}02e-01_{6{,}0e-02}$	$5{,}41e-01_{1{,}1e-01}$	$4{,}00e-01_{6{,}6e-02}$	$2{,}87e-01_{2{,}9e-02}$
BB12010	$2{,}84e-01_{1{,}1e-01}$	$3{,}30e-01_{8{,}5e-02}$	$3{,}55e-01_{7{,}8e-02}$	$4{,}72e-01_{6{,}6e-02}$	$3{,}82e-01_{7{,}1e-02}$	$4{,}06e-01_{7{,}9e-02}$
BB12011	$5{,}58e-01_{1{,}8e-02}$	$5{,}74e-01_{8{,}1e-02}$	$6{,}02e-01_{1{,}7e-02}$	$6{,}05e-01_{4{,}0e-02}$	$5{,}90e-01_{1{,}9e-02}$	$4{,}77e-01_{1{,}5e-01}$
BB12012	$2{,}73e-01_{1{,}1e-01}$	$3{,}81e-01_{7{,}8e-02}$	$2{,}99e-01_{4{,}7e-02}$	$5{,}39e-01_{1{,}2e-01}$	$3{,}72e-01_{7{,}9e-02}$	$2{,}01e-01_{4{,}4e-02}$
BB12013	$1{,}92e-01_{8{,}4e-02}$	$2{,}32e-01_{5{,}4e-02}$	$2{,}49e-01_{6{,}0e-02}$	$4{,}25e-01_{6{,}5e-02}$	$2{,}21e-01_{1{,}0e-01}$	$3{,}65e-01_{8{,}4e-02}$
BB12014	$2{,}67e-01_{5{,}9e-02}$	$2{,}55e-01_{9{,}2e-02}$	$3{,}48e-01_{5{,}0e-02}$	$4{,}90e-01_{7{,}2e-02}$	$3{,}24e-01_{3{,}4e-02}$	$3{,}68e-01_{5{,}4e-02}$
BB12015	$7{,}11e-01_{6{,}2e-02}$	$6{,}83e-01_{7{,}4e-02}$	$7{,}42e-01_{3{,}8e-02}$	$8{,}95e-01_{3{,}1e-01}$	$7{,}60e-01_{2{,}2e-02}$	$6{,}39e-01_{7{,}1e-02}$
BB12016	$3{,}59e-01_{7{,}4e-02}$	$2{,}23e-01_{8{,}1e-02}$	$4{,}20e-01_{1{,}3e-01}$	$5{,}22e-01_{1{,}2e-02}$	$3{,}76e-01_{1{,}1e-01}$	$4{,}95e-01_{1{,}6e-02}$
BB12017	$3{,}73e-01_{1{,}3e-01}$	$3{,}93e-01_{1{,}6e-01}$	$3{,}90e-01_{8{,}2e-02}$	$4{,}33e-01_{7{,}2e-02}$	$4{,}41e-01_{9{,}7e-02}$	$3{,}45e-01_{8{,}1e-02}$
BB12018	$3{,}78e-01_{1{,}3e-01}$	$4{,}16e-01_{5{,}1e-02}$	$3{,}65e-01_{9{,}3e-02}$	$3{,}70e-01_{1{,}0e-01}$	$3{,}93e-01_{7{,}4e-02}$	$3{,}28e-01_{1{,}4e-01}$
BB12019	$1{,}42e-01_{2{,}3e-02}$	$1{,}48e-01_{4{,}8e-02}$	$1{,}46e-01_{3{,}7e-02}$	$5{,}59e-01_{1{,}8e-01}$	$1{,}47e-01_{2{,}8e-02}$	$1{,}89e-01_{6{,}8e-02}$
BB12020	$4{,}16e-01_{1{,}4e-01}$	$4{,}53e-01_{1{,}7e-01}$	$4{,}26e-01_{1{,}1e-01}$	$4{,}98e-01_{1{,}3e-01}$	$5{,}02e-01_{6{,}1e-02}$	$2{,}77e-01_{4{,}7e-02}$
BB12021	$3{,}70e-01_{5{,}9e-02}$	$3{,}72e-01_{1{,}1e-01}$	$4{,}22e-01_{5{,}2e-02}$	$5{,}99e-01_{6{,}3e-02}$	$4{,}17e-01_{4{,}6e-02}$	$2{,}96e-01_{1{,}5e-01}$
BB12022	$3{,}67e-01_{2{,}0e-02}$	$3{,}12e-01_{2{,}1e-01}$	$3{,}83e-01_{1{,}2e-02}$	$3{,}73e-01_{2{,}8e-02}$	$3{,}74e-01_{1{,}8e-02}$	$3{,}46e-01_{3{,}2e-02}$
BB12023	$4{,}35e-01_{1{,}2e-01}$	$4{,}17e-01_{8{,}9e-02}$	$4{,}02e-01_{1{,}1e-01}$	$7{,}90e-01_{3{,}1e-01}$	$4{,}40e-01_{9{,}7e-02}$	$2{,}11e-01_{5{,}1e-02}$
BB12024	$4{,}67e-01_{1{,}4e-01}$	$5{,}21e-01_{1{,}6e-01}$	$4{,}56e-01_{9{,}2e-02}$	$4{,}71e-01_{7{,}3e-02}$	$4{,}75e-01_{9{,}0e-02}$	$3{,}51e-01_{1{,}3e-01}$
BB12025	$2{,}66e-01_{5{,}6e-02}$	$2{,}59e-01_{8{,}0e-02}$	$3{,}04e-01_{6{,}7e-02}$	$5{,}66e-01_{1{,}1e-01}$	$2{,}93e-01_{5{,}9e-02}$	$3{,}26e-01_{1{,}2e-01}$
BB12026	$8{,}22e-01_{3{,}6e-02}$	$8{,}33e-01_{2{,}2e-02}$	$8{,}60e-01_{1{,}0e-02}$	$8{,}05e-01_{7{,}4e-02}$	$8{,}51e-01_{1{,}8e-02}$	$4{,}88e-01_{3{,}5e-01}$
BB12027	$6{,}97e-01_{2{,}9e-02}$	$5{,}53e-01_{1{,}8e-01}$	$7{,}26e-01_{3{,}7e-02}$	$6{,}98e-01_{1{,}6e-01}$	$7{,}18e-01_{3{,}8e-02}$	$5{,}87e-01_{9{,}2e-02}$
BB12028	$1{,}77e-01_{4{,}1e-02}$	$2{,}15e-01_{3{,}5e-02}$	$2{,}34e-01_{2{,}1e-02}$	$5{,}50e-01_{9{,}3e-02}$	$2{,}29e-01_{3{,}5e-02}$	$3{,}54e-01_{2{,}5e-02}$
BB12029	$6{,}78e-01_{2{,}0e-02}$	$7{,}12e-01_{3{,}2e-02}$	$7{,}17e-01_{1{,}6e-02}$	$6{,}91e-01_{5{,}3e-02}$	$7{,}22e-01_{1{,}1e-02}$	$6{,}44e-01_{6{,}7e-02}$
BB12030	$3{,}15e-01_{4{,}7e-02}$	$1{,}85e-01_{1{,}4e-01}$	$3{,}47e-01_{3{,}8e-02}$	$4{,}14e-01_{6{,}2e-02}$	$3{,}31e-01_{4{,}2e-02}$	$2{,}80e-01_{3{,}7e-02}$
BB12031	$1{,}99e-01_{2{,}3e-02}$	$2{,}70e-01_{3{,}7e-02}$	$2{,}96e-01_{3{,}3e-02}$	$6{,}04e-01_{6{,}3e-02}$	$2{,}93e-01_{2{,}7e-02}$	$3{,}34e-01_{2{,}6e-02}$
BB12032	$5{,}23e-01_{7{,}5e-02}$	$4{,}33e-01_{8{,}2e-02}$	$5{,}51e-01_{6{,}3e-02}$	$3{,}87e-01_{8{,}1e-02}$	$5{,}73e-01_{6{,}1e-02}$	$3{,}38e-01_{6{,}4e-02}$
BB12033	$3{,}16e-01_{3{,}0e-02}$	$3{,}52e-01_{3{,}5e-02}$	$3{,}57e-01_{1{,}9e-02}$	$4{,}50e-01_{1{,}3e-01}$	$3{,}62e-01_{2{,}9e-02}$	$3{,}34e-01_{2{,}8e-02}$
BB12034	$7{,}13e-01_{2{,}6e-01}$	$6{,}60e-01_{1{,}9e-01}$	$6{,}84e-01_{1{,}9e-01}$	$1{,}29e+00_{3{,}1e-01}$	$7{,}33e-01_{2{,}3e-01}$	$5{,}91e-01_{1{,}2e-01}$
BB12035	$9{,}34e-01_{4{,}8e-03}$	$9{,}36e-01_{2{,}4e-02}$	$9{,}44e-01_{3{,}2e-03}$	$9{,}50e-01_{7{,}8e-02}$	$9{,}41e-01_{5{,}9e-03}$	$9{,}10e-01_{1{,}0e-01}$
BB12036	$4{,}35e-01_{1{,}3e-01}$	$5{,}50e-01_{1{,}2e-01}$	$4{,}71e-01_{1{,}7e-01}$	$5{,}94e-01_{1{,}2e-01}$	$4{,}87e-01_{1{,}1e-01}$	$3{,}54e-01_{8{,}8e-02}$
BB12037	$8{,}64e-01_{1{,}2e-02}$	$8{,}77e-01_{7{,}9e-03}$	$8{,}76e-01_{4{,}0e-02}$	$8{,}92e-01_{2{,}1e-02}$	$8{,}85e-01_{2{,}6e-02}$	$8{,}69e-01_{1{,}7e-01}$
BB12038	$6{,}55e-01_{9{,}5e-02}$	$6{,}00e-01_{2{,}7e-01}$	$7{,}04e-01_{5{,}1e-02}$	$6{,}25e-01_{1{,}5e-01}$	$7{,}12e-01_{5{,}8e-02}$	$4{,}13e-01_{2{,}2e-01}$
BB12039	$4{,}64e-01_{7{,}1e-02}$	$4{,}28e-01_{1{,}1e-01}$	$5{,}20e-01_{6{,}5e-02}$	$5{,}26e-01_{1{,}6e-01}$	$5{,}24e-01_{5{,}9e-02}$	$3{,}51e-01_{3{,}4e-02}$
BB12040	$3{,}87e-01_{6{,}7e-02}$	$4{,}29e-01_{9{,}4e-02}$	$3{,}81e-01_{1{,}1e-01}$	$4{,}47e-01_{1{,}7e-01}$	$4{,}61e-01_{1{,}1e-01}$	$2{,}42e-01_{7{,}4e-02}$
BB12041	$4{,}85e-01_{9{,}7e-02}$	$5{,}22e-01_{1{,}0e-01}$	$5{,}23e-01_{7{,}6e-02}$	$3{,}77e-01_{5{,}1e-02}$	$5{,}02e-01_{8{,}1e-02}$	$3{,}11e-01_{1{,}0e-01}$
BB12042	$5{,}16e-01_{1{,}8e-01}$	$5{,}61e-01_{1{,}3e-01}$	$6{,}42e-01_{2{,}7e-01}$	$1{,}11e+00_{1{,}9e-01}$	$5{,}16e-01_{1{,}7e-01}$	$4{,}54e-01_{1{,}8e-01}$
BB12043	$7{,}01e-01_{3{,}0e-02}$	$7{,}19e-01_{2{,}2e-02}$	$6{,}87e-01_{3{,}1e-02}$	$9{,}79e-01_{3{,}1e-01}$	$6{,}86e-01_{2{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB12044	$3{,}72e-01_{8{,}4e-02}$	$4{,}07e-01_{8{,}7e-02}$	$4{,}00e-01_{7{,}2e-02}$	$3{,}71e-01_{8{,}1e-02}$	$4{,}23e-01_{5{,}5e-02}$	$3{,}43e-01_{3{,}6e-02}$

Tabla B.9: Mediana y rango intercuartílico del indicador $I_{\epsilon+}$ para los problemas de la familia RV20 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB20001	$6{,}26e-01_{6{,}8e-02}$	$5{,}37e-01_{7{,}3e-02}$	$6{,}97e-01_{3{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}37e-01_{4{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20002	$2{,}64e-01_{4{,}8e-02}$	$3{,}35e-01_{4{,}0e-02}$	$3{,}46e-01_{4{,}4e-02}$	$5{,}57e-01_{7{,}7e-02}$	$5{,}57e-01_{7{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20003	$5{,}67e-01_{1{,}5e-01}$	$6{,}25e-01_{1{,}7e-01}$	$5{,}58e-01_{3{,}5e-01}$	$7{,}43e-01_{3{,}4e-01}$	$7{,}81e-01_{9{,}0e-02}$	$7{,}43e-01_{3{,}4e-01}$
BB20004	$2{,}56e-01_{1{,}0e-01}$	$4{,}14e-01_{1{,}0e-01}$	$4{,}22e-01_{1{,}5e-01}$	$5{,}94e-01_{1{,}1e-01}$	$5{,}94e-01_{1{,}1e-01}$	$4{,}88e-01_{1{,}8e-01}$
BB20005	$7{,}66e-01_{1{,}9e-02}$	$7{,}98e-01_{2{,}4e-02}$	$7{,}96e-01_{1{,}6e-02}$	$8{,}02e-01_{1{,}4e-02}$	$8{,}05e-01_{1{,}5e-02}$	$8{,}02e-01_{1{,}4e-02}$
BB20006	$9{,}61e-01_{5{,}2e-03}$	$9{,}64e-01_{4{,}5e-03}$	$9{,}65e-01_{3{,}8e-03}$	$9{,}64e-01_{2{,}9e-03}$	$9{,}64e-01_{2{,}9e-03}$	$1{,}00e+00_{3{,}5e-02}$
BB20007	$7{,}78e-01_{1{,}5e-02}$	$7{,}88e-01_{3{,}4e-02}$	$8{,}14e-01_{1{,}2e-02}$	$8{,}03e-01_{1{,}7e-02}$	$8{,}03e-01_{1{,}7e-02}$	$9{,}77e-01_{1{,}0e-01}$
BB20008	$1{,}40e-01_{7{,}1e-02}$	$1{,}40e-01_{8{,}1e-02}$	$1{,}40e-01_{8{,}1e-02}$	$1{,}40e-01_{8{,}1e-02}$	$1{,}40e-01_{8{,}1e-02}$	$1{,}40e-01_{8{,}1e-02}$
BB20009	$9{,}09e-01_{2{,}3e-02}$	$9{,}28e-01_{1{,}6e-02}$	$9{,}20e-01_{1{,}1e-02}$	$9{,}16e-01_{1{,}4e-02}$	$9{,}16e-01_{1{,}4e-02}$	$1{,}00e+00_{4{,}1e-02}$
BB20010	$8{,}41e-01_{8{,}5e-03}$	$8{,}30e-01_{3{,}7e-02}$	$8{,}56e-01_{6{,}7e-03}$	$8{,}85e-01_{1{,}5e-01}$	$8{,}42e-01_{8{,}3e-03}$	$8{,}85e-01_{1{,}5e-01}$
BB20011	$7{,}06e-01_{3{,}6e-02}$	$7{,}11e-01_{2{,}2e-02}$	$7{,}01e-01_{4{,}0e-02}$	$7{,}31e-01_{2{,}7e-02}$	$7{,}31e-01_{2{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20012	$6{,}84e-01_{4{,}7e-02}$	$5{,}41e-01_{1{,}1e-01}$	$7{,}40e-01_{4{,}0e-02}$	$5{,}41e-01_{1{,}1e-01}$	$7{,}51e-01_{2{,}3e-02}$	$7{,}65e-01_{2{,}8e-01}$
BB20013	$7{,}49e-01_{6{,}4e-02}$	$6{,}43e-01_{1{,}2e-01}$	$8{,}06e-01_{4{,}1e-02}$	$8{,}05e-01_{1{,}3e-01}$	$7{,}85e-01_{3{,}2e-02}$	$8{,}05e-01_{1{,}3e-01}$
BB20014	$6{,}73e-01_{4{,}5e-02}$	$7{,}27e-01_{2{,}3e-02}$	$6{,}74e-01_{4{,}5e-02}$	$7{,}27e-01_{2{,}3e-02}$	$7{,}12e-01_{2{,}0e-02}$	$7{,}04e-01_{4{,}5e-02}$
BB20015	$6{,}27e-01_{3{,}9e-02}$	$6{,}83e-01_{4{,}8e-02}$	$6{,}56e-01_{5{,}0e-02}$	$6{,}83e-01_{4{,}8e-02}$	$8{,}47e-01_{1{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20016	$6{,}81e-01_{2{,}5e-02}$	$6{,}87e-01_{4{,}1e-02}$	$7{,}16e-01_{2{,}1e-02}$	$6{,}87e-01_{4{,}1e-02}$	$7{,}63e-01_{2{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20017	$8{,}91e-01_{1{,}1e-02}$	$8{,}59e-01_{4{,}0e-02}$	$9{,}05e-01_{2{,}1e-02}$	$9{,}06e-01_{1{,}0e-02}$	$9{,}00e-01_{1{,}3e-02}$	$9{,}06e-01_{1{,}0e-02}$
BB20018	$9{,}68e-01_{3{,}7e-03}$	$9{,}73e-01_{3{,}5e-03}$	$9{,}74e-01_{2{,}0e-03}$	$9{,}73e-01_{3{,}5e-03}$	$9{,}73e-01_{2{,}7e-03}$	$9{,}74e-01_{2{,}8e-02}$
BB20019	$8{,}43e-01_{1{,}3e-02}$	$8{,}39e-01_{4{,}4e-02}$	$8{,}56e-01_{1{,}3e-02}$	$8{,}39e-01_{4{,}4e-02}$	$8{,}43e-01_{2{,}9e-02}$	$8{,}62e-01_{1{,}4e-01}$
BB20020	$8{,}71e-01_{8{,}6e-03}$	$8{,}50e-01_{5{,}0e-02}$	$8{,}83e-01_{1{,}5e-02}$	$8{,}83e-01_{1{,}5e-02}$	$8{,}91e-01_{1{,}0e-02}$	$8{,}61e-01_{2{,}8e-02}$
BB20021	$4{,}44e-01_{6{,}9e-02}$	$4{,}16e-01_{1{,}2e-01}$	$5{,}15e-01_{8{,}8e-02}$	$5{,}15e-01_{8{,}8e-02}$	$5{,}93e-01_{3{,}2e-02}$	$5{,}64e-01_{7{,}4e-02}$
BB20022	$9{,}27e-01_{1{,}1e-02}$	$9{,}32e-01_{6{,}9e-03}$	$9{,}39e-01_{8{,}0e-03}$	$9{,}37e-01_{7{,}4e-03}$	$9{,}33e-01_{7{,}2e-03}$	$9{,}37e-01_{7{,}4e-03}$
BB20023	$8{,}15e-01_{9{,}9e-03}$	$8{,}25e-01_{1{,}4e-02}$	$8{,}36e-01_{6{,}7e-03}$	$8{,}36e-01_{6{,}7e-03}$	$8{,}32e-01_{1{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20024	$4{,}29e-01_{4{,}2e-02}$	$4{,}91e-01_{2{,}8e-02}$	$4{,}55e-01_{7{,}0e-02}$	$7{,}90e-01_{4{,}9e-01}$	$7{,}99e-01_{5{,}0e-02}$	$7{,}90e-01_{4{,}9e-01}$
BB20025	$4{,}76e-01_{9{,}5e-02}$	$4{,}90e-01_{2{,}9e-01}$	$5{,}43e-01_{1{,}4e-01}$	$5{,}43e-01_{1{,}4e-01}$	$7{,}29e-01_{6{,}7e-02}$	$6{,}09e-01_{2{,}0e-01}$
BB20026	$6{,}97e-01_{4{,}0e-02}$	$6{,}75e-01_{8{,}3e-02}$	$7{,}02e-01_{3{,}9e-02}$	$7{,}02e-01_{3{,}9e-02}$	$7{,}80e-01_{3{,}1e-02}$	$7{,}61e-01_{2{,}9e-01}$
BB20027	$6{,}41e-01_{2{,}0e-02}$	$6{,}76e-01_{2{,}2e-02}$	$6{,}74e-01_{1{,}5e-02}$	$6{,}74e-01_{1{,}5e-02}$	$6{,}75e-01_{1{,}1e-02}$	$6{,}81e-01_{1{,}8e-01}$
BB20028	$8{,}63e-01_{2{,}7e-02}$	$8{,}97e-01_{2{,}5e-02}$	$8{,}75e-01_{8{,}8e-02}$	$8{,}75e-01_{8{,}8e-02}$	$8{,}38e-01_{5{,}3e-02}$	$8{,}53e-01_{7{,}6e-02}$
BB20030	$8{,}13e-01_{2{,}9e-02}$	$8{,}54e-01_{2{,}7e-02}$	$8{,}38e-01_{2{,}9e-02}$	$8{,}54e-01_{2{,}7e-02}$	$8{,}47e-01_{2{,}6e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20031	$8{,}84e-01_{2{,}3e-02}$	$8{,}87e-01_{3{,}5e-02}$	$9{,}09e-01_{2{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$	$9{,}19e-01_{1{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20032	$9{,}28e-01_{1{,}3e-02}$	$9{,}35e-01_{4{,}2e-02}$	$9{,}38e-01_{1{,}2e-02}$	$9{,}35e-01_{4{,}2e-02}$	$9{,}39e-01_{1{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20033	$9{,}27e-01_{7{,}5e-03}$	$9{,}39e-01_{1{,}5e-02}$	$9{,}38e-01_{8{,}9e-03}$	$9{,}39e-01_{1{,}5e-02}$	$9{,}31e-01_{5{,}3e-03}$	$1{,}00e+00_{0{,}0e+00}$
BB20034	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$
BB20035	$8{,}55e-01_{1{,}5e-02}$	$8{,}57e-01_{2{,}3e-02}$	$8{,}62e-01_{1{,}1e-02}$	$8{,}57e-01_{2{,}3e-02}$	$8{,}54e-01_{1{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20036	$8{,}13e-01_{3{,}2e-02}$	$8{,}22e-01_{3{,}7e-02}$	$8{,}18e-01_{3{,}6e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}15e-01_{3{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20037	$2{,}80e-01_{2{,}9e-02}$	$3{,}00e-01_{6{,}0e-02}$	$3{,}19e-01_{4{,}3e-02}$	$3{,}00e-01_{6{,}0e-02}$	$6{,}42e-01_{1{,}8e-02}$	$6{,}42e-01_{1{,}8e-02}$
BB20038	$8{,}98e-01_{1{,}5e-02}$	$9{,}13e-01_{1{,}1e-02}$	$9{,}08e-01_{1{,}1e-02}$	$9{,}13e-01_{1{,}1e-02}$	$9{,}09e-01_{1{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20039	$3{,}32e-01_{3{,}8e-02}$	$3{,}24e-01_{3{,}3e-02}$	$3{,}97e-01_{4{,}2e-02}$	$4{,}77e-01_{4{,}5e-01}$	$8{,}53e-01_{4{,}2e-02}$	$4{,}77e-01_{4{,}5e-01}$
BB20040	$2{,}14e-01_{1{,}3e-01}$	$2{,}97e-01_{8{,}8e-02}$	$2{,}69e-01_{9{,}2e-02}$	$2{,}97e-01_{8{,}8e-02}$	$8{,}54e-01_{4{,}5e-02}$	$3{,}39e-01_{5{,}9e-01}$
BB20041	$2{,}28e-01_{6{,}0e-02}$	$2{,}43e-01_{1{,}1e-01}$	$2{,}43e-01_{1{,}0e-01}$	$2{,}43e-01_{1{,}1e-01}$	$3{,}36e-01_{7{,}3e-01}$	$2{,}43e-01_{1{,}0e-01}$

Tabla B.10: Mediana y rango intercuartílico del indicador $I_{\epsilon+}$ para los problemas de la familia RV30 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB30001	$5{,}71e-01_{1{,}3e-01}$	$6{,}79e-01_{1{,}9e-01}$	$1{,}21e-01_{2{,}4e-01}$	$8{,}77e-01_{1{,}5e-01}$	$8{,}77e-01_{1{,}5e-01}$	$7{,}87e-01_{1{,}2e-01}$
BB30002	$8{,}75e-01_{1{,}4e-02}$	$8{,}90e-01_{2{,}5e-02}$	$8{,}81e-01_{2{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}71e-01_{1{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB30003	$5{,}61e-01_{6{,}1e-02}$	$5{,}56e-01_{8{,}2e-02}$	$5{,}63e-01_{9{,}5e-02}$	$5{,}63e-01_{9{,}5e-02}$	$9{,}64e-01_{3{,}1e-02}$	$5{,}97e-01_{1{,}7e-01}$
BB30004	$9{,}02e-01_{3{,}2e-02}$	$8{,}51e-01_{5{,}9e-02}$	$9{,}02e-01_{2{,}6e-02}$	$8{,}97e-01_{3{,}8e-02}$	$8{,}93e-01_{3{,}1e-02}$	$8{,}97e-01_{3{,}8e-02}$
BB30005	$5{,}86e-01_{8{,}6e-02}$	$5{,}73e-01_{2{,}0e-01}$	$6{,}60e-01_{3{,}8e-02}$	$5{,}73e-01_{2{,}0e-01}$	$7{,}44e-01_{5{,}3e-02}$	$5{,}72e-01_{1{,}4e-01}$
BB30006	$8{,}35e-01_{2{,}3e-02}$	$7{,}88e-01_{6{,}7e-02}$	$8{,}62e-01_{1{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}59e-01_{1{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB30007	$8{,}67e-01_{2{,}9e-02}$	$8{,}42e-01_{5{,}5e-02}$	$8{,}78e-01_{5{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}64e-01_{3{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB30008	$8{,}95e-01_{2{,}9e-02}$	$8{,}88e-01_{3{,}2e-02}$	$9{,}05e-01_{3{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}84e-01_{3{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB30010	$9{,}02e-01_{2{,}0e-02}$	$9{,}03e-01_{1{,}7e-02}$	$9{,}10e-01_{1{,}3e-02}$	$9{,}15e-01_{1{,}3e-02}$	$9{,}15e-01_{1{,}1e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB30011	$3{,}93e-01_{1{,}1e-01}$	$4{,}89e-01_{5{,}5e-02}$	$4{,}78e-01_{9{,}9e-02}$	$5{,}09e-01_{7{,}4e-02}$	$5{,}38e-01_{7{,}0e-02}$	$5{,}38e-01_{7{,}0e-02}$
BB30012	$5{,}66e-01_{1{,}0e-01}$	$4{,}67e-01_{3{,}2e-01}$	$6{,}36e-01_{6{,}7e-02}$	$6{,}19e-01_{4{,}9e-02}$	$6{,}19e-01_{4{,}2e-02}$	$6{,}19e-01_{4{,}9e-02}$
BB30013	$4{,}02e-01_{1{,}2e-01}$	$3{,}10e-01_{2{,}9e-01}$	$4{,}71e-01_{7{,}8e-02}$	$5{,}23e-01_{4{,}2e-01}$	$8{,}92e-01_{8{,}4e-02}$	$5{,}23e-01_{4{,}2e-01}$
BB30014	$9{,}08e-01_{8{,}2e-03}$	$9{,}05e-01_{2{,}4e-02}$	$9{,}24e-01_{7{,}1e-03}$	$1{,}00e+00_{0{,}0e+00}$	$9{,}12e-01_{8{,}8e-03}$	$1{,}00e+00_{0{,}0e+00}$
BB30015	$8{,}64e-01_{4{,}4e-02}$	$8{,}49e-01_{7{,}9e-02}$	$8{,}80e-01_{3{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}93e-01_{1{,}9e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB30017	$7{,}24e-01_{7{,}4e-02}$	$6{,}57e-01_{5{,}8e-02}$	$7{,}63e-01_{5{,}8e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}80e-01_{3{,}6e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB30018	$7{,}01e-01_{8{,}0e-02}$	$6{,}11e-01_{2{,}1e-01}$	$7{,}15e-01_{7{,}4e-02}$	$7{,}09e-01_{6{,}1e-02}$	$7{,}15e-01_{6{,}9e-02}$	$7{,}09e-01_{6{,}1e-02}$
BB30019	$7{,}40e-01_{3{,}4e-02}$	$6{,}73e-01_{9{,}9e-02}$	$7{,}87e-01_{3{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}87e-01_{3{,}0e-02}$	$1{,}00e+00_{2{,}0e-01}$
BB30021	$4{,}94e-01_{1{,}2e-01}$	$4{,}78e-01_{1{,}1e-01}$	$6{,}11e-01_{9{,}4e-02}$	$6{,}65e-01_{2{,}9e-01}$	$9{,}14e-01_{9{,}1e-02}$	$6{,}65e-01_{2{,}9e-01}$
BB30022	$9{,}00e-01_{1{,}2e-02}$	$9{,}10e-01_{3{,}0e-02}$	$9{,}14e-01_{1{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}94e-01_{1{,}4e-02}$	$1{,}00e+00_{1{,}0e-01}$
BB30024	$3{,}71e-01_{1{,}3e-01}$	$4{,}62e-01_{1{,}2e-01}$	$3{,}49e-01_{8{,}5e-02}$	$4{,}68e-01_{6{,}0e-01}$	$9{,}31e-01_{4{,}2e-02}$	$4{,}68e-01_{6{,}0e-01}$
BB30025	$7{,}96e-01_{6{,}0e-02}$	$8{,}26e-01_{4{,}2e-02}$	$8{,}44e-01_{4{,}2e-02}$	$8{,}26e-01_{4{,}2e-02}$	$8{,}12e-01_{3{,}2e-02}$	$8{,}26e-01_{4{,}2e-02}$
BB30026	$4{,}65e-01_{1{,}6e-01}$	$6{,}33e-01_{1{,}8e-01}$	$5{,}75e-01_{1{,}3e-01}$	$6{,}33e-01_{1{,}8e-01}$	$7{,}09e-01_{5{,}8e-02}$	$6{,}33e-01_{1{,}8e-01}$
BB30027	$8{,}42e-01_{2{,}6e-02}$	$8{,}42e-01_{2{,}6e-02}$	$8{,}65e-01_{1{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}70e-01_{1{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB30028	$5{,}50e-01_{1{,}3e-01}$	$8{,}31e-01_{1{,}7e-01}$	$5{,}16e-01_{2{,}7e-01}$	$6{,}93e-01_{2{,}4e-01}$	$8{,}31e-01_{1{,}7e-01}$	$6{,}93e-01_{2{,}4e-01}$
BB30029	$5{,}79e-01_{9{,}4e-02}$	$6{,}56e-01_{6{,}1e-02}$	$6{,}56e-01_{6{,}1e-02}$	$7{,}16e-01_{1{,}9e-01}$	$8{,}31e-01_{1{,}0e-01}$	$7{,}16e-01_{1{,}9e-01}$
BB30030	$2{,}15e-01_{8{,}0e-02}$	$2{,}37e-01_{9{,}9e-02}$	$2{,}42e-01_{9{,}4e-02}$	$2{,}37e-01_{9{,}9e-02}$	$2{,}37e-01_{9{,}9e-02}$	$2{,}37e-01_{9{,}9e-02}$

Tabla B.11: Mediana y rango intercuartílico del indicador $I_{\epsilon+}$ para los problemas de la familia RV40 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB40001	$3{,}44e-01_{2{,}5e-02}$	$3{,}95e-01_{4{,}1e-02}$	$3{,}91e-01_{1{,}8e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}62e-01_{1{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40002	$2{,}22e-01_{3{,}0e-02}$	$2{,}22e-01_{4{,}4e-02}$	$2{,}22e-01_{4{,}4e-02}$	$8{,}13e-01_{2{,}2e-02}$	$8{,}13e-01_{2{,}2e-02}$	$8{,}13e-01_{2{,}2e-02}$
BB40003	$3{,}14e-01_{4{,}7e-02}$	$4{,}09e-01_{5{,}5e-02}$	$4{,}06e-01_{5{,}2e-02}$	$3{,}83e-01_{8{,}7e-02}$	$4{,}27e-01_{3{,}6e-02}$	$2{,}61e-01_{3{,}7e-02}$
BB40004	$8{,}94e-01_{2{,}5e-02}$	$9{,}10e-01_{4{,}5e-02}$	$9{,}08e-01_{2{,}1e-02}$	$9{,}08e-01_{8{,}7e-02}$	$8{,}96e-01_{3{,}8e-02}$	$9{,}08e-01_{8{,}7e-02}$
BB40005	$7{,}89e-01_{2{,}1e-02}$	$7{,}80e-01_{3{,}9e-02}$	$8{,}02e-01_{2{,}6e-02}$	$7{,}61e-01_{9{,}1e-02}$	$8{,}17e-01_{1{,}8e-02}$	$7{,}41e-01_{1{,}9e-01}$
BB40006	$7{,}81e-01_{1{,}4e-02}$	$8{,}22e-01_{1{,}2e-02}$	$8{,}08e-01_{1{,}9e-02}$	$7{,}91e-01_{1{,}2e-01}$	$8{,}15e-01_{1{,}2e-02}$	$7{,}49e-01_{7{,}1e-02}$
BB40007	$3{,}00e-01_{1{,}7e-02}$	$2{,}19e-01_{7{,}0e-02}$	$3{,}29e-01_{9{,}8e-03}$	$7{,}75e-01_{3{,}6e-02}$	$4{,}23e-01_{4{,}9e-02}$	$8{,}19e-01_{4{,}3e-02}$
BB40008	$7{,}60e-01_{2{,}2e-02}$	$7{,}43e-01_{7{,}3e-02}$	$7{,}89e-01_{2{,}5e-02}$	$7{,}93e-01_{2{,}2e-01}$	$8{,}10e-01_{1{,}2e-02}$	$7{,}33e-01_{3{,}5e-01}$
BB40009	$8{,}68e-01_{1{,}1e-02}$	$8{,}44e-01_{4{,}6e-02}$	$8{,}78e-01_{2{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}80e-01_{1{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40010	$4{,}92e-01_{7{,}1e-02}$	$4{,}18e-01_{8{,}4e-02}$	$5{,}26e-01_{5{,}8e-02}$	$4{,}37e-01_{7{,}5e-02}$	$5{,}10e-01_{4{,}5e-02}$	$3{,}26e-01_{7{,}1e-02}$
BB40011	$3{,}74e-01_{4{,}0e-02}$	$4{,}58e-01_{4{,}7e-02}$	$4{,}35e-01_{3{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}47e-01_{1{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40012	$8{,}17e-01_{3{,}2e-02}$	$8{,}56e-01_{1{,}2e-02}$	$8{,}34e-01_{2{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}37e-01_{2{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40013	$7{,}34e-01_{3{,}2e-02}$	$7{,}59e-01_{5{,}6e-02}$	$7{,}26e-01_{5{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}29e-01_{4{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40014	$3{,}63e-01_{5{,}7e-02}$	$4{,}10e-01_{5{,}8e-02}$	$4{,}23e-01_{3{,}8e-02}$	$6{,}90e-01_{8{,}3e-02}$	$4{,}28e-01_{3{,}7e-02}$	$3{,}84e-01_{3{,}9e-02}$
BB40015	$6{,}16e-02_{1{,}4e-02}$	$1{,}35e-01_{2{,}2e-02}$	$1{,}49e-01_{1{,}8e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}27e-01_{8{,}4e-03}$	$1{,}00e+00_{2{,}6e-01}$
BB40016	$1{,}00e+00_{1{,}8e-01}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{1{,}8e-01}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$
BB40017	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{1{,}4e-01}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$
BB40018	$7{,}09e-01_{2{,}6e-02}$	$7{,}64e-01_{1{,}6e-02}$	$7{,}60e-01_{2{,}9e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}70e-01_{7{,}0e-03}$	$5{,}74e-01_{2{,}2e-01}$
BB40019	$2{,}10e-01_{9{,}2e-02}$	$2{,}27e-01_{4{,}9e-02}$	$3{,}12e-01_{4{,}0e-02}$	$5{,}40e-01_{6{,}3e-02}$	$2{,}97e-01_{5{,}2e-02}$	$2{,}14e-01_{3{,}6e-02}$
BB40020	$8{,}79e-01_{1{,}6e-02}$	$8{,}76e-01_{7{,}1e-02}$	$8{,}99e-01_{1{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}76e-01_{1{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40021	$6{,}11e-01_{3{,}9e-02}$	$6{,}73e-01_{3{,}9e-02}$	$6{,}36e-01_{3{,}6e-02}$	$1{,}00e+00_{0{,}0e+00}$	$6{,}52e-01_{3{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40022	$2{,}64e-01_{4{,}3e-02}$	$2{,}84e-01_{8{,}6e-02}$	$3{,}15e-01_{5{,}8e-02}$	$1{,}00e+00_{0{,}0e+00}$	$5{,}55e-01_{4{,}9e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40023	$6{,}98e-01_{1{,}5e-02}$	$6{,}82e-01_{9{,}6e-02}$	$7{,}23e-01_{3{,}3e-02}$	$7{,}29e-01_{2{,}9e-01}$	$7{,}11e-01_{3{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40024	$6{,}60e-01_{4{,}7e-02}$	$7{,}03e-01_{5{,}2e-02}$	$7{,}01e-01_{5{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}41e-01_{2{,}6e-02}$	$1{,}00e+00_{2{,}5e-01}$
BB40025	$4{,}62e-01_{5{,}4e-02}$	$4{,}18e-01_{1{,}2e-01}$	$5{,}82e-01_{6{,}8e-02}$	$5{,}23e-01_{5{,}3e-02}$	$6{,}10e-01_{4{,}6e-02}$	$3{,}51e-01_{6{,}6e-02}$
BB40026	$7{,}02e-01_{3{,}2e-02}$	$7{,}31e-01_{2{,}6e-02}$	$7{,}15e-01_{2{,}1e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}14e-01_{1{,}6e-02}$	$1{,}00e+00_{1{,}3e-01}$
BB40027	$5{,}56e-01_{7{,}4e-02}$	$6{,}56e-01_{8{,}7e-02}$	$5{,}32e-01_{7{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}72e-01_{4{,}3e-02}$	$1{,}00e+00_{2{,}3e-01}$
BB40028	$8{,}67e-01_{1{,}4e-02}$	$8{,}31e-01_{6{,}8e-02}$	$8{,}88e-01_{1{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$	$9{,}08e-01_{1{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40029	$6{,}41e-01_{4{,}1e-02}$	$6{,}68e-01_{1{,}0e-01}$	$7{,}55e-01_{8{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}14e-01_{5{,}9e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40030	$8{,}27e-01_{1{,}3e-02}$	$8{,}52e-01_{1{,}7e-02}$	$8{,}47e-01_{1{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}37e-01_{1{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40031	$5{,}20e-01_{2{,}9e-02}$	$5{,}74e-01_{1{,}8e-02}$	$5{,}59e-01_{3{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}80e-01_{9{,}5e-03}$	$1{,}00e+00_{0{,}0e+00}$
BB40032	$1{,}50e-01_{3{,}1e-02}$	$9{,}91e-02_{8{,}9e-02}$	$1{,}74e-01_{1{,}9e-02}$	$6{,}38e-01_{5{,}3e-02}$	$1{,}70e-01_{1{,}8e-02}$	$3{,}85e-01_{3{,}3e-02}$
BB40033	$7{,}87e-01_{1{,}5e-02}$	$8{,}12e-01_{1{,}6e-02}$	$7{,}96e-01_{2{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}91e-01_{2{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40034	$7{,}45e-01_{3{,}5e-02}$	$6{,}24e-01_{6{,}0e-02}$	$7{,}95e-01_{2{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}74e-01_{5{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40035	$4{,}85e-01_{2{,}0e-02}$	$5{,}20e-01_{4{,}5e-02}$	$5{,}22e-01_{2{,}1e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}48e-01_{1{,}6e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40036	$8{,}25e-01_{2{,}4e-02}$	$8{,}55e-01_{1{,}4e-02}$	$8{,}57e-01_{3{,}1e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}31e-01_{3{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40037	$7{,}64e-01_{4{,}5e-02}$	$7{,}65e-01_{1{,}0e-01}$	$7{,}23e-01_{4{,}9e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}53e-01_{9{,}1e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40038	$4{,}76e-01_{6{,}5e-02}$	$5{,}17e-01_{3{,}9e-02}$	$5{,}65e-01_{2{,}1e-02}$	$1{,}00e+00_{0{,}0e+00}$	$5{,}43e-01_{2{,}6e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40039	$8{,}84e-01_{7{,}1e-03}$	$9{,}02e-01_{1{,}1e-02}$	$8{,}93e-01_{1{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}92e-01_{1{,}1e-02}$	$1{,}00e+00_{1{,}1e-01}$
BB40040	$7{,}04e-01_{3{,}3e-02}$	$6{,}79e-01_{3{,}4e-02}$	$7{,}33e-01_{2{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}11e-01_{2{,}6e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40041	$4{,}33e-01_{4{,}6e-02}$	$4{,}53e-01_{4{,}8e-02}$	$4{,}69e-01_{2{,}7e-01}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}57e-01_{2{,}6e-02}$	$1{,}00e+00_{2{,}4e-01}$
BB40042	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$
BB40043	$7{,}28e-01_{4{,}1e-02}$	$7{,}24e-01_{5{,}2e-02}$	$7{,}62e-01_{2{,}9e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}85e-01_{1{,}8e-02}$	$1{,}00e+00_{2{,}8e-02}$
BB40044	$6{,}68e-01_{4{,}1e-02}$	$6{,}99e-01_{4{,}5e-02}$	$7{,}04e-01_{4{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}80e-01_{2{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40045	$8{,}31e-01_{7{,}6e-02}$	$8{,}63e-01_{3{,}4e-02}$	$8{,}53e-01_{2{,}8e-02}$	$7{,}52e-01_{8{,}4e-02}$	$8{,}61e-01_{3{,}2e-02}$	$8{,}02e-01_{5{,}4e-02}$
BB40046	$6{,}61e-01_{4{,}6e-02}$	$6{,}70e-01_{7{,}4e-02}$	$7{,}12e-01_{3{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}09e-01_{2{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40047	$3{,}55e-01_{3{,}1e-02}$	$4{,}63e-01_{6{,}0e-02}$	$4{,}63e-01_{6{,}0e-02}$	$4{,}63e-01_{6{,}0e-02}$	$4{,}86e-01_{2{,}1e-01}$	$4{,}63e-01_{6{,}0e-02}$
BB40048	$7{,}59e-01_{2{,}9e-02}$	$6{,}91e-01_{1{,}3e-01}$	$7{,}83e-01_{2{,}6e-02}$	$7{,}83e-01_{2{,}6e-02}$	$7{,}77e-01_{3{,}1e-02}$	$7{,}38e-01_{7{,}1e-02}$
BB40049	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$	$0{,}00e+00_{0{,}0e+00}$

Tabla B.12: Mediana y rango intercuartílico del indicador $I_{\epsilon+}$ para los problemas de la familia RV50 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB50001	$8{,}47e-01_{5{,}6e-02}$	$7{,}91e-01_{1{,}0e-01}$	$8{,}31e-01_{5{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}47e-01_{6{,}8e-02}$	$1{,}00e+00_{1{,}3e-01}$
BB50002	$3{,}35e-01_{4{,}8e-02}$	$3{,}28e-01_{4{,}1e-02}$	$3{,}77e-01_{2{,}6e-02}$	$1{,}00e+00_{0{,}0e+00}$	$6{,}49e-01_{3{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB50003	$7{,}27e-01_{6{,}3e-02}$	$5{,}15e-01_{8{,}3e-02}$	$7{,}50e-01_{9{,}2e-02}$	$7{,}71e-01_{2{,}6e-01}$	$7{,}44e-01_{2{,}4e-02}$	$7{,}71e-01_{2{,}6e-01}$
BB50004	$3{,}69e-01_{9{,}8e-02}$	$3{,}95e-01_{1{,}0e-01}$	$3{,}88e-01_{1{,}3e-01}$	$4{,}58e-01_{1{,}1e-01}$	$3{,}82e-01_{7{,}1e-02}$	$2{,}87e-01_{8{,}5e-02}$
BB50005	$1{,}57e-01_{5{,}0e-02}$	$2{,}79e-01_{3{,}6e-02}$	$2{,}96e-01_{3{,}7e-02}$	$4{,}65e-01_{9{,}0e-02}$	$2{,}75e-01_{5{,}3e-02}$	$3{,}68e-01_{7{,}2e-02}$
BB50006	$2{,}46e-01_{9{,}1e-03}$	$3{,}09e-01_{7{,}7e-02}$	$3{,}02e-01_{3{,}5e-02}$	$3{,}09e-01_{7{,}7e-02}$	$3{,}46e-01_{6{,}2e-01}$	$3{,}09e-01_{7{,}7e-02}$
BB50007	$6{,}87e-01_{2{,}4e-02}$	$7{,}14e-01_{4{,}1e-02}$	$7{,}25e-01_{1{,}8e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}13e-01_{1{,}9e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB50008	$8{,}63e-01_{2{,}9e-02}$	$8{,}72e-01_{3{,}6e-02}$	$8{,}79e-01_{2{,}6e-02}$	$1{,}00e+00_{3{,}9e-02}$	$8{,}83e-01_{2{,}3e-02}$	$1{,}00e+00_{6{,}1e-02}$
BB50009	$7{,}81e-01_{5{,}3e-02}$	$8{,}06e-01_{2{,}4e-02}$	$8{,}01e-01_{5{,}1e-02}$	$1{,}00e+00_{0{,}0e+00}$	$8{,}03e-01_{3{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB50010	$7{,}14e-01_{2{,}9e-02}$	$7{,}43e-01_{2{,}4e-02}$	$7{,}43e-01_{2{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}49e-01_{1{,}1e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB50011	$6{,}86e-01_{4{,}3e-02}$	$6{,}93e-01_{1{,}1e-01}$	$6{,}76e-01_{7{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$	$6{,}88e-01_{2{,}6e-02}$	$1{,}00e+00_{3{,}1e-01}$
BB50012	$9{,}60e-02_{3{,}1e-02}$	$1{,}55e-01_{6{,}9e-02}$	$8{,}07e-01_{6{,}0e-03}$	$8{,}07e-01_{6{,}0e-03}$	$8{,}07e-01_{6{,}0e-03}$	$8{,}07e-01_{6{,}0e-03}$
BB50013	$5{,}40e-01_{2{,}7e-02}$	$5{,}39e-01_{7{,}9e-02}$	$5{,}51e-01_{4{,}1e-02}$	$4{,}88e-01_{5{,}0e-02}$	$5{,}81e-01_{4{,}2e-02}$	$3{,}61e-01_{4{,}3e-02}$
BB50014	$8{,}20e-01_{8{,}8e-03}$	$8{,}35e-01_{2{,}7e-02}$	$8{,}40e-01_{1{,}0e-02}$	$8{,}38e-01_{1{,}3e-01}$	$8{,}20e-01_{1{,}1e-02}$	$9{,}90e-01_{2{,}8e-02}$
BB50015	$5{,}27e-01_{7{,}3e-02}$	$5{,}65e-01_{1{,}0e-01}$	$5{,}24e-01_{1{,}1e-01}$	$1{,}00e+00_{0{,}0e+00}$	$7{,}71e-01_{4{,}4e-02}$	$1{,}00e+00_{2{,}3e-01}$
BB50016	$3{,}94e-01_{8{,}4e-03}$	$4{,}37e-01_{1{,}3e-02}$	$4{,}36e-01_{1{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$	$6{,}17e-01_{1{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$

Tabla B.13: Mediana y rango intercuartílico del indicador I_Δ para los problemas de la familia RV11 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB11001	$1{,}28e+00_{5{,}7e-02}$	$8{,}01e-01_{1{,}6e-01}$	$1{,}27e+00_{6{,}3e-02}$	$1{,}38e+00_{9{,}4e-02}$	$1{,}26e+00_{3{,}9e-02}$	$1{,}39e+00_{5{,}8e-02}$
BB11002	$8{,}24e-01_{7{,}3e-02}$	$7{,}03e-01_{1{,}1e-01}$	$8{,}46e-01_{8{,}1e-02}$	$1{,}34e+00_{5{,}0e-02}$	$8{,}20e-01_{6{,}5e-02}$	$1{,}45e+00_{8{,}8e-02}$
BB11003	$9{,}97e-01_{9{,}5e-02}$	$8{,}21e-01_{7{,}3e-02}$	$1{,}06e+00_{1{,}0e-01}$	$1{,}26e+00_{7{,}8e-02}$	$1{,}02e+00_{1{,}5e-01}$	$1{,}40e+00_{6{,}7e-02}$
BB11004	$8{,}94e-01_{1{,}6e-01}$	$8{,}69e-01_{9{,}5e-02}$	$1{,}00e+00_{1{,}3e-01}$	$1{,}30e+00_{4{,}1e-02}$	$9{,}93e-01_{2{,}1e-01}$	$1{,}46e+00_{1{,}2e-01}$
BB11005	$7{,}10e-01_{2{,}3e-02}$	$6{,}03e-01_{3{,}8e-02}$	$7{,}46e-01_{4{,}7e-02}$	$1{,}00e+00_{4{,}0e-03}$	$9{,}59e-01_{7{,}9e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB11006	$6{,}75e-01_{3{,}5e-02}$	$5{,}74e-01_{7{,}6e-02}$	$7{,}36e-01_{4{,}3e-02}$	$1{,}06e+00_{6{,}9e-02}$	$6{,}65e-01_{4{,}8e-02}$	$1{,}72e+00_{7{,}3e-02}$
BB11007	$7{,}80e-01_{3{,}6e-02}$	$6{,}55e-01_{6{,}3e-02}$	$7{,}88e-01_{3{,}8e-02}$	$1{,}12e+00_{2{,}3e-02}$	$7{,}61e-01_{4{,}8e-02}$	$1{,}51e+00_{9{,}9e-02}$
BB11008	$7{,}67e-01_{5{,}8e-02}$	$6{,}82e-01_{4{,}1e-02}$	$6{,}44e-01_{6{,}8e-02}$	$1{,}53e+00_{7{,}9e-02}$	$5{,}94e-01_{8{,}1e-02}$	$1{,}52e+00_{5{,}1e-02}$
BB11009	$8{,}29e-01_{6{,}7e-02}$	$8{,}17e-01_{1{,}5e-01}$	$8{,}62e-01_{1{,}2e-01}$	$1{,}18e+00_{2{,}5e-01}$	$8{,}70e-01_{8{,}0e-02}$	$1{,}50e+00_{1{,}4e-01}$
BB11010	$9{,}47e-01_{1{,}1e-01}$	$8{,}78e-01_{9{,}0e-02}$	$9{,}58e-01_{2{,}0e-01}$	$1{,}42e+00_{1{,}2e-01}$	$1{,}04e+00_{9{,}9e-02}$	$1{,}49e+00_{1{,}6e-01}$
BB11011	$7{,}48e-01_{6{,}0e-02}$	$5{,}79e-01_{8{,}3e-02}$	$7{,}51e-01_{8{,}3e-02}$	$1{,}39e+00_{6{,}5e-02}$	$7{,}19e-01_{8{,}6e-02}$	$1{,}62e+00_{1{,}0e-01}$
BB11012	$1{,}28e+00_{6{,}2e-02}$	$8{,}19e-01_{9{,}3e-02}$	$1{,}28e+00_{4{,}7e-02}$	$1{,}30e+00_{1{,}0e-01}$	$1{,}28e+00_{4{,}0e-02}$	$1{,}41e+00_{5{,}7e-02}$
BB11013	$7{,}49e-01_{7{,}0e-02}$	$7{,}94e-01_{9{,}6e-02}$	$8{,}92e-01_{8{,}5e-02}$	$1{,}32e+00_{2{,}5e-02}$	$8{,}70e-01_{7{,}4e-02}$	$1{,}50e+00_{3{,}8e-02}$
BB11014	$7{,}66e-01_{6{,}1e-02}$	$6{,}78e-01_{1{,}3e-01}$	$7{,}20e-01_{4{,}2e-02}$	$1{,}39e+00_{6{,}1e-02}$	$7{,}15e-01_{7{,}4e-02}$	$1{,}60e+00_{1{,}7e-01}$
BB11015	$1{,}17e+00_{8{,}4e-02}$	$8{,}35e-01_{9{,}2e-02}$	$1{,}15e+00_{1{,}2e-01}$	$1{,}15e+00_{8{,}7e-02}$	$1{,}16e+00_{9{,}3e-02}$	$1{,}37e+00_{1{,}4e-01}$
BB11016	$7{,}15e-01_{3{,}6e-02}$	$5{,}88e-01_{7{,}7e-02}$	$7{,}39e-01_{8{,}6e-02}$	$1{,}12e+00_{1{,}2e-02}$	$7{,}22e-01_{6{,}4e-02}$	$1{,}63e+00_{1{,}5e-01}$
BB11017	$1{,}02e+00_{9{,}9e-02}$	$7{,}79e-01_{4{,}0e-02}$	$9{,}93e-01_{1{,}5e-01}$	$1{,}40e+00_{1{,}2e-01}$	$1{,}09e+00_{1{,}3e-01}$	$1{,}48e+00_{8{,}1e-02}$
BB11018	$6{,}03e-01_{4{,}3e-02}$	$4{,}03e-01_{4{,}6e-02}$	$6{,}23e-01_{4{,}1e-02}$	$1{,}01e+00_{8{,}3e-03}$	$9{,}24e-01_{3{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB11019	$6{,}97e-01_{4{,}0e-02}$	$5{,}86e-01_{7{,}3e-02}$	$7{,}00e-01_{4{,}2e-02}$	$1{,}17e+00_{2{,}6e-02}$	$6{,}76e-01_{8{,}7e-02}$	$1{,}59e+00_{1{,}3e-01}$
BB11020	$7{,}12e-01_{5{,}9e-02}$	$5{,}77e-01_{3{,}7e-02}$	$7{,}17e-01_{3{,}9e-02}$	$1{,}31e+00_{1{,}7e-02}$	$6{,}43e-01_{4{,}5e-02}$	$1{,}59e+00_{8{,}4e-02}$
BB11021	$1{,}17e+00_{4{,}0e-02}$	$8{,}05e-01_{6{,}6e-02}$	$1{,}18e+00_{3{,}8e-02}$	$1{,}38e+00_{1{,}0e-01}$	$1{,}19e+00_{4{,}6e-02}$	$1{,}41e+00_{8{,}5e-02}$
BB11022	$8{,}48e-01_{7{,}0e-02}$	$8{,}26e-01_{1{,}3e-01}$	$9{,}04e-01_{1{,}1e-01}$	$1{,}33e+00_{4{,}4e-02}$	$9{,}30e-01_{1{,}7e-01}$	$1{,}55e+00_{6{,}2e-02}$
BB11023	$7{,}07e-01_{4{,}9e-02}$	$5{,}77e-01_{7{,}2e-02}$	$7{,}50e-01_{3{,}9e-02}$	$1{,}29e+00_{1{,}6e-02}$	$7{,}02e-01_{9{,}2e-02}$	$1{,}64e+00_{9{,}8e-02}$
BB11024	$8{,}39e-01_{1{,}4e-01}$	$8{,}39e-01_{1{,}1e-01}$	$9{,}31e-01_{1{,}2e-01}$	$1{,}32e+00_{4{,}9e-02}$	$9{,}54e-01_{1{,}5e-01}$	$1{,}45e+00_{1{,}3e-01}$
BB11025	$8{,}09e-01_{8{,}3e-02}$	$7{,}81e-01_{1{,}3e-01}$	$9{,}83e-01_{1{,}3e-01}$	$1{,}31e+00_{6{,}4e-02}$	$8{,}82e-01_{1{,}5e-01}$	$1{,}58e+00_{1{,}1e-01}$
BB11027	$7{,}67e-01_{6{,}2e-02}$	$5{,}83e-01_{7{,}1e-02}$	$7{,}81e-01_{5{,}2e-02}$	$1{,}22e+00_{1{,}7e-02}$	$7{,}68e-01_{6{,}3e-02}$	$1{,}59e+00_{6{,}9e-02}$
BB11028	$7{,}20e-01_{6{,}1e-02}$	$5{,}99e-01_{3{,}7e-02}$	$7{,}33e-01_{4{,}2e-02}$	$1{,}03e+00_{2{,}9e-02}$	$6{,}82e-01_{3{,}6e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB11029	$1{,}11e+00_{4{,}2e-02}$	$7{,}92e-01_{1{,}1e-01}$	$1{,}11e+00_{4{,}1e-02}$	$1{,}37e+00_{5{,}2e-02}$	$1{,}12e+00_{1{,}1e-01}$	$1{,}50e+00_{8{,}8e-02}$
BB11031	$6{,}41e-01_{4{,}4e-02}$	$6{,}10e-01_{5{,}0e-02}$	$7{,}39e-01_{4{,}2e-02}$	$1{,}02e+00_{1{,}7e-02}$	$9{,}41e-01_{6{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB11032	$7{,}44e-01_{4{,}8e-02}$	$5{,}99e-01_{2{,}7e-02}$	$7{,}05e-01_{3{,}0e-02}$	$1{,}24e+00_{3{,}1e-02}$	$6{,}50e-01_{4{,}5e-02}$	$1{,}55e+00_{4{,}8e-02}$
BB11033	$6{,}29e-01_{5{,}1e-02}$	$5{,}78e-01_{1{,}1e-01}$	$6{,}88e-01_{9{,}0e-02}$	$1{,}02e+00_{1{,}7e-02}$	$6{,}33e-01_{1{,}7e-01}$	$1{,}00e+00_{0{,}0e+00}$
BB11034	$7{,}47e-01_{3{,}3e-02}$	$6{,}75e-01_{3{,}5e-02}$	$8{,}05e-01_{6{,}2e-02}$	$1{,}12e+00_{1{,}2e-02}$	$7{,}85e-01_{6{,}0e-02}$	$1{,}61e+00_{1{,}2e-01}$
BB11035	$7{,}94e-01_{7{,}0e-02}$	$6{,}20e-01_{1{,}3e-01}$	$7{,}36e-01_{6{,}3e-02}$	$1{,}51e+00_{5{,}5e-02}$	$7{,}36e-01_{1{,}2e-01}$	$1{,}58e+00_{3{,}5e-02}$
BB11036	$7{,}16e-01_{3{,}5e-02}$	$6{,}04e-01_{5{,}3e-02}$	$7{,}60e-01_{5{,}3e-02}$	$1{,}19e+00_{2{,}0e-02}$	$7{,}05e-01_{4{,}4e-02}$	$1{,}64e+00_{1{,}3e-01}$
BB11037	$8{,}63e-01_{5{,}5e-02}$	$7{,}92e-01_{9{,}8e-02}$	$8{,}66e-01_{9{,}0e-02}$	$1{,}31e+00_{9{,}0e-02}$	$8{,}84e-01_{1{,}3e-01}$	$1{,}44e+00_{5{,}3e-02}$
BB11038	$6{,}94e-01_{5{,}2e-02}$	$5{,}56e-01_{2{,}8e-02}$	$6{,}91e-01_{2{,}5e-02}$	$1{,}32e+00_{8{,}0e-02}$	$6{,}23e-01_{2{,}2e-02}$	$1{,}63e+00_{8{,}9e-02}$

Tabla B.14: Mediana y rango intercuartílico del indicador I_Δ para los problemas de la familia RV12 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB12001	$6{,}37e-01_{5{,}1e-02}$	$5{,}18e-01_{6{,}6e-02}$	$6{,}41e-01_{4{,}0e-02}$	$1{,}29e+00_{3{,}2e-02}$	$5{,}59e-01_{3{,}4e-02}$	$1{,}70e+00_{1{,}1e-01}$
BB12002	$1{,}20e+00_{4{,}7e-02}$	$8{,}21e-01_{8{,}6e-02}$	$1{,}18e+00_{4{,}2e-02}$	$1{,}25e+00_{8{,}2e-02}$	$1{,}18e+00_{5{,}7e-02}$	$1{,}32e+00_{1{,}2e-01}$
BB12003	$1{,}26e+00_{2{,}3e-02}$	$7{,}09e-01_{4{,}7e-02}$	$1{,}26e+00_{2{,}6e-02}$	$1{,}59e+00_{7{,}3e-02}$	$1{,}25e+00_{3{,}4e-02}$	$1{,}51e+00_{8{,}9e-02}$
BB12004	$8{,}85e-01_{1{,}2e-01}$	$6{,}90e-01_{1{,}2e-01}$	$8{,}84e-01_{1{,}0e-01}$	$1{,}32e+00_{5{,}6e-02}$	$9{,}27e-01_{1{,}6e-01}$	$1{,}61e+00_{8{,}4e-02}$
BB12005	$8{,}58e-01_{1{,}2e-01}$	$6{,}40e-01_{1{,}1e-01}$	$7{,}85e-01_{2{,}2e-01}$	$1{,}44e+00_{6{,}9e-02}$	$7{,}52e-01_{1{,}9e-01}$	$1{,}59e+00_{1{,}1e-01}$
BB12006	$1{,}24e+00_{1{,}0e-01}$	$8{,}38e-01_{2{,}2e-02}$	$1{,}32e+00_{7{,}8e-02}$	$1{,}25e+00_{9{,}1e-02}$	$1{,}30e+00_{1{,}3e-01}$	$1{,}36e+00_{1{,}3e-01}$
BB12007	$6{,}05e-01_{8{,}3e-02}$	$4{,}04e-01_{3{,}6e-02}$	$6{,}01e-01_{4{,}5e-02}$	$1{,}54e+00_{3{,}2e-02}$	$4{,}90e-01_{6{,}2e-02}$	$1{,}71e+00_{6{,}9e-02}$
BB12008	$8{,}81e-01_{2{,}0e-02}$	$8{,}20e-01_{2{,}8e-02}$	$8{,}75e-01_{9{,}2e-03}$	$1{,}04e+00_{3{,}2e-02}$	$8{,}51e-01_{7{,}9e-03}$	$1{,}26e+00_{1{,}0e-01}$
BB12009	$9{,}70e-01_{8{,}8e-02}$	$6{,}00e-01_{5{,}4e-02}$	$9{,}75e-01_{7{,}2e-02}$	$1{,}57e+00_{5{,}1e-02}$	$9{,}46e-01_{1{,}1e-01}$	$1{,}52e+00_{4{,}8e-02}$
BB12010	$7{,}91e-01_{9{,}3e-02}$	$5{,}30e-01_{8{,}8e-02}$	$8{,}09e-01_{1{,}4e-01}$	$1{,}56e+00_{1{,}1e-01}$	$8{,}40e-01_{1{,}4e-01}$	$1{,}64e+00_{6{,}2e-02}$
BB12011	$7{,}74e-01_{3{,}4e-02}$	$6{,}70e-01_{3{,}4e-02}$	$7{,}87e-01_{2{,}0e-02}$	$1{,}06e+00_{5{,}8e-02}$	$8{,}06e-01_{4{,}8e-02}$	$1{,}54e+00_{2{,}5e-01}$
BB12012	$8{,}07e-01_{7{,}6e-02}$	$7{,}27e-01_{1{,}1e-01}$	$7{,}46e-01_{1{,}2e-01}$	$1{,}49e+00_{1{,}0e-01}$	$7{,}95e-01_{1{,}4e-01}$	$1{,}53e+00_{1{,}0e-01}$
BB12013	$7{,}28e-01_{6{,}9e-02}$	$5{,}75e-01_{7{,}5e-02}$	$5{,}72e-01_{4{,}5e-02}$	$1{,}53e+00_{7{,}2e-02}$	$5{,}72e-01_{4{,}0e-02}$	$1{,}62e+00_{1{,}0e-01}$
BB12014	$7{,}73e-01_{3{,}5e-02}$	$6{,}10e-01_{3{,}4e-02}$	$7{,}05e-01_{3{,}3e-02}$	$1{,}43e+00_{1{,}6e-01}$	$6{,}93e-01_{2{,}9e-02}$	$1{,}49e+00_{3{,}2e-02}$
BB12015	$8{,}54e-01_{5{,}1e-02}$	$7{,}10e-01_{6{,}9e-02}$	$8{,}14e-01_{4{,}4e-02}$	$1{,}05e+00_{5{,}1e-02}$	$7{,}83e-01_{5{,}7e-02}$	$1{,}07e+00_{2{,}7e-01}$
BB12016	$8{,}17e-01_{8{,}1e-02}$	$6{,}07e-01_{1{,}1e-01}$	$8{,}35e-01_{1{,}4e-01}$	$1{,}02e+00_{9{,}6e-03}$	$7{,}98e-01_{1{,}6e-01}$	$1{,}09e+00_{2{,}0e-02}$
BB12017	$9{,}01e-01_{9{,}0e-02}$	$7{,}83e-01_{8{,}7e-02}$	$9{,}06e-01_{1{,}7e-01}$	$1{,}43e+00_{1{,}7e-01}$	$9{,}79e-01_{1{,}1e-01}$	$1{,}51e+00_{1{,}5e-01}$
BB12018	$1{,}25e+00_{7{,}2e-02}$	$7{,}72e-01_{1{,}8e-01}$	$1{,}27e+00_{6{,}6e-02}$	$1{,}35e+00_{1{,}9e-01}$	$1{,}25e+00_{6{,}1e-02}$	$1{,}42e+00_{7{,}9e-02}$
BB12019	$7{,}11e-01_{6{,}2e-02}$	$5{,}80e-01_{1{,}6e-01}$	$6{,}81e-01_{6{,}6e-02}$	$1{,}60e+00_{4{,}6e-02}$	$5{,}99e-01_{7{,}2e-02}$	$1{,}53e+00_{7{,}2e-02}$
BB12020	$1{,}22e+00_{9{,}9e-02}$	$8{,}03e-01_{1{,}0e-01}$	$1{,}24e+00_{7{,}5e-02}$	$1{,}41e+00_{8{,}1e-02}$	$1{,}22e+00_{1{,}0e-01}$	$1{,}42e+00_{9{,}6e-02}$
BB12021	$1{,}22e+00_{5{,}3e-02}$	$7{,}41e-01_{6{,}4e-02}$	$1{,}23e+00_{7{,}5e-02}$	$1{,}32e+00_{6{,}7e-02}$	$1{,}23e+00_{3{,}3e-02}$	$1{,}47e+00_{1{,}1e-01}$
BB12022	$7{,}10e-01_{5{,}4e-02}$	$4{,}83e-01_{1{,}8e-01}$	$6{,}62e-01_{4{,}3e-02}$	$1{,}42e+00_{5{,}6e-02}$	$5{,}86e-01_{6{,}5e-02}$	$1{,}50e+00_{3{,}5e-02}$
BB12023	$9{,}73e-01_{8{,}7e-02}$	$6{,}76e-01_{9{,}9e-02}$	$9{,}64e-01_{1{,}8e-01}$	$1{,}49e+00_{4{,}7e-02}$	$9{,}95e-01_{1{,}7e-01}$	$1{,}57e+00_{1{,}2e-01}$
BB12024	$1{,}32e+00_{5{,}2e-02}$	$7{,}60e-01_{6{,}3e-02}$	$1{,}34e+00_{4{,}7e-02}$	$1{,}44e+00_{8{,}7e-02}$	$1{,}31e+00_{4{,}1e-02}$	$1{,}46e+00_{1{,}2e-01}$
BB12025	$9{,}36e-01_{7{,}2e-02}$	$7{,}17e-01_{8{,}5e-02}$	$9{,}75e-01_{1{,}3e-01}$	$1{,}50e+00_{6{,}0e-02}$	$9{,}80e-01_{1{,}2e-01}$	$1{,}49e+00_{1{,}2e-01}$
BB12026	$8{,}86e-01_{2{,}5e-02}$	$8{,}11e-01_{2{,}0e-02}$	$8{,}51e-01_{5{,}0e-03}$	$1{,}09e+00_{4{,}6e-02}$	$8{,}51e-01_{5{,}4e-03}$	$1{,}55e+00_{3{,}1e-01}$
BB12027	$8{,}54e-01_{2{,}9e-02}$	$7{,}02e-01_{7{,}8e-02}$	$8{,}54e-01_{4{,}3e-02}$	$1{,}07e+00_{5{,}6e-02}$	$8{,}14e-01_{1{,}6e-02}$	$1{,}32e+00_{1{,}8e-01}$
BB12028	$6{,}09e-01_{4{,}2e-02}$	$3{,}92e-01_{3{,}6e-02}$	$5{,}55e-01_{6{,}0e-02}$	$1{,}49e+00_{1{,}7e-02}$	$4{,}65e-01_{4{,}7e-02}$	$1{,}74e+00_{8{,}6e-02}$
BB12029	$8{,}12e-01_{2{,}1e-02}$	$7{,}25e-01_{3{,}5e-02}$	$8{,}02e-01_{2{,}2e-02}$	$1{,}11e+00_{1{,}2e-02}$	$7{,}67e-01_{1{,}8e-02}$	$1{,}35e+00_{6{,}1e-02}$
BB12030	$7{,}44e-01_{4{,}9e-02}$	$5{,}96e-01_{1{,}5e-01}$	$7{,}42e-01_{1{,}0e-01}$	$1{,}32e+00_{8{,}6e-02}$	$7{,}02e-01_{9{,}0e-02}$	$1{,}49e+00_{1{,}3e-01}$
BB12031	$5{,}81e-01_{4{,}6e-02}$	$3{,}57e-01_{3{,}8e-02}$	$5{,}70e-01_{3{,}8e-02}$	$1{,}30e+00_{3{,}3e-02}$	$5{,}52e-01_{4{,}5e-02}$	$1{,}69e+00_{1{,}0e-01}$
BB12032	$1{,}12e+00_{5{,}6e-02}$	$5{,}75e-01_{6{,}3e-02}$	$1{,}14e+00_{3{,}2e-02}$	$1{,}62e+00_{9{,}7e-02}$	$1{,}15e+00_{2{,}7e-02}$	$1{,}58e+00_{8{,}6e-02}$
BB12033	$6{,}78e-01_{3{,}1e-02}$	$5{,}15e-01_{3{,}9e-02}$	$6{,}42e-01_{2{,}8e-02}$	$1{,}45e+00_{7{,}6e-02}$	$5{,}81e-01_{2{,}9e-02}$	$1{,}60e+00_{4{,}5e-02}$
BB12034	$1{,}29e+00_{9{,}2e-02}$	$8{,}27e-01_{1{,}2e-01}$	$1{,}29e+00_{1{,}5e-01}$	$1{,}31e+00_{1{,}7e-01}$	$1{,}23e+00_{4{,}1e-01}$	$1{,}41e+00_{1{,}1e-01}$
BB12035	$8{,}71e-01_{1{,}8e-02}$	$8{,}17e-01_{2{,}8e-02}$	$8{,}63e-01_{1{,}1e-02}$	$1{,}01e+00_{8{,}9e-03}$	$9{,}35e-01_{6{,}6e-03}$	$1{,}00e+00_{3{,}4e-02}$
BB12036	$1{,}27e+00_{2{,}1e-02}$	$7{,}71e-01_{1{,}2e-01}$	$1{,}25e+00_{7{,}1e-02}$	$1{,}55e+00_{8{,}8e-02}$	$1{,}25e+00_{4{,}2e-02}$	$1{,}48e+00_{1{,}3e-01}$
BB12037	$9{,}31e-01_{1{,}5e-02}$	$9{,}02e-01_{1{,}9e-02}$	$9{,}39e-01_{1{,}1e-01}$	$1{,}00e+00_{5{,}9e-03}$	$9{,}42e-01_{1{,}2e-01}$	$1{,}00e+00_{1{,}6e-01}$
BB12038	$8{,}69e-01_{3{,}3e-02}$	$7{,}61e-01_{8{,}4e-02}$	$8{,}59e-01_{3{,}8e-02}$	$1{,}07e+00_{2{,}2e-01}$	$8{,}27e-01_{4{,}1e-02}$	$1{,}19e+00_{1{,}0e-01}$
BB12039	$7{,}81e-01_{6{,}0e-02}$	$5{,}80e-01_{1{,}3e-01}$	$6{,}79e-01_{1{,}6e-01}$	$1{,}44e+00_{2{,}4e-01}$	$6{,}91e-01_{1{,}5e-01}$	$1{,}67e+00_{1{,}1e-01}$
BB12040	$1{,}23e+00_{3{,}5e-02}$	$6{,}88e-01_{8{,}8e-02}$	$1{,}25e+00_{4{,}3e-02}$	$1{,}48e+00_{1{,}2e-01}$	$1{,}24e+00_{4{,}2e-02}$	$1{,}47e+00_{1{,}1e-01}$
BB12041	$1{,}07e+00_{7{,}7e-02}$	$6{,}30e-01_{6{,}2e-02}$	$1{,}07e+00_{8{,}8e-02}$	$1{,}53e+00_{9{,}0e-02}$	$1{,}07e+00_{8{,}7e-02}$	$1{,}57e+00_{1{,}2e-01}$
BB12042	$1{,}20e+00_{7{,}0e-02}$	$8{,}49e-01_{6{,}7e-02}$	$1{,}21e+00_{8{,}5e-02}$	$1{,}16e+00_{7{,}3e-02}$	$1{,}22e+00_{4{,}5e-02}$	$1{,}37e+00_{1{,}1e-01}$
BB12043	$7{,}53e-01_{2{,}8e-02}$	$6{,}16e-01_{3{,}1e-02}$	$7{,}82e-01_{2{,}2e-02}$	$1{,}00e+00_{2{,}3e-02}$	$9{,}77e-01_{1{,}1e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB12044	$8{,}06e-01_{6{,}5e-02}$	$6{,}94e-01_{9{,}8e-02}$	$7{,}73e-01_{5{,}2e-02}$	$1{,}25e+00_{7{,}3e-01}$	$7{,}47e-01_{5{,}6e-02}$	$1{,}47e+00_{1{,}9e-01}$

Tabla B.15: Mediana y rango intercuartílico del indicador I_Δ para los problemas de la familia RV20 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB20001	$8{,}36e-01_{4{,}3e-02}$	$6{,}43e-01_{2{,}3e-01}$	$8{,}60e-01_{9{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}02e+00_{1{,}3e-01}$	$1{,}00e+00_{0{,}0e+00}$
BB20002	$6{,}86e-01_{4{,}2e-02}$	$6{,}80e-01_{1{,}2e-01}$	$8{,}00e-01_{9{,}2e-02}$	$1{,}17e+00_{3{,}4e-02}$	$1{,}17e+00_{3{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20003	$7{,}23e-01_{7{,}5e-02}$	$6{,}36e-01_{1{,}3e-01}$	$7{,}71e-01_{2{,}7e-01}$	$9{,}70e-01_{2{,}8e-01}$	$1{,}03e+00_{6{,}7e-02}$	$9{,}70e-01_{2{,}8e-01}$
BB20004	$6{,}09e-01_{6{,}4e-02}$	$5{,}53e-01_{6{,}3e-02}$	$6{,}54e-01_{1{,}4e-01}$	$9{,}76e-01_{5{,}6e-02}$	$9{,}76e-01_{5{,}6e-02}$	$7{,}24e-01_{3{,}4e-01}$
BB20005	$8{,}01e-01_{4{,}7e-02}$	$7{,}25e-01_{2{,}8e-02}$	$9{,}21e-01_{2{,}1e-02}$	$7{,}44e-01_{3{,}7e-01}$	$1{,}08e+00_{7{,}3e-03}$	$7{,}44e-01_{3{,}7e-01}$
BB20006	$8{,}79e-01_{1{,}6e-02}$	$8{,}31e-01_{3{,}1e-02}$	$8{,}62e-01_{2{,}0e-02}$	$9{,}12e-01_{3{,}8e-02}$	$9{,}12e-01_{3{,}8e-02}$	$1{,}00e+00_{8{,}9e-02}$
BB20007	$7{,}93e-01_{2{,}5e-02}$	$6{,}84e-01_{2{,}7e-02}$	$8{,}18e-01_{1{,}8e-02}$	$9{,}57e-01_{1{,}4e-02}$	$9{,}57e-01_{1{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20008	$5{,}39e-01_{1{,}6e-01}$	$4{,}58e-01_{1{,}7e-01}$	$4{,}58e-01_{1{,}7e-01}$	$4{,}58e-01_{1{,}7e-01}$	$4{,}58e-01_{1{,}7e-01}$	$4{,}58e-01_{1{,}7e-01}$
BB20009	$8{,}64e-01_{3{,}0e-02}$	$8{,}43e-01_{7{,}1e-02}$	$9{,}00e-01_{5{,}3e-02}$	$1{,}02e+00_{8{,}3e-02}$	$1{,}02e+00_{8{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20010	$8{,}53e-01_{2{,}1e-02}$	$7{,}83e-01_{4{,}8e-02}$	$8{,}97e-01_{3{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}07e+00_{6{,}1e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20011	$8{,}02e-01_{4{,}5e-02}$	$6{,}69e-01_{1{,}2e-01}$	$7{,}88e-01_{1{,}9e-01}$	$9{,}49e-01_{2{,}4e-01}$	$9{,}49e-01_{2{,}4e-01}$	$1{,}00e+00_{0{,}0e+00}$
BB20012	$7{,}54e-01_{3{,}5e-02}$	$5{,}99e-01_{1{,}1e-01}$	$8{,}17e-01_{3{,}2e-02}$	$5{,}99e-01_{1{,}1e-01}$	$9{,}73e-01_{2{,}0e-02}$	$9{,}71e-01_{1{,}4e-01}$
BB20013	$7{,}61e-01_{3{,}5e-02}$	$6{,}19e-01_{6{,}8e-02}$	$8{,}12e-01_{2{,}2e-02}$	$9{,}75e-01_{1{,}8e-01}$	$9{,}79e-01_{1{,}1e-02}$	$9{,}75e-01_{1{,}8e-01}$
BB20014	$7{,}74e-01_{4{,}3e-02}$	$7{,}29e-01_{5{,}5e-02}$	$8{,}52e-01_{1{,}9e-02}$	$7{,}29e-01_{5{,}5e-02}$	$1{,}09e+00_{4{,}8e-02}$	$1{,}00e+00_{2{,}6e-01}$
BB20015	$7{,}19e-01_{3{,}8e-02}$	$6{,}84e-01_{4{,}1e-02}$	$8{,}40e-01_{3{,}9e-02}$	$6{,}84e-01_{4{,}1e-02}$	$1{,}05e+00_{4{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20016	$8{,}27e-01_{3{,}1e-02}$	$7{,}69e-01_{1{,}0e-01}$	$8{,}35e-01_{9{,}6e-03}$	$7{,}69e-01_{1{,}0e-01}$	$9{,}68e-01_{1{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20017	$8{,}40e-01_{1{,}9e-02}$	$7{,}12e-01_{4{,}5e-02}$	$8{,}68e-01_{2{,}5e-02}$	$1{,}00e+00_{1{,}9e-01}$	$1{,}06e+00_{3{,}7e-02}$	$1{,}00e+00_{1{,}9e-01}$
BB20018	$9{,}15e-01_{2{,}3e-02}$	$8{,}80e-01_{2{,}7e-02}$	$9{,}15e-01_{4{,}6e-02}$	$8{,}80e-01_{2{,}7e-02}$	$9{,}58e-01_{5{,}1e-02}$	$9{,}93e-01_{7{,}6e-02}$
BB20019	$8{,}49e-01_{4{,}3e-02}$	$7{,}93e-01_{7{,}9e-02}$	$9{,}19e-01_{4{,}0e-02}$	$7{,}93e-01_{7{,}9e-02}$	$1{,}09e+00_{2{,}4e-02}$	$1{,}00e+00_{7{,}5e-02}$
BB20020	$8{,}98e-01_{1{,}3e-02}$	$8{,}22e-01_{5{,}1e-02}$	$8{,}89e-01_{2{,}7e-02}$	$8{,}89e-01_{2{,}7e-02}$	$8{,}76e-01_{2{,}4e-02}$	$1{,}20e+00_{5{,}3e-02}$
BB20021	$6{,}84e-01_{4{,}2e-02}$	$5{,}95e-01_{8{,}8e-02}$	$7{,}62e-01_{8{,}9e-02}$	$7{,}62e-01_{8{,}9e-02}$	$1{,}08e+00_{4{,}5e-02}$	$8{,}08e-01_{3{,}4e-01}$
BB20022	$8{,}86e-01_{3{,}5e-02}$	$8{,}35e-01_{8{,}4e-02}$	$9{,}15e-01_{8{,}7e-02}$	$9{,}76e-01_{8{,}7e-02}$	$9{,}77e-01_{5{,}4e-03}$	$9{,}76e-01_{8{,}7e-02}$
BB20023	$8{,}37e-01_{2{,}8e-02}$	$7{,}89e-01_{2{,}4e-02}$	$8{,}57e-01_{1{,}7e-02}$	$8{,}57e-01_{1{,}7e-02}$	$9{,}76e-01_{3{,}8e-03}$	$1{,}00e+00_{0{,}0e+00}$
BB20024	$7{,}08e-01_{3{,}6e-02}$	$6{,}92e-01_{1{,}1e-01}$	$7{,}99e-01_{7{,}8e-02}$	$1{,}00e+00_{2{,}2e-01}$	$1{,}08e+00_{9{,}0e-02}$	$1{,}00e+00_{2{,}2e-01}$
BB20025	$7{,}13e-01_{5{,}6e-02}$	$5{,}63e-01_{1{,}4e-01}$	$7{,}29e-01_{5{,}8e-02}$	$7{,}29e-01_{5{,}8e-02}$	$9{,}86e-01_{6{,}2e-02}$	$8{,}03e-01_{2{,}5e-01}$
BB20026	$7{,}82e-01_{2{,}7e-02}$	$7{,}29e-01_{8{,}7e-02}$	$8{,}11e-01_{3{,}7e-02}$	$8{,}11e-01_{3{,}7e-02}$	$9{,}79e-01_{7{,}6e-03}$	$9{,}74e-01_{1{,}9e-01}$
BB20027	$7{,}32e-01_{3{,}3e-02}$	$6{,}19e-01_{4{,}5e-02}$	$7{,}81e-01_{2{,}0e-02}$	$7{,}81e-01_{2{,}0e-02}$	$9{,}91e-01_{2{,}4e-02}$	$9{,}84e-01_{2{,}2e-01}$
BB20028	$8{,}37e-01_{3{,}5e-02}$	$7{,}77e-01_{4{,}4e-02}$	$8{,}65e-01_{4{,}3e-02}$	$8{,}65e-01_{4{,}3e-02}$	$9{,}97e-01_{3{,}6e-02}$	$9{,}83e-01_{1{,}3e-01}$
BB20030	$8{,}61e-01_{6{,}2e-02}$	$8{,}48e-01_{1{,}1e-01}$	$9{,}07e-01_{5{,}7e-02}$	$8{,}48e-01_{1{,}1e-01}$	$1{,}05e+00_{1{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20031	$8{,}54e-01_{2{,}1e-02}$	$7{,}90e-01_{1{,}2e-02}$	$8{,}93e-01_{1{,}9e-02}$	$1{,}00e+00_{0{,}0e+00}$	$9{,}89e-01_{1{,}1e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20032	$8{,}70e-01_{1{,}6e-02}$	$8{,}16e-01_{3{,}2e-02}$	$8{,}92e-01_{1{,}9e-02}$	$8{,}16e-01_{3{,}2e-02}$	$9{,}68e-01_{7{,}3e-03}$	$1{,}00e+00_{0{,}0e+00}$
BB20033	$8{,}27e-01_{1{,}3e-02}$	$7{,}57e-01_{1{,}4e-02}$	$8{,}17e-01_{1{,}3e-02}$	$7{,}57e-01_{1{,}4e-02}$	$9{,}13e-01_{1{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20034	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$
BB20035	$8{,}71e-01_{2{,}9e-02}$	$8{,}07e-01_{3{,}9e-02}$	$9{,}10e-01_{2{,}8e-02}$	$8{,}07e-01_{3{,}9e-02}$	$1{,}04e+00_{4{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20036	$8{,}08e-01_{5{,}0e-02}$	$7{,}58e-01_{1{,}9e-02}$	$7{,}79e-01_{1{,}8e-01}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}10e+00_{6{,}6e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20037	$6{,}61e-01_{3{,}4e-02}$	$5{,}77e-01_{8{,}2e-02}$	$6{,}88e-01_{4{,}5e-02}$	$5{,}77e-01_{8{,}2e-02}$	$1{,}01e+00_{5{,}2e-02}$	$1{,}01e+00_{5{,}2e-02}$
BB20038	$8{,}43e-01_{2{,}5e-02}$	$7{,}97e-01_{6{,}0e-02}$	$8{,}57e-01_{2{,}7e-02}$	$7{,}97e-01_{6{,}0e-02}$	$9{,}50e-01_{7{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB20039	$6{,}33e-01_{5{,}3e-02}$	$5{,}56e-01_{9{,}9e-02}$	$7{,}37e-01_{6{,}8e-02}$	$7{,}80e-01_{3{,}1e-01}$	$1{,}05e+00_{4{,}2e-02}$	$7{,}80e-01_{3{,}1e-01}$
BB20040	$5{,}51e-01_{6{,}9e-02}$	$3{,}82e-01_{6{,}3e-02}$	$5{,}92e-01_{3{,}9e-02}$	$3{,}82e-01_{6{,}3e-02}$	$9{,}77e-01_{9{,}4e-03}$	$6{,}20e-01_{3{,}8e-01}$
BB20041	$5{,}33e-01_{1{,}4e-01}$	$4{,}54e-01_{9{,}1e-02}$	$5{,}48e-01_{1{,}7e-01}$	$4{,}54e-01_{9{,}1e-02}$	$6{,}28e-01_{5{,}3e-01}$	$5{,}48e-01_{1{,}7e-01}$

Tabla B.16: Mediana y rango intercuartílico del indicador I_Δ para los problemas de la familia RV30 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB30001	$6{,}67e-01_{6{,}0e-02}$	$5{,}67e-01_{1{,}1e-01}$	$5{,}59e-01_{8{,}9e-02}$	$9{,}93e-01_{1{,}4e-02}$	$9{,}93e-01_{1{,}4e-02}$	$6{,}28e-01_{5{,}3e-02}$
BB30002	$8{,}71e-01_{3{,}2e-02}$	$7{,}73e-01_{3{,}7e-02}$	$8{,}62e-01_{1{,}8e-02}$	$1{,}00e+00_{3{,}7e-04}$	$9{,}84e-01_{1{,}9e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB30003	$7{,}02e-01_{5{,}2e-02}$	$6{,}50e-01_{7{,}1e-02}$	$8{,}22e-01_{3{,}0e-02}$	$8{,}22e-01_{3{,}0e-02}$	$1{,}01e+00_{2{,}7e-02}$	$6{,}35e-01_{1{,}2e-02}$
BB30004	$8{,}19e-01_{1{,}6e-02}$	$7{,}49e-01_{3{,}6e-02}$	$8{,}43e-01_{2{,}1e-02}$	$9{,}91e-01_{9{,}2e-03}$	$9{,}90e-01_{5{,}5e-03}$	$9{,}91e-01_{9{,}2e-03}$
BB30005	$6{,}99e-01_{6{,}1e-02}$	$5{,}54e-01_{9{,}5e-02}$	$7{,}31e-01_{4{,}2e-02}$	$5{,}54e-01_{9{,}5e-02}$	$9{,}89e-01_{4{,}5e-03}$	$6{,}58e-01_{5{,}5e-02}$
BB30006	$8{,}63e-01_{1{,}7e-02}$	$7{,}63e-01_{1{,}9e-01}$	$8{,}57e-01_{9{,}5e-02}$	$1{,}00e+00_{1{,}5e-03}$	$9{,}24e-01_{1{,}3e-01}$	$1{,}00e+00_{0{,}0e+00}$
BB30007	$8{,}98e-01_{2{,}9e-02}$	$8{,}55e-01_{1{,}3e-01}$	$8{,}73e-01_{2{,}2e-02}$	$1{,}00e+00_{4{,}5e-03}$	$9{,}50e-01_{8{,}9e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB30008	$8{,}37e-01_{1{,}5e-02}$	$7{,}42e-01_{4{,}5e-02}$	$8{,}46e-01_{2{,}2e-02}$	$1{,}00e+00_{4{,}4e-04}$	$9{,}93e-01_{4{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB30010	$8{,}31e-01_{2{,}1e-02}$	$7{,}67e-01_{5{,}5e-02}$	$8{,}57e-01_{3{,}0e-02}$	$8{,}92e-01_{1{,}5e-01}$	$1{,}01e+00_{5{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB30011	$6{,}39e-01_{5{,}7e-02}$	$5{,}25e-01_{6{,}2e-02}$	$6{,}33e-01_{5{,}5e-02}$	$6{,}85e-01_{2{,}4e-01}$	$8{,}72e-01_{1{,}0e-02}$	$8{,}70e-01_{1{,}3e-02}$
BB30012	$7{,}19e-01_{5{,}3e-02}$	$5{,}75e-01_{2{,}0e-01}$	$7{,}32e-01_{2{,}5e-02}$	$7{,}78e-01_{2{,}6e-01}$	$9{,}81e-01_{9{,}2e-03}$	$7{,}78e-01_{2{,}6e-01}$
BB30013	$6{,}39e-01_{6{,}8e-02}$	$5{,}03e-01_{1{,}1e-01}$	$7{,}18e-01_{3{,}4e-02}$	$8{,}00e-01_{2{,}9e-01}$	$1{,}01e+00_{3{,}0e-02}$	$8{,}00e-01_{2{,}9e-01}$
BB30014	$8{,}19e-01_{3{,}0e-02}$	$7{,}17e-01_{3{,}2e-02}$	$8{,}23e-01_{2{,}0e-02}$	$1{,}00e+00_{5{,}4e-04}$	$9{,}67e-01_{5{,}1e-03}$	$1{,}00e+00_{0{,}0e+00}$
BB30015	$8{,}61e-01_{4{,}3e-02}$	$7{,}63e-01_{8{,}2e-02}$	$9{,}02e-01_{5{,}0e-02}$	$1{,}00e+00_{1{,}9e-03}$	$1{,}02e+00_{1{,}2e-01}$	$1{,}00e+00_{0{,}0e+00}$
BB30017	$8{,}08e-01_{3{,}5e-02}$	$7{,}14e-01_{7{,}0e-02}$	$8{,}70e-01_{5{,}2e-02}$	$1{,}00e+00_{2{,}9e-03}$	$9{,}21e-01_{4{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB30018	$7{,}77e-01_{3{,}9e-02}$	$5{,}94e-01_{6{,}9e-02}$	$8{,}01e-01_{4{,}9e-02}$	$9{,}19e-01_{2{,}0e-01}$	$9{,}88e-01_{2{,}5e-02}$	$9{,}19e-01_{2{,}0e-01}$
BB30019	$7{,}51e-01_{3{,}9e-02}$	$6{,}76e-01_{8{,}4e-02}$	$8{,}11e-01_{3{,}8e-02}$	$1{,}00e+00_{5{,}6e-04}$	$9{,}73e-01_{2{,}0e-02}$	$1{,}00e+00_{2{,}6e-02}$
BB30021	$6{,}29e-01_{6{,}1e-02}$	$4{,}86e-01_{8{,}4e-02}$	$7{,}58e-01_{3{,}8e-02}$	$8{,}70e-01_{2{,}2e-01}$	$9{,}89e-01_{3{,}5e-03}$	$8{,}70e-01_{2{,}2e-01}$
BB30022	$8{,}08e-01_{3{,}0e-02}$	$7{,}11e-01_{7{,}4e-02}$	$8{,}37e-01_{4{,}1e-02}$	$1{,}00e+00_{4{,}4e-04}$	$9{,}77e-01_{8{,}3e-03}$	$1{,}00e+00_{2{,}4e-02}$
BB30024	$6{,}55e-01_{6{,}2e-02}$	$5{,}17e-01_{1{,}0e-01}$	$6{,}30e-01_{7{,}8e-02}$	$8{,}16e-01_{3{,}6e-01}$	$9{,}90e-01_{1{,}7e-02}$	$8{,}16e-01_{3{,}6e-01}$
BB30025	$8{,}44e-01_{5{,}2e-02}$	$9{,}85e-01_{8{,}9e-02}$	$9{,}11e-01_{6{,}7e-02}$	$9{,}85e-01_{8{,}9e-02}$	$1{,}01e+00_{1{,}1e-01}$	$9{,}85e-01_{8{,}9e-02}$
BB30026	$6{,}38e-01_{6{,}1e-02}$	$7{,}55e-01_{2{,}8e-01}$	$7{,}14e-01_{3{,}9e-02}$	$7{,}55e-01_{2{,}8e-01}$	$9{,}89e-01_{2{,}1e-02}$	$7{,}55e-01_{2{,}8e-01}$
BB30027	$8{,}58e-01_{5{,}1e-02}$	$8{,}58e-01_{5{,}1e-02}$	$8{,}50e-01_{1{,}4e-01}$	$1{,}02e+00_{1{,}2e-02}$	$8{,}76e-01_{1{,}7e-01}$	$1{,}00e+00_{0{,}0e+00}$
BB30028	$6{,}95e-01_{9{,}7e-02}$	$1{,}00e+00_{2{,}2e-02}$	$8{,}05e-01_{6{,}2e-02}$	$8{,}40e-01_{2{,}0e-01}$	$1{,}00e+00_{2{,}2e-02}$	$8{,}40e-01_{2{,}0e-01}$
BB30029	$7{,}02e-01_{5{,}3e-02}$	$7{,}36e-01_{2{,}4e-02}$	$7{,}36e-01_{2{,}4e-02}$	$7{,}98e-01_{2{,}5e-01}$	$9{,}91e-01_{9{,}0e-03}$	$7{,}98e-01_{2{,}5e-01}$
BB30030	$5{,}68e-01_{5{,}7e-02}$	$6{,}37e-01_{1{,}3e-01}$	$7{,}13e-01_{8{,}8e-02}$	$6{,}37e-01_{1{,}3e-01}$	$6{,}37e-01_{1{,}3e-01}$	$6{,}37e-01_{1{,}3e-01}$

Tabla B.17: Mediana y rango intercuartílico del indicador I_Δ para los problemas de la familia RV40 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB40001	$5{,}99e-01_{4{,}8e-02}$	$4{,}11e-01_{9{,}5e-02}$	$6{,}68e-01_{4{,}5e-02}$	$1{,}00e+00_{3{,}4e-04}$	$9{,}75e-01_{1{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40002	$4{,}07e-01_{6{,}0e-02}$	$4{,}36e-01_{2{,}9e-01}$	$4{,}36e-01_{2{,}9e-01}$	$1{,}13e+00_{5{,}7e-02}$	$1{,}13e+00_{5{,}7e-02}$	$1{,}13e+00_{5{,}7e-02}$
BB40003	$7{,}36e-01_{5{,}0e-02}$	$6{,}17e-01_{3{,}3e-02}$	$6{,}95e-01_{3{,}0e-02}$	$1{,}18e+00_{2{,}0e-01}$	$7{,}02e-01_{2{,}5e-02}$	$1{,}53e+00_{5{,}3e-02}$
BB40004	$8{,}27e-01_{3{,}0e-02}$	$7{,}52e-01_{2{,}1e-02}$	$8{,}44e-01_{2{,}5e-02}$	$1{,}00e+00_{7{,}6e-03}$	$9{,}97e-01_{1{,}2e-02}$	$1{,}00e+00_{7{,}6e-03}$
BB40005	$8{,}14e-01_{2{,}5e-02}$	$7{,}20e-01_{3{,}2e-02}$	$8{,}20e-01_{2{,}0e-02}$	$1{,}09e+00_{3{,}2e-02}$	$7{,}97e-01_{2{,}2e-02}$	$1{,}37e+00_{1{,}2e-01}$
BB40006	$8{,}41e-01_{2{,}4e-02}$	$7{,}56e-01_{2{,}0e-02}$	$8{,}23e-01_{1{,}8e-02}$	$1{,}02e+00_{2{,}7e-02}$	$8{,}52e-01_{2{,}7e-02}$	$1{,}03e+00_{3{,}8e-02}$
BB40007	$6{,}86e-01_{4{,}9e-02}$	$4{,}49e-01_{7{,}2e-02}$	$6{,}33e-01_{3{,}3e-02}$	$1{,}02e+00_{7{,}3e-02}$	$7{,}81e-01_{4{,}1e-02}$	$1{,}00e+00_{2{,}8e-02}$
BB40008	$8{,}24e-01_{2{,}1e-02}$	$7{,}25e-01_{3{,}7e-02}$	$8{,}16e-01_{2{,}6e-02}$	$1{,}04e+00_{4{,}0e-02}$	$8{,}51e-01_{5{,}3e-02}$	$1{,}40e+00_{4{,}9e-01}$
BB40009	$8{,}48e-01_{1{,}0e-02}$	$7{,}50e-01_{4{,}6e-02}$	$8{,}45e-01_{1{,}2e-02}$	$1{,}00e+00_{1{,}1e-03}$	$9{,}51e-01_{1{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40010	$8{,}25e-01_{7{,}3e-02}$	$5{,}90e-01_{8{,}5e-02}$	$7{,}94e-01_{1{,}1e-01}$	$1{,}46e+00_{1{,}4e-01}$	$7{,}55e-01_{1{,}2e-01}$	$1{,}62e+00_{1{,}3e-01}$
BB40011	$6{,}66e-01_{3{,}5e-02}$	$5{,}66e-01_{7{,}2e-02}$	$7{,}06e-01_{5{,}6e-02}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{8{,}0e-02}$	$1{,}00e+00_{5{,}3e-07}$
BB40012	$8{,}30e-01_{1{,}9e-02}$	$7{,}60e-01_{5{,}7e-02}$	$8{,}39e-01_{2{,}5e-02}$	$1{,}00e+00_{1{,}1e-03}$	$9{,}44e-01_{2{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40013	$8{,}06e-01_{5{,}9e-02}$	$7{,}38e-01_{1{,}4e-01}$	$8{,}15e-01_{9{,}1e-02}$	$1{,}00e+00_{5{,}1e-04}$	$9{,}73e-01_{1{,}8e-01}$	$1{,}00e+00_{0{,}0e+00}$
BB40014	$6{,}43e-01_{4{,}9e-02}$	$4{,}78e-01_{4{,}3e-02}$	$6{,}52e-01_{4{,}2e-02}$	$1{,}34e+00_{6{,}1e-02}$	$5{,}58e-01_{4{,}3e-02}$	$1{,}66e+00_{1{,}8e-01}$
BB40015	$5{,}90e-01_{5{,}0e-02}$	$3{,}27e-01_{1{,}5e-01}$	$5{,}37e-01_{1{,}9e-01}$	$1{,}00e+00_{1{,}0e-04}$	$9{,}98e-01_{9{,}2e-02}$	$1{,}00e+00_{7{,}6e-05}$
BB40016	$1{,}00e+00_{5{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{3{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$
BB40017	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{2{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$
BB40018	$1{,}02e+00_{1{,}7e-01}$	$8{,}53e-01_{1{,}4e-02}$	$8{,}80e-01_{9{,}6e-02}$	$1{,}06e+00_{4{,}8e-02}$	$8{,}66e-01_{6{,}2e-03}$	$1{,}38e+00_{3{,}4e-01}$
BB40019	$7{,}25e-01_{8{,}0e-02}$	$7{,}82e-01_{1{,}7e-01}$	$9{,}91e-01_{1{,}4e-01}$	$1{,}46e+00_{2{,}9e-02}$	$9{,}49e-01_{2{,}2e-01}$	$1{,}67e+00_{9{,}7e-02}$
BB40020	$8{,}46e-01_{2{,}5e-02}$	$7{,}88e-01_{4{,}2e-02}$	$8{,}42e-01_{1{,}5e-02}$	$1{,}00e+00_{1{,}3e-03}$	$9{,}47e-01_{2{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40021	$7{,}06e-01_{3{,}7e-02}$	$5{,}89e-01_{3{,}4e-02}$	$7{,}51e-01_{2{,}1e-02}$	$1{,}01e+00_{4{,}6e-03}$	$9{,}54e-01_{1{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40022	$6{,}22e-01_{5{,}3e-02}$	$4{,}30e-01_{1{,}1e-01}$	$6{,}23e-01_{2{,}9e-02}$	$1{,}01e+00_{1{,}1e-02}$	$8{,}56e-01_{5{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40023	$8{,}88e-01_{6{,}5e-02}$	$8{,}81e-01_{1{,}7e-01}$	$8{,}67e-01_{7{,}3e-02}$	$1{,}00e+00_{3{,}7e-02}$	$1{,}01e+00_{6{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40024	$7{,}77e-01_{4{,}0e-02}$	$7{,}17e-01_{4{,}3e-02}$	$8{,}25e-01_{4{,}4e-02}$	$1{,}00e+00_{5{,}0e-05}$	$9{,}89e-01_{3{,}9e-03}$	$1{,}00e+00_{9{,}0e-03}$
BB40025	$7{,}61e-01_{3{,}8e-02}$	$5{,}86e-01_{9{,}6e-02}$	$7{,}70e-01_{4{,}8e-02}$	$1{,}09e+00_{2{,}0e-02}$	$7{,}67e-01_{4{,}9e-02}$	$1{,}53e+00_{1{,}4e-01}$
BB40026	$7{,}56e-01_{2{,}0e-02}$	$6{,}65e-01_{3{,}1e-02}$	$8{,}07e-01_{1{,}2e-02}$	$1{,}00e+00_{9{,}3e-05}$	$9{,}89e-01_{8{,}0e-03}$	$1{,}00e+00_{0{,}0e+00}$
BB40027	$8{,}18e-01_{3{,}6e-02}$	$8{,}56e-01_{9{,}5e-02}$	$7{,}75e-01_{6{,}4e-02}$	$1{,}00e+00_{1{,}7e-06}$	$9{,}87e-01_{1{,}8e-02}$	$1{,}00e+00_{5{,}8e-03}$
BB40028	$8{,}39e-01_{2{,}2e-02}$	$7{,}82e-01_{9{,}2e-02}$	$8{,}38e-01_{1{,}3e-02}$	$1{,}01e+00_{6{,}7e-03}$	$9{,}29e-01_{1{,}8e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40029	$7{,}90e-01_{5{,}3e-02}$	$7{,}46e-01_{1{,}2e-01}$	$8{,}48e-01_{6{,}1e-02}$	$1{,}01e+00_{2{,}8e-03}$	$9{,}28e-01_{8{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40030	$8{,}00e-01_{2{,}4e-02}$	$7{,}01e-01_{2{,}1e-02}$	$8{,}23e-01_{1{,}7e-02}$	$1{,}00e+00_{1{,}0e-03}$	$9{,}76e-01_{9{,}8e-03}$	$1{,}00e+00_{0{,}0e+00}$
BB40031	$6{,}67e-01_{4{,}0e-02}$	$5{,}33e-01_{2{,}2e-02}$	$7{,}58e-01_{2{,}3e-02}$	$1{,}00e+00_{2{,}5e-03}$	$9{,}75e-01_{1{,}0e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40032	$5{,}69e-01_{5{,}0e-02}$	$3{,}07e-01_{9{,}3e-02}$	$5{,}26e-01_{3{,}9e-02}$	$1{,}29e+00_{2{,}5e-02}$	$4{,}37e-01_{2{,}5e-02}$	$1{,}71e+00_{5{,}2e-02}$
BB40033	$8{,}08e-01_{1{,}7e-02}$	$7{,}06e-01_{2{,}6e-02}$	$8{,}17e-01_{2{,}7e-02}$	$1{,}00e+00_{2{,}8e-03}$	$9{,}53e-01_{2{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40034	$7{,}68e-01_{2{,}1e-02}$	$6{,}28e-01_{6{,}4e-02}$	$7{,}94e-01_{2{,}1e-02}$	$1{,}00e+00_{4{,}7e-04}$	$9{,}78e-01_{3{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40035	$6{,}61e-01_{4{,}0e-02}$	$5{,}55e-01_{8{,}5e-02}$	$7{,}21e-01_{4{,}5e-02}$	$1{,}00e+00_{3{,}0e-04}$	$9{,}77e-01_{1{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40036	$8{,}14e-01_{2{,}7e-02}$	$7{,}57e-01_{2{,}0e-02}$	$8{,}30e-01_{2{,}5e-02}$	$1{,}00e+00_{2{,}7e-03}$	$9{,}57e-01_{2{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40037	$7{,}76e-01_{4{,}6e-02}$	$7{,}01e-01_{1{,}4e-01}$	$8{,}14e-01_{1{,}0e-01}$	$1{,}00e+00_{2{,}5e-04}$	$1{,}02e+00_{1{,}5e-01}$	$1{,}00e+00_{0{,}0e+00}$
BB40038	$8{,}96e-01_{1{,}1e-01}$	$9{,}97e-01_{1{,}0e-01}$	$9{,}77e-01_{4{,}8e-02}$	$1{,}00e+00_{3{,}7e-04}$	$1{,}18e+00_{7{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40039	$8{,}87e-01_{1{,}3e-02}$	$8{,}62e-01_{2{,}2e-02}$	$8{,}99e-01_{2{,}6e-02}$	$1{,}00e+00_{2{,}4e-04}$	$1{,}04e+00_{3{,}0e-02}$	$1{,}00e+00_{3{,}6e-02}$
BB40040	$8{,}13e-01_{2{,}8e-02}$	$8{,}22e-01_{7{,}3e-02}$	$7{,}78e-01_{1{,}9e-02}$	$1{,}00e+00_{5{,}7e-04}$	$1{,}04e+00_{1{,}1e-01}$	$1{,}00e+00_{0{,}0e+00}$
BB40041	$6{,}65e-01_{7{,}9e-02}$	$5{,}86e-01_{9{,}6e-02}$	$7{,}60e-01_{2{,}8e-01}$	$1{,}00e+00_{5{,}4e-05}$	$9{,}96e-01_{3{,}2e-02}$	$1{,}00e+00_{1{,}6e-03}$
BB40042	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{3{,}4e-06}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$	$1{,}00e+00_{0{,}0e+00}$
BB40043	$7{,}99e-01_{3{,}3e-02}$	$7{,}03e-01_{3{,}9e-02}$	$8{,}09e-01_{6{,}3e-02}$	$1{,}00e+00_{2{,}3e-03}$	$8{,}86e-01_{1{,}3e-01}$	$1{,}00e+00_{0{,}0e+00}$
BB40044	$7{,}43e-01_{5{,}5e-02}$	$6{,}85e-01_{2{,}9e-02}$	$8{,}61e-01_{6{,}5e-02}$	$1{,}00e+00_{0{,}0e+00}$	$9{,}90e-01_{1{,}1e-01}$	$1{,}00e+00_{0{,}0e+00}$
BB40045	$9{,}30e-01_{2{,}6e-02}$	$9{,}11e-01_{1{,}0e-01}$	$9{,}33e-01_{1{,}1e-01}$	$1{,}28e+00_{5{,}9e-02}$	$9{,}40e-01_{1{,}0e-01}$	$1{,}15e+00_{6{,}0e-02}$
BB40046	$7{,}95e-01_{3{,}8e-02}$	$7{,}10e-01_{5{,}0e-02}$	$8{,}15e-01_{6{,}6e-02}$	$1{,}00e+00_{1{,}3e-04}$	$9{,}85e-01_{1{,}6e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB40047	$6{,}52e-01_{4{,}2e-02}$	$4{,}85e-01_{7{,}6e-02}$	$4{,}85e-01_{7{,}6e-02}$	$4{,}85e-01_{7{,}6e-02}$	$6{,}35e-01_{5{,}1e-01}$	$4{,}85e-01_{1{,}2e-01}$
BB40048	$8{,}38e-01_{1{,}9e-02}$	$7{,}26e-01_{8{,}2e-02}$	$8{,}29e-01_{1{,}7e-02}$	$8{,}29e-01_{1{,}7e-02}$	$8{,}97e-01_{3{,}1e-02}$	$1{,}00e+00_{8{,}6e-02}$
BB40049	$1{,}93e-01_{0{,}0e+00}$	$1{,}93e-01_{0{,}0e+00}$	$1{,}93e-01_{0{,}0e+00}$	$1{,}93e-01_{0{,}0e+00}$	$1{,}93e-01_{0{,}0e+00}$	$1{,}93e-01_{0{,}0e+00}$

Tabla B.18: Mediana y rango intercuartílico del indicador I_Δ para los problemas de la familia RV50 del BAliBASE 3.0 y para los algoritmos: NSGAII, MOCell, SPEA2, MOEADMSA, SMS-EMOA y GWASFGA.

	NSGAII	MOCell	SPEA2	MOEADMSA	SMS-EMOA	GWASFGA
BB50001	$8{,}35e-01_{4{,}2e-02}$	$7{,}30e-01_{5{,}6e-02}$	$8{,}54e-01_{1{,}2e-02}$	$1{,}00e+00_{1{,}9e-04}$	$9{,}86e-01_{3{,}3e-02}$	$1{,}00e+00_{1{,}2e-02}$
BB50002	$6{,}69e-01_{4{,}8e-02}$	$5{,}70e-01_{7{,}3e-02}$	$6{,}89e-01_{3{,}7e-02}$	$1{,}00e+00_{1{,}3e-03}$	$8{,}98e-01_{4{,}2e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB50003	$7{,}39e-01_{5{,}5e-02}$	$5{,}19e-01_{7{,}5e-02}$	$8{,}04e-01_{2{,}6e-02}$	$9{,}92e-01_{1{,}1e-02}$	$9{,}89e-01_{4{,}0e-03}$	$9{,}92e-01_{1{,}1e-02}$
BB50004	$1{,}13e+00_{5{,}8e-02}$	$6{,}83e-01_{8{,}8e-02}$	$1{,}15e+00_{5{,}3e-02}$	$1{,}45e+00_{1{,}5e-01}$	$1{,}14e+00_{3{,}8e-02}$	$1{,}54e+00_{1{,}9e-01}$
BB50005	$6{,}14e-01_{5{,}0e-02}$	$5{,}10e-01_{6{,}0e-02}$	$6{,}14e-01_{4{,}3e-02}$	$1{,}47e+00_{3{,}0e-02}$	$5{,}49e-01_{5{,}9e-02}$	$1{,}70e+00_{4{,}4e-02}$
BB50006	$5{,}94e-01_{4{,}6e-02}$	$4{,}87e-01_{2{,}8e-02}$	$6{,}12e-01_{4{,}2e-02}$	$4{,}87e-01_{2{,}8e-02}$	$5{,}15e-01_{5{,}4e-01}$	$4{,}87e-01_{2{,}8e-02}$
BB50007	$7{,}74e-01_{2{,}9e-02}$	$6{,}74e-01_{3{,}8e-02}$	$8{,}04e-01_{2{,}6e-02}$	$1{,}00e+00_{1{,}6e-03}$	$9{,}52e-01_{1{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB50008	$8{,}46e-01_{2{,}5e-02}$	$7{,}61e-01_{3{,}8e-02}$	$8{,}42e-01_{2{,}1e-02}$	$1{,}01e+00_{4{,}3e-03}$	$9{,}45e-01_{1{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB50009	$8{,}23e-01_{2{,}4e-02}$	$7{,}51e-01_{4{,}1e-02}$	$8{,}50e-01_{2{,}4e-02}$	$1{,}00e+00_{7{,}2e-04}$	$9{,}79e-01_{3{,}1e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB50010	$8{,}23e-01_{4{,}8e-02}$	$8{,}23e-01_{5{,}7e-02}$	$8{,}89e-01_{5{,}6e-02}$	$1{,}00e+00_{1{,}7e-03}$	$1{,}09e+00_{3{,}7e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB50011	$7{,}44e-01_{2{,}1e-02}$	$6{,}46e-01_{4{,}4e-02}$	$7{,}95e-01_{5{,}4e-02}$	$1{,}00e+00_{3{,}5e-04}$	$9{,}93e-01_{5{,}9e-02}$	$1{,}00e+00_{1{,}6e-02}$
BB50012	$5{,}28e-01_{4{,}9e-02}$	$3{,}21e-01_{4{,}1e-02}$	$9{,}89e-01_{4{,}1e-03}$	$9{,}89e-01_{4{,}1e-03}$	$9{,}89e-01_{4{,}1e-03}$	$9{,}89e-01_{4{,}1e-03}$
BB50013	$8{,}21e-01_{3{,}1e-02}$	$6{,}67e-01_{6{,}6e-02}$	$6{,}80e-01_{7{,}7e-02}$	$1{,}25e+00_{5{,}3e-02}$	$7{,}23e-01_{7{,}5e-02}$	$1{,}66e+00_{1{,}3e-01}$
BB50014	$8{,}13e-01_{1{,}6e-02}$	$7{,}30e-01_{3{,}5e-02}$	$8{,}14e-01_{1{,}8e-02}$	$1{,}00e+00_{6{,}6e-04}$	$9{,}81e-01_{1{,}3e-02}$	$1{,}00e+00_{0{,}0e+00}$
BB50015	$7{,}19e-01_{3{,}8e-02}$	$6{,}56e-01_{6{,}3e-02}$	$7{,}73e-01_{4{,}3e-02}$	$1{,}00e+00_{1{,}9e-04}$	$1{,}00e+00_{5{,}2e-02}$	$1{,}00e+00_{1{,}2e-02}$
BB50016	$6{,}56e-01_{3{,}9e-02}$	$4{,}92e-01_{1{,}8e-02}$	$6{,}74e-01_{2{,}7e-02}$	$1{,}00e+00_{1{,}1e-03}$	$9{,}37e-01_{2{,}4e-02}$	$1{,}00e+00_{0{,}0e+00}$

Tabla B.19: Mediana y rango intercuartílico del indicador I_{e+} para la familia RV11 del BAliBASE 3.0 y para los algoritmos: GWASF-GA, NSGA-II, MOCell, y NSGA-III

	GWASF-GA	NSGA-II	MOCell	NSGA-III
BB11001	$5{,}00e-01_{1{,}0e-01}$	$2{,}28e-01_{8{,}6e-02}$	$3{,}93e-01_{1{,}9e-01}$	$3{,}00e-01_{1{,}0e-01}$
BB11002	$7{,}22e-01_{3{,}0e-01}$	$3{,}57e-01_{6{,}7e-02}$	$6{,}96e-01_{1{,}8e-01}$	$3{,}90e-01_{9{,}3e-02}$
BB11003	$6{,}01e-01_{8{,}3e-02}$	$1{,}80e-01_{2{,}5e-02}$	$3{,}41e-01_{1{,}8e-01}$	$3{,}61e-01_{8{,}9e-02}$
BB11004	$7{,}93e-01_{2{,}4e-02}$	$2{,}16e-01_{4{,}7e-02}$	$3{,}57e-01_{1{,}6e-01}$	$3{,}28e-01_{6{,}5e-02}$
BB11005	$3{,}26e-01_{6{,}3e-03}$	$3{,}32e-01_{6{,}2e-03}$	$3{,}36e-01_{7{,}4e-03}$	$3{,}32e-01_{7{,}5e-03}$
BB11007	$4{,}39e-01_{9{,}5e-02}$	$1{,}75e-01_{1{,}8e-02}$	$3{,}97e-01_{1{,}7e-01}$	$1{,}97e-01_{3{,}7e-02}$
BB11009	$7{,}11e-01_{9{,}5e-02}$	$2{,}05e-01_{5{,}9e-02}$	$3{,}45e-01_{4{,}5e-02}$	$3{,}75e-01_{9{,}0e-02}$
BB11010	$8{,}07e-01_{4{,}7e-02}$	$1{,}96e-01_{3{,}3e-02}$	$2{,}27e-01_{3{,}0e-02}$	$3{,}86e-01_{6{,}7e-02}$
BB11011	$6{,}01e-01_{1{,}2e-01}$	$2{,}19e-01_{5{,}3e-02}$	$2{,}47e-01_{5{,}2e-02}$	$3{,}52e-01_{1{,}3e-01}$
BB11012	$4{,}05e-01_{1{,}5e-01}$	$1{,}59e-01_{6{,}4e-02}$	$1{,}59e-01_{2{,}6e-02}$	$2{,}68e-01_{1{,}4e-01}$
BB11014	$3{,}48e-01_{1{,}2e-01}$	$1{,}35e-01_{2{,}3e-02}$	$1{,}54e-01_{2{,}1e-02}$	$2{,}00e-01_{5{,}7e-02}$
BB11015	$4{,}67e-01_{9{,}9e-02}$	$1{,}47e-01_{2{,}5e-02}$	$1{,}60e-01_{2{,}4e-02}$	$3{,}34e-01_{8{,}8e-02}$
BB11016	$7{,}87e-01_{5{,}2e-02}$	$2{,}67e-01_{2{,}9e-02}$	$2{,}79e-01_{3{,}7e-02}$	$2{,}72e-01_{5{,}3e-02}$
BB11017	$4{,}88e-01_{6{,}8e-02}$	$1{,}46e-01_{5{,}3e-02}$	$1{,}56e-01_{4{,}2e-02}$	$1{,}86e-01_{5{,}9e-02}$
BB11018	$5{,}36e-01_{1{,}1e-01}$	$2{,}29e-01_{2{,}9e-02}$	$2{,}44e-01_{4{,}3e-02}$	$2{,}34e-01_{4{,}0e-02}$
BB11019	$4{,}65e-01_{1{,}8e-01}$	$1{,}45e-01_{9{,}2e-02}$	$1{,}52e-01_{8{,}8e-02}$	$2{,}24e-01_{4{,}0e-02}$
BB11020	$6{,}09e-01_{1{,}0e-01}$	$1{,}76e-01_{5{,}5e-02}$	$1{,}95e-01_{4{,}6e-02}$	$2{,}29e-01_{4{,}3e-02}$
BB11021	$6{,}60e-01_{3{,}0e-02}$	$2{,}09e-01_{8{,}5e-02}$	$2{,}08e-01_{2{,}3e-02}$	$3{,}00e-01_{6{,}0e-02}$
BB11023	$4{,}50e-01_{1{,}1e-01}$	$2{,}32e-01_{1{,}1e-01}$	$2{,}83e-01_{5{,}4e-02}$	$4{,}06e-01_{9{,}7e-02}$
BB11025	$5{,}71e-01_{7{,}7e-02}$	$2{,}07e-01_{5{,}2e-02}$	$2{,}96e-01_{8{,}4e-02}$	$3{,}22e-01_{1{,}1e-01}$
BB11026	$1{,}94e+00_{2{,}1e+00}$	$1{,}01e+00_{3{,}5e-01}$	$1{,}00e+00_{1{,}1e-01}$	$1{,}09e+00_{3{,}9e-01}$
BB11027	$4{,}04e-01_{6{,}6e-02}$	$4{,}00e-01_{8{,}6e-02}$	$4{,}02e-01_{2{,}0e-02}$	$4{,}12e-01_{5{,}3e-02}$
BB11028	$4{,}42e-01_{1{,}7e-01}$	$4{,}87e-01_{3{,}0e-01}$	$4{,}85e-01_{2{,}6e-01}$	$4{,}87e-01_{1{,}8e-01}$
BB11029	$5{,}02e-01_{8{,}8e-02}$	$2{,}34e-01_{5{,}0e-02}$	$2{,}64e-01_{5{,}1e-02}$	$3{,}38e-01_{8{,}3e-02}$
BB11030	$6{,}61e-01_{2{,}4e-01}$	$4{,}93e-01_{2{,}9e-01}$	$4{,}32e-01_{6{,}5e-02}$	$5{,}06e-01_{1{,}3e-01}$
BB11031	$6{,}72e-01_{1{,}5e-01}$	$2{,}70e-01_{9{,}5e-02}$	$2{,}89e-01_{8{,}0e-02}$	$2{,}91e-01_{5{,}1e-02}$
BB11032	$7{,}07e-01_{2{,}0e-02}$	$2{,}47e-01_{5{,}6e-02}$	$3{,}12e-01_{8{,}7e-02}$	$2{,}98e-01_{6{,}9e-02}$
BB11033	$3{,}34e-01_{1{,}6e-02}$	$3{,}24e-01_{1{,}9e-02}$	$3{,}32e-01_{3{,}5e-02}$	$3{,}29e-01_{7{,}4e-03}$
BB11034	$6{,}49e-01_{2{,}3e-01}$	$1{,}86e-01_{2{,}3e-01}$	$2{,}10e-01_{1{,}3e-01}$	$2{,}67e-01_{9{,}6e-02}$
BB11035	$3{,}63e-01_{5{,}0e-02}$	$2{,}10e-01_{4{,}9e-01}$	$2{,}72e-01_{8{,}9e-02}$	$3{,}15e-01_{1{,}0e-01}$
BB11036	$6{,}30e-01_{1{,}5e-01}$	$2{,}13e-01_{1{,}8e-01}$	$2{,}34e-01_{9{,}2e-02}$	$2{,}86e-01_{9{,}0e-02}$
BB11038	$5{,}56e-01_{2{,}7e-02}$	$1{,}71e-01_{1{,}7e-01}$	$2{,}20e-01_{5{,}4e-02}$	$2{,}37e-01_{6{,}3e-02}$

B.2. Comparativa triobjetivo - indicadores de calidad

Tabla B.20: Mediana y rango intercuartílico del indicador I_{IGD+} para la familia RV11 del BAliBASE 3.0 y para los algoritmos: GWASF-GA, NSGA-II, MOCell, y NSGA-III.

	GWASF-GA	NSGA-II	MOCell	NSGA-III
BB11001	$1{,}77e-01_{3{,}4e-02}$	$6{,}51e-02_{2{,}8e-02}$	$2{,}66e-01_{1{,}7e-01}$	$1{,}59e-01_{8{,}6e-02}$
BB11002	$4{,}42e-01_{1{,}2e-01}$	$2{,}43e-01_{9{,}3e-02}$	$6{,}42e-01_{2{,}0e-01}$	$2{,}90e-01_{1{,}2e-01}$
BB11003	$2{,}02e-01_{4{,}0e-02}$	$9{,}40e-02_{2{,}1e-02}$	$2{,}10e-01_{1{,}0e-01}$	$2{,}00e-01_{6{,}1e-02}$
BB11004	$3{,}08e-01_{1{,}5e-02}$	$1{,}13e-01_{2{,}4e-02}$	$2{,}41e-01_{1{,}5e-01}$	$1{,}48e-01_{2{,}7e-02}$
BB11005	$1{,}36e-01_{1{,}4e-02}$	$1{,}13e-01_{1{,}7e-02}$	$2{,}23e-01_{8{,}4e-02}$	$1{,}34e-01_{1{,}1e-02}$
BB11007	$2{,}10e-01_{4{,}4e-02}$	$8{,}92e-02_{2{,}9e-02}$	$3{,}18e-01_{1{,}3e-01}$	$1{,}12e-01_{3{,}4e-02}$
BB11009	$3{,}50e-01_{8{,}3e-02}$	$1{,}05e-01_{3{,}5e-02}$	$2{,}13e-01_{3{,}7e-02}$	$1{,}90e-01_{3{,}6e-02}$
BB11010	$2{,}85e-01_{3{,}7e-02}$	$1{,}01e-01_{1{,}5e-02}$	$1{,}18e-01_{2{,}3e-02}$	$1{,}69e-01_{4{,}9e-02}$
BB11011	$2{,}31e-01_{2{,}9e-02}$	$1{,}28e-01_{3{,}4e-02}$	$1{,}92e-01_{4{,}4e-02}$	$2{,}07e-01_{5{,}9e-02}$
BB11012	$1{,}32e-01_{7{,}7e-02}$	$6{,}18e-02_{2{,}3e-02}$	$8{,}16e-02_{2{,}6e-02}$	$1{,}25e-01_{5{,}4e-02}$
BB11014	$1{,}34e-01_{4{,}5e-02}$	$7{,}07e-02_{2{,}4e-02}$	$1{,}08e-01_{2{,}9e-02}$	$1{,}21e-01_{3{,}6e-02}$
BB11015	$1{,}96e-01_{3{,}8e-02}$	$7{,}95e-02_{2{,}6e-02}$	$9{,}50e-02_{2{,}7e-02}$	$1{,}95e-01_{3{,}0e-02}$
BB11016	$3{,}70e-01_{3{,}9e-02}$	$1{,}50e-01_{3{,}4e-02}$	$2{,}12e-01_{3{,}9e-02}$	$1{,}57e-01_{3{,}7e-02}$
BB11017	$1{,}78e-01_{5{,}1e-02}$	$6{,}63e-02_{2{,}2e-02}$	$8{,}74e-02_{2{,}7e-02}$	$9{,}44e-02_{3{,}7e-02}$
BB11018	$1{,}87e-01_{7{,}6e-02}$	$8{,}59e-02_{1{,}9e-02}$	$1{,}19e-01_{2{,}0e-02}$	$9{,}53e-02_{2{,}2e-02}$
BB11019	$1{,}82e-01_{6{,}7e-02}$	$7{,}81e-02_{3{,}6e-02}$	$1{,}10e-01_{3{,}5e-02}$	$1{,}08e-01_{3{,}7e-02}$
BB11020	$2{,}95e-01_{7{,}2e-02}$	$9{,}95e-02_{2{,}1e-02}$	$1{,}39e-01_{3{,}2e-02}$	$1{,}13e-01_{3{,}5e-02}$
BB11021	$2{,}30e-01_{2{,}3e-02}$	$7{,}97e-02_{2{,}1e-02}$	$1{,}07e-01_{2{,}2e-02}$	$1{,}43e-01_{3{,}7e-02}$
BB11023	$2{,}15e-01_{4{,}3e-02}$	$1{,}41e-01_{4{,}7e-02}$	$1{,}94e-01_{5{,}7e-02}$	$2{,}04e-01_{6{,}0e-02}$
BB11025	$2{,}66e-01_{5{,}9e-02}$	$1{,}40e-01_{7{,}5e-02}$	$2{,}52e-01_{1{,}3e-01}$	$2{,}03e-01_{1{,}5e-01}$
BB11026	$1{,}92e+00_{1{,}9e+00}$	$9{,}35e-01_{3{,}9e-01}$	$8{,}59e-01_{3{,}1e-01}$	$9{,}81e-01_{3{,}7e-01}$
BB11027	$2{,}68e-01_{1{,}4e-01}$	$2{,}86e-01_{1{,}4e-01}$	$3{,}29e-01_{9{,}8e-02}$	$2{,}87e-01_{1{,}1e-01}$
BB11028	$1{,}57e-01_{1{,}3e-01}$	$2{,}26e-01_{1{,}4e-01}$	$2{,}45e-01_{1{,}0e-01}$	$2{,}58e-01_{1{,}0e-01}$
BB11029	$2{,}26e-01_{7{,}9e-02}$	$1{,}62e-01_{4{,}9e-02}$	$1{,}87e-01_{6{,}0e-02}$	$2{,}05e-01_{5{,}3e-02}$
BB11030	$3{,}49e-01_{1{,}9e-01}$	$1{,}70e-01_{9{,}2e-02}$	$1{,}75e-01_{3{,}2e-02}$	$2{,}87e-01_{1{,}1e-01}$
BB11031	$3{,}21e-01_{7{,}8e-02}$	$1{,}08e-01_{3{,}7e-02}$	$1{,}62e-01_{2{,}0e-02}$	$1{,}50e-01_{4{,}3e-02}$
BB11032	$2{,}35e-01_{2{,}3e-02}$	$1{,}04e-01_{2{,}5e-02}$	$1{,}41e-01_{2{,}2e-02}$	$1{,}79e-01_{6{,}2e-02}$
BB11033	$2{,}02e-01_{7{,}9e-02}$	$1{,}51e-01_{4{,}2e-02}$	$2{,}16e-01_{1{,}2e-01}$	$1{,}80e-01_{6{,}1e-02}$
BB11034	$2{,}81e-01_{2{,}0e-01}$	$1{,}05e-01_{2{,}3e-01}$	$1{,}73e-01_{1{,}5e-01}$	$1{,}71e-01_{7{,}2e-02}$
BB11035	$1{,}66e-01_{4{,}5e-02}$	$1{,}30e-01_{3{,}7e-01}$	$1{,}85e-01_{9{,}5e-02}$	$2{,}15e-01_{6{,}9e-02}$
BB11036	$2{,}52e-01_{6{,}3e-02}$	$1{,}40e-01_{2{,}1e-01}$	$1{,}77e-01_{1{,}2e-01}$	$1{,}71e-01_{4{,}1e-02}$
BB11038	$1{,}90e-01_{3{,}3e-02}$	$9{,}85e-02_{1{,}7e-01}$	$1{,}18e-01_{2{,}5e-02}$	$1{,}24e-01_{3{,}6e-02}$

Tabla B.21: Mediana y rango intercuartílico del indicador I_{e+} para la familia RV12 del BAliBASE 3.0 y para los algoritmos: GWASF-GA, NSGA-II, MOCell, y NSGA-III.

	GWASF-GA	NSGA-II	MOCell	NSGA-III
BB12001	$7{,}48e-01_{8,1e-02}$	$1{,}61e-01_{8,3e-02}$	$1{,}61e-01_{8,3e-02}$	$2{,}04e-01_{7,3e-02}$
BB12003	$3{,}13e-01_{1,3e-01}$	$3{,}77e-01_{1,2e-01}$	$3{,}77e-01_{1,2e-01}$	$4{,}80e-01_{3,1e-01}$
BB12004	$5{,}27e-01_{2,9e-02}$	$2{,}39e-01_{1,2e-01}$	$2{,}39e-01_{1,2e-01}$	$2{,}57e-01_{8,6e-02}$
BB12005	$4{,}35e-01_{7,1e-02}$	$1{,}58e-01_{4,3e-02}$	$1{,}57e-01_{2,9e-02}$	$1{,}63e-01_{4,9e-02}$
BB12007	$8{,}46e-01_{4,7e-02}$	$3{,}21e-01_{1,4e-01}$	$3{,}38e-01_{8,6e-02}$	$2{,}78e-01_{1,2e-01}$
BB12008	$3{,}93e-01_{4,2e-02}$	$2{,}33e-01_{1,0e-01}$	$2{,}24e-01_{9,2e-02}$	$2{,}20e-01_{5,6e-02}$
BB12009	$8{,}74e-01_{0,0e+00}$	$2{,}72e-01_{1,1e-01}$	$3{,}29e-01_{1,3e-01}$	$3{,}07e-01_{6,4e-02}$
BB12010	$6{,}28e-01_{2,0e-02}$	$1{,}57e-01_{6,8e-02}$	$1{,}50e-01_{4,7e-02}$	$2{,}05e-01_{7,6e-02}$
BB12011	$7{,}37e-01_{8,5e-02}$	$2{,}08e-01_{5,4e-02}$	$2{,}28e-01_{6,7e-02}$	$1{,}96e-01_{8,2e-02}$
BB12012	$5{,}89e-01_{2,4e-01}$	$1{,}60e-01_{4,5e-02}$	$2{,}18e-01_{6,2e-02}$	$3{,}62e-01_{1,5e-01}$
BB12013	$6{,}04e-01_{8,3e-02}$	$2{,}08e-01_{1,8e-01}$	$2{,}39e-01_{1,0e-01}$	$2{,}08e-01_{1,1e-01}$
BB12015	$7{,}05e-01_{6,4e-02}$	$2{,}49e-01_{1,6e-01}$	$2{,}56e-01_{1,2e-01}$	$3{,}60e-01_{5,4e-02}$
BB12016	$7{,}70e-01_{2,7e-01}$	$2{,}38e-01_{8,5e-02}$	$3{,}63e-01_{1,5e-01}$	$2{,}96e-01_{9,6e-02}$
BB12017	$7{,}32e-01_{4,0e-02}$	$1{,}45e-01_{8,0e-02}$	$1{,}56e-01_{3,6e-02}$	$1{,}77e-01_{6,7e-02}$
BB12018	$2{,}65e-01_{6,6e-02}$	$1{,}18e-01_{4,6e-02}$	$1{,}56e-01_{4,6e-02}$	$1{,}82e-01_{4,3e-02}$
BB12020	$5{,}84e-01_{6,5e-02}$	$1{,}59e-01_{7,7e-02}$	$1{,}40e-01_{3,9e-02}$	$1{,}60e-01_{6,8e-02}$
BB12021	$2{,}33e-01_{1,2e-01}$	$3{,}14e-01_{1,3e-01}$	$3{,}14e-01_{1,1e-01}$	$3{,}14e-01_{4,6e-02}$
BB12022	$3{,}13e-01_{1,5e-01}$	$2{,}36e-01_{0,0e+00}$	$2{,}36e-01_{0,0e+00}$	$2{,}36e-01_{6,2e-02}$
BB12023	$6{,}15e-01_{5,7e-02}$	$1{,}86e-01_{7,7e-02}$	$1{,}56e-01_{8,4e-02}$	$1{,}56e-01_{8,5e-02}$
BB12024	$4{,}48e-01_{5,6e-02}$	$9{,}15e-02_{4,7e-02}$	$1{,}14e-01_{3,0e-02}$	$1{,}56e-01_{4,1e-02}$
BB12025	$7{,}95e-01_{4,6e-03}$	$2{,}52e-01_{9,4e-02}$	$3{,}54e-01_{2,6e-02}$	$3{,}99e-01_{1,2e-01}$
BB12026	$3{,}29e-01_{8,2e-03}$	$2{,}53e-01_{2,9e-02}$	$2{,}78e-01_{9,1e-02}$	$2{,}59e-01_{4,0e-02}$
BB12027	$2{,}29e-01_{9,5e-02}$	$1{,}72e-01_{7,0e-02}$	$1{,}85e-01_{5,8e-02}$	$2{,}12e-01_{6,1e-02}$
BB12028	$5{,}76e-01_{5,0e-01}$	$1{,}23e-01_{2,0e-02}$	$1{,}47e-01_{5,9e-02}$	$1{,}75e-01_{6,5e-02}$
BB12031	$5{,}38e-01_{1,4e-01}$	$2{,}38e-01_{1,0e-01}$	$2{,}23e-01_{4,4e-02}$	$2{,}45e-01_{3,8e-02}$
BB12032	$5{,}83e-01_{3,3e-02}$	$1{,}58e-01_{3,8e-02}$	$2{,}30e-01_{6,0e-02}$	$1{,}98e-01_{8,1e-02}$
BB12033	$8{,}04e-01_{3,5e-02}$	$1{,}85e-01_{7,4e-02}$	$1{,}77e-01_{4,6e-02}$	$1{,}99e-01_{7,1e-02}$
BB12034	$6{,}83e-01_{5,5e-02}$	$1{,}88e-01_{9,3e-02}$	$1{,}58e-01_{6,2e-02}$	$1{,}88e-01_{4,1e-02}$
BB12035	$5{,}12e-01_{5,3e-02}$	$2{,}64e-01_{1,1e-01}$	$3{,}27e-01_{8,8e-02}$	$3{,}26e-01_{9,9e-02}$
BB12036	$4{,}29e-01_{1,2e-01}$	$1{,}87e-01_{5,9e-02}$	$1{,}87e-01_{6,2e-02}$	$1{,}87e-01_{9,0e-02}$
BB12037	$8{,}92e-01_{4,4e-02}$	$2{,}22e-01_{6,7e-02}$	$2{,}80e-01_{7,5e-02}$	$2{,}64e-01_{8,4e-02}$
BB12038	$7{,}41e-01_{1,6e-01}$	$2{,}47e-01_{5,0e-02}$	$3{,}07e-01_{8,9e-02}$	$3{,}79e-01_{1,6e-01}$
BB12039	$9{,}62e-01_{9,6e-03}$	$3{,}22e-01_{1,4e-01}$	$4{,}46e-01_{1,2e-01}$	$3{,}97e-01_{8,5e-02}$
BB12041	$3{,}73e-01_{2,1e-02}$	$2{,}07e-01_{4,6e-02}$	$2{,}27e-01_{3,9e-02}$	$2{,}80e-01_{8,8e-02}$
BB12042	$6{,}11e-01_{6,1e-02}$	$1{,}31e-01_{5,7e-02}$	$1{,}52e-01_{4,5e-02}$	$1{,}78e-01_{4,9e-02}$
BB12043	$6{,}34e-01_{4,5e-01}$	$3{,}80e-01_{1,4e-01}$	$4{,}47e-01_{1,2e-01}$	$4{,}57e-01_{1,1e-01}$
BB12044	$5{,}84e-01_{4,7e-02}$	$1{,}83e-01_{6,1e-02}$	$1{,}83e-01_{6,1e-02}$	$2{,}21e-01_{5,9e-02}$

Tabla B.22: Mediana y rango intercuartílico del indicador I_{IGD+} para la familia RV12 del BAliBASE 3.0 y para los algoritmos: GWASF-GA, NSGA-II, MOCell, y NSGA-III.

	GWASF-GA	NSGA-II	MOCell	NSGA-III
BB12001	$3{,}29e-01_{4{,}9e-02}$	$7{,}00e-02_{1{,}7e-02}$	$7{,}00e-02_{1{,}7e-02}$	$7{,}99e-02_{3{,}3e-02}$
BB12003	$2{,}26e-01_{1{,}2e-01}$	$2{,}80e-01_{1{,}3e-01}$	$2{,}80e-01_{1{,}3e-01}$	$3{,}69e-01_{2{,}2e-01}$
BB12004	$2{,}60e-01_{6{,}0e-02}$	$1{,}14e-01_{6{,}8e-02}$	$1{,}14e-01_{6{,}8e-02}$	$1{,}41e-01_{5{,}8e-02}$
BB12005	$1{,}67e-01_{3{,}5e-02}$	$7{,}31e-02_{2{,}8e-02}$	$1{,}15e-01_{2{,}9e-02}$	$8{,}24e-02_{3{,}6e-02}$
BB12007	$3{,}63e-01_{5{,}0e-02}$	$8{,}08e-02_{4{,}9e-02}$	$1{,}18e-01_{2{,}2e-02}$	$8{,}08e-02_{3{,}1e-02}$
BB12008	$1{,}35e-01_{3{,}1e-02}$	$1{,}00e-01_{2{,}9e-02}$	$1{,}13e-01_{4{,}1e-02}$	$1{,}11e-01_{3{,}0e-02}$
BB12009	$5{,}87e-01_{2{,}6e-02}$	$1{,}72e-01_{8{,}8e-02}$	$2{,}61e-01_{1{,}2e-01}$	$1{,}92e-01_{9{,}7e-02}$
BB12010	$2{,}21e-01_{2{,}4e-02}$	$5{,}08e-02_{1{,}6e-02}$	$7{,}15e-02_{3{,}3e-02}$	$9{,}16e-02_{2{,}4e-02}$
BB12011	$3{,}13e-01_{6{,}6e-02}$	$7{,}25e-02_{2{,}2e-02}$	$1{,}30e-01_{4{,}2e-02}$	$1{,}14e-01_{4{,}9e-02}$
BB12012	$2{,}54e-01_{1{,}1e-01}$	$8{,}64e-02_{2{,}3e-02}$	$1{,}05e-01_{2{,}0e-02}$	$1{,}55e-01_{6{,}6e-02}$
BB12013	$1{,}65e-01_{5{,}2e-02}$	$4{,}65e-02_{1{,}4e-02}$	$6{,}67e-02_{2{,}0e-02}$	$5{,}04e-02_{1{,}7e-02}$
BB12015	$2{,}24e-01_{2{,}1e-02}$	$1{,}07e-01_{1{,}6e-02}$	$1{,}27e-01_{3{,}0e-02}$	$1{,}56e-01_{7{,}0e-02}$
BB12016	$2{,}57e-01_{9{,}1e-02}$	$1{,}00e-01_{5{,}2e-02}$	$1{,}87e-01_{8{,}2e-02}$	$1{,}72e-01_{9{,}6e-02}$
BB12017	$2{,}62e-01_{3{,}7e-02}$	$5{,}56e-02_{1{,}2e-02}$	$7{,}21e-02_{2{,}0e-02}$	$7{,}85e-02_{2{,}5e-02}$
BB12018	$1{,}30e-01_{4{,}3e-02}$	$5{,}80e-02_{2{,}3e-02}$	$8{,}05e-02_{3{,}8e-02}$	$9{,}32e-02_{3{,}1e-02}$
BB12020	$2{,}13e-01_{2{,}7e-02}$	$5{,}63e-02_{1{,}7e-02}$	$6{,}84e-02_{1{,}8e-02}$	$7{,}44e-02_{2{,}4e-02}$
BB12021	$1{,}71e-01_{8{,}6e-02}$	$1{,}18e-01_{7{,}4e-02}$	$2{,}04e-01_{7{,}3e-02}$	$1{,}77e-01_{5{,}7e-02}$
BB12022	$1{,}44e-01_{8{,}9e-02}$	$1{,}04e-01_{2{,}9e-02}$	$1{,}23e-01_{3{,}0e-02}$	$1{,}29e-01_{3{,}2e-02}$
BB12023	$2{,}13e-01_{3{,}6e-02}$	$4{,}09e-02_{1{,}3e-02}$	$4{,}74e-02_{1{,}0e-02}$	$5{,}09e-02_{1{,}6e-02}$
BB12024	$1{,}37e-01_{3{,}6e-02}$	$4{,}48e-02_{1{,}9e-02}$	$6{,}31e-02_{2{,}6e-02}$	$7{,}28e-02_{1{,}8e-02}$
BB12025	$4{,}33e-01_{1{,}3e-02}$	$1{,}16e-01_{5{,}0e-02}$	$2{,}00e-01_{6{,}0e-02}$	$1{,}59e-01_{7{,}9e-02}$
BB12026	$1{,}54e-01_{2{,}7e-02}$	$1{,}09e-01_{3{,}4e-02}$	$1{,}60e-01_{4{,}4e-02}$	$1{,}34e-01_{2{,}6e-02}$
BB12027	$1{,}14e-01_{4{,}4e-02}$	$1{,}07e-01_{3{,}5e-02}$	$1{,}47e-01_{3{,}9e-02}$	$1{,}32e-01_{5{,}7e-02}$
BB12028	$2{,}15e-01_{1{,}9e-01}$	$6{,}34e-02_{1{,}7e-02}$	$9{,}57e-02_{3{,}2e-02}$	$9{,}88e-02_{4{,}8e-02}$
BB12031	$2{,}28e-01_{1{,}1e-01}$	$1{,}19e-01_{6{,}0e-02}$	$1{,}38e-01_{4{,}7e-02}$	$1{,}47e-01_{6{,}4e-02}$
BB12032	$1{,}92e-01_{3{,}7e-02}$	$8{,}79e-02_{3{,}6e-02}$	$1{,}55e-01_{4{,}0e-02}$	$1{,}27e-01_{4{,}6e-02}$
BB12033	$3{,}21e-01_{2{,}7e-02}$	$7{,}15e-02_{3{,}4e-02}$	$1{,}15e-01_{3{,}9e-02}$	$9{,}03e-02_{4{,}5e-02}$
BB12034	$2{,}06e-01_{3{,}3e-02}$	$6{,}13e-02_{1{,}5e-02}$	$9{,}39e-02_{3{,}5e-02}$	$8{,}89e-02_{2{,}4e-02}$
BB12035	$2{,}16e-01_{5{,}5e-02}$	$1{,}22e-01_{4{,}2e-02}$	$1{,}77e-01_{4{,}3e-02}$	$1{,}37e-01_{3{,}6e-02}$
BB12036	$1{,}09e-01_{3{,}9e-02}$	$6{,}94e-02_{1{,}9e-02}$	$6{,}54e-02_{2{,}1e-02}$	$9{,}26e-02_{3{,}0e-02}$
BB12037	$2{,}69e-01_{4{,}0e-02}$	$8{,}69e-02_{3{,}2e-02}$	$1{,}09e-01_{2{,}4e-02}$	$1{,}21e-01_{4{,}3e-02}$
BB12038	$2{,}59e-01_{1{,}3e-01}$	$9{,}91e-02_{3{,}2e-02}$	$1{,}41e-01_{4{,}3e-02}$	$1{,}69e-01_{5{,}0e-02}$
BB12039	$4{,}68e-01_{1{,}7e-02}$	$1{,}64e-01_{9{,}4e-02}$	$2{,}61e-01_{8{,}9e-02}$	$2{,}02e-01_{5{,}3e-02}$
BB12041	$1{,}65e-01_{3{,}0e-02}$	$1{,}14e-01_{2{,}9e-02}$	$1{,}63e-01_{4{,}5e-02}$	$1{,}35e-01_{2{,}7e-02}$
BB12042	$2{,}06e-01_{3{,}1e-02}$	$6{,}10e-02_{1{,}9e-02}$	$9{,}58e-02_{1{,}8e-02}$	$1{,}03e-01_{4{,}6e-02}$
BB12043	$3{,}17e-01_{2{,}8e-01}$	$2{,}31e-01_{1{,}2e-01}$	$3{,}36e-01_{1{,}5e-01}$	$3{,}49e-01_{1{,}5e-01}$
BB12044	$2{,}37e-01_{4{,}0e-02}$	$7{,}44e-02_{3{,}0e-02}$	$7{,}44e-02_{3{,}0e-02}$	$1{,}11e-01_{3{,}8e-02}$

Tabla B.23: Mediana y rango intercuartílico del indicador I_{e+} para la familia RV30 del BAliBASE 3.0 y para los algoritmos: GWASF-GA, NSGA-II, MOCell, y NSGA-III.

	GWASF-GA	NSGA-II	MOCell	NSGA-III
BB30001	$1{,}68e-01_{9{,}3e-01}$	$1{,}63e-01_{1{,}0e-01}$	$1{,}92e-01_{8{,}6e-01}$	$2{,}64e-01_{7{,}7e-01}$
BB30003	$1{,}41e-01_{4{,}8e-01}$	$1{,}10e-01_{2{,}1e-02}$	$1{,}22e-01_{4{,}8e-01}$	$1{,}95e-01_{4{,}6e-01}$
BB30004	$3{,}78e-01_{3{,}5e+00}$	$3{,}28e-01_{1{,}2e-01}$	$5{,}16e-01_{6{,}8e+00}$	$3{,}97e-01_{3{,}5e+00}$
BB30005	$2{,}17e-01_{6{,}5e+00}$	$1{,}64e-01_{4{,}6e-02}$	$2{,}17e-01_{1{,}3e+01}$	$2{,}07e-01_{1{,}3e+01}$
BB30007	$4{,}16e-01_{2{,}1e-03}$	$1{,}84e-01_{4{,}4e-02}$	$2{,}32e-01_{7{,}7e-01}$	$2{,}56e-01_{1{,}2e-01}$
BB30008	$1{,}50e-01_{5{,}9e-02}$	$1{,}13e-01_{1{,}0e-01}$	$1{,}71e-01_{8{,}8e-01}$	$2{,}15e-01_{8{,}4e-01}$
BB30010	$8{,}08e-01_{6{,}0e-02}$	$2{,}26e-01_{9{,}4e+00}$	$2{,}51e-01_{7{,}0e-02}$	$2{,}15e-01_{8{,}6e-02}$
BB30011	$3{,}81e-01_{8{,}5e-02}$	$2{,}79e-01_{1{,}6e+01}$	$2{,}31e-01_{9{,}1e-02}$	$2{,}10e-01_{1{,}1e-01}$
BB30012	$1{,}27e-01_{9{,}5e-02}$	$2{,}14e-01_{1{,}6e-01}$	$2{,}21e-01_{7{,}3e-02}$	$2{,}92e-01_{7{,}6e-01}$
BB30013	$1{,}95e-01_{4{,}0e-02}$	$7{,}04e-02_{9{,}4e-01}$	$6{,}73e-02_{9{,}5e-01}$	$1{,}78e-01_{1{,}2e-01}$
BB30015	$1{,}56e-01_{5{,}5e-01}$	$3{,}15e-01_{7{,}3e-01}$	$3{,}26e-01_{1{,}0e-01}$	$3{,}42e-01_{1{,}3e-01}$
BB30017	$3{,}35e-01_{1{,}9e-02}$	$3{,}32e-01_{6{,}7e-01}$	$6{,}58e-01_{6{,}7e-01}$	$3{,}62e-01_{3{,}3e-01}$
BB30019	$1{,}05e-01_{3{,}1e-02}$	$9{,}21e-02_{9{,}3e-01}$	$9{,}49e-02_{4{,}6e-02}$	$1{,}36e-01_{8{,}2e-02}$
BB30021	$1{,}91e-01_{6{,}0e-02}$	$2{,}33e-01_{8{,}2e-01}$	$2{,}24e-01_{8{,}2e-01}$	$3{,}27e-01_{2{,}2e-01}$
BB30022	$1{,}50e-01_{1{,}1e-01}$	$9{,}22e-02_{8{,}1e-02}$	$1{,}10e-01_{9{,}2e-01}$	$1{,}35e-01_{5{,}9e-02}$
BB30023	$6{,}24e-01_{2{,}5e-01}$	$5{,}15e-01_{2{,}3e+00}$	$5{,}06e-01_{2{,}3e+00}$	$5{,}16e-01_{3{,}9e-01}$
BB30024	$2{,}20e-01_{1{,}4e-01}$	$2{,}00e-01_{5{,}0e-01}$	$2{,}06e-01_{9{,}8e-02}$	$2{,}13e-01_{1{,}6e-01}$
BB30025	$1{,}96e-01_{1{,}9e-01}$	$2{,}19e-01_{2{,}0e-01}$	$3{,}02e-01_{1{,}4e-01}$	$2{,}57e-01_{2{,}5e-01}$
BB30026	$2{,}68e-01_{3{,}0e-03}$	$2{,}58e-01_{1{,}7e-02}$	$2{,}62e-01_{3{,}5e-02}$	$2{,}69e-01_{4{,}0e-01}$
BB30027	$6{,}46e-01_{2{,}8e-02}$	$3{,}09e-01_{3{,}6e-02}$	$3{,}15e-01_{4{,}7e-02}$	$3{,}42e-01_{2{,}4e-01}$
BB30028	$2{,}23e-01_{1{,}6e-01}$	$1{,}64e-01_{8{,}7e-01}$	$1{,}61e-01_{6{,}5e-02}$	$3{,}35e-01_{7{,}3e-01}$
BB30030	$2{,}89e-01_{4{,}2e-01}$	$3{,}92e-01_{6{,}7e-01}$	$3{,}55e-01_{6{,}4e-02}$	$4{,}25e-01_{1{,}5e-01}$

Tabla B.24: Mediana y rango intercuartílico del indicador I_{IGD+} para la familia RV30 del BAliBASE 3.0 y para los algoritmos: GWASF-GA, NSGA-II, MOCell, y NSGA-III.

	GWASF-GA	NSGA-II	MOCell	NSGA-III
BB30001	$8{,}23e-02_{7{,}9e-01}$	$7{,}10e-02_{5{,}5e-02}$	$9{,}03e-02_{7{,}7e-01}$	$1{,}21e-01_{7{,}3e-01}$
BB30003	$7{,}26e-02_{3{,}9e-01}$	$2{,}81e-02_{1{,}3e-02}$	$3{,}83e-02_{4{,}0e-01}$	$4{,}73e-02_{4{,}0e-01}$
BB30004	$2{,}91e-01_{3{,}3e+00}$	$1{,}89e-01_{6{,}6e-02}$	$3{,}11e-01_{6{,}6e+00}$	$2{,}07e-01_{3{,}4e+00}$
BB30005	$9{,}43e-02_{6{,}3e+00}$	$7{,}09e-02_{2{,}4e-02}$	$1{,}30e-01_{1{,}3e+01}$	$9{,}42e-02_{1{,}3e+01}$
BB30007	$1{,}55e-01_{1{,}0e-02}$	$8{,}30e-02_{2{,}3e-02}$	$1{,}20e-01_{6{,}0e-01}$	$1{,}30e-01_{5{,}0e-02}$
BB30008	$1{,}07e-01_{5{,}9e-02}$	$4{,}98e-02_{3{,}7e-02}$	$6{,}42e-02_{8{,}7e-01}$	$7{,}06e-02_{8{,}6e-01}$
BB30010	$3{,}78e-01_{4{,}8e-02}$	$1{,}23e-01_{9{,}0e+00}$	$1{,}49e-01_{3{,}7e-02}$	$1{,}26e-01_{4{,}9e-02}$
BB30011	$1{,}78e-01_{6{,}2e-02}$	$1{,}95e-01_{1{,}6e+01}$	$1{,}84e-01_{9{,}3e-02}$	$1{,}62e-01_{6{,}4e-02}$
BB30012	$4{,}49e-02_{2{,}6e-02}$	$6{,}37e-02_{4{,}9e-02}$	$6{,}45e-02_{2{,}5e-02}$	$8{,}35e-02_{8{,}5e-01}$
BB30013	$9{,}20e-02_{2{,}8e-02}$	$2{,}87e-02_{7{,}7e-01}$	$3{,}47e-02_{7{,}7e-01}$	$7{,}81e-02_{5{,}0e-02}$
BB30015	$5{,}46e-02_{4{,}7e-01}$	$1{,}06e-01_{8{,}3e-01}$	$1{,}37e-01_{6{,}0e-02}$	$1{,}13e-01_{6{,}9e-02}$
BB30017	$1{,}49e-01_{2{,}7e-02}$	$1{,}47e-01_{6{,}4e-01}$	$3{,}53e-01_{6{,}0e-01}$	$1{,}95e-01_{1{,}9e-01}$
BB30019	$5{,}45e-02_{1{,}7e-02}$	$4{,}70e-02_{8{,}8e-01}$	$4{,}72e-02_{1{,}3e-02}$	$5{,}76e-02_{2{,}7e-02}$
BB30021	$3{,}67e-02_{2{,}5e-02}$	$3{,}81e-02_{7{,}5e-01}$	$6{,}02e-02_{7{,}4e-01}$	$7{,}12e-02_{1{,}4e-01}$
BB30022	$8{,}76e-02_{7{,}6e-02}$	$4{,}56e-02_{3{,}7e-02}$	$8{,}54e-02_{7{,}7e-01}$	$6{,}89e-02_{3{,}2e-02}$
BB30023	$3{,}67e-01_{2{,}1e-01}$	$3{,}31e-01_{2{,}0e+00}$	$3{,}62e-01_{2{,}0e+00}$	$3{,}55e-01_{3{,}1e-01}$
BB30024	$8{,}62e-02_{4{,}6e-02}$	$6{,}63e-02_{4{,}3e-01}$	$9{,}27e-02_{5{,}8e-02}$	$1{,}30e-01_{6{,}1e-02}$
BB30025	$1{,}23e-01_{1{,}4e-01}$	$1{,}30e-01_{1{,}3e-01}$	$1{,}90e-01_{9{,}0e-02}$	$1{,}33e-01_{1{,}6e-01}$
BB30026	$7{,}18e-02_{1{,}8e-02}$	$8{,}26e-02_{1{,}3e-02}$	$1{,}00e-01_{4{,}0e-02}$	$1{,}09e-01_{4{,}2e-01}$
BB30027	$2{,}60e-01_{2{,}2e-02}$	$1{,}73e-01_{4{,}5e-02}$	$2{,}28e-01_{3{,}7e-02}$	$1{,}99e-01_{6{,}9e-02}$
BB30028	$1{,}32e-01_{6{,}9e-02}$	$7{,}51e-02_{7{,}9e-01}$	$9{,}37e-02_{3{,}5e-02}$	$1{,}58e-01_{7{,}4e-01}$
BB30030	$1{,}38e-01_{3{,}5e-01}$	$1{,}90e-01_{5{,}7e-01}$	$1{,}90e-01_{4{,}7e-02}$	$1{,}84e-01_{4{,}4e-02}$

Tabla B.25: Mediana y rango intercuartílico del indicador I_{e+} para la familia RV40 del BAliBASE 3.0 y para los algoritmos: GWASF-GA, NSGA-II, MOCell, y NSGA-III.

	GWASF-GA	NSGA-II	MOCell	NSGA-III
BB40001	$3{,}63e-01_{7{,}2e-02}$	$2{,}14e-01_{4{,}3e-02}$	$2{,}22e-01_{4{,}8e-02}$	$3{,}08e-01_{1{,}1e-01}$
BB40002	$9{,}69e-01_{4{,}4e-02}$	$2{,}38e-01_{2{,}5e-02}$	$2{,}10e-01_{5{,}7e-02}$	$7{,}77e-01_{5{,}6e-01}$
BB40003	$7{,}96e-01_{5{,}0e-01}$	$2{,}70e-01_{6{,}4e-02}$	$2{,}81e-01_{4{,}8e-02}$	$2{,}95e-01_{3{,}6e-02}$
BB40004	$3{,}64e-01_{2{,}5e-01}$	$3{,}03e-01_{6{,}8e-02}$	$3{,}32e-01_{4{,}0e-02}$	$3{,}03e-01_{4{,}9e-02}$
BB40005	$7{,}80e-01_{2{,}6e-01}$	$2{,}82e-01_{9{,}6e-02}$	$2{,}68e-01_{7{,}2e-02}$	$2{,}95e-01_{1{,}4e-01}$
BB40006	$2{,}02e-01_{5{,}0e-02}$	$2{,}08e-01_{5{,}4e-02}$	$2{,}14e-01_{7{,}2e-02}$	$2{,}17e-01_{5{,}5e-02}$
BB40008	$5{,}21e-01_{7{,}3e-02}$	$2{,}80e-01_{7{,}7e-02}$	$2{,}22e-01_{4{,}7e-02}$	$2{,}91e-01_{3{,}8e-02}$
BB40009	$4{,}81e-01_{6{,}1e-02}$	$3{,}58e-01_{8{,}8e-02}$	$3{,}44e-01_{8{,}1e-02}$	$3{,}51e-01_{5{,}1e-02}$
BB40010	$9{,}21e-01_{0{,}0e+00}$	$3{,}50e-01_{1{,}5e-01}$	$3{,}80e-01_{2{,}0e-01}$	$3{,}06e-01_{2{,}2e-01}$
BB40011	$8{,}71e-01_{5{,}7e-02}$	$6{,}50e-01_{1{,}1e-01}$	$5{,}46e-01_{1{,}6e-01}$	$8{,}18e-01_{1{,}5e-01}$
BB40012	$4{,}53e-01_{1{,}2e-01}$	$2{,}57e-01_{1{,}6e-01}$	$3{,}27e-01_{1{,}6e-01}$	$3{,}24e-01_{1{,}6e-01}$
BB40013	$8{,}02e-01_{1{,}7e-02}$	$3{,}03e-01_{5{,}8e-02}$	$3{,}12e-01_{2{,}3e-02}$	$5{,}42e-01_{3{,}3e-01}$
BB40014	$7{,}31e-01_{1{,}3e-01}$	$2{,}32e-01_{1{,}5e-01}$	$2{,}84e-01_{1{,}0e-01}$	$2{,}40e-01_{9{,}2e-02}$
BB40016	$7{,}39e-01_{2{,}0e-01}$	$3{,}39e-01_{3{,}3e-01}$	$6{,}56e-01_{3{,}3e-01}$	$6{,}60e-01_{1{,}1e-02}$
BB40017	$7{,}37e-01_{2{,}5e-01}$	$6{,}57e-01_{3{,}4e-01}$	$3{,}54e-01_{3{,}4e-01}$	$6{,}57e-01_{3{,}2e-01}$
BB40018	$1{,}01e+00_{3{,}8e-03}$	$4{,}10e-01_{1{,}6e-01}$	$4{,}73e-01_{1{,}2e-01}$	$4{,}73e-01_{1{,}3e-01}$
BB40019	$7{,}24e-01_{1{,}1e-01}$	$8{,}95e-02_{4{,}2e-02}$	$1{,}26e-01_{5{,}0e-02}$	$1{,}54e-01_{9{,}2e-02}$
BB40021	$8{,}16e-01_{2{,}1e-02}$	$4{,}37e-01_{1{,}5e-01}$	$4{,}65e-01_{1{,}5e-01}$	$4{,}00e-01_{1{,}1e-01}$
BB40022	$8{,}02e-01_{5{,}5e-01}$	$2{,}23e-01_{8{,}3e-02}$	$2{,}63e-01_{5{,}7e-02}$	$2{,}96e-01_{1{,}4e-01}$
BB40023	$3{,}60e-01_{8{,}6e-02}$	$2{,}16e-01_{7{,}0e-02}$	$2{,}58e-01_{1{,}2e-01}$	$2{,}48e-01_{1{,}2e-01}$
BB40024	$7{,}92e-01_{6{,}9e-02}$	$2{,}93e-01_{5{,}8e-02}$	$3{,}10e-01_{2{,}9e-01}$	$3{,}26e-01_{7{,}4e-02}$
BB40026	$7{,}91e-01_{1{,}1e-01}$	$5{,}94e-01_{1{,}6e-01}$	$5{,}34e-01_{2{,}3e-01}$	$3{,}68e-01_{2{,}9e-01}$
BB40027	$6{,}36e-01_{1{,}3e-01}$	$5{,}10e-01_{7{,}5e-02}$	$5{,}64e-01_{1{,}1e-01}$	$5{,}99e-01_{1{,}7e-01}$
BB40028	$2{,}86e-01_{7{,}4e-02}$	$1{,}80e-01_{7{,}2e-02}$	$1{,}82e-01_{4{,}3e-02}$	$2{,}42e-01_{4{,}1e-02}$
BB40029	$3{,}50e-01_{2{,}2e-01}$	$1{,}32e-01_{2{,}7e-02}$	$1{,}59e-01_{4{,}8e-02}$	$1{,}40e-01_{2{,}9e-02}$
BB40030	$5{,}03e-01_{1{,}0e-01}$	$2{,}37e-01_{2{,}4e-01}$	$3{,}31e-01_{2{,}2e-01}$	$4{,}05e-01_{2{,}5e-01}$
BB40034	$2{,}80e-01_{4{,}1e-02}$	$1{,}60e-01_{6{,}2e-02}$	$1{,}60e-01_{4{,}2e-02}$	$2{,}42e-01_{6{,}6e-02}$
BB40035	$5{,}36e-01_{1{,}7e-01}$	$3{,}40e-01_{1{,}7e-01}$	$3{,}07e-01_{9{,}6e-02}$	$3{,}72e-01_{1{,}5e-01}$
BB40036	$5{,}35e-01_{2{,}3e-01}$	$2{,}19e-01_{5{,}4e-02}$	$2{,}44e-01_{7{,}9e-02}$	$3{,}17e-01_{2{,}1e-01}$
BB40037	$5{,}54e-01_{3{,}6e-02}$	$2{,}92e-01_{0{,}0e+00}$	$2{,}92e-01_{0{,}0e+00}$	$3{,}28e-01_{1{,}3e-01}$
BB40038	$4{,}57e-01_{1{,}3e-01}$	$4{,}50e-01_{8{,}9e-02}$	$7{,}00e-01_{3{,}0e-01}$	$4{,}79e-01_{9{,}6e-02}$
BB40039	$9{,}31e-01_{6{,}4e-02}$	$2{,}36e-01_{7{,}3e-02}$	$3{,}19e-01_{1{,}1e-01}$	$2{,}47e-01_{9{,}0e-02}$
BB40040	$9{,}24e-01_{8{,}0e-02}$	$2{,}79e-01_{1{,}0e-01}$	$4{,}00e-01_{1{,}6e-01}$	$3{,}49e-01_{2{,}3e-01}$
BB40041	$4{,}44e-01_{4{,}9e-02}$	$2{,}49e-01_{8{,}4e-02}$	$2{,}84e-01_{7{,}6e-02}$	$2{,}70e-01_{7{,}2e-02}$
BB40042	$3{,}46e-01_{5{,}6e-01}$	$2{,}18e-01_{5{,}4e-02}$	$2{,}18e-01_{8{,}0e-02}$	$3{,}05e-01_{1{,}9e-01}$
BB40043	$3{,}78e-01_{1{,}1e-01}$	$2{,}83e-01_{7{,}5e-02}$	$3{,}12e-01_{1{,}0e-01}$	$2{,}59e-01_{9{,}3e-02}$
BB40044	$2{,}17e-01_{4{,}2e-03}$	$1{,}82e-01_{4{,}6e-02}$	$1{,}68e-01_{5{,}1e-02}$	$2{,}04e-01_{3{,}2e-02}$
BB40045	$4{,}70e-01_{2{,}7e-01}$	$2{,}24e-01_{3{,}9e-02}$	$2{,}28e-01_{3{,}3e-02}$	$3{,}47e-01_{1{,}6e-01}$
BB40046	$5{,}97e-01_{7{,}5e-02}$	$3{,}53e-01_{8{,}5e-02}$	$3{,}39e-01_{8{,}4e-02}$	$3{,}62e-01_{4{,}4e-02}$
BB40047	$8{,}82e-01_{6{,}1e-02}$	$3{,}97e-01_{1{,}6e-01}$	$4{,}40e-01_{1{,}3e-01}$	$3{,}99e-01_{1{,}6e-01}$
BB40049	$8{,}41e-01_{1{,}8e-01}$	$3{,}64e-01_{1{,}2e-01}$	$4{,}25e-01_{1{,}1e-01}$	$5{,}51e-01_{2{,}5e-01}$

Tabla B.26: Mediana y rango intercuartílico del indicador I_{IGD+} para la familia RV40 del BAliBASE 3.0 y para los algoritmos: GWASF-GA, NSGA-II, MOCell, y NSGA-III.

	GWASF-GA	NSGA-II	MOCell	NSGA-III
BB40001	$1{,}70e-01_{3{,}2e-02}$	$1{,}33e-01_{3{,}4e-02}$	$1{,}38e-01_{4{,}3e-02}$	$1{,}81e-01_{5{,}2e-02}$
BB40002	$3{,}76e-01_{4{,}7e-02}$	$1{,}11e-01_{1{,}5e-02}$	$1{,}09e-01_{2{,}7e-02}$	$3{,}13e-01_{2{,}1e-01}$
BB40003	$3{,}36e-01_{2{,}2e-01}$	$9{,}33e-02_{2{,}6e-02}$	$1{,}14e-01_{3{,}2e-02}$	$1{,}26e-01_{3{,}3e-02}$
BB40004	$1{,}76e-01_{8{,}8e-02}$	$1{,}54e-01_{4{,}8e-02}$	$2{,}17e-01_{6{,}9e-02}$	$1{,}71e-01_{3{,}4e-02}$
BB40005	$2{,}76e-01_{1{,}3e-01}$	$9{,}75e-02_{5{,}1e-02}$	$1{,}45e-01_{4{,}5e-02}$	$1{,}25e-01_{3{,}1e-02}$
BB40006	$9{,}45e-02_{4{,}7e-02}$	$8{,}98e-02_{2{,}8e-02}$	$1{,}28e-01_{4{,}8e-02}$	$9{,}87e-02_{3{,}1e-02}$
BB40008	$1{,}54e-01_{4{,}1e-02}$	$8{,}69e-02_{2{,}1e-02}$	$1{,}13e-01_{1{,}4e-02}$	$1{,}16e-01_{2{,}1e-02}$
BB40009	$1{,}18e-01_{2{,}5e-02}$	$9{,}18e-02_{3{,}2e-02}$	$1{,}09e-01_{2{,}2e-02}$	$1{,}05e-01_{1{,}5e-02}$
BB40010	$5{,}34e-01_{6{,}7e-03}$	$1{,}30e-01_{1{,}4e-01}$	$2{,}13e-01_{8{,}7e-02}$	$1{,}45e-01_{1{,}3e-01}$
BB40011	$1{,}41e-01_{5{,}2e-02}$	$1{,}15e-01_{4{,}6e-02}$	$1{,}46e-01_{1{,}7e-02}$	$1{,}28e-01_{4{,}8e-02}$
BB40012	$1{,}97e-01_{5{,}4e-02}$	$1{,}30e-01_{5{,}8e-02}$	$1{,}54e-01_{4{,}5e-02}$	$1{,}31e-01_{4{,}9e-02}$
BB40013	$2{,}71e-01_{3{,}5e-02}$	$1{,}32e-01_{3{,}5e-02}$	$1{,}61e-01_{4{,}4e-02}$	$1{,}56e-01_{8{,}2e-02}$
BB40014	$2{,}43e-01_{8{,}4e-02}$	$9{,}29e-02_{4{,}5e-02}$	$1{,}43e-01_{3{,}1e-02}$	$1{,}03e-01_{4{,}2e-02}$
BB40016	$4{,}57e-01_{3{,}4e-01}$	$1{,}42e-01_{9{,}3e-02}$	$1{,}97e-01_{8{,}4e-02}$	$2{,}19e-01_{7{,}0e-02}$
BB40017	$4{,}19e-01_{2{,}6e-01}$	$1{,}92e-01_{8{,}6e-02}$	$1{,}93e-01_{8{,}5e-02}$	$2{,}23e-01_{1{,}7e-01}$
BB40018	$5{,}94e-01_{6{,}4e-02}$	$2{,}87e-01_{1{,}3e-01}$	$3{,}33e-01_{1{,}3e-01}$	$3{,}05e-01_{7{,}9e-02}$
BB40019	$3{,}48e-01_{8{,}6e-02}$	$4{,}27e-02_{2{,}6e-02}$	$6{,}60e-02_{2{,}3e-02}$	$8{,}40e-02_{5{,}0e-02}$
BB40021	$2{,}85e-01_{2{,}0e-02}$	$1{,}78e-01_{7{,}3e-02}$	$2{,}24e-01_{7{,}3e-02}$	$1{,}85e-01_{6{,}3e-02}$
BB40022	$4{,}20e-01_{3{,}0e-01}$	$1{,}20e-01_{5{,}2e-02}$	$1{,}49e-01_{4{,}0e-02}$	$1{,}64e-01_{9{,}3e-02}$
BB40023	$2{,}05e-01_{6{,}8e-02}$	$1{,}47e-01_{8{,}7e-02}$	$1{,}94e-01_{1{,}3e-01}$	$1{,}56e-01_{1{,}4e-01}$
BB40024	$2{,}25e-01_{5{,}2e-02}$	$1{,}35e-01_{4{,}0e-02}$	$1{,}68e-01_{3{,}9e-02}$	$1{,}91e-01_{5{,}1e-02}$
BB40026	$2{,}48e-01_{5{,}9e-02}$	$1{,}81e-01_{4{,}2e-02}$	$1{,}98e-01_{4{,}6e-02}$	$1{,}81e-01_{8{,}7e-02}$
BB40027	$4{,}27e-01_{2{,}6e-01}$	$3{,}56e-01_{1{,}6e-01}$	$4{,}66e-01_{2{,}0e-01}$	$4{,}20e-01_{1{,}6e-01}$
BB40028	$1{,}27e-01_{3{,}8e-02}$	$9{,}18e-02_{2{,}9e-02}$	$1{,}03e-01_{2{,}9e-02}$	$1{,}07e-01_{2{,}6e-02}$
BB40029	$1{,}65e-01_{8{,}7e-02}$	$7{,}93e-02_{3{,}7e-02}$	$1{,}18e-01_{5{,}6e-02}$	$8{,}74e-02_{2{,}9e-02}$
BB40030	$2{,}36e-01_{7{,}5e-02}$	$1{,}50e-01_{1{,}2e-01}$	$1{,}73e-01_{1{,}1e-01}$	$2{,}01e-01_{9{,}1e-02}$
BB40034	$1{,}03e-01_{2{,}4e-02}$	$7{,}29e-02_{3{,}5e-02}$	$8{,}55e-02_{1{,}9e-02}$	$9{,}05e-02_{2{,}3e-02}$
BB40035	$2{,}00e-01_{1{,}1e-01}$	$9{,}79e-02_{1{,}2e-01}$	$1{,}56e-01_{8{,}6e-02}$	$1{,}93e-01_{5{,}5e-02}$
BB40036	$1{,}40e-01_{3{,}9e-02}$	$9{,}01e-02_{3{,}1e-02}$	$1{,}08e-01_{3{,}3e-02}$	$1{,}09e-01_{4{,}1e-02}$
BB40037	$2{,}93e-01_{3{,}2e-02}$	$1{,}96e-01_{3{,}0e-02}$	$1{,}96e-01_{3{,}1e-02}$	$2{,}22e-01_{8{,}4e-02}$
BB40038	$3{,}79e-01_{1{,}8e-01}$	$3{,}36e-01_{2{,}0e-01}$	$6{,}30e-01_{3{,}0e-01}$	$3{,}85e-01_{1{,}3e-01}$
BB40039	$3{,}62e-01_{3{,}8e-02}$	$9{,}98e-02_{4{,}6e-02}$	$1{,}35e-01_{2{,}9e-02}$	$1{,}30e-01_{4{,}5e-02}$
BB40040	$3{,}57e-01_{7{,}7e-02}$	$9{,}57e-02_{3{,}0e-02}$	$1{,}60e-01_{4{,}4e-02}$	$1{,}32e-01_{8{,}5e-02}$
BB40041	$2{,}82e-01_{4{,}9e-02}$	$1{,}33e-01_{1{,}3e-01}$	$1{,}41e-01_{3{,}4e-02}$	$1{,}36e-01_{1{,}1e-01}$
BB40042	$2{,}16e-01_{5{,}8e-01}$	$1{,}04e-01_{5{,}7e-02}$	$1{,}16e-01_{6{,}7e-02}$	$1{,}61e-01_{1{,}0e-01}$
BB40043	$1{,}33e-01_{9{,}2e-02}$	$1{,}15e-01_{1{,}0e-01}$	$1{,}71e-01_{1{,}0e-01}$	$1{,}25e-01_{7{,}2e-02}$
BB40044	$7{,}62e-02_{2{,}2e-02}$	$7{,}13e-02_{2{,}2e-02}$	$9{,}62e-02_{3{,}0e-02}$	$8{,}39e-02_{1{,}9e-02}$
BB40045	$1{,}86e-01_{8{,}8e-02}$	$1{,}00e-01_{4{,}2e-02}$	$1{,}45e-01_{4{,}8e-02}$	$1{,}76e-01_{1{,}4e-01}$
BB40046	$2{,}01e-01_{3{,}9e-02}$	$1{,}22e-01_{3{,}1e-02}$	$1{,}35e-01_{2{,}2e-02}$	$1{,}44e-01_{4{,}6e-02}$
BB40047	$3{,}59e-01_{8{,}3e-02}$	$2{,}52e-01_{1{,}8e-01}$	$2{,}79e-01_{1{,}1e-01}$	$2{,}33e-01_{1{,}4e-01}$
BB40049	$3{,}24e-01_{1{,}4e-01}$	$2{,}04e-01_{1{,}1e-01}$	$2{,}04e-01_{8{,}4e-02}$	$3{,}40e-01_{2{,}5e-01}$

Tabla B.27: Mediana y rango intercuartílico del indicador I_{e+} para la familia RV50 del BAliBASE 3.0 y para los algoritmos: GWASF-GA, NSGA-II, MOCell, y NSGA-III.

	GWASF-GA	NSGA-II	MOCell	NSGA-III
BB50001	$5{,}17e-01_{1{,}3e-01}$	$3{,}44e-01_{7{,}0e-03}$	$3{,}45e-01_{6{,}1e-03}$	$3{,}47e-01_{8{,}2e-03}$
BB50002	$5{,}42e-01_{1{,}8e-01}$	$3{,}50e-01_{2{,}7e-02}$	$3{,}45e-01_{2{,}1e-02}$	$3{,}58e-01_{1{,}3e-02}$
BB50003	$6{,}03e-01_{4{,}4e-02}$	$2{,}88e-01_{9{,}2e-02}$	$3{,}43e-01_{6{,}8e-02}$	$2{,}96e-01_{1{,}2e-01}$
BB50004	$5{,}70e-01_{4{,}1e-01}$	$2{,}55e-01_{7{,}5e-02}$	$1{,}97e-01_{1{,}2e-01}$	$2{,}55e-01_{7{,}7e-02}$
BB50005	$7{,}68e-01_{4{,}7e-01}$	$3{,}06e-01_{1{,}0e-01}$	$3{,}33e-01_{6{,}1e-02}$	$3{,}24e-01_{1{,}4e-01}$
BB50006	$2{,}76e-01_{4{,}5e-01}$	$2{,}63e-01_{2{,}0e-02}$	$2{,}71e-01_{1{,}1e-01}$	$2{,}89e-01_{6{,}1e-02}$
BB50008	$9{,}15e-01_{5{,}0e-01}$	$2{,}61e-01_{1{,}9e-01}$	$2{,}72e-01_{1{,}4e-01}$	$2{,}73e-01_{1{,}5e-01}$
BB50011	$2{,}52e-01_{1{,}3e-01}$	$2{,}34e-01_{8{,}9e-02}$	$1{,}51e-01_{2{,}5e-02}$	$2{,}22e-01_{6{,}8e-02}$
BB50012	$4{,}51e-01_{9{,}4e-02}$	$1{,}79e-01_{2{,}9e-01}$	$1{,}98e-01_{3{,}7e-02}$	$2{,}40e-01_{1{,}7e-01}$
BB50013	$7{,}29e-01_{2{,}8e-01}$	$4{,}47e-01_{2{,}0e-01}$	$3{,}01e-01_{1{,}3e-01}$	$4{,}11e-01_{2{,}0e-01}$
BB50014	$8{,}03e-01_{5{,}8e-01}$	$2{,}08e-01_{8{,}8e-02}$	$2{,}57e-01_{9{,}7e-02}$	$2{,}61e-01_{1{,}5e-01}$
BB50015	$5{,}88e-01_{4{,}8e-02}$	$2{,}60e-01_{7{,}5e-02}$	$2{,}79e-01_{4{,}2e-02}$	$2{,}79e-01_{3{,}8e-02}$
BB50016	$8{,}37e-01_{4{,}4e-02}$	$1{,}97e-01_{6{,}6e-01}$	$1{,}86e-01_{2{,}3e-02}$	$1{,}99e-01_{5{,}0e-03}$

Tabla B.28: Mediana y rango intercuartílico del indicador I_{IGD+} para la familia RV50 del BAliBASE 3.0 y para los algoritmos: GWASF-GA, NSGA-II, MOCell, y NSGA-III.

	GWASF-GA	NSGA-II	MOCell	NSGA-III
BB50001	$1{,}98e-01_{6{,}1e-02}$	$1{,}33e-01_{2{,}3e-02}$	$1{,}47e-01_{1{,}6e-02}$	$1{,}50e-01_{3{,}8e-02}$
BB50002	$2{,}14e-01_{6{,}8e-02}$	$1{,}27e-01_{2{,}4e-02}$	$1{,}48e-01_{4{,}0e-02}$	$1{,}29e-01_{2{,}2e-02}$
BB50003	$1{,}22e-01_{2{,}8e-02}$	$6{,}12e-02_{1{,}6e-02}$	$8{,}04e-02_{3{,}5e-02}$	$7{,}13e-02_{4{,}5e-02}$
BB50004	$2{,}01e-01_{2{,}1e-01}$	$6{,}47e-02_{1{,}4e-02}$	$6{,}84e-02_{1{,}7e-02}$	$7{,}12e-02_{1{,}5e-02}$
BB50005	$2{,}61e-01_{1{,}9e-01}$	$8{,}21e-02_{2{,}8e-02}$	$1{,}13e-01_{2{,}1e-02}$	$8{,}94e-02_{4{,}2e-02}$
BB50006	$1{,}63e-01_{1{,}4e-01}$	$1{,}30e-01_{4{,}8e-02}$	$1{,}50e-01_{7{,}2e-02}$	$1{,}58e-01_{5{,}6e-02}$
BB50008	$4{,}02e-01_{2{,}1e-01}$	$1{,}16e-01_{1{,}0e-01}$	$1{,}45e-01_{1{,}2e-01}$	$1{,}22e-01_{9{,}1e-02}$
BB50011	$1{,}22e-01_{9{,}1e-02}$	$1{,}15e-01_{5{,}4e-02}$	$9{,}76e-02_{2{,}3e-02}$	$1{,}21e-01_{4{,}3e-02}$
BB50012	$2{,}02e-01_{7{,}6e-02}$	$1{,}08e-01_{1{,}1e-01}$	$1{,}27e-01_{5{,}5e-02}$	$1{,}16e-01_{4{,}7e-02}$
BB50013	$3{,}55e-01_{1{,}4e-01}$	$2{,}46e-01_{7{,}8e-02}$	$1{,}80e-01_{1{,}0e-01}$	$2{,}08e-01_{8{,}6e-02}$
BB50014	$3{,}59e-01_{2{,}5e-01}$	$9{,}61e-02_{7{,}9e-02}$	$1{,}16e-01_{6{,}9e-02}$	$1{,}22e-01_{7{,}7e-02}$
BB50015	$2{,}00e-01_{5{,}1e-02}$	$1{,}13e-01_{6{,}9e-02}$	$1{,}46e-01_{4{,}6e-02}$	$1{,}38e-01_{6{,}1e-02}$
BB50016	$3{,}20e-01_{3{,}5e-02}$	$9{,}81e-02_{2{,}3e-01}$	$1{,}23e-01_{1{,}6e-02}$	$9{,}62e-02_{1{,}8e-02}$

Apéndice C

Resultados de M2Align

C.1. Rendimiento paralelo de M2Align

Tabla C.1: Evaluación del rendimiento paralelo de M2Aling con 5000 evaluaciones (unidades de tiempo en milisegundos). SR: Tiempo de ejecución secuencial. SU: Speed-up (SR dividido para el número de núcleos). Eff: Eficiencia (SU dividida por el número de núcleos) alineando todas las instancias de la familia/grupo RV11 del BAliBASE v3.0.

Problema	**SR**	**4 núcleos**			**10 núcleos**			**20 núcleos**		
		Tiempo	**SU**	**Eff**	**Tiempo**	**SU**	**Eff**	**Tiempo**	**SU**	**Eff**
BB11001	3447	1665	2.07	52 %	1350	2.55	26 %	1571	2.19	11 %
BB11002	9679	3558	2.72	68 %	2267	4.27	43 %	2112	4.58	23 %
BB11003	96218	26150	3.68	92 %	13371	7.20	72 %	11036	8.72	44 %
BB11004	82988	23759	3.49	87 %	11840	7.01	70 %	9562	8.68	43 %
BB11005	245267	68566	3.58	89 %	31885	7.69	77 %	20927	11.72	59 %
BB11006	31023	8854	3.50	88 %	4882	6.35	64 %	3818	8.13	41 %
BB11007	185091	50504	3.66	92 %	23667	7.82	78 %	17093	10.83	54 %
BB11008	26608	7821	3.40	85 %	4745	5.61	56 %	4311	6.17	31 %
BB11009	25470	7444	3.42	86 %	4080	6.24	62 %	3742	6.81	34 %
BB11010	104346	28231	3.70	92 %	14184	7.36	74 %	11807	8.84	44 %
BB11011	21180	6798	3.12	78 %	4450	4.76	48 %	3613	5.86	29 %
BB11012	67966	19248	3.53	88 %	8893	7.64	76 %	7374	9.22	46 %
BB11013	2978	1518	1.96	49 %	1428	2.09	21 %	1293	2.30	12 %
BB11014	186854	51180	3.65	91 %	25717	7.27	73 %	21156	8.83	44 %
BB11015	41521	11645	3.57	89 %	6634	6.26	63 %	5196	7.99	40 %
BB11016	210645	55405	3.80	95 %	27782	7.58	76 %	21305	9.89	49 %
BB11017	28846	8831	3.27	82 %	5097	5.66	57 %	4686	6.16	31 %
BB11018	483303	136164	3.55	89 %	62015	7.79	78 %	48225	10.02	50 %
BB11019	124619	33996	3.67	92 %	17024	7.32	73 %	12297	10.13	51 %
BB11020	46470	13496	3.44	86 %	7615	6.10	61 %	6536	7.11	36 %
BB11021	6007	2462	2.44	61 %	1712	3.51	35 %	1665	3.61	18 %
BB11022	7302	2657	2.75	69 %	1804	4.05	40 %	1763	4.14	21 %
BB11023	75311	21010	3.58	90 %	10524	7.16	72 %	8037	9.37	47 %
BB11024	38436	10516	3.66	91 %	6181	6.22	62 %	5169	7.44	37 %
BB11025	4548	1953	2.33	58 %	1389	3.27	33 %	1540	2.95	15 %
BB11026	120534	35456	3.40	85 %	18258	6.60	66 %	15303	7.88	39 %
BB11027	46404	12685	3.66	91 %	7111	6.53	65 %	5343	8.69	43 %
BB11028	26386	7976	3.31	83 %	4660	5.66	57 %	4032	6.54	33 %
BB11029	2814	1650	1.71	43 %	1263	2.23	22 %	1396	2.02	10 %
BB11030	136827	38393	3.56	89 %	18648	7.34	73 %	13128	10.42	52 %
BB11031	172051	52676	3.27	82 %	22587	7.62	76 %	17359	9.91	50 %
BB11032	80259	22115	3.63	91 %	11461	7.00	70 %	9160	8.76	44 %
BB11033	35460	10363	3.42	86 %	5659	6.27	63 %	4587	7.73	39 %
BB11034	220742	58272	3.79	95 %	28860	7.65	76 %	22558	9.79	49 %
BB11035	3489	2024	1.72	43 %	1520	2.30	23 %	1496	2.33	12 %
BB11036	129164	34526	3.74	94 %	17293	7.47	75 %	12752	10.13	51 %
BB11037	788	738	1.07	27 %	653	1.21	12 %	711	1.11	6 %
BB11038	126057	33670	3.74	94 %	16928	7.45	74 %	13367	9.43	47 %

Tabla C.2: Evaluación del rendimiento paralelo de M2Aling con 5000 evaluaciones (unidades de tiempo en milisegundos). SR: Tiempo de ejecución secuencial. SU: Speed-up (SR dividido para el número de núcleos). Eff: Eficiencia (SU dividida por el número de núcleos) alineando todas las instancias de la familia/grupo RV12 del BAliBASE v3.0.

Problema	SR	4 núcleos			10 núcleos			20 núcleos		
		Tiempo	SU	Eff	Tiempo	SU	Eff	Tiempo	SU	Eff
BB12001	64044	17975	3.56	89 %	9179	6.98	70 %	7462	8.58	43 %
BB12002	5145	2166	2.38	59 %	1558	3.30	33 %	1439	3.58	18 %
BB12003	1727	1278	1.35	34 %	1284	1.35	13 %	1195	1.45	7 %
BB12004	17084	5306	3.22	80 %	3159	5.41	54 %	2823	6.05	30 %
BB12005	13946	4656	3.00	75 %	3125	4.46	45 %	3357	4.15	21 %
BB12006	19241	5789	3.32	83 %	3367	5.71	57 %	2846	6.76	34 %
BB12007	124048	35359	3.51	88 %	19029	6.52	65 %	17581	7.06	35 %
BB12008	68975	18713	3.69	92 %	9573	7.21	72 %	7717	8.94	45 %
BB12009	1017	938	1.08	27 %	868	1.17	12 %	1076	0.95	5 %
BB12010	24275	7639	3.18	79 %	4755	5.11	51 %	4248	5.71	29 %
BB12011	76294	21383	3.57	89 %	10754	7.09	71 %	9523	8.01	40 %
BB12012	18297	6009	3.04	76 %	3630	5.04	50 %	3645	5.02	25 %
BB12013	171250	48228	3.55	89 %	24181	7.08	71 %	19422	8.82	44 %
BB12014	2458	1504	1.63	41 %	1104	2.23	22 %	1252	1.96	10 %
BB12015	11565	4048	2.86	71 %	2358	4.90	49 %	2294	5.04	25 %
BB12016	4773	2108	2.26	57 %	1480	3.23	32 %	1470	3.25	16 %
BB12017	95135	25925	3.67	92 %	13080	7.27	73 %	10575	9.00	45 %
BB12018	164422	43897	3.75	94 %	22333	7.36	74 %	18413	8.93	45 %
BB12019	1181	937	1.26	32 %	756	1.56	16 %	677	1.74	9 %
BB12020	8292	3277	2.53	63 %	2150	3.86	39 %	2055	4.04	20 %
BB12021	2654	1553	1.71	43 %	1239	2.14	21 %	1356	1.96	10 %
BB12022	4365	2101	2.08	52 %	1547	2.82	28 %	1575	2.77	14 %
BB12023	57256	15888	3.60	90 %	8515	6.72	67 %	7541	7.59	38 %
BB12024	17163	5359	3.20	80 %	3133	5.48	55 %	2869	5.98	30 %
BB12025	13893	4403	3.16	79 %	2786	4.99	50 %	2546	5.46	27 %
BB12026	32246	10057	3.21	80 %	4885	6.60	66 %	3888	8.29	41 %
BB12027	27630	8166	3.38	85 %	4650	5.94	59 %	4375	6.32	32 %
BB12028	44209	12949	3.41	85 %	6363	6.95	69 %	5394	8.20	41 %
BB12029	1049	1142	0.92	23 %	926	1.13	11 %	931	1.13	6 %
BB12030	31201	9310	3.35	84 %	5525	5.65	56 %	5273	5.92	30 %
BB12031	86509	23841	3.63	91 %	12612	6.86	69 %	10933	7.91	40 %
BB12032	2687	1465	1.83	46 %	1134	2.37	24 %	1298	2.07	10 %
BB12033	58928	16193	3.64	91 %	8951	6.58	66 %	7265	8.11	41 %
BB12034	15619	4861	3.21	80 %	2996	5.21	52 %	2621	5.96	30 %
BB12035	45691	12842	3.56	89 %	7347	6.22	62 %	5095	8.97	45 %
BB12036	33494	9622	3.48	87 %	5401	6.20	62 %	4663	7.18	36 %
BB12037	72107	21299	3.39	85 %	12062	5.98	60 %	11498	6.27	31 %
BB12038	32171	9490	3.39	85 %	5068	6.35	63 %	4777	6.73	34 %
BB12039	17421	5273	3.30	83 %	3460	5.03	50 %	2903	6.00	30 %
BB12040	1632	1254	1.30	33 %	976	1.67	17 %	1113	1.47	7 %
BB12041	4724	2419	1.95	49 %	1673	2.82	28 %	1884	2.51	13 %
BB12042	58281	16163	3.61	90 %	8459	6.89	69 %	7380	7.90	39 %
BB12043	58237	17386	3.35	84 %	8957	6.50	65 %	6750	8.63	43 %
BB12044	37996	10887	3.49	87 %	6146	6.18	62 %	5518	6.89	34 %

Tabla C.3: Evaluación del rendimiento paralelo de M2Aling con 5000 evaluaciones (unidades de tiempo en milisegundos). SR: Tiempo de ejecución secuencial. SU: Speed-up (SR dividido para el número de núcleos). Eff: Eficiencia (SU dividida por el número de núcleos) alineando todas las instancias de la familia/grupo RV20 del BAliBASE v3.0.

Problema	SR	4 núcleos			10 núcleos			20 núcleos		
		Tiempo	SU	Eff	Tiempo	SU	Eff	Tiempo	SU	Eff
BB20001	2263	1733	1.31	33 %	1385	1.63	16 %	1176	1.92	10 %
BB20002	13228	5218	2.54	63 %	3225	4.10	41 %	2597	5.09	25 %
BB20003	86370	25077	3.44	86 %	13181	6.55	66 %	11303	7.64	38 %
BB20004	57105	17116	3.34	83 %	9656	5.91	59 %	8149	7.01	35 %
BB20005	190809	53635	3.56	89 %	25562	7.46	75 %	17826	10.70	54 %
BB20006	38403	11924	3.22	81 %	6196	6.20	62 %	4679	8.21	41 %
BB20007	226345	63734	3.55	89 %	28991	7.81	78 %	20177	11.22	56 %
BB20008	68356	20687	3.30	83 %	15096	4.53	45 %	11582	5.90	30 %
BB20009	25839	7986	3.24	81 %	4743	5.45	54 %	3813	6.78	34 %
BB20010	129653	37485	3.46	86 %	17715	7.32	73 %	14493	8.95	45 %
BB20011	15199	5530	2.75	69 %	3209	4.74	47 %	3000	5.07	25 %
BB20012	87130	24551	3.55	89 %	12470	6.99	70 %	9270	9.40	47 %
BB20013	56648	16698	3.39	85 %	8578	6.60	66 %	7854	7.21	36 %
BB20014	44276	16125	2.75	69 %	9492	4.66	47 %	9226	4.80	24 %
BB20015	4793	2225	2.15	54 %	1797	2.67	27 %	1734	2.76	14 %
BB20016	30491	9399	3.24	81 %	4958	6.15	61 %	4063	7.50	38 %
BB20017	119313	33960	3.51	88 %	17200	6.94	69 %	13200	9.04	45 %
BB20018	70079	19957	3.51	88 %	10199	6.87	69 %	7978	8.78	44 %
BB20019	135487	37846	3.58	89 %	18551	7.30	73 %	14558	9.31	47 %
BB20020	40002	11557	3.46	87 %	6422	6.23	62 %	5340	7.49	37 %
BB20021	178464	52461	3.40	85 %	25331	7.05	70 %	19582	9.11	46 %
BB20022	62358	21364	2.92	73 %	9377	6.65	67 %	6724	9.27	46 %
BB20023	55260	15555	3.55	89 %	8674	6.37	64 %	6253	8.84	44 %
BB20024	16631	4964	3.35	84 %	3675	4.53	45 %	3139	5.30	26 %
BB20025	51850	15071	3.44	86 %	8632	6.01	60 %	6745	7.69	38 %
BB20026	114317	32784	3.49	87 %	16528	6.92	69 %	13323	8.58	43 %
BB20027	49044	14902	3.29	82 %	8266	5.93	59 %	5994	8.18	41 %
BB20028	85171	24092	3.54	88 %	13251	6.43	64 %	10619	8.02	40 %
BB20029	5335	2513	2.12	53 %	1754	3.04	30 %	1625	3.28	16 %
BB20030	5325	2456	2.17	54 %	1915	2.78	28 %	1867	2.85	14 %
BB20031	41574	13012	3.20	80 %	6809	6.11	61 %	5624	7.39	37 %
BB20032	8063	3361	2.40	60 %	2418	3.33	33 %	2047	3.94	20 %
BB20033	6693	2932	2.28	57 %	2054	3.26	33 %	1889	3.54	18 %
BB20034	93631	27858	3.36	84 %	13591	6.89	69 %	8750	10.70	54 %
BB20035	83777	24494	3.42	86 %	13091	6.40	64 %	11449	7.32	37 %
BB20036	26682	8336	3.20	80 %	5252	5.08	51 %	4145	6.44	32 %
BB20037	226514	64528	3.51	88 %	31301	7.24	72 %	24166	9.37	47 %
BB20038	5139	2499	2.06	51 %	1902	2.70	27 %	1987	2.59	13 %
BB20039	111074	32415	3.43	86 %	17120	6.49	65 %	11396	9.75	49 %
BB20040	18083	6592	2.74	69 %	4171	4.34	43 %	4175	4.33	22 %
BB20041	114964	34539	3.33	83 %	17710	6.49	65 %	15913	7.22	36 %

Tabla C.4: Evaluación del rendimiento paralelo de M2Aling con 5000 evaluaciones (unidades de tiempo en milisegundos). SR: Tiempo de ejecución secuencial, SU: Speed-up (SR dividido para el número de núcleos), Eff: Eficiencia (SU dividida por el número de núcleos) alineando todas las instancias de la familia/grupo RV30 del BAliBASE v3.0.

Problem	**SR**	**4 núcleos**			**10 núcleos**			**20 núcleos**		
		Tiempo	**SU**	**Eff**	**Tiempo**	**SU**	**Eff**	**Tiempo**	**SU**	**Eff**
BB30001	223555	62883	3.56	89 %	32098	6.96	70 %	29289	7.63	38 %
BB30002	69561	20182	3.45	86 %	10242	6.79	68 %	8037	8.66	43 %
BB30003	212740	63093	3.37	84 %	28713	7.41	74 %	21629	9.84	49 %
BB30004	110419	30018	3.68	92 %	15366	7.19	72 %	11019	10.02	50 %
BB30005	244773	63687	3.84	96 %	31363	7.80	78 %	22096	11.08	55 %
BB30006	22460	7254	3.10	77 %	4133	5.43	54 %	3730	6.02	30 %
BB30007	14536	4955	2.93	73 %	3014	4.82	48 %	2587	5.62	28 %
BB30008	83158	23132	3.59	90 %	12286	6.77	68 %	10398	8.00	40 %
BB30009	4713	2104	2.24	56 %	1679	2.81	28 %	1878	2.51	13 %
BB30010	122325	34246	3.57	89 %	18463	6.63	66 %	14487	8.44	42 %
BB30011	71777	21280	3.37	84 %	10499	6.84	68 %	7698	9.32	47 %
BB30012	87207	24538	3.55	89 %	12911	6.75	68 %	9550	9.13	46 %
BB30013	215733	57301	3.76	94 %	30005	7.19	72 %	24971	8.64	43 %
BB30014	24924	7430	3.35	84 %	4519	5.52	55 %	3796	6.57	33 %
BB30015	44793	12671	3.54	88 %	7010	6.39	64 %	6251	7.17	36 %
BB30016	14353	5682	2.53	63 %	3439	4.17	42 %	2754	5.21	26 %
BB30017	43142	12449	3.47	87 %	7294	5.91	59 %	6322	6.82	34 %
BB30018	33956	11154	3.04	76 %	6494	5.23	52 %	5845	5.81	29 %
BB30019	121759	34561	3.52	88 %	18158	6.71	67 %	14060	8.66	43 %
BB30020	27238	8569	3.18	79 %	5082	5.36	54 %	3896	6.99	35 %
BB30021	47343	15237	3.11	78 %	8638	5.48	55 %	6237	7.59	38 %
BB30022	8170	3574	2.29	57 %	2479	3.30	33 %	2351	3.48	17 %
BB30023	81216	24011	3.38	85 %	12097	6.71	67 %	8682	9.35	47 %
BB30024	119734	32832	3.65	91 %	19469	6.15	61 %	16084	7.44	37 %
BB30025	12772	4805	2.66	66 %	3183	4.01	40 %	2839	4.50	22 %
BB30026	117676	34136	3.45	86 %	17414	6.76	68 %	12783	9.21	46 %
BB30027	9813	3546	2.77	69 %	2334	4.20	42 %	1999	4.91	25 %
BB30028	63158	19204	3.29	82 %	9819	6.43	64 %	7325	8.62	43 %
BB30029	11462	6277	1.83	46 %	4163	2.75	28 %	4955	2.31	12 %
BB30030	92661	26724	3.47	87 %	14119	6.56	66 %	11089	8.36	42 %

Tabla C.5: Evaluación del rendimiento paralelo de M2Aling con 5000 evaluaciones (unidades de tiempo en milisegundos). SR: Tiempo de ejecución secuencial. SU: Speed-up (SR dividido para el número de núcleos). Eff: Eficiencia (SU dividida por el número de núcleos) alineando todas las instancias de la familia/grupo RV40 del BAliBASE v3.0.

Problema	SR	4 núcleos			10 núcleos			20 núcleos		
		Tiempo	SU	Eff	Tiempo	SU	Eff	Tiempo	SU	Eff
BB40001	9354	4462	2.10	52 %	2872	3.26	33 %	2866	3.26	16 %
BB40002	61824	17768	3.48	87 %	10109	6.12	61 %	10643	5.81	29 %
BB40003	23121	7127	3.24	81 %	4015	5.76	58 %	3678	6.29	31 %
BB40004	80862	23337	3.46	87 %	12943	6.25	62 %	11577	6.98	35 %
BB40005	20531	6336	3.24	81 %	3986	5.15	52 %	3352	6.13	31 %
BB40006	83299	23416	3.56	89 %	11788	7.07	71 %	8887	9.37	47 %
BB40007	6269	3504	1.79	45 %	2274	2.76	28 %	2116	2.96	15 %
BB40008	103190	28161	3.66	92 %	14234	7.25	72 %	10647	9.69	48 %
BB40009	157125	42589	3.69	92 %	20043	7.84	78 %	14701	10.69	53 %
BB40010	1075	1066	1.01	25 %	928	1.16	12 %	898	1.20	6 %
BB40011	36605	11644	3.14	79 %	7331	4.99	50 %	5721	6.40	32 %
BB40012	42465	12843	3.31	83 %	7059	6.02	60 %	5727	7.41	37 %
BB40013	33569	10804	3.11	78 %	5610	5.98	60 %	4304	7.80	39 %
BB40014	29033	8956	3.24	81 %	5485	5.29	53 %	4716	6.16	31 %
BB40015	44347	15282	2.90	73 %	10724	4.14	41 %	9579	4.63	23 %
BB40016	25273	7930	3.19	80 %	4904	5.15	52 %	4599	5.50	27 %
BB40017	24532	7845	3.13	78 %	4840	5.07	51 %	4454	5.51	28 %
BB40018	8059	3185	2.53	63 %	2032	3.97	40 %	1873	4.30	22 %
BB40019	72579	20243	3.59	90 %	10179	7.13	71 %	8902	8.15	41 %
BB40020	4280	2280	1.88	47 %	1777	2.41	24 %	1634	2.62	13 %
BB40021	12987	4396	2.95	74 %	3030	4.29	43 %	3012	4.31	22 %
BB40022	13313	4541	2.93	73 %	3324	4.01	40 %	2907	4.58	23 %
BB40023	164289	44646	3.68	92 %	23737	6.92	69 %	19317	8.50	43 %
BB40024	157447	43787	3.60	90 %	21491	7.33	73 %	16391	9.61	48 %
BB40025	31789	9603	3.31	83 %	5490	5.79	58 %	4921	6.46	32 %
BB40026	281226	81690	3.44	86 %	40970	6.86	69 %	30248	9.30	46 %
BB40027	141353	40983	3.45	86 %	20836	6.78	68 %	22002	6.42	32 %
BB40028	32583	9494	3.43	86 %	5207	6.26	63 %	4061	8.02	40 %
BB40029	51487	14888	3.46	86 %	7995	6.44	64 %	6657	7.73	39 %
BB40030	50071	15224	3.29	82 %	8077	6.20	62 %	6702	7.47	37 %
BB40031	5049	2792	1.81	45 %	2321	2.18	22 %	1806	2.80	14 %
BB40032	32198	9339	3.45	86 %	5134	6.27	63 %	4693	6.86	34 %
BB40033	43351	13824	3.14	78 %	6752	6.42	64 %	5693	7.61	38 %
BB40034	125073	36334	3.44	86 %	17867	7.00	70 %	14766	8.47	42 %
BB40035	36872	11403	3.23	81 %	6526	5.65	57 %	4622	7.98	40 %
BB40036	72728	20608	3.53	88 %	10656	6.83	68 %	8837	8.23	41 %
BB40037	23738	7853	3.02	76 %	4453	5.33	53 %	4163	5.70	29 %
BB40038	12718	4687	2.71	68 %	2905	4.38	44 %	3025	4.20	21 %
BB40039	151404	47632	3.18	79 %	25180	6.01	60 %	22159	6.83	34 %
BB40040	40480	11538	3.51	88 %	6157	6.57	66 %	4856	8.34	42 %
BB40041	278227	81142	3.43	86 %	41084	6.77	68 %	35434	7.85	39 %
BB40042	84450	24154	3.50	87 %	12174	6.94	69 %	8842	9.55	48 %
BB40043	53213	14765	3.60	90 %	8377	6.35	64 %	6329	8.41	42 %
BB40044	23701	7834	3.03	76 %	4649	5.10	51 %	4063	5.83	29 %
BB40045	5261	2628	2.00	50 %	1714	3.07	31 %	1805	2.91	15 %
BB40046	127106	36044	3.53	88 %	17227	7.38	74 %	13955	9.11	46 %
BB40047	231731	68096	3.40	85 %	62208	3.73	37 %	45366	5.11	26 %
BB40048	3275	1811	1.81	45 %	1655	1.98	20 %	1335	2.45	12 %
BB40049	137292	40475	3.39	85 %	18575	7.39	74 %	20105	6.83	34 %

Tabla C.6: Evaluación del rendimiento paralelo de M2Aling con 5000 evaluaciones (unidades de tiempo en milisegundos). SR: Tiempo de ejecución secuencial. SU: Speed-up (SR dividido para el número de núcleos). Eff: Eficiencia (SU dividida por el número de núcleos) alineando todas las instancias de la familia/grupo RV50 del BAliBASE v3.0.

Problema	**SR**	**4 núcleos**			**10 núcleos**			**20 núcleos**		
		Tiempo	**SU**	**Eff**	**Tiempo**	**SU**	**Eff**	**Tiempo**	**SU**	**Eff**
BB50001	69877	20116	3.47	87 %	10273	6.80	68 %	7863	8.89	44 %
BB50002	310248	84531	3.67	92 %	40874	7.59	76 %	30950	10.02	50 %
BB50003	83185	24528	3.39	85 %	12184	6.83	68 %	10532	7.90	39 %
BB50004	76135	21546	3.53	88 %	10469	7.27	73 %	8614	8.84	44 %
BB50005	165264	43586	3.79	95 %	23033	7.18	72 %	18785	8.80	44 %
BB50006	454952	125185	3.63	91 %	62743	7.25	73 %	48356	9.41	47 %
BB50007	63873	18623	3.43	86 %	9430	6.77	68 %	7265	8.79	44 %
BB50008	48099	13750	3.50	87 %	7666	6.27	63 %	6186	7.78	39 %
BB50009	78311	22041	3.55	89 %	11407	6.87	69 %	8160	9.60	48 %
BB50010	45708	13261	3.45	86 %	6856	6.67	67 %	5640	8.10	41 %
BB50011	44373	13349	3.32	83 %	7465	5.94	59 %	6314	7.03	35 %
BB50012	121702	38039	3.20	80 %	17554	6.93	69 %	15934	7.64	38 %
BB50013	23355	7132	3.27	82 %	4310	5.42	54 %	3992	5.85	29 %
BB50014	140839	39044	3.61	90 %	20065	7.02	70 %	16471	8.55	43 %
BB50015	120110	35252	3.41	85 %	17712	6.78	68 %	14281	8.41	42 %
BB50016	21111	6829	3.09	77 %	3709	5.69	57 %	3473	6.08	30 %

C.2. Comparativa M2Align con otras técnicas MSA

Tabla C.7: Resultados de M2Align evaluando algunas instancias del BAliBASE v3.0 versus la versión original de MOSAStrE y otras técnicas MSA bien conocidas y clásicas del estado del arte (Las columnas indican los objetivos a optimizar STR: STRIKE, TC y NG: Non-Gaps)

Método	BB11001			BB11009			BB11011			BB11013			BB12002		
	STR	**TC**	**NG**	**STR**	**TC**	**NG**	**STR**	**TC**	**NG**	**STR**	**TC**	**NG**	**STR**	**TC**	**NG**
ClustalW	2.45	1.04	89.84	0.30	0.29	63.34	0.63	0.79	79.45	0.84	0.00	69.51	1.22	3.75	86.67
Muscle	2.60	1.04	89.84	0.56	0.80	57.60	0.60	0.38	77.31	0.78	0.00	59.67	1.31	3.72	85.95
Kalign	2.44	3.03	87.12	1.01	1.02	36.80	1.17	1.06	71.28	0.95	0.00	66.92	1.27	3.64	84.21
RetAlign	2.07	3.60	77.70	0.58	1.69	52.05	1.20	1.07	71.79	0.60	0.76	54.66	1.33	4.08	84.90
Tcoffee	2.33	1.89	81.37	0.64	0.00	47.26	0.59	0.62	61.85	0.68	0.00	57.28	1.31	3.29	85.60
ProbCons	2.51	1.04	89.84	0.96	1.20	37.18	0.86	0.91	61.28	0.57	0.00	56.38	1.24	3.66	84.55
Mafft	2.36	1.01	87.12	0.62	0.23	48.65	0.50	0.29	58.94	0.24	0.00	56.38	1.28	3.27	84.90
FSA	2.19	0.80	69.00	0.53	0.31	33.23	0.12	0.00	29.47	0.07	0.00	25.21	1.28	3.21	83.53
BAliBASE Ref	2.53	1.04	89.84	1.36	0.86	36.99	1.26	1.73	69.55	0.65	0.00	62.26	1.29	3.29	85.60
MO-SAStrE	2.82	1.98	85.40	1.16	0.87	37.37	1.25	0.35	70.53	1.25	0.80	57.28	1.23	1.24	85.95
M2Align	**3.21**	**6.80**	**92.74**	**1.51**	**2.79**	**63.53**	**1.76**	**2.25**	**82.72**	**2.00**	**0.99**	**70.89**	**1.60**	**4.86**	**89.27**

Tabla C.8: Resultados de M2Align evaluando algunas instancias del BAliBASE v3.0 versus la versión original de MOSAStrE y otras técnicas MSA bien conocidas y clásicas del estado del arte (Las columnas indican los objetivos a optimizar STR: STRIKE, TC y NG: Non-Gaps).

Método	BB12004			BB12010			BB12015			BB30009		
	STR	**TC**	**NG**	**STR**	**TC**	**NG**	**STR**	**TC**	**NG**	**STR**	**TC**	**NG**
ClustalW	2.38	5.47	79.94	2.00	4.73	78.44	1.37	0.65	41.40	1.07	0.00	**61.45**
Muscle	2.50	5.43	79.42	2.23	5.52	76.88	2.27	0.75	36.04	1.77	0.00	62.34
Kalign	2.43	5.15	75.33	2.18	5.49	76.55	2.17	0.71	34.24	1.71	0.00	51.95
RetAlign	2.56	5.11	74.65	2.32	6.06	75.39	2.20	0.76	36.31	0.52	0.00	50.28
Tcoffee	2.52	5.41	79.17	2.22	5.65	75.71	2.06	0.73	35.12	1.75	0.00	48.02
ProbCons	2.50	5.26	76.97	2.18	5.63	75.39	2.17	0.73	35.18	1.37	0.00	31.75
Mafft	2.52	5.36	78.42	2.31	5.53	77.05	2.23	0.75	36.04	1.41	0.00	53.24
FSA	2.49	4.96	72.48	2.06	4.71	68.29	2.11	0.61	29.24	1.22	0.00	16.71
BAliBASE Ref	2.49	5.45	79.68	2.23	4.89	77.40	2.22	0.77	36.80	1.14	0.00	61.67
MO-SAStrE	2.63	5.06	78.67	2.50	4.81	76.21	2.32	0.75	36.04	2.12	0.00	49.84
M2Align	**2.91**	**5.48**	**84.27**	**2.64**	**6.14**	**83.32**	**2.50**	**1.00**	**44.67**	**2.33**	**0.00**	61.01

Apéndice D

Lista de parámetros de MO-Phylogenetics

Parámetro Abreviado	Valor por Defecto	Descripción
Parámetros de las Metaheurísticas		
experimentid	1	Experiment ID, los archivos resultados serán nombrados con este ID
DATAPATH*	data	Nombre del directorio con los datos
algorithm	NSGAII	Nombre del algoritmo a usar, disponibles: NSGAII, SMSEMOA, MOEAD y PAES
populationsize	100	Tamaño de la población
maxevaluations	2000	Número de evaluaciones
offset	100	SMSEMOA Parámetro
bisections	5	Número de bisecciones, Parámetro PAES
archivesize	100	Tamaño del archivo, Parámetro PAES
intervalupdateparameters	500	Intervalo de evaluaciones para cada nueva optimización de las longitudes de rama y los parámetros del modelo de sustitución
printtrace	false	Imprime los tiempos transcurridos y los scores obtenidos durante las estrategias de optimización aplicadas
printbestscores	false	Imprime los mejores scores encontrados durante la ejecución del algoritmo.
Operadores Genéticos		
selection.method	binarytournament	Operadores de Selección disponibles: *binarytournament* y *randomselection* (solo para SMSEMOA)
crossover.method	PGD	Operador de cruce, disponible solo el operador PDG (Prune-Delete-Graf)

crossover.probability	0.8	Probabilidad de ejecución del operador de cruce
crossover.offspringsize	2	Number of descendents generated by the crossover
mutation.method	NNI	Operadores de mutación topológica disponibles: NNI, SPR y TRB
mutation.probability	0.2	Probabilidad de ejecución del operador de mutación
Parámetros Filogenéticos		
alphabet	DNA	Valores *Protein* y *DNA*, disponibles
input.sequence.file*	**REQUERIDO**	El fichero con el alineamiento múltiple de secuencias.
input.sequence.format	Phylip(split=spaces, type=extended, order=interleaved)	El formato del MSA. Opciones: Order=sequential, interleaved y type=extended o classic. Ejemplo de MSA en formato Fasta: Fasta(charsbyline=100, checksequencenames=true, extended=true,strictsequencenames=true)
input.sequence.print	false	Imprime las secuencias leídas
partitionmodelfilepll*	**REQUERIDO**	Nombre del fichero con información del Modelo Evolutivo, define esquema de particionamiento del MSA y la asignación de un modelo de sustitución para cada partición definida. Basado en el formato RAxML
model	GTR · G	Modelo Evolutivo, solo GTR · G
init.population	stepwise	Método para generar o crear los árboles filogenéticos iniciales, los valores son: *user*, *random* o *stepwise*
init.population.tree.file	userinittree.txt	Nombre del fichero con el conjunto de árboles filogenéticos definidos por el usuario para la generación de la población inicial. Si los árboles filogenéticos iniciales son definidos por el usuario, este parámetro es obligatorio
init.population.tree.format	Newick	formato de los árboles filogenéticos: *Newick*
Parámetros del Modelo Evolutivo		
model	GTR · G	Modelo Evolutivo, sólo GTR · G
Frecuencias de bases, aplica sólo para el modelo de sustitución GTR (DNA)		
model.frequences	user	Las frecuencias de las bases puede ser definida por el usuario o estimada empíricamente por el software: *none* o *user*
model.piA, model.piC, model.piG, model.piT	0.25 0.25 0.25 0.25	Valores de ΠA, ΠC, ΠG, ΠT, Todos los valores deben sumar 1
Parámetros de las tasas relativas, aplcia solo para el modelo de sustitución GTR (DNA)		
model.AC,model.AG, model.AT, model.CG , model.CT, model.GT	1 1 1 1 1 1	AC, AG, AT, CG, CT, GT siempre debe ser 1

Distribución Gamma		
rate_distribution	Gamma	Gamma disponible
rate_distribution.ncat	4	Número de categorías, 4 por defecto
rate_distribution.alpha	0.5	Valor α
Optimización Filogenética		
Estrategias de alto nivel para la exploración del espacio de búsqueda:		
opt.method	h2	*h1* o *h2*
opt.method.perc	0.5	Si la estrategia h2 es seleccionada, representa al porcentaje de aplicar una de las siguientes técnicas *pllRearrangeSearch* o *PPN*
Parámetros de la técnica PPN		
opt.ppn.numiterations	500	Número de iteraciones de la técnica PPN
opt.ppn.maxsprbestmoves	20	Máximo número de mejores movimientos a ser aplicados en la técnica PPN
Parámetros de la técnica pllRearrangeSearch		
opt.pll.percnodes	0.2	Porcentaje de nodos a ser usados como "nodo raíz" por la función
opt.pll.mintranv	1	Define el tamaño del anillo en el que los movimientos topológicos tienen lugar
opt.pll.maxtranv	20	
opt.pll.newton3sprbranch	true	Optimización de las longitudes de las ramas afectadas por los movimiento topológicos
Optimización de las longitudes de las ramas		
Optimización de los árboles de la población inicial.		
bl_opt.starting	false	true or false
bl_opt.starting.method	newton	Los métodos numéricos disponibles son: *newton* y *gradient*
bl_opt.starting.maxiteval	1000	Máximo número de iteraciones
bl_opt.starting.tolerance	0.01	Tolerencia
Optimización de los árboles generados durante la exploración		
bl_opt	false	*true* o *false*
bl_opt.method	newton	Los métodos numéricos disponibles son: *newton* y *gradient*
bl_opt.maxitevaluation	1000	Máximo número de iteraciones
bl_opt.tolerance	0.01	Tolerencia
Optimización a árboles obtenidos al final del algoritmo.		
bl_opt.final	false	*true* o *false*
bl_opt.final.method	newton	Los métodos numéricos disponibles son: *newton* y *gradient*
bl_opt.final.maxitevaluation	1000	Máximo número de iteraciones
bl_opt.final.tolerance	0.01	Tolerencia
Optimización del modelo evolutivo		
model_opt	false	*true* o *false*
model_opt.method	brent	Los métodos numéricos disponibles son: *brent* y *bfgs*
model_opt.maxitevaluation	1000	Máximo número de iteraciones

model_opt.tolerance	0.01	Tolerencia
Optimización de la Tasa de Distribución		
ratedistribution_opt	false	*true* o *false*
ratedistribution_opt.method	Los métodos numéricos disponibles son: *brent* y *bfgs*	

Tabla D.1: Lista de parámetros de MO-Phylogenetics, el parámetro marcado con * es obligatorio. Contiene un nombre corto, un valor predeterminado para el parámetro opcional y una breve descripción.

Apéndice E

Lista de parámetros de MORPHY

Parámetro abreviado	Valor por defecto	Descripción
	Parámetross de la Metaheurística	
experimentid	1	Experiment ID, los archivos resultados serán nombrados con este ID
DATAPATH*	data	Nombre del directorio con los datos
populationsize	100	Tamaño de la población
maxevaluations	2500	Número de evaluaciones
threshold	0.2	Valor umbral del algoritmo, porcentaje mínimo de nuevas soluciones permitidas antes de aplicar una búsqueda local exhaustiva
intervalupdateparameters	500	Intervalo de evaluaciones para cada nueva optimización de las longitudes de rama y los parámetros del modelo de sustitución
printtrace	false	Imprime los tiempos transcurridos y los scores obtenidos durante las estrategias de optimización aplicadas
printbestscores	false	Imprime los mejores scores encontrados durante la ejecución del algoritmo
NumPreliminarResults	0	Imprime Resultados Preliminares cada x número de evaluaciones del algoritmo. 0 = ninguno
	Operadores Genético de Mutación	
mutation.method	NNI	Operadores de mutación topológica disponibles: NNI, SPR y TRB
mutation.probability	0.2	Probabilidad de ejecución del operador de mutación
	Parámetros Filogenéticos	
alphabet	Protein	Valores *Protein* y *DNA*, disponibles
sequencefile*	**REQUERIDO**	El fichero con el alineamiento múltiple de secuencias.

input.sequence.format	Phylip(split=spaces, type=extended, order=interleaved)	El formato del MSA. Opciones: Order=sequential, interleaved y type=extended o classic. Ejemplo de MSA en formato Fasta: Fasta(charsbyline=100, checksequencenames=true, extended=true,strictsequencenames=true)
input.sequence.print	false	Imprime las secuencias leídas
init.population	stepwise	Método para generar o crear los árboles filogenéticos iniciales, los valores son: *user*, *random* o *stepwise*
init.population.tree.file	userinittree.txt	Nombre del fichero con el conjunto de árboles filogenéticos definidos por el usuario para la generación de la población inicial. Si los árboles filogenéticos iniciales son definidos por el usuario, este parámetro es obligatorio
init.population.tree.format	Newick	formato de los árboles filogenéticos: *Newick*
bootstrapSize	101	Número de los árboles filogenéticos a obtener del fichero init.population.tree.file
Parámetros del Modelo Evolutivo		
partitionmodelfile*	**REQUERIDO**	Nombre del fichero con información del Modelo Evolutivo, define esquema de particionamiento del MSA y la asignación de un modelo de sustitución para cada partición definida. Basado en el formato RAxML
model	GTR · G	Modelo Evolutivo, solor GTR · G
rateheterogeneity_model	GAMMA	Los modelos de tasa heterogénea entre sitios disponibles están, GAMMA con un valor por defecto de 4 categorías y CAT con N categorías
Frecuencias de bases, aplica sólo para el modelo de sustitución GTR (DNA)		
model.frequences	none	Las frecuencias de las bases puede ser definida por el usuario o estimada empíricamente por el software: *none* o *user*
model.piA, model.piC, model.piG, model.piT	0.25 0.25 0.25 0.25	Valores de IIA, IIC, IIG, IIT, Todos los valores deben sumar 1
Parámetros de las tasas relativas, aplcia solo para el modelo de sustitución GTR (DNA)		
model.AC,model.AG, model.AT, model.CG , model.CT, model.GT	1 1 1 1 1 1	AC, AG, AT, CG, CT, GT siempre debe ser 1
Distribución Gamma		
rate_distribution	Gamma	Gamma y CAT disponible
rate_distribution.ncat	4	Número de categorías, 4 por defecto
rate_distribution.alpha	0.5	Valor α
Optimización Filogenética		

Estrategias de alto nivel para la exploración del espacio de búsqueda:		
opt.method	h2	*h1* o *h2*
opt.method.perc	0.5	Si la estrategia h2 es seleccionada, representa al porcentaje de aplicar una de las siguientes técnicas *pllRearrangeSearch* o *PPN*
Parámetros de la técnica PPN		
opt.ppn.numiterations	500	Número de iteraciones de la técnica PPN
opt.ppn.maxsprbestmoves	20	Máximo número de mejores movimientos a ser aplicados en la técnica PPN
Parámetros de la técnica pllRearrangeSearch		
opt.pll.percnodes	0.2	Porcentaje de nodos a ser usados como "nodo raíz" por la función
opt.pll.mintranv	1	Define el tamaño del anillo en el que los movimientos topológicos tienen lugar
opt.pll.maxtranv	20	
opt.pll.newton3sprbranch	true	Optimización de las longitudes de las ramas afectadas por los movimiento topológicos
Optimización de las longitudes de las ramas - método Newton-Rapshon		
Optimización de los árboles de la población inicial		
bl_opt.starting	false	*true* o *false*
bl_opt.starting.maxiteval	1000	Máximo número de iteraciones
Optimización de los árboles generados durante la exploración		
bl_opt	false	*true* o *false*
bl_opt.maxitevaluation	1000	Máximo número de iteraciones
Optimización a árboles obtenidos al final del algoritmo.		
bl_opt.final	false	*true* o *false*
bl_opt.final.maxitevaluation	1000	Máximo número de iteraciones
Optimización del modelo evolutivo		
model_opt	false	*true* o *false*
model_opt.tolerance	0.01	Tolerencia
Árbol consenso del conjunto final de soluciones no-dominadas (árboles filogenéticos)		
consensus	false	Realizar o NO el Consenso, *true* o *false*
threshold	0.5	Score mínimo aceptable= número de ocurrencias de una bipartición/número de árboles (0.<=threshold<=1.). 0 generará un árbol consensuado totalmente, 0.5 corresponde a la regla mayoritaria y 1 a un consenso estricto.

Tabla E.1: Lista de parámetros de MORPHY, el parámetro marcado con * es obligatorio. Contiene un nombre corto, un valor predeterminado para el parámetro opcional y una breve descripción.

Descubre tu próxima lectura

Si quieres formar parte de nuestra comunidad,
regístrate en **https://www.grupocompas.org/suscribirse**
y recibirás recomendaciones y capacitación

compAs

Grupo de capacitación e investigación pedagógica

@grupocompas.ec
compasacademico@icloud.com

compAs
Grupo de capacitación e investigación pedagógica

@grupocompas.ec
compasacademico@icloud.com

ISBN: 978-9942-33-198-4

@grupocompas.ec
compasacademico@icloud.com

compAs
Grupo de capacitación e investigación pedagógica